Dartmouth College

Dartmouth Digital Commons

Computer Science Technical Reports   Computer Science

2-1-1997

An Efficient Scheme for a Distributed Video Retrieval System for Remote Users

Fillia Makedon
Dartmouth College

James Matthews
Dartmouth College

Charles Owen
Dartmouth College

Samuel Rebelsky
Dartmouth College

Follow this and additional works at: https://digitalcommons.dartmouth.edu/cs_tr

Part of the Computer Sciences Commons

An Efficient Scheme for a Distributed Video Retrieval System for Remote Users

Fillia Makedon
James Matthews
Charles Owen
Samuel Rebelsky

The Dartmouth Experimental Visualization Laboratory
Department of Computer Science
Dartmouth College
Hanover, NH 03755
Tel: 603-646-3048
Fax: 603-646-1672

Introduction

The new era of digital video and multimedia technologies has created the potential for large libraries of digital video. With this new technology come the challenges of creating usable means by which such large and diverse depositories of digital information (digital libraries) can be efficiently queried and accessed so that (a) the response is fast, (b) the communication over the Internet is minimal and (c) the retrieval is characterized by high precision and recall.

In this paper we discuss how existing digital video editing tools, together with data compression techniques, can be combined to create a fast, accurate and cost effective video retrieval system for remote users. The traditional approaches employed in text databases, such as keyword searching and volume browsing, are inadequate mechanisms for a video retrieval system for remote users because, (a) they don't apply to video at all, or (b) they are not practical due to the amounts of data involved, or (c) they have insufficient resolution to be useful in a video archive. New techniques must be developed that facilitate the query and selection of digital video. This paper presents one such scheme.

Let us first consider the primary problem, i.e., the fact that video data is very large relative to textual data. Searches of the entire database can take hours and are, therefore, impractical. Simple queries based on title and textual annotation information can yield gigabytes of data when a simple 30 second edited collection is all that is desired. If the user must view all of the data residing in a given video database, then an enormous amount of resources must be consumed. Furthermore, accessing these data incurs a cost in server and communications resources which can rapidly consume almost any budget.

The data size issue is complicated further by the current nature of the Internet. The Internet is an ideal delivery mechanism for users of a large scale video archival and retrieval mechanism. It is a large, distributed network with reasonably high capacity which can boast connectivity to a wide spectrum of users. Most research facilities already have Internet connectivity and the user base is constantly growing. Indeed, the Internet is a rapidly expanding resource. New development should exploit both the current wide scale and the planned growth of this network. However, digital video is such a large data object that every effort must be made to limit the amount of full resolution video transmitted over the network in order to avoid its eventual degradation for communicating other types of data.   This major contribution of this paper is the integration of enabling technologies and a model for preventing the degradation of Internet performance due to video communication.

A Distributed Approach and The Value of Central Video Databases

Digital video presents a stark contrast to conventional textual sources. The characters generated in textual sources are typically produced one at a time by an author. Digital video, on the other hand, is produced in bulk by a sampling process. Entire libraries of text information can be stored in the space required for only a few hours of digital video. Hence, it is not practical for local facilities to acquire and archive video libraries. The typical user may be able to store only a few minutes of practical video. To be cost effective, the storage should be amortized over a community of users. For the data to be delivered a network is required.

A distributed approach to video databases is attractive since it is scalable. As an example, the Internet is a system that scales to a large number of users. Video databases attached to the Internet can reach large communities of users. It is important to devise of cost-effective and efficient models for large scale network accessible video repositories. In such a system it is possible to limit the local facility requirements.

However, novice users need a basic tool that allows for easy selection and retrieval of this video data in a cost and time efficient manner. This paper presents a system that balances both the needs of the novice user and the network and server resource requirements.

Applications of a Video Retrieval System

In the paper we will consider some of the numerous applications for large-scale databases of digital video, starting with education. An easily accessed video library allows an educator to create exciting and relevant presentations for the classroom. Such presentations might show important historic events, illustrate recent scientific achievements, or show a class the process of dismantling an engine block. Since the teacher can create exactly the desired presentation these showings can be made to actively relate to the material. This is a stark contrast to the traditional, canned, video presentation.

