A Comprehensive Method for Multilingual Video Text Detection, Localization, and Extraction

Michael R. Lyu, *Fellow, IEEE*, Jiqiang Song, *Member, IEEE*, and Min Cai

***Abstract*—Text in video is a very compact and accurate clue for video indexing and summarization. Most video text detection and extraction methods hold assumptions on text color, background contrast, and font style. Moreover, few methods can handle multilingual text well since different languages may have quite different appearances. This paper performs a detailed analysis of multilingual text characteristics, including English and Chinese. Based on the analysis, we propose a comprehensive, efficient video text detection, localization, and extraction method, which emphasizes the multilingual capability over the whole processing. The proposed method is also robust to various background complexities and text appearances. The text detection is carried out by edge detection, local thresholding, and hysteresis edge recovery. The coarse-to-fine localization scheme is then performed to identify text regions accurately. The text extraction consists of adaptive thresholding, dam point labeling, and inward filling. Experimental results on a large number of video images and comparisons with other methods are reported in detail.**

***Index Terms*—Extraction, localization, multilingual texts, video text detection.**

I. Introduction

NOWADAYS, video is the most popular media type delivered via TV broadcasting, Internet, and wireless network. To enable users to quickly locate their interested content in an enormous quantity of video data, many research efforts [1], [2] have been put into video indexing and summarization. Textual, visual, and audio information is most frequently used for this purpose. Among them, text in video, especially the superimposed text, is the most reliable clue for three reasons: 1) it is closely related to the current content of video; 2) it has distinctive visual characteristics; and 3) the state-of-the-art optical character recognition (OCR) techniques are far more robust than the existing speech analysis techniques and visual object analysis techniques. Therefore, almost all video indexing research work begins with video text recognition.

Video text recognition is generally divided into four steps: detection, localization, extraction, and recognition. The detection step roughly classifies text regions and nontext regions. The localization step determines the accurate boundaries of text strings. The extraction step filters out background pixels in the text strings, so that only the text pixels are left for the recognition. Since the above three steps generate a binary text image, the recognition step can be done by commercial document OCR software. Therefore, similar to other papers on text processing in binary documents, only the first three steps are discussed in this paper. However, the text detection, localization, and extraction in video frames present many more difficulties than the text segmentation in binary documents, due to complex background, unknown text color, degraded text quality caused by the lossy compression, and different language characteristics.

Although many methods have been proposed for this task in the last decade [3]–[23], few of them address the multilingual problem. Some only claim the multilingual ability in the detection step. However, the language differences in the detection step are about the font size and the stroke density, which are not as critical, whereas the structural differences in the extraction step are very critical since they affect the binary results to be further processed by OCR. The automatic multilingual adaptivity of video text processing is very important to digital video libraries emphasizing multilingual capabilities [3]. Therefore, this paper proposes a new method that emphasizes the multilingual attributes throughout the detection, localization, and extraction.

The rest of this paper is organized as follows. Section II reviews the related work. Section III analyzes the multilingual text characteristics in video. Then, our new method for detection, localization, and extraction is described in Section IV. Experimental results are presented and discussed in Section V. Finally, in Section VI, we draw conclusions.

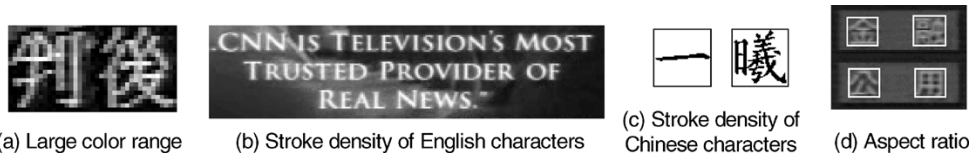
II. Related Work

Video text detection methods can be classified into two classes. The first class treats text as a type of texture. These methods usually divide a whole image into blocks. They first use various approaches, e.g., Gabor filter [6], spatial variance [7], or wavelet transform [8], to calculate the texture features of blocks. Then they employ proper classifiers, e.g., neural network [8] or a support vector machine [9], to classify text blocks and nontext blocks. Some methods [10], [11] assume that the text strokes have a certain contrast against the background. Therefore, those areas with dense edges are detected as text regions. Textures can also be detected in the MPEG-compressed domain. Gargi *et al.* [12] proposed to use the number of intracoded blocks in each P- and B- frame to detect the appearance of captions. However, the texts appearing in I-frames are not handled. Zhong *et al.* [13] and Lim *et al.* [14] proposed to utilize DCT coefficients to detect text regions. The second class [15]–[17] assumes that a text string contains a uniform color. Thus, at first they perform color reduction and segmentation in some selected color channel ([16] only used the red channel)

Manuscript received December 2, 2002; revised May 9, 2003. This work was supported by the Hong Kong Special Administrative Region through the Hong Kong Research Grants Council under Project CUHK4182/03E and through the Innovation and Technology Fund under Project ITS/29/00.

The authors are with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Shatin, N.T., Hong Kong.

Digital Object Identifier 10.1109/TCSVT.2004.841653



(a) Large color range (b) Stroke density of English characters (c) Stroke density of Chinese characters (d) Aspect ratio

Fig. 1. Characteristics of multilingual texts.

or color space (*Lab* space was chosen in [17]), and then they perform connected-component analyses to detect text regions.

Generally, there are two paradigms for the text localization: bottom-up paradigm and top-down paradigm. The bottom-up paradigm groups small text regions into entire text strings based on some heuristic rules and geometric constraints. Region growing is a conventional technique for this purpose. Wu *et al.* [7] proposed a nonregion-based approach, which uses edges in the text regions to form strokes, and then strokes are aggregated to form chips (text strings). The top-down paradigm is based on splitting the image regions (the initial region is the whole image) alternately in horizontal and vertical directions based on the texture, color, or edge distribution. Gao and Tang [18] proposed to use horizontal and vertical projections of edges to localize text strings; however, it can only handle captions and cannot deal with complex text layouts. Some methods combines these two paradigms. For example, Wernicke and Lienhart [19] performed multiple passes of horizontal and vertical projections within initial bounding boxes to cope with complex text layouts; however, they obtained initial bounding boxes by region growing. Therefore, if the initial bounding boxes are oversegmented, which happens for Chinese texts, there will be no chance to recover the correct text regions. In our previous research [11], we proposed a coarse-to-fine localization scheme to handle both multilingual texts and complex text layouts.

The text extraction methods fall into two groups. One group includes color-based methods, and the other includes stroke-based methods. The former holds the assumption that the text pixels are of different color from the background pixels, so that they can be segmented by thresholding. Besides the validity of the assumption, another difficulty in this approach is the color polarity of text, i.e., light or dark, must be determined. Antani *et al.* [21] proposed a color-based method that first generates two segmented results in both polarities for each text string and then selects the one with a higher score on text-like characteristics as the final result. The stroke-based methods, on the other hand, employ some filters to output only those pixels likely on the strokes to the final results, such as the asymmetric filter [22], the four-direction character extraction filter [10], and the topographical feature mask [23]. These filters are intended to enhance the stripe (i.e., stroke-like) shapes and to suppress others; however, the intersection of strokes may also be suppressed due to the lack of stripe shape.

III. Multilingual Text Characteristics

Based on the review of previous methods, we sum up eight kinds of text characteristics that are frequently used in video text detection, localization, and extraction: contrast, color, orientation, stationary location, stroke density, font size, aspect ratio, and stroke statistics. From the point of view of multilingual text processing, we classify them into language-independent characteristics and language-dependent characteristics. In this section, we discuss how to define the constraints of these characteristics for our new multilingual method.

A. Language-Independent Characteristics

1) Contrast: Some methods assume that a text has high contrast against its background; however, this cannot be guaranteed due to complex background in video scenes. Actually, against a clear background, even a low-contrast text can be easily recognized, while in a complex background, only the high-contrast text can be discerned. Thus, we set an adaptive contrast threshold according to the background complexity.

