Multimedia Tools and Applications, 11, 347-358, 2000
© 2000 Kluwer Academic Publishers. Manufactured in The Netherlands.

Time-Constrained Keyframe Selection Technique

ANDREAS GIRGENSOHN andreasg@pal.xerox.com
JOHN BORECZKY johnb@pal.xerox.com
FX Palo Alto Laboratory, 3400 Hillview Avenue, Palo Alto, CA 94304, USA

Abstract. In accessing large collections of digitized videos, it is often difficult to find both the appropriate video file and the portion of the video that is of interest. This paper describes a novel technique for determining keyframes that are different from each other and provide a good representation of the whole video. We use keyframes to distinguish videos from each other, to summarize videos, and to provide access points into them. The technique can determine any number of keyframes by clustering the frames in a video and by selecting a representative frame from each cluster. Temporal constraints are used to filter out some clusters and to determine the representative frame for a cluster. Desirable visual features can be emphasized in the set of keyframes. An application for browsing a collection of videos makes use of the keyframes to support skimming and to provide visual summaries.

Keywords: video collections, video browsing, key frame selection, temporal constraints

1. Introduction

When reviewing collections of videos such as recorded meetings or presentations, users are often interested only in an overview of these documents. At FX Palo Alto Laboratory, weekly staff meetings and other seminars and presentations are held in a conference room outfitted with several video cameras. All formal meetings and most presentations are videotaped, MPEG-encoded, and made available to the staff via the company intranet. These videos amount to about three hours per week; currently we have more than 200 hours of video in our database. It is often difficult to find both the appropriate video file and the portion of the video that is of interest. As video is used more and more as a permanent record for decisions made in meetings and video conferences, it becomes more important to locate the passages containing relevant information or even the meeting in which a decision was made. We want to help users locate specific video passages quickly and provide them with visual summaries of the videos.

Keyframes can be used to distinguish videos from each other, to summarize videos, and to provide access points into them. Well-chosen keyframes can help video selection and make the listing more visually appealing. However, it is difficult to determine a single frame that best represents the whole video. It also can be difficult to distinguish videos based on a single keyframe, so we provide a number of keyframes. Our goal is to determine a set of keyframes that describes an entire video clip well.

We want to provide a variable number of distinct keyframes that provide a good representation of all the frames in the video. We use hierarchical clustering and select one frame from each cluster. Furthermore, if more or fewer keyframes are desired, one can just increase or decrease the number of clusters. We use temporal constraints to filter out unsuitable clusters and to select a representative frame for each cluster.

In the next section, we compare our approach to the related work described in the literature. After describing the details of our approach, we introduce an application that makes use of our keyframe selection technique to provide access to a collection of videos. We conclude with directions for future work.

2. Related work

Most of the related work has been applied to professionally produced material such as movies, TV comedies, and news programs. That work has concentrated on breaking video into shots and then finding keyframes corresponding to those shots. The results of that work are not directly applicable to our application. First, videotaped meetings and presentations are produced in a more ad hoc fashion so that we cannot rely on established production practices. Second, using one or more keyframes from each shot produces more keyframes than needed for our application.

Many of the systems described in the literature use a constant number of keyframes for each detected shot. Tonomura et al. [9] use the first frame of each shot as a keyframe. Ueda et al. [l0] represent shots with two keyframes-the first and last frames of each shot. Ferman et al. [2] use clustering on the frames within each shot. The frame closest to the center of the largest cluster is selected as the keyframe for that shot. Taniguchi et al. [8] generate a composite image to represent shots with camera motion.

Other systems use more keyframes to represent shots that have more interesting visual content. Zhang et al. [12] and Günsel et al. [5] segment the video into shots and select the first clean frame of each shot as a keyframe. Other frames in the shot that are sufficiently different from the last keyframe are marked as keyframes as well.

One way to reduce the number of keyframes is to remove redundancies. Yeung et al. [11] select one keyframe for each video shot. These keyframes are then clustered based on visual similarity and temporal distance. Since their purpose is to group shots to determine video structure, the temporal constraints are used to prevent keyframes that occur far apart in time from being grouped together. Our work uses temporal constraints to prevent keyframes from appearing too close together in time.

