Robustly Extracting Captions in Videos Based on Stroke-Like Edges and Spatio-Temporal Analysis

Xiaoqian Liu, *Member, IEEE*, and Weiqiang Wang, *Member, IEEE*

***Abstract*—This paper presents an effective and efficient approach to extracting captions from videos. The robustness of our system comes from two aspects of contributions. First, we propose a novel stroke-like edge detection method based on contours, which can effectively remove the interference of non-stroke edges in complex background so as to make the detection and localization of captions much more accurate. Second, our approach highlights the importance of temporal feature, i.e., inter-frame feature, in the task of caption extraction (detection, localization, segmentation). Instead of regarding each video frame as an independent image, through fully utilizing the temporal feature of video together with spatial analysis in the computation of caption localization, segmentation and post-processing, we demonstrate that the use of inter-frame information can effectively improve the accuracy of caption localization and caption segmentation. In the comprehensive our evaluation experiments, the experimental results on two representative datasets have shown the robustness and efficiency of our approach.**

***Index Terms*—Spatio-temporal, stroke-like edges, text detection, text extraction.**

I. Introduction

As the technology of digital multimedia develops rapidly, we are exposed to a huge number of images and videos in everyday life. Since captions in movies or TV programs can supply rich semantic information which is very useful in video content analysis, indexing, and retrieval, research efforts have been made toward extracting video captions in various applications, such as traffic monitoring [1], [2], video indexing and retrieval [3], and computerized aid for visually impaired [4]. Generally speaking, the main related techniques involve three aspects: caption detection and localization, caption segmentation and optical character recognition (OCR). Although it appears the same as that for image-based text extraction, it is not a wise choice to directly apply the image-based techniques for video caption extraction, where each video frame is regarded as an independent image, since a very large number of images in videos makes such computation way very inefficient. For instance, a 90-min movie played by 30 frames per second contains 1 620 000 images, and it is computationally too costly if naively applying image-based techniques. Thus, more efficient techniques need to be investigated by exploiting the strong correlation of video content over time. Compared with the image-based counterpart, the segmentation technique in video caption extraction has a broader meaning. Besides text spatial segmentation for images, the temporal segmentation as a new issue for videos is very necessary to make video caption extraction very efficient. The temporal segmentation aims to temporally partition a video into a disjoint set of clips, and each clip contains either the same caption or simply no caption. It can bring two advantages: 1) similar to shot segmentation in content-based video indexing and retrieval, it greatly reduces the overall computational cost by effectively removing the content redundancy of adjacent frames; 2) once a clip is identified to have a same caption, the frames in the clip can be collectively utilized to extract a more accurate caption than only relying on a single frame.

Manuscript received April 15, 2011; revised August 05, 2011 and November 13, 2011; accepted November 17, 2011. Date of publication November 29, 2011; date of current version March 21, 2012. This work was supported by National Natural Science Foundation of China under Grant 60873087. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Jia Li.

X. Liu is with the Graduate University of Chinese Academy of Sciences, Beijing 100190, China (e-mail: xqliu@jdl.ac.cn).

W. Wang is with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: wqwang@ict.ac.cn).

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/TMM.2011.2177646

Existing text extraction methods mainly focus on text in static images, and the related works address two subproblems, text localization and text segmentation. Text localization refers to determining whether and where some characters exist in an image. The corresponding methods can be classified into four categories: edge-based, corner-based, texture-based and stroke-based methods. The edge-based methods [5], [6], [7], [8] locate potential text regions by dense edges. They are often simple and efficient, nevertheless their performance might degrade when the background also contains some regions with dense edges. Corner-based methods [9] identify candidate text regions by seeking densely distributed corners. This is based on the observation that a character generally consists of many strokes with different directions, where the tip point between adjacent strokes usually forms a corner. For instance, Hua *et al.* [9] deal with video frames independently. They utilize the SUSAN detector to find corners in each frame, and then identify the text regions. Texture-based methods [10], [11], [12] utilize various texture descriptors, such as Gabor filter [13] and wavelet transform [14], [15], and then the machine learning techniques such as SVM [16] or neural networks [1] are applied to classify the text and non-text regions. The methods stem from the following fact: The strokes of characters usually possess similar width, color (or grayscale value), and multiple characters in a horizontal (or vertical) line often share similar width, height and spacing, which makes the text regions have the visually repetitive property similar to that of a texture. Since this kind of method needs to train a classifier, relatively high computational cost and the need of huge training dataset are their disadvantages. Stroke-based features have been widely utilized in recent years. Liu *et al.* [17], [18] design a stroke filter which is based on the assumption that strokes have certain width and specific spatial distribution, in which the strokes

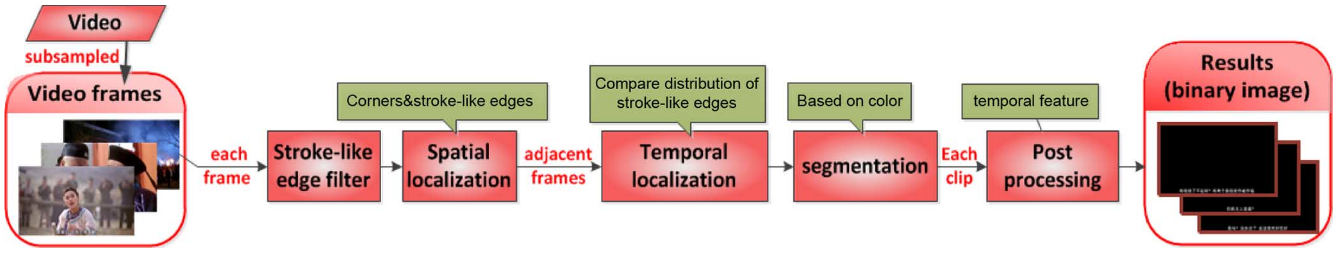

Fig. 1. Framework of the proposed approach.

can be described by its two-side edges. Epshtein *et al.* [19] propose a stroke width transform (SWT) to decide whether the adjacent edges have stroke-like width. After localizing a text region, text segmentation aims to segment out text pixels, in order to produce a binary image subsequently used for character recognition [20]. The existing methods mainly resort to connected component analysis (CCA), K-means algorithm, which is motivated by the fact that text strokes generally have homogeneous color. For instance, Hase *et al.* [21] segment the text regions with similar color based on CCA technique, and Gllavata *et al.* [22] utilize a slightly modified K-means algorithm to extract text pixels from background.

