Multimed Tools Appl (2009) 42:233–250
DOI 10.1007/s11042-008-0236-x

# Dynamic video summarization using two-level redundancy detection

**Yue Gao · Wei-Bo Wang · Jun-Hai Yong · He-Jin Gu**

Published online: 1 November 2008

**Abstract** The mushroom growth of video information, consequently, necessitates the progress of content-based video analysis techniques. Video summarization, aiming to provide a short video summary of the original video document, has drawn much attention these years. In this paper, we propose an algorithm for video summarization with a two-level redundancy detection procedure. By video segmentation and cast indexing, the algorithm first constructs story boards to let users know main scenes and cast (when this is a video with cast) in the video. Then it removes redundant video content using hierarchical agglomerative clustering in the key frame level. The impact factors of scenes and key frames are defined, and parts of key frames are selected to generate the initial video summary. Finally, a repetitive frame segment detection procedure is designed to remove redundant information in the initial video summary. Results of experimental applications on TV series, movies and cartoons are given to illustrate the proposed algorithm.

**Keywords** Video summarization · Redundancy detection · Cast indexing · Important factor

Authors of Tsinghua University were supported by Chinese 973 Program(2004CB719400) and the National Science Foundation of China (60533070,90715043).

Y. Gao (✉)
Department of Automation, Tsinghua University, Beijing, China
e-mail: kevin.gaoy@gmail.com

W.-B. Wang · J.-H. Yong
School of Software, Tsinghua University, Beijing, China

H.-J. Gu
Jiangxi Academy of Sciences, Nanchang, China

## 1 Introduction

With the wide spread of multimedia applications over the past decade, multimedia techniques have found applications in nearly every aspect of our life. Efficient methods for video analysis are required to confront this situation. Since it is time-consuming to download and browse most parts of a long video before we know the content, much research attention [3, 5, 6, 17, 30] has been focused on effective video summarization in order to help users to grasp the video content quickly.

Video summarization is to generate a condensed sequence of still or moving images to represent the content in the original video. According to the summary mode, previous works can be categorized into "story board" and "video skimming". Story board [29] is a collection of still images, such as a key frame list of shots or scenes. Uchihashi et al. [29] proposed a video manga system aiming to generate story board and [2] proposed a comic-like video summarization algorithm. This kind of summaries can be constructed fast but the descriptive ability is limited since they lost dynamic audiovisual content of the original video. Therefore, more efforts are devoted to video skimming.

Video skimming is made up of moving images, and it shows the viewers the important video segments for efficient browsing. Based on information theory, Gong and Liu [8] proposed a video skimming approach with minimal visual content redundancies. They first clustered key frames and then concatenated short video segments of key frames to construct a video skimming. Li and Schuster [14] proposed an optimal video summary solution to the temporal distortion minimization formulation. Otsuka and Nakane [18] proposed a highlights summarization approach, where they used audio feature to detect sports highlights. Generally, video summarization is based on video content cluster methods, and effective redundancy detection method is needed.

In [15], Liu and Katpelly provided a content-adaptive video summarization with key frame selection. As a video sequence is composed with continuous frames containing meaningful information, it is difficult to represent the video structure and help users to understand the video content exactly using only key frame comparison. Zhu and Wu [33] grouped video shots into clusters to form shot cluster sequence, and created video summary by considering users specification of summary information. In [1], Benini et al. provided a video summarization method with content analysis in the shot level, where they used a vector quantization codebook using low-level features to calculate shot similarities.

In [12], Lee et al. provided a scenario based dynamic video abstractions using graph matching algorithm. This method contains two procedures: multi-level scenario generations and dynamic video abstractions. Multi-level scenarios are generated by a graph-based video segmentation and a hierarchy of the segments. Dynamic video abstractions are accomplished by accessing the generated hierarchy level by level. Scharcanski and Gaviao [24] provided a hierarchical technique to identify clinically relevant segments in diagnostic hysteroscopy videos, and their associated key-frames, and then create a rich video summary. This approach is adaptive to video contents, and represents the clinically relevant video segments hierarchically to facilitate fast video browsing.

A video summarization algorithm based on video clips comparison was provided in [20]. In their method, video clip similarity was measured using bipartite graph matching algorithms, maximal matching and optimal matching. A graph partition method Ncuts was employed to group video clips into clusters. Other video summarization algorithms including [16, 19, 21, 25, 27, 33] have been proposed recent years.

Video summarization aims to provide a brief introduction of the original video to users. Based on this goal, how to remove similar video content in the original video is important for video summarization task. To remove redundant information and keep essential video clips, we propose a video summarization approach based on a two-level repetitive information detection procedure in this paper.

First the original video sequence is divided into shots and scenes, and key frames are extracted from these shots and scenes. Our approach first constructs a story board to let users know about the main scenes and the main actors in the video (the main actors are only shown when this is a video with people). With cast detection, users could know the main cast through the cast story board. The story boards of main cast and main scenes could provide a brief view of the original video. These story boards are showing the video content in some static frames like the static video abstract.

Redundant video content in each scene is detected using the hierarchical agglomerative clustering method in the key frame level, and then removed. We also set the impact factors of scenes and key frames, and the a original dynamic summary is constructed using some key frames. To remove redundant information in the original video summary, a similar frame clip detection step is employed here.

The reasons for using the two-level repetitive information detection procedure are as follows. Generally, two similar shots are with similar key frames. When there are large camera or object movements, key frames generated from similar shots may have large differences. Although the key frame clustering step could remove most of similar video content, when two shots are with long time length and large camera/object movements, the key frames extracted may be not similar enough to be grouped into one cluster. Therefore, the repetitive video clips detection in the key frame level could not remove this type redundant information. The frame segment level repetitive information detection could further remove this redundant video content.

