2013 IEEE International Symposium on Multimedia

A Video Text Detection and Tracking System

Tuoerhongjiang Yusufu[1],Yiqing Wang[1], Xiangzhong Fang[1]

[1] *Dep. of Electronic Engineering, Shanghai Jiao Tong University*
Shanghai, China
turgun@sjtu.org

Abstract— **Faced with the increasing large scale video databases, retrieving videos quickly and efficiently has become a crucial problem. Video text, which carries high level semantic information, is a type of important source that is useful for this task. In this paper, we introduce a video text detecting and tracking approach. By these methods we can obtain clear binary text images, and these text images can be processed by OCR (Optical Character Recognition) software directly. Our approach including two parts, one is stroke-model based video text detection and localization method, the other is SURF (Speeded Up Robust Features) based text region tracking method. In our detection and localization approach, we use stroke model and morphological operation to roughly identify candidate text regions. Combine stroke-map and edge response to localize text lines in each candidate text regions. Several heuristics and SVM (Support Vector Machine) used to verifying text blocks. The core part of our text tracking method is fast approximate nearest-neighbour search algorithm for extracted SURF features. Text-ending frame is determined based on SURF feature point numbers, while, text motion estimation is based on correct matches in adjacent frames. Experimental result on large number of different video clips shows that our approach can effectively detect and track both static texts and scrolling texts.**

***Keywords**— Stroke-model; Text Detection; Text Tracking; SURF*

I. Introduction

With the development of internet, more video databases are available online. In such databases, there are constantly growing huge number of videos. As example, the total number of YouTube videos over 120 million, it is a pressing task to develop a fast and efficient method to retrieve these resources. Traditionally, manual annotations have been used for this purpose. An alternative approach is to automatically annotate the videos with the text that appears in the frames, and video can be retrieved by these texts. Video text recognition is crucial to the research in all video indexing and summarization.

Fig.1 shows the main procedure of video text recognition system. The whole procedure is mainly divided into five steps: text detection, localization, tracking, extraction and recognition. In the text detection step, text regions and non-text regions in a video frame is roughly identified. Accurate boundaries of text rows are determined in the localization step. The text tracking step is performed to locate specific text information across video frames and enhance text segmentation and recognition over time. The text extraction step removes background pixels in the text rows and the text pixels are left for the recognition. The text recognition step executed by the OCR software.

Although much works has been done on text detection and localization, which mainly focus on handling single video frame or equal interval frames, little attention has been paid to video text tracking. A text tracking step is crucial for a real time text detection, extraction and recognition system as following reasons:

1) Generally, tracking is faster than detection and localization steps. Although there are some fast text detection approaches, most of the published algorithms about text detection are time consuming.
2) All video frames will be processed by tracking step; as a result, theoretically we can detect some texts with very short duration. It would be fail to detect all text information the video contains, if we process equally interval frames or just process key frames. Because there are still a lot of text information in other frames.
3) Text pixels can be enhanced by tracking. As in most cases background is not stable as foreground text during throughout time, we can remove background by using bitmap/stroke integration over time.

In this paper we mainly discuss text detection, localization, tracking methods for video text. Using text tracking result background subtraction and text enhancement can be performed in a very simple way.

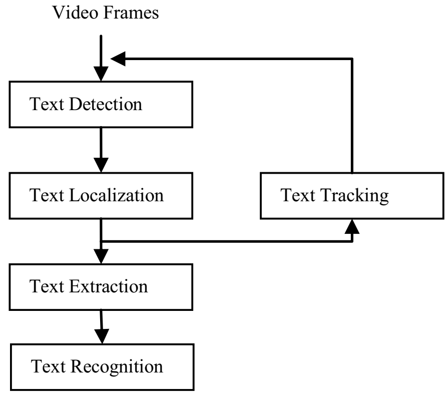


Fig. 1 Video text recognition system

Text detection and localization methods can be classified into three approaches [16]: connected component-based, edge-based, and texture-based. The connected component-based methods are proposed based on the fact that texts in one frame usually have similar color and satisfy certain size, shape and spatial alignment constraints[3][4]. These methods employ color quantization and region growing (or splitting) to group neighboring pixels of similar colors into connected components. However, these connected components may not preserve the full shape of the characters due to color bleeding

978-0-7695-5140-1/13 $26.00 © 2013 IEEE
DOI 10.1109/ISM.2013.106

and the low contrast of the text lines. Therefore, these methods do not work well for video images.

The edge-based methods focus on the rich edge information in text region, implying high contrast of edge between texts and the background [5][6][7]. The edges of texts are detected by an edge filter, and then merged by morphological operation in the spatial domain. The non-text regions are removed by text verification. Because background and non-text objects also contain strong edge information, this method produces many false positives for images with complex backgrounds.

The texture-base methods take into account the fact that text regions have special texture, distinguishing from the background [8][9]. These methods apply Fast Fourier Transform, Discrete Cosine transform, wavelet decomposition, and Gabor filter for feature extraction. These methods are usually treated as a classification problem. Text can be detected by machine learning classifier, such as K-means, neural networks and SVM. The texture-based methods are not effective when the background or the objects have similar structural texture to texts. Furthermore, the computational complexity is high because of heavy operations on the input image in the machine training and classification process.