Sun et al. [7] divide a video into intervals of equal length and determine the intervals with the largest dissimilarity between the first and last frame. All frames from those intervals are kept whereas only two frames from each of the remaining intervals is kept. The process is repeated until the desired number of frames or less is left. This approach takes only fairly localized similarities into consideration and cannot apply constraints for frame distribution or minimum distance.

The system described by Tonomura [9] also can provide an alternate representation of a video sequence that uses keyframes that are evenly spaced, ignoring shot boundaries. Our efforts found that evenly spaced keyframes do not provide sufficient coverage of video content.

The reviewed systems do not meet our goal of extracting an exact number of representative keyframes. Existing systems either provide only limited control over the number of keyframes or do not perform an adequate job of finding truly representative frames. In addition, other systems do not apply temporal constraints for keyframe distribution and spacing.

3. Technical details

This paper describes a novel technique for determining keyframes that are different from each other and provide a good representation of the whole video. We cluster similar frames so that we can extract different keyframes by selecting one frame from each cluster. Several steps are performed in determining keyframes.

First, we determine a number of candidate frames to be used as the input to the hierarchical clustering algorithm. Selecting evenly spaced frames either returns too many frames to compute the cluster hierarchy in a reasonable amount of time or uses intervals between frames that are so large that important details can be missed. Instead, we start with a collection of frames that are already fairly different from each other. We accomplish that by collecting all adjacent pairs of frames that exhibit large differences in an image comparison. Rather than setting a fixed threshold for the comparison score, we keep a fixed number of frames that are most dissimilar to their neighbors.

The selected candidates are then clustered with a hierarchical agglomerative clustering technique. Some of the clusters are filtered out using temporal constraints. Afterwards, we select as many clusters as keyframes needed. From each cluster the member that best meets some temporal constraints is selected as the keyframe. Such constraints require an even distribution of keyframes over the length of the video and a minimum distance between keyframes.

To change the emphasis of certain classes of keyframes such as close-ups or slide images, we modify the distance function used by the clustering algorithm. Decreasing the distance between less desirable images increases the likelihood that they will end up in the same cluster and therefore be represented less in the set of keyframes. Increasing the distance between desirable images has the opposite effect.

3.1. Selecting candidate frames

In a collection of equidistant frames, long scenes without much change are heavily emphasized, as shown in figure 1. At the same time, frames from very short shots might be missed

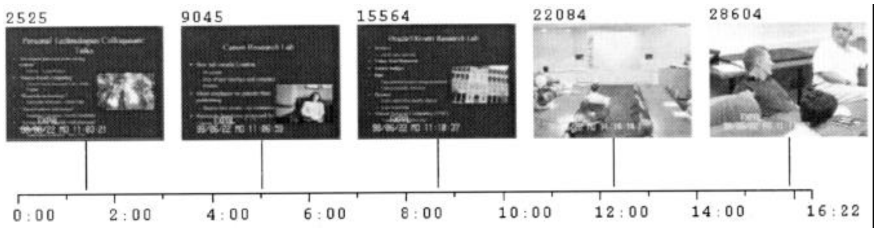


Figure 1. Equidistant keyframes.

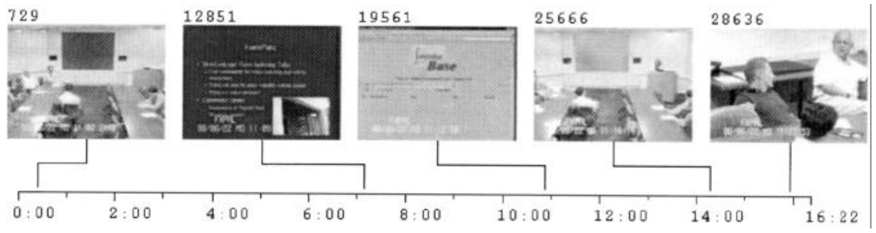


Figure 2. Automatically selected keyframes.

with an equidistant selection. Hierarchical clustering of the frames in the video produces better results as shown in figure 2. The selected frames provide a good account of the meeting.