As video analysis with the whole video frame segment is with large computation load, and the key frame comparison is with little computation load, the first redundancy detection in the key frame level is necessary to improve the video processing speed.

Using both the key frame level and the frame segment level redundancy detection procedures could detect most of similar video content. With the content selection step, our summary could emphasis the important content of the original video sequence. Combining this story boards and the following dynamic video summary, this method could achieve the advantages of both static video abstract and dynamic video skimming.

The major contributions of our approach are as follows.

- *Video summarization in two levels.* We propose a video summarization algorithm with a two-level redundancy detection scheme. This method could remove similar video clips both in the key frame level and the frame segment level.
- *A refined video frame segment repetitive information detection procedure.* When two video shots are with long time length and large camera movements, these

key frames may change much. Therefore, the repetitive video clips detection in the key frame level is not enough, and we provide a refined video frame segment repetitive information detection procedure using the *Smith-Waterman* algorithm.

The remaining parts of the paper are organized as follows. The algorithm of dynamic video summarization is presented in Section 2. Experimental results and analysis are shown in Section 3. Conclusions and comments on future directions are given in Section 4.

## 2 Dynamic video summarization

The proposed video summarization method is presented in this section. The architecture of our video summarization algorithm is given in Fig. 1. After video segmentation step, the cast indexing procedure is employed to generate the story boards of cast in the video. Then similar key frames are removed using HAC in each scene. An original video summary is constructed by extending the selected key frames with impact factors of scenes and key frames. A further repetitive frame segment detection step is designed to remove redundant information left in the initial video summary.

### 2.1 Video segmentation

In this part, the original video sequence is segmented into shots and scenes, and key frames are extracted from each shot and scene.

Usually a shot is a group of frames with consistent visual characteristics, such as color, texture and motion, and a shot is generated from camera on/off operations. Shot detection is the first step for video analysis. We first detect shot boundaries by color histogram and optical-flow motion features, and extract key frames in each shot by a leader-follower clustering algorithm [22]. First, the middle frame of the shot selected as the first key frame. Then, from this key frame to the two shot boundaries, when the similarity between the current frame and existing key frames exceeds a predefined threshold, a new key frame is extracted. This method could assure that key frames are distinct and, therefore, prevent redundancy [22].

Unlike a shot, a scene is a series of adjacent shots referring to continuous and/or related action, and a scene could contain more syntactic meaning compared with a shot. The graph partition method, *NCuts* (Normally cuts) clustering approach, is employed to segment video into scenes [23]. *NCuts* is a clustering method, and the technical detail of *NCuts* can obtained in [7]. A representative key frame in each scene is selected [32] to construct a story board including scene information.

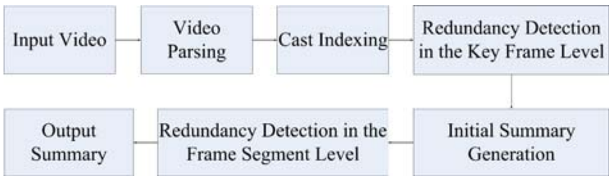
Input Video
Video Parsing
Cast Indexing
Redundancy Detection in the Key Frame Level
Initial Summary Generation
Redundancy Detection in the Frame Segment Level
Output Summary

**Fig. 1** Architecture of the proposed dynamic video summarization algorithm

However, here what we should point out is that these video segmentation algorithms are just used for video preprocessing. Here we choose the effective algorithms in the most recent papers [22, 23, 32] to do video segmentation. Usually the effective algorithms for video segmentation lead to similar video segmentation results.

## 2.2 Cast indexing

When this is a video with cast, we use the cast indexing step to show the main cast list. Cast indexing is useful to discover actors for efficient browsing using face tracking and clustering. As shown in Fig. 2, face tracker first detects faces in each shot and outputs face sets of continuous faces belonging to the same person [13]. Then the local SIFT features of each face is extracted, and cluster similar face sets by *NCuts* clustering method to find main actors and their video clips [7]. One face image for each actor is selected to construct a story board of cast information.

This cast detection procedure is only applied to such videos with cast. This decision is made mutually. When there is no cast in the video, our algorithm will skip this step, and only generate a video summary with a story board of main scenes and the dynamic video skimming using the following two-level redundancy detection procedure. In this step, cast indexing is adapted to enhance the summary performance, and the technical details about how to index cast is not the focus of this paper. More details about cast indexing could be obtained from [7].

The cast detection procedure aims to provide a brief way to show the video cast and video content. When the video contains casts, this procedure could make the summary much easier to understand. Under the situation of the video contains no cast, the story board of main scenes and the dynamic video skimming still could provide a good summary.

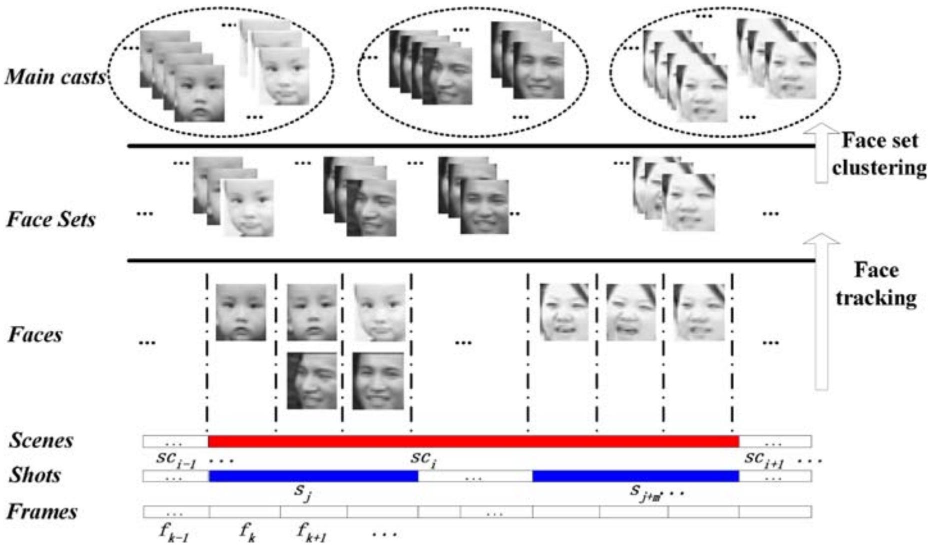

**Fig. 2** Illustration of the cast indexing procedure

## 2.3 Redundancy detection in the key frame level

Video sequences contain much redundant information. The proposed algorithm can shorten the video length without losing main information in the video data by removing similar video content.

