Synchronization of Lecture Videos and Electronic Slides by Video Text Analysis

Feng Wang
Department of CS
Hong Kong of University of Science & Technology
wfeng@cs.ust.hk

Chong-Wah Ngo
Department of CS
City University of Hong Kong
cwngo@cs.cityu.edu.hk

Ting-Chuen Pong
Department of CS
Hong Kong University of Science & Technology
tcpong@cs.ust.hk

ABSTRACT

An essential goal of structuring lecture videos captured in live presentation is to provide a synchronized view of video clips and electronic slides. This paper presents an automatic approach to match video clips and slides based on the analysis of text embedded in lecture videos. We describe a method to reconstruct high-resolution video texts from multiple keyframes for robust OCR recognition. A two-stage matching algorithm based on the title and content similarity measures between video clips and slides is also proposed.

Categories and Subject Descriptors

I.2.10 [**Vision and Scene Understanding**]: Video Analysis

General Terms

Algorithms, Design, Management, Experimentation

Keywords

Lecture videos, Video text analysis, Synchronization

1. INTRODUCTION

In the past few years, issues in multimedia authoring of live presentation have attracted numerous research attentions. Recent achievements include the software and hardware tools for recording live presentation and the standards for document presentation. Demonstrated systems include Classroom 2000 [1], BMRC lecture browser [11], and interactive virtual classroom [4].

The main streams of multimedia documents (MD) captured in live presentation include video and audio information. To support the semantic understanding and querying of MD for distance learning applications, numerous efforts have been attempted to structure the content of MD. These efforts include topics detection [5, 10], gesture analysis [7], multi-streams synchronization [8, 9] and presentation summarization [6]. In this paper, we focus issues on the automatic matching of MD content so as to provide a synchronized view of lecture video clips and electronic slides.

To date, synchronization remains primarily a manual and labor-intensive task. In most systems [1, 11], the instructors need to manually edit time stamp information in order to match the electronic slides with the relevant audio-video clips. To tackle this problem, several approaches have been proposed to automate the synchronization by matching the spatial layout and background color of slides and videos [8, 9]. Today, most electronic slides prepared in a presentation are designed by a same design template. The geometric and visual hints alone, in general, are not enough for reliable matching. They can easily create false matches. In [8], speech transcription and spoken document retrieval techniques are also employed for synchronization. Nevertheless, the result of synchronization is very dependent on the content of speech and a presenter's accent and pronunciation.

This paper presents an automatic way of synchronization by video text analysis. Correct recognizing of video texts is a difficult task [2]. However, we show in this paper that the recognition can be greatly improved by employing the super-resolution reconstruction of multiple video text boxes. We separate the matching process into two stages: title similarity measure and content similarity measure. This approach not only speeds up the matching time, but also enhances the quality of matching since the reconstructed high-resolution titles can always be reliably recognized. The hardware setup of our system is as follows. A camera is mounted in the lecture hall so as to capture the presenter and the projected electronic slides (see Figure 2 for examples). The camera is fixed and it stays stationary throughout the lecture. A presenter can move freely in front of the project screen.

2. VIDEO TEXT ANALYSIS

We employ the algorithm in [10] to temporally partition a lecture video into shots according to the topics of discussion. Here we refer to a shot as a sequence of frames that capture a same electronic slide. For each shot, multiple keyframes are extracted for video text analysis.

2.1 Text Detection

The aim of text detection is to localize the exact text region in videos. Some factors including complex background, text-like scenes and the contrast of foreground texts to background scenes, will affect the results of detection. The current algorithms for text detection can be separated into two categories: geometry-based and texture-based approaches.

MM'03, November 2–8, 2003, Berkeley, California, USA.

Compared with the latter,, the geometry-based approach is easier to implement and more efficient, but much attention should be paid to noises. In our system, we use a geometry-based algorithm [3] to detect text regions in the keyframes. It can work very well when the background scenes are not too complicated, which is usually the case in lecture videos.

The algorithm operates as follows. The LOG (Laplacian of Gaussian) is employed to detect edges. We obtain the rectangles surrounding the edges, and then an attribution set is computed for each rectangle. The attributes include the center, height and width of the rectangle, the edge intensity, the mean and variance corresponding to the foreground and background color distribution. After getting the edges and their attributes, the following criteria are used to exclude non-text regions: i) one or both dimensions of the text box are too large or small; ii) the edge intensity is too low; iii) the edge inside the region is too simple.