Unfortunately, clustering is a computationally expensive operation so that it is not feasible to attempt to cluster all frames in a video. To overcome the run-time limitations, we collect candidates for keyframes that are dissimilar from each other. Frames that are very similar to other frames will most likely end up in the same cluster so that they do not contribute much to the keyframe selection. We collect all pairs of adjacent frames that show large differences in an image comparison. Including both frames of the pair rather than just one of them addresses situations in which a video sequence changes very gradually over time. Rather than selecting a fixed threshold, we keep a buffer with the N frames that are most dissimilar to their neighbors. If a video contains fewer samples than the buffer size, all sampled frames are clustered. Otherwise, only the most promising ones are considered.

Subsampling frames at half-second intervals produces better results than looking at every frame because very gradual changes become more pronounced in the subsampled set of frames. If fades, dissolves, and video noise created by switching among non-synchronized cameras are issues, one can use detection algorithms described in the literature. To avoid selecting of rolled frames caused by synchronization problems, we have had good success with using an 18-frame window where we compare the current frame both to its immediate predecessor and to the frame at the beginning of the window. Frames with the largest combined distance are used as candidate frames.

The boundaries between dissimilar frames are potential shot boundaries so that our approach can be seen as performing an implicit shot segmentation. However, the number of boundaries between candidate frames is selected to be much larger than the number of shot boundaries expected in the source material, so that our approach considers many more candidates than the usual approaches that use up to three keyframes per shot [2, 9, 10]. The average shot duration is five seconds in a mixture of material [l], so that one can expect about 720 shot boundaries per hour. For our material of captured meetings, that number is significantly lower. We use 1300 candidates for up to one hour long videos with good success. Once all the candidate frames have been determined, they are clustered.

3.2. Clustering frames

Our goal in selecting keyframes is to determine a set of frames that are different from each other and provide a good representation of all the frames in the video. Clustering combines similar frames so that selecting one frame from each cluster meets our goal. Furthermore, if more or fewer keyframes are desired, one can just increase or decrease the number of clusters.

For comparing frames, we investigated a wide variety of common image comparison methods and selected a histogram technique. Histogram-based approaches are very good for detecting overall differences in images. To make the comparison even less susceptible to noise, smooth histograms were used. In such histograms, each bin spills a fraction of its contents into the neighboring bins. To allow for color effects, a three-dimensional histogram was used in the YUV color space with 8 bins for Y (luminance) and 4 bins each for U and V (chroma) for a total of 128 bins. We normalized the luminance before populating the histogram and used a χ^2 metric to compare histograms.

For extracting *M* keyframes, the material is divided into *M* clusters. This approach sidesteps the problem of selecting an appropriate threshold for the cluster size. Frames are clustered using the complete link method of the hierarchical agglomerative clustering technique [6]. This method uses the maximum of the pair-wise distances between the frames in two clusters to determine the intercluster similarity. Other clustering methods can be used as well with slightly different results. Figure 3 shows that the hierarchical clustering is performed by combining the two clusters that produce the smallest combined cluster. Initially, each image represents its own cluster. The altitude of a node in the tree represents the diameter (maximum pair-wise distance of the members) of the combined cluster. Clusters are represented by the member frame closest to the centroid of the cluster. Note that the frames in the tree are not in temporal order. Their position in time can be determined via the frame number at the top-left of every image (30 frames per second). In the example shown, the clusters Cl through C4 are split to extract five keyframes, the direct children of the split clusters (see the emboldened parts of figure 3).