After the key frame extraction (in each shot) step, the original video is represented by key frame sequence. We first use HAC to remove similar key frames in each scene. The most similar key frames could be merged by HAC until the intra-cluster distance between any pair of key frames exceeds a pre-selected threshold. The reason for employing HAC is that we could change the threshold of clustering to analyze the repetitive information detection performance. The distance threshold makes sure that the two merged key frames are similar to each other. Only the first appeared key frame in each cluster is left for further processing.

Three features are selected to calculate the distance between each two key frames in the HAC step, and they are the histogram distance based on blocks ($DB$), edge information, and $HSV_H$.

Histogram comparison is widely used in image processing. Compared with pixel comparison, histogram comparison is more tolerant to slow and small object motion from frame to frame. Each frame of the original data first is converted from a color image to a gray image. We break each frame into $r$ equal size regions, and the distance between frame $i$ and frame $j$ is measured by comparing the global histogram difference. The histogram difference is calculated by:

$$DB(i,j) = \frac{\sum_{k=1}^{r} D_b(i,j,k) - \max_{0<k\le r}\{D_b(i,j,k)\}}{r-1}, \tag{1}$$

and

$$D_b(i,j,k) = \frac{\sum_{p=1}^{m-1} |H_{i,k}(p) - H_{j,k}(p)|}{n_p}, \tag{2}$$

where $r$ is the number of blocks, $H_{i,k}(p)$ is the histogram value at gray level $p$ for block $k$ in the $i$th frame and $m$ is the quantity of gray levels; $\max_{0<k\le r}\{D_b(i,j,k)\}$ is the maximum of $D_b(i,j,k)$, and $n_p$ is the pixel number in each block. The purpose of regarding $\max_{0<k\le r}\{D_b(i,j,k)\}$ is to alleviate the influence of noise signals and the effect of camera movements.

One disadvantage of $DB$ is that only small change could be detected with $DB$ when two different images are with similar histograms [11]. Another disadvantage is that it is sensitive to light intensity, such as flashlight.

To achieve accurate analysis of two images with different content but similar histogram, we add the edge information as the second feature. The advantage of edge information is that it is sufficiently invariant to illumination changes and several types of motion, and it is related to the human visual perception of a scene. The main disadvantage is the noise sensitivity. The technical detail of edge distance calculation can obtained from [31].

The histogram distance of H component $HSV_H$ is selected as the third feature to reduce the error analysis caused by using gray histogram comparison and edge analysis. In the HSV color space, the H (hue) component is insensitive to such changes of the frame-to-frame differences caused by change in intensity or shade since hue is independent of saturation and intensity. The $HSV_H$ difference is calculated by:

$$DHSV_H(i, j) = \frac{\sum_{k=1}^{r} HSV_H(i, j, k)}{r-1}, \tag{3}$$

and

$$HSV_H(i, j, k) = \frac{\sum_{p=1}^{m-1} |H_{HSV_{H}i,k}(p) - H_{HSV_{H}j,k}(p)|}{n_p}, \tag{4}$$

where $r$ is the number of blocks, $H_{HSV_{H}i,k}(p)$ is the $HSV_H$ histogram value at $H$ level $p$ for block $k$ in the $i$th frame and $m$ is the quantity of $H$ levels; $num$ is the pixel number in each block.

We set different weight values for these features, and the final key frame difference measure is computed combing these three feature differences by:

$$D(i, j) = W_{DB} \times DB + W_{DEdge} \times DEdge + W_{DHSV_H} \times DHSV_H, \tag{5}$$

where $W$ and $D$ are weight value and distance for each feature, respectively.

As two similar clips may have two key frames with relative large difference, we do not calculate each D value directly between two key frames. Ten frames around each key frame are treated as one frame set, and we compute the distance between these two frame sets as the key frame distance, as shown in Fig. 3. From experiment results, if two video clips represented by two key frames are similar, there should be at least 10 similar frames in these two video clips.

The distance between two frame sets is defined as follows. We compute the distance between each combination of two frames selected from the two sets, and the final $D$ value of the current feature is the mean $D$ of these frame distance values:

$$D = \frac{1}{n \times n} \sum_{i=1}^{n} \sum_{j=1}^{n} D(i, j), \tag{6}$$

and here $n = 11$.

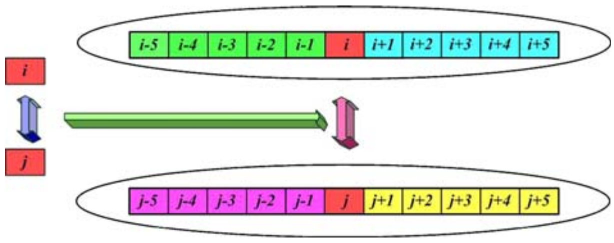

**Fig. 3** Distance between two frame sets

## 2.4 Initial video summary generation

To construct a good summary, we discard or emphasize many segments [4]. To select appropriate key frames for summarization, we define importance factors in the scene level and the key frame level.

