COARSE-TO-FINE VIDEO TEXT DETECTION

Guangyi Miao[1], *Qingming Huang*[1,2], *Shuqiang Jiang*[2], *Wen Gao*[2,3]

[1]Graduate University of Chinese Academy of Sciences, Beijing, 100049, P.R. China
[2]Key Lab of Intell. Info. Process., Inst. of Comput. Tech., CAS, Beijing, 100190, P.R. China
[3]Institute of Digital Media, Peking University, Beijing, 100871, P.R. China
{gymiao, qmhuang, sqjiang, wgao}@jdl.ac.cn

ABSTRACT

In this paper, we propose an effective coarse-to-fine algorithm to detect text in video. Firstly, in coarse-detection section, stroke filter is employed to detect all candidate stroke pixels, and then a fast region growing method is developed to connect these pixels into regions which are further separated into candidate text lines by projection operation. Secondly, in fine-detection section, correct text regions are selected from candidate ones by support vector machine (SVM) model and stroke features, and text regions in multi-resolution are integrated. Finally, the result is optimized significantly according to temporal correlation information. Experimental results show that our algorithm achieves real-time performance and is robust for the variation of language, font, size, color and noise of text caused by low frame resolution in video.

Index Terms**—** Text detection, stroke features, SVM

1. INTRODUCTION

In recent years, videos on webs and in databases are increasing with a fast speed. It is difficult for the users to quickly find their interested content in enormous quantity of video databases. Text in video frames carries important information which is a very compact and accurate clue for video summarization and retrieval.

The role of video text detection is to find and locate the text regions in images and video frames. Existing text detection algorithm can be classified into four kinds: color and connected component based algorithm [1-2], edge and texture based algorithm [3-9], video temporal information based algorithm [10], and stroke based algorithm [11-12]. Edge and texture based algorithm is most popular and stroke based algorithm is the latest.

Although a lot of algorithms have been developed, they are all still far away from practical application. Fast and robust algorithms for text detection under various conditions need to be further investigated. First, better filter is needed for text localization. Second, better features are needed for text identification. Third, temporal correlation information need to be deeper developed for video text detection. Finally, fast speed is needed for real-time application.

In this paper, we try to find more effective and fast algorithm to detect text in video. Our algorithm flow chart is shown in Fig. 1. We detect video text in a coarse-to-fine scheme. To realize fast performance, a fast region growing method is proposed to locate candidate text regions in coarse-detection section. To better characterize text, stroke features are extracted from candidate regions and then put into SVM model for region identification in the fine-detection section. Temporal correlation information is also used for optimization. It contributes a lot to both the speed and the precision.

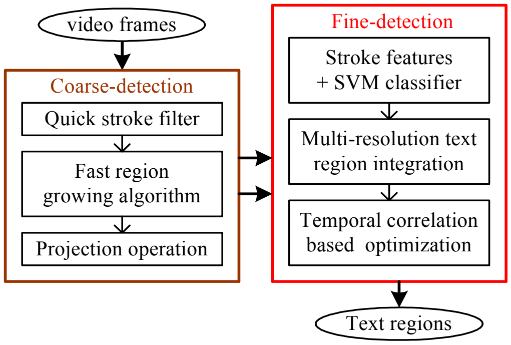

video frames
Coarse-detection
Quick stroke filter
Fast region growing algorithm
Projection operation
Fine-detection
Stroke features + SVM classifier
Multi-resolution text region integration
Temporal correlation based optimization
Text regions

Fig. 1. Algorithm flow chart.

2. VIDEO TEXT DETECTION

2.1. Stroke Pixels Detection

In Liu's work [11-12], he proposes a novel stroke filter, and discusses the relationship between stroke filter and other filters. Theory and experiments proves that stroke filter is better for text localization. We employ his quick stoke filter [12] to obtain stroke map (see Fig. 2(a)), as the first step of our algorithm.

