Human-Computer Interaction -- INTERACT'03
M. Rauterberg et al. (Eds.)
Published by IOS Press, (c) IFIP, 2003, pp. 769-772

A Hierarchical Keyframe User Interface for Browsing Video over the Internet

Maël Guillemot, Pierre Wellner, Daniel Gatica-Pérez & Jean-Marc Odobez

IDIAP, Rue du Simplon 4, Martigny, Switzerland

{guillemo, wellner, gatica, odobez}@idiap.ch

Abstract*:* We present an interactive content-based video browser allowing fast, non linear and hierarchical navigation of video over the Internet through multiple levels of key-frames that provide a visual summary of video content. Our method is based on an XML framework, dynamically generated parameterized XSL style sheets, and SMIL. The architecture is designed to incorporate additional recognized features (*e.g.* from audio) in future versions. The last part of this paper describes a user study which indicates that this browsing interface is more comfortable to use and approximately three times faster for locating remembered still images within videos compared to the simple VCR controls built into RealPlayer.

Keywords: *video structuring, information visualization, XML, dynamic XSL, SMIL, VTOC.*

1 Internet Video Browsing

Browsing, retrieving, and manipulating digital video are increasingly important tasks within multiple domains including professional, entertainment, consumer applications, and in digital libraries. Interactive video browsers that make use of automatic video analysis allow for such capabilities. A current paradigm is to structure a video into a *video table of contents* (*VTOC*) composed of scenes, shots and key-frames. Several tools have been developed under this framework (Girgensohn et al, 2001), (Tseng et al, 2002), (Divakaran et al, 2002).

Obviously, multiple hierarchies can be defined for any given video, so flexible browsers that allow for extensions to deal with multiple presentations are desirable.

The capability to browse video over the Internet represents an additional advantage. While many available tools are based on a desktop paradigm, recent work has looked at the combination of content-analysis techniques and web-based solutions for video retrieval. A web-based video retrieval engine was described in (Ng et al, 2002) in which the proposed *VTOC* is static and does not change in response to user interaction. (Lee, 2001) focused on the design and evaluation of user interfaces for multimedia IR platform-dependent systems (i.e., no XML-based solutions).

In this paper we present the implementation of an interactive content-based system to browse video over the Internet. Section 2 describes the algorithms for video structuring including shot boundary detection, key-frames selection, and scene boundary extraction. Section 3 presents a detailed description of the XML-based user interface. A user behavior experiment is analyzed in last section.

2 Video Segmentation

Effective video browsing depends on appropriate video representations and the availability of automatic analysis tools for generating these representations. The goal of video segmentation is to divide the video stream into a set of meaningful segments (*i.e. shots*) that are used as basic elements for content-based video analysis. Furthermore multiple temporally adjacent shots can be organized into groups (*i.e. scenes)* that convey semantic meaning. Shots and scenes are extracted as follows.

Detecting video shot boundaries has been the subject of substantial research (Hanjalic, 2002) over the last decade, and is now a mature subject. Our current implementation employs color histograms because they can be calculated from video frames efficiently. In addition, we have developed a robust

key-frame selection technique based on sub-shot boundary detection. The key-frame selection process is invoked each time a new shot is identified.

In (Odobez et al., 2003) we proposed a method to cluster shots into scenes. The method is based on spectral clustering techniques and has been shown to be effective in capturing perceptual organization of videos based on general image appearance.

Results of these methods are stored in an XML (XML spec, 2000) format for further access and browsing. The basic XML elements of the video tree structure are based on MPEG-7 and defined in Table 1.

<video>	root level component of video
<scene>	video scene component
<shot>	video shot component
<subshot>	video subshot component
<keyframe>	timestamp value of keyframe

Table 1: Set of XML elements defined

3 User Interface implementation

The structuring process (bottom of Figure 1) reads a video uploaded onto the media file server and produces an XML file describing the video content as detailed in the previous section.