Generally, a long video shot could bring users more information, and the important score of scene is based on its rarity and duration. A video clip should be important if it is with long time duration and much information. This penalizes both repetitive shots and shots of insignificant duration. The importance factor of the *i*th scene is calculated by:

$$IF_{Scene_i} = 1 + \sqrt{\frac{F_c}{F_t} \times \frac{Shot_c}{Shot_t}}, \tag{7}$$

where $IF_{Scene_i}$ is the important factor of the *i*th scene, $F_c$ is the the frame number in current scene and $F_t$ is the total frame number in the video sequence. $Shot_c$ is the shot number in current scene, and $Shot_t$ is the total shot number.

Concerning the video content importance, key frames with faces, sounds, moving objects and high repetitive number bring users more information and they are important to be reserved in the video summary. Therefore, the important factor of the *k*th key frame is related with the appeared face number, audio energy, motion magnitude, and repetitive number, and calculated by:

$$IF_{\text{key}}(k) = 1 + w_f L_f(k) + w_a L_a(k) + w_m L_m(k) + w_r L_r(k)\,, \tag{8}$$

where $IF_{key}$ is the important factor of the *k*th key frame, $L_f$, $L_a$, $L_m$, $L_r$ are the level of face number, audio's MFCC feature energy, motion magnitude and key frame repetitive number (the key frame cluster size in HAC step), respectively. The motion magnitude $L_m$ is calculated as follows. Assuming $F^x$ and $F^y$ are the motion vectors in the horizontal axis and the vertical axis, respectively. Then $L_m = |F^x| + |F^y|$. The detail description of $L$ is given in Table 1, where $N_{face}$ is the face number in the key frame, $E_a$ and $E_m$ are the audio energy and motion magnitude values of the key frame, $T_a$ and $T_m$ are the thresholds of audio energy and motion magnitude, and $N_r$ is repetitive number of key frames in HAC step. $w_f$, $w_a$,$w_m$, $w_r$ are their corresponding weights, as shown in Table 1. We set a minimum value of IF as 1.0 to guarantee that each key frame selected could be represented well for easy understanding.

**Table 1** Weight values of different component

| Type | $L = 0$ | $L = 1$ | $L = 2$ | $W$ |
|---|---|---|---|---|
| *Face* | $N_{face} = 0$ | $N_{face} = 1$ | $N_{face} > 1$ | 0.4 |
| *Audio* | $E_a < T_a$ | $E_a > T_a$ | – | 0.2 |
| *Motion* | $E_m < T_m$ | $E_m > T_m$ | – | 0.2 |
| *Repetitive* | $N_r = 1$ | $N_r = 2$ | $N_r > 2$ | 0.3 |

Experiments [8] show that the ideal playback length for each video segment is between 2.0–3.0 s. A playback time shorter than 2 s may result in a non-smooth and choppy video show, while a playback time longer than 3.0 s yields a lengthy and slow-paced one. Based on this criterion, we set the basic extension time length of key frames as 2.0–3.0 s. The close-up shots and the long shots are different for human reading. The close-up shots always contain less information and could be understood quickly compared with long shots. On the other hand, the long shots have more information and need more time to obtain all content in it for human observers. With this analysis, we set the close-up shots with an extension length of 2.0 s, and the long shots with an extension length of 3.0 s.

Face information could bring users more information, and the video clips with faces may be important for the description of video content meaning. Here we use one rule to detect the close-up shots and long shots. Only the frames with large faces are set as the close-up shot frames. All other frames are long shot frames.

The final extension time length of each key frame is calculated by:

$$T_k = IF_{\text{Scene}}(k) * IF_{\text{key}}(k) * T_B. \tag{9}$$

Here the $T_B$ value is 2.0 s for close-up shot key frames and 3.0 s for long shot key frames. We extend each reserved key frame to $T_k$ seconds video segment. We also make sure that the extended video segment does not cross the shot boundaries. An initial video summary is generated by combining these extended video clips under the original temporal order.

## 2.5 Repetitive segment detection in the frame segment level

Though the redundancy detection in the key frame level could remove redundant key frames, there could be some repetitive segments in the initial video summary. In the HAC step, key frames are the analyzed units. Generally, two similar shots are with similar key frames. Large camera or object movements in the shot could add the difficulties for key frame extraction. Algorithms for key frame extraction could be mainly classified into two categories. In the first category, these algorithms, such as the one in [28], try to group frames into several clusters, and select one frame from each clusters as one key frame. These algorithms can be called the CLUSTING methods. In the second category, the algorithm, such as those in [9, 10, 22], first define a similarity measure between frames, and key frames are detected if the similarity value between the current frame and the last key frame is smaller than a given threshold. These algorithms can be called the THRESHOLD methods. Dramatic camera or object movements could enlarge the distance between two frames in one shot. Both the CLUSTING methods and the THRESHOLD methods depend on the similarities of frames. When the similarities of frames are large, these key frame extraction algorithms could achieve good performance. When the similarities of frames are not large enough, key frames extracted may not have large similarities compared with other frames in the same shot. So key frames generated

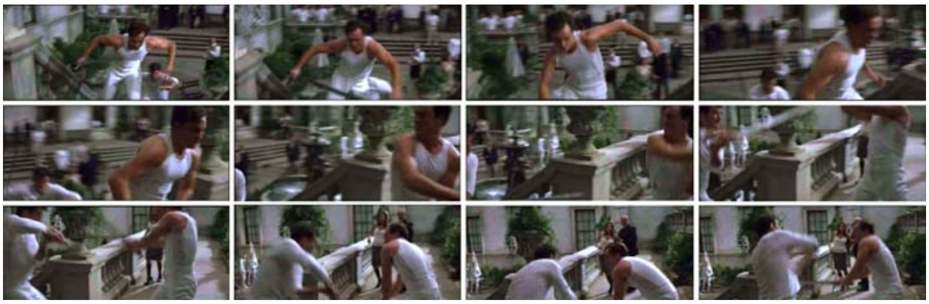
**Fig. 4** An example video shot in the movie *Die another day*

from similar shots may have relative large difference if there are large camera or object movements in the shot.

