11-3-1994

Multimedia authoring, development environments, and digital video editing

Fillia Makedon
Dartmouth College

James W. Matthews
Dartmouth College

Charles B. Owen
Dartmouth College

Samuel A. Rebelsky
Dartmouth College

Multimedia authoring, development environments, and digital video editing

Fillia Makedon, James W. Matthews, Charles B. Owen, Samuel A. Rebelsky

Dartmouth College, Department of Computer Science
Dartmouth Experimental Visualization Laboratory
Hanover, New Hampshire 03755

ABSTRACT

Multimedia systems integrate text, audio, video, graphics, and other media and allow them to be utilized in a combined and interactive manner. Using this exciting and rapidly developing technology, multimedia applications can provide extensive benefits in a variety of arenas, including research, education, medicine, and commerce. While there are many commercial multimedia development packages, the easy and fast creation of a useful, full-featured multimedia document is not yet a straightforward task.

This paper addresses issues in the development of multimedia documents, ranging from user-interface tools that manipulate multimedia documents to multimedia communication technologies such as compression, digital video editing and information retrieval. It outlines the basic steps in the multimedia authoring process and some of the requirements that need to be met by multimedia development environments. It also presents the role of video, an essential component of multimedia systems and the role of programming in digital video editing. A model is described for remote access of distributed video. The paper concludes with a discussion of future research directions and new uses of multimedia documents.

Keywords: multimedia authoring, multimedia environments, digital video, video query, videobase, compression decimation, electronic conference proceedings, information retrieval, video editing.

1. MULTIMEDIA DEVELOPMENT ENVIRONMENTS

Multimedia development environments facilitate and automate the *authoring* (creation) of *multimedia documents.* There is a high diversity of such environments (also referred to as authoring systems), depending on the different applications that drive them. The type of applications (or multimedia documents) span a variety of topics and levels, from simple electronic brochures to sophisticated academic publications. An example of an application or multimedia document is a biology journal that includes the visualization of molecules, facilities to access and search associated research papers, collaborative communications facilities, printing, annotation, or animation.

The authoring of a multimedia document is a complex process.[19,23] It is more than the combination of multimodal elements from diverse media, as discussed

in a later section. A desirable feature of a multimedia document is interactivity. This means that it should be designed so that its users can traverse the document materials in a variety of ways, both in ways pre-scripted by the authors of the document and in ways not predicted. A user should be able to quickly jump to another part of the document or to search for a key concept. Tools must also be available that allow users to easily *annotate* and, thereby, extend (personalize) a document. For example, one might add marginal notes to particular "nodes" in the document, add new hyper-links to the document, and even add new materials and scripts to the document.

In general, the authoring process results in a multimedia document (or application) which can be anything from an electronic book to an interactive course, from an interactive slide presentation to a multimedia newspaper, from an interactive auto manual to a clinical record that joins and links X-rays, MRI's, physician's comments and even recorded interviews with the patient. While there is an increasing need for multimedia office documents, conference proceedings, information kiosks, professional brochures, course materials, virtual reality museum presentations and others yet to be discovered, there is a lack of efficient mechanisms that automate the document creation process.

Figure 1, an example of a multimedia document, presents the interface to a sophisticated interactive multimedia conference proceedings that allows virtual *participants* of a conference to experience the conference in a variety of ways. In addition to reading papers and viewing talks, the virtual participant can follow and create paths of topics through the proceedings, add notes to individual slides and pages, keep a notebook, search through the proceedings for instances of topics, add bookmarks to the proceedings, and much more.[23] While this plethora of features may not be necessary for every multimedia document, it is very important that the document provide the reader many alternative opportunities for interaction.

Section 2 of this paper discusses the issues and steps involved in the authoring of multimedia documents. Section 3 presents a digital video editing system which facilitates the multimedia authoring process as well as the large scale communication process of digital video. Section 4 presents issues of distributed and remote video retrieval. Section 5 concludes with a discussion of future research.