The remaining edges are regarded as belonging to some characters. Since each character/word may consist of several edges or components, a loop is done to combine all edges that belong to the same character/word. The attributes obtained are used to check whether they are possibly of the same character/word. A GMM (Gaussian Mixture Model) is used to represent background and foreground. Since characters in the same context share some common properties, they are used to analyze the layout and refine detection results.

The text detection result may vary for different keyframes from a shot. This is mainly due to the changes of lighting condition, shadow and the movement of the presenter, which may hide some texts. We integrate the results from multiple keyframes to get all the textboxes. Some of them may be lost in some frames. If a textbox is detected in one of the keyframes and it satisfies the above three criteria, then we add it to the textbox set. This may include some noise in the detected textboxes and add a little workload to OCR recognition, but usually will not affect synchronization.

2.2 Super-Resolution Reconstruction

The main problem of recognizing video texts is the poor visual quality due to low image resolution. In our lecture videos, the height of a character is usually no more than 10 pixels which is too small for the commercial OCR systems. To improve the resolution, we employ the super-resolution based approach. Our approach is mainly lain in two aspects: i) linear interpolation to expand a textbox, ii) multi-frames integration to smooth background scene while enhancing the contrast of foreground texts to background scene.

Denote L as a low resolution textbox, and $\mathcal{S}$ as the high-resolution textbox of L. Let (X, Y) as the pixel index to $\mathcal{S}$ and (x, y) as the pixel index to L. The relationship between $\mathcal{S}$ and L is

$$\mathcal{S}(X,Y) = L(\frac{X}{a}, \frac{Y}{a}) = L(x^{'}, y^{'}) \quad (1)$$

where a is the interpolation factor, $(x^{'}, y^{'})$ is a subpixel index to L, and $x \leq x^{'} < x+1$ and $y \leq y^{'} < y+1$. By linear interpolation, we have

$$\begin{aligned} L(x, y^{'}) &= L(x,y) + (y^{'} - y) \times \\ &\quad (L(x, y+1) - L(x,y)) \end{aligned} \quad (2)$$

$$\begin{aligned} L(x+1, y^{'}) &= L(x+1,y) + (y^{'} - y) \times \\ &\quad (L(x+1, y+1) - L(x+1,y)) \end{aligned} \quad (3)$$

By further manipulating the above equations, we have

$$\begin{aligned} &\mathcal{S}(X,Y) = L(x^{'}, y^{'}) \\ &= L(x, y^{'}) + (x^{'} - x) \times (L(x+1, y^{'}) - L(x, y')) \end{aligned} \quad (4)$$

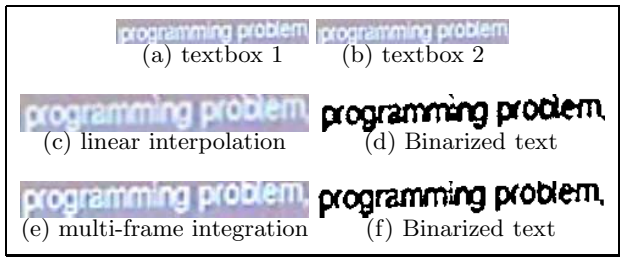
After linear interpolation, the final high-resolution textbox is obtained by integrating the results of text boxes obtained from multiple keyframes. The approach can enhance the foreground and background contrast. Let $\mathcal{S}_k$ as the high-resolution textbox of k^{th} keyframe, we compute the statistical information of these text boxes as follow

$$\mu_k(X,Y) = \frac{1}{|w|} \times \sum_{p,q \in w} \mathcal{S}_k(X-p, Y-q) \quad (5)$$

$$\mu(X,Y) = \frac{1}{k} \times \sum_{k} \mu_k(X,Y) \quad (6)$$

$$\sigma(X,Y) = \frac{1}{|w|} \times \max_{k} \sqrt{\sum_{p,q \in w} \{\mathcal{S}_k(X-p, Y-q) - \mu_k(X,Y)\}^2} \quad (7)$$