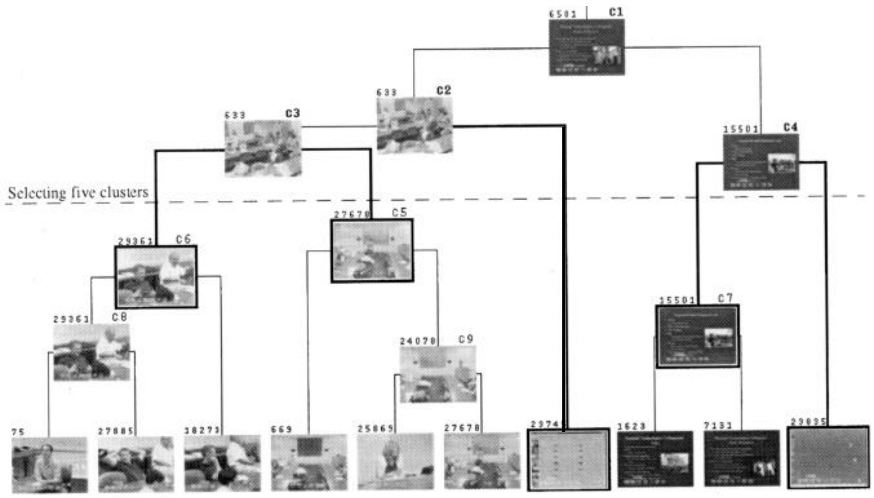


Figure 3. Selecting five keyframes from the cluster hierrarchy.

3.3. *Filtering out clusters*

Clusters containing only a single image are very likely to represent video artifacts such as switching distortion or other types of noise. Frames with such artifacts are undesirable as keyframes, so temporal constraints are used to filter out the corresponding clusters. In general, a cluster should represent at least one uninterrupted sequence of frames of a minimum duration to make sure that video artifacts and other non-significant events are not used as keyframes. The duration threshold depends on the source material. The number of keyframes per hour also influences the minimum duration because the constraint has to be relaxed when more keyframes are requested. In tests with different source materials, we found that using three percent of the average time between keyframes as a threshold produced good results. For example, if there is on average one keyframe every five minutes (300 seconds), each cluster has to have at least one uninterrupted nine-second sequence of frames.

3.4. *Applying temporal constraints for selecting representative frames from the clusters*

Some of the earlier examples produced keyframes that were very unevenly distributed over time and sometimes very close together. Because keyframes are intended to be attached to a time line in the application discussed in the next section, it is desirable to spread them out over time. The application presents keyframes to support skimming through a video so that a semi-uniform distribution of keyframes is desirable. Keyframes should also not be too close together. These issues are addressed by a proper selection of representative frames from the clusters. Members of the same cluster are supposed to be reasonably similar to each other so that any member can serve as the representative of a cluster. This leaves room for applying temporal constraints to the selection of a keyframe from each cluster.

Usually, clusters have members spread all over the duration of the video. It is desirable to pick a representative frame from a time that the cluster dominates to give a good summary of a video. To that end, the longest sequence of members is determined for each selected cluster that is not interrupted by members of another selected cluster. The frame closest to the center of that sequence is chosen. This approach has the added benefit that it maximizes the distance in time between the representative frame and any frame from another cluster so that it becomes more likely that keyframes are not too close together.

Applying the selection strategy described above does not guarantee that there are no large gaps between keyframes. To ensure a semi-uniform distribution of keyframes, the total duration of the source clip is divided into intervals of equal duration that each should contain at least one keyframe. Using half as many intervals as requested keyframes produces good results. The number of selected keyframes in each interval is counted. For intervals that do not contain a keyframe, keyframes are determined with the following method. All intervals containing at least two keyframes are checked in descending keyframe count order. For each of the keyframes in an interval, it is checked whether the corresponding cluster also has a member in the interval without a keyframe. If such a member is found, it is used as a keyframe instead of the previously selected one.

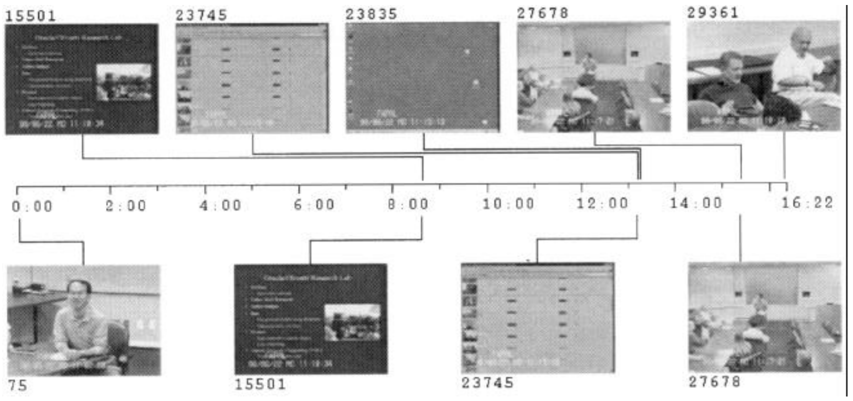


Figure 4. Keyframes with sufficient separation.