1.1 Support requirements for multimedia development environments

The creation of a multimedia document is a complex process which requires significant support from the multimedia authoring environment to (1) help the developer create the components of the document fast, (2) allow the developer to easily tie these components together, and (3) present the materials to the user in an interactive manner. Surprisingly, few commercial multimedia development systems provide the types of support that author-developers of multimedia documents need.[1] Some of the requirements of multimedia authoring environments are:[17,19,21,23]

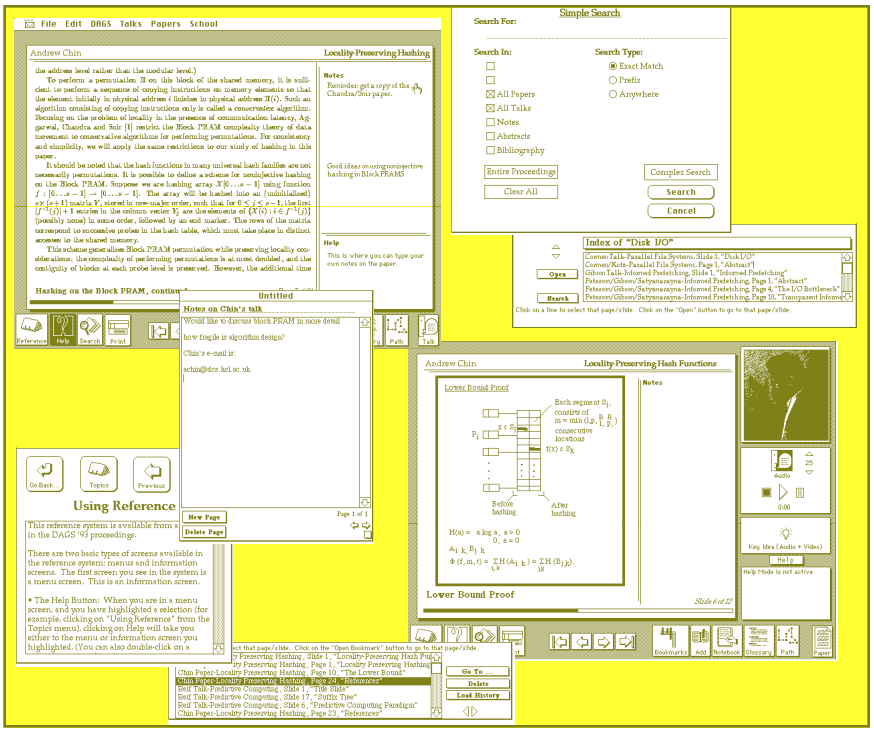

Fig. 1. An interactive multimedia conference proceedings. This multimedia document includes the text of papers, slides from talks, audio and video of speakers, and many features for annotating and interacting with the proceedings.

(a) **Support for multiple computing platforms:** An environment that, supports only Macintosh, only PC/Windows, or only UNIX Workstation/X Window system, severely limits the audience of the documents created in that environment. Unfortunately, the best development environments are initially available on only selected platforms. Porting a multimedia document from one software platform to another is quite expensive, time-consuming, and error prone.

(b) **Support for significant amounts of text:** For multimedia documents to be more than travelogues, videotapes or games, they must include significant amounts of text as well as links between text, audio, video, and graphics. Features should include sophisticated searching, annotations, hyperlinks (both those created by the author and by the user of the document), and the ability to create new "paths of ideas" through the document.

(c) **Provisions for extending and adding features:** Multimedia environments should include provisions for extending the interface and adding features. Scripting languages must be sufficiently powerful to allow the addition of

complex features (e.g., a more complex search or similarity match algorithm). Video manipulation tools should provide access to the pixels and timeline so that users of a document can become new authors and can develop anything from new segmenting algorithms to new lookup algorithms. (In a later section the role of language and basic editing features in a digital video editing system are discussed.)