Compared with the research efforts on extracting text in static images, the studies on extracting video captions are relatively very few, especially where inter-frame content correlation is considered. Tang *et al.* [23] utilize the temporal feature (frame difference) to locate the frames containing the same caption, and their approach is more suitably applied in news videos, due to the assumption that the scene changes gradually within a shot. In this paper, we present an effective and efficient approach to extracting video captions in complex background. Compared with the existing methods, our contributions are summarized as follows: 1) we describe a framework of extracting video captions, where the conception of temporal localization is used to cluster frames with the same captions together, and this enables us to more effectively extract captions based on multiple frames, instead of many independent frames; 2) compared with [23], our usage of temporal features is multi-facets: not only in localization, but also in segmentation and post-processing, and the related techniques can be used to general video types; 3) a dedicated stroke-like edge detector is proposed to effectively removing the interference of non-stroke edges in complex background to facilitate the detection and localization of captions. In fact, this detector can also be applied directly to text detection and localization in static images.

II. Our Approach

Different from other approaches, our approach fully utilizes the content correlation of consecutive frames in the detection, spatio-temporal localization and segmentation of video captions, to improve the efficiency and accuracy of the proposed system. As illustrated in Fig. 1, a video is first sub-sampled uniformly by 3 frames per second. A stroke-like edge filter is then utilized to detect the caption regions from each frame. This is followed by caption localization in both space and time. The spatial localization refers to the identification of candidate text regions from stroke-like edge and corner distributions.

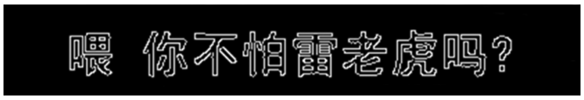

Fig. 2. Contours of captions.

During the temporal localization, we compare the distribution of stroke-like edges between adjacent frames in order to partition a video into clips in which each frame contains the same caption. For each clip containing captions, we first extract candidate caption pixels based on the estimated caption color information. Further, the temporal "and" operation is applied to filter out inconsistent false positive pixels that have similar color to that of foreground captions. This is followed by a post-processing step to refine the segmentation results by *a priori* knowledge of character structures. The generated binary images are used by the OCR module. We will elaborate on these steps in what follows.

A. Detecting Stroke-Like Edges

Although character strokes usually generate dense edges (e.g., Fig. 2), edges might also come from complex background which could be mis-categorized as text edges and thus decrease the accuracy for text detection. To alleviate this issue, we propose a stroke-like edge detector to remove the interferences of non-stroke edges from complex background. First, for a video frame $I(x,y)$, we exploit the canny edge detector to obtain its binary edge image $E(x,y)$, where 1 denotes edge pixels, and 0 denotes background. These edge pixels are aggregated into different contours c_i, and let $C = \{c_i | i = 1, 2, \ldots, L\}$, where each contour c_i is a maximum set of continuous edge pixels $e_k^{(i)}$, i.e., $c_i = \{e_k^{(i)} | k = 0, \ldots, t, \ldots, and\ e_t^{(i)}$ is eight-connected to $e_{t+1}^{(i)}\}$. In our implementation, contour c_i is naturally obtained by the edge tracking step of the Canny edge detection algorithm. Since the strokes of characters always have a range of width and height which depend on the size of characters, the length of edges associated with strokes cannot be too long. Let $w(c_i)$ and $h(c_i)$ denote the width and height of minimum enclosing rectangle (MER) of contour c_i, if

$$w(c_i) \leq \eta, h(c_i) \leq \tau \tag{1}$$

Contour c_i is retained as the candidate contour associated with strokes, and denoted by $\tilde{c}_i$, where η, τ are the thresholds and the choice of them depends on the size of characters to be detected. In the case of no knowledge about the text size, we take the loose strategy as $\eta = 41, \tau = 41$. Otherwise, contour c_i is removed as the non-stroke edges in complex background.

Additionally, the edge detection on a stroke always generates a pair of parallel edges in a specific direction, and the interval between them should not be too large, as shown in Fig. 2. In what follows, we describe how to detect the existence of parallel edges within a specific range for each contour $\tilde{c}_i$. Let $\tilde{C}$ denote the set of all the candidate contours $\tilde{c}_i$. For each edge pixel $e_k^{(i)} \in \tilde{c}_i$, there is a gradient direction $\theta_k^{(i)}$ for it, and we quantize $\theta_k^{(i)}$ into four values, $\theta_k^{(i)} \in \{0,1,2,3\}$, corresponding to the horizontal, vice-diagonal,vertical, main diagonal directions, respectively, as illustrated in Fig. 3. Apparently, pixels on two parallel lines or correspondences on parallel curves should have the same gradient direction. Thus, each contour $\tilde{c}_i$ is broken down into more sub-contours $\tilde{c}_{i,j}$ based on both connectivity of edge pixels and gradient directions, so that each $\tilde{c}_{i,j}$ has the similar gradient direction, i.e., the same value of $\theta_k^{(i)}$. As shown in Fig. 4(a), the closed contour for character "K" is decomposed into 13 subcontours. Let $\hat{C}$ denote the set of all these subcontours, and $\theta^{(i,j)}$ denote the gradient direction of each edge pixel belonging to $\tilde{c}_{i,j}$, and we can partition the subcontour set $\hat{C}$ into four sets $\hat{C}_s = \{\tilde{c}_{i,j} | \theta^{(i,j)} = s, i = 1,2,\ldots,\tilde{l}, j = 1,2,\ldots,m_i\}, s \in \{0,1,2,3\}$, i.e., $\hat{C} = \bigcup_{s=0}^{3} \hat{C}_s$. To keep the notation simple, in what follows, we use $c_k^{(s)}, k = 1,2,\ldots,n_k$, to denote a subcontour in $\hat{C}_s$. For each subcontour $c_k^{(s)}$, we adaptively construct a detection window, as shown in Fig. 5, and count the number of edge pixels on other subcontours with the same direction values as $\theta^{(i,j)}$ to judge whether $c_k^{(s)}$ has a corresponding parallel edge within a specific range. To obtain the detection window for subcontour $c_k^{(s)}$, we first compute its MER, and then dilate the MER according to different strategies based on $\theta^{(i,j)}$. Concretely, let $(x_0, y_0), (x_1, y_1)$ denote the coordinates of top-left vertex and bottom-right vertex of an MER, respectively, if the gradient direction of the subcontour enclosed by the MER is horizontal, the coordinates of top-left vertex and bottom-right vertex of the corresponding detection window are $(x_0, y_0 - \omega), (x_1, y_1 + \omega)$; otherwise, the detection window is defined by vertex coordinates $(x_0 - \omega, y_0)$ and $(x_1 + \omega, y_1)$, as shown in Fig. 5, where parameter ω depends on the width of strokes and reflects the searching range of parallel edge pixels. Based on subcontour set $\hat{C}_s$, we easily obtain the corresponding edge image $\hat{E}_s$ which consists of edge pixels belonging to $c_k^{(s)}$. Once the detection window for subcontour $c_k^{(s)}$ is determined, we can efficiently compute the number of edge pixels, $v_{s,k}$, in the detection window exploiting the integral image of $\hat{E}_s$. Supposing that $q_{s,k} = |c_k^{(s)}|$ denotes the number of pixels belonging to $c_k^{(s)}$, if and only if the detection window for $c_k^{(s)}$ contains enough edge pixels with the same gradient direction, i.e.,