This work was supported in part by National Natural Science Foundation of China under Grant 60773136 and 60702035, in part by National Hi-Tech Development Program (863 Program) of China under Grant 2006AA01Z117 and 2006AA01Z315

2.2. Text Region Localization

To localize text region, [12] uses connected components analysis (CCA) which is not stable enough because too many rules have to be defined. [7] uses a slide window to localize candidate regions. But the size of the window and its step of shifting affect the performance to a large extent, and they are hard to specify in different kinds of videos.

A text line is made up of a ‘cluster’ of text strokes. None but ‘dense’ stroke pixels can construct a text region and the isolated candidate pixels are often noises. To obtain candidate text region, [9] proposes a ‘density-based’ region growing method, which achieves better result than morphological operations such as ‘close’ operation. A pixel *P* will be labeled as high-density pixel if the percentage of candidate pixels in its neighborhood is larger than the threshold *TD*. Unfortunately, this method is region-size-sensitive and time consuming. We improve the method and propose a fast region growing algorithm which is also stable. The stroke density map (SDM) is calculated according to Eq.(1). In this equation, $Dens(x,y)$ denotes SDM, *w* and *h* denote that the region size is $(2w+1)\times(2h+1)$, and $R(x,y)$ denotes the stroke map.

$$Dens(x,y)=\frac{1}{(2w+1)(2h+1)}\sum_{n=-h}^{h}\sum_{m=-w}^{w}R(x+m,y+n) \qquad (1)$$

In order to realize region-size-insensitive localization, Eq. (1) is performed by the following steps. 1) Obtain temp map one (TM1) according to stroke map: the starting pixel of each row is calculated according to Eq. (2), and the other pixels following the starting pixel of each row are calculated according to Eq. (3). 2) Obtain temp map two (TM2) according to TM1: the starting pixel of each array is calculated according to Eq. (4), and the other pixels following the starting pixel of each array are calculated according to Eq. (5). 3) Obtain SDM according to TM2: every pixel is calculated according to Eq. (6).

$$Tm1(w+1,y)=\sum_{x=1}^{x=2w+1}R(x,y) \qquad (2)$$

$$Tm1(x,y)=Tm1(x-1,y)-R(x-w-1,y)+R(x+w,y), \quad x>w+1 \qquad (3)$$

$$Tm2(x,h+1)=\sum_{y=1}^{y=2h+1}Tm1(x,y) \qquad (4)$$

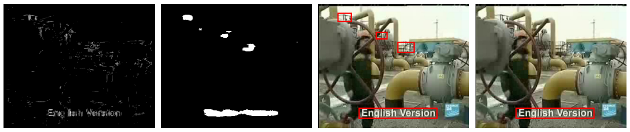
$$Tm2(x,y)=Tm2(x,y-1)-Tm1(x,y-w-1)+Tm1(x,y+w), \quad y>h+1 \qquad (5)$$

$$Dens(x,y)=\frac{1}{(2w+1)(2h+1)}Tm2(x,y) \qquad (6)$$

Our fast region growing algorithm is region-size-insensitive, and needs only 4 times of “*add*” operation for each pixel. By comparison, if the region size is 13×13 , method in [9] needs $13\times13-1=168$ times of “*add*” operations which is much more than ours’. The fast speed guarantees the real-time performance of our whole algorithm. The threshold *TD* is calculated according to Eq. (7), where *N* is the size of the SDM and *k* is a parameter which is set as 0.35 experientially. Then *TD* is restricted from 20 to 40, to make sure it is not too big or too small. If the density of candidate pixel *P* is bigger than *TD*, we label it as high-density pixel. After this step, high-density pixels often converge into some regions (see Fig. 2(a) and (b)).

$$TD=30+k\times\left[\frac{1}{N}\sum_{y}\sum_{x}Dens(x,y)-30\right] \qquad (7)$$

To separate text regions into candidate text line, we make use of projection information. If the width of the region is bigger than its height, we calculate its horizontal projection and separate it into sub horizontal lines. Otherwise, we calculate its vertical projection and separate it into sub vertical lines. Text localization result is shown in Fig. 2(c). In this paper, we don’t consider text which is too small to be recognized (the height is smaller than 9 pixels).