One method to solve this problem is to use more key frames, while the drawback of using more key frames is the increase of computation load. The application of key frames is to reduce the computation complexity, so in this step, utilizing more key frames is not suitable. When we compare two video clips using all the frames, this could be treated as an extreme condition: using all frames as key frames. These frames, of course, could describe the video content exactly, and could achieve good redundancy detection performance in the frame segment level.

To analyze this problem, two clips with similar content are compared. In this experiment, a shot with 150 frames is selected as the testing data set. In these two clips, clip 1 contains the first 100 frames of this shot, and clip 2 contains the last 100 frames of this shot. These examples are given in Figs. 4, 5, 6. Figure 4 provides the overview of the original video shot. Two key frames extracted from clip 1 are shown in Fig. 5, and two key frames extracted from clip 2 are shown in Fig. 6. Since created from the same shot, clip 1 and clip 2 contain similar video content. Because there are large camera and the object (actors) movements in the shot, key frames extracted from clip 1 and clip 2 vary greatly. Assuming both of these two clips are in the video sequence, the HAC step may not remove this similar video content, and leave these two clips in the initial video summary. Under this situation, there will be some repetitive information left after the HAC step.

In this part, a repetitive frame segment detection method is proposed to remove redundant information contained in the initial video summary. The *Smith-Waterman* local alignment algorithm [26] (SW) is employed to detect similar frame segments. The SW algorithm is a classical string matching method, and see [26] for the details about the SW algorithm.

**Fig. 5** Two key frames extracted from the first 100 frames of the shot in Fig. 4

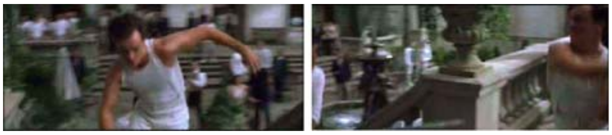

**Fig. 6** Two key frames extracted from the last 100 frames of the shot in Fig. 4

Given two frame segments $S1$ and $S2$ with length $l_1$ and $l_2$, the SW algorithm computes the similarity matrix $H(i, j)$ to detect the similar video segment, using the following equations:

$$H(i,j) = \max \begin{cases} 0 \\ E(i,j) \\ F(i,j) \\ H(i-1,j-1) + sbt(S1_i, S2_j) \end{cases}$$

$$E(i,j) = \max \begin{cases} H(i,j-1) - \alpha \\ E(i,j-1) - \beta \end{cases}$$

$$F(i,j) = \max \begin{cases} H(i-1,j) - \alpha \\ F(i-1,j) - \beta \end{cases}$$

$$\begin{aligned} with H(i,0) &= E(i,0) = 0, 0 \le i \le l_1 \\ H(0,j) &= F(i,0) = 0, 0 \le j \le l_2, \end{aligned} \tag{10}$$

where $sbt(S1_i, S2_j)$ is the similarity value between character $S1_i$, and $S2_j$. Affine gap costs are defined as follows: $\alpha$ is the cost of the first gap; $\beta$ is the cost of the following gaps. Each position of the matrix H is a similarity value ending local alignment at the position $(i, j)$. The common subsequence of $S1$ and $S2$ producing this value can be determined by a tracing back procedure.

In the task of finding video segments with similar content in the initial video summary, we only trace back the positions in the upper triangle matrix of $H$ because $H$ is a symmetrical matrix when $S1 = S2$. Then we merge matched blocks located in the same horizontal, or vertical strip regions and output long enough repetitive segments satisfying its average similarity exceeds a predefined threshold.

After repetitive segment detection, all similar segments except the first appeared segments are discarded and output the final video summary. The final dynamic video summary consists the scene list, the cast list, and the summary generated by the two-level redundancy detection method.

## 3 Experimental results

We have tested the proposed video summary algorithm on 10 video sequences. The testing data set is selected from movies (Die another day (*Die*) and Mission impossible III (*M III*)), sports programs (one match of AC Milan (*AC*) and one tennis program (*Tennis*)), news (one CNN student news (*CNN*) and one BBC world news (*BBC*)), TV series (Dachangjin (*Da*) and Friends) and other documents (Shrek II and the lion king (*King*)), lasting from 25 to 85 min. As most of movies are saved by two or more CDs, we only tested one part of these movies. The details of the testing

**Table 2** Video summarization performance

| Video | $L$ | $N_{sc}$ | $N_{kf}$ | $K_l$ | $N_i$ | $C_1$ | $In$ | Sa | $F_l$ | $C_2$ | $Cr$ | $In$ | Sa |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Die | 5137 | 33 | 2489 | 203 | 609 | 11.9 | 9.0 | 8.5 | 391 | 64.2 | 7.6 | 8.5 | 10.0 |
| M III | 5124 | 28 | 2532 | 218 | 610 | 11.9 | 10.0 | 8.0 | 416 | 68.1 | 8.1 | 9.8 | 9.0 |
| AC | 4382 | 21 | 2103 | 153 | 428 | 9.8 | 8.5 | 7.6 | 336 | 78.5 | 7.7 | 8.0 | 8.0 |
| Ball | 2847 | 10 | 1871 | 129 | 284 | 10.0 | 9.2 | 6.6 | 238 | 83.8 | 8.4 | 8.8 | 7.0 |
| CNN | 1317 | 12 | 273 | 94 | 244 | 18.6 | 9.4 | 8.1 | 191 | 78.3 | 14.5 | 9.2 | 8.5 |
| BBC | 1238 | 15 | 241 | 69 | 193 | 15.6 | 8.9 | 7.5 | 145 | 75.0 | 11.7 | 8.7 | 8.0 |
| Da | 2650 | 23 | 766 | 142 | 369 | 13.9 | 9.0 | 7.9 | 284 | 77.0 | 10.7 | 8.6 | 8.5 |
| Friends | 1509 | 13 | 472 | 82 | 238 | 15.8 | 9.4 | 7.3 | 201 | 84.6 | 13.3 | 9.3 | 7.5 |
| Shrek | 5053 | 20 | 2386 | 203 | 508 | 10.0 | 7.4 | 7.1 | 394 | 77.7 | 7.8 | 7.4 | 8.0 |
| King | 3638 | 17 | 1845 | 127 | 305 | 8.4 | 8.5 | 7.8 | 237 | 77.7 | 6.6 | 8.2 | 8.0 |
| Average | 3290 | 19 | 1498 | 142 | 379 | 12.6 | 8.9 | 7.6 | 283 | 77.0 | 9.6 | 8.7 | 8.3 |