$$\frac{v_{s,k} - q_{s,k}}{q_{s,k}} \geq \rho \tag{2}$$

the subcontour $c_k^{(s)}$ is classified as stroke-like edge, as shown in Fig. 4(b), where ρ is a pre-defined threshold ($\rho = 0.3$ in our experiment). In formula (3), $v_{s,k} - q_{s,k}$ computes the number of edge pixels with the same gradient direction as $c_k^{(s)}$ on other subcontours. Here we assume all such edge pixels come from a same parallel edge in the detection window. The parameter ω ensures the interval between these edge pixels and contour $c_k^{(s)}$ falls in the appropriate range. In conclusion, we define stroke-like edges as the edges with the following two properties: 1) the length of a contour should not be too long due to the limited size of characters in videos, 2) a pair of parallel edges should exist, which lie within a specific interval depending on the width of character strokes. If a contour does not satisfy the above properties, it will be removed as a non-stroke edge in complex background. Fig. 6 gives such an example, where the stroke-like edges detected by our stroke-like edge detector are highlighted with the red rectangle, and the non-stroke edges in complex background are highlighted with the green ovals.

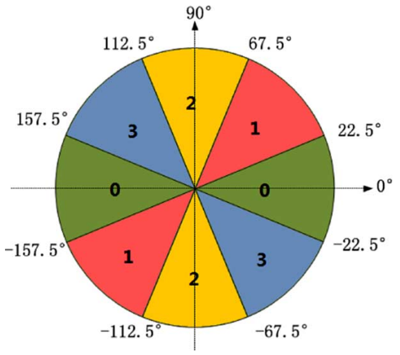

Fig. 3. Quantize $\theta_k^{(i)}$ into four values.

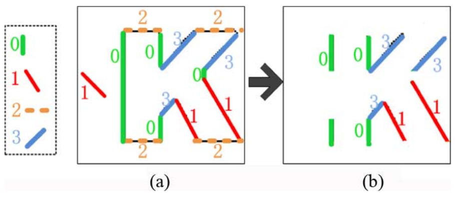

Fig. 4. Contours, subcontours, and stroke-like edges (a) Closed contour for character "K" is decomposed into 13 subcontours. (b) Stroke-like edges are detected by the proposed method.

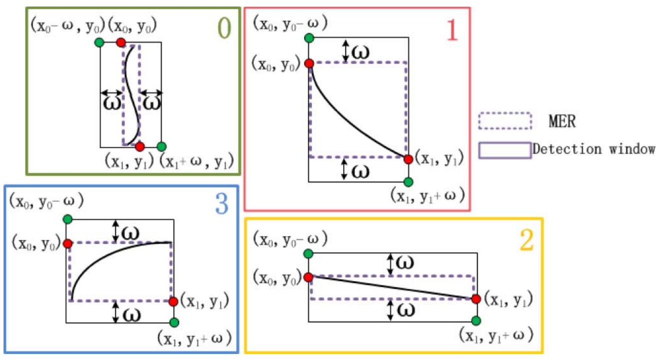

Fig. 5. Detection windows in four directions.

B. Spatial Localization

The task of spatial localization is to locate the caption regions. Many effective methods for text localization are based on the distribution of edges and corners detected in an image. Our work utilizes the stroke-like edge pixels detected from the previous step. This enable us to produce more accurate results based on the distributions of the obtained stroke-like edges. Concretely, a video frame is uniformly divided into $W \times H$ blocks, as shown in Fig. 7(a). The values of W and H depend on the size of video

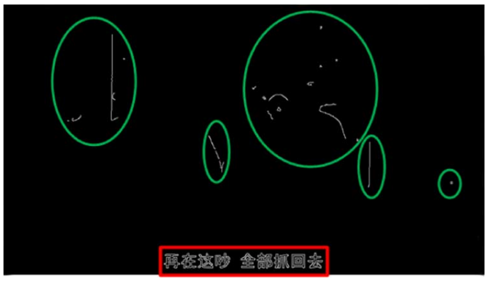

Fig. 6. Stroke-like edges versus non-stroke edges in complex background.