(a) stroke map (b) SDM map (c) coarse result (d) after SVM

Fig. 2. Text localization.

2.3. Text Region Identification

Sometimes, false text line is detected because of line structure which is similar with stroke. To fine correct text line from candidate ones, features and classifier are both very important. In the following, stroke features and SVM classifier are used to finish this job.

In most of previous works, features are extracted from the maps which are obtained by edge filters or texture filters. These features don’t really consider the intrinsic characteristics of text. Text has its own stroke pattern. It contains some character strokes that form a text line in special rules. So we extract stroke features from stroke maps. Four kinds of stroke features are extracted as follows:

1) Stroke moment features: Compared with background, text region has different stroke density moment distribution. In our work, the mean, second-order and third-order central moments are extracted from stroke map.

2) Histogram of the stroke illumination and stroke directions: Illumination histogram is quantized into 8 bins. For text region, the contrast between stroke pixels and background pixels should be large. So the distributions of illumination should be like an arc. The beginning bin and the ending bin of the histogram should be larger than the center bins. Stroke direction histogram is quantized into 4 bins, indicating four directions of the stroke: horizontal, vertical and diagonal. The direction of each pixel can be obtained in the process of getting stroke map (see [12]).

3) Color distribution of strokes: We binarize the stroke map using region local Ostu [13] threshold. All the highlighted pixels in the binarized stroke map can be clustered into two classes: pixels on the strokes and pixels between strokes. After unsupervised color clustering, we can obtain the color distance of two classes and the color variances of each class. The class with more pixels is considered to be stroke pixels. The rate of text pixels and non-text pixels is also one feature.

4) Histogram of stroke length: In the binarized stroke map, we calculate the length of strokes and obtain a 5 bins' histogram. For text region, the length of strokes should be similar. So the distribution of stroke length histogram should be like an arc. The beginning bin and the ending bin should be smaller than the center bins. If not, for example, the beginning bin is very large, it may be non-text region.

All the stroke features mentioned above make up a 3+12+4+5=24 dimensions vector for each candidate text region. This vector is put into SVM classifier to classify it into two classes: text region and non-text region.

SVM is easier to train, needs fewer training samples and has better generalization ability. It is very effective for text identification [9]. Considering the limited number of training sample, we use SVM classifier in our work. The SVM model was trained off-line on a dataset consisting of 1000 text and 2000 non-text labeled samples.

Some false detection regions are discarded by SVM classifier (see Fig. 2(c) and (d)).

2.4. Multi-resolution Integration

To detect text with different size, Liu in [12] uses different scaled stroke filter. But stroke filter is time consuming when the stroke scale is large. In addition, stroke in different scale may affect each other. For example, small text and large text are detected in one candidate region which is hard to process.

In order to save time, we use only the minimum scale of stroke filter in different scaled initial images. In order to minimize the effect between different scales of texts, we use similar technique used in [6]. If text is detected in the first scale image, this region is masked in the second region. This way is also time-saving.

2.5. Temporal Correlation Based Optimization

For video text detection, temporal correlation information is very important. Text region always stays several seconds. So we choose two frames from one second for text detection, which is enough for temporal correlation. Here *NF* means the number of frames a text region stays. By checking *NF*, we can eliminate those text regions that are transitory or have already been detected in the previous frames, and only accept text regions whose *NF* is more than 2. We realize it in following steps.

For a new text region, we set its *NF* as 1. If the overlapped area of two text region in two connective frames is larger than 80 percent of each area of them, we compare the two regions according to Eq. (8 and 9). In Eq. (8), $P_i^{cor}(x,y)$ means the value of the pixel (x,y) at frame *i* in channel *cor* of color RGB. $CorDist_i(x,y)$ means the color distance at the position of (x,y) between frame *i* and *i*+1. In Eq. (9), *OR* means the overlap area. $R_i(x,y)$ is the value of pixel (x,y) in frame *i* in the stroke map. It means the weight of the distance. Highlighted pixels on the stroke map should provide larger contribution to the distance between two regions ($RegDist_i$). If $RegDist_i$ is smaller than a threshold (15 in this paper), we consider the two regions have the same text content. The one with larger SVM probability to be text is chosen as the text area, and its *NF* adds 1.

