AUTOMATIC TEXT TRACKING IN DIGITAL VIDEOS

Huiping Li and David Doermann
Language and Media Processing Laboratory
Institute for Advanced Computer Studies, University of Maryland
College Park, MD 20742-3275
huiping@cfar.umd.edu, doermann@cfar.umd.edu

Abstract - In this paper we address the problem of automatically tracking moving text in digital videos. Our scheme consists of two separate processes including monitoring which detects the new text line entering a frame, and tracking which uses prediction techniques to reconcile the text from frame to frame. Temporal consistency allows us to conduct the monitoring process periodically and reduce the computation complexity. The tracking process uses a rapid prediction/search scheme to update the position of the text blocks. We provide details of the implementation and results both on text moving in the scene and text which appears to move as a result of camera motion.

INTRODUCTION

Text which appears in digital videos can provide important supplemental information for indexing and retrieval. Although extensive work has been done on the extraction of text from images ([1, 2, 3, 4, 7, 8]) and isolated video frames ([5, 6]), very little work has considered the temporal aspects of video. In our previous work, we provided a methodology which identifies both scene and graphic text from low resolution video key-frames [5]. We have found it necessary, however, to consider temporal changes to detect new information, to avoid processing redundant information, and to provide a means of integrating noisy results over multiple instances.

Text may span tens or even hundreds of frames, providing a tremendous amount of redundant information. By tracking moving text we can not only reduce the processing time by not applying all steps (detection, extraction, enhancement, OCR, etc) to every frame, but also maintain integrity by detecting when new information appears. For example, for "ticker" or "scrolling" text, new text continuously appears the screen so the system may need to process parts of subsequent frames to achieve a complete detection. Furthermore, we can make use of the temporal consistency to filter out false detections while

The support of this research by the Department of Defense under contract MDA 9049-6C-1250 is gratefully acknowledged.

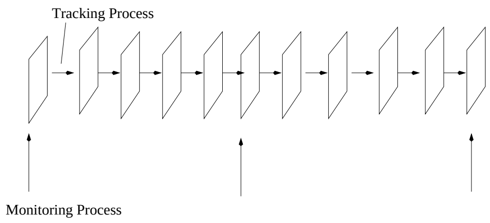

Figure 1: The scheme for text tracking in digital videos.

having multiple instances to provide enhancement for OCR. In [6] Lienhart and Stuber do make use of temporal information to enhance the extraction result, but they do not consider the problem of tracking moving text.

This paper is organized as follows: In Section 2 we introduce our tracking scheme. Text element generation is presented in Section 3 and details of the tracking algorithm are given in Section 4. In Section 5 we present experimental results and in Section 6, we summarize the system and provide a brief discussion.

TEXT TRACKING SCHEME

Text tracking in digital videos can be viewed as a multi-target tracking problem since multiple text blocks can appear at any time and in theory move in different directions in the frame. If we treat all non-text as background, we can have four combinations of static and moving text and static and moving background. Since there can be both camera and scene motion it will be very hard, if not impossible, to classify moving text simply by motion analysis. Furthermore, since new text lines can enter at any frame at any time, we must periodically run a text detection process in conjunction with tracking.

A very simplistic solution to text tracking would be to conduct the text detection frame by frame and then match the corresponding text elements between the consecutive frames. Considering there are close to 30 frames per second, the text extraction phase in the absence of context would be prohibitive. We can, however, make use of the fact that text remains in the scene for a number of consecutive frames to reduce the complexity by performing text detection process periodically and focus on the tracking process (Figure 1).