Of course, the research community could also benefit from central video resources. Video data such as satellite imagery, historic footage, biological experiments, and numerous other examples could be searched and accessed, allowing the researcher to access video data as easily as written publications.

Some other areas which could easily benefit from a central store of digital video are the management of equipment documentation, remote sensing and data collection, scientific data analysis, and the field of broadcasting. Digital video is a rich information medium. Applications await the ability to efficiently access and deliver this medium.

Basic Scheme for Remote Video Querying

In this section, we will consider existing approaches for video querying and then introduce our basic scheme which is based on the use of decimated video communication. Video query technology is very new and is closely related to the problem of querying image databases. It suffers from the inevitable problem of too much data volume and from the fact that motion video does not have a simple structure to search as text does. Let us consider (a) the naive approach of video query, (b) the image processing approach and (c) what we call the *representational query* approach.

The naive approach involves simple text searches of information associated with the video. Typically, every piece of video will have an associated title which can yield some information. The problem with this approach is that the granularity of the search is very large, being the entire duration of the video. With as much as a gigabyte of storage and transmission required for an hour of video, this is obviously not a reasonable level of granularity. The granularity can be improved by textual annotations, but this is a labor intensive and manual process which assumes the video is viewed by the annotator. It is conceivable that many databases may have either too large a volume to view or video which cannot easily be annotated since the very annotation is a research project in itself.

Image processing approaches to video queries attempt to analyze video in order to satisfy a query. A query may be content items or a manual drawing which is to be matched. Techniques such as *cut detection, auto correlation, feature extraction, spectral analysis, and motion analysis* are all valid components in the image processing tool kit and can be used to segment or select portions of video in response to a query.

The *representational video approach* represents the image using approaches such as generated image descriptions or combinatorial hashing on extracted feature information.

These techniques are all limited by the fact that an ideal video query is a still-unsolved machine vision problem. For a query to be perfect, it must not only decide what it is the viewer desires, but also retrieve as accurately as possible the relevant video segments with minimal Internet traffic. Ideally, these video segments should then be provided in a final edited copy. It is unlikely that such precise query technology will exist for some time to come.

Manual Query Enhancement

Since query technology is a long way from perfection, we enhance the query process by bringing the user into the system. Figure 1 illustrates the distributed video retrieval system scheme. This illustration shows an iterative process of evaluating and querying. It is assumed that the query can only be crudely answered by the system. At this point the user must become involved. We refer to this process as *manual query enhancement*. In order for the user to be able to make decisions about visual material she cannot see, the selected video must be sent to her for final selection and editing. It is at this point that we differ from the traditional, simple approach of simply sending the video.

If all query results are sent to the user and the user must then edit, either the network will be swamped with high volume traffic, most of which represents information that will be discarded, or the user must wait for very long times for the information to be transmitted. In either case, vast amounts of network resources are wasted. The solution to this problem is *representational video.*

Representational video is video that has been decimated in time and resolution. You might say that it has been much more highly compressed. While the original image data may represent full screen images with 30 frames per second, the representational video may only represent postage-stamp size images at a frame per second. The user can then quickly discard unneeded query responses and edit the desired video into a "rough cut" using a tool we have developed called VideoScheme. VideoScheme is an interactive video editing package developed at Dartmouth and is described in more detail, later.

A Network Architecture Using Representational Video

Figure 1 illustrates how users edit and retrieve images from the videobase. To summarize, a user makes a query and the information retrieval system determines a collection of video clips that match that query. A lower-quality summary (decimated version) of the videos is presented to the user by VideoScheme. The user, based on the contents of the summary view, selects appropriate clips which are then presented in higher quality. The use of multiple video qualities and its benefits for network communication are discussed in detail in the final version of this paper.