2) Color: We believe that most text strings originally have uniform color, but the lossy compression causes the color bleeding effect at the edge of text. Thus, the color of a text string is in a range whose breath is related to the local background color. Fig. 1(a) shows a text string whose color range is quite large. Therefore, we would rather assume the text color is either lighter or darker than the background color than hold the uniform color assumption.

3) Orientation: No matter whether it is an artificial text or a scene text, if it is closely related to the video content, it must have been presented in a way easy to read, i.e., with a certain contrast, a reasonable size, and an approximately horizontal orientation. Thus, the constraint of horizontal text orientation is adequate for the video indexing purpose.

4) Stationary Location: Most texts of interest are stationary over time. Scrolling texts are only used in credit titles and special effects. To keep the efficiency of video indexing system, we only handle stationary texts.

B. Language-Dependent Characteristics

According to the linguistic classification, English, French, and Spanish belong to alphabetic literal, whereas Chinese, Japanese, and Korean belong to ideograph. Their differences in the following four aspects affect the video text processing.

1) Stroke Density: The stroke density of an English text string is roughly uniform [Fig. 1(b)], but that of a Chinese text string is not. Because the stroke number of a Chinese character may vary from 1 to more than 20 and each character occupies the same-sized block space, the stroke density varies significantly [Fig. 1(c)]. This poses an obstacle to employ the region-growing techniques. However, the stroke density of a whole string is still high.

2) Font Size: To display all strokes clearly, an English string requires at least 8-pixel-high font size, while a Chinese string requires at least 20-pixel-high font size due to the large stroke

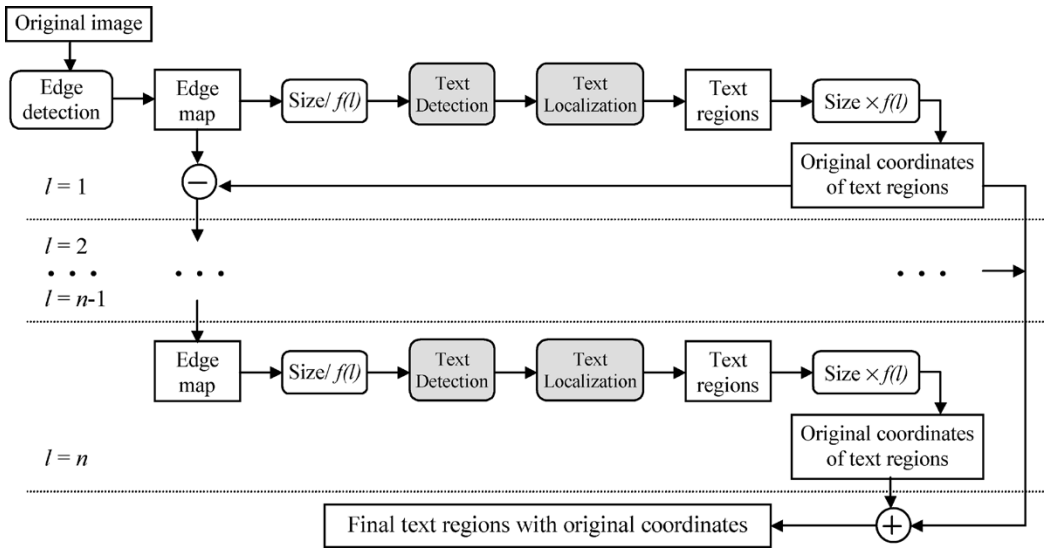


Fig. 2. Sequential multiresolution paradigm.

number. This is the basis of size constraint to discard regions that are too small.

3) Aspect Ratio: In English, characters (letters) must be combined to form meaningful words. In Chinese, each individual character has its own meaning, and it can also be combined with other characters to express more meaning in terms of phrases. Sometimes, a text string may contain some characters with long spaces, but they are not separable [Fig. 1(d)]. Thus, the aspect ratio constraint should be adapted to Chinese characters.

4) Stroke Statistics: Since English characters consist of mainly vertical strokes, many methods [10], [13] only consider vertical edges to detect or extract English texts. However, Chinese characters consist of four directional strokes. Moreover, those filters to extract strokes do enhance the stripe-like strokes but suppress the intersections of strokes. Since English characters do not contain many intersections, this will not affect the character recognition too much. However, Chinese characters contain many intersections and these intersections are important clues for character recognition. Therefore, a multilingual method has to handle both strokes of all directions and intersections well.

IV. Comprehensive Multilingual Method

A. Overview

Based on the analysis of multilingual text characteristics, we can regard multilingual texts as a special type of symbol that has the union of the above characteristics. The first problem we face is the large font-size range of multilingual texts. The multiresolution technique is a good solution; however, previous methods [7]–[9] all fall into a parallel multiresolution paradigm. This is an approach that first scales the original image into different resolution levels, then performs the same operations in each level, and finally merges the results of all levels into one result in the original resolution. Since a text string may appear acceptable in more than one resolution level, the parallel paradigm spends unnecessary time on detecting the same text string in many levels and on merging overlapping text regions into one. To remove this redundancy, we propose a sequential multiresolution paradigm (Fig. 2).

A video segment is sampled at two frames per second, and then each frame is converted into a 256-level grayscale image, which becomes the input "Original image" in Fig. 2. Since the contrast of text is more reliable than the color uniformity, we mainly depend on the edge information to detect texts. The Sobel detector [24] is chosen to detect edges for two reasons: 1) it is isotropic, so that the strokes of all directions are equally treated and 2) it generates double edges, which make text areas much denser than nontext areas. The Sobel detector consists of four directional gradient masks: horizontal (S_H), vertical (S_V), left diagonal (S_{LD}), and right diagonal (S_{RD}). Since the traditional simplified Sobel detector [without the second term in (1)] tends to suppress corners, whereas text strokes often form perpendicular corners, we calculate the edge strength (S) using (1) to highlight both edges and corners as follows:

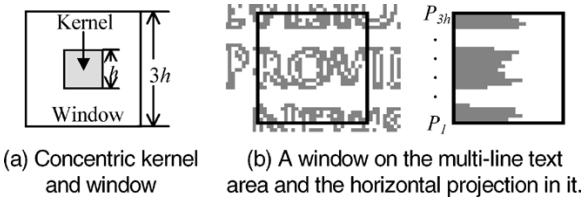
$$S = \mathrm{MAX}(|S_H|, |S_V|, |S_{LD}, |S_{RD}|) + k \times |S_{\perp-\mathrm{MAX}}| \quad (1)$$

where $\perp -\mathrm{MAX}$ is normal to the maximum gradient direction and k is set to be 0.5 in favor of perpendicular corners. The edge map is formed by

$$\mathrm{EdgeMap}(x,y) = \begin{cases} S(x,y), & S(x,y) > T_e \\ 0, & S(x,y) \le T_e \end{cases} \quad (2)$$

where T_e is set to be 25, which is obtained empirically according to the recognizable edge strength.

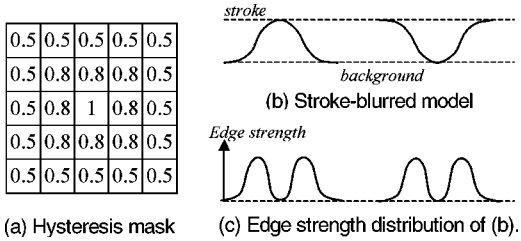
At each level l ($l = 1 \ldots n$), the current edge map is first scaled down by the factor $f(l)$ to generate the working image with desired resolution, then the text detection and localization algorithms are performed to localize text regions, and finally the text regions are scaled up by the same $f(l)$ to get their original image coordinates. Note that these text regions have two functions: one is to be added to the final text regions, and the other is to be used as masks to erase the edges inside these regions from the current edge map. The modified edge map then acts as the input of the next level; therefore, the detected text regions



(a) Concentric kernel and window

(b) A window on the multi-line text area and the horizontal projection in it.