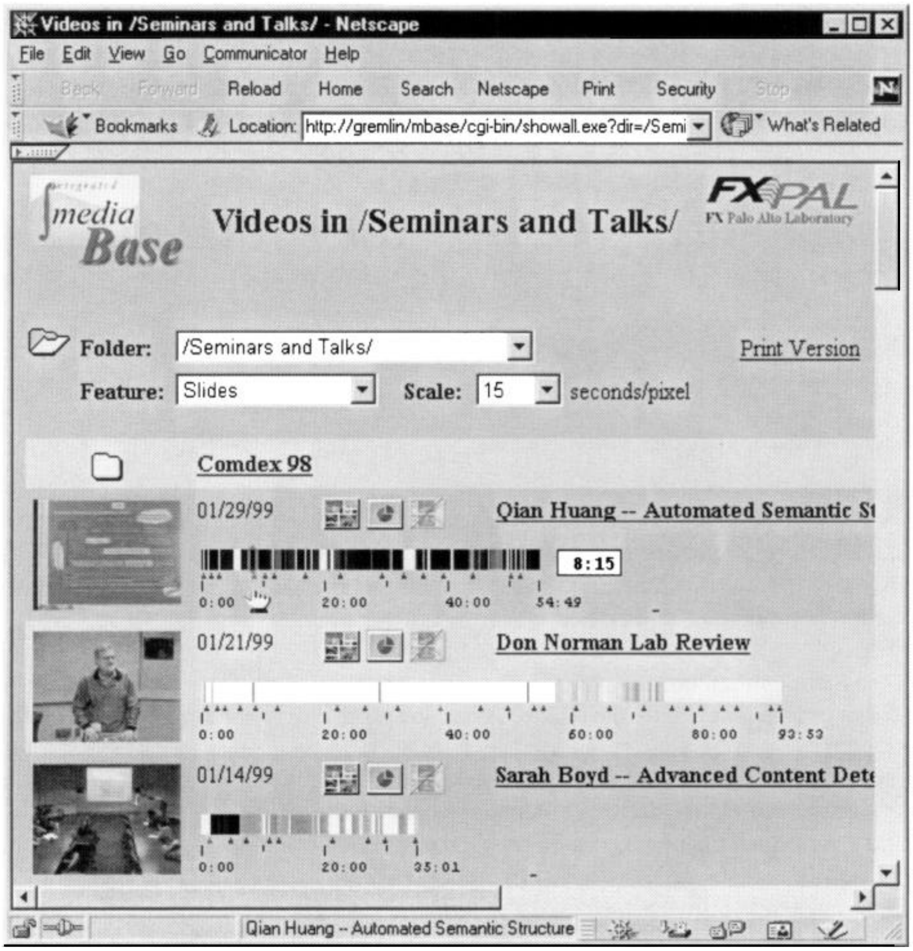
Some applications require a minimum distance between keyframes, for example, to attach all keyframes to a time line at the appropriate locations or to improve skimming through keyframes. After the minimum time between keyframes has been determined, the following algorithm is applied:

1. Find the minimum time between any two selected keyframes.
2. If that time is longer than the required time, the algorithm is done.
3. For the two clusters the keyframes belong to, try to select another member first for one cluster, then for the other cluster, and finally for both clusters at the same time until frames are found that meet the minimum distance requirement.
4. If no such cluster members can be found, decrement the number of selected keyframes and remove one of the two conflicting frames.
5. Continue until all frames meet the constraint.

As an example, this algorithm is applied to the five selected keyframes shown at the top of figure 4 with a minimum distance requirement of two minutes (3600 frames). Frames 23745 and 23835 are only 90 frames apart. Both come from clusters containing only a single frame so that no alternatives are available. Frame 23835 is dropped and the algorithm continues with four keyframes. The distance between frames 27678 and 29361 is 1685 frames. Using frame 75 as a replacement for frame 29361 maximizes the distance to all the other keyframes. Now the minimum distance between any two keyframes is 3933 frames and the algorithm stops. The bottom of figure 4 shows the resulting keyframes.