Automation of Video Delivery: Erring on the side of "Too Much"

In the final version of the paper we will discuss delivery criteria which, if applied, can automate the query process so that the appropriate amount of video to be delivered in chosen. This selection process places some unusual demands on the information retrieval system. Video not selected by the query process is lost. A user can reject video that is not appropriate, but they cannot make a positive decision about video they cannot see. Hence, the query process must err on the side of delivering too much video rather than forcing too strict a delivery criteria which would result in desirable selections being lost.

In any video query scheme, it is also important to adopt approaches which are not dependent upon the query technology used so that the video retrieval system can easily incorporate new and improved techniques. In the full paper we show how this can be accomplished.

Compression Decimation

There are several approaches to producing representational video in response to a query. The simplest is *simultaneous compression*. When the video is first compressed, a parallel process performs compression to a higher degree in order to create the representational video. This approach has the disadvantage that compression becomes computationally more complex and the video is stored redundantly in two formats, an inefficient use of storage. Also, if multiple decimated resolutions are to be supported, allowing the resolution of the representational video to vary, several versions would have to be stored.
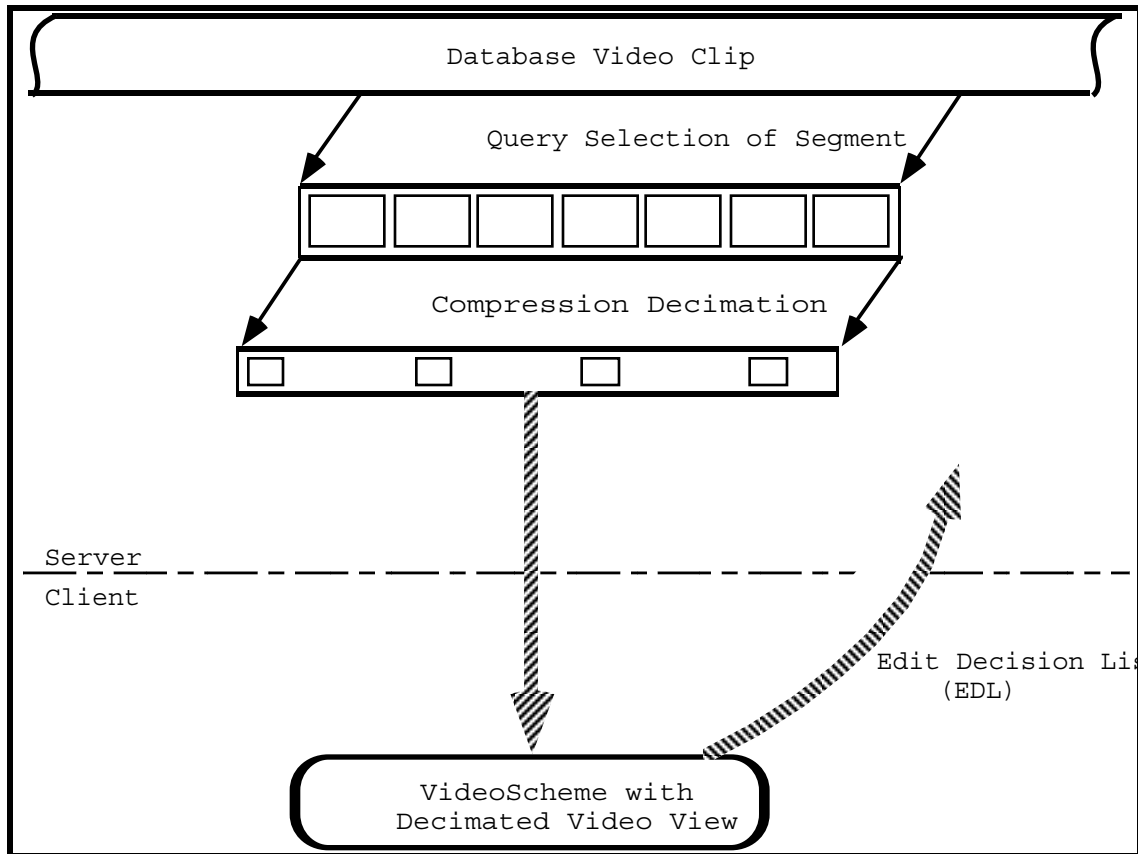
Figure 1: The Retrieval Model

Another simple approach is *recompression.* When the representational video is required, the server decompresses the image and then recompresses it at a decreased resolution. The problem with this approach is the huge volume of intermediate data that is generated. Also, this approach stacks two uncorellated compression processes, which can degrade the picture more than the direct compression of the original video data.