Fig. 3. Local thresholding in a window.



(a) Hysteresis mask

(c) Edge strength distribution of (b).

Fig. 4. Hysteresis edge recovery.

will not be detected again. After passing all levels, the final text regions are definitely not overlapping. The selection of $f(l)$ and n depends on how large the maximum text size to be detected is. For the video-indexing purpose, we are not interested in too large texts; therefore, we set $f(l) = l$ and $n = 3$, and, in each level, the text detection and localization algorithms handle text with 8 to 24 pixels high. Thus, this configuration enables the texts ranging from 8 to 72 pixels in height to be detected.

Before the final text regions in a frame are fed to the text extraction algorithm, a multiframe verification process takes place to eliminate those regions that are transitory or have already been detected in the previous frames. Therefore, the text extraction only processes the first-emerging text regions to produce the binary OCR-ready text images. The following subsections describe the text detection, localization, verification and extraction algorithms in detail.

B. Text Detection

The aim of text detection is to classify text areas and nontext areas, i.e., to highlight only text areas and suppress other areas. Since a global threshold cannot fit different text contrast levels, a local thresholding is necessary. On a clear background, the threshold should be relatively low to let both low-contrast texts and high-contrast texts be detected, while on a complex background, it should be relatively high to eliminate background and highlight texts. Thus, we propose a background-complexity-adaptive local thresholding algorithm.

We use two concentric squares [Fig. 3(a)] to scan the edge map scaled for the current level, called the *EMP*. The larger square is called "*window*" and the smaller one is called "*kernel*." The edge map is scanned kernel by kernel. We analyze the background complexity and the edge strength histogram inside the window to determine the local threshold for the kernel. This makes the local threshold vary smoothly across the edge map. The height of kernel is h and that of window is $3h$. The selection criteria of h are related to the minimum and maximum font sizes to be handled in a level, i.e., 8 and 24, respectively. The kernel should not be much larger than the minimum font size and the window should not be smaller than the maximum font size. Thus, h is set to 10 according to experimental statistics.

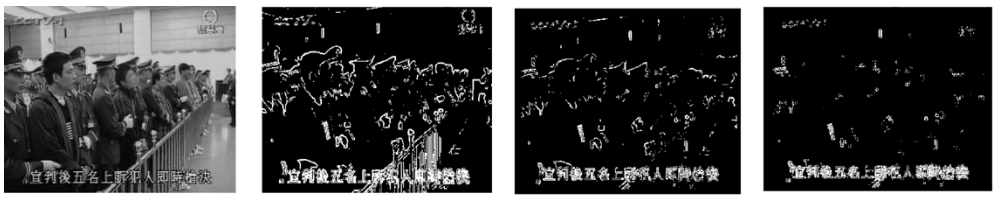
The background complexity is analyzed by the horizontal projection profile of the window, as shown in Fig. 3(b). P_i, $i = 1 \ldots 3h$, is the number of edge pixels in the Row_i. During the projection, the highest edge strength $S_{\max}$ and the edge strength histogram of the window are also obtained. By analyzing from P_1 to P_{3h}, we can get the maximum number of continuous *blank* rows (if $P_i < \delta$, Row_i is a *blank* row), denoted by *MAX_NUM_BL_ROWS*. Thus, the local threshold for the kernel T_{kernel} is determined as follows:

$$T_{\text{kernel}} = \begin{cases} S_{\max}, & \sum_{i=1\ldots3h} P_i < \varepsilon \\ T_{\text{low}}, & MAX\_NUM\_BL\_ROWS \geq MIN\_SPC \\ T_{\text{high}}, & MAX\_NUM\_BL\_ROWS < MIN\_SPC. \end{cases} \tag{3}$$

Equation (3) shows that, if the total number of edge pixels in the window is too few, the kernel cannot be in a text area. So T_{kernel} is set to be $S_{\max}$ to suppress all edge pixels in the kernel. Otherwise, it may be in a text area, and then its background complexity is checked. According to the assumption of horizontal orientation, if a text string lies on a clear background, there should be some blank rows above and below it. Considering the case of multiline captions, the number of continuous blank rows may not be too large. We set *MIN_SPC* to 4 by statistical analysis of *MAX_NUM_BL_ROWS* in many known clear-background and complex-background cases. If $MAX\_NUM\_BL\_ROWS \geq MIN\_SPC$, it is a clear background; otherwise, it is a complex background. T_{kernel} is set to be T_{low} for the former, and set to be T_{high} for the latter.

T_{low} and T_{high} are determined from the edge strength histogram of the window. The range of edge strength is quantized to 64 bins for quick computation. We first divide the histogram into two parts by the global mean of edge strength in the window, denoted by M. To get T_{low}, we only consider the lower part of histogram, i.e., from 0 to M, in order to eliminate the interference of strong edges. For the similar reason, we only use the higher part of histogram, i.e., from M to 64, to find T_{high}. The Otsu method [25] is employed to automatically locate the optimal threshold in the selected part of histogram.

The local thresholding method can successfully suppress the background edges around the text areas. However, in complex background areas, some edge pixels of characters are also suppressed due to the high threshold used, which decreases the density of the characters. Therefore, we design a text-like area recovery filter to bring some text edges back. The filter consists of two steps: the text-like edge labeling and the hysteresis edge recovery. Note that the result of applying the local threshold on the *EMP* is stored in a new edge map, called the EMP'. Initially, all edge pixels in the EMP' are labeled as "*NON_TEXT*". The text-like edge labeling employs a 10×4 rectangle to scan the EMP' row by row stepping 5×2. If the edge density in



(a) Original image (b) Global thresholding (c) Local thresholding (d) Text-like area recovery

Fig. 5. Stepwise results of text detection.

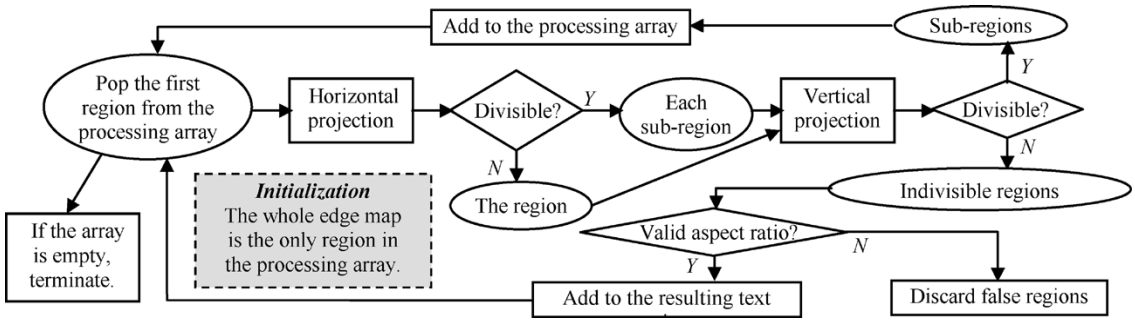


Fig. 6. Workflow of coarse-to-fine localization.

the current rectangle is larger than ξ, all edge pixels in the rectangle are labeled as "*TEXT*." ξ is trained from 100 Chinese text regions, 100 English regions, and 100 nontext regions on different backgrounds. After the labeling, all edge pixels with the "*NON_TEXT*" label are suppressed. Thus, the labeling also acts as a spatial high-pass filtering. For each "*TEXT*" edge pixel, a hysteresis thresholding mask [Fig. 4(a)] is applied to bring back some lower-contrast edge pixels in its neighborhood. The mask is derived from the stroke-blurred model [Fig. 4(b)]. Due to the video compression, edges between character strokes and their background become smooth, as shown in Fig. 4(b), where the left one is the model of light text and the right one is that of dark text. Correspondingly, Fig. 4(c) shows the edge strength distribution of Fig. 4(b), from which we know that a strong edge pixel is surrounded by some lower strength edge pixels. Therefore, a 5×5 mask [Fig. 4(a)] is derived to create a hysteresis threshold surface around the strong edge pixel. Let $H(i,j)(-2 \le i,j \le 2)$ denote the mask. When H is centered at a "*TEXT*" edge pixel $EMP'(x,y)$, the hysteresis thresholding is performed as shown in (4) at the bottom of the page.