3.5. *Emphasizing video features*

Some classes of images are preferable as keyframes. For example, close-ups on people tend to provide more information than long shots. In addition, images of slides often are more different from each other than video images. If that tendency remains unchecked, sets of


Videos in /Seminars and Talks/ - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://gremlin/mbase/cgi-bin/showall.exe?dir=/Semi
What's Related
media Base
FX PAL FX Palo Alto Laboratory
Print Version
Qian Huang -- Automated Semantic Structure

Videos in /Seminars and Talks/

Folder: /Seminars and Talks/

Feature: Slides Scale: 15 seconds/pixel

Comdex 98

01/29/99 Qian Huang -- Automated Semantic St
8:15
0:00 20:00 40:00 54:49

01/21/99 Don Norman Lab Review
0:00 20:00 40:00 60:00 80:00 93:53

01/14/99 Sarah Boyd -- Advanced Content Det
0:00 20:00 35:01

Figure 5. Web-based video directory browser.

keyframes are heavily populated by slide images. Therefore, it is desirable to increase the proportion of images such as close-ups on people and to decrease the proportion of slide images in the set of selected keyframes.

To emphasize or de-emphasize certain classes of keyframes, the comparison score can be modified according to class membership. In previous work [4], we discussed the use of confidence scores to represent detected features and the computation of such scores with statistical models. One example is the detection of Microsoft PowerPoint slides. Figure 5 presents the likelihood for the presence of a slide as a grayscale plot along the time line. Such slides are often fairly different from each other so that they would dominate the set of selected keyframes. To avoid this effect and to de-emphasize that class of images, the distance measure of two images is reduced by a factor a times the product of the feature confidences if the feature confidence for both of them is above a threshold b (e.g., 0.5):

$$\text{dist}' = \text{dist}(x, y) \times (1 - a \times \text{conf}(x) \times \text{conf}(y)); \quad 0 \le \alpha \le 1; \quad \text{conf}(x) > b; \text{conf}(y) > b$$

Reducing the distance measure among less desirable images increases the likelihood that those images will be included in the same cluster and thus represented less in the set of keyframes. A feature can be emphasized by increasing the distance among the desirable images or by decreasing the distance among all the remaining images:

$$\text{dist}' = \text{dist}(x, y) \times (1 + (a \times \text{conf}(x) \times \text{conf}(y)); \quad 0 \le a < \infty; \text{conf}(x) > b; \text{conf}(y) > b$$

$$\text{dist}'' = \text{dist}(x, y) \times (1 - a \times (1 - \text{conf}(x)) \times (1 - \text{conf}(y))); \quad 0 \le a \le 1; \text{conf}(x) < b; \text{conf}(y) < b$$

Manipulating the distance function is sufficient for manipulating the clustering behavior so that the less desirable images are more likely to be clustered together. For our collection of captured meetings, this approach has reduced the number of slide images included in the keyframes to one or two per slide presentation.

4. Application for keyframes

To facilitate access to a large collection of digitized videos of meetings and other events, we implemented a web-based video directory browser (described in more detail in [3]) that presents directory listings of videos (see figure 5). Videos are organized by content in directories (e.g., staff meetings, seminar presentations, conference reports) and sorted by date within each directory. Clicking on a video opens a viewer to play it. The use of a standard web browser and the MPEG file format enables casual access to the video archive for almost all potential users without the need for additional software or plug-ins. For users outside our local network, we provide a streaming version of the videos in the RealVideo format. To ease the access to the videos, we provide keyframes.

We enhance each video directory listing with representative frames to help recognize the desired video, and to provide access points into the video. Well-chosen keyframes can help video selection and make the listing more visually appealing. Because it can be difficult to distinguish videos based on a single keyframe, we provide a number of keyframes. Initially, a keyframe chosen by the video database administrator is displayed in the keyframe window (see figure 6). The positions of the keyframes are marked by blue triangles along a mouse-sensitive time scale adjacent to the keyframe. As the mouse moves over the timeline (shown as a hand cursor in figure 6), a slider thumb shows the position on the timeline, the keyframe closest to the mouse position is displayed, and the triangle for that keyframe turns red. This method shows only a single keyframe at a time, preserving screen space while making other frames accessible through simple mouse motion. This interface supports very quick skimming that provides a good impression of the content of the video. Clicking anywhere