1.2 Enabling technologies for a large-scale information-retrieval system

Multimedia development environments are meant to become part of a vast database of such documents which are physically distributed, or their components are physically distributed. For example, the video components of a multimedia document (e.g., law patents) may be in a court archive while the notes and diagrams may be in a legal analysis database. The retrieval of information that is composed of multiple media is termed *Multimedia Information Retrieval*.[24,25] In this section, the authoring of multimedia documents is viewed from a user's point of view, a user who is part of a multimedia information retrieval system.[16,18,26,27,28,29,32,33]

In order to effective, multimedia development environments must also operate with the awareness of where and how the documents produced will be used. In other words, the process of multimedia authoring needs to be viewed as one of an array of enabling technologies is needed for the efficient processing of multimedia documents in a large scale multimedia information retrieval system. These technologies include multimedia authoring systems, data compression, network systems, pattern recognition, user interfaces, human computer interaction, information retrieval systems, large storage system technologies, and others.[11,12,13,16,18] In this paper we will discuss a subset of these technologies.

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 (commonly termed *digital libraries*) can be efficiently queried and accessed so that (a) the response is fast, (b) the communications cost is minimized, 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.

Digital libraries[10,16,18] are large scale systems which must include a repository for large quantities of information combined with mechanisms for searching and delivering this information to end users. Access to information stored locally, as well as public and private information available via national networks, must be equitable across entire populations which may have diverse needs and diverse datasets. Datasets may include digital video clips, reference volumes, image data, sound and voice recordings, scientific data, and private information services. A practical system must adapt to changing user, information, and

equipment needs. Multimedia authoring capabilities will become part of digital library interfaces and, as such, must reflect and incorporate these needs.

In view of the above, a digital library system[10] is not merely an expansion of the networks of today, but a vast and powerful repository of diverse types of information that can be accessed at high speed by a large number of users. It is important, therefore, to carefully plan for and design a system and interface that will anticipate the new types of data that will be available in the future, rather than simply networking large numbers of general purpose computers. Issues which should be considered in such a design, include volume information delivery, adaptability, redundancy, storage backup and scalability.

This implies that multimedia development environments should comply with certain critical characteristics of a viable and successful Digital Library for remote access:

a. The system should enable remote users to access basic information without overloading the communications medium (Internet). This implies a communication architecture that tries to minimize traffic.

b. The system should support different types of users: (i) expert researchers and (ii) novice users. The system should also support data selection and editing by non-expert users without extensive training.

c. The system must be scalable, both in terms of the user population size and in terms of the size of the image database: information must be segmented and archived in a way that allows fast image/text/audio search engines to be applied.[24,25] We propose a digital video editing system that allows such a segmentation.

d. The system must be validated over time and with realistic testbeds while the system's performance improvement remains seamless: no capabilities are lost in the process of introducing new capabilities, new users, new types of data, and new access routes.

e. Hand-in-hand with the acquisition of data, equipment installation, and basic research, it is important to involve users early on in the development in order to understand the mental processes and activities that a user goes through to seek information.

1.3 Issues of digital video and video communication

Another important aspect of multimedia development environments is how the overwhelming amount of information produced, in the form of multimedia applications, can be processed or transmitted over a network. The widespread use of computer networking and distributed databases can be traced to a central problem of the information age: there is more digital information than can be reasonably stored and processed in one place, much less in all the places where it

might be useful. This problem is magnified in the domain of multimedia documents, particularly those incorporating digital video. Video data can occupy thousands of times as much storage as text or image files, and so the need for *distributed approaches* is acute. But the size of video data also puts a strain on distributed systems, due to the limited communications bandwidth of data networks.

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.[1] 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 often 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. Mechanisms and software are needed that efficiently segment and store, retrieve and disseminate chunks of video data over the network. More of these issues are discussed in Section 4 of this paper.

2. MULTIMEDIA AUTHORING

The previous section gave an overview of the context within which multimedia development environments should be viewed. In this section we focus on the process of multimedia authoring. We give the steps involved and a closer look of the obstacles that are most common. It should be pointed out that there are several different approaches to multimedia authoring that include object oriented programming.[12,15] However, we believe that for mass production purposes and within the realm of the publishing world, these techniques have not been proven to be practical. What is needed are techniques which can be implemented quickly and by novice programmers.