After the text-like area recovery, the nontext edges are largely removed, so that the text areas are highlighted. Fig. 5(a) shows an example of a text string on a complex background. The result of global thresholding is shown in Fig. 5(b), where the right part of the text string is embedded in a complex "barrier" area. The local thresholding separate the text string from the background [Fig. 5(c)]. Finally, the text-like area recovery enhances the edge density of text area and removes those sparse edges [Fig. 5(d)]. Until now, the text detection is accomplished and the current EMP' is ready for the text localization.

C. Text Localization

In the current EMP', the high-density areas indicate text regions. Since the local stroke density of Chinese characters varies significantly, the bottom-up method is prone to breaking an entire text string. On the other hand, the top-down method is able to handle this problem due to its global vision. According to the horizontal orientation assumption, the combined horizontal and vertical projection method is an efficient way to localize text strings. However, the conventional one pass of horizontal and vertical projections cannot separate complex text layouts which appear frequently in videos showing tabular reports. Wernicke and Lienhart [19] at first engage the region-growing technique on an edge map to obtain initial bounding boxes and then perform multiple passes of horizontal and vertical projections within initial bounding boxes to segment text regions. Therefore, the entirety of the initial bounding boxes is critical because they will only be divided and never combined. However, the region growing may break a ground-truth text region into several initial bounding boxes due to the variety of local stroke density of Chinese characters. Therefore, we propose a coarse-to-fine localization method [11], which gets rid of region growing while keeping multiple passes of horizontal and vertical projection, so as to handle both multilingual texts and complex layouts. Fig. 6 shows the workflow.

Initially, the whole EMP' is the only region to be processed. Each time, the first region in the processing array, denoted by R_0, goes through this workflow. First, we perform a horizontal projection inside the region. If R_0 is not vertically divisible, it is used as a whole for the vertical projection; otherwise, all sub-

$$EMP'(x+i,y+j)_{-2\le i\le 2,-2\le j\le 2} = \begin{cases} EMP(x+i,y+j), & EMP(x+i,y+j) \ge EMP'(x,y)\cdot H(i,j) \\ EMP'(x+i,y+j), & \text{otherwise} \end{cases} \tag{4}$$

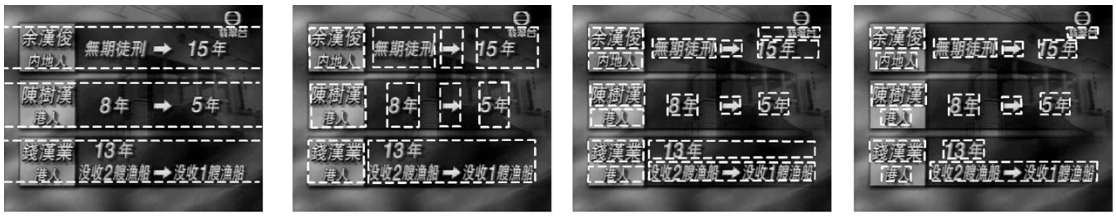


Fig. 7. Stepwise illustration of coarse-to-fine localization on a complex text layout.

regions of R_0 enter the vertical projection. If an input region of vertical projection is horizontally divisible, all its subregions are added to the processing array; otherwise, the indivisible region will be added to the resulting text regions after passing the aspect ratio check. Another pass begins after all input regions of vertical projection have been processed. These steps iterate until the processing array is empty.

Fig. 7 illustrates the localization steps on a video image with a complex text layout, where text strings are overlapping in both horizontal and vertical directions. The first horizontal projection roughly divides three text rows [Fig. 7(a)], and then the first vertical projection divides them horizontally [Fig. 7(b)]. However, there are still many text strings that have not been divided. The second pass of horizontal and vertical projections locates all texts correctly [Fig. 7(c) and (d)].

In order to divide a region, besides the conventional peak/valley classification in the horizontal and vertical projection profiles, we introduce the following two rules according to the multilingual text characteristics. Let *MIN_FONT* and *MAX_FONT* denote the minimum font size and the maximum font size to be handled in a level, respectively. Let the minimum aspect ratio of a Chinese character be *MIN_ASP_RATIO*. We have the following.

1) For the horizontal projection, if the width of a peak, i.e., the height of the potential subregion, is less than *MIN_FONT*, the peak should be suppressed.
2) For the vertical projection, if the input region is indivisible in the previous horizontal projection and the height of the region is between *MIN_FONT* and *MAX_FONT*, a valley whose width is less than $1.5 \times MIN\_ASP\_RATIO \times \text{height}$ between two peaks should be skipped. This is to avoid separating a text string at low stroke-density parts.

Before a region (Width × Height) is added to the resulting text regions, we check its aspect ratio by the following rule: Height $\leq MAX\_FONT$ and Width $\geq$ Height $\times$ MIN_ASP_RATIO. A region that dissatisfies this rule will be discarded.

For each localized text region, the last horizontal and vertical projection profiles within the region can be recorded as its signature ($h\_\text{sign}_i, i = 1 \ldots \text{Height}$ and $v\_\text{sign}_j, j = 1 \ldots \text{Width}$). This signature will be utilized to compare the identity of two text regions with similar locations in different frames.

D. Multiframe Verification

The function of multiframe verification is to eliminate those text regions that are transitory or have already been detected in the previous frames. The algorithm for multiframe verification has already been well described in the literature [10], [20]. Checking whether a text region is "new" or "old" first depends on the location comparison between the current one and those on the previous frame. If two text regions on consecutive frames have similar locations, we then use the signature comparison to determine whether they are the same text or not. To tolerate possible location offsets and edge density changes of the same text over time, the signature distance metric is designed as given in (5), shown at the bottom of the page.

The smaller $Dist(Sign^A, Sign^B)$ is, the more identical the two text regions A and B are. According to our experimental statistics, we set the threshold for signature distance to be 4.0. The multiframe verification only accepts the text regions lasting for more than two seconds.

E. Text Extraction

The goal of text extraction is to convert the grayscale image in an accepted text region into the OCR-ready binary image, where all pixels of characters are in black and others are in white. The difficulties of text extraction come from three aspects: 1) the unknown color polarity, i.e., text is light or dark; 2) various stroke widths; and 3) the complex background.

After the multiframe verification, bitmap integration over time [10] is often used to remove the moving background of a

$$\begin{cases} Dist(Sign^A, Sign^B) = \frac{Dist_{\text{Horizontal}}(Sign^A, Sign^B) + Dist_{\text{Vertical}}(Sign^A, Sign^B)}{2} \\ Dist_{\text{Horizontal}}(Sign^A, Sign^B) = \frac{1}{\text{Height}-2} \\ \times \sum_{i=2}^{\text{Height}-1} \text{Min}(|h\_\text{sign}_i^A - h\_\text{sign}_i^B|, |h\_\text{sign}_i^A - h\_\text{sign}_{i-1}^B|, |h\_\text{sign}_i^A - h\_sign_{i+1}^B|)^2 \\ Dist_{\text{Vertical}}(Sign^A, Sign^B) = \frac{1}{\text{Width}-2} \\ \times \sum_{j=2}^{\text{Width}-1} \text{Min}(|v\_\text{sign}_j^A - v\_\text{sign}_j^B|, |v\_\text{sign}_j^A - v\_\text{sign}_{j-1}^B|, |v\_\text{sign}_j^A - v\_\text{sign}_{j+1}^B|)^2 \end{cases} \tag{5}$$