Conventional text tracking methods are mainly based on block matching algorithm with different features. In pixel domain, Lienhart *et al.* [12] used the difference between two luminance frames. In another work [11], Lienhart and Wernickes also proposed a projection profile based text tracking method; the matching measure was based on the vertical and horizontal projection profile. Shi *et al.* [15] proposed a tracking algorithm based on sum of absolute difference of text blocks in consecutive frames. Yinan Na and Di Wen [14] proposed a text tracking scheme based on matching SIFT features by NN.

In the compressed domain, Qian *et al.* [13] measured the similarity by calculating the mean absolute difference (MAD) of direct current images. Cargi *et al.* [2] utilized motion similarity of the macro blocks in the text region to track the detected texts.

Although the key importance of text tracking in real time application, little work has been published on it. Moreover, most of the texts tracking algorithms are based on matching simple or global feature. In this paper, new text tracking method is proposed, which based on SURF (Speed-Up Robust Feature) and fast approximate nearest neighbor algorithm. It has better performance than SIFT feature in video text tracking. Prior to text tracking method, we also propose a new text detection and localization method.

The remaining parts of this paper are organized as follows. Section two introduces a new text detection and localization approach. Section three describes the text tracking algorithm in detail. Section four shows the experiment results. Section five present the conclusion and future work.

II. Text Detection and Localization

In our detection and localization approach, we use coarse to fine strategy, candidate text regions are quickly detected by using stroke-map and morphological operations. Then, stroke-map and Canny response are combined to determine boundaries of text rows. Finally, several heuristics and SVM applied to filter out incorrect text blocks.

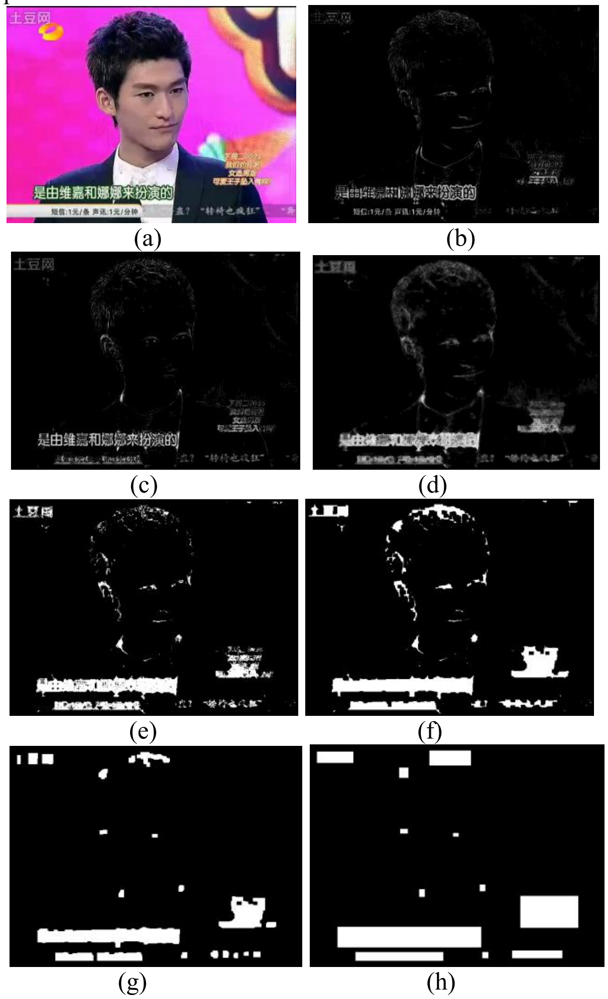


Fig. 2 Video text detection. (a) Original image. (b) Stroke-map of positive image. (c) Stroke-map of negative image. (d) Combined stroke-maps. (e) Binarization. (f) Close. (c) Open. (d) Merged bounding blocks.

A. Video Text detection

We utilize the stroke-map [1] to obtain the low level representation of image content. Stroke-map is proposed for document image text enhancement and extraction. It is based on stroke width restriction. In this algorithm, original image is transformed into double-edge image. Double-edge image is decomposed into positive and negative parts, which represents information and noise respectively. In a one-dimensional (1-D) profile, double-edge response calculated as:

$$DE(x) = Max_{i=1}^{W-1}\{Min(f(x-i), f(x+W-i))\} - f(x). \quad (1)$$

Where W is stroke width. For the strokes in two-dimensional space, double-edge is estimated as:

$$DE_W(p) = Max_{d=0}^{3}\{DE_{Wd}\} \quad (2)$$

Here, $d = 0,1,2,3$ refer to four direction, horizontal, vertical, left-diagonal, right-diagonal direction, that approximate most possible stoke direction.

Double-edge image then decomposed into positive and negative parts as:

$$DE_W = DE_W^+ + DE_W^- \tag{3}$$

Where

$$DE_W^+(x,y) = \begin{cases} DE_W(x,y), \ \mathit{if} \ DE_W(x,y) > 0 \\ 0, \qquad \mathit{otherwise} \end{cases} \tag{4}$$

$$DE_W^-(x,y) = \begin{cases} (-1) * DE_W(x,y), \ \mathit{if} \ DE_W(x,y) < 0 \\ 0, \qquad \mathit{otherwise} \end{cases} \tag{5}$$

Through the stroke map transformation each pixel in the original image can be classified into object or background according to the intensity of its local feature. Positive parts are seen as objects, while negative parts are can be seen as background or noise. Most outstanding feature of stroke-map algorithm is that, it is very fast and effective. It enables us to detect candidate text regions quickly and efficiently.