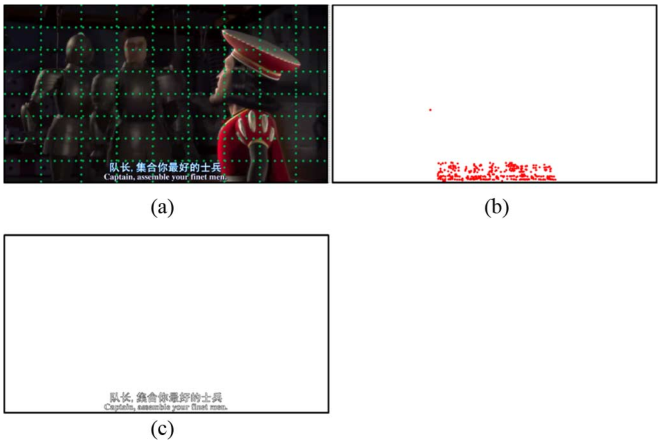

Fig. 7. Spatial localization. (a) Original frame is divided into 64 blocks by green grid. (b) Localization by the corners distribution. (c) Localization by the stroke-like edges distribution.

frame. We choose $W = 8$, $H = 8$ in our system. Let N_x^i and $N_e^i (i = 1, 2, \ldots, W \times H)$ denote the number of corners and stroke-like edge pixels in the ith block, respectively. We adopt the method presented by [24] to detect corner pixels. Concretely, we consider an m by m neighborhood $\vartheta(p)$ of pixel p in video frame I, and calculate its gradient correlation matrix Θ_p by

$$\Theta_p = \begin{bmatrix} \sum_{\vartheta(p)} \left(\frac{dI}{dx}\right)^2 & \sum_{\vartheta(p)} \left(\frac{dI}{dx} \cdot \frac{dI}{dy}\right) \\ \sum_{\vartheta(p)} \left(\frac{dI}{dx} \cdot \frac{dI}{dy}\right) & \sum_{\vartheta(p)} \left(\frac{dI}{dx}\right)^2 \end{bmatrix}. \quad (3)$$

Let ϖ_p denote the minimum eigenvalue of matrix Θ_p. If $\varpi_p > \pi$, where $\pi = 0.05 * \max_p\{\varpi_p\}$, the pixel p is declared as a corner. We choose $m = 3$ in computing Θ_p. If the ith pixel block contains enough stroke-like edge pixels and corner pixels, i.e.,

$$N_x^i > T_x, N_e^i > T_e \quad (4)$$

it is marked as a text block, where T_x, T_e are thresholds. The choice of T_x and T_e depends on the total number of corners N_x and stroke-like edge pixels N_e in the frame, and $T_x = 0.003 * N_x$, $T_e = 0.1 * N_e$ in our system. The incorporation of text blocks is chosen as candidate caption regions.

C. Temporal Localization

Temporal localization aims to find the boundaries of consecutive different captions, so that a video can be decomposed into a sequence of clips, where all the frames in a clip share a same caption, or no caption. Since the characters in captions are composed of strokes, the change of captions makes the change of the distribution of stroke-like edges. Compared with Liu *et al.* [25], where temporal segmentation is performed by comparing the histograms of the gradient directions of edge pixels, our approach instead exploits the more robust stroke-like edge feature, and design a more accurate comparison measurement. For two consecutive sampled frames $I_t(x, y)$ and $I_{t+1}(x, y)$, let $E_t(x, y)$ and $E_{t+1}(x, y)$ denote the corresponding binary stroke-like edge images, where stroke-like edge pixels are denoted by 1 and other pixels by 0, and $E_t^{(s)}(x, y)$ and $E_{t+1}^{(s)}(x, y), s = 0, 1, 2, 3$, denote the corresponding binary stroke-like edge images with quantized gradient direction s defined in Section II-A, so $E_t(x, y) = \bigcup_{s=0}^{3} E_t^{(s)}(x, y)$. Then we compute how many stroke-like pixels keep unchanged in terms of both spatial location and gradient direction by

$$S_{t,t+1}(x, y) = \sum_{s=0}^{3} \left(E_t^{(s)}(x, y) \wedge E_{t+1}^{(s)}(x, y)\right) \quad (5)$$

$$\varphi_{t,t+1} = \sum_{(x,y) \in R} S_{t,t+1}(x, y) \quad (6)$$

where R is the candidate caption region spatially located in frame $I_t(x, y)$, and $\wedge$ denotes logic "and" operation. Further, the ratio $\gamma_{t,t+1}$ of unchanged stroke-like pixels to the total stroke-like pixels in frame $I_t(x, y)$ and $I_{t+1}(x, y)$ can be evaluated by

$$\gamma_{t,t+1} = \frac{\varphi_{t,t+1}}{\sum_{(x,y) \in R} E_t(x, y) \vee E_{t+1}(x, y)} \quad (7)$$

where $\vee$ denotes logic "or" operation. If almost all the stroke-like edge pixels in caption candidate region R of $I_t(x, y)$ keep unchanged in terms of both spatial location and gradient direction, i.e., $\gamma_{t,t+1} \geq \xi$, frame $I_{t+1}(x, y)$ should contain the same caption as frame $I_t(x, y)$. Otherwise, our system will consider a new caption appears, or the current caption disappears and no caption at frame $I_{t+1}(x, y)$, and frame $I_t(x, y)$ is the last one in the current clip. To guarantee the accuracy of our system, we choose the threshold $\xi = 0.92$ to follow the loose strategy. The following caption segmentation is based on the results of temporal localization.

D. Localization Post-Processing

It is significant to identify whether a clip generated by temporal localization contains captions, since only the clips containing captions are meaningful for subsequent computation. In our system, if frame $I_t(x, y)$ does not contain enough stroke-like edge pixels, i.e., $\sum_{(x,y)} E_t(x, y) \leq \lambda$, where λ is a threshold, frame $I_t(x, y)$ is regarded containing no caption. If a group of consecutive frames $I_t(x, y), I_{t+1}(x, y), \ldots, I_{t+K-1}(x, y)$ does not contain captions, but frame $I_{t+K}(x, y)$ contains captions, frames $I_t(x, y), I_{t+1}(x, y), \ldots, I_{t+K-1}(x, y)$ will form a clip with no caption.