where w is a 5×5 local support window and $|w|$ is the cardinality of the window. Denote $\mathcal{S}^{'}$ as the final high-resolution textbox. We update the pixel values in $\mathcal{S}^{'}$ based on the computed statistical information. If $\sigma(X,Y)$ is smaller than a predefined threshold, $\mathcal{S}^{'}(X,Y) = \mu(X,Y)$. Otherwise, $\mathcal{S}(X,Y) = \min_k \mathcal{S}_k(X,Y)$ or $\mathcal{S}(X,Y) = \max_k \mathcal{S}_k(X,Y)$ by guessing whether $\mathcal{S}(X,Y)$ lies on a character. The guessing is done by checking the pixel values outside a small region of the low-resolution text boxes. Figure 1 shows the difference of reconstructed text boxes before and after multi-frame integration. As shown in the figure, multi-frame integration can enhance the quality of text binarization.



(a) textbox 1 (b) textbox 2
(c) linear interpolation (d) Binarized text
(e) multi-frame integration (f) Binarized text

Figure 1: (a) & (b) are text boxes in low resolution, (c) & (e) are the reconstructed high resolution text boxes, (d) & (f) are the segmented text boxes.

2.3 Text Binarization

Since most OCR systems use binary images as input, binarization is a preprocessing step of text recognition. Given a high-resolution text box, the task is to determine whether the pixels belong to foreground characters or just lie in background scene. The high resolution texts usually have distinguishable colors between the foreground and background, and also have a high intensity contrast in a gray scale image. This makes it easy to segment text and to describe the character using marginal distribution in a color space.

We utilize R/G/B/H/I components for text binarization. The foreground mean μ_f, background mean μ_b, foreground variance σ_f, and background variance σ_b are calculated for

each component. Then the GMM (Gaussian mixture model) parameters of a text box are calculated and they can reflect how well each component is in segmenting and describing character properties. Each component is associated with a confidence as follows:

$$C_i = \frac{|\mu_b^i - \mu_f^i|}{\sigma_b^i + \sigma_f^i} \quad (8)$$

$$C_H = \frac{\min\left(|\mu_b^H - \mu_f^H|, 256 - |\mu_b^H - \mu_f^H|\right)}{\sigma_b^H + \sigma_f^H} \quad (9)$$

where $i = \{R, G, B, I\}$. The higher the value C, the more confident the corresponding component. The component with the highest confidence is selected to carry out the segmentation of foreground texts and background scene.

The binarized text boxes are fed to OCR system for recognition. In our experiment, we use the commercial system in [13] and the recognition results can be found in Table 1.

3. MATCHING VIDEOS AND SLIDES

The extracted texts from videos are used to synchronize the videos and electronic slides. The texts from videos and slides are separated into titles and contents. The similarity between a shot and a slide is based on the title and content similarities. Given a slide, a video shot with the highest similarity is linked as its associated video clip.

Both title and content similarities are based on word matching. First, the extracted texts of title and content are separated into a list of words. Given two words w_1 and w_2, the edit distance is calculated. The definition and algorithm to compute the edit distance of two strings can be found in [12]. We define the matching of two words $M(w_1, w_2)$ as

$$\begin{cases} 1 & \text{if } max(Ed(w_1, w_2), Ed(w_2, w_1)) < \frac{min(len(w_1), len(w_2))}{4} \\ 0 & \text{otherwise} \end{cases}$$

where $len(w)$ is the length of a word. We say w_1 and w_2 are matched if $M(w_1, w_2) = 1$.

3.1 Title Similarity

Titles usually have a larger font size, as a result, can be recognized with higher possibility. Let W_v and W_s denote the word sets, respectively, from a video shot v and from a slide s, the set of the matched words, W_m, between W_v and W_s is defined as

$$W_m = \{w_1 | w_1 \in W_s, \exists w_2 \in W_v, M(w_1, w_2) = 1\}$$

The similarity between the titles of a video shot v and a slide s is then defined as follows:

$$Sim_T(v, s) = \frac{1}{2} \times \left\{ \frac{\sum_{w_1 \in W_m} len(w_1)}{\sum_{w_2 \in W_s} len(w_2)} + \frac{\sum_{w_1 \in W_m} len(w_1)}{\sum_{w_2 \in W_v} len(w_2)} \right\} \quad (10)$$

Here a longer matched word will contribute more to the similarity. If there is no title in either v or s, $Sim_T(v, s) = 0$.