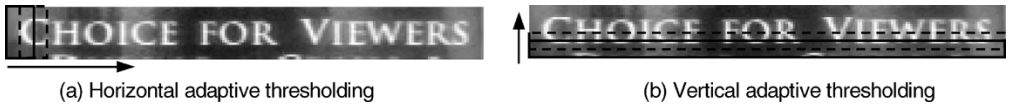


Fig. 8. Adaptive thresholding of the text image.

text region; however, this algorithm works on the condition that the text color is light, which is not true for many video texts. Some methods [10], [18], [22] assume that the text is of high intensity compared to the background, which obviously limits their applications. Antani *et al.* [21] proposed a text extraction method that can detect the color polarity by connected-component analysis. It at first binarizes the text image into two polar images: a positive one and a negative one. Then, the one with the higher score on text-like characteristics is selected and the other is discarded. This method works well for a text on a clear and contrastive background. However, since the text-like characteristics are based on the connected-component analysis, it depends heavily on the thresholding results. Thus, it cannot handle complex backgrounds and various text appearances. Wernicke and Lienhart [19] proposed a color-histogram-based color polarity detection method, which compares the histogram of four central rows of a text region and the histogram of the two rows above plus two rows beneath to determine the text color. This method avoids thresholding and works well when text color and background color are contrasting. However, it cannot handle the case that text and background have similar colors. To solve this problem, we developed a statistic color-polarity classification approach [26], which first uses the Otsu thresholding method to generate a binary text image and then explores the statistic relationships between the number of black edges and that of white edges to classify the color polarity. The classification accuracy reaches 98.5%. To ease the bitmap integration process, if the text color is dark, we inverse the grayscale color of the text image so that the text is always of high intensity. Now it is safe to perform bitmap integration over time to remove the moving background of accepted text regions. Therefore, we obtain an image of a text region with text in light color (hereinafter, called text image) by three processes: color polarity classification, color inversion (if necessary), and bitmap integration over time.

Using the multiresolution paradigm, we handle the text strings whose heights range from 8 to 72 pixels. Usually, the larger the font size is, the thicker the stroke width is. To handle this difficulty, we normalize the height of text image to 24 pixels while keeping its aspect ratio. Both English and Chinese characters have moderate stroke widths in this font size, which is also a favorite size for most OCR software. For some fonts, even strokes in different directions have different widths. According to our observation, the stroke width (in pixels) in a normalized text image varies between $MIN\_W = 1$ and $MAX\_W = 3$. Since our extraction method needs to analyze the background of the text, the four borders of the normalized text image are further extended by four pixels to include more background pixels. The following processes are based on the extended 32-pixel-high text image. The region inside the original four borders is defined as TEXT_REGION, and the outside region is defined as EXTENDED_REGION. Some

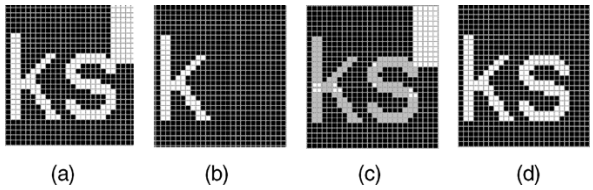


Fig. 9. Comparison between seed-fill algorithm and inward filling algorithm.

preprocesses on the text image, e.g., the grayscale histogram equalization, are performed to enhance the contrast.

Since bitmap integration over time only affects moving backgrounds, the still background of a text remains unchanged. However, still backgrounds may also be complex. Simple global thresholding is not robust to extract text pixels since background pixels may have an intensity similar to that of the text pixels. Sato *et al.* [10] proposed a character extraction filter integrated by four directional filters, which extract the linear elements in horizontal, vertical, left diagonal, and right diagonal directions. This filter highlights linear strokes well but the output is weaker at corners and intersections of strokes. Sato *et al.* thought this limitation "does not present much of problem in segmenting and recognizing characters," as they considered only English characters. However, Chinese characters contain much more corners and intersections that are important to OCR. Even if we can expand the filter bank, e.g., adding eight directional corner filters and train them to adapt Chinese characters, it will be not efficient for English characters then. Thus, we propose a language-independent text extraction method that consists of three steps: 1) adaptive thresholding; 2) dam point labeling; and 3) inward filling.

Considering the different background intensities along the text, we design an adaptive thresholding to produce a better binary image. Let $G(x, y)$ and $B(x, y)$ denote the input grayscale text image and the output binary text image, respectively, where $0 \leq x < image\_width$ and $0 \leq y < 32$. $B(x, y)$ is initialized as totally "*White*." The adaptive thresholding is performed by first moving a 16×32 window along the image horizontally with stepping 8 [Fig. 8(a)], and then moving an $image\_width \times 8$ window along the image vertically with stepping 4 [Fig. 8(b)]. In each window, if $G(x, y)$ is below the local threshold calculated by the Otsu method, $B(x, y)$ is set to be "*Black*." Finally, $B(x, y)$ contains the resulting binary image, where text pixels are "*White*."

Since some background pixels may have very similar intensities to the text, they are also "*White*" in the binary image. We find that most background areas in the TEXT_REGION spread into the EXTENDED_REGION due to the continuity of background. Therefore, filling from the EXTENDED_REGION is a good way to remove the background pixels. However, filling is also unsafe since the disconnectivity between the background pixels and the text pixels cannot be guaranteed. For example, the

Fig. 10. Text pixel extraction.

seed-fill algorithm proposed in [20] fills from each pixel on the extended boundary of a text region. It assumed "the pixels on the boundary do not belong to the text and since the text contrasts with its background," so the filling would not remove any character pixels. However, this assumption is not true for some complex backgrounds [Fig. 9(a)], which contain regions with similar color to characters. If such region is connected with a character, this character will be removed by seed-fill algorithm [Fig. 9(b)]. Thus, we introduce the "dam point" inside the TEXT_REGION to prevent the filling from flooding into text pixels. The dam points are defined as follows:

$$\begin{aligned}\text{Dam points} = \{(x,y) | B(x,y) = \text{"}White\text{"} \wedge MIN\_W \\ \leq \text{MIN}[\text{H\_Len}(x,y), \text{V\_Len}(x,y)] \leq MAX\_W\}\end{aligned}$$

where $\text{H\_Len}(x,y)$ and $\text{V\_Len}(x,y)$ return the horizontal connectivity length and vertical connectivity length, respectively, of the given position (x,y). $\text{H\_Len}(x,y)$ is calculated as the maximum length of horizontally continuous white segments passing (x,y), and $\text{V\_Len}(x,y)$ is calculated as the maximum length of vertically continuous white segments passing (x,y). For example, the H_Len of the bottom-left pixel of "k" in Fig. 9(a) is 2, while its V_Len is 17. Therefore, the pixels of text strokes are labeled as "dam points," which are set to be "Gray" in $B(x,y)$, as shown in Fig. 9(c).

After labeling the dam points, it is safe to perform the inward filling, which scans every pixel in the EXTENDED_REGION of $B(x,y)$. If a pixel is "*White*," we use the traditional flood fill method to find all its connected "*White*" pixels. All the connected "*White*" pixels and the original pixel itself will be set to be "*Black*." After the inward filling finishes, all non-"*Black*" pixels are set to be "*White*." The inward filling result of Fig. 9(a) is shown in Fig. 9(d). Fig. 10 demonstrates the dam-point-based inward filling result of a real text image. Thus, the current $B(x,y)$ is the final result of the text extraction.

V. Experimental Results

Since the evaluation criteria for the text detection and localization and those for the text extraction are different, we divide the experiments into two phases. The first phase focuses on the text detection and localization algorithms, which take a video clip as input and identify the localized rectangular text regions. The second phase evaluates the text extraction algorithm, which retrieves the grayscale text images according to the localized text regions, and produces the binary text images where text pixels are white.