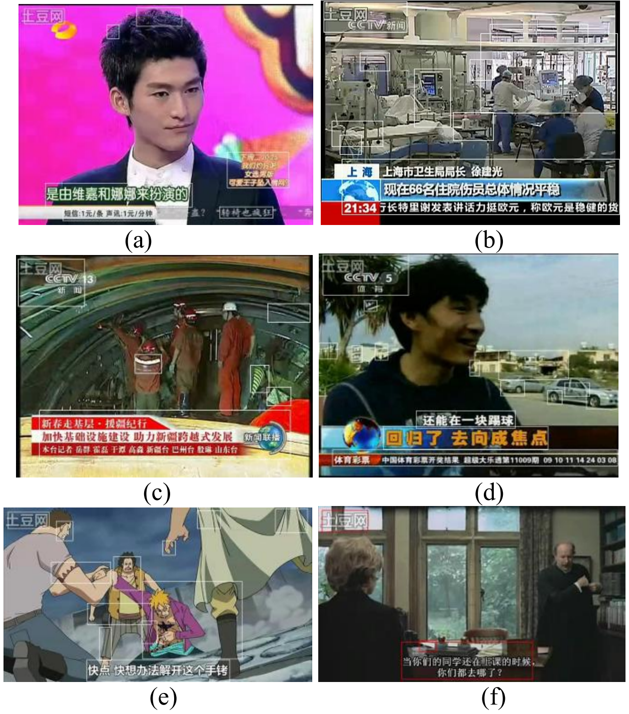
The text detection process is described as follow:

1) Combined stroke-map generation: as mentioned above stroke-map is previously proposed for document text enhancement and extraction. It is based on an assumption of dark strokes/lines appearing on light background. For video text images, there are different colour polarities, such as white on black, black on white, white on white, black on black. To tackle the problem we calculate stroke-map for negative image of original image at the same time. As shown in Fig.2 (a), we can see that original image contains texts with different colour polarities and different contrasts. Fig.2 (b) and Fig.2 (c) shows the stroke-map of original gray image and negative image respectively. To detect and localize video texts of any colour polarity we just add these two response image to generate our combined stroke map, as shown in Fig.2 (d).
2) Binarization: after adding two stroke-maps, we conduct binarization by applying Otsu method. Binarization result is shown in Fig.2 (e).
3) Morphological close and open operation: using 7×7 structuring element to implement close operation first. Morphological open operation is followed by using the same structuring element. As it shown in Fig.2 (f) and Fig.2 (g).
4) Merging horizontally near blocks: bounding box for each connected component is obtained. Horizontally near bounding boxes merged into larger bounding boxes then. As it shown in Fig.2 (h).

Merged bounding boxes denote candidate text regions. Detected candidate text regions of some sample images are shown in Fig.3.

B. Video Text Localization

As it shown in Fig.3, after text detection step, the candidate text regions contain almost all ground truth text regions. However, it also contains some non-text blocks, and boundary lines of text blocks are not precisely determined. Some candidate text regions include more than one text rows. The coarse-to-fine localization scheme is then performed to identify text regions accurately.



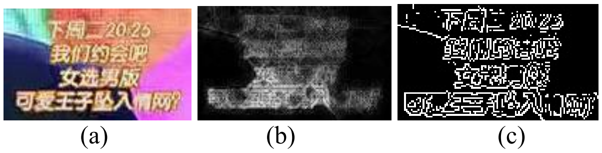
(a) (b) (c) (d) (e) (f)

Fig. 3 Text detection results

First, we employ projection profile to separate multiple text rows in one candidate text regions and identify text boundary lines for each text rows precisely. To localize text row lines we use Canny edge response image instead of combined stroke-map image which generated in previous section. As it shown in Fig. 4(b) and Fig. 4(c), it is obvious that canny response has better performance with separating text rows. We adopt a projection analysis technique to split these text blocks into individual text lines. Our projection scheme is similar to the technique proposed by Lienhart[11]. The numbers of candidate text pixels of each row and column in the text candidate image are used as the criteria to determine whether a text block should be separated.

$$thresh_{text} = \min_{profile} + (\max_{profile} - \min_{profile}) \times Thresh\_factor \tag{6}$$

Every line with a projection profile value exceeding $thresh_{text}$ is classified as containing text. However, there are some important differences.

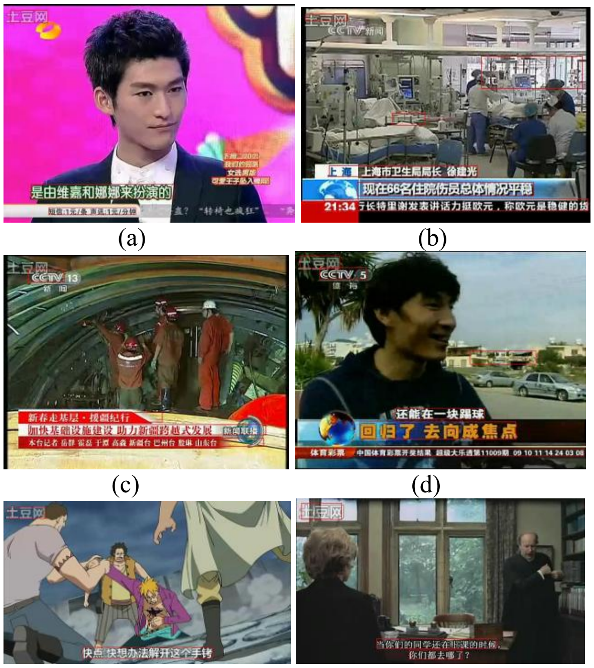


(a) (b) (c)

Fig. 4 Sample text block.(a)Original text block.(b)Combined stroke-map.(c) Canny response.