3.2 Content Similarity

The content matching is similar to title matching. However, compared with titles, the size of characters in content is usually smaller and with lower visual quality. The recognition of content characters is usually not reliable. To avoid using the wrongly recognized characters for similarity measure, the content similarity is defined as

$$Sim_C(v, s) = \frac{\sum_{w_1 \in W_m} len(w_1)}{\sum_{w_2 \in W_s} len(w_2)} \quad (11)$$

where W_v, W_s and W_m have the similar meaning as Eqn(10), however, the words are not of the titles, but the content.

To reduce the amount of computation, content similarity of a shot and a slide is performed only when the title similarity between them is higher than 0.7 or there is no title in the shot or slide. The final similarity between a shot and a slide is defined as the sum of the title and content similarities.

4. EXPERIMENTS

We conducted experiments on three videos. The duration of each video is about 45 to 60 minutes, and each one displaying about twenty slides. For each shot in the videos, we evenly extract 5 keyframes along the time dimension. The textboxes from multiple frames are integrated and reconstructed as one high-resolution textbox before text binarization. The binary textboxes are then fed to OCR system. Figure 2 shows the detected text boxes of several keyframes. We can see that when the background is not too complicated, the text detection algorithm works well. Some noise may be included if the text connects with other edges.

Figure 3 shows the binarized high-resolution textboxes of a keyframe. In fact, either low or high-resolution, most of the characters in titles can be segmented correctly. The difference lies in two aspects: i) the edges of high-resolution characters are much smoother; ii) the adjacent characters are better separated in high resolution textboxes. These two factors can make great impact on OCR recognition. Compared with titles, the texts in content are usually more difficult to segment due to the small character size and over-illumination. Nevertheless, the results from high-resolution textboxes are much better than low-resolution ones.

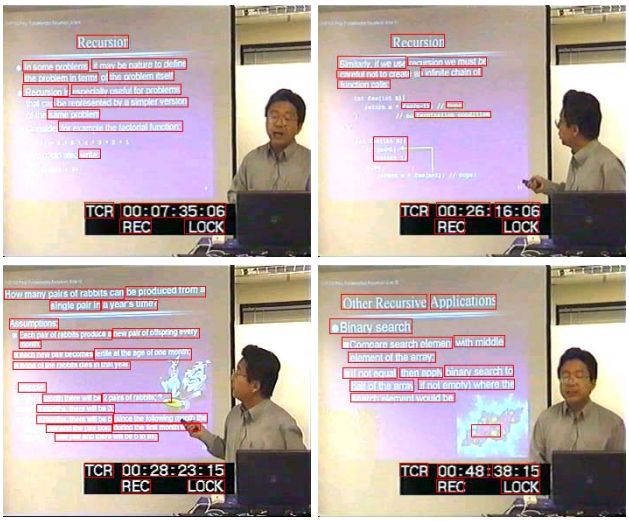


Figure 2: Experimental results for video text detection.

Tables 1 and 2 compare the OCR results for the high and low resolution texts. The recognition accuracy for high-resolution titles is about 80% to 90%, much better than the accuracy of 20% to 40% for low resolution. The recognition of texts in the main content is a difficult task. In our experiment, due to the low video quality, more than half of

Table 1: Results of video text recognition (High Resolution) N_g: number of ground-truth characters, N_c: number of correctly recognized characters, N_{ocr}: number of characters output by OCR, N_h: number of characters recognized by human.

Lecture	*Title*					*Content*					
Video	N_g	N_c	N_{ocr}	Recall	Precision	N_g	N_c	N_{ocr}	N_h	Recall	Precision
1	620	494	582	0.80	0.85	4117	432	1660	1586	0.27	0.26
2	230	218	230	0.95	0.95	3162	739	2037	1388	0.53	0.32
3	560	515	552	0.92	0.93	5058	1124	2029	1875	0.59	0.55

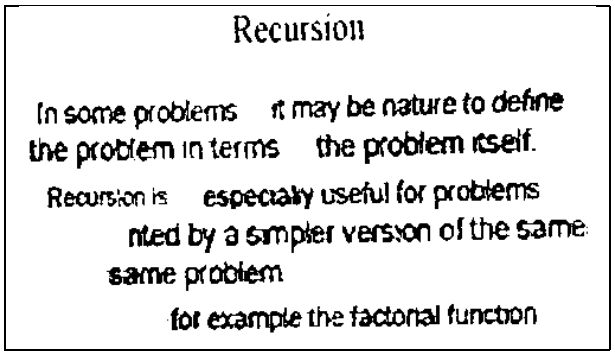


Figure 3: High-resolution text of top left image in Fig 2.

the characters are indeed not recognized by human. The OCR fails to recognize almost all the low resolution characters. Approximately 30% to 60% of characters that are recognized by human are successfully recognized by the OCR when the high-resolution characters are reconstructed. To measure the performance, we compute the value of *recall* as $\frac{N_c}{N_h}$, where N_c is the number characters recognized by OCR and N_h is the number of characters recognized by human.