A. Evaluation of Text Detection and Localization

The proposed text detection and localization algorithms have been tested on a number of real-life video clips. Chinese videos are captured from the TVB programs aired by the Hong Kong Jade station. The video resolution is 288×352 (PAL). English videos are captured from both Hong Kong and CNN channels. The video resolution is 240×352 (NTSC). A total of 180 min of programming is used to calibrate the previously mentioned thresholds. Another fifty video clips covering news, financial reports, sports videos, and advertisements, totaling about 100 min, are utilized as testing data.

Fig. 11 shows the experimental results of video text detection and localization. In Fig. 11(a), both low-contrast texts and high-contrast texts are successfully detected. The texts embedded in the complicated background [Fig. 11(b)] are correctly located. Fig. 11(c) shows that coarse-to-fine detection handles the complex text layouts well. Fig. 11(d) demonstrates that our approach is robust to various font-sizes. These results also confirm that the proposed video text detection and localization algorithms are capable of handling multilingual texts. Note that the missed texts in the rightmost image of the second row are scrolling texts, which cannot be handled by the current algorithm. Fig. 11(e) shows some examples of misses or false detections. The text region in the leftmost image is enlarged due to the similar textured background. The slant texts in the second image from the left are missed since our method only handles horizontal texts. In the third image from the left, the clustered microphones are misdetected as a text region. The text in the rightmost image is separated due to two neighboring sparse characters.

For a quantitative evaluation, we define that a detected text region is correct on the condition that the intersection of the detected text region (DTR) and the ground-truth text region (GTR) covers more than both 90% of the DTR and 90% of the GTR. The GTRs in the testing video clips are localized manually. Thus, we define a set of quantitative evaluation measurements for the text detection and localization algorithms as follows.

1) ***Speed***. The speed is indicated by the average processing time per frame for the text detection and localization tested on a PC with one PIII 1G CPU.
2) ***Detection Rate***. The detection rate evaluates how many percents of all ground-truth text regions are correctly detected. That is,

$$\text{Detection Rate} = \frac{\text{Num(correct DTRs)}}{\text{Num(GTRs)}}.$$

3) ***Detection Accuracy***. The detection accuracy evaluates how many percents of the detected text regions are correct. Namely

$$\text{Detection Accuracy} = \frac{\text{Num(correct DTRs)}}{\text{Num(all DTRs)}}.$$

(a) Different contrast (b) Complex background (c) Horizontal/vertical overlaps (d) Different font-size

(e) Examples of misses and false detections

Fig. 11. Detection and localization results on real video images.

TABLE I
PERFORMANCE COMPARISON FOR VIDEO TEXT DETECTION AND LOCALIZATION

Item / Method	Speed (sec/frame)	Detection Rate	Detection Accuracy	Temporal Coverage
Our method	0.25	91.1%	90.8%	93.2%
Method in [10]	0.34	51.8%	67.3%	88.7%

4) ***Temporal Coverage***. The temporal coverage averages the time span overlapping rates of all pairs of detected text region and ground-truth text region, defined as follows:

$$\text{Temporal Coverage} = \frac{1}{\text{Num(DTR)}} \cdot \sum_{i=1}^{\text{Num(DTR)}} \frac{\text{TimeSpan}(\text{DTR}_i) \cap \text{TimeSpan}(\text{GTR}_i)}{\text{TimeSpan}(\text{GTR}_i).}$$

Table I shows the experimental results of our method and the method in [10], which is based on vertical edge detection and region growing. Our proposed method processes a frame in 0.25 s. As stated before, we sample two frames/s; therefore, considering the time for text extraction, which depends on the number of text in the video, the total processing time for a video clip is usually shorter than the playback length of the video clip. The missed texts of our method include scrolling texts, transitory texts, nonhorizontal texts, and texts with very low contrast to background. The main cause of false detections in our method is the random alignment of textured objects since our algorithms impose a relatively loose constraint on the edge distribution inside a candidate region for the multilingual adaptability. This constraint, however, can be tightened for an adjusted detection accuracy. The method in [10] loses points mainly on breaking Chinese text strings and enlarging text regions on complex backgrounds.

B. Evaluation of Text Extraction

In this experiment, the proposed text extraction method is compared with two other methods: the Otsu thresholding method [25] (hereinafter called the Otsu method) and the Sato filtering method [10] (hereinafter called the Sato method). We choose them because the Otsu method is a simple but classic solution employed by many text extraction schemes, while the Sato method is a complex stroke-based solution proposed

Fig. 12. Comparison of three text extraction methods.

recently. Since both of these methods hold the assumption of color polarity, we use the color polarity detection algorithm [26] described in Section IV-E as the preprocessing for them.

We test the three text extraction methods using 100 testing text images, which are generated by the previous text detection and localization experiments. Fig. 12 displays their experimental results on a set of typical text images, which contain different font sizes, font styles, languages, and background complexities. Both the proposed method and the Sato method normalize the height of text image, while the Otsu method does not. To ease the comparison, their results are normalized to the same size, as shown in Fig. 12. The text pixels should be white in the final binary images.

The experimental results show clear differences among the three methods. The Otsu method is a histogram-based global thresholding method. Therefore, it can extract text pixels from simple backgrounds, and it is insensitive to font and language. However, it cannot handle complex backgrounds, such as texts in Fig. 12(a)–(c), (f), (g), (l), (m), (o), and (p). The Sato method uses four directional filters, whose values are trained by some fixed English fonts, to calculate the probability of each pixel being on a text stroke. Since these filters assume that an ideal stroke has a certain space from other strokes, they do not fit for Chinese characters since the stroke densities are quite different. Thus, the Sato method extracts English texts well, but it fails to extract most Chinese texts [Fig. 12(c), (f), (h), (l), (n), and (o)]. Moreover, neither the Otsu method nor the Sato method can handle the case that the background and the text have similar colors [Fig. 12(c), (m), (o), and (p)]. In contrast, our proposed method demonstrates good extraction quality in these cases. It consists of the adaptive thresholding and the inward filling, both of which are insensitive to font and language variances. For a simple background, the adaptive thresholding is enough and the inward filling is actually skipped. For a complex background or the case where background color and text color are similar, the inward filling can remove largely the background pixels while keeping the text pixels.

Since the text extraction is actually an image segmentation task that segments the text pixels from the background pixels, we employ a well-known image segmentation evaluation protocol—Probability of Error (*PE*) [27]—to quantitatively evaluate the extraction results. *PE* is defined as

$$PE = P(O) \times P(B|O) + P(B) \times P(O|B)$$

where $P(B|O)$ is the probability of error in classifying text pixels as background pixels, $P(O|B)$ is the probability of error in classifying background pixels as text pixels, and $P(O)$ and