The advent of computers, sophisticated word processors, and desktop publishing systems have significantly changed the authoring process. Multimedia technology further extends the role of authors to that of *multimedia authors/developers* who are now more than just writers, but also software engineers, graphic designers, human-computer interface engineers, and even editors.

While different multimedia documents can have different requirements for the multimedia author, there is a relatively consistent set of steps that authors should follow as they develop multimedia documents. These steps, summarized in Figure 2, guide the author from initial inspiration to finished document. While the steps may seem linear, there is significant feedback from step to step. For example, the content and media of the materials included in the document significantly affect the features chosen for the interface. Similarly, the processes of editing and annotation may reveal both new materials and new features.

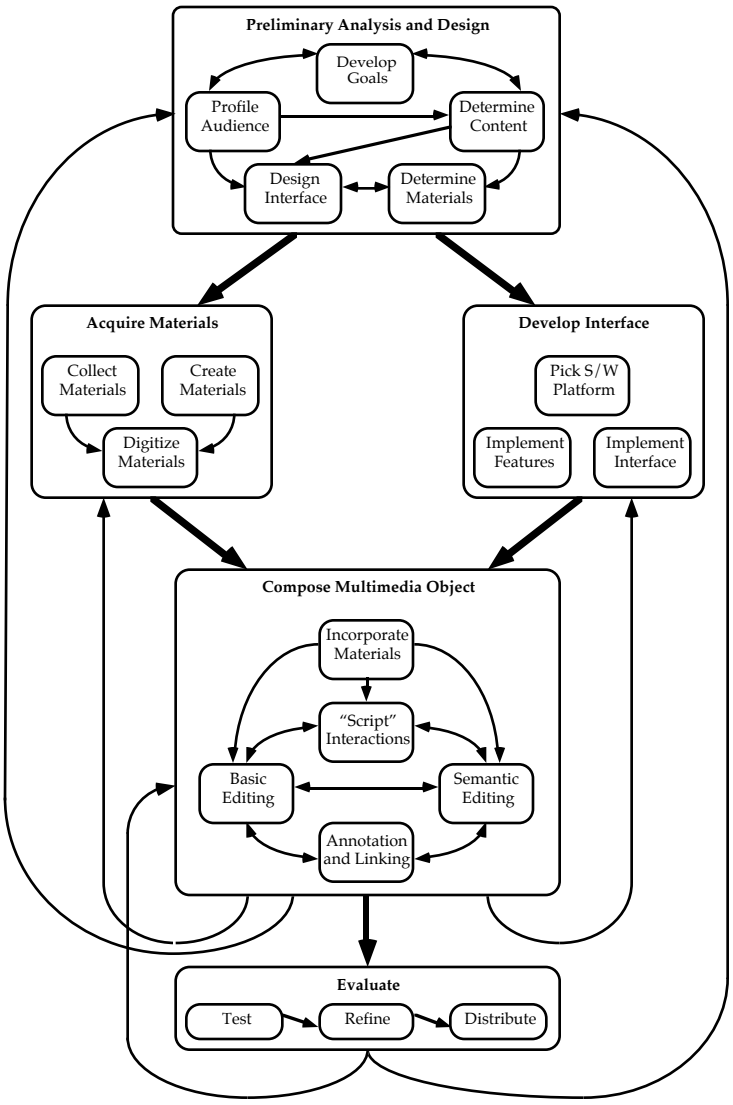

Fig. 2. Steps in the construction of multimedia documents.

As Figure 2 suggests, the authoring process can be broken down into five primary stages:[5,19,23] (1) *preliminary analysis and design*, in which the requirements for the document, its content, and its interface are developed; (2) *interface development*, in which the basic software platform is chosen and the features are implemented and put together to form a coherent interface; (3) *materials acquisition*, in which the materials that will form the document are collected, created, or digitized; (4) *multimedia object composition*, in which the components of the document are synthesized, put together, edited, and extended

with annotations, scripts, and links; and (5) *evaluation and delivery*, in which the multimedia document is tested, refined, and, finally, distributed to its audience.