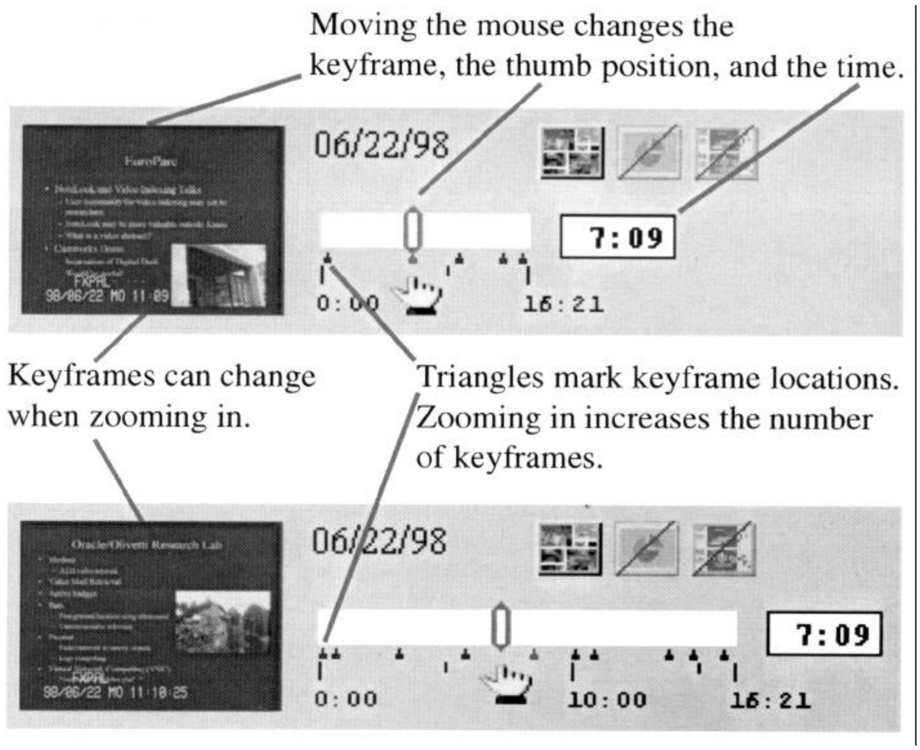

Moving the mouse changes the keyframe, the thumb position, and the time.
06/22/98
7:09
0:00
16:21
Keyframes can change when zooming in.
Triangles mark keyframe locations. Zooming in increases the number of keyframes.
06/22/98
7:09
0:00
10:00
16:21

Figure 6. Time slider for displaying keyframes.

on the time scale opens the video and starts playback at the corresponding time. Clicking on the keyframe also starts playback exactly at that keyframe. Using multiple keyframes in this way gives users an idea of the context and temporal structure of a video.

The time scale is also used to present the results of our statistical models for detecting features such as Powerpoint slides. A grayscale confidence score is displayed along the time scale in which dark areas represent high confidence. The combination of keyframes and the confidence score provides a good overview of a video and allows for quick navigation.

The number of keyframes can be modified by zooming in or out. Zooming modifies the width of the time scale on the screen. The average spacing between keyframes on the screen remains the same so that increasing the width of the time scale also increases the number of attached keyframes (see figure 6).

5. Conclusions

We described a technique for selecting keyframes based on image similarity that can produce a variable number of keyframes that meet various temporal constraints. We use a hierarchical clustering approach that can determine exactly as many clusters as requested keyframes. Temporal constraints determine which representative frame from each cluster is chosen as a keyframe. The detection of features such as slide images and close-ups of people is used to modify the clustering of frames to emphasize keyframes with desirable features. This approach has been applied to a large number of videos producing keyframes similar to the ones chosen by hand.

An application using the keyframe extraction mechanism has been used by the employees of our company to access a collection of video-taped staff meetings and presentations. The keyframe skimming interface greatly simplifies the task of finding the appropriate video and getting an overview of it. To further simplify navigation in a video, we plan to incorporate the keyframe display in a video player. We also use our keyframe extraction approach to create visual summaries of videos in different layouts.