Spatial localization in Section II-B provides only a rough estimation of caption regions, as block-aligned. After temporal localization, spatial and temporal information can be integrated to provide a more accurate estimation of caption regions. Assuming that a clip generated by temporal localization contains captions and it is composed of K frames, i.e., $I_t(x,y), I_{t+1}(x,y), \ldots, I_{t+K-1}(x,y)$, we first identify the reliable stroke-like edge pixels by checking how many times such a pixel is consecutively detected, i.e.,

$$\Gamma(x,y) = \frac{\sum_{t'=t}^{t+K-2} S_{t',t'+1}(x,y)}{K-1} \tag{8}$$

where $S_{t',t'+1}(x,y)$ is defined by (5). If the stroke-like edge pixel at location (x,y) is often consecutively detected, i.e., $\Gamma(x,y) \geq \mu, \mu \in [0,1]$, the pixel is then regarded as a reliable stroke-like edge pixel. In our system, $\mu = 0.65$. The coordinates of all the reliable stroke-like edge pixels form a set Ω. Let R_{mer} denote the minimum enclosing rectangle which covers all the coordinates in Ω, w_{mer} and h_{mer} denote the width and height of R_{mer}, and $(x_0, y_0), (x_1, y_1)$ denote the coordinates of top-left vertex and bottom-right vertex of R_{mer}, we can obtain a finer location of caption regions by appropriately dilating R_{mer} horizontally and vertically. The final caption region R_{final} is defined by the coordinates of top-left vertex and bottom-right vertex, i.e., $(x_0 - \Delta_x, y_0 - \Delta_y), (x_1 + \Delta_x, y_1 + \Delta_y)$, where $\Delta_x = min\{6, w_{mer}/40\}$ and $\Delta_y = min\{4, h_{mer}/10\}$ in our system.

E. Caption Segmentation

The aim of caption segmentation is to further identify text pixels from pixels in caption region located. Our system adopts a simple and efficient color-based method to extract candidate text pixels. For a clip containing a same caption, we assume that the color of captions is very similar both within a frame and over different frames. In general, in the candidate text regions, the number of character pixels is usually larger than that of pixels with another color, so we estimate the caption color by finding the largest frequency bin in the color histogram corresponding to the caption region localized previously. In practice, we partition the whole RGB color space into 64 cubes, $h_i, i = 1, 2, \ldots, 64$, by uniformly quantizing each channel into four bins. For a caption clip, at most five frames are randomly chosen as an observation sample, and its 64-bin color histogram $|h_i|$, is computed. The index k of the color cube with the largest number of pixels is identified by $k = arg\max_{i=1}^{64}\{|h_i|\}$, and the kth color cube is chosen as the initial estimation of caption color range. Since the text color can lie on or near the border of two neighboring bins, i.e., text pixels can be put into multiple neighboring bins, which may affect the segmentation accuracy, we need to adjust the initial estimation. Concretely, the average color of the pixels in cube h_k is computed by $\kappa = 1/|h_k| \sum_{p \in h_k} r(p)$, where $r(p)$ denotes the color value of pixel p. Then, a new cube $\bar{h}$ is constructed with center κ and the same size as h_k. For any pixel in the caption regions of previously chosen frames, if its color value belongs to the cube $\bar{h}$, the pixel is labeled as candidate text pixel and the remaining pixels are classified as background pixels. As a result, a binary image $B_t(x,y)$ is obtained, where 1 denotes candidate caption pixels and 0 denotes background pixels.

Since some background pixels can also have the similar color, some false stroke areas or character pixels are possible to appear in the output binary images, which will degrade the recognition rate of OCR. So we further exploit the temporal homogeneity of color of text pixels to filter out some background pixels with similar color. Let $B_{t_1}(x,y), B_{t_2}(x,y), \ldots, B_{t_n}(x,y)$ denote the binary segmentation results of the previous color-based method for the randomly chosen frames in a clip, and non-text pixels with the similar color can be removed though applying the logic "and" operation at each pixel location on $B_{t_1}(x,y), B_{t_2}(x,y), \ldots, B_{t_n}(x,y)$, i.e.,

$$\hat{B}(x,y) = B_{t_1}(x,y) \wedge B_{t_2}(x,y) \wedge \cdots \wedge B_{t_n}(x,y) \tag{9}$$

where $\hat{B}(x,y)$ is the segmentation result by integrating multiple frames in a clip. Fig. 8 shows an example of caption segmentation. The clip contains four consecutive frames totally. Fig. 8(a) shows one of them. Their corresponding binary images after color-based segmentation are shown in Fig. 8(b)–(e). Applying the temporal "and" operation, we obtain the final result in Fig. 8(f) where the pixels with similar color in complex background are removed.

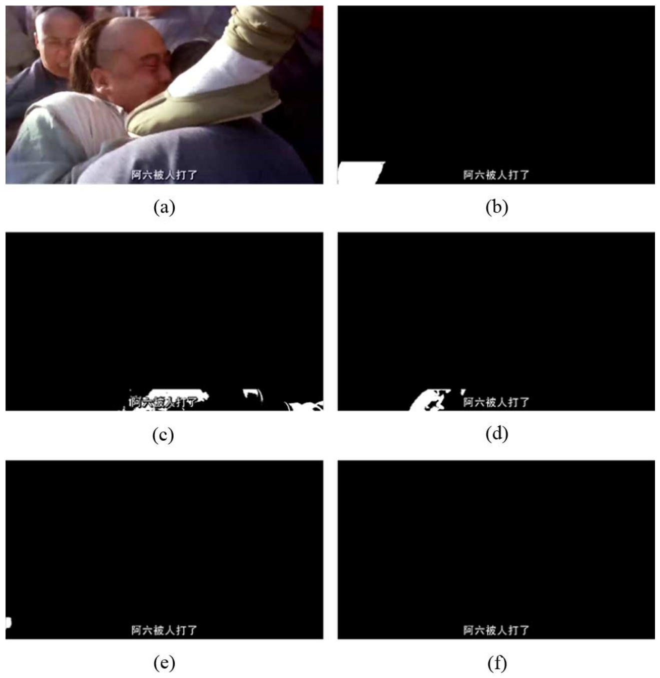

Fig. 8. Caption segmentation. (a) Original frame in a clip. (b)–(e) Segmentation result of each frame in a clip. (f) Result after applying the temporal "and" operation.

F. Refining Segmentation Results

As the segmentation result, the binary image $\hat{B}(x,y)$ can still include some background pixels, or lose some true text pixels. Some post-processing techniques based on our *a priori* knowledge on stroke structure of characters can help us to further refine the segmentation result to some extent. Our observation shows that it is true for Chinese characters and English characters that the gaps exist among strokes and between adjacent characters. So we can remove some background pixels through checking whether a local window contains too many text pixels. To do this, a sliding window is used, which is sufficiently large to cover different strokes or characters. In our system, we choose an 8 by 8 window. At each sliding position in caption regions, if the number of text pixels in it exceeds a predefined threshold, it

TABLE I
DATASET DESCRIPTION

Dataset	Dataset I	Dataset II
Video type	movies	news, sports, talk shows entertainment shows
No. of videos	25	46
Duration(frames)	29213	67921
Resolution	512*384, ... ,608*344, 640*336,...,720*404 800*432, ..., 1024*576	288*208,320*240 384*288,400*300 480*320,480*360

means few gaps between characters or strokes, so all the pixels in the window are labeled as background and the sliding window moves forward 8 pixels. Otherwise, the sliding window will move forward only with a smaller pace of three pixels. In the segmentation of text pixels described above, each pixel is classified independently, but in practice the strokes of a character are always continuous. So the morphological dilation operation is performed to link broken strokes by filling missing pixels.