Compression Decimation is the post processing of a compressed data set to a lower resolution (spatial or temporal) destination. It has the advantage that information, such as motion estimation and information frame insertion points, from the original compression process can be exploited to improve the resolution of the target resolution video data. Also, no large, uncompressed, intermediate files are produced. Compression decimation is currently under development at Dartmouth College.

Using Representational Video

Representational video is obviously not the ideal user interface. It is a compromise between the user interface and the network resource requirements. The user sees the actual query result, albeit decimated in time and resolution. The network is not required to transmit the entire query result volume and the transmission requires considerably less time. As an simple example, suppose the query only produced twice as much video as is required, an optimistic assumption at best. Using a 10:1 compression decimation basis, the total network traffic is reduced by 40%. The query overhead on the final video selection is only 20% as opposed to 100% for the naive approach. We show in the paper the variation of decimation factors in order to effectively compromise between edit resolution and bandwidth consumption.

Use of VideoScheme: A Digital Video Editing System

VideoScheme is a prototype programmable digital video editing system, developed at Dartmouth College in order to investigate programming approaches to video. It is implemented as an application for the Apple Macintosh. It

provides a visual browser for viewing and listening to digital movies, using Apple's QuickTime system software for movie storage and decompression. The browser displays video and audio tracks in a time-line fashion, at various levels of temporal detail. Clicking on the video tracks displays individual frames in a "flip book" fashion, while clicking on the audio track plays it back; clicking twice in rapid succession plays back both audio and video.

As a visual interface to digital media VideoScheme is nothing out of the ordinary; what separates it from conventional computer-based video editors is its programming environment. VideoScheme includes an interpreter for the LISP-dialect Scheme, built on the SIOD (Scheme-in-one-Defun) implementation, along with text windows for editing and executing Scheme functions. Functions typed into the text windows can be immediately selected and evaluated. The environment, while deficient in debugging facilities, offers such standard LISP/Scheme programming features as garbage collection and a context-sensitive editor (for parentheses matching). In addition it offers a full complement of arithmetic functions for dynamically-sized arrays, an important feature for handling digital video and audio.

Scheme was chosen over other alternatives (such as Tcl, Pascal, and HyperTalk) for a number of reasons. Scheme treats functions as first class objects, so they can be passed as arguments to other functions. This makes it easier to compose new functions out of existing ones, and adds greatly to the expressive power of the language. Scheme is also easily interpreted, a benefit for rapid prototyping. Scheme includes vector data types, which map very naturally to the basic data types of digital multimedia, namely pixel maps and audio samples. Finally, Scheme is easily implemented in a small amount of portable code, an advantage for research use.

VideoScheme extends the Scheme language to accommodate digital media. In addition to the standard number, string, list, and array data types VideoScheme supports the following objects:

movie	--	a stored digital movie, with one or more tracks.
track	--	a time-ordered sequence of digital audio, video, or other media.
monitor	--	a digital video source, such as a camera, TV tuner, or videotape player.
image	--	an array of pixel values, either 24-bit RGB or 8-bit gray level values.
sample	--	an array of 8-bit Pulse Code Modulation audio data.