MONITORING PROCESS

Automatic Text Detection in Single Frame

We use a wavelet/neural network segmentor to detect text blocks in individual frames. The process relies on decomposing the image using wavelets,

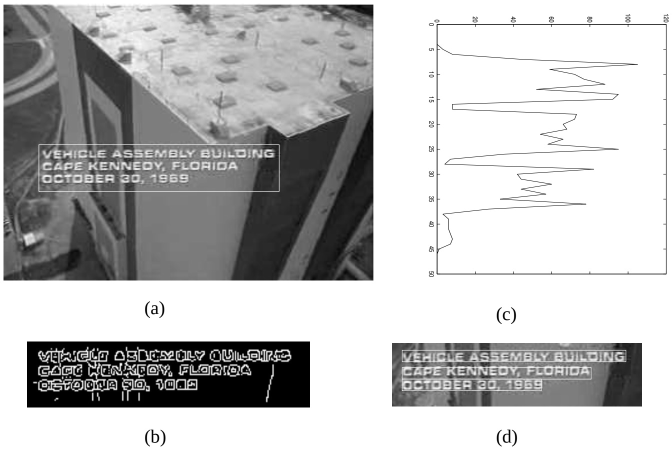

Figure 2: (a) The result of text detection, (b) Canny edge map of detected text block, (c) Horizontal projection profile of (b), (d) Generated text lines.

extracting features from the first level wavelet decomposition, classifying using a neural network and integrating results between different scales. This work has been published previously in [5].

Text Element Generation

The result of our text detection algorithm is a set of text blocks, which may contain multiple text lines if text is spatially compact (Figure 2a). These blocks are not typically good candidates for tracking because of their size. Furthermore, different text lines in the same text block may move in different speeds, or some text may remain static while other text moves. For these reasons, we need to divide the text blocks into more compact text elements, typically a single word and at most one line.

In order to simplify the problem, we currently assume the text line is oriented horizontally and use projection profile to extract text elements from text blocks. We use the projection profile of Canny edges to locate the text elements to avoid difficulties in dealing with normal and reversed text. Figure 2b shows the Canny edges of the detected text in Figure 2a. Figure 2c is the horizontal projection profile of Figure 2b. Figure 2d is the set of generated text elements. Blocks which do not have horizontal orientation will be tracked as a whole.

TRACKING PROCESS

Tracking Model

As either the camera or the text moves, the patterns of image intensities change in complex ways. We can describe these changes as image motion using Equation 1:

$$I(x, y, t+\tau) = I(x - \xi(x, y, t, \tau), y - \eta(x, y, t, \tau)) \tag{1}$$

A subsequent image taken at time $t + \tau$ can be obtained by moving every point in the image taken at t, by an amount $\delta = (\xi, \eta)$, represented by an affine transform:

$$\delta = D\vec{x} + \vec{d} \tag{2}$$

where D is a 2×2 deformation matrix and $\vec{d}$ is a 2×1 displacement vector.

If we choose the minimization of the sum of square difference (SSD) as matching metric, then the basic tracking algorithm might be described as: Given a reference block in image I, and a search region W in image J, determining the six parameters that appear in the deformation matrix D and displacement vector $\vec{d}$, which minimizes Equation 3.

$$\epsilon = \int\int_W [J(D\vec{x} + \vec{d}) - I(\vec{x})]^2 d\vec{x} \tag{3}$$

The solution of Equation 3 can be very complex. Fortunately, in most cases text in digital videos are rigid and the text motion can be locally described as linear motion. This happens, for example, with movie credits where text moves up the screen and with ticker text where text lines move cross the screen. Under this assumption, Equation 3 can be simplified to

$$\epsilon = \int\int_W [J(\vec{x} + \vec{d}) - I(\vec{x})]^2 d\vec{x} \tag{4}$$

Reducing Complexity

One variable in Equation 4 is the choice of the size of the window W. Although we can constrain the search window to a fixed range by assuming that the text motion between two consecutive frames is small, this assumption still has several shortcomings. First is the computation complexity. In digital videos with standard frame size 320×240, we need loosely set the search window range to be $\vec{x} - 10 \sim \vec{x} + 10$, where $\vec{x}$ is the current position. This however, will require over 400 image matches. If the video resolution is increased to 640×480, the range will double in size.