III. Experimental Results

In our evaluation experiments, two representative and comprehensive datasets are used. The related information of the datasets are summarized in Table I, where duration is measured in terms of the number of frames after 3 frames per second subsampling. The first dataset (called dataset I) consists of 25 movie videos with diverse and complex background. These videos have 17 different kinds of resolutions, and different time durations. The dataset I involves Chinese characters, English words, punctuation symbols, and digits, in which 56% are Chinese characters. To our knowledge, there is no public video dataset available for video text detection and extraction, so most researchers do their experiments on some news videos or images collected from the Internet by themselves. To demonstrate that the proposed system can also handle other video types, we set up dataset II which includes 46 videos covering news, sports, talk shows and entertainment shows, where captions lie on different locations in a frame, and more font styles and sizes are involved. After each video is subsampled by three frames per second, we obtain 67 921 frame images totally. In these frames of dataset II, the smaller size and diverse locations of characters, as well as low resolutions, make it become very challenging.

A. Temporal Localization Performance

Firstly, we evaluate the performance of temporal localization, since the localization results will directly influence the performance of segmentation outcomes as the basis of the following computation. Two metrics are used in the evaluation, i.e., precision $P_{tem} = N_c/N_d$ and recall $R_{tem} = N_c/N_g$, where N_c denotes the number of boundaries correctly detected, N_d denotes the number of boundaries output by our system, and N_g denotes the number of ground-truth boundaries labeled manually. The ground truth boundaries include three cases: 1) from a caption to a different caption; 2) from no caption to a caption; 3) from a caption to no caption. Since labeling the boundaries is a tedious and time-consuming work, we evaluate the performance only on the longest video, which contains 12 888 frames with 1024*576 resolution, and involves Chinese, punctuation symbols and digits. The experimental results are summarized in Table II. We compare the proposed method with two temporal localization methods [23], [25] . Tang *et al.* [23] exploited the quantized spatial difference density (QSDD) to detect the caption transition frames while [25] was based on comparing the histogram of edge gradient directions. In [23], shot boundaries were first detected to decompose a video into a sequence of shots and then temporal segmentation computation was carried out on each shot. Thus, their method cannot process the captions crossing shot boundaries. In Table II, the experimental results about [23] are obtained based on two facts: 1) we assume that the system of Tang *et al.* [23] can detect shot boundaries perfectly (no missing and no errors); 2) all captions crossing shot boundaries are assumed to be detected perfectly. In the comparison experiments on the same dataset, the proposed approach has shown more excellent performance than the other two methods, with the increase of 8.6%, 8.5% in P_{tem} and 0.53%, 7% in R_{tem}, respectively. We attribute the performance increase to the usage of the proposed stroke-like edge detector, and the better similarity function.

TABLE II
PERFORMANCE COMPARISON OF OUR TEMPORAL LOCALIZATION METHOD, THE METHOD OF [23], AND THE METHOD OF [25]

Method	N_g	N_d	N_c	P_{tem}	R_{tem}
Method [23]	1660	1730	1572	0.909	0.947
Method [25]	1660	1695	1543	0.910	0.930
Our method	1660	1669	1660	0.995	1.000

B. Spatial Localization Performance

In order to show the adaptivity of our system, we randomly sample 100 different clips produced by temporal segmentation (50 clips from dataset I and 50 clips from dataset II), so each clip contains the same caption or no caption. For each clip, the first sampled frame is chosen and the text box is manually labeled as the ground truth to evaluate the proposed text localization method. A correct localization rectangle is counted if and only if the intersection of a located text rectangle (LTR)and a ground-truth text rectangle (GTR) covers at least 90% of their union. We also use two metrics: recall $R_{loc} = N_{rc}/N_{tc}$ and precision $P_{loc} = N_{rc}/N_{gc}$ in the evaluation, where N_{rc} denotes the number of text rectangles located correctly, N_{gc} denotes the number of GTRs, and N_{tc} denotes the total number of rectangles located by the proposed system. As shown in Table III, the recall and the precision of localization are both above 95%. Since some test samples have low resolutions and very small characters, the outcome is very satisfactory. Some experimental results are shown in Fig. 9. The blue rectangles shows the localization results of the proposed method, and the red ovals are the missing text region. There are some sticky cases that our system cannot deal with well in the experiments, such as extremely deformed characters, blurring characters owing to small size or high similarity with background where even human eyes are difficult to recognize sometimes, as shown in Fig. 9(g)–(i).

C. Evaluation of Segmentation

We evaluate our segmentation performance based on two groups of experiments. One group of experiments are based on

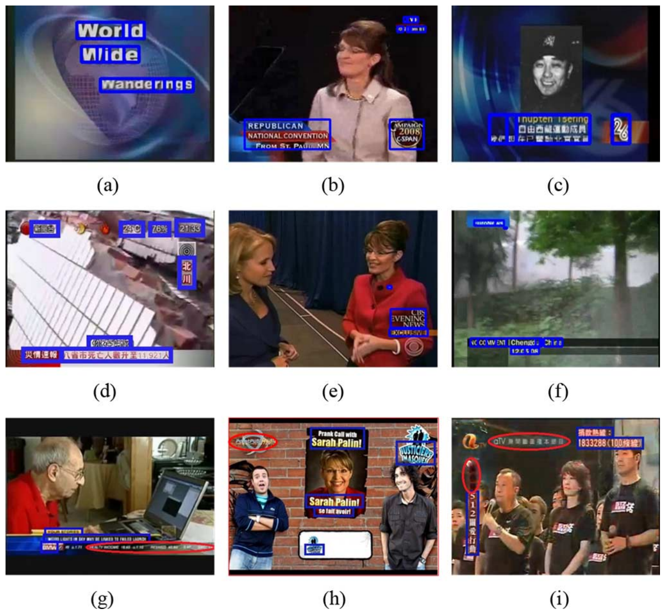

Fig. 9. Text spatial localization. (a)–(f) are the correct localization results. (g)–(i) show the imperfect results due to low resolution or too small characters.