Portions of this development process can be done in parallel (e.g., the development of the interface and the acquisition of materials). There is significant feedback in the process (e.g., semantic editing may suggest new materials to add). The individual steps are discussed further below.

2.1 Preliminary analysis and design

The design of a multimedia document begins with an analysis of the goals and expectations of the resulting product. This involves the consideration of issues taken from traditional, (text-based) authoring, as well as from computer program design. These issues include: the reasons the author has for creating the document, the audience(s) for the document, what and how materials should be presented, the resources (manpower, materials, finances) available, the special features that empower readers (users) of the document, and the specific content of the document.

Assessing the profile of the potential audience plays a key role in the design and usability of a multimedia document. A document created for novices with too much sophisticated material is as useless as a document created for experts with too much introductory material. Since multimedia documents are both collections of information and software packages, the authors of multimedia documents must evaluate audience expertise and expectations from the point of view of both content and presentation technology (e.g., the experts in a particular field may not be experts in the use of a complex user interface).

The components of the document and the form these components take must also be decided. Because multimedia documents can draw upon a broad range of materials, authors of multimedia must decide what media best inform the audience and when multiple forms of media are necessary for particular segments of the material. Many issues can come into play in this decision process. The costs and benefits of nontraditional media and the ways in which these media will be included must be weighed. For example, transcription of audio in the case of interactive multimedia conference proceedings is costly and time-consuming, but transcriptions also provide for much more sophisticated interaction with the proceedings. The expected time frame for producing the document also affects decisions concerning costs and benefits: a more sophisticated document takes longer to produce, so market windows may require the designers of the multimedia document to eliminate some desirable features.

2.2 Interface development

As a next step, it is important to design and implement as much of the interface as possible, before incorporating materials into the multimedia document. This will facilitate the early determination of the format to record or

produce materials. This will, in turn, permit the timely production of the document, one of the most important cost criteria.

However, creation of the interface may be, and often is, done in parallel with creation, collection, and digitization of materials. The developers and designers of the interface need to consider ways to present materials, features to include and/or exclude (e.g., what types of searching should the interface include), the hardware platforms on which the document will be made available, and the software platform used to implement the interface. It is important in the design of the interface to take into account the protocol, assumptions and sensitivities of the audience.

The intended dissemination mechanism of a multimedia document is another factor that will influence the design of the interface. (For example, in the case of conference proceedings, questions that come into play are: will it be made available on CD-ROM or networked and will it be accompanied by a printed copy of the text?) Eventually, it will be preferable to use a prepackaged multimedia environment. However, as section 3 suggests, authoring packages that support all the features that sophisticated interactive multimedia documents require are not yet available. So, at present, development of the interface is a necessary step in the construction of a multimedia document.

2.3 Materials creation and acquisition

The next step is to collect or create the materials that will make up the document and convert them to a common electronic format. It is usually not necessary to create every component of a multimedia document anew; some materials already exist in created databases, or in the public domain; others may be licensed from outside sources. The determination of which materials need to be collected specifically for the document and which need to be newly authored is a time consuming process. The automation of the collection and integration process of diverse materials (text, photographs, audio clips, video frames, etc.) is a hard problem. For example, in the creation of a multimedia course to teach parallel computation to novices at Dartmouth College, one source of materials is the videotaping of classes on parallel computation, another source is lab exercises and still another source is algorithm animations. When possible, it is preferable to obtain the materials in both electronic and hard copy format. The electronic format eases transition to digital form; the hard copy format provides an accurate master record to use as reference (and, when necessary, a source to be digitized or redigitized).

During the design and development of a multimedia document, it is advisable to obtain additional materials for backup purposes. These additional materials may be incorporated if the design changes, or they may be used to correct materials included in the document. For example, even if a multimedia document includes only the text of a speech, the audio of the speech can be used during development in order to verify that what was said matches what was

printed. Similarly, the video from a speech can be used to annotate the text of the speech.

Once the materials have been collected and created, they are converted to a standard electronic format. Some materials will need to be digitized or, if obtained in electronic format, they may need to be converted to another format or formats. For example, TeX documents (the format employed my many computer scientists and mathematicians) may be converted to PostScript™, HTML (the HyperText Markup Language which is used for networked hypertext documents in the World-Wide-Web), and ASCII.