In our work we use horizontal projection value several time with different experimentally determined $Thresh\_factor$ to obtain precise text lines. The procedure of determining horizontal text line as follow:

1) Apply horizontal projection profile with $Thresh\_factor$ =1.5 to find horizontal text line positions and generate new bounding boxes.
2) If the height of new bounding box is greater than T (in our work it is set to 35 experimentally.), we employ horizontal projection profile to this new bounding box with $Thresh\_factor$=3.5.
3) New bounding boxes positions in previous large bounding boxes are set to zero. New text lines are detected from remaining edge response in previous large bounding boxes by horizontal projection profile with $Thresh\_factor$ = 1.2.



(a) (b)

(c) (d)

(e) (f)

Fig. 5 Text localization results

Next, some geometric constraints are employed to verify the bounding boxes. We use the following three heuristics to remove noise regions.

1) Size constraint: the height of a text line is larger that 8 pixels and width is larger that 10 pixels.

2) Shape constraint: the aspect ratio, which is defined as the ratio of width to height to height of a text line, is expected to be larger that 2 (horizontal text) or less than 0.5 (vertical text).

3) Fill factor constraint: the fill factor, which is defined as the is defined as the percentage of the number of candidate text pixel in a text line to the total number of pixels in this text line should be than 60%.

Finally, we employ SVM (Support Vector Machine) method to further verify candidate text blocks. In our work gray pixel value is directly used as feature. We take 500 positive and 500 negative samples; each of them is quadrate shape. After resizing these samples to size of 10*10, train SVM with these positive and negative samples. Because each candidate text region only contain one text row, before the classifying process, every candidate text blocks resized to 10 pixel height. SVM classification is conducted with a 10*10 sliding window and sliding step is 5. Each sliding window classified as text or non-text. For a candidate text block, the number of sliding window which classified as text is calculated. The probability of the candidate text block belonging to text is calculated as follow:

$$text\_pro = \frac{10 \times classified\_text\_count}{resized\_block\_width} \quad , (7)$$

Where $classified\_text\_count$ is the number of sliding window which classified as text, $resized\_block\_width$ is the width of candidate bounding boxes after resizing with the same resize factor with block height. If the $text\_pro$ is larger than 0.6, the candidate bounding box is classified as text, else it belongs to background region and eliminated. Fig. 5 shows some localization results.

III. Video Text Tracking

Through previous steps we obtained text region in rectangular bounding boxes. Once text is detected, the tracking process is started. Text motion in digital video is one of three types: static; simple linear motion (for example, scrolling movie credits); or complex non-linear motion (for example, zooming in/out, rotation and free movement of scene text). As graphic text is the main object of our text detection and localization step, our tracking method primarily focused on tracking static and rigid moving text objects.

Our text tracking approach will be described in detail in the following subsections.

A. Feature Selection

Selecting the right features plays a critical role in tracking. In general, the most desirable property of a visual feature is its uniqueness so that objects can be easily distinguished in the feature space. Local invariant features are a powerful tool that has been applied successfully in a wide range of systems and applications. In the context of motion, stereo, and tracking problems, a desirable quality of an interest point is its invariance to changes in illumination and camera viewpoint. Apparently local invariant features are better choice for this purpose compared to simple or global features. In the literature, we discuss commonly used interest point detectors include Harris interest point detector, KLT detector, SIFT detector and SURF detector.

Both the Harris [21] detector and KLT [22] detector are based on the auto-correlation matrix. Quantitatively both Harris and KLT emphasize the intensity variations using very similar measures. In practice, both of these methods find almost the same interest points. The only difference is the additional KLT criterion that enforces a predefined spatial distance between detected interest points.

Both SIFT and SURF feature detection process related to Hessian matrix. Hessian matrix was used to compute the principal curvature and eliminate the false key points in SIFT algorithm, while SURF [17] makes use of integral images to efficiently compute a rough approximation of Hessian matrix. In SIFT (Scale Invariant Feature Transform), image pyramid is used. The images are repeatedly smoothed with a Gaussian and then sub-sampled in order to achieve higher level of the pyramid. SIFT detector subtracts these pyramid layers in order to get the DoG (Difference Of Gaussian) images. While, in SURF, instead of iteratively applying the same filter to the output of previously filtered layer, box filter of any size at exactly the same speed directly applied on the original image.