References

1. J. Boreczky and L. Rowe, “Comparison of video shot boundary detection techniques,” in Storage and Retrieval for Still Image and Video Databases IV, Proc. SPIE 2670, San Jose, CA, 1996, pp. 170-179.
2. A.M. Ferman and A.M. Tekalp, “Multiscale content extraction and representation for video indexing,” in Multimedia Storage and Archiving Systems II, Proc. SPIE 3229, Dallas, TX, 1997, pp. 23-31.
3. A. Girgensohn, J. Boreczky, L. Wilcox, and J. Foote, “Facilitating video access by visualizing automatic analysis,” in Human-Computer Interaction INTERACT ’99, 1999, IOS Press, pp. 205-212.
4. A. Girgensohn and J. Foote, “Video classification using transform coefficients,” in Proceedings of the Intema-tional Conference on Acoustics, Speech, and Signal Processing (Phoenix, AZ), 1999, Vol. 6, pp. 3045-3048.
5. B. Günsel, Y. Fu, and A.M. Tekalp, “Hierarchical temporal video segmentation and content characterization,” in Multimedia Storage and Archiving Systems II, Proc. SPIE 3229, Dallas, TX, 1997, pp. 46-56.
6. E. Rasmussen, “Clustering algorithms”, in Information Retrieval: Data Structures and Algorithms, W.B. Frakes and R. Baeza-Yates (Eds.), Prentice Hall, 1992, pp. 419-442.
7. X. Sun, MS. Kankanhalli, Y. Zhu, and J. Wu, “Content-based representative frame extraction for digital video,” 1998 IEEE Conference of Multimedia Computing and Systems, Austin TX, 1998, pp. 190-193.
8. Y. Taniguchi, A. Akutsu, and Y. Tonomura, “PanoramaExcerpts: Extracting and packing panoramas for video browsing,” in Proc. ACM Multimedia 97, Seattle, WA, 1997, pp. 427-436.
9. Y. Tonomura, A. Akutsu, K. Otsuji, and T. Sadakata, “VideoMAP and VideoSpaceIcon: Tools for anatomizing video content,” in INTERCHI ’93 Conference Proceedings, 1993, ACM Press, pp. 131-141.
10. H. Ueda, T. Miyatake, and S. Yoshizawa, “An interactive natural-motion-picture dedicated multimedia authoring system,” in CHI ’91 Conference Proceedings, 1991, ACM Press, pp. 343-350.
11. M.M. Yeung, B.L. Yeo, W. Wolf, and B. Liu, “Video browsing using clustering and scene transitions on compressed sequences,” in SPIE Vol. 2417 Multimedia Computing and Networking 1995, 1995, pp. 399-413.
12. H.J. Zhang, C.Y. Low, S.W. Smoliar, and J.H. Wu, “Video parsing, retrieval and browsing: An integrated and content-based solution,” ACM Multimedia 95, San Francisco, CA, 1995, pp. 15-24.

Andreas Girgensohn is a Senior Research Scientist at FX Palo Alto Laboratory. He received his Ph.D. in Computer Science from the University of Colorado, Boulder, in 1992, and his M.S. in Computer Science from the

University of Stuttgart, Germany, in 1987. In the last five years, his work has focussed on using the Web to design and develop applications to support collaboration, corporate memory, and video access. At FX Palo Alto Laboratory, he is working on Web-based video indexing and summarization. Prior to that, he worked at NYNEX Science & Technology (now Bell Atlantic) on task-oriented user interface design and development, on support for software developers, and on tools for improving communication and collaboration using the World Wide Web and Lotus Notes. Dr. Girgensohn has delivered papers and tutorials on collaborative applications and innovative user interaction techniques at a number of conferences.

John Boreczky is a Research Scientist at FX Palo Alto Laboratory. He received his M.S. in Computer Science from the University of Michigan in 1989 and is pursuing a Ph.D. from the University of California Berkeley. At FX Palo Alto Laboratory he is working on video indexing and retrieval and multimedia user interfaces. Prior to that, he conducted research on database support for video, user interface toolkits, and automotive control and display design. He has delivered papers on multimedia signal processing, video analysis, and media databases at a number of conferences.