These objects are manipulated by new built-in functions. Movies can be created, opened, edited, and recorded:

```
(new-movie)                 ; creates and returns a reference to a new movie

(open-movie filename)       ; opens a stored movie file

(cut-movie-clip movie time duration)
                                ; moves a movie segment to the system clipboard
(copy-movie-clip movie time duration)
                                ; copies a movie segment to the system clipboard
(past-movie-clip movie time duration)
                                ; replaces a segment with the clipboard segment
(delete-track movie trackno)
                                ; removes a movie track
(copy-track movie trackno target)
                                ; copies a movie track to another movie
(record-segment monitor filename duration)
                                ; records a segment of live video from the monitor
```

Images and sound samples can be extracted from movie tracks or monitors, and manipulated with standard array functions:

```
(get-video-frame movie trackno time image)
                                        ; extracts a frame from a video track
(get-monitor-image monitor image)
                                        ; copies the current frame from a video source
(get-audio-samples movie trackno time duration samples)
```

With this small set of primitive objects, and small number of built-in functions, we can rapidly build a wide variety of useful functions with applications in research, authoring and education. For example, VideoScheme functions can scan video for scene breaks using cut-detection heuristics.

We believe that VideoScheme's melding of direct manipulation with programmability is a promising approach for manipulating digital video. It is even more promising in a distributed environment, where the user is separated from the video data by a limited-capacity network. The VideoScheme system can naturally be extended to support remote execution of VideoScheme programs. For example, a cut-detection function can be sent to a remote video database, where it can be efficiently operate on centrally stored video. The results of such an operation can be returned to the user in the form of decimated video, which represents the full-fidelity data but requires much less bandwidth. Figure 2 represents the conventional, local processor implementation of VideoScheme. The distributed client-server extension of the VideoScheme system is illustrated in the figure 3.
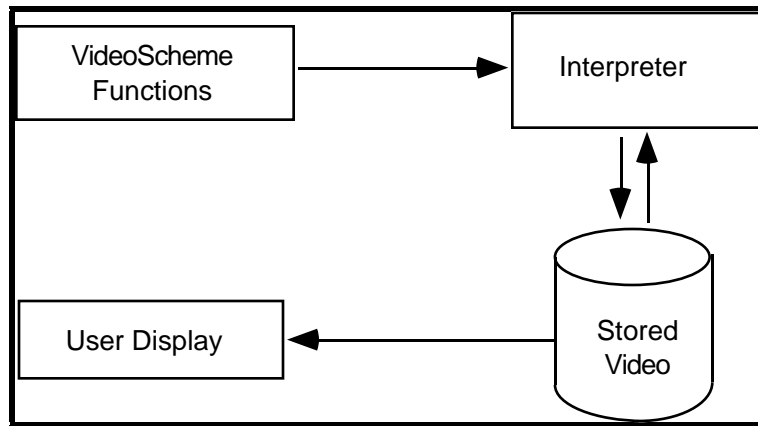
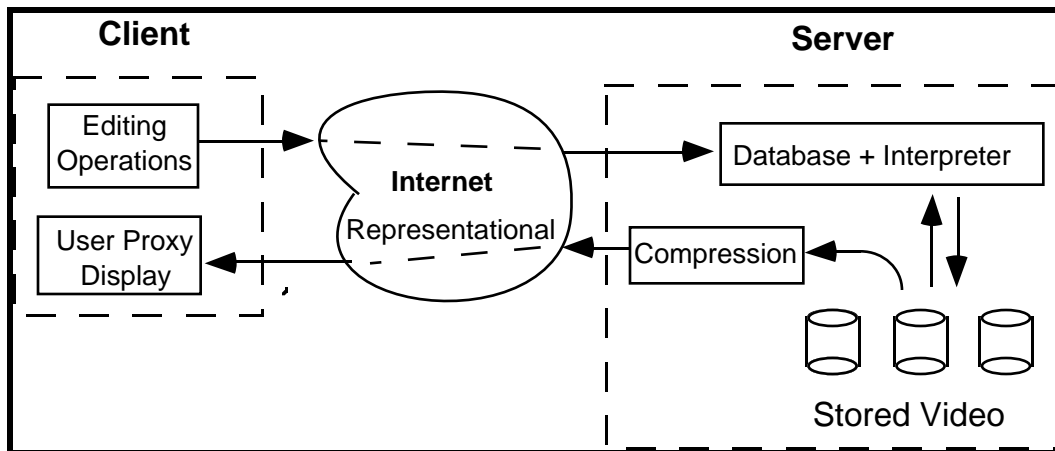