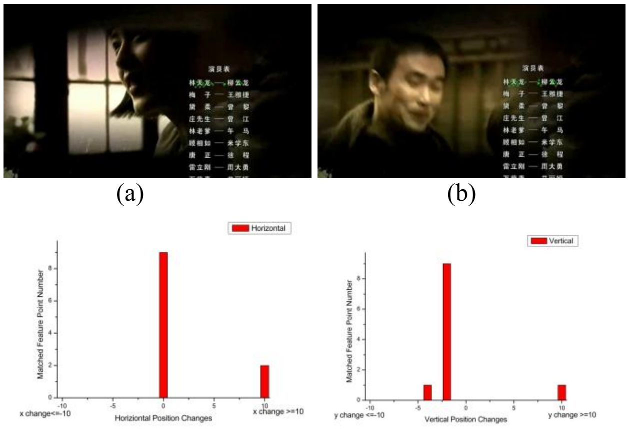
In theory, Harris and KLT detectors are invariant to both rotation and translation, but not invariant to affine or projective transformations. SIFT and SURF features are more robust features. Both feature invariant to image transformation or distortion. According to experimental comparison at work [19], SIFT shows best performance at rotation, scale changes and affine transformations, while it is slow and not good at illumination changes. SURF is more than 3 times faster than SIFT and has good performance as the same as SIFT. Especially, it has better performance to illumination changes. For video texts, illumination changes are more frequent than scale and affine changes. In that point, SURF feature is better choice for video text tracking. We also conduct an experiment to verify our opinion.

Experimental method and more specific experiment results are given in section four.

B. *Text Tracking*

Every tracking method requires an object detection mechanism either in every frame or when the object first appears in the video. In our work it is performed by text detection and localization step. Therefore, the main tasks of text tracking scheme are as follows:

1) Determine text ending frame.
2) Motion estimation for moving text blocks.



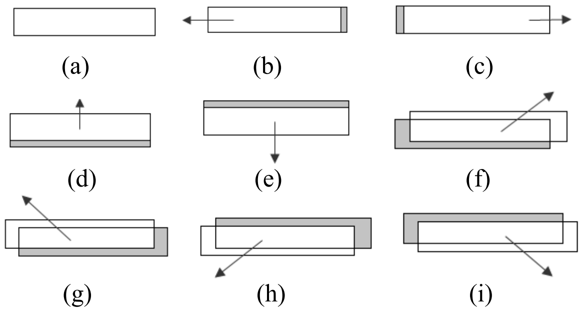
(a) (b) (c) (d)

Fig. 7 Motion estimation by histogram.(a)Previous frame.(b)Current Frame.(c) Horizontal position changes of matched surf pairs.(d)Vertical position changes of matched surf pairs.

Text ending frame include two type of video frame, one is text disappearing frame, the other is the frame in which moving text block lines exceeds frame boundary.

For static texts, there is only one kind of text ending frame, namely text disappearing frame. Text disappearing frames are determined based on the change of SURF feature numbers in text regions. If the feature point numbers in corresponding regions in neighbouring frames changes drastically, we can determine current frame as text disappearing frame.

For moving text tracking, there is also another kind of text ending frame. In this kind of frames, text blocks will crosses the image borders. Detecting this kind of text ending frame for moving text blocks is simply done by checking whether block positions exceeded the frame boundaries. In our approach, motion estimation is based on feature matching. Fast approximate nearest-neighbour algorithm[18] is employed. For false match elimination, we propose a fast histogram based method instead of Ransac algorithm.



(a) (b) (c) (d) (e) (f) (g) (h) (i)

Fig. 8 Motion type.(a)Static.(b)To left.(c) To right.(d)To above.(e) To blow.(f)To up-right.(g)To up-left.(h)To down-left.(i)To down-right.

The procedure of text tracking is described as follow:

1) Let “Prev” and “Cur” denote the frame indices of two neighbouring frames, namely previous frame and current frame. For frame Prev extract SURF feature points for each “detected” text region respectively, and calculate SURF descriptors. For frame Cur, SURF feature extracted at predefined feature search regions. Feature searching regions are calculated at step 5. As to the first and the second frames, just use original detected region for both of them.
2) For each currently tracked text block, compare its previous and current SURF feature point numbers. If the number of current key points is less than half of previous key point number, then current frame is determined as text disappearing frame and the tracking process for this text block go to step 6. Otherwise, tracking process continues to step 3.
3) Key point matching between two sets of points is implemented using fast approximate nearest-neighbour algorithm. [18]
4) Eliminate false matches and calculate text moving steps. For each text block, 21 bin histograms of horizontal and vertical position changes are created. Maximum values of these histograms are set to horizontal and vertical position changes of text region. In Fig 7, horizontal and vertical estimation values are 0

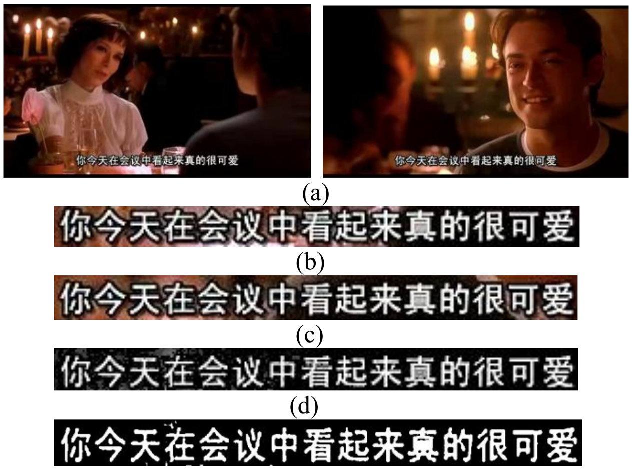
and -2 respectively. It would be 13 and 7, if we use average value to estimate the motion. Max moving step is set to 10 pixels heuristically. If the moving step is larger than max moving step, the current frame also determined as text ending frame.