video data are given in Table 2, where $L$, $N_{sc}$, $N_{kf}$ are the video length (seconds), the number of scenes, and the number of key frames in original videos, respectively. $K_l$ is the key frame selected after HAC step, $C_1$ and $C_2$ are the compress ratio values (%) in the two redundancy detection procedures. $N_i$ is the initial video summary length (seconds), and $F_l$ is the final summary length (seconds).

Three evaluation scores could be obtained based on the comparison between the video sequences and the output summaries: Informativeness ($In$), Compression ratio ($Cr$) and Satisfaction ($Sa$). Informativeness score evaluates how many objects/events of the original video are included in the summarized video. Compression ratio is the ratio of the summarized video length $L'$ divided by the original video length $L$. Satisfaction score indicates how satisfactory the reviewers understand and prefer the summarized video. For $In$ and $Sa$ scores, five human observers watch these summaries and give the scores. The best score is 10 and the worst score is 0.

Our video summaries first show the story boards to let users know about main scenes and main cast and then show the dynamic video summary.

Figure 7 shows the story boards of TV series *Friends*. A video summary example extracted from *Friends* is given in Fig. 8. Table 2 describes the performance of the two-level redundancy detection procedure and the final video summary results.

We evaluate the performance of the two-level redundancy detection procedure. As the key frame clustering could remove most of the repetitive key frames, the key frame level redundancy detection procedure could achieve an average compress ratio of 12.6%. The repetitive segment detection further shortens the initial video

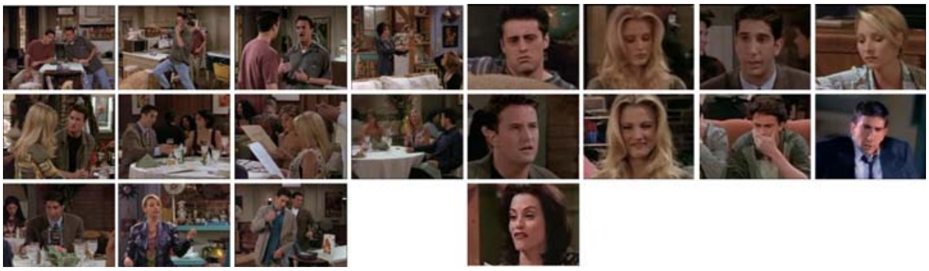

**Fig. 7** Story boards of main scenes and main cast in *Friends*

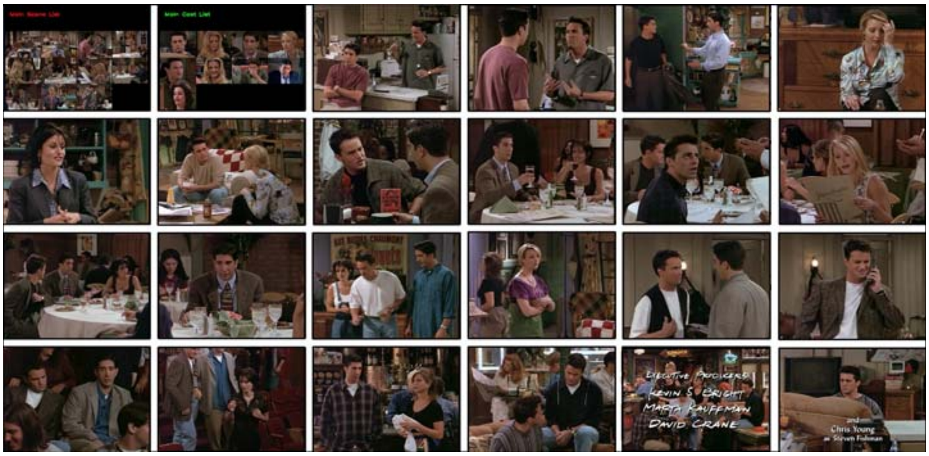

**Fig. 8** A summary generated from *Friends* using the proposed algorithm

summary length with a $C_2 = 77.0\%$. Totally, the two-level redundancy removing procedure could achieve a compress ratio of 9.6%.

The average performance of the final summary with both story board and video skimming achieves $In = 8.7$, $Cr = 9.6\%$ and $Sa = 8.25$. The compress ratio changes significantly when the types of videos are different. The *Die* movie contains many repetitive and short shots and is compressed with the best $Cr = 7.6\%$ among all the testing videos. However, since the shots of "BBC news" are long and independent, its compression ratio is $Cr = 14.5\%$. The proposed dynamic videos summarization algorithm could reserve most of important video content in the original video sequences when the $Cr$ value is low. In the summary of *Die*, the $In$ value is still 8.5 with the best $Cr$ value of 7.6%. In the summary of *CNN*, the $In$ value is 9.2 with a large $Cr$ value of 14.5%. The results show that the summaries generated by the proposed algorithm could retain most of important information in the original video sequence.