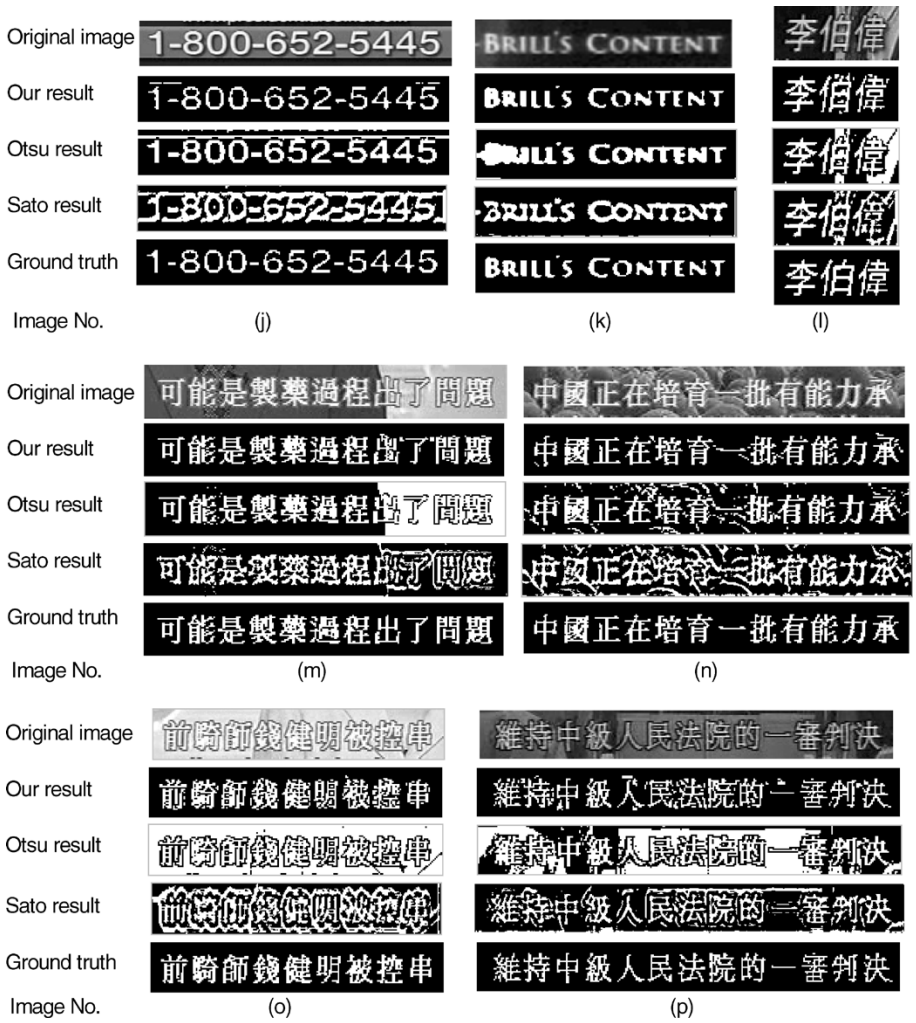


Fig. 12. *(Continued.)* Comparison of three text extraction methods.

TABLE II
PE EVALUATION OF THREE TEXT EXTRACTION METHODS

Method \ Text	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
Our method	**0.028**	**0.059**	**0.068**	**0.040**	**0.042**	**0.088**	**0.070**	**0.114**
Otsu method	0.183	0.499	0.526	0.041	0.117	0.196	0.214	0.282
Sato method	0.087	0.114	0.223	0.136	0.151	0.299	0.210	0.315
Method \ Text	(i)	(j)	(k)	(l)	(m)	(n)	(o)	(p)
Our method	**0.001**	**0.044**	**0.030**	**0.069**	**0.064**	**0.057**	**0.104**	**0.086**
Otsu method	0.058	0.139	0.075	0.268	0.311	0.154	0.598	0.413
Sato method	0.068	0.253	0.107	0.237	0.221	0.262	0.409	0.258

$P(B)$ are, respectively, *a priori* probabilities of text pixels and background pixels in images, i.e., the probabilities in the ground-truth images. The smaller the *PE* is, the smaller the difference between the extracted text image and the ground-truth image is, i.e., the higher the extraction accuracy is. The *PE*s of these three methods for the text images in Fig. 12 are given in Table II.

In each column of Table II, the minimum *PE* is boldfaced. We can see that the proposed method always achieves the minimum *PE*, i.e., the best extraction accuracy, for all 16 images. The means of *PE* for our method, the Otsu method, and the Sato method are 0.060, 0.255, and 0.209, respectively. This again confirms the superiority of our proposed method. We also evaluate the robustness of these three methods by calculating their mean square deviations of *PE* in Table II. They are 0.0008, 0.0281, and 0.0082 for the proposed method, the Otsu method, and the Sato method, respectively. As we can observe, the proposed method has a much smaller deviation of *PE* than the other two methods; therefore, it is the most robust method among the three tested methods. Since PE evaluation is not very intuitive,

TABLE III
CERs EVALUATION OF THREE TEXT EXTRACTION METHODS

Method \ CER	English CER	Chinese CER	Overall CER
Our method	**0.093**	**0.215**	**0.154**
Otsu method	0.376	0.603	0.490
Sato method	0.153	0.744	0.449

we also provide the character error rates (CER) of the three methods by a commercial OCR software—PenPower® Chinese OCR Professional V3.1[1]—which can recognize English, Traditional Chinese, and Simplified Chinese. For a total of 1136 English and Chinese characters in the 100 test text images, the CERs of these three methods are shown in Table III.

From Table III, we can see that our method achieves significantly lower error rates. This is due to the capability of handling multilanguage and complex backgrounds. Its Chinese CER is higher than its English CER because the Chinese character is much denser; therefore, some background pixels inside the character may be misclassified as dam points so as to survive after the filling, such as Fig. 12(o) and (p). The Otsu method, on the other hand, is too simple to handle complex backgrounds; therefore, it generates high error rates for both English and Chinese characters. However, the Sato method produces the highest error rate for Chinese characters since it not only fails to extract text pixels out of complex backgrounds, but also introduces distortions to the text pixels on clean backgrounds.

VI. CONCLUSION

The automatic multilingual adaptability of video text processing is very important to digital video libraries emphasizing multilingual capabilities. This paper proposes a novel multilingual video text detection, localization, and extraction method. Although this paper focuses on Chinese and English, the proposed method is capable of handling multilingual texts because it depends on language-independent characteristics. According to our analysis of multilingual text characteristics, the proposed method can be applied to French, Spanish, Japanese, and Korean without much adjustment. Besides emphasizing the multilingual attribute, the proposed method is also robust to various font sizes, font styles, contrast levels, and background complexities. This is a comprehensive approach which systematically integrates the following techniques to accomplish multilingual video text detection, localization and extraction.

1) **Text detection:** the sequential multiresolution paradigm, the background-complexity-adaptive local thresholding of edge map, and the hysteresis text edge recovery.
2) **Text localization:** the coarse-to-fine localization scheme, the multilanguage-oriented region dividing rules, and the signature-based multiframe verification.
3) **Text extraction:** the color polarity classification, the adaptive thresholding of grayscale text images, and the dam-point-based inward filling.

[1]Available. [Online]. http://www.penpower.com.tw/enversion/default.htm

The novel concepts and techniques introduced in this paper include the detailed classification and analyses of multilingual text characteristics, the sequential multiresolution text detection paradigm, the background complexity analysis, the coarse-to-fine text localization scheme, the multilanguage-oriented region dividing rules, and the dam-point-based inward filling. Detailed experimental results and the comparisons with other methods are also reported, confirming that the proposed method is capable of handling multilingual video texts accurately and is robust to various background complexities and text appearances.

A known limitation of the current method is that it cannot detect motion texts due to the assumption of stationary text. However, this capability can be enabled by adding a signature-based text-region tracking algorithm to the multi-frame verification process. Another limitation is that nonhorizontally aligned texts cannot be localized. Nonhorizontally aligned texts have seldom been observed in English videos; however, they occur with low probabilities in Chinese, Japanese, and Korean videos, mostly in vertical alignments. These issues will be addressed in our future research.