2.4 Multimedia document composition

Once in digital form, the materials are combined to form the multimedia document. This involves a temporal element of presentation, where the right type of information, in the right type of format, must appear at the right time and the correct place. Such an orchestration of multimedia information, when done manually (rather than via object oriented programming means)[12] is perhaps the most time-consuming step in the development process: a multimedia author must segment materials, edit them (both for format and content), "script" the relationship between the components, annotate the materials, map them onto a timeline and create links between related materials. Comprehensive editing of the audio/video components is very important in the production of useful multimedia documents. Traditionally, two types of editing are done to the materials that comprise multimedia documents: basic, format-based, editing and more sophisticated semantic editing.

Non-experts may do the *basic editing*. This form of editing involves simple "clean up" required by the basic materials, the recording process, or the conversion process. For example, one may remove "um"s and "ah"s, pauses, and verbal "ticks" from the audio. This makes the audio much more pleasant to listen to and significantly reduces the length of audio materials. One may also need to retype or redraw text and figures that were not adequately digitized and reformat some documents to fit the requirements of the computer screen. Some of these simple editing tasks may be performed automatically, but many must still be performed by hand to ensure a quality product.

Semantic editing is performed by experts in the field. It includes tasks such as segmenting the materials into coherent self-contained "chunks," checking the content of these materials, identifying key components of the video and audio tracks, and annotating individual materials. Semantic editing creates a new position in the world of electronic publishing, that of the expert electronic editor. Careful semantic editing can make the difference between a useful, successful publication and a useless, boring one.

It is important to recognize that a wide range of specialized tools are required for editing multimedia document materials. Whereas simple word processors are useful for editing text documents, multimedia components such as audio,

video, and animations have both spatial and temporal components and much larger volumes of material which cannot be simply retyped by hand. Multimedia editors such as digital video and digital audio editors are an important tool in the multimedia authoring process. Most multimedia authoring packages include simple versions of these tools, though they remain somewhat primitive.

The materials may be edited both before and after they are incorporated into the interface. Once in the interface, the materials can undergo further semantic editing. For example, experts can determine links between different parts of the multimedia document; annotate materials (with text and audio); synchronize the text, audio, video, visualization, etc components; add similar semantic links to the materials. Experts may also create new "paths" through the document, so that a reader can follow selected topics through the multimedia document. For example, if the document is an interactive multimedia software package on diseases, a path may be created specifically for infant related diseases, or diseases affected by a certain drug, etc. Experts may also "script" presentations, suggesting what components should be shown simultaneously and when to switch from one component to another.

Well-designed semantic annotations is an important mechanism in the design of multimedia since they can add coherency to a collection of components, as well as make a disjoint collection of information usable and useful to a particular audience. These benefits are not without cost; it requires significant expert time to add this type of semantic information.

Semantic editing does not occur just at the development phase of the authoring process. It can also occur when the multimedia document is distributed. For knowledgeable users this "user-level authoring" allows for the manipulation and customization of multimedia documents, an important consideration for fields such as education, where the documents cannot be simply static presentations, but should, instead, be tailored to the needs of the educator and audience.

One area of research is the temporal and spatial composition of multimedia materials. Early systems merely mapped the materials onto a time line with related screen position information stored as metadata. More complex techniques, such as object composition Petri Nets (OCPN) and composition trees make more flexible compositions practical.[1,12,15] Projects in the Dartmouth Experimental Visualization Laboratory (DEVLAB) are exploring more efficient means of composing multimedia documents from component objects.

2.5 Evaluation and refinement

At this point, the complete multimedia document should be tested and refined through various cycles of evolution. A test group of users determines appropriateness of content, features, and semantic links. When possible, the document is also distributed to an appropriate sample of novice and expert users to obtain a range of opinions. Careful selection and distribution to a few

"beta-test sites" can provide valuable suggestions on use of the product. In the case of multiple "primary authors", it is important to obtain verification that the materials are presented correctly. Early dissemination to authors is very important, as it reassures the authors that they have control over their materials and helps in repairing incorrect semantic links. This testing and evaluation process should not be skipped as it invariably catches many errors.