The proposed summarization algorithm runs at CPU Dothan 1.86 GHz, 2 G DDR533 RAM and Microsoft Windows XP Professional OS. The average running speed is 124 frames per second, which could be fit for real-time computing.

We have tested other video summarization methods on the same data set, and the comparison of the summarization performance is shown in Table 3. These algorithms are selected for the following reasons. First, these algorithms are effective summarization algorithms published in recent years. Second, [12, 24] provide video summarization algorithms with multi-level summarization. We could compared with these two algorithms in the filed of multi-level summarization performance.

**Table 3** Comparison with other methods

| | Our algorithm | Method [12] | Method [24] |
|---|---|---|---|
| $Cr$ | 9.6 | 14.1 | 10.8 |
| $In$ | 8.7 | 8.4 | 8.1 |
| $Sa$ | 8.3 | 8.1 | 7.6 |

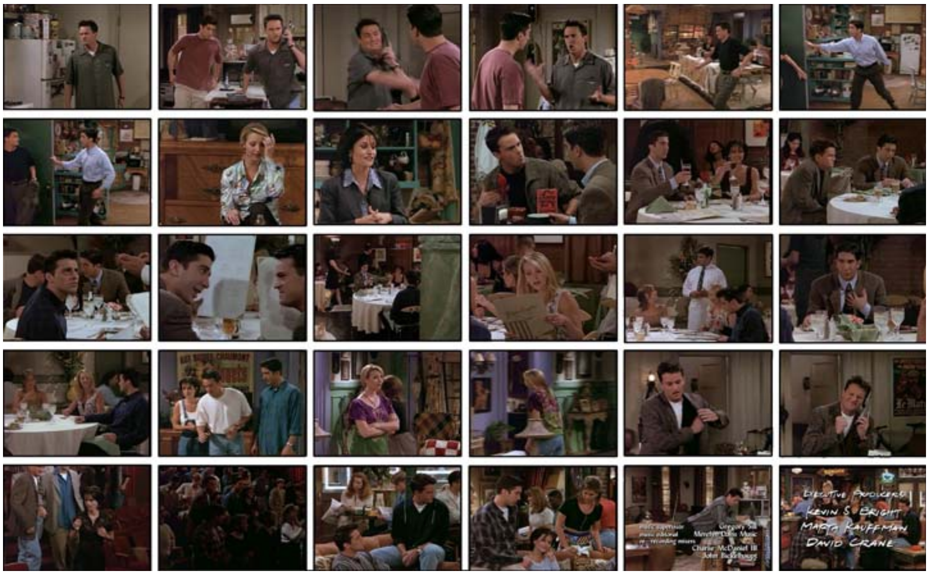

**Fig. 9** A summary example generated from *Friends* using the algorithm in [24]

As shown in Table 3, the compress effect of the proposed algorithm is the best, and it achieves an average informativeness score of 8.7. Other two methods obtain the informativeness scores of 8.4 and 8.1. As we use two redundancy information detection steps, our method could remove most of repetitive information in the original video sequence. The proposed algorithm could achieve a compress ratio of 9.6%, while other methods are with compress ratios of 14.1% and 10.8%. As shown in the Table 3, with a low compress ratio, the summaries generated by the proposed algorithm still could gain a *Sa* value of 8.3.

A video summary generated by [24] from *Friends* is given in Fig. 9. As shown in Fig. 9, there is still some similar content in the summary generated by [24]. Comparing summaries in Fig. 9 and Fig. 8, our algorithm could remove most similar video content and achieve a brief video summary.

## 4 Conclusions and future work

In this paper, we propose an algorithm for video summarization with a two-level redundancy detection procedure. The summary algorithm first generates story boards to let users know about main scenes and main cast (only when this is a video with cast). Then it removes redundancy information in the key frame level. An initial video summary is constructed based on important factors of scenes and key frames. Finally, it further removes redundancy information of the initial video summary in the frame segment level. The experimental results indicate a very promising performance for the proposed video summarization method. Compared with other methods, our algorithm could achieve a compress ratio of 9.6%, an informativeness score of 8.7 and a satisfaction score of 8.3.

The future work includes precise key frame clustering method and summary reconstruction approach. Effective method based on audio and motion analysis is another further work.

**Acknowledgements** The authors would like to thank the reviewers and editors whose comments and suggestions have greatly improved this paper.