REFERENCES

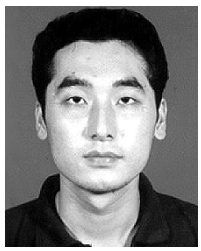
[1] Q. Huang, Z. Liu, A. Rosenberg, D. Gibbon, and B. Shahraray, "Automated generation of news content hierarchy by integrating audio, video, and text information," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 6, 1999, pp. 3025–3028.
[2] W. Qi, L. Gu, H. Jiang, X.-R. Chen, and H.-J. Zhang, "Integrating visual, audio and text analysis for news video," in *Proc. Int. Conf. Image Process.*, vol. 3, 2000, pp. 520–523.
[3] M. R. Lyu, E. Yau, and K. S. Sze, "A multilingual, multimodal digital video library system," in *Proc. Joint Conf. Digital Libraries*, Portland, OR, Jul. 2002, pp. 145–153.
[4] R. Lienhart, "Automatic text recognition for video indexing," in *Proc. ACM Multimedia*, Boston, MA, Nov. 1996, pp. 11–20.
[5] S. Sato and T. Kanade, "NAME-IT: Association of face and name in video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, San Juan, Puerto Rico, Jun. 1997, pp. 368–373.
[6] A. Jian and S. Bhattacharjee, "Text segmentation using gabor filters for automatic document processing," *Machine Vis. Applicat.*, vol. 5, pp. 169–184, 1992.
[7] V. Wu, R. Manmatha, and E. M. Riseman, "Textfinder: An automatic system to detect and recognize text in images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 11, pp. 1224–1229, Nov. 1999.
[8] H. Li, D. Doermann, and O. Kia, "Automatic text detection and tracking in digital video," *IEEE Trans. Image Process.*, vol. 9, no. 1, pp. 147–156, Jan. 2000.
[9] K. C. Jung, J. H. Han, K. I. Kim, and S. H. Park, "Support vector machines for text location in news video images," in *Proc. IEEE Region 10 Conf. Syst. Technolog. Next Millennium*, vol. 2, 2000, pp. 176–180.
[10] T. Sato, T. Kanade, E. K. Hughes, and M. A. Smith, "Video OCR for digital news archive," in *Proc. IEEE Workshop Content-Based Access Image Video Database*, 1998, pp. 52–60.
[11] M. Cai, J. Song, and M. R. Lyu, "A new approach for video text detection," in *Proc. Int. Conf. Image Process.*, Rochester, NY, Sep. 2002, pp. 117–120.

[12] U. Gargi, S. Antani, and R. Kasturi, "Indexing text events in digital video databases," in *Proc. 14th Int. Conf. Pattern Recognit.*, vol. 1, 1998, pp. 916–918.
[13] Y. Zhong, H.-J. Zhang, and A. K. Jain, "Automatic caption localization in compressed video," in *Proc. Int. Conf. Image Process.*, vol. 2, 1999, pp. 96–100.
[14] Y.-K. Lim, S.-H. Choi, and S.-W. Lee, "Text extraction in MPEG compressed video for content-based indexing," in *Proc. Int. Conf. on Pattern Recognit.*, vol. 4, 2000, pp. 409–412.
[15] A. K. Jain and B. Yu, "Automatic text location in images and video frames," *Pattern Recognit.*, vol. 31, no. 12, pp. 2055–2076, 1998.
[16] L. Agnihotri and N. Dimitrova, "Text detection for video analysis," in *Proc. IEEE Workshop Content-Based Access Image Video Libraries*, 1999, pp. 109–113.
[17] V. Y. Mariano and R. Kasturi, "Locating uniform-colored text in video frames," in *Proc. 15th Int. Conf. Pattern Recognit.*, vol. 4, 2000, pp. 539–542.
[18] X. Gao and X. Tang *et al.*, "Automatic news video caption extraction and recognition," in *Proc. LNCS 1983: 2nd Int. Conf. Intell. Data Eng. Automated Learning Data Mining, Financial Eng., Intell. Agents*, K. S. Leung *et al.*, Eds., Hong Kong, 2000, pp. 425–430.
[19] A. Wernicke and R. Lienhart, "On the segmentation of text in videos," in *Proc. IEEE Int. Conf. Multimedia Expo*, vol. 3, Jul. 2000, pp. 1511–1514.
[20] R. Lienhart and A. Wernicke, "Localizing and segmenting text in images, videos and web pages," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 4, pp. 256–268, Apr. 2002.
[21] S. Antani, D. Crandall, and R. Kasturi, "Robust extraction of text in video," in *Proc. 15th Int. Conf. Pattern Recognit.*, vol. 1, 2000, pp. 831–834.
[22] D. Chen, K. Shearer, and H. Bourlard, "Text enhancement with asymmetric filter for video OCR," in *Proc. 11th Int. Conf. Image Anal. Process.*, 2001, pp. 192–197.
[23] B. T. Chun, Y. Bae, and T.-Y. Kim, "Text extraction in videos using topographical features of characters," in *Proc. IEEE Int. Fuzzy Syst. Conf.*, vol. 2, 1999, pp. 1126–1130.
[24] I. Sobel, "An isotropic 3×3 image gradient operator," in *Machine Vision for Three-Dimensional Scenes*, H. Freeman, Ed. New York: Academic, 1990, pp. 376–379.
[25] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst., Man, Cybernet.*, vol. SMC-9, no. 1, pp. 62–66, Jan. 1979.
[26] J. Song, M. Cai, and M. R. Lyu, "A robust statistic method for classifying color polarity of video text," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. 2003*, vol. 3, Apr. 2003, pp. 581–584.
[27] 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.

Michael R. Lyu (S'84–M'88–SM'97–F'04) received the B.S. degree in electrical engineering from National Taiwan University, Taipei, in 1981, the M.S. degree in computer engineering from the University of California, Santa Barbara, in 1985, and the Ph.D. degree in computer science from the University of California, Los Angeles, in 1988.

He is currently a Professor withat the Computer Science and Engineering Department, Chinese University of Hong Kong. He was with the Jet Propulsion Laboratory, Pasadena, CA, as a Technical Staff Member from 1988 to 1990. From 1990 to 1992, he was with the Electrical and Computer Engineering Department, University of Iowa, Ames, as an Assistant Professor. From 1992 to 1995, he was a Member of the Technical Staff in the Applied Research Area of Bell Communications Research (Bellcore). From 1995 to 1997, he was a Research Member of the Technical Staff with Bell Laboratories, Murray Hill, NJ, which was first part of AT&T and later became part of Lucent Technologies. His research interests include software reliability engineering, distributed systems, fault-tolerant computing, wireless communication networks, Web technologies, digital library, and E-commerce systems. He has published over 190 refereed journal and conference papers in these areas. He has participated in more than 30 industrial projects and helped to develop many commercial systems and software tools. He has been frequently invited as a keynote or tutorial speaker to conferences and workshops in the U.S., Europe, and Asia. He is the editor of two book volumes, *Software Fault Tolerance* (New York: Wiley, 1995) and *The Handbook of Software Reliability Engineering* (IEEE/McGraw-Hill: New York, 1996).

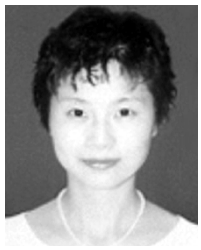
Dr. Lyu initiated the first International Symposium on Software Reliability Engineering (ISSRE) in 1990. He was the Program Chair for ISSRE'96 and has served on program committees for many conferences, including ISSRE, SRDS, HASE, ICECCS, ISIT, FTCS, ICDSN, EUROMICRO, APSEC, PRDC, PSAM and ICCCN. He is the General Chair for ISSRE'2001 and the WWW10 Program Co-Chair. He is on the editorial board of the IEEE TRANSACTIONS ON RELIABILITY and the *Journal of Information Science and Engineering*. He was an editor for the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING.



Jiqiang Song (S'99–A'01–M'02) received the B.S. degree in computer science and the Ph.D. degree in computer science and application from Nanjing University, Nanjing, China, in 1996 and 2001, respectively.

He is currently a Postdoctoral Fellow with the Department of Computer Science and Engineering, Chinese University of Hong Kong. His research interests include graphics recognition, automatic interpretation of engineering drawings, image processing, and video processing. He has published over 20 papers in these areas.

Dr. Song is a member of the IEEE Computer Society and IAPR. He was the recipient of the first place prize in the GREC'2003 Arc Segmentation Contest.



Min Cai received the B.S. degree in computer science and the M.S. degree in computer science and application from Nanjing University, Nanjing, China, in 1998 and 2001, respectively.

From 2001 to 2003, she was a Research Assistant with the Department of Computer Science and Engineering, Chinese University of Hong Kong. She is currently with UTStarCom Co., Ltd., Shenzhen, China. Her research interests include automatic interpretation of engineering drawings, image processing, and video processing. She has published over 10 papers in these areas.