Table 3 shows the results of synchronizing video shots and electronic slides. Each slide is matched with a video shot that has the highest similarity between them. The performance is evaluated by $accuracy = \frac{N_c}{N_s}$, where N_c is the number of slides that can be correctly matched and N_s is the total number of slides that appear in the presentation. By just using video text for synchronization, we achieve the accuracy of approximately 80% to 90% for the three tested videos. The reasons for mis-match are: i) the titles or contents of some slides are similar, ii) not enough texts are extracted or recognized, especially for those slides with no titles, the texts from main contents are too few for matching.

Table 2: Video text recognition (Low Resolution)

Lecture	*Title*		*Content*	
Video	Recall	Precision	Recall	Precision
1	0.19	0.69	0.00	0.12
2	0.22	0.58	0.00	0.00
3	0.43	0.78	0.00	0.10

5. CONCLUSION

We have presented a novel approach for synchronizing lecture video shots and electronic slides. Through experiments, we show that the super-resolution reconstruction of video texts can have great impact on the results of character recognition and slide synchronization. Empirical results also indicate that the proposed matching algorithm based on title and content similarity is robust and effective.

Table 3: Synchronization results

Lecture Video	Total Number of Slides	# of correctly matched slides	Accuracy
1	25	23	92%
2	23	19	82.6%
3	17	14	82.4%

Acknowledgment

The work described in this paper was supported in part by a grant from City University of Hong Kong (Project No. 7100249), a RGC Grant CityU 1072/02E (Project No. 9040693) and grants from Hong Kong University of Science and Technology (DAG01/02.EG16, SSRI99/00.EG11 and VPAAO01/02.EG01).

6. REFERENCES

[1] G. D. Abowd *et. al.*, "Teaching and Learning as Multimedia Authoring: The Classroom 2000 Project," *ACM Multimedia*, pp. 187-198, 2000.

[2] H. Aradhye *et. al.*, "Study of Embedded Font Context and Kernel Space Methods for Improved Videotext Recognition," *IBM Research Report RC 22064*, 2001.

[3] X. Chen *et. al.*, "Automatic Detection of Signs with Affine Transformation," *IEEE. WACV*, Dec, 2002.

[4] S. G. Deshpande & J.-N. Hwang, "A Real-time Interactive Virtual Classroom Multimedia Distance Learning System," *IEEE Trans on Multimedia*, vol. 3, no. 4, pp. 432-444, 2001.

[5] D. Phung, S. Venkatesh & C. Dorai, "High Level Segmentation of Instructional Videos Based on Content Density," *ACM Multimedia*, 2002.

[6] L. He *et. al.*, "Auto-Summarization of Audio-Video Presentations," *ACM Multimedia*, pp. 489-498, 1999.

[7] S. X. Ju *et. al*, "Summarization of Videotaped Presentations: Automatic Analysis of Motion and Gesture," *IEEE Trans on CSVT*, vol. 8, no. 5, pp. 686-696, 1998.

[8] T. F. S. -Mahmood, "Indexing for topics in videos using foils," *Int. Conf. CVPR*, pp. 312-319, 2000.

[9] S. Mukhopadhyay & B. Smith, "Passive Capture and Structuring of Lectures," *ACM Multimedia*, 1999.

[10] C. W. Ngo, T. C. Pong & T. S. Huang, "Detection of Slide Transition for Topic Indexing," *Proc. ICME*, 2002.

[11] L. A. Rowe & J. M. Gonzlez, "BMRC Lecture Browser," http://bmrc.berkeley.edu/frame/projects/lb/index.html

[12] E. Ukkonen, "Algorithms for Approximate String Matching," *Information and Control*, 100-118, 1985.

[13] OmniPage Pro 12, http://www.scansoft.com/omnipage/