## References

1. Benini S, Bianchetti A, Leonardi R, Migliorati P (2006) Extraction of significant video summaries by dendrogram analysis. In: Proceeding of IEEE international conference on image processing. IEEE, Piscataway, pp 133–136
2. Calic J, Gibson D, Campbell N (2007) Efficient layout of comic-like video summaries. IEEE Trans Circuits Syst Video Technol 17(7):931–936
3. Cernekova Z, Nikou C (2006) Information theory-based shot cut/fade detection and video summarization. IEEE Trans Circuits Syst Video Technol 16(1):82–91
4. Cheng W, Xu D (2003) An approach to generating two-level video abstraction. In: Proceeding of international conference on machine learning and cybernetics, vol 5. IEEE, Piscataway, pp 2896–2900
5. Ciocca G, Schettini R (2006) Supervised and unsupervised classification post-processing for visual video summaries. IEEE Trans Consum Electron 52(2):630–638
6. Ferman A, Tekalp A (2003) Two-stage hierarchical video summary extraction to match low-level user browsing preferences. IEEE Trans Multimedia 5(2):244–256
7. Gao Y, Wang T, Li J (2007) Cast indexing for videos by ncuts and page ranking. In: Proceeding of ACM international conference on image and video retrieval. ACM, New York, pp 441–447
8. Gong Y, Liu X (2001) Video summarization with minimal visual content redundancies. In: Proceeding of IEEE international conference on image processing. IEEE, Piscataway, pp 362–265
9. Jain A, Vailaya A, Wei X (1999) Query by video clip. Multimedia Syst 7(5):369–384
10. Kim S, Park R (2002) An efficient algorithm for video sequence matching using the modified hausdorff distance and the directed divergence. IEEE Trans Circuits Syst Video Technol 12(7):592–596
11. Koprinska I, Carrato S (2001) Temporal video segmentation: a survey. Signal Process Image Commun 16(5):477–500
12. Lee J, Oh J, Hwang S (2005) Scenario based dynamic video abstractions using graph matching. In: Proceeding of ACM international conference on multimedia. ACM, New York, pp 810–819
13. Li Y, Ai H, Huang C (2006) Robust head tracking with particles based on multiple cues fusion. In: Proceeding of European conference on computer vision. Springer, Heidelberg, pp 29–39
14. Li Z, Schuster G, Katsaggelos A (2005) Rate-distortion optimal video summary generation. IEEE Trans Image Process 14(10):1550–1560
15. Liu T, Katpelly R (2006) Content-adaptive video summarization combining queueing and clustering. In: Proceeding of IEEE international conference on image processing. IEEE, Piscataway, pp 145–148
16. Lu S, Lyu M, King I (2004) Video summarization by spatial-temporal graph optimization. In: Proceedings of the 2004 international symposium on circuits and systems. IEEE, Piscataway, pp 197–200
17. Ngo C, Ma Y, Zhang HJ (2003) Automatic video summarization by graph modeling. In: Proceeding of IEEE international conference on computer vision, vol 1. IEEE, Piscataway, pp 104–109
18. Otsuka I, Nakane K, Divakaran A (2005) A highlight scene detection and video summarization system using audio feature for a personal video recorder. IEEE Trans Consum Electron 51(1):112–116
19. Peker K, Otsuka I, Divakaran A (2006) Broadcast video program summarization using face tracks. In: Proceedings of international conference on multimedia and exp. IEEE, Piscataway, pp 1053–1056
20. Peng Y, Ngo C (2006) Clip-based similarity measure for query-dependent clip retrieval and video summarization. IEEE Trans Circuits Syst Video Technol 16(5):612–627

21. Porter S, Mirmehdi M, Thomas B (2003) A shortest path representation for video summarization. In: Proceeding of IEEE international conference on image analysis and processing. IEEE, Piscataway, pp 460–465
22. Rasheed Z, Shah M (2003) Scene detection in Hollywood movies and TV shows. In: Proceeding IEEE international conference on computer vision and pattern recognition. IEEE, Piscataway, pp 343–348
23. Rasheed Z, Shah M (2005) Detection and representation of scenes in videos. IEEE Trans Multimedia 7(6):1097–1105
24. Scharcanski J, Gaviao W (2006) Hierarchical summarization of diagnostic hysteroscopy videos. In: Proceeding of IEEE international conference on image processing. IEEE, Piscataway, pp 129–132
25. Shipman S, Radhakrishan R, Divakaran A (2006) Architecture for video summarization services over home networks and the Internet. In: Proceedings of international conference on consumer electronics. IEEE, Piscataway, pp 201–202
26. Smith T, Waterman M (1981) Identification of common molecular subsequences. J Mol Biol 147:195–197
27. Song B, Vaswani N, Roy-Chowdhury A (2006) Summarization and indexing of human activity sequences. In: Proceeding of IEEE international conference on image processing. IEEE, Piscataway, pp 2925–2928
28. Sze K, Lam K, Qiu G (2005) A new key frame representation for video segment retrieval. IEEE Trans Circuits Syst Video Technol 15(9):1148–1155
29. Uchihashi S, Foote J, Girgensohn A (1999) Video manga: generating semantically meaningful video summaries. In: Proceeding of ACM conference on multimedia. ACM, New York, pp 383–392
30. You J, Liu G, Sun L (2007) A multiple visual models based perceptive analysis framework for multilevel video summarization. IEEE Trans Circuits Syst Video Technol 17(3):273–285
31. Zabih R, Miller J, Mai K (1999) A feature-based algorithm for detecting and classifying production effects. Multimedia Syst 7:119–128
32. Zhao Y, Wang T, Wang P (2007) Scene segmentation and categorization using ncuts. In: Proceeding of IEEE workshop of computer vision and pattern recognition. IEEE, Piscataway, pp 1–7
33. Zhu X, Wu X (2003) Sequential association mining for video summarization. In: Proceeding of IEEE international conference on multimedia and expo. IEEE, Piscataway, pp 333–336

**Yue Gao** is currently a Ph.D candidate student in Department of Automation at Tsinghua University, China. He received his B.S. in Electronic Engineering from the Harbin Institute of Technology, China, in 2005, and his M.E in School of Software from Tsinghua University, China, in 2008, respectively. His research interests include computer vision, machine learning, and pattern recognition.

**Wei-Bo Wang** is currently a master candidate student in School of Software at Tsinghua University. He received his B.S. in 2006, from Nanjing University, as a Software Engineering major. His research interest contains computer graphics and computer vision.

**Jun-Hai Yong** is currently a professor in School of Software at Tsinghua University, China. He received his B.S. and Ph.D. in computer science from Tsinghua University, China, in 1996 and 2001, respectively. He held a visiting researcher position in the Department of Computer Science at Hong Kong University of Science & Technology in 2000. He was a post doctoral fellow in the Department of Computer Science at the University of Kentucky from 2000 to 2002. His research interests include computer-aided design, computer graphics, computer animation, and software engineering.

**He-Jin Gu** is a senior professor at Jiangxi Academy of Sciences. He was a visiting professor at the State University of New York in USA from 2005 to 2006. He held a visiting professor position at School of Software, Tsinghua University in 2007. He has obtained three Awards of Science and Technology from the ministries of China and Jiangxi Province. His research interests include computer-aided design, computer graphics, and software engineering.