5) Set feature searching region for next frame based on motion estimation result. In our work, there are eight different cases, as shown in Fig.8.
6) If text ending frames are found based on SURF point number or bounding box positions, delete this block from tracking list. Text detection and localization process conducted again. After checking each new localized text bounding box, newly detected text blocks will be added to tracking list and continue to step 1.

C. *Text Enhancement*

Having the tracking result, we can enhance text pixels. Take advantage of text redundancy, we can simply perform And operation for a list of tracked text blocks. Because in many cases background is changing while foreground text stay unchanged, we can obtain enhanced text image with simple background. As shown in Fig.9, we obtained enhanced text image Fig.9 (d) through combining text blocks from text appearing frame to text ending frame.

Using above enhanced text image with simple background, simple binirazation method such as Otsu[23] method is enough for obtaining clear binary text images which can processed directly by OCR software , as shown in Fig. 9(e).


你今天在会议中看起来真的很可爱
你今天在会议中看起来真的很可爱
你今天在会议中看起来真的很可爱
你今天在会议中看起来真的很可爱
你今天在会议中看起来真的很可爱
你今天在会议中看起来真的很可爱

(a)

(b)

(c)

(d)

(e)

Fig. 9 Text Enhancement. (a)Text start and ending frames.(b)Text block in starting frame.(c) Text block in ending frame.(d)Enhancement result.(e) Otsu result.

IV. Experimental Result

In this section, the proposed system is implemented and evaluated to show the efficiency of the proposed method.

We first evaluate the performance of text detection and localization, then compare performance of different feature for tracking purpose, and evaluate performance of text tracking. All the programs are written in C++ and the average computational costs are obtained by running the programs on a P4 2.4G PC with 2.0G RAM.

A. *Performance Evaluation of Text Detection*

For this experiment we collected five video sequences from one of the largest video website in China www.tudou.com. These video sequences including wide variety of video sources, such as news, movie credits, variety shows and TV dramas with subtitles and credits. The total number of frames is 12629, and the frame resolutions are between 352*264 to 720*540.

In order to evaluate and compare the performance of text detection and localization method, we compare it to single frame based method. In the first step, we developed ground truth recoding tool and performance evaluation tool respectively. Using ground truth recording tool, whether it contains text or not, we took the video frame images in every 30 interval, and semi handedly selected ground truth text block positions. Total 420 frames are taken. The ground truth text position information will be saved in the corresponding xml files. Every time we implement our single frame based text detection and localization program, the localization results are also saved in xml files. Then we use performance evaluation tool to compare the detected text position and ground truth positions in above two xml files. A correct detection is counted if and only if the intersection of a detected text region and a ground-truth text region covers at least 90% of both regions.

For quantitative comparisons, two metrics are adopted to evaluate the performance of our method, i.e.,

(1) **Recall**: The ration of the text regions correctly detected to the ground-truth text regions;

(2) **Accuracy**: The ration of the text regions correctly detected to the text region claimed by our system.

TABLE I presents the experimental result of our method. In the TABLE II, GT, D and CD represents ground-truth text block numbers, detected block numbers and correctly detected block numbers respectively. Because background and non-text objects also contain strong edge information, the edge based method produced many false positives.

TABLE I
Performance Evaluation Of Text Detection Method

Test Set	**Our Method**					**Method[7]**	
	GT	***D***	***CD***	***Recall***	***Accuracy***	***D***	***CD***
1	76	89	75	98.7%	85.4%	112	69
2	95	102	86	90.5%	84.3%	131	75
3	12	15	11	91.7%	73.3%	19	9
4	37	44	35	94.6%	79.5%	56	30
5	262	318	256	97.7%	80.5%	378	217
Total	482	568	463	96.1%	81.5%	696	400