The method we proposed is based on the the fact that the current motion of the text is typically qualitatively consistent with previous motion. We use its past and current positions to predict its future position by using a simple prediction scheme. Although we could use more precise and robust models, such as adaptive linear prediction or Calman filtering, this will add new computation complexity. We predict the text position in the next frame as the sum of its position in the current frame and its current moving speed, which can be represented as:

$$\vec{x}_{n+1} = \vec{x}_n + (\vec{x}_n - \vec{x}_{n-1}) \tag{5}$$

where $\vec{x}_{n+1}, \vec{x}_n, \vec{x}_{n-1}$ are text positions in the next, current and past frame respectively and $\vec{x}_n - \vec{x}_{n-1}$ is its current moving speed.

EXPERIMENT

We have tested our scheme on 6 video sequences coming from different sources, including TV business news, movie, scanning camera across the wall, etc and the results are encouraging.

Figure 3a is an example when we use our system on movie credit tracking. The text lines in the video are detected and tracked correctly. Figure 3b is the part of the tracking result when we span camera across the wall. Although there is slight rotation when camera moves, our algorithm follows the text well. However, since our tracking model is based on linear case, the system will fail in typically non-linear cases such as zooming-in text in digital videos.

The text detection requires about 1 second/frame in Sun Workstation Untra 1. Since we conduct detection process periodically (every 5 frames in our case) and make use of fast prediction/search to reduce complexity, the average processing time for each frame is about 0.2 second.

We have made demos by displaying the tracking result in the form of mpeg videos. Each text element detected and tracked are surrounded by a red bounding box. It can be easily displayed using any software which can show mpeg videos. We are investigating quantitative metrics to evaluate the system.

CONCLUSION AND FURTHER WORK

We have addressed the problem of text tracking in digital videos in this paper. The algorithm consists of two processes including monitoring process and tracking process. We conduct the expensive monitoring process periodically to reduce the computation complexity. A rapid prediction/search scheme is used to update the position of the text blocks in the tracking process.

Currently our algorithm works on rigid, linearly moving text. We are actively investigating algorithms for tracking text in more complex cases.

Frame 100 Frame 116 Frame 143 Frame 160

(a)

Frame 100 Frame 180 Frame 260 Frame 299

(b)

Figure 3: (a) Tracking movie credits, (b) Tracking moving scene text generated by spanning the camera cross the wall. All demos are available at http://www.cfar.umd.edu/~huiping/demo.html

References

[1] Y. Dai, N. Zheng, X. Zhang, and G. Xuan. Automatic recognition of province name on the license plate of moving vehicle. pages 927–929, 1988.

[2] T. Gotoh, T. Toriu, S. Sasaki, and M. Yoshida. A flexible vision-based algorithm for a book sorting system. 10:393–399, 1988.

[3] S.K. Kim, D.W. Kim, and H.J. Kim. A recognition of vehicle license plate using a genetic algorithm based segmentation. In *ICIP*, pages 661–664, 1996.

[4] E. Lee, P. K. Kim, and H. J. Kim. Automatic recognition of a car license plate using color image processing. pages 301–305, 1994.

[5] H. Li and D. Doermann. Automatic identification of text in digital video key frames. In *Proceedings of ICPR'98*, pages 129–132, 1998.

[6] R. Lienhart and F. Stuber. Automatic text recognition in digital videos. In *University of Mannheim, Department of Computer Science, Technical Report TR-95-036*, 1995.

[7] R. Manmatha. Multimedia indexing and retrieval research at the center for intelligent information retreival. In *Symposium on Document Image Understanding Technology*, pages 16–30, Annopolis, Maryland, 1997.

[8] G. Piccioli, E. De Micheli, P. Parodi, and M. Campani. Robust method for road sign detection and recognition. *IVC*, 14(3):209–254, 1996.