TABLE III
SPATIAL LOCALIZATION AND SEGMENTATION RESULTS

	R	P
Location	$R_{loc} = 0.952$	$P_{loc} = 0.968$
Segmentation	$R_{seg} = 0.871$	$P_{seg} = 0.857$

TABLE IV
RECOGNITION RESULTS

	N_t	N_r	Q	Q'
Chinese characters	5152	5011	**0.973**	0.939
English Letter	4279	3982	**0.931**	0.928
Punc.Symb./Digits	651	608	**0.934**	0.920
Sum	10082	9601	**0.952**	0.930

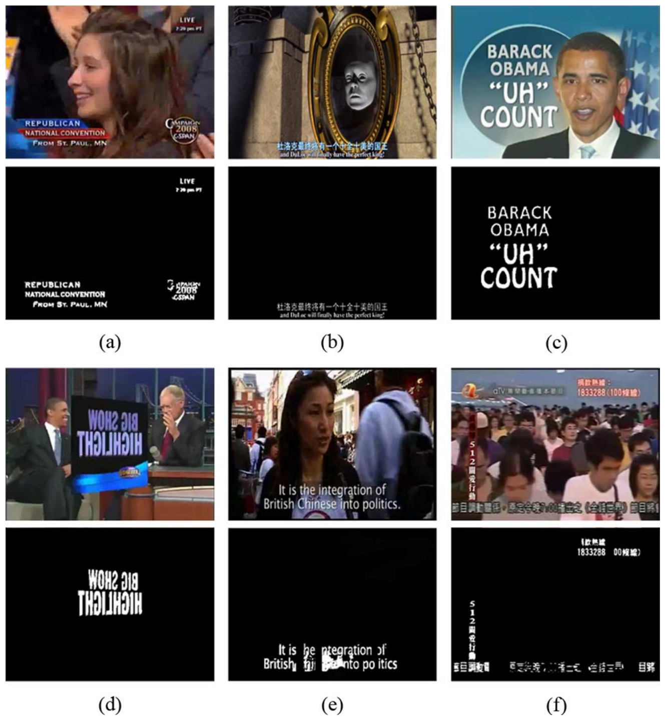

Fig. 10. Results of caption extraction. (a)–(d) are the correct segmentation results. (e) and (f) show the imperfect segmentation results.

the character recognition rate obtained by using the commercial Hanwang OCR software to recognize the binary images generated by our caption extraction approach. The character recognition rate Q is defined as the ratio of the number of characters correctly recognized N_r to the total number of input characters N_t, i.e., $Q = N_r/N_t$. In the experiments, we random sample the segmented captions in dataset I and feed them into the OCR module. The corresponding experimental results are summarized in Table IV. According to Table IV, the recognition rate of Chinese character is higher than that of English and Punc.Symb./Digits because of high similarities between English letters, such as "v" and "y", as well as the lower resolution of English letters in our dataset. Further, to check the gain of using the temporal "and" operation in our segmentation method, we conduct a separate experiment to compare the recognition rate Q after applying the temporal "and" operation with that of only color-based method, Q', as shown in the fifth column of Table IV. The experiment results show that the temporal "and" operation can bring 2.2% gain to the recognition rate, and even more gain can be obtained for Chinese characters.

The other group of experiments are based on the ground truth manually obtained at the pixel level. We randomly choose 110 different frames from dataset II with more challenges, and manually labeled text pixels in them as ground truth. In the same way, two metrics R_{seg} and P_{seg} are used in the evaluation, i.e., $R_{seg} = N_{rp}/N_{gp}$, $P_{seg} = N_{rp}/N_{tp}$, where N_{rp} denotes the number of text pixels segmented correctly, N_{gp} denotes the number of text pixels in ground truth and N_{tp} denotes the total number of text pixels segmented by the system. The third row in Table III gives the corresponding results of segmentation experiments. Also, some typical segmentation examples are shown in Fig. 10, which involves different types of videos both in dataset I and dataset II, different resolutions and character sets. In the evaluation, the misclassification of pixels mainly come from two aspects of reasons: 1) failure or inaccuracy of localization makes entire or partial text regions are missing before segmentation, as shown in Fig. 10(f), where the caption in the left-top region is not localized, i.e., completely missing; 2) to obtain the high efficiency in the segmentation computation, we only exploit the color information, which makes some background pixels with very similar color are misclassified as text pixels, as shown in Fig. 10(e).

D. Evaluation of Computation Efficiency

We evaluate the computation efficiency of our system on dataset II. The related experiments are performed on a DELL desktop with an Intel Core 2.83 GHz CPU and 4G memories, and the experimental results are listed in Table V. As shown in Table V, the average processing time of each frame in our system is 214.06 milliseconds. Since the system samples 3 frames per second, it means that the proposed approach is able to extract captions from videos at the real time speed. The dataset II is chosen in the evaluation, because caption locations in it vary greatly, compared with those in dataset I, which

TABLE V
COMPUTING COST OF THE PROPOSED APPROACH ON DATASET II

Number of videos	46
The total duration of videos	6h 17m 20s
location and segmentation time	4h 2m 19s
Average running time	214.06ms/frame

mainly lie at the bottom of frames. That means that the system has to execute the localization computation on entire frames. It can be expected that the proposed approach can achieve more high computation efficiency, if some priori knowledge about videos can be used, such as dataset I.

IV. CONCLUSION

In this paper, we propose an efficient and effective caption extraction approach for videos, in which a set of elementary steps are sequentially performed, and the spatio-temporal information are utilized throughout the entire approach (localization, segmentation and post-processing). We present a stroke-like edge detector based on contours, which can effectively remove non-caption edges introduced from complex background and greatly benefit the follow-up spatial and temporal localization processes. The experimental results on two representative datasets have verified the effectiveness and efficiency of the proposed approach.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable suggestions.