B. *Performance Evaluation of Text Tracking*

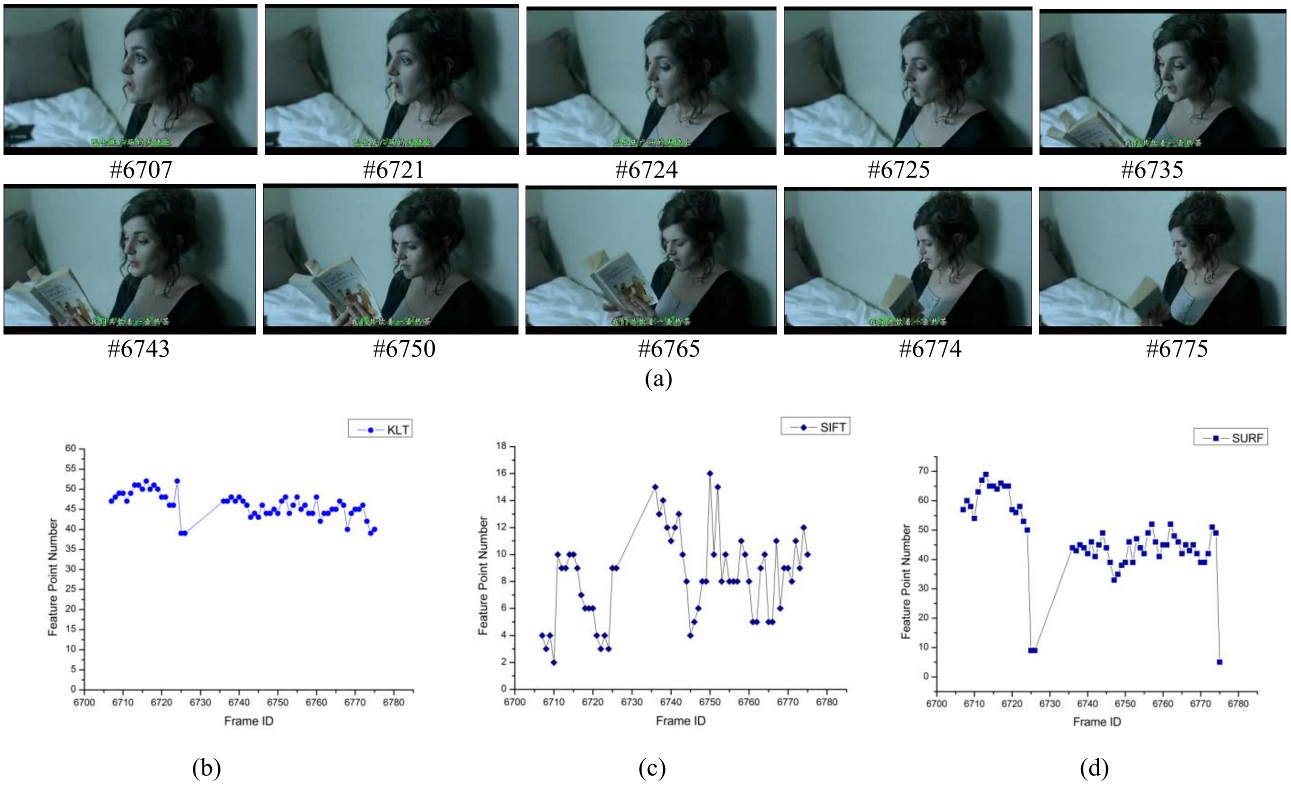


Fig. 6 Invariant features of Text Region. (a) Vide frames. (b) KLT feature number. (c) SIFT feature number. (d) SURF feature number.

In our work, text disappearing frame is determined based on the changes of feature point numbers on corresponding text regions in neighboring frames. We conduct an experiment and compare the performance of different features for determining text disappearing frames.

For the experiment for different features, we used two video sequences only contains static texts. We selected 65 and 80 text blocks that last more than 20 frames from each video sequence, and manually recorded the text disappearing frames and their position in video sequences. When we implement our text tracking system, it saves the text block positions and the feature point numbers in the duration between text appearing frame and text disappearing frame. We set current frame as text disappearing frame, if the number of current key points is less than 60% of previous key point number.

As shown in Fig.7, the number of SURF feature points is dropped drastically in frame #6725 and #6775 (frames are shown in Fig. 6). There are also some changes in KLT and SIFT feature point numbers when text disappearing frame occur. However, for the KLT features, these changes in feature number are very small, and it is hard to distinguish text appearing frame and text disappearing frame. As to SIFT feature, the number of feature points sometime fluctuate very largely between text appearing frames, and it will cause many false text disappearing frames.

The SURF features have higher sensibility for text disappearing frames than SIFT and KLT features. But we found SURF features are also very sensible to shot changes. When scene change take place between two neighboring key frames, the abrupt change between two neighboring frames is caused by the background change, not by the appearing or disappearing text, matched SURF features number decline drastically, and the scene changed frame would be falsely determined as the text disappearing frame. To deal with this problem, in our work we use unmatched SURF feature points number for determining text disappearing frames. It is robust against strong background variation caused by shot boundary and complex motion. TABLE II shows the correct tracked block numbers when we use different feature points.

TABLE II
PERFORMANCE EVALUATION OF TEXT TRACKING WITH DIFFERENT FEATURE

Test Set	**Text Block Numbers**	**Correctly Tracked Block Numbers**		
		KLT	***SIFT***	***SURF***
1	65	38	55	61
2	80	20	38	63

Then we compare the performance of our text tracking method with the method [14]. We use the same five video sequences used in first experiment. Experiment results are shown in TABLE III and TABLE IV.

Because we only track text objects, only the frames that contain text rows are processed for text detection and record the process times after processing. The average text detection time per frame is 0.34904s, and the average tracking time per frame is 0.12709s. As we predicted tracking is faster than text detection. Apparently, our tracking method is faster than method [14] and also has better performance with text tracking.

Another advantage of text tracking is that video text of very short period can be detected while processing equal interval frames may miss these texts. As it shown in TABLE I, when we process test sequence four with equal intervals 35 actually different text blocks are correctly detected. While when we use text tracking, as it shown in TABLE IV, 44 different text blocks are correctly detected.