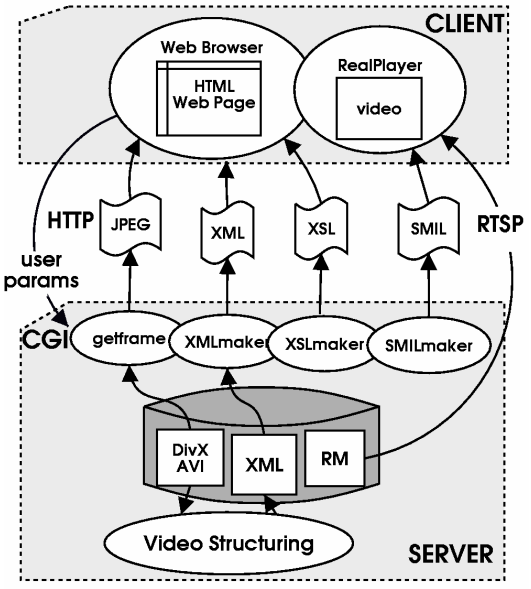

Figure 1: XML-based system architecture

The XML file is transformed into an HTML user interface under the control of a dynamically generated XSL style sheet (XSL spec. 2001). The transformation into HTML can run on the server or in the web browser (*e.g.* Mozilla or Internet Explorer's XSLT processor). The XSL style sheet is dynamically generated by a Perl/CGI script (*XSLmaker*) based on user input parameters which specify the video name and level at which to expand or collapse the hierarchy. Dynamic generation of XSL style sheets under user control is our approach to flexible presentation of multiple video browsing interfaces based on the same underlying XML data, and this technique will also be used in the future for presentation of additional segmentations (e.g. speech segments).

A sample of the video browser user interface is shown in Figure 2 where all levels of the hierarchy are expanded to show the full VTOC. A header displays the current video name, length, number of scenes and number of shots.

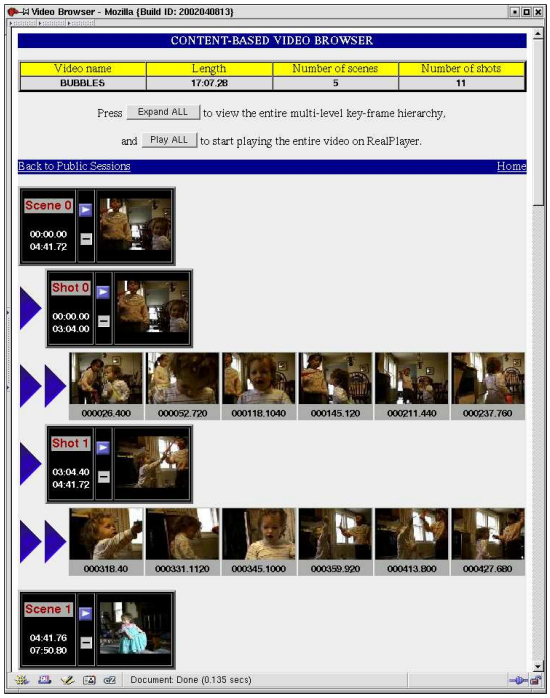

Figure 2: Screen showing VTOC

The video components are ordered along scenes, shots and a number of shot key-frames. Key-frames are dynamically extracted from the compressed DivX video files on a media file server by means of a *getframe* CGI program that takes as input parameters the video source and a timestamp both extracted from the XML file. The extracted key-frames are saved by getframe in cache so they don't have to be extracted from the compressed video file each time a frame is displayed.

The graphical user interface is browser independent (tested on Internet Explorer, Netscape, and versions of Mozilla on Unix, Linux, and Windows). Starting and ending timestamps of scenes and shots display segment duration and each key-frame is also time stamped in the GUI.

Each time a play button is pushed on the HTML page, a RealPlayer window pops up with basic VCR controls (see Figure 3). As illustrated in Figure 1, A CGI script (*SMILmaker*) dynamically generates a SMIL file (SMIL spec 2001). This program takes as parameters the video source location (compressed in DivX on the media file server), as well as the starting and ending timestamps. RealPlayer permits basic VCR capabilities to play and navigate within the selected segment (entire video or a specific scene or shot).

Figure 3: RealPlayer VCR controls

SMIL benefits from the XML plain-text property and the SMILmaker CGI program dynamically configures the most appropriate media object for streaming, depending on client display capabilities (e.g. version of RealPlayer) and connection speed.

4 User study

We conducted a small user study to observe people's behavior while using the video browser. We selected two video segments from the *Kodak Home Video Database*, and selected a single frame from each of these videos at random as the *target image*. We ensured that these target images did not include any of the automatically selected key frames that might appear in the browser interface.

Figure 4: Examples of target images

We asked each of seven subjects to locate a target image (*e.g.* Figure 4) from each video as quickly as possible after allowing them to study the target image for 10 seconds on a printed page before starting each search. Our participants had computer systems background but not particularly in video analysis. We did not allow subjects to refer to the target image during their search because we wanted the task to approximate the process of searching through videos from memory. This task attempts to simulate the situation where a person searches within an Internet-shared home video recording for a particular instant kept in mind – remembered not from having seen the video before, but from having been present at the original event.

While observing subjects interact with the system, we asked them to talk about what they were doing as they worked, and avoided giving them any assistance. We timed how long it took them to complete the task, and took note of difficulties and comments they had as they worked. With one of the example videos, they simply used RealPlayer to browse the video, and with the other, they used our video browser in combination with RealPlayer. At the end of their sessions, participants were asked to express their satisfaction with regard to the use of the full browser and the video player.