REFERENCES

[1] S. H. Park, K. I. Kim, K. Jung, and H. J. Kim, “Locating car license plates using neural networks,” *Electron. Lett.*, vol. 35, no. 17, pp. 1475–1477, 1999.
[2] W. Wu, X. Chen, and J. Yang, “Detection of text on road signs from video,” *Intell. Transp. Syst.*, vol. 6, no. 4, pp. 378–390, 2005.
[3] J. C. Shimy, C. Doraiz, and R. Bollez, “Automatic text extraction from video for content-based annotation and retrieval,” in *Proc. Int. Conf. Pattern Recognition*, Australia, Aug. 16–20, 1998, pp. 618–620.
[4] N. Ezaki, M. Bulacu, and L. Schomaker, “Text detection from natural scene images: Towards a system for visually impaired persons,” in *Proc. Int. Conf. Pattern Recognition*, Cambridge, U.K., Aug. 23–26, 2004, pp. 683–686.
[5] Y. Liu, H. Lu, X. Y. Xue, and Y. P. Tan, “Effective video text detection using line features,” in *Proc. Int. Conf. Control, Automation, Robotics and Vision*, 2004, pp. 1528–1532.
[6] P. Dubey, “Edge based text detection for multi-purpose application,” in *Proc. Int. Conf. Signal Processing*, Beijing, China, Nov. 16–20, 2006.
[7] M. Cai, J. Song, and M. R. Lyu, “A new approach for video text detection,” in *Proc. Int. Conf. Image Processing*, Rochester, NY, Sep. 22–25, 2002, pp. 117–120.
[8] P. Shivakumara, T. Q. Phan, and C. L. Tan, “Video text detection based on filters and edge features,” in *Proc. IEEE Int. Conf. Multimedia and Expo*, New York, Jun. 3, 2009, pp. 514–517.
[9] X. S. Hua, X. R. Chen, W. Y. Liu, and H. J. Zhong, “Automatic location of text in video frames,” in *Proc. ACM Workshop Multimedia: Information Retrieval*, Ottawa, ON, Canada, Sep. 5, 2001.
[10] I. Ar and M. E. Karsligli, “Text area detection in digital documents images using textural features,” in *Proc. Int. Conf. Computer Analysis of Images and Patterns*, Vienna, Austria, Aug. 27–29, 2007.
[11] K. I. Kim, K. Jung, and J. H. Kim, “Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm,” *Pattern Anal. Mach. Intell.*, vol. 25, no. 12, pp. 1631–1639, 2003.
[12] C. Garcia and X. Apostolidis, “Text detection and segmentation in complex color images,” in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Istanbul, Turkey, 2000, pp. 2326–2329.
[13] A. K. Jain and S. Bhattacharjee, “Text segmentation using Gabor filters for automatic document processing,” *Mach, Vis. Appl,*, vol. 5, no. 3, pp. 169–184, 1992.
[14] W. Mao, F. Chung, K. K. M. Lam, and W. Siu, “Hybrid Chinese/English text detection in images and video frames,” in *Proc. Int. Conf. Pattern Recognition*, Quebec City, QC, Canada, Aug. 11–15, 2002.
[15] Q. Ye, Q. Huang, W. Gao, and D. Zhao, “Fast and robust text detection in images and video frames,” *Image Vis. Comput.*, vol. 23, no. 6, pp. 565–576, 2005.
[16] K. I. Kim, K. Jung, S. H. Park, and H. J. Kim, “Support vector machine-based text detection in digital video,” *Pattern Recognit.*, vol. 34, no. 2, pp. 527–529, 2001.
[17] Q. Liu, C. Jung, and Y. Moon, “Text segmentation based on stroke filter,” in *Proc. ACM Int. Conf. Multimedia*, Santa Barbara, CA, Oct. 23–27, 2006, pp. 129–132.
[18] Q. Liu, C. Jung, S. Kim, Y. Moon, and J. Kim, “Stroke filter for text localization in video images,” in *Proc. IEEE Int. Conf. Image Processing*, Atlanta, GA, Oct. 8–11, 2006, pp. 1473–1476.
[19] B. Epshtein, E. Ofek, and Y. Wexler, “Detecting text in natural scenes with stroke width transform,” in *Proc. Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, Jun. 13–18, 2010, pp. 2963–2970.
[20] K. Jung, K. I. Kim, and A. K. Jain, “Text information extraction in images and video: A survey,” *Pattern Recognit.*, vol. 37, no. 5, pp. 977–997, 2004.
[21] H. Hase, T. Shinokawa, M. Yoneda, and C. Y. Suen, “Character string extraction from color documents,” *Pattern Recognit.*, vol. 34, no. 7, pp. 1349–1365, 2001.
[22] J. Gllavata, R. Ewerth, T. Stefi, and B. Freisleben, “Unsupervised text segmentation using color and wavelet features,” in *Proc. Int. Conf. Image and Video Retrieval*, Dublin, Ireland, Jul. 21–23, 2004, pp. 216–224.
[23] X. Tang, X. Gao, J. Liu, and H. J. Zhang, “A spatial-temporal approach for video caption detection and recognition,” *IEEE Trans. Neural Netw.*, vol. 13, no. 4, pp. 961–971, 2002.
[24] J. Shi and C. Tomasi, “Good features to track,” in *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition*, Seattle, WA, Jun. 21–23, 1994, pp. 593–600.
[25] X. Q. Liu, W. Q. Wang, and T. S. Zhu, “Extracting captions in complex background from videos,” in *Proc. Int. Conf. Pattern Recognition*, Istanbul, Turkey, Aug. 23–26, 2010, pp. 3232–3235.

Xiaoqian Liu (M’10) received the B.E. degree from Beijing Forestry University, Beijing, China, in 2008 and she is currently pursuing the Ph.D. degree in the Graduate University of Chinese Academy of Sciences, Beijing.

Her research interests include multimedia technology, pattern recognition, and text extraction in images and videos.

Weiqiang Wang (M’04) received the B.E. and M.E. degrees in computer science from Harbin Engineering University, Harbin, China, in 1995 and 1998, respectively, and the Ph.D. degree in computer science from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, China, in 2001.

He is currently a Professor with the Graduate School, CAS, and a Guest Researcher in ICT, CAS. His research interests include multimedia content analysis and computer vision.