TABLE III
PERFORMANCE EVALUATION OF TEXT TRACKING

Average Process Time Per Frame		
Text Detection	***Our Tracking Method***	***Tracking With Method[14]***
0.34904s	0.12709s	0.296034s

TABLE IV
PERFORMANCE COMPARISON OF DIFFERENT TRACKING METHODS

Test Set	**Frame Numbers**	**Text Block Numbers**	**Correctly Tracked Block Numbers**	
			Method[14]	***Our Method***
Static	4877	80	70	76
Static	4565	97	54	89
Static	680	6	5	5
Static	1432	46	38	44
Moving	1075	78	56	63

V. CONCLUSIONS

This paper proposes a unified framework for video text detection, localization, and tracking.

In our video text detection method we use stroke map for detecting candidate text regions instead of edge information, and also use the strong edge information for localizing individual text lines, so our method can reduce the false positives.

Our tracking algorithm is efficient, because it enjoys not only a low time complexity but also robustness against interference from scene change, target moving. It brings the stable SURF feature into video text tracking field. By defining and calculating a match score of text boxes in consecutive frames, the changes of text content can also be detected by our matching algorithm. Experiments on different video sequences show that the approach is viable and effective in text tracking.

In addition, the proposed scheme does not work well when texts are in complex non-linear motion, such as zooming in/out, rotation and free movement of scene text. However, such kind texts appear rarely in video sequences compared to the static texts and texts with simple linear motion. All these issues will be addressed in future research.

REFERENCES

[1] Xiangyun Ye, Cheriet M., Suen C.Y. "Stroke Model Based Character Extraction from Gray Level Document Images," IEEE Transactions on Image Processing 10(8) , 2001, pp.1152-1161.

[2] Gargi, U., Crandall, D., Antani, S., Gandhi, T., Keener, R., Kasturi, R. "A system for automatic text detection in video," Proc. Int. Conf. on Document Analysis and Recognition, Seattle, USA, 1999, pp. 29-32

[3] C.M. Lee, A. Kankanhalli, "Automatic extraction of characters in complex images," Int. J. Pattern Recognition Artif. Intell. 9(1) , 1995, pp.67-82.

[4] H. Hase, T. Shinokawa, M. Yoneda, C.Y. Suen, "Character string extraction from color documents," Pattern Recognition 34 (7), 2001, pp.1349-1365.

[5] C. Liu, C. Wang, R. Dai, "Text detection in images based on unsupervised classification of edge based features," IEEE ICDAR, 2005, pp. 610-614.

[6] D. Chen, K. Shearer, H.Bourlard, "Text enhancement with asymmetric Alter for video OCR," Proceedings of International Conference on Image Analysis andProcessing, 2001, pp. 192-197.

[7] S. Jianyong, L. Xiling, Z. Jun, "An Edge-Based Approach for Video Text Extraction," Computer Technology and Development, ICCTD '09, 2009, pp.331-335.

[8] Ye, Q., Huang, Q., Gao, W. and Zhao, D. "Fast and robust text detection in images and video frames." Image and Vision Computing (23), 2005, pp. 565-576.

[9] Chun, Y. Bae, T.Y. Kim, "Automatic text extraction in digital videos using FFT and neural network," Proceedings of IEEE International Fuzzy Systems Conference, Vol. 2, 1999, pp. 1112-1115.

[10] Q. Ye, W. Gao, Q. Huang, "Automatic Text Segmentation from Complex Background," Proc. of IEEE Int'l Conf. on Image Processing, 2004, pp.2905-2908.

[11] R. Lienhart, A. Wernicke, "Localizing and segmenting text in images and videos," IEEE Transactions on Circuits and Systems for Video Technology 12(4), 2002, pp.256–268.

[12] Lienhart, R, Effelsberg, "Automatic text segmentation and text recognition for video indexing", Multimedia Syst. (8), 2000, pp.69-81.

[13] Qian, X., Liu, G., Wang, H., Su, R. "Text detection, localization, and tracking in compressed video," Signal Processing: Image Communication (22), 2007, pp.752-768.

[14] Yinan Na, Di Wen, "An effective video text tracking algorithm based on SIFT feature and geometric constraint," Advances in Multimedia Information Processing PCM (1), 2010, pp.392-403.

[15] Shi, J., Luo, X., "DCT Feature Based Text Capturing and Tracking," Chinese Conference on Pattern Recognition, 2009, pp. 294–297.

[16] K Jung, K In Kim, AK Jain, "Text information extraction in images and video: a survey," Pattern Recognition (37), 2004, pp.977-997.

[17] Bay, H. and Tuytelaars, T. and Van Gool, L. "SURF Speeded Up Robust Features," 9th European Conference on Computer Vision, 2006

[18] Marius Muja, David G. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration," International Conference on Computer Vision Theory and Applications (VISAPP), 2009, pp.331-340.

[19] Luo Juan, Oubong Gwon, "A comparison of SIFT, PCA-SIFT and SURF," International Journal of Image Processing (IJIP), Vol. 3, 2009, pp.143-152

[20] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," International Journal of Computer Vision,60(2), 2004, pp.91-110.

[21] Harris, C., Stephens, M. "A combined corner and edge detector," In 4th Alvey Vision Conference. 1998, pp.147-151.

[22] Shi, J., Tomasi, C., "Good features to track," In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1994, pp.593-600.

[23] N.Otsu, "A thresholds election method from gray-level histograms," IEEE Transactions on System, Man and Cybenretics, Vo1.9, No.1, 1979, pp.62-66.