Figure 2: VideoScheme Single Processing Model

Figure 3: VideoScheme Client-Server Model

Future Work and Conclusions

We are currently exploring the implementation issues of using both video query systems and the selection process in a digital video server environment. As we explore this new landscape we are analyzing the components for efficiency and usability. Since one target user base we are working toward is high school teachers, we must ensure the user interface is robust and responsive for this class of user. The questions of how fast a system must respond and how must decimation of the video is acceptable must be studied. Compression decimation is an ongoing

research topic. Since this operation occurs in the server, a shared resource, it is essential that it is fast and efficient. We are looking at means of accelerating this process, including nonsequential decimation, effective user of existing compression information, and caching of decimated video selections.

References

1. Califano, A. and I. Rigoutsos. FLASH: A fast look-up algorithm for string homology. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, New York City, June 1933.

2. Deerwester, S., S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. J. Amer. Soc. Information Science **41**(6), (1990) 391-407.

3. Susan Dumais. Improving the Retrieval of Information from External Sources. Behaviour Research Methods, Instruments and Computers 1991, 23 (2) 229-236.

4. Susan Dumais. Enhancing Performance in Latent Semantic Indexing (LSI) Retrieval. TM-ARH-017527 Technical Report, Bellcore, 1990.

5. Johnson, D. B. and F. Makedon, Building a Digital Library with Extensible Indexing and Interface: A Discussion of Research in Progress, Digital Libraries Workshop, Rutgers Univ., 5/94.

6. Makedon, F., J. Matthews, and P. Gloor, VideoScheme: A Programmable Video Editing System for Automation and Media Recognition, 1993 Proc. of the ACM Multimedia Conf.,Anaheim, CA 8/1-8/6/93.

7. Makedon, F., S. A. Rebelsky, M. Cheyney, C. Owen, and P. Gloor, Issues and Obstacles with Multimedia Authoring, Proc. of EdMedia '94.

8. Makedon, F., P. Metaxas, J. Matthews, P. Gloor, D. B. Johnson, M. Cheyney, Conference on a Disk: An Experiment in Hypermedia Publishing, Proc. of EdMedia '94.

9. Makedon, F. and C. Owen, A Digital Library Proposal for Support of Multimedia in a Campus Educational Environment, EdMedia '94 Poster Presentation.

10. Matthews, J., F. Makedon and P. Gloor, VideoScheme: A Research, Authoring and Teaching Tool for Multimedia, Proceedings of EdMedia 1994.

11. Rigoutsos, I. and R. Hummel. Massively parallel model matching: geometric hashing on the connection machine. IEEE Computer: Special Issue on Parallel Processing for Computer Vision and Image Understanding, February, 1992.

12. Rigoutsos, I. and A. Califano. dFLASH: A distributed fast look-up algorithm for string homology. (manuscript) August, 1993.

13. Salton, G. Automatic information organization and retrieval. McGraw Hill (New York) 1968.

14. G. Salton and M.J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

15. G. Salton. Automatic Text Indexing Using Complex Identifiers. *ACM Conference on Document Processing Systems.* Dec. 5–9, Sant Fe, 1988. 135–145

16. Salton, G. Buckley, C. Automatic Text Structuring and Retrieval - Experiments in Automatic Encyclopedia Searching. TR 91-1196, Cornell U., Ithaca NY, 1991.

17. van Rijsbergen, C.J. A theoretical basis for the use of concurrence data in information retreival. Journal of Documentation **33**, (1977)106-119.

18. van Rijsbergen, C.J. A Non-Classical Logic for Information Retrieval. The Computer Journal, vol. 29, No. 6, 1986.

19.   van Rijsbergen, C.J. Towards and Information Logic. SIGIR '89. pp. 77-86.