$$CorDist_i(x,y) = \left\{ \frac{1}{3} \sum_{cor=R,G,B} \left[P^{cor}{}_{i+1}(x,y) - P^{cor}{}_{i}(x,y) \right]^2 \right\}^{\frac{1}{2}} \quad (8)$$

$$RegDist_i = \left[\sum_{(x,y)\in OR} R_i(x,y) \times CorDist_i(x,y) \right] / \left[\sum_{(x,y)\in OR} R_i(x,y) \right] \quad (9)$$

If *NF* of a text region is bigger than 2, it means that this text region stays more than one seconds and we consider it as a stable region. In this case, we compare it with the same region of next frame directly before detecting text in that frame, thus speed up the processing time.

3. EXPERIMENTAL RESULTS

Our experiments were conducted at a personal computer with P4 3.4Ghz processor and 1G main memory. Three languages of test video clips are used: English, Chinese, and Japanese. These video clips are all captured from TV or downloaded in internet, including news, advertisements, movies, entertainment, talk show and so on. They are converted to the format of MPEG-1 with resolution of 352×288 . 1.5 hours of video clips are used for setting experimental thresholds and SVM training. Another 3 hours of video clips are used for testing.

Fig. 3 and 4 illustrates some of the detection results. In Fig. 3, texts with different languages, fonts, sizes, colors, contrast are well detected. In Fig. 4, texts are embedded in complex background containing branches, lines, or other objects full of edges and texture. Though our algorithm works well in complex cases, sometimes it may have false detection, which is mainly caused by background object with stroke-like structure. Detection missing is mainly caused by text with special font whose strokes are not bar like, or text with only one single letter or digit character.

Based on the idea of performance evaluation in [9], we use recall rate and precision rate to evaluate our algorithm (shown in Table 1). For Chinese, the recall rate is lower than the other three. That's because the strokes of some complex Chinese characters are very dense due to the low video resolution, and may be missed by stroke filter.

Contrarily, for English and Japanese, the strokes of the character are always sparser, and can be detected easily.

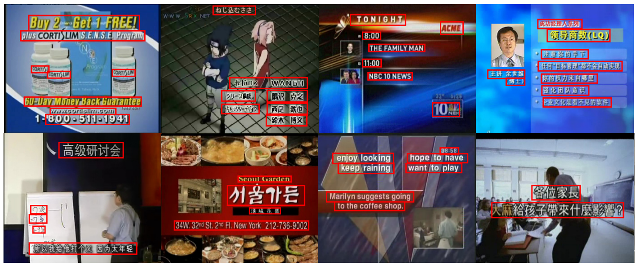


Fig. 3. Texts with different colors, sizes, fonts, and contrast.

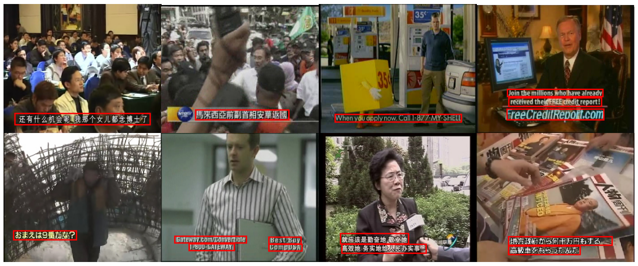

还有什么机会呢 我那个女儿都念博士了
馬來西亞前副首相安華返國
When you apply now. Call 1-877-MY-SHELL
Join the millions who have already
received their FREE credit report!
FreeCreditReport.com
おまえは9番だな?
Gateway.com/Convertible
1-800-GATEWAY
Best Buy
CompUSA
就应该是勤奋地 敬业地
高效地 务实地给人民办实事
地方政府から何千万円もする
高級車をもらったんだ

Fig. 4. Texts embedded in complex background.