Finally, the multimedia document can be released (a) as a retrievable software package on the network, (b) as a remote "document server" on the network (issues pertaining to such servers are discussed further in section 4), (c) on CD-ROM (usually with commercial distribution), or (d) on a related medium.

3. DIGITAL VIDEO EDITING

In most such applications, the use of video and its contribution to a document must be carefully assessed due to the excessive amount of resources needed to process it. In this section, we describe a digital video editing system that permits multimedia authors and users of multimedia documents to interactively manipulate large depositories of video both remotely and locally.

While existing digital video editing systems are easy-to-use and very polished, they rely on a direct manipulation style of user interface. This makes them inflexible tools for automating repetitive or media-sensitive editing operations. It also makes them reliant on a high-bandwidth connection between the video data and the user interface, making remote editing infeasible. At the Dartmouth Experimental Visualization Laboratory (DEVLAB) we are investigating programmable video editing systems as an answer to these shortcomings, and a prototype of such a system, called VideoScheme, has been developed.

3.1 VideoScheme: a programmable video editor

VideoScheme is a prototype programmable digital video editing system, developed at Dartmouth College in order to investigate programming approaches to video.[20,21] 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 user-selectable levels of temporal detail.

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, along with text windows for editing and executing Scheme functions. Functions typed into the text windows can be immediately selected and evaluated. The environment 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

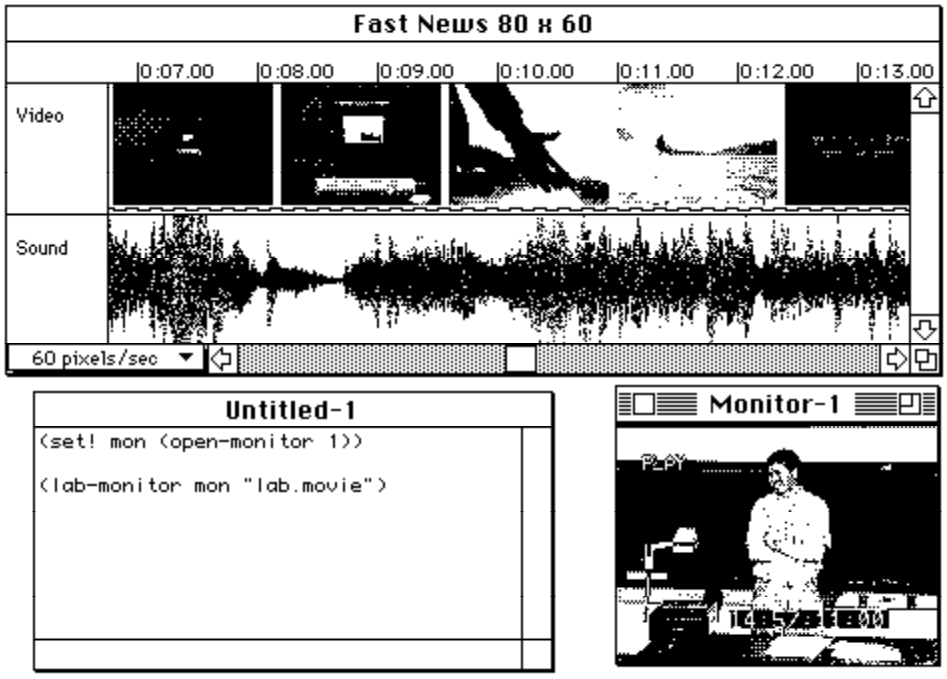

Fig. 3. VideoScheme User Interface

video and audio. Figure 3 shows an instance of the user interface of VideoScheme which allows direct user manipulation.

The concept of providing additional power in an editing program through the use of programmability is not new.[30] Common text editing programs such as Microsoft Word and Emacs provide a programming engine and build complex functions in their associated languages. Emacs, in fact, has a very LISP-like programming language. VideoScheme is the first video editor to implement this