Each user performed the task only once for each example video with one interface or the other, so each measurement was performed with users that were completely unfamiliar with the material.

5 User study results

The performances of image search within a video using RealPlayer compared to the full video browser are analyzed based on a one hour video. The seek time for RealPlayer varied from *7 min. 20s* up to *11 min.* (with two failures). The mean time was *9 min. 30s*. The time for retrieving the randomly selected *target frame* with help of our video browser in combination with RealPlayer varied from *1 min. 10s* up to *5 min. 30s*, with an average seek time of *3 min* (no failure were reported). In other words, the browser is approximately three times faster for locating remembered still images within videos compared to the simple VCR controls built into RealPlayer.

We noticed that subjects did not need help, such as a demo before starting to use the tool, so the proposed interface is intuitive in terms of user controls. We reported the following comment regarding time latency while a user was performing the task: *"I feel I don't wait for a long time the images to be loaded. It is even faster than the loading time of a web page full of still images stored in local disks."* Effectively, the getframe CGI function (see Figure 1) sends cached jpeg images on the fly to the web browser permitting fast browsing.

When observing subjects navigating through the browser levels, they usually started expanding the first scene so as to view the corresponding shot key-frames even if the first scene key-frame was far in similarity from the target image. The scene level for this searching task in home videos did not appear as useful as expected. This is particularly due to the unstructured content of home videos. However, most of the subjects agree that it makes sense having such a higher level abstraction than shot for getting a first global view of a video.

Participants' comments about the hierarchical key-frame user interface support these conclusions:

"When using RealPlayer only, I do a kind of segmentation clicking at regular intervals on the time bar, but it is a complete blind random access. Indeed, having key-frames from corresponding shots gives a better view of the video-content."

"The use of the video browser is much more comfortable to use than only RealPlayer for retrieving a picture or a special action because it allows hierarchical access."

The following suggestions for further enhancements were reported:

"It would be nice that clicking on one key-frame makes open a new window with a full size image. Also, the graphical design could be improved."

"From a scene, having 2 buttons would help to either expand first level or all derived levels from that specific scene."

"What about retrieving a particular speech segment or a person, based on an a priori knowledge of her/his speech signal?"

5 Conclusions and future work

The XML-based video browser allows fast browsing of video recordings through multi-level key-frames. This hierarchical key-frame user interface is web-based, interactive and platform independent. The whole framework has been designed with scalability in mind to permit straightforward expansion of the video and audio capabilities. Future enhancements include the integration of higher level video processes such as event and text recognition as well as the inclusion of other segmentation hierarchies derived from audio.

Acknowledgements

The authors thank the Eastman Kodak Company for providing the Home Video Database. This work was carried out in the framework of the National Centre of Competence in Research (NCCR) on Interactive Multimodal Information Management (IM2), and the work was also funded by the European project M4: MultiModal Meeting Manager, through the Swiss Federal Office for Education and Science (OFES). We are particularly grateful to Sébastien Marcel and Frank Formaz for implementing key components of the system.

References

Divakaran, A., Radhakrishnan, R. & Peker, K.A. (2002), Motion Activity Based Extraction of Key Frames from Video Shots", IEEE International Conference on Image Processing (ICIP), NY, USA, Sept. 2002.

Girgensohn A., Boreczky J & Wilcox L. (2001) Keyframe-Based User Interfaces for Digital Video, IEEE Computer Society, pp. 61-67, Sept. 2001.

Hanjalic, A. (2002), Shot-boundary detection: Unraveled and resolved? In IEEE Proceedings Trans. on Circuits and Systems for Video Technology, pp 90-105, vol. 12, Feb. 2002.

Lee H. (2001) UI Design for Keyframe-Based Content Browsing of Digital Video, PhD thesis, School of Computer Applications, Dublin City University.

Ng C.W. & Lyu MR., (2002), ADVISE: Advanced Digital Video Information Segmentation Engine, The 11th International World Wide Web Conference (WWW2002), Hawaï, USA, May 2002.

Odobez, J-M., Gatica-Pérez, D. & Guillemot, M. (2003), Spectral Structuring of Home Videos, Int. Conf. in Image & Video Retrieval (CIVR), IL, US, July 2003.

Tseng, B.L., Lin C-Y. & John R. Smith, (2002) "Video Summarization and Personalization for Pervasive Mobile Devices," SPIE Electronic Imaging- Storage & Retrieval for Media Databases, San Jose, US, Jan. 2002.

SMIL, W3C Recommendation, Synchronized Multimedia Integration Language (SMIL), specification, 2001

XML, W3C Recommendation, Extensible Markup Lanuguage (XML), specification, 2000.

XSL, W3C Recommendation, Extensible Stylesheet Language (XSL), specification, 2001.