Video Text Detection and Extraction Using Temporal Information

By

LUO Bo

A Thesis Submitted in Partial Fulfillment

of the Requirements for the Degree of

Master of Philosophy

in

Information Engineering

June 2003

Abstract

Text detection and recognition in images and videos is to automatically convert these graphically included visual content into text characters, which can be directly processed by text document processing techniques. Although text in images and videos are easily distinguishable by human eyes, there is usually no significant difference in gray levels between the text and surrounding background. Therefore special algorithm has to be designed for text detection and recognition. It is an important step for information retrieval in video databases. It enables automatic access to high-level semantic content of visual data.

Video caption detection and recognition is similar to text detection and recognition in images. But it suffers from the low resolution of video frames and the dynamically changing background. On the other hand, video captions usually remain the same in a number of consecutive frames, thus contain abundant temporal information. In this thesis, we extract text information in video by fully utilizing this temporal information. We define temporal feature vectors to describe the temporal behavior of each pixel across a number of consecutive frames. We first divide a video stream into overlapped slices with fixed number of frames. Using a supervised classification of temporal feature vectors extracted for each pixel in these video slices, each slice is represented by a binary abstract image. By analyzing the statistical pixel changes in the

sequence of abstract images, the appearance frames and disappearance frames of captions are located. We can then divide the video into fractions that contain stable captions. For each fraction, a final classification is carried out to extract the indexing key frames with refined captions in order to create a summary of the video segment. These frames are of high quality and can be sent to an OCR system for recognition. Our algorithm does not make any assumption on the shape of the caption, i.e. we do not need the captions to be monochromatic, in horizontal of direction, constant size, and font. Experimental results show our method is highly effective.

摘要

視頻和圖像中包含的文字信息和圖像背景融合在一起。雖然人在理解過程中可以將它們辨識開來，但是文字和背景在計算機數據結構上是不可區分的。視頻和圖像中的文字檢測、提取和識別技術即是自動地將這些融入背景的的圖形化文字信息轉化成為文本字符，以使得計算機可以使用普通文字處理的方法對其進行加工處理。它從視覺數據中提取了高層次語義信息，因此成為信息檢索和視頻數據庫的重要環節。

視頻中的文字提取和圖像中的文字提取技術相似。但是視頻信息具有分辨率相對較低和背景更為複雜的特點，使得文字提取更加困難。另一方面，視頻中的文字總是持續一段時間，即在相鄰的一系列圖像幀中出現，因此帶來了大量的時域特徵信息。

在本文中，我們完全地利用這些時域信息，從視頻裏提取文字。我們定義了時域特徵向量來描述視頻中每一個象素點在相鄰幀中表現的時域地特性。首先將視頻流分為固定長度的互有重疊的片段，使用有監督聚類的方法將每一個片段的象素點分為文字和背景兩類，這樣對每一片斷都建立了一幅單色的摘要圖像。通過對象素點在這些摘要圖像間的變化的統計分析，我們定位出文字在哪一幀出現和消失，並以此將整個視頻流分段，使得每一段都包含有穩定的文字。最後再次從每一段中提取時域特徵向量並聚類生成一幅含有文字的單色圖像。這樣就為整個視頻流建立了一系列概要圖像，其中每一幅都包含了已

被分割的文字信息。這些被檢測和提取出的文字圖形具有清晰的外觀，可以被光學字符識別（OCR）系統識別出來。

我們的算法不需要對視頻中包含的文字字型做出任何假設，也就是說，我們不需要假定文字是單色、水平、固定字號、某種字型或是出現在某一特殊位置。實驗證明我們的方法是非常有效的。

To My Family

Acknowledgments

First of all, I would like to take this chance to express my heartily thanks to my supervisor Professor Sean Tang, for his patient and professional direction in the past two years. He not only provides me with valuable ideas, insights, and comments, but also teaches me the way of thinking and researching. It is a great pleasure and fortune to have him as my supervisor.

I would also like to thank Dr. Jianzhuang Liu, for numerous discussions and suggestions, also for his carefully revision of my ICIP paper.

Also, I want to express my thanks to all the members of the Multimedia Lab. They are Ph.D candidates Mr. Feng Lin, Mr. Lifeng Sha, Mr. Zhifeng Li, Mr. Feng Zhao, and M.Phil candidates Mr. Xiaogang Wang, Miss Hua Shen, Mr. Tong Wang and Mr. Dacheng Tao. We together create a friendly and competitive research atmosphere, from which we all benefit.

Finally, I would like to give my sincere thanks to all the members of my family, for their love, care and support. I also have to thank my fiancée Cathy Li, for all the love and understanding in the past 7 years.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

KLT	Karhunen Lòeve Transform
MAE	Mean Absolute Error
MSRE	Mean Square Root Error
OCR	Optical Character Recognition
PCA	Principal Component Analysis
QSDD	Quantized Spatial Difference Density
TFV	Temporal Feature Vector
VHS	Video Home System

Chapter 1 Introduction

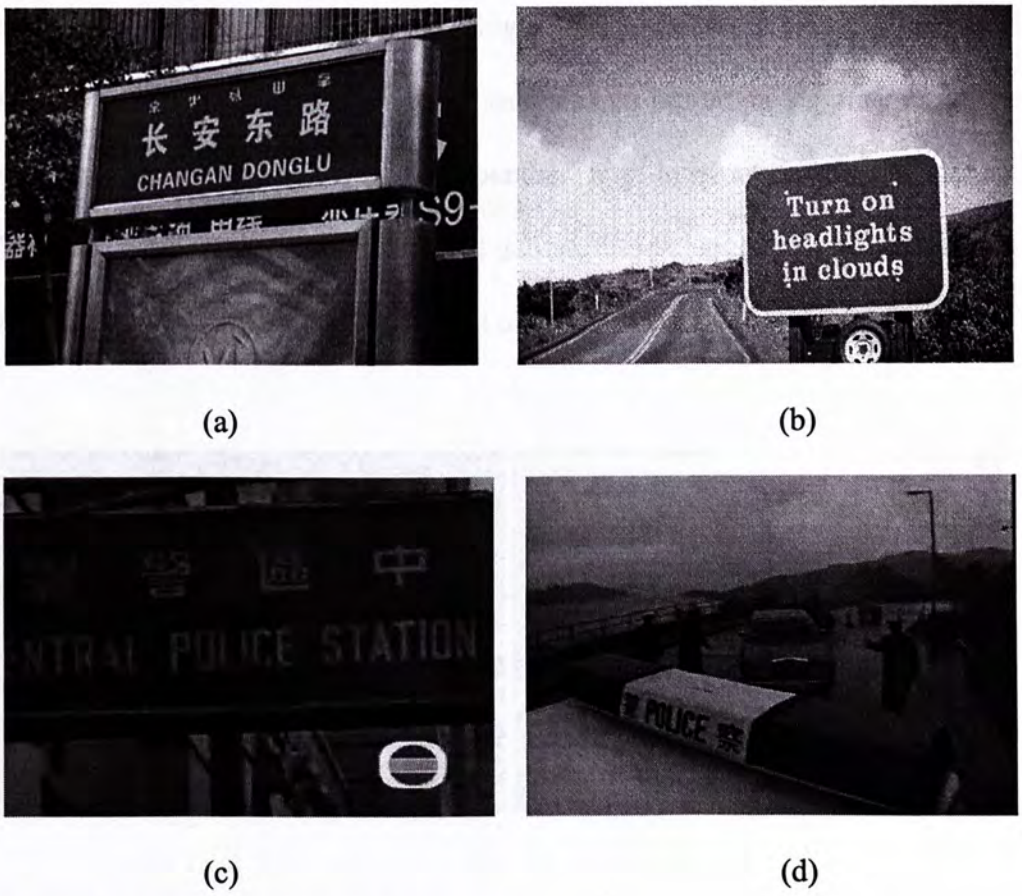
1.1 Background

With the rapid growth of multimedia content, research and applications in related areas such as database, digital library, content-based multimedia indexing and retrieval are becoming more and more active in recent years. Early indexing and retrieval schemes mainly focus on text document. Images and videos are first annotated by text terms and then the text-based Database Management Systems are used to perform image retrieval, [3][4]. In this framework, manual multimedia document annotation is extremely laborious and the visual content of images and videos are difficult to be described precisely by a limited set of text terms. To overcome these difficulties, content-based multimedia retrieval systems index images and videos by their visual content, such as color, shape, texture, motion etc [1][2][5][6][7][8][9]. Complement to the low level features, researchers are beginning to use such high level features as text in video for video indexing because of the rich content information contained in them [32][33][38][39][40].

1.2 Text in Videos

There are two classes of text embedded in video frames: the scene text and the graphic text [14]. Scene text appears in the video scene as an integral part of

the scene content. Typical scene texts are traffic signs, street nameplates, car plates, and text on billboards. Figures 1.1 gives some examples of scene text.



(a) (b)

(c) (d)

Figure 1.1 Examples of scene text.

Figure 1.1 (a) is a multi-language street nameplate and (b) shows a traffic sign with text. Figure 1.1 (c) and (d) are video frames with scheme texts. We can see their meanings might not be consequentially tied with the video contents, thus they are usually not used for content-based video retrieval. On the other hand, detection and recognition of scene text, especially real-time schemes,

have been proposed for video surveillance, automatic assistance of disabled, and other applications [10][11][12][13].

Graphic text contains the mechanically embedded characters, such as news video captions and movie subtitles. Figure 1.2 gives some examples of these superimposed captions. Graphic text serves as an important supplement of the audio-visual content and provides abundant high-level semantic information. Efforts have been made to detect and extract these characters automatically to enable access to the high-level content of video data.

(a) (b)

(c) (d)

Figure 1.2 Examples of graphical text.

1.3 Related Work

Current text detection and extraction schemes can be generally grouped into three categories [16] – connected component based methods [21][23][25][29], texture classification based methods [14][15][36], and edge detection based methods [20][24][26][27][28][37]. We give a brief review of these methods in this section.

1.3.1 Connected Component Based Methods

Connected component based methods use connected component analysis to process images and video frames that have text of uniform color or brightness. In [21], Jain and Yu carry out multi-value image decomposition, foreground image generation and selection to decompose images. A color space reduction is used to process color images. Finally, they apply connected component analysis on the decomposed binary images. In [23], Lienhart and Stuber use a split and merge algorithm on a hierarchically decomposed frame to find the homogeneous text regions. They also make use of contrast, fill factor, and width-to-height ratio to enhance the segmentation. The text is assumed to be monochromatic, rigid, of high contrast with the background and of restricted width-to-height ratio. In [25] and [29], Shim et al develop a generalized region labeling (GRL), and use it to extract homogenous text regions. For connected component based methods, the computation is usually low and the localization

accuracy is high. But these methods have difficulties in handling the instances that characters touch each other, or characters touch foreground objects.

1.3.2 Texture Classification Based Methods

Texture classification based methods utilize the fact that text have specific color or brightness, and are formed by strokes. Thus the text area is regarded as a distinct texture, which is different from the background texture. Texture based methods make use of these observations to distinguish text from background using supervised or unsupervised classifications of texture. Jain and Bhattacharjee [15] use Gabor features to represent the texture surroundings of each pixel. Then, unsupervised clustering is used to distinguish text and non-text pixels. Li et al [14] use small windows (typically 16×16) to scan through each video frame and compute texture features (wavelet features are selected) of each window. Finally they use a neural network to provide supervised classification of the windows, thus each window is classified as text or non-text block. Texture based methods are more accurate, but are often sensitive to the style of text appearance, e.g. color and size. And they are usually expensive to compute.

1.3.3 Edge Detection Based Methods

Edge detection based methods rely on the fact that text regions usually have rich stroke edges or high frequency components. Lienhard and Wernicke [26][27] propose a generic and scale-invariant scheme that makes use of edge information. They calculate the edge orientation image from the gradient image

of the input, then use a neural network to classify 20×10 regions of the edge orientation image into text or non-text class. They recursively reduce the image at a factor of 1.5, then apply the fixed scale text detector, thus the method is able to detect text of different scales. In [20], Agnihotri and Dimitrova propose a seven-stage approach to detect text in VHS quality video. The text detection steps are: channel separation, image enhancement, edge detection, edge filtering, character detection, text box detection, and text line detection. In [24], Sato et al develop a system to index news video using the recognized captions. They consider the low resolution and complex background of video frames, and propose a combination of sub-pixel interpolation on individual frames and multi-frame integration across time to enhance the captions thus improve the recognition rate. They report a text region detection rate of 89.6%, words detection rate of 76%, character recognition rate of 83.5% (based on the correctly detected characters) and word recognition rate of 70.4% (based on the correctly detected characters) on seven 30-minute CNN news programs. In [28], Hua et al propose a text detection algorithm based on corner detection with the observation that text region are typically rich of corners as well as edges. They detect the corners in video frames to build a corner map, and then detect the intensive areas in the corner map. The candidate text areas are found through corner merging. Finally they combine the vertical and horizontal edge information with the corner information for text line decomposition and text box verification to create final text boxes. In [16] and [17], Tang et al first divide video sequence into camera shots, then propose a quantized spatial

difference density (QSDD) method in each shot to detect the caption transition frame. The difference image between caption appearance frame and its previous frame contains enhanced caption, which can be located more easily. Caption regions are first separated into characters using edge information, and then individual characters are segmented from background after a multi-frame enhancement. On a test data set of twelve 30-minute news video segments with Chinese captions, they achieve a caption line detection result of 97.44% precision and 99.56% recall, and character recognition rate of 85.82% for the first candidate and 92.10% for the top 3 candidates.

1.3.4 Multi-frame Enhancement

Most of the current text detection and extraction methods aim at still images or individual video frames. While some schemes claim to be designed to process text in video [14][16][17][20][22][23][24][25][26][28], they treat text in video frames the same way as that in still images [20][21][23][36]. This means each frame is regarded as one independent image, the temporal information is neglected while the low resolution and complex background seriously hurt the text detection and extraction performance.