Table 1. Text detection results of our algorithm

Language	English	Chinese	Japanese
Recall rate	93.6%	88.5%	94.3%
Precision rate	94.4%	93.9%	95.5%

Table 2. Performance comparison of two algorithm

Approach	Speed (frames/sec)	Recall rate	Precision rate
Our algorithm	33.7	92.1%	94.5%
Ye's algorithm	15.1	90.9%	89.3%

The speed of our algorithm is very fast. It can process 33.7 frames per second. In our program, we choose two frames from one second for text detection. It means that our program can process 33.7/2=17 seconds' video per second. This speed guarantees that application is real-time.

We compare our algorithm with Ye's work [9] (see Table 2). In his work, wavelet features are used for text detection. For the same test video clips, our approach is much better, especially in respect of the speed and the precision rate.

4. CONCLUSIONS

In this paper, a coarse-to-fine video text detection algorithm is proposed, which is fast and robust. Stroke filter is employed to obtain stroke map. The fast region growing method guarantees both the speed and the good result of our whole algorithm. Stroke features, which represent the intrinsic characteristic of text, are extracted for text identification. Temporal correlation information based optimization also contributes to the performance significantly. In our future work, we will focus on video text segmentation and develop an integrated system for fast and robust video text detection, segmentation and recognition.

5. ACKNOWLEDGEMENT

This work was supported by "Science100 Plan" of Chinese Academy of Sciences under Grant 99T3002T03, and the Knowledge Innovation Program of the Chinese Academy of Sciences under Grant No. 20076032.

6. REFERENCES

[1] V. Wu, R. Manmatha, and E.M.Riseman. "Finding text in images," *Proceedings of the second ACM international conference on Digital libraries*, pp. 23-26, 1997.

[2] A. K. Jain and B. Yu, "Automatic text location in images and video frames," *Pattern Recognition*, vol. 31, pp. 2055–2076, 1998.

[3] T. Sato, T. Kanade, E. K. Hughes, and M. A. Smith, "Video OCR for digital news archive," *Proc. IEEE Workshop on Content-Based Access of Image and Video Database*, pp. 52–60, 1998.

[4] V. Wu, R. Manmatha, and E. M. Riseman, "Textfinder: An automatic system to detect and recognize text in images," *IEEE Transactions on Pattern Analysis and Matching Intelligence*, vol. 21, no. 11, pp. 1224–1229, Nov. 1999.

[5] R. Lienhart and A. Wernicke, "Localizing and Segmenting Text in Images and Videos," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 12, No. 4, April 2002.

[6] M.R. Lyu, J. Song, and M. Cai, "A Comprehensive Method for Multilingual Video Text Detection, Localization, and Extraction," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 15, pp. 243- 255, Feb. 2005.

[7] H. Li, D. Doermann, and O. Kia, "Automatic text detection and tracking in digital video," *IEEE Transactions on Image Processing*, vol. 9, no. 1, pp. 147–156, Jan. 2000.

[8] K.I. Kim, K. Jung, H. Kim, "Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm," *IEEE Transactions on Pattern Analysis and Matching Intelligence*, vol. 25, pp.1631–1639, December 2003.

[9] Q. Ye, Q. Huang, W. Gao, and D. Zhao, "Fast and robust text detection in images and video frames," *Image and Vision Computing*, Vol.23, pp565-576, March 2005.

[10] X. Tang, X.B. Gao, J. Liu, and H. Zhang, "A spatial-temporal approach for video caption detection and recognition", *IEEE Transactions on Neural Networks*, Vol. 13, pp. 961–971, July 2002.

[11] Q. Liu, Ch. Jung, and Y. Moon, "Text Segmentation based on Stroke Filter," *Proceedings of ACM international conference on Multimedia*, pp. 129-132, 2006.

[12] Q. Liu, Ch. Jung, S. Kim, Y. Moon, and J. Kim, "Stroke Filter for Text Localization in Video Images," *IEEE International Conference on Image Processing*, pp. 1473-1476, 2006.

[13] N.Otsu, "A threshold Selection method from gray-level histogram," *IEEE Transactions on systems Man and Cybernet*, pp.62-66, 1989.