Other existing methods use multi-frame enhancement to increase the contrast between background and captions. Frequently used enhancement algorithms include multi-frame averaging and maximal/minimal searching. Multi-frame averaging is to compute the average of all the fames with the same caption. Because captions always have fairly constant brightness values and they are usually very bright or dark. Their brightness values remain extremeness after

averaging, while those of the changing background tend to be softened. It is denoted as follows:

$$\bar{f} = \frac{1}{C} \sum_{f_t \in C_i} f_t \tag{1.1}$$

or

$$\bar{\gamma}_i = \frac{1}{C} \sum_{f_t \in C_i} \gamma_i(f_t), \tag{1.2}$$

where C_i is the frame cluster with the same caption, $f_t \in C_i$ is a frame of that cluster. Equation 1.1 computes the average of all frames, while equation 1.2 focuses on each of the small areas with characters, γ_i.

Maximal/minimal pixel search is to find the minimal/maximal brightness value of each pixel in all the frames. When there is a bright caption, a minimal pixel search is applied. Otherwise, when there is a dark caption, a maximal pixel search is applied. Minimal pixel search is represented as follows.

$$f_{\min} = \{\phi_{x,y} = \min_{f_t \in C_i}(\phi_{x,y}(f_t)\} \tag{1.3}$$

or

$$\gamma_{\min} = \{\phi_{\gamma_i,x,y} = \min_{f_t \in C_i}(\phi_{\gamma_i,x,y}(f_t)\} \tag{1.4}$$

In Eq. 1.3 and 1.4, $\phi_{x,y}$ denotes the pixel at location (x, y) of the image or image block. These methods search the minimal/maximal gray-level values of each of the locations (x, y) along that segment, and assign it to the same location (x,y) of the output image.

Recently, Hua et al [30] propose a multi-frame integration method, which first apply a multiple frame verification (MFV) to obtain frames with the same text,

and use high contrast frame selection and high contrast block averaging to enhance the caption text. Then they use a block adaptive thresholding to segment the characters. They reported a 26% increasing of character recognition rate.

Although the above methods employ some of the temporal information to enhance the text, it is obvious that the abundant temporal information contained in video frames is not fully utilized.

1.4 Our Contribution

In [18], we detect the caption transition frames using Quantized Spatial Difference Density (QSDD) method in each camera shot, then locate the caption blocks and trace the (dis)appearance of captions, thus each segment is divided into slices, with a stable caption. Then we propose the *temporal feature vectors* (TFV) to fully describe the temporal behavior of each pixel along a slice. By using a supervised classification, the captions are segmented from the background.

In [18], the caption detection and tracing is still based on spatial methods. In [19], we present a method to achieve the entire caption detection and extraction by taking full advantage of temporal information. First we create a binary abstract sequence from a video segment. By analyzing the statistical pixel changes in the sequence, we can effectively locate the (dis)appearing frames of captions. Finally we extract the captions to create a summary of the video segment.

Chapter 2 Caption Segmentation

2.1 Temporal Feature Vectors

As described in Chapter 1, with the observation that video text stay the same over a number of consecutive frames, several methods have been proposed to enhance the strokes thus to improve the recognition rate of video OCR. These methods first detect captions in some key frames, and then trace the detected caption to locate its appearance and disappearance. Thus a video segment with the same caption is obtained. Then the average, variance or minimal/maximal values of the segment are computed. Since captions normally have very bright or dark appearances and their pixels normally have relatively stable brightness values, these methods enhance the visual quality of caption text to a certain degree. After the enhancement, these methods still need further steps to separate the text from background before the characters can be used for recognition.

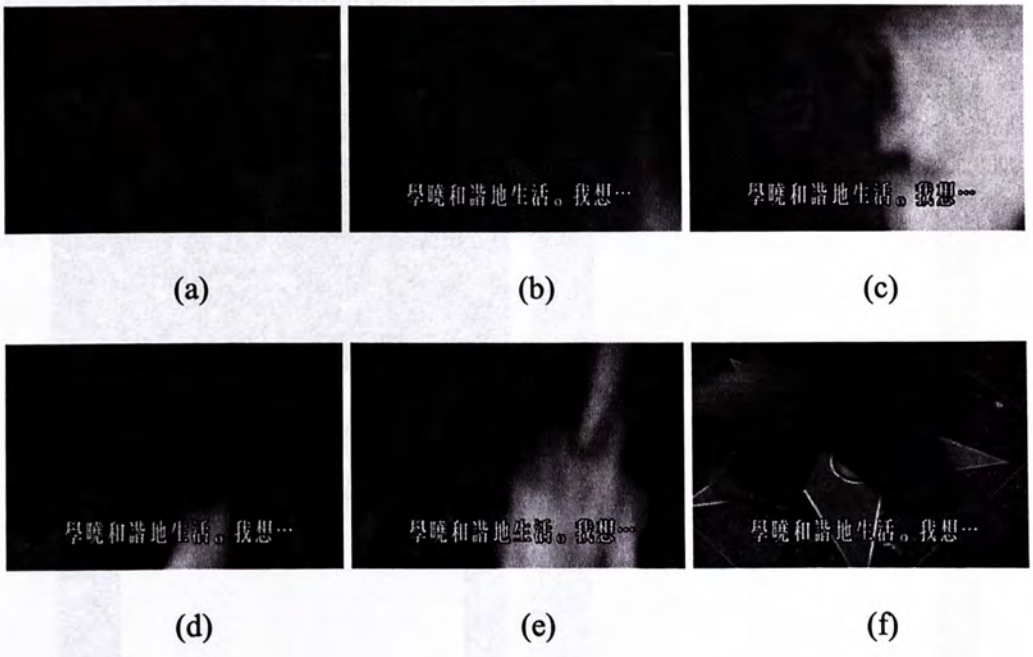
Although these caption enhancement methods lead to improvement in the segmentation performance and recognition rate of caption text, it is obvious that the rich temporal information is not fully utilized. In order to take advantage of the temporal information, we trace the gray-level of each pixel in

a video segment. For each pixel, we thus obtain a sequence of gray-level values, which form a vector. This vector fully describes the temporal characteristics of that pixel during the period. It can be used as feature vector to represent the pixel. We name it the *temporal feature vector (TFV)*.

In a sequence of frames extracted from a video segment, let $G(x,y,t)$ be the gray-scale level of pixel (x, y) at time t. For a pixel at (x_0, y_0), $G(x_0, y_0, t)$ shows how the grayscale level of that pixel changes in time through the segment. It is defined as the temporal feature vector of the pixel,

$$\overline{TFV}\Big|_{\substack{x=x_0 \\ y=y_0}} = G(x_0, y_0, t) = [G(x_0, y_0, t_0), G(x_0, y_0, t_1), \cdots, G(x_0, y_0, t_n)]. \quad (2.1)$$

In this way, each pixel is described by an n-dimensional vector, while n is the total number of frames in the video segment.



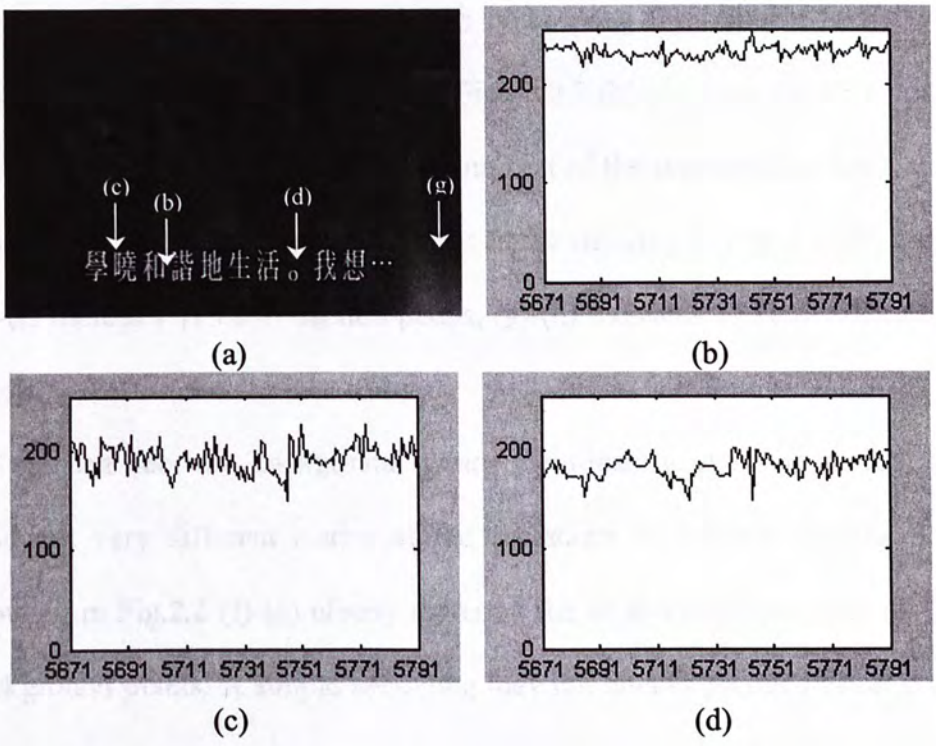
(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 2.1 Sample frames from a 4-second segment of a movie.

Figure 2.1 shows some sample frames of a video segment. In the period of total 123 consecutive frames, a caption appears at the 2nd frame, stays for around 4 seconds (121 frames), and then disappears at the 122nd frame. The frame numbers of the samples shown in Fig. 2.1 are 5670, 5671, 5685, 5703, 5723, 5745, 5755, 5765, and 5792. They demonstrate the whole process from appearance, to disappearance of one caption. With the segment, the background keeps changing while the caption stays the same.



(a) (b)

(c) (d)

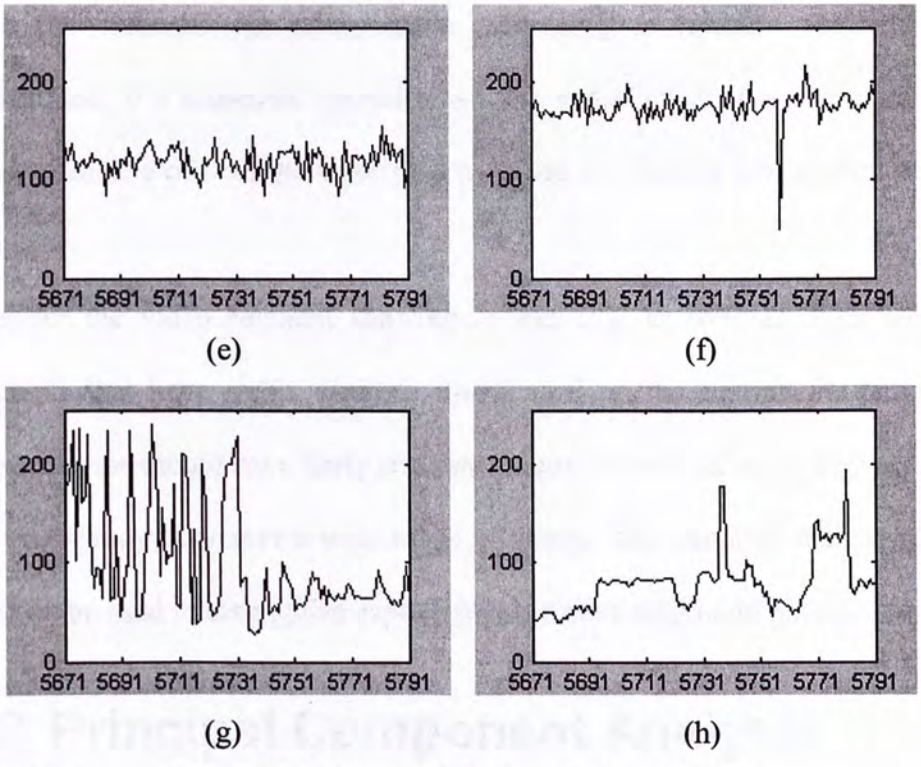


Figure 2.2 Examples of *temporal feature vectors (TFVs)*

Figure 2.2 (a) shows the frame of Fig. 2.1 (b), with some pixels pointed out. We trace the gray-scale value of each pixel along the segment to build the *temporal feature vector* of that pixel. Figure 2.2 (b)-(h) show the *TFVs* of the indicated pixels. They are extracted from part of the segment that has a stable caption, i.e. excluding the first and last frame showing in Fig. 2.1. Figure 2.2 (b)-(f) indicates *TFV*s of caption pixels, (g)-(h) indicates *TFV*s of background pixels.

As we can see, the background varies significantly over the period, thus produces very different clarity of the characters in different frames. TFVs showing in Fig.2.2 (f)-(h) clearly represent the time-domain property of these background pixels. A simple averaging may not always produce better results

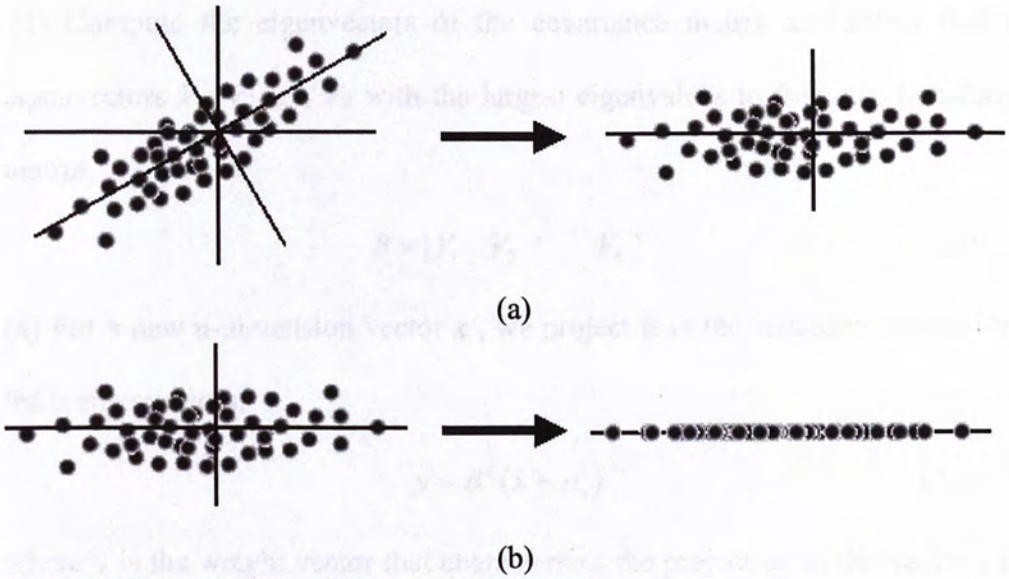
since the character can have similar averaging or variance value to the background. If a minimum operation is used, some pixels on a character may be lost because of random noise thus reducing the already low quality of the character.

Consider the video segment showing in Fig. 2.1, as well as other similar segments that have stable captions, for a pixel on the caption, its temporal feature vector should have fairly constant values. For a pixel at the background, the vector may vary over a wide range of values. The temporal feature vector thus can be used to distinguish caption pixels from background pixels.

2.2 Principal Component Analysis

The temporal feature vectors extracted in Chapter 2.1 has n dimensions, where n denotes the number of frames of the segment. Usually, the length of the segment that contains the same caption varies from tens to hundreds of frames. The large number of dimensions causes difficulty in the classification of these vectors. Thus we need to compress the temporal feature vectors to gain efficiency. Principle Component Analysis (PCA) [34][35][31] is used for this purpose.

PCA uses *Karhunen Lòeve Transform* (KLT) to produce a set of projection vectors describing the data distribution, which is optimal in the sense of energy compaction. Figure 2.3 illustrates the procedure of PCA. Figure 2.3 (a) shows how the original data space is projected to the transform space. We only need to retain some dimensions that contain most of the energy. Figure 2.3 (b) shows this process.



(a)

(b)

Figure 2.3 Principal Component Analysis

The procedure of PCA can be described as follows [31].

Let $x_1, x_2, \ldots x_m \in x$ represent a set of n-dimension random vectors and μ_x be the mean vector.

$$\mu_x = E\{x\} = \frac{1}{m}\sum_{x_i \in x} x_i \tag{2.2}$$

(1) Form the n by m sample matrix

$$A = [\overline{x}_1 \quad \overline{x}_2 \quad \ldots \quad \overline{x}_m] = \begin{bmatrix} x_1(1) & x_2(1) & \ldots & x_m(1) \\ x_1(2) & x_2(2) & \ldots & x_m(2) \\ \ldots & \ldots & \ldots & \ldots \\ x_1(n) & x_2(n) & \ldots & x_m(n) \end{bmatrix} \tag{2.3}$$

where n denotes the length of each vector, and m is the number of vectors.

(2) Estimate the covariance matrix,

$$\begin{aligned} W = E\{(x-\mu_x)(x-\mu_x)^T\} &= \frac{1}{m}\sum_{i=1}^{m}\{(x_i-\mu_x)(x_i-\mu_x)^T\} \\ &= \frac{1}{m}(A-\mu_x)(A-\mu_x)^T \end{aligned} \tag{2.4}$$

(3) Compute the eigenvectors of the covariance matrix and select first k eigenvectors V_1, V_2, ..., V_k with the largest eigenvalues to form the transform matrix,

$$B = [V_1 \quad V_2 \quad \ldots \quad V_k] \tag{2.5}$$

(4) For a new n-dimension vector x', we project it in the subspace spanned by the k eigenvectors,

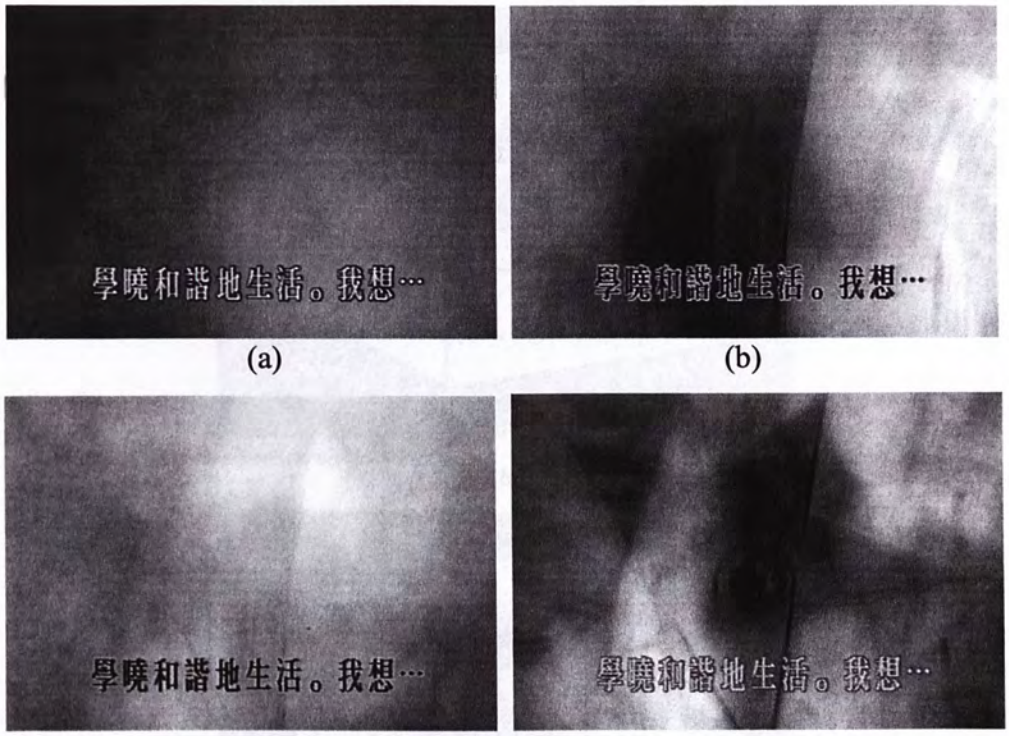
$$y = B^T(x' - \mu_x) \tag{2.6}$$

where y is the weight vector that characterizes the projection of the vector x in the subspace supported by the k eigenvectors.

The most prominent advantage of PCA is that it can remove the correlation between features thus reduce the feature vector dimension.

2.3 PCA of Temporal Feature Vectors

By using the temporal gray-scale vector as a feature vector, we retain all the information that can distinguish a caption pixel from a background pixel. To illustrate the vector difference between the caption and background, we use the principal component analysis method described above to compress the vector then show the first four principal components in Fig. 2.4.

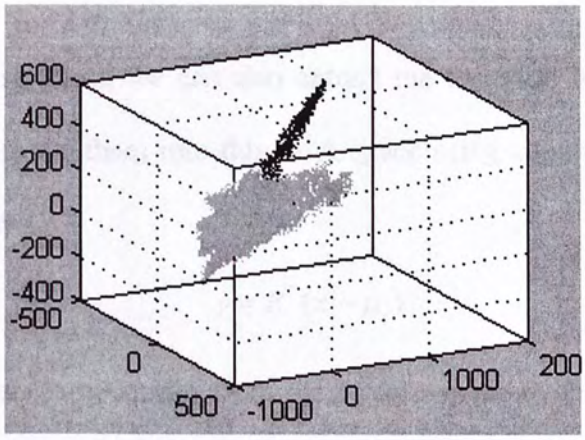


(a) (b)

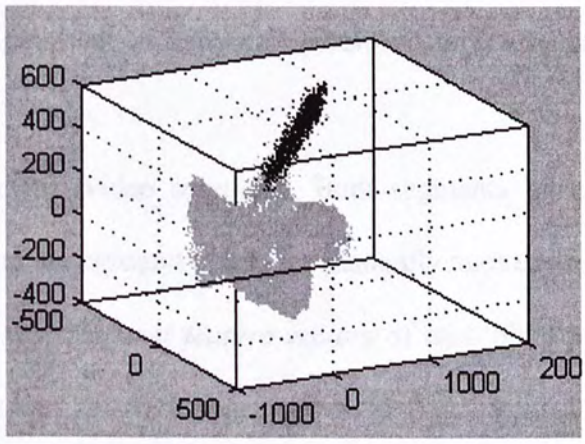
(c) (d)

Figure 2.4 First 4 principal components of temporal feature vectors rescaled to (0,255) showing through images.

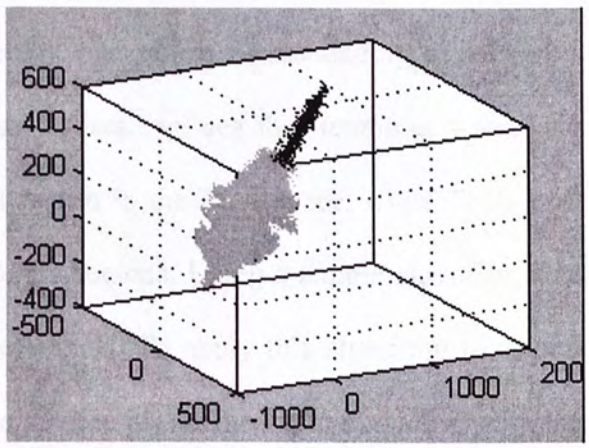
Figure 2.5 (a) illustrates the first three principle components in a 3-D coordinate with each point in the 3-D space corresponding to a pixel in the video frame. Black points denote caption pixels while gray points denote background pixels (caption pixels and background pixels are manually marked as ground truth). We can clearly see that the caption pixels and background pixels do not overlap with each other. They are distinctly separable in this 3-D space.



(a)



(b)



(c)

Figure 2.5 Feature vector distribution of caption and background. Feature vectors are extracted from different video clips and projected into the same PCA space.

For other video segment, we can also extract the temporal feature vectors of each pixel and project them into this PCA space using equation 2.7, which is rewritten as follows.

$$y = B^T(x' - \mu_x) \tag{2.6}$$

B denotes the transform matrix from the feature space to the PCA space, x' denotes a newly extracted temporal feature vector, and μ_x denotes the mean vector of the original set of temporal feature vectors, also known as training samples.

We obtain other two video segments. Both segments have stable captions. Caption pixels and background pixels are manually marked for reference.

Then we extract the *temporal feature vector*s of each pixel in these segments, and project them into the PCA space shown in Fig. 2.5 (a). Results are shown in Fig 2.5 (b) and (c). We also show the points corresponding to the caption pixels in black color, and points corresponding to background pixels in gray color. From these figures, we see that temporal feature vectors of captions gather at a small region in the PCA space, while TFVs of background pixels distribute in different regions. Using a simple classifier we can easily classify the two classes of pixels. To apply this algorithm to segment video captions automatically, we need to divide the video into slices with the same caption, i.e. to detect the appearance and disappearance of each caption. The following chapter describes an algorithm to detect the (dis)appearance of captions utilizing the temporal information.

Chapter 3
Caption (Dis)Appearance Detection

3.1 Abstract Image Sequence

If we keep on tracing the gray-level values through a longer segment, with several occurrences of caption (dis)appearances, we observe that at a caption appearing frame, a number of background pixels turn to caption pixels; likewise, at a caption disappearing frame, a number of caption pixels turn to background pixels. Based on this observation, we have designed the following process to create a sequence of images that represents these collective actions more clearly.

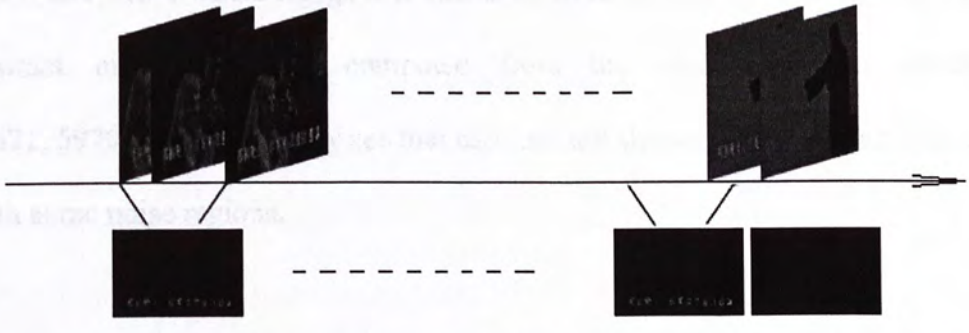
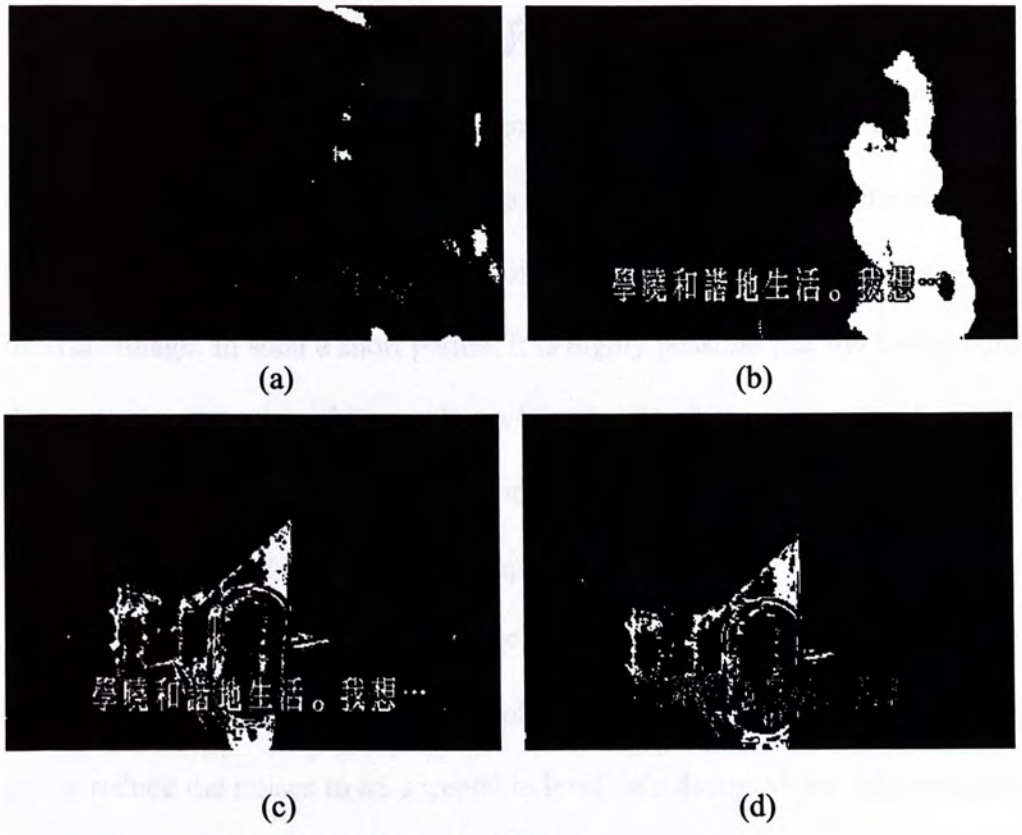


Figure 3.1 A brief demonstration of the process of extracting abstract image sequence.

First, we pick up a video slice of the first 30 frames (i.e. frames [F_1, F_{30}]), and apply the algorithm described in chapter 2 to cluster the pixels into caption and background, which result a binary image with "1" representing caption pixels. If a stable caption is contained in these frames, the caption pixels will be segmented and shown in the resulting binary image; otherwise, shown in the image are only some noise pixels, whose temporal feature vectors act like caption during this 30-frame-period. Then we move forward to another video slice at a step length of five frames (i.e. move from [F_1, F_{30}] to [F_6, F_{35}]) to compute another segmented binary image, which shows the caption status of the next slice (i.e. frames [F_6, F_{35}]). By repeating this process, a sequence of binary images of segmented captions is finally obtained. Each binary image I_i in the sequence represents abstract textual information of original frames [F_{5i+1}, F_{5i+30}]. We call it an *abstract image* sequence. Figure 3.1 gives a brief demonstration of this process as well as some examples of abstract images.

Figure 3.2 gives some examples of abstract images. They are computed from the video segment showing in Fig 2.1. Their abstract image IDs are 1134, 1135, 1153, and 1154, where image k is extracted from frames $5k+1$ to $5k+30$, e.g. abstract image 1134 is computed from the video slice of frames [5671, 5970]. We can clearly see that captions are shown in the abstract images with some noise regions.



(a) (b)

(c) (d)

Figure 3.2 Samples of *abstract images*.

The selection of the length of the video slice, i.e. number of frames in each period, is based on the assumption that each caption is present at least 1 second, within which there are 30 frames as designed in many major video standards. Thus any caption appearing in at least 30 frames will be shown in at least one image in the abstract sequence.

With the abstract image sequence, to detect the caption (dis)appearance is to analyze the statistical change of pixels in the sequence. Before this analysis, we need to refine the abstract images.

3.2 Abstract Image Refinement

From the extracted abstract images shown in Fig. 3.2, we observe that some background pixels are classified into captions. These errors are basically because we only took a video slice of 30 frames each time to compute an abstract image. In such a short period, it is highly possible that the background does not change too much, e.g. when a large bright object moves slowly across the background, many pixels remain bright during a 30-frame period and they might be falsely classified into caption pixels. Large number of false classifications in one abstract image will cause mistakes in the caption (dis)appearance detection. Consider both computation and feasibility, we only try to reduce the noises to an acceptable level. We designed the following two refinement methods.

3.2.1 Refinement One

Refinement one removes all $n \times n$ areas with more than or equal to NH caption pixels or less than or equal to NL caption pixels. Parameters n, NH and NL are preset constants. Although their values should be set depending on the status of the caption text, e.g. language, font, size, in our experiments, we simply set $a=3$, $NH=8$ and $NL=1$. This means remove all 3 by 3 areas with only one caption pixel, or more than seven caption pixels (i.e. with 1, 8 or 9 caption pixels.). These parameters work well in all test video slips with different caption styles. This means the values of these parameters are not sensitive to the style of captions.

In our experiments, refinement one is implemented in Matlab as follows.

```
mtxK= ones(n,n);
imgIConv= conv2(imgI,mtxK,'same');
imgIM=conv2(imgIConv>=NH|(imgIConv<=NL),mtxK,'same')
imgIRef1= imgI-imgIM;
```

First defines an $n \times n$ matrix as the convolution kernel. Convolution of the kernel with the binary abstract image results matrix *imgIConv* indicating the number of $n \times n$ neighbors of the corresponding pixel in the image. After applying a threshold to *imgIConv*, a second convolution of *imgIConv* and the kernel results matrix *imgIM* showing the areas to be removed. Finally we remove the pixels using *imgI-imgIM*.

To implement refinement one in VC, only a sequential scan of all the pixels in each abstract image is needed. The computation is lower.

3.2.2 Refinement Two

To gain better performance, refinement two first removes all connected caption areas (8-connection is used) with area size greater than N, another preset constant, then performs refinement one on the result image. This method certainly brings better performance, but it needs a connected area analysis, which is more expensive in computation.

3.2.3 Discussions

Figure 3.3 shows the refined version of abstract images in Fig. 3.2 (b) and (c). We can find that majority of wrongly classified background pixels are removed

while majority of true caption pixels still remain. Although a small number of true caption pixels are removed, it will not hurt the caption (dis)appearance detection performance, since the detection is based on statistics of the pixels. Figure 3.3 (a) and (c) show result of refinement one, while (b) and (d) are results of refinement two. From them, it is also clear that refinement two proposes better results than refinement one. This also results better caption (dis)appearance detection results.

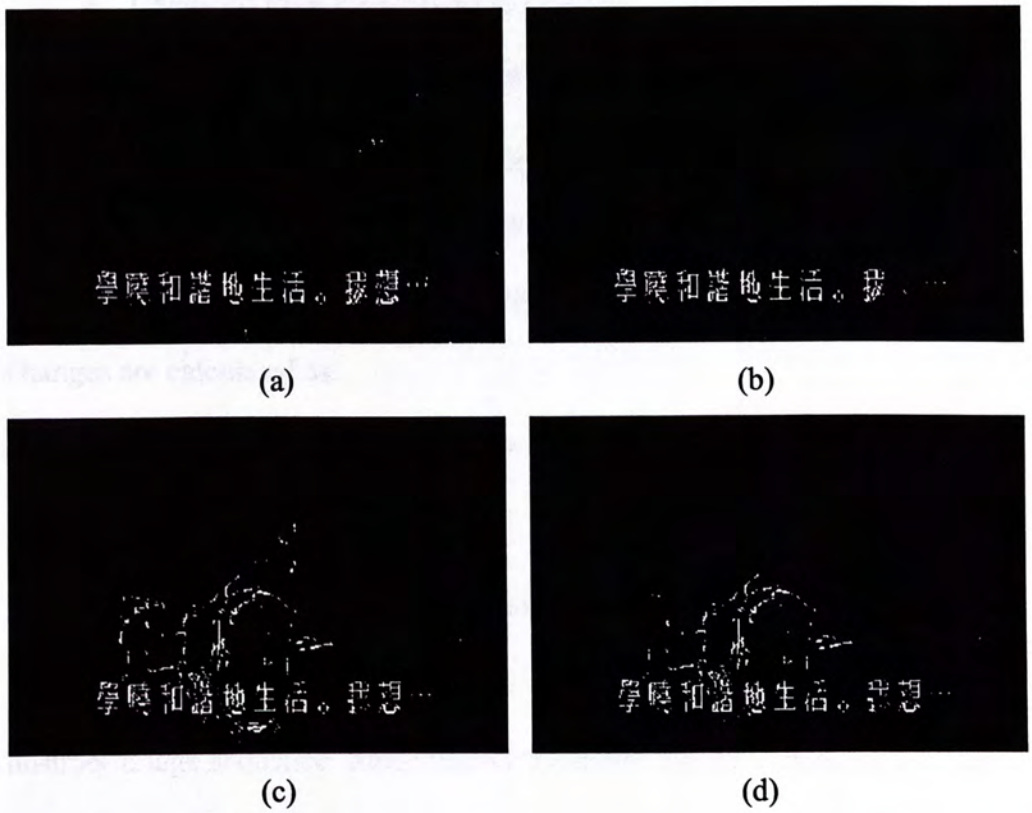


Figure 3.3 Original and refined abstract images.

3.3 Detection of Caption (Dis)Appearance

Examining each pixel across two consecutive abstract images, its behavior can be one of the four types:

- Stays as caption
- Stays as background
- Changing from caption to background
- Changing from background to caption

To detect appearance and disappearance of captions, we are particularly interested in the pixels that is changing between two abstract images. We call change from background to caption a positive change, while change from caption to background a negative change. The numbers of pixels taking these changes are calculated as:

$$| PC |_i = | PositiveChanges |_i = | I_{i+1} \text{ AND NOT } I_i | \quad (3.1)$$

And

$$| NC |_i = | NegativeChanges |_i = | I_i \text{ AND NOT } I_{i+1} | \quad (3.2)$$

In Eq. 3.1 and 3.2, I_i and I_{i+1} denote two consecutive binary images in the abstract image sequence. Since these images are binary, I_i and I_{i+1} are logical matrices in which a true value corresponds to a caption pixel. Operation $|\cdot|$ denotes number of true values in the matrix. Computing $|PC|$ and $|NC|$ over the entire abstract sequence, we get two curves that describe statistically the state

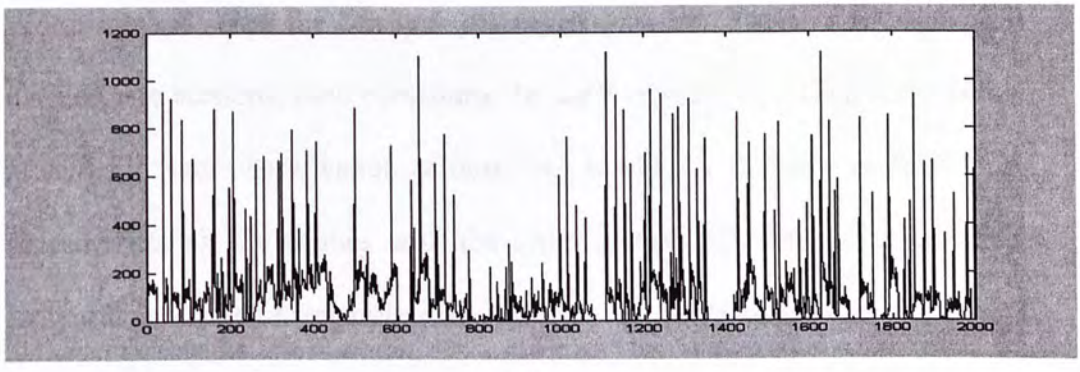
of pixels taking changes. Figure 3.4 shows these curves. They are computed from more than 1800 abstract images over a 5-minute movie segment.

The appearance of one caption implies a relatively large number of pixels taking positive changes at the same frame, which creates a peak in the $|PC|$ curve. Likewise, disappearance of one caption corresponds to a peak in the $|NC|$ curve. We can clearly see these peaks in curves showing in Fig. 3.4. By detecting these peak values in the $|PC|$ and $|NC|$ curves, we can locate the (dis)appearance of captions. We develop the following scheme to detect the caption changes.

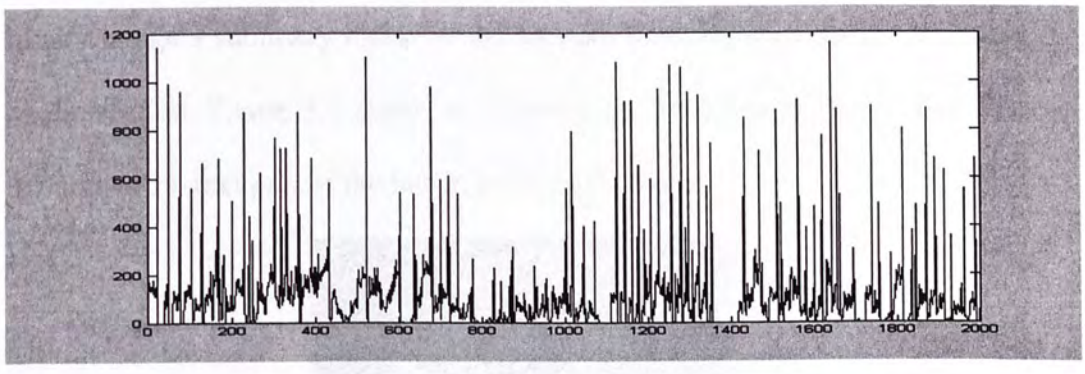
1) A global threshold α is set. For any $|PC|_i \geq \alpha$, F_{5i+1}, the first frame corresponding to I_i, is marked as the appearance frame of a caption. For any $|NC|_i \geq \alpha$, F_{5i+30}, the last frame corresponding to I_i, is marked as the disappearance frame of a caption.
2) If there are more than one consecutive caption appearance frames without any disappearance frame between them, only the last appearance frame is marked.
3) If there are more than one consecutive caption disappearance frames without any appearance frame between them, only the first disappearance is marked.

If a group of falsely classified noise pixels take positive or negative changes at the same time, they also cause a peak in $|PC|$ or $|NC|$ curve. Cases 2 and 3 are used to reduce these cases.

Figure 3.5 shows a small part of the $|PC|$ and $|NC|$ curves with the caption (dis)appearing marks found by our method described above, as well as the caption (dis)appearing frames manually labeled for comparison. In Fig. 3.5, gray curve denotes the |PC| curve; and black curve denotes the |NC| curve. In the figure, gray marks correspond to appearance of captions and black ones correspond to disappearances. Marks "+" are caption (dis)appearances detected by our system; marks "*" are caption (dis)appearances manually labeled as ground truth. We can see that our caption (dis)appearance detection method is quit effective and accurate.

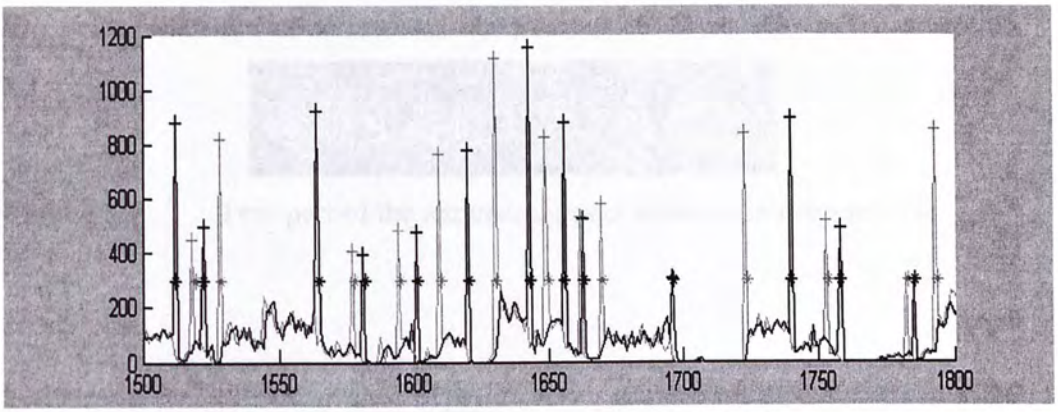


(a)



(b)

Figure 3.4 An example of |PC| and |NC| curves.



(c)

Figure 3.5 Caption (dis)appearance detection results.

This way we detect the caption (dis)appearance using the temporal feature vector method. With the detected (dis)appearance, the whole video segment is divided into sections, each containing the same caption text. Then each section is sent for final classification as described in chapter 2. Since each of these sections has all the frames with the same caption, it contains the maximal temporal information regarding this caption. Its segmentation results should contain the caption with best appearance and the least noises. We call this binary image a summary image of the section, for it brings a textual summary of the section. Figure 3.6 shows an example of the summary image. And Fig. 3.7 shows the text part of the image in its original size.

Figure 3.6 An example of final segmented summary image.

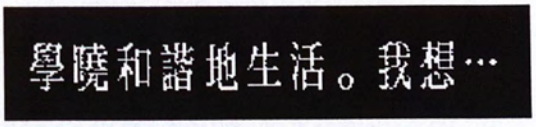


Figure 3.7 Text part of the summary image showing in original size.

Chapter 4

System Overview

4.1 System Implementation

To reduce the storage demand and operational complexity, the implementation of our system is carried out to as follows.

First, a few buffers and a flag are defined. They are: a FIFO (first-in-first-out) queue as *frame buffer*, a *previous abstract image buffer* (PIB) which is for one binary image only, and an *in-caption flag* (ICF). At the beginning, the first 30 frames are read into *frame buffer* and the segmentation result is stored in *previous abstract image buffer*. Then repeat the following procedure to compute final summaries directly.

1. When there is no caption, remove the first 5 frames in the *frame buffer* and append 5 more from the video segment. Perform classification and calculate the current |PC| and |NC| values by operations relative to PIB (see Eq. (1) and (2)), then refresh the PIB with the new classification result. Figure 4.1 shows this process.

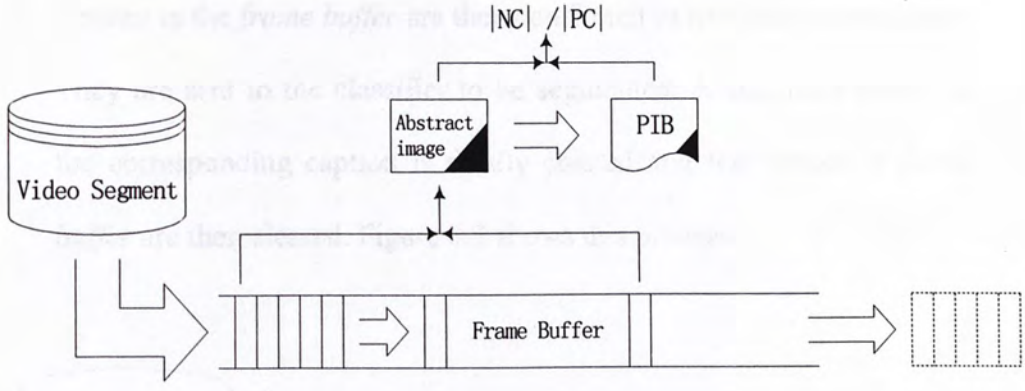


Figure 4.1 Demonstration of step 1 of the system.

2. If appearance of caption is detected by comparing the current |PC| with a preset threshold, set the *in-caption flag*. Then stop removing frames from *frame buffer* and keep on appending 5 frames at the end in each loop. This way, the *frame buffer* length keeps increasing with the same caption in it. However, we only process the last 30 frames, until the disappearance of caption is found. The process of step 2 is shown in Fig. 4.2.

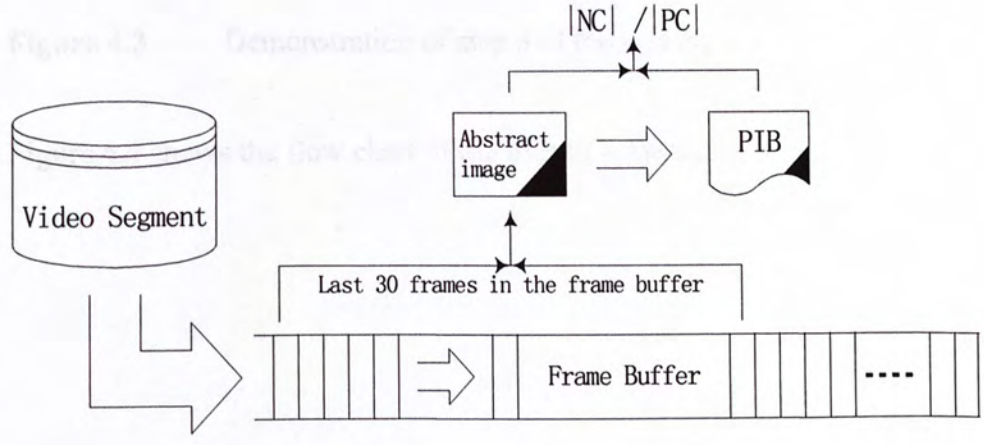


Figure 4.2 Demonstration of step 2 of the system.

3. Frames in the *frame buffer* are then confirmed to have the same caption. They are sent to the classifier to be segmented. A summary image of the corresponding caption is finally created and the frames in *frame buffer* are then cleared. Figure 4.3 shows this process.

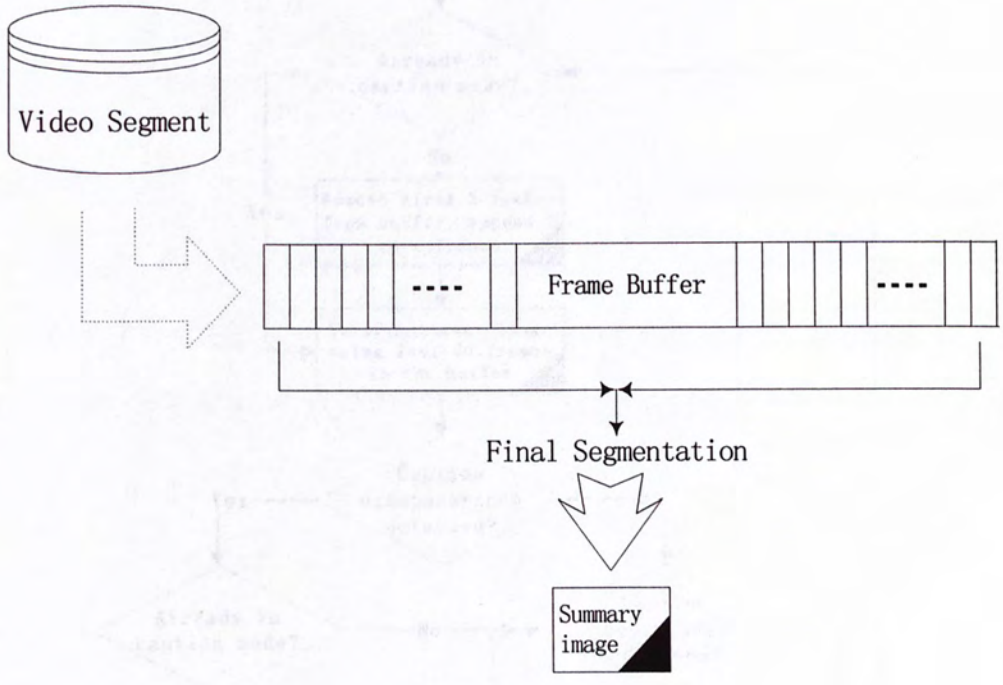


Figure 4.3 Demonstration of step 3 of the system.

Figure 4.4 shows the flow chart of the overall system.

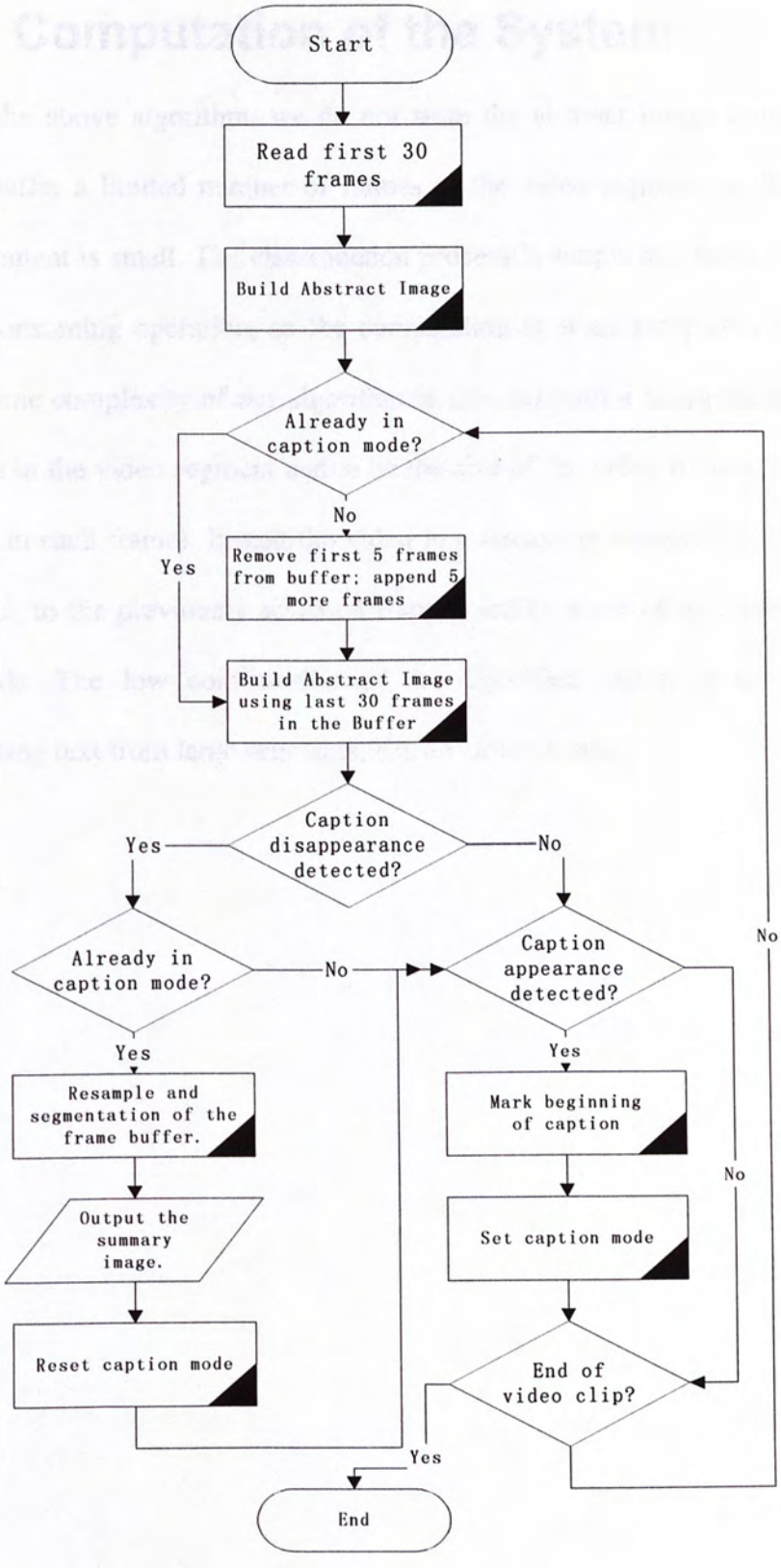


Figure 4.4 Flow chart of the whole system.

4.2 Computation of the System

With the above algorithm, we do not store the abstract image sequence and only buffer a limited number of frames of the video segment, so the storage requirement is small. The classification process is simple and there is no other time-consuming operation, so the computation is at an acceptable level. The total time complexity of our algorithm is $O(n, m)$ with n being the number of frames in the video segment and m be the size of the video frames (number of pixels in each frame). It read the video in a streaming manner, i.e. it does not go back to the previously accessed frames, unlike some other caption tracing methods. The low computation of the algorithm makes it applicable to extracting text from large segments, e.g. an entire movie.

Chapter 5
Experiment Results and Performance Analysis

5.1 The Gaussian Classifier

In our experiment, we select a simple Gaussian classifier to classify the pixels into caption and background. Let the class mean and covariance matrix of the feature vectors be μ_i and W_i, where $i \in \{0,1\}$ denotes caption and background respectively. The distance measure and decision rules are defined by,

$$D_i = (x-\mu_i)^T W_i^{-1}(x-\mu_i) + \ln|W_i| \qquad (5.1)$$

$$x \in C_L \quad \text{when } D_L = \min\{D_i\} \qquad (5.2)$$

The first term on the right of Eq. 5.1 is the Mahalanobis distance. This Gaussian classifier is to classify a feature vector into the class whose center is closer under the distance measure in Eq. 5.1. This classifier is simple and the computation for both training and classification are low. Our results show that, even with this simple classifier, our method achieves high performance.

5.2 Training Samples

Training samples are picked from a 78-frame segment with a stable caption. Thirty frames are equidistantly sampled to build a 30-dimensional vector for each pixel. For the balance of numbers of caption and background samples, we only keep a small area (the area with caption) for training. Figure 5.1 gives an example of these frames, with the area of training samples marked out. The size of the area is 26×218, with 1537 caption pixels and 4331 background pixels. As described in chapter 2, a principal component analysis is applied. We keep the first 5 principal components, which retain 98.8% of the energy in training samples. To select the number of principal components kept, we consider both the compression rate of vector size and retaining of information. Ground truth information is marked by hand. We manually select the caption pixels of the training samples using the principal component images.

Figure 5.1 A frame of the training samples

5.3 Testing Data

To test the performance of the proposed caption (dis)appearance detection and summary image creation algorithms, our testing data includes seven segments from three US movies. Each of the segments lasts 2.5 minutes, with around 4500 frames to be extracted (more than 32000 frames in total). These movies segments have Chinese captions of different font and style. Each segment contains around 40 different captions.

Ground truth information is also marked by hand. We manually went through all these frames; mark each of the first frames with caption as the caption appearance frames, and each of the last frames with caption as the caption disappearance frames. There are totally 260 captions in the testing data set.

5.4 Caption (Dis)appearance Detection

We apply the algorithm described in chapter 3 to detect caption (dis)appearances in the testing data. Table 5.1 shows the caption detection results. The first column indicates the number of segments. The second column indicates the refinement method, which is described in chapter 3.2. The Third column indicates the threshold α used in the caption (dis)appearance detection, which is described in chapter 3.3. Column *caps*. indicates the number of captions in that segment. Column *detects* and the corresponding *rate* indicates the number of correctly detected captions and the detection rate, which is calculated as following:

$$DetectRate = |DetectedCaptions| / GroundTruth \quad (5.3)$$

We also show the number of missed captions and the rate, as

$$MissRate = |MissedCaptions| / GroundTruth \quad (5.4)$$

Table 5.1 Caption (dis)appearance detection results – by segment

Seg	Ref	T	Caps.	Detects	Rate	Misses	Rate
1	1	350	32	28	87.5%	4	12.5%
	2	350		29	90.6%	3	9.4%
	2	330		29	90.6%	3	9.4%
2	1	350	30	27	90%	3	10%
	2	350		29	96.7%	1	3.3%
	2	330		29	96.7%	1	3.3%
3	1	350	29	26	89.7%	3	10.3%
	2	350		27	93.1%	2	6.39%
	2	330		27	93.1%	2	6.39%
4	1	350	39	33	84.6%	6	15.4%
	2	350		33	84.6%	6	15.4%
	2	330		36	92.3%	3	7.7%
5	1	350	36	35	97.2%	1	2.8%
	2	350		35	97.2%	1	2.8%
	2	330		35	97.2%	1	2.8%
6	1	350	32	27	84.4%	5	15.6%
	2	350		27	84.4%	5	15.6%
	2	330		27	84.4%	5	15.6%
7	1	350	39	38	97.4%	1	2.6%
	2	350		37	97.4%	1	2.6%
	2	330		38	97.4%	1	2.6%

Table 5.2 Caption (dis)appearance detection results - overall

Ref	T	Caps.	Detects	Rate	Misses	Rate
1	350	237	214	90.3%	23	9.7%
2	350		218	92.0%	19	8.0%
2	330		221	93.2%	16	6.8%

Other important caption detection performance evaluations are shown in Fig. 5.2. Because of noise, some captions are detected to have more than one appearances and disappearances. Captions appearing at the 6281st frame, disappearing at the 6419th frame, is detected as appearing at 6281st frame, disappearing at 6315th frame, appearing again at 6391st frame and disappearing at 6420th frame. The number of breaks is shown in Fig. 5.2. Note that, the shown number is actually number of broken clusters, e.g. if one caption is broken into 5 in the detection result, 5 rather than 1 is recorded as *breaks*. Number of *False Alarms* (*FAs*) is also shown in table 5.3.

Table 5.3 Caption (dis)appearance detection performance- by segment

Seg	Ref	T	Detects	Breaks	FAs
1	1	350	28	0	1
	2	350	29	0	0
	2	330	29	0	0
2	1	350	27	5	2
	2	350	29	5	1
	2	330	29	5	2
3	1	350	26	2	6
	2	350	27	1	5
	2	330	27	1	6
4	1	350	33	5	0
	2	350	33	4	0
	2	330	36	3	0
5	1	350	35	0	0
	2	350	35	0	0
	2	330	35	0	0
6	1	350	27	0	1
	2	350	27	0	1
	2	330	27	0	1
7	1	350	38	13	11
	2	350	37	14	11
	2	330	38	14	13

In table 5.4, we show the overall performance of caption detection results calculated from all 7 segments.

Table 5.4 Caption (dis)appearance detection performance - overall

Ref	T	Detects	Breaks	FAs
1	350	214	25	21
2	350	218	24	18
2	330	221	23	22

We also have to mention that most false alarms can be remove in the text line extraction process, which will be described in chapter 5.6. This is because we cannot detect any text lines in those false alarm summary images.

Another benchmark is the accuracy of the detected position of (dis)appearance boundary frames. The *mean absolute error* (MAE) and the *mean square root error* (MSRE) are calculated by:

$$MAE = \frac{1}{N_{CA} + N_{CD}} \left(\sum_{C_i \in CA} \left| T_A - \tilde{T}_A \right| + \sum_{C_i \in CD} \left| T_D - \tilde{T}_D \right| \right) \qquad (5.6)$$

and:

$$MAE = \sqrt{\frac{1}{N_{CA} + N_{CD}} \left(\sum_{C_i \in CA} (T_A - \tilde{T}_A)^2 + \sum_{C_i \in CD} (T_D - \tilde{T}_D)^2 \right)} \qquad (5.7)$$

N_{CA} denotes number of caption appearances and N_{CD} denotes number of caption disappearances. Normally N_{CA} and N_{CD} are the same, unless a caption exists at the beginning of the segment or a caption exists at the end of the segment. In these cases, we do not count them as "detected (dis)appearances".

CA indicates caption appearances and *CD* indicate caption disappearances. T_A denotes the detected time of caption appearance, which is actually the detected caption appearing frame number. T_D denotes the detected time of caption appearance. $\tilde{T}$ denotes the actual caption (dis)appearance time, i.e. the ground truth. MAE is the mean of differences between the detected caption (dis)appearance time and the real caption (dis)appearance time. And MSRE is the square root of the mean of the square of differences between the detected caption (dis)appearance time and the real caption (dis)appearance time. These two measures indicates the accuracy of the detected location (frame ID) of the caption (dis)appearance. We show the *MAE* and *MSRE* in table 5.5. As stated in chapter 4, we selected a step length of five frames when calculating the abstract images. So the precision of our method is five frames, i.e. the detected position of boundary frame is within eight frames from the actual boundary frame, the detection is regarded accurate. Number of accurate detections (ADN) and the rate of accurate detections (ADR) are also shown in table 5.5. *MAE*, *MSRE*, *AND*, and *ADR* values shown in table 5.5 are computed using the method with highest detection rate, i.e. abstract image refinement method two and detection threshold 330.

Table 5.5 Accuracy of the detected location of the caption (dis)appearance

	MAE	MSRE	ADN	ADR
Appearance	6.29	7.35	213	96.4%
Disappearance	9.82	11.61	196	88.7%

5.5 Caption Segmentation

After the detection of caption (dis)appearance, each video segment is divided into short clips. Non-caption clips are not of any interest in this scheme. A stable caption remains in each caption clip, which will be further processed. Since each clip has different number of frames, the temporal feature vectors extracted from pixels of different clip has different sizes. Referring to Eq. 2.6, we cannot project these vectors into the training PCA space directly. A simple way to solve this problem is to re-sample each of the clips into a fixed number of frames. In our experiment, each clip is equidistantly re-sampled into 30 frames, and then temporal feature vectors of pixels are extracted, projected into the PCA space built by training samples, and finally classified into caption and background pixels. The classification result is represented as a binary image, with 1 denoting a caption pixel and 0 denoting a background pixel. Detailed classification processing is described in chapter 3. Figure 5.2 and 5.3 shows some examples of the final results. Results shown in Fig. 5.2 are those without any noise, while images shown in figure 5.3 still have wrongly classified areas. Our next step is to crop out the texts then send them to OCR. These wrongly classified areas will be removed automatically in that step.

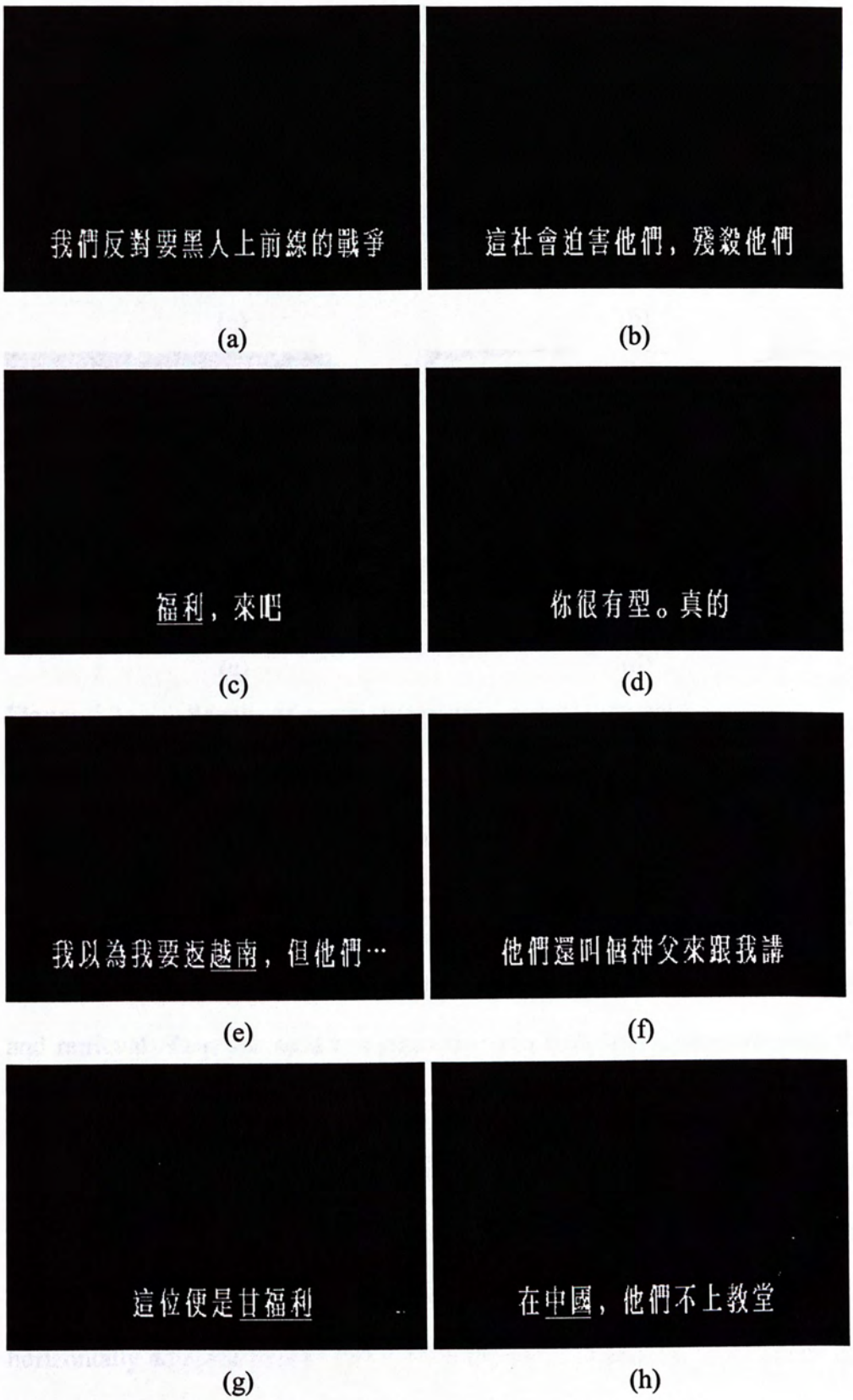


Figure 5.2 Result summary images - perfect results.

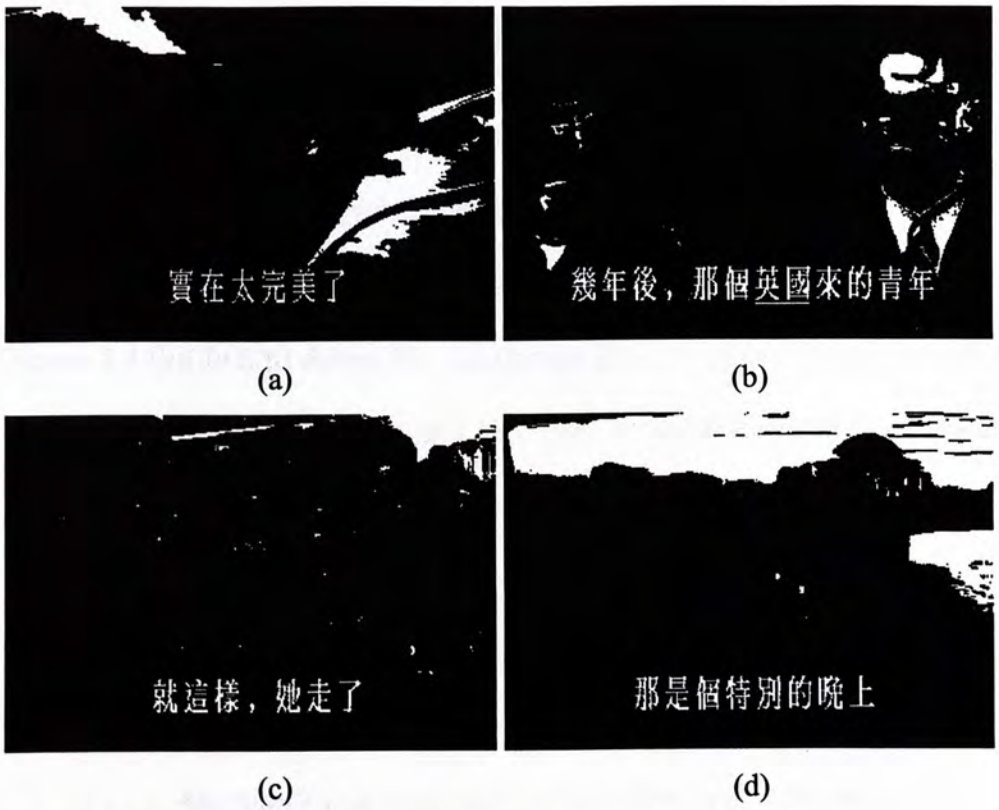


Figure 5.3 Results of summary images – results with noises.

5.6 Text Line Extraction

The ultimate goal of caption detection and extraction is to send the images of caption characters to video OCR, then use the recognition results for indexing and retrieval. Thus we need to extract the area with text characters from the segmented images shown in Fig. 5.2 and 5.3. With the observation that text characters have rich edge information, we detect the horizontal crossing point and project them to Y-axis. A horizontal crossing point is a change from background to caption or a change from caption to background between two horizontally adjacent pixels. The calculation of horizontal crossing points can be described as follows:

$$CP = \sum_{p \in r} |r - (r \rightarrow 1)| \tag{5.8}$$

In Eq. 5.8, r denotes a row of pixels from a summary image. Since the summary image is binary, r is a vector contains only values of 0 or 1. The $\rightarrow$ operator indicates the rotate right operation.

Figure 5.4 (b)(d)(f)(h) shows the y-projection of the summary images shown in Fig. 5.3. It is very clear that text line can be easily detected. We use the following algorithm to extract text lines:

1. A threshold T for number of horizontal crossing points is set. Any row of pixels with more than or equal to T crossing points is marked as a candidate text row.
2. Any single non-candidate text row between two candidate text rows is also marked as candidate text row.
3. Another threshold Ls for number of consecutive candidate text rows is set. Any set of consecutive candidate text rows with more than Ls rows is marked as a text line.

The thresholds T and Ls are pre-set parameters. In our experiments, T is set to 14, and Ls is set to16, which is almost the smallest size for Chinese characters. These values are good for all the test video segments. Further experiments also show that T can take values between 12 and 18, with little affection to the performance. And Ls can take values between 16 and 24, with no affection to the performance.

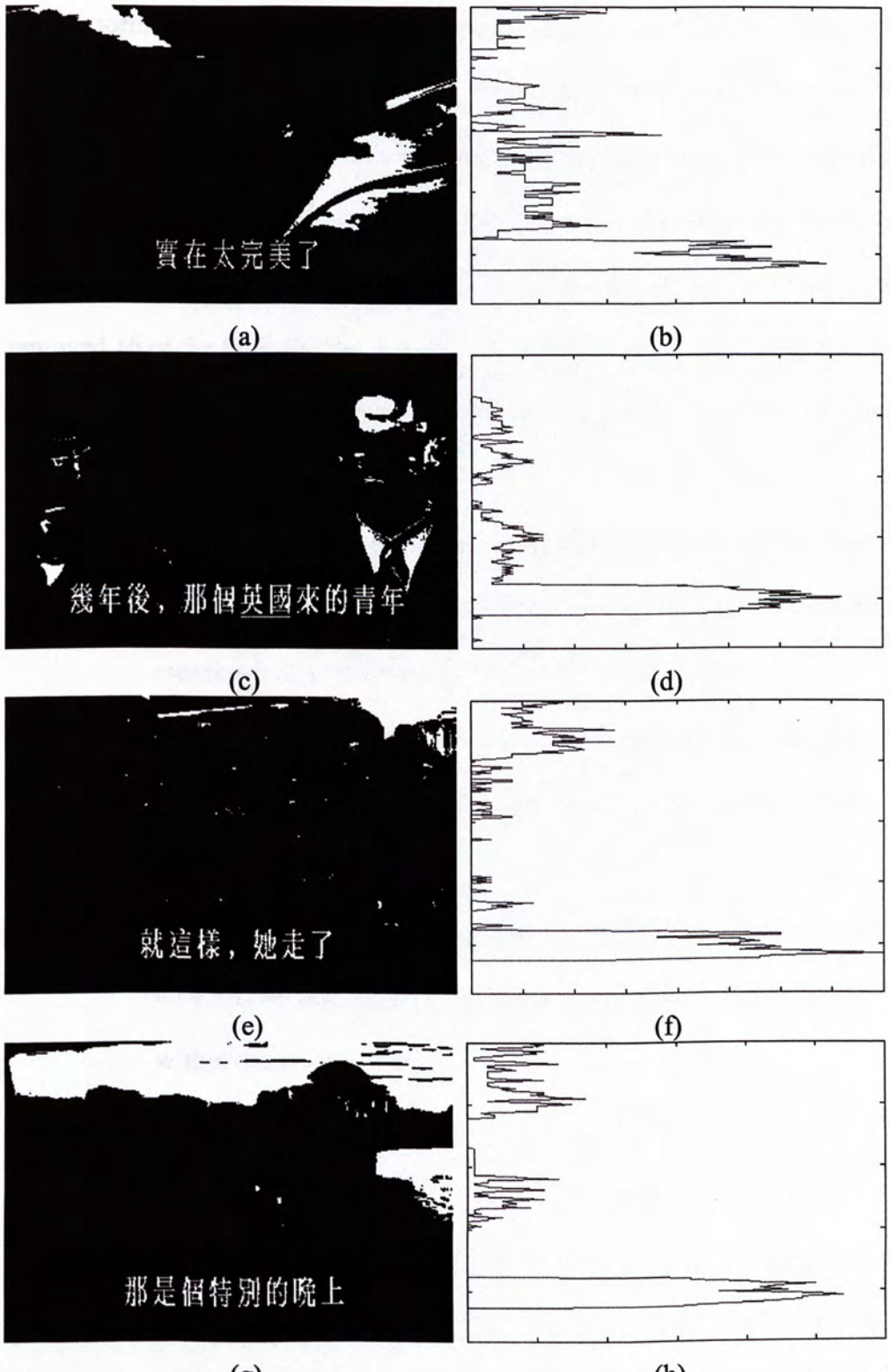


Figure 5.4 Y-projection of the horizontal crossing points.

We perform the text line detection experiments on the segmented summary images, as shown in figure 5.2 and 5.3. We only extracted text lines from the summary images computed using the method with highest detection rate listed in table 5.2, i.e. abstract image refinement method 2 and detection threshold 330. Out of the 221 detected captions, we extracted all the text lines, and removed 16 of the total 23 false alarms. We also extracted 7 fake text lines.

To remove the background at the beginning and end of the text lines, we apply the same algorithm vertically.

1. A threshold T for number of vertical crossing points is set. Any column in the detected text line with more than or equal to T crossing points is marked as a candidate text column.
2. Any one or two non-candidate text columns between two candidate text columns are also marked as candidate text column.
3. Another threshold Cs for number of consecutive candidate text columns is set. Any set of adjacent candidate text columns with more than Cs columns is marked as a text block.

In figure 5.5, we show the extracted text area of the summary images shown in figure 5.3. To illustrate how the algorithm described above works, we draw the removed areas in gray color, and the retained text areas in their original colors. We can clearly see that most of the noise areas have been removed together with the pure background, left only the area with text, which is of our interest.

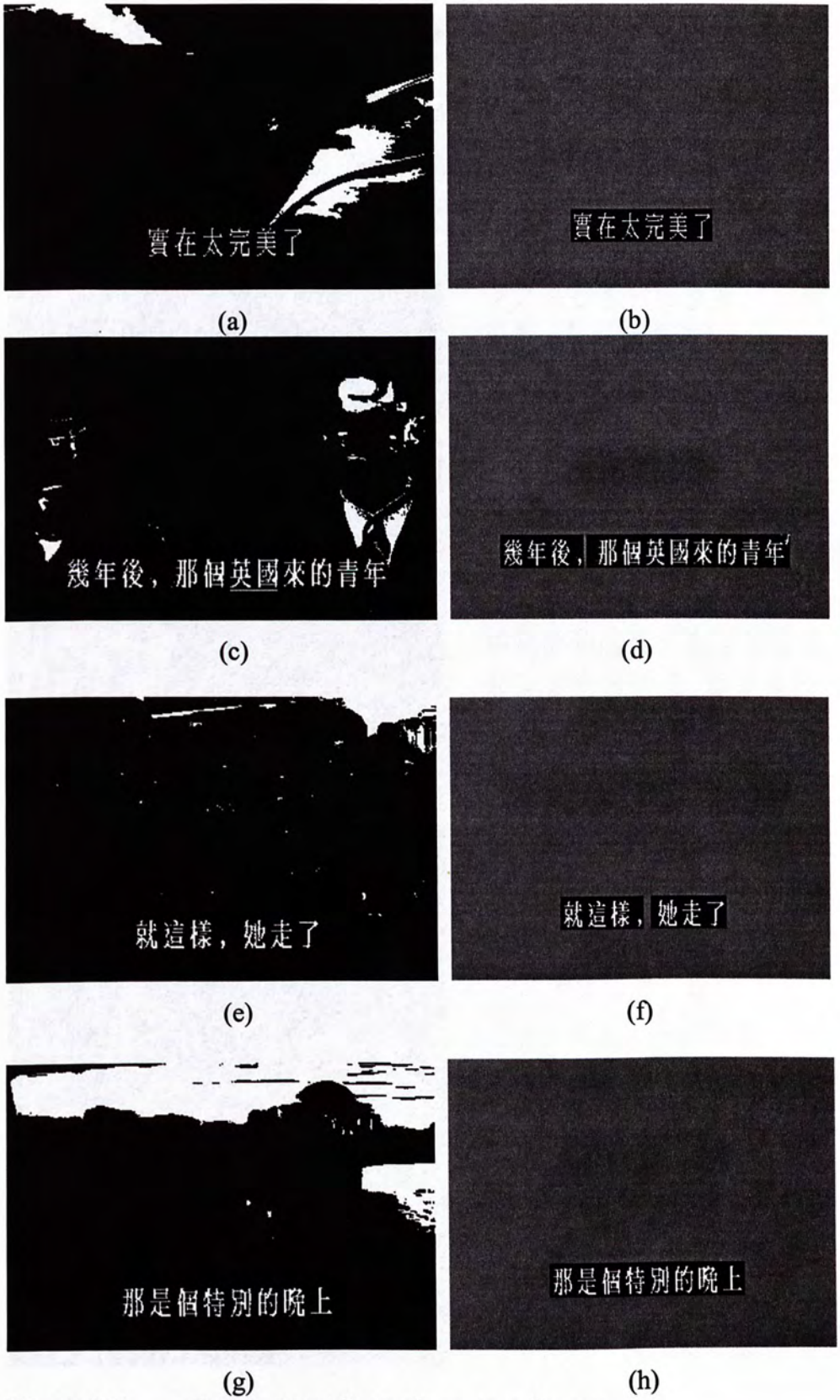
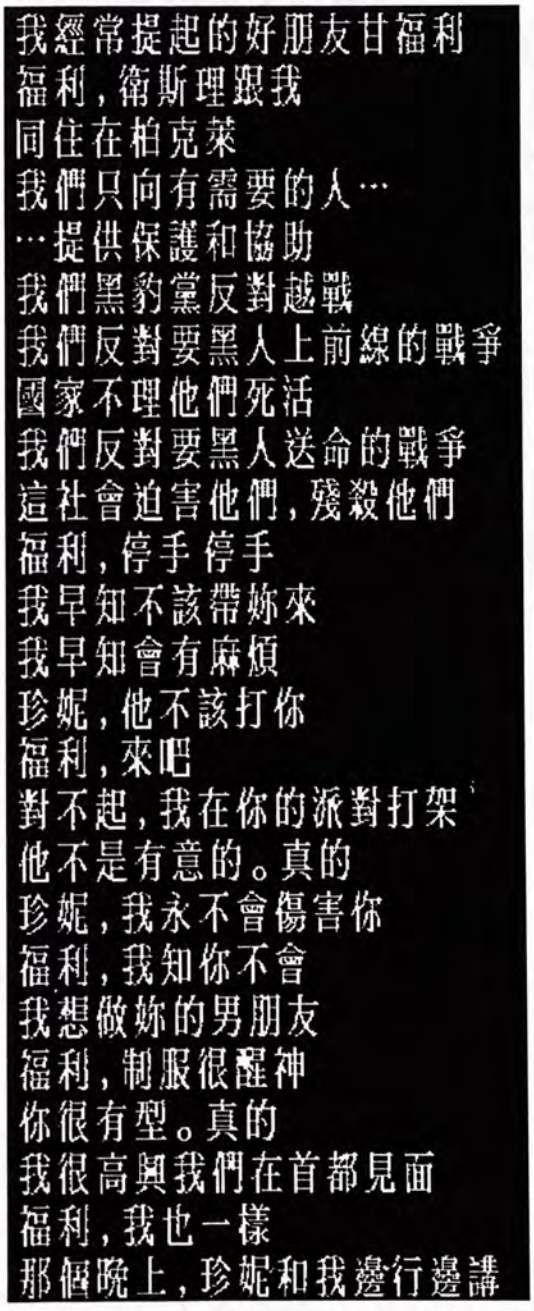


Figure 5.5 Detected text lines of summary images.

5.7 Caption Recognition

After text line extraction, segmented characters are extracted, as shown in Fig. 5.6. To send it to recognition, we need to reverse the color first. The reversed image is also shown in Fig 5.6.



我經常提起的好朋友甘福利
福利，衛斯理跟我
同住在柏克萊
我們只向有需要的人…
…提供保護和協助
我們黑豹黨反對越戰
我們反對要黑人上前線的戰爭
國家不理他們死活
我們反對要黑人送命的戰爭
這社會迫害他們，殘殺他們
福利，停手停手
我早知不該帶妳來
我早知會有麻煩
珍妮，他不該打你
福利，來吧
對不起，我在你的派對打架
他不是有意的。真的
珍妮，我永不會傷害你
福利，我知你不會
我想做妳的男朋友
福利，制服很醒神
你很有型。真的
我很高興我們在首都見面
福利，我也一樣
那個晚上，珍妮和我邊行邊講

Figure 5.6 Extracted text lines

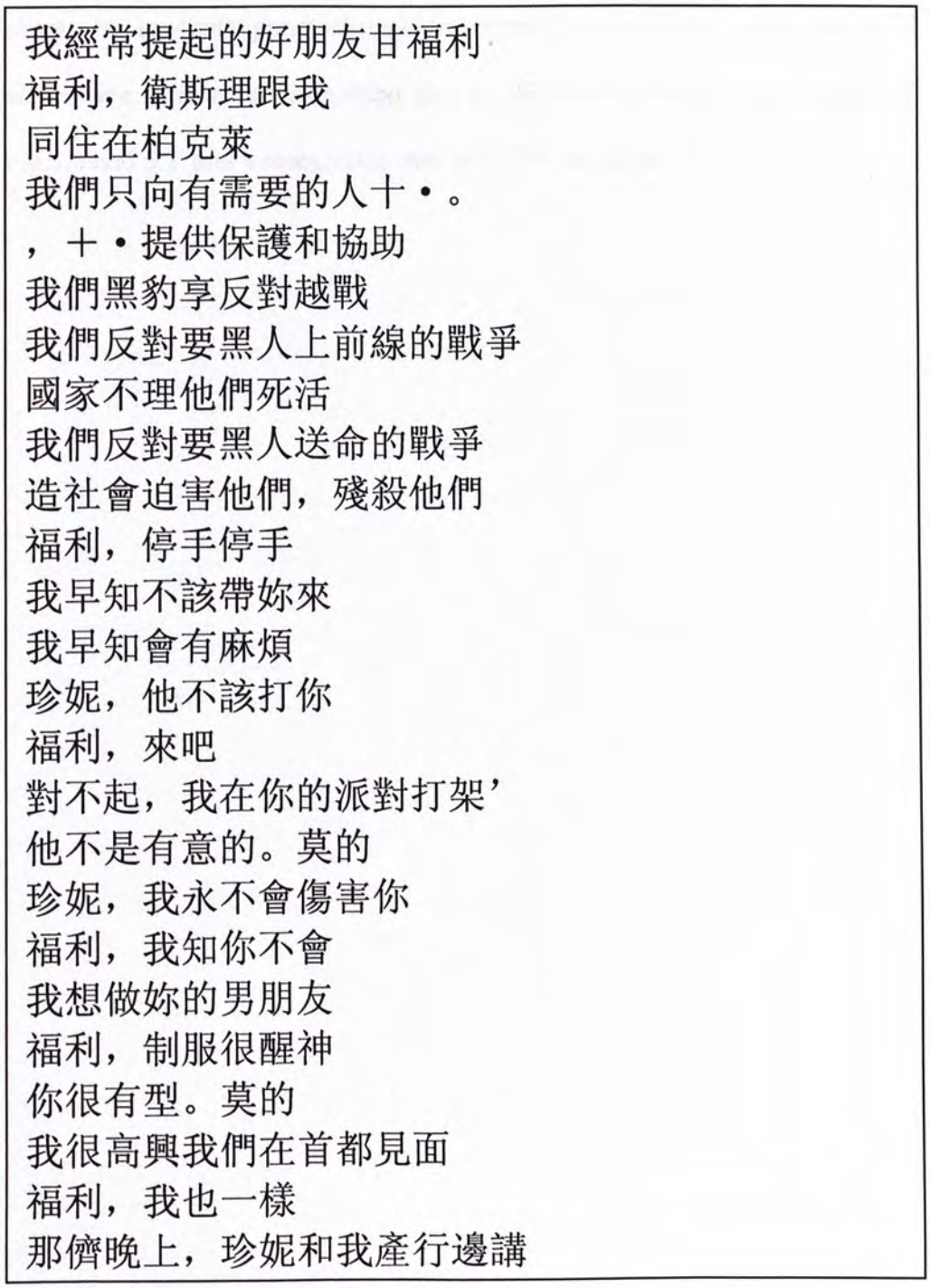


Figure 5.7 Recognition results.

The extracted text lines are enlarged at a factor of 4 and recognized using TH-OCR Version 2000. The output is shown in Fig. 5.7. Overall 205 out of 211

characters (exclude punctuations) are correctly recognized, with one false alarm. We achieve a recognition rate of 97.2% on the text lines shown in Fig.5.6 and 5.7. and a recognition rate of 94.5% on all the characters.

Chapter 6 Summary

In this thesis, we present a video caption detection and extraction method that takes full advantage of temporal information. We define temporal feature vector to describe the temporal features of pixels across a video clip. We trace over the video segment to extract an abstract image sequence with coarsely segmented caption text, and refine the abstract images to remove the falsely classified regions. Then we statistically analyze the pixels changing between adjacent abstract images and detect the (dis)appearance of captions thus create video clips each containing all the frames with the same caption. Refined caption text is then extracted and a summary of captions is finally created. The final summary images give a summary of captions contained in the video segment. These frames are of high quality and can be sent to OCR recognition. With the implementation scheme described in chapter 4, the computational complexity of our system is low. In experiments, we applied our method on seven video segments with 260 captions in total. Our method achieved an average recognition rate of 94.5% on the extracted caption text. This algorithm does not make any assumptions on the shape of the caption, i.e. we do not need the captions to be horizontal, constant size, certain font or fixed location. In the future, we plan to analyze and implement more spatial information based methods and combine them with the temporal information based method to

achieve more effective and robust methods of video text detection and recognition. We also plan to implement some video indexing and retrieval schemes using the extracted text.

Bibliography

[1] S. W. Smoliar and Hongjiang Zhang, "Content-based video indexing and retrieval," *IEEE Multimedia*, pp. 62--72, Summer 1994.

[2] HongJiang Zhang, Chien Yong Low, Stephen W. Smoliar, and Jian Hua Wu, "Video Parsing, Retrieval and Browsing: An Integrated and Content-Based Solution," *in Proceedings of ACM Multimedia*, San Francisco, California, United States, 1995.

[3] Marc Davis, "Media Streams: Representing Video for etrieval and Repurposing," Ph.D. Thesis, Massachusetts Institute of Technology,1995

[4] Rune Hjelsvold, Stein Langørgen, Roger Midtstraum, and Olav Sandstå, "Integrated video archive tools," *in Proceedings of ACM Multimedia*, San Francisco, CA, USA, 1995.

[5] Edoardo Ardizzone, and Marco La Cascia, "Video indexing using optical flow field, " *in Proceedings of IEEE International Conference on Image Processing*, Sept. 1996.

[6] Edoardo Ardizzone, Marco La Cascia, and Davide Molinelli, "Motion and color-based video indexing and retrieval, " *in Proceedings of International Conference on Pattern Recognition*, Aug. 1996

[7] Hongjiang Zhang, John Y. A. Wang, and Yucel. Altunbasak. "Content-based video retrieval and compression: A unified solution." *In*

Proceedings of the IEEE International Conference on Image Processing, 1997

[8] Hongjiang Zhang, Jian Hua Wu, Di Zhong, and Stephen W. Smoliar, "An integrated system for content-based video retrieval and browsing," Pattern Recognition, vol. 30, no. 4, pp. 643--658, April 1997.

[9] Atsuo Yoshitaka, and Tadao Ichikawa, "A survey on content-based retrieval for multimedia databases," *IEEE Transactions on Knowledge and Data Engineering,* vol. 11, no. 1, pp. 81-93, Jan.-Feb. 1999.

[10] Arturo de la Escalera, Miguel Angel Salichs, "Road Traffic Sign Detection and Classification", *IEEE Transactions on Industrial Electronics*, Vol. 44, No. 6., 1997.

[11] Paolo Comelli, Palo Ferragina, Mario Notturno Granieri, and Flavio Stabile, "Optical Recognition of Motor Vehicle License Plates", *IEEE Transactions on Vehicular Technology*, Vol. 44, No. 4, November 1995.

[12] Dong-Su Kim, Sung-Il Chien, "Automatic Car License Plate Extraction Using Modified Generalized Symmetry Transform and Image Warping", *in Proceedings of IEEE International Symposium on Industrial Electronics* 2001, Pusan, KOREA.

[13] Jie Yang, Xilin Chen, Jing Zhang, Ying Zhang, and Alex Waibel "Automatic Detection And Translation of Text from Natural Scenes," *in Proceedings of the IEEE International Conference on Acoustic Speech and Signal Processing*, Orlando, May 2002.

[14] Huiping Li, Davide Doermann, and Omid Kia, "Automatic Text

detection and tracking in digital video," *IEEE Transactions on Image Processing,* vol. 9, no.1, pp. 147-156, 2000.

[15] Anil K. Jain, and S. Bhattacharjee, "Text Segmentation Using Gabor Filters for Automatic Document Processing," *Machine Vision and Applications,* vol. 5, pp. 169-184, 1992.

[16] Xiaoou Tang, Xinbo Gao, Jianzhuang Liu and Hongjiang Zhang, "A spatial-temporal approach for video caption detection and recognition," *IEEE Transactions on Neural Networks, Special Issue on Intelligent Multimedia Processing*, vol. 13, no. 4, July, 2002.

[17] Xinbo Gao and Xiaoou Tang, "Unsupervised video shot segmentation and model-free anchorperson detection for news video story parsing," *IEEE Transactions on Circuits, Systems and Video Technology*, vol. 12, no. 9, Sept., 2002.

[18] Xiaoou Tang, Bo Luo, Xinbo Gao, E. Pissaloux, and Hongjiang Zhang, "Video text extraction using temporal feature vectors," *in Proceedings of the IEEE International Conference on Multimedia and Expo*, Lausanne, Switzerland, Aug. 2002.

[19] Bo Luo, Xiaoou Tang, Jianzhuang Liu and Hongjiang Zhang, "Video Caption Detection and Extraction Using Temporal Information," *in Proceedings of the IEEE International Conference on Image Processing,* Barcelona, Spain, September 2003.

[20] Lalitha Agnihotri and Nevenka Dimitrova, "Text detection for video analysis," *Workshop on Content-based access to image and video libraries*

in conjunction with IEEE International conference on Computer Vision and Pattern Recognition, Colorado, June, 1999.

[21] Anil K. Jain and Bin Yu, "Automatic text location in images and video frames," *Pattern recognition*, Vol.31, No.12, pp.2055-2076, 1998

[22] E. K. Wong and M. Chen, "A robust algorithm for text extraction in color video," *Proc. of IEEE International Conference on Multimedia and Expo,* Vol. 2, pp. 797-800, 2000

[23] Rainer Lienhart and F. Stuber, "Automatic text recognition in digital videos," *Proceedings of SPIE Image and Video Processing IV 2666,* pp.180-188, 1996.

[24] Toshio Sato, Takeo Kanade, Ellen K. Kughes, Michael A. Smith, and Shin'ichi Satoh, "Video OCR: indexing digital news libraries by recognition of superimposed captions," *Multimedia Systems,* 7(5), pp.385-395, 1999.

[25] Jae-Chang Shim, Chitra Dorai and Ruud Bolle, "Automatic text extraction from video for content-based annotation and retrieval," *in Proceedings of the IEEE International Conference on Pattern Recognition*, pp.618-620, Brisbane, Australia, 1998.

[26] Axel Wernicke and Rainer Lienhart, "On the segmentation of text in videos," *in Proceedings of IEEE International Conference on Multimedia and Expo,* Vol. 3, pp. 1511-1514, 2000.

[27] Rainer Lienhart and Axel Wernicke, "Localizing and segmenting text in images and videos," *IEEE Transactions on Circuits and Systems for Video*

Technology, Vol. 12 no.4, April 2002.

[28] Xian-Sheng Hua, Xiang-Rong Chen, Liu Wenyin, and Hong-Jiang Zhang, "Automatic Location of Text in Video Frames," *in Proceedings of 3rd Intl Workshop on Multimedia Information Retrieval,* Ottawa, Canada, October 2001.

[29] Jae-Chang Shim, and Chitra Dorai, "A Fast and Generalized Region Labeling Algorithm," Technical Report, IBM Thomas J. Watson Research Center, Yorktown Heights, New York, 1998.

[30] Xian-Sheng Hua, Pei Yin, and Hong-Jiang Zhang, "Efficient Video Text Recognition Using Multiple Frame Integration," *in Proceedings of the IEEE International Conference on Image Processing,* Rochester, New York, 2002.

[31] Rafael C. Gonzalez and Richard E. Woods. *Digital image processing.* Addison Wessley Publishing Company, 1992

[32] Huiping Li, and Davide Doermann, "Video indexing and retrieval based on recognized text," *in Proceedings of the IEEE workshop on Multimedia Signal Processing*, St. Thomas, US Virgin Islands, December 2002.

[33] Rainer Lienhart, and Wolfgang Effelsberg, "Automatic Text Segmentation and Text Recognition for Video Indexing", *Multimedia Systems*, Volume 8, Issue 1, pp 69-81, 2000.

[34] K. Pearson, "On Lines and Planes of Closest Fit to Systems of Points in Space," *The London, Edinburgh and Dublin Philosophical Magazine and*

Journal of Science, 6(2), pp 559–572, 1901.

[35] H. Hotelling, "Analysis Of A Complex Of Statistical Variables Into Principal Components," *Journal of Educational Psychology*, 24:417–441, 498–520, 1933.

[36] Wenge Mao, Fu-lai Chung, Kenneth K.M Lam, and Wan-chi Siu, "Hybrid Chinese/English Text Detection in Images and Video Frames," *in Proceedings of the 16th International Conference on Pattern Recognition*, Quebec, Canada, 2002.

[37] Jiqiang Song, Min Cai, and Michael R. Lyu, "A Robust Statistic Method for Classifying Color Polarity of Video Text," *in Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing,* Hong Kong, 2003.

[38] Si-Hun Sung, and Woo-Sung Chun, "Knowledge-Based Numeric Open Caption Recognition for Live Sportscast," *in Proceedings of the 16th International Conference on Pattern Recognition*, Quebec, Canada, 2002.

[39] Milan Petkovic, Vojkan Mihajlovic, Willem Jonker, and S. Djordjevic-Kajan, "Multi-Modal Extraction of Highlights From TV Formula 1 Programs," *in Proceedings of the IEEE International Conference on Multimedia and Expo*, Lausanne, Switzerland, Aug. 2002.

[40] Chung Wing Ng and Michael R. Lyu, "ADVISE: Advanced Digital Video Information Segmentation Engine," *in Proceeding of the Seventh International Conference on World Wide Web*, Honolulu, Hawaii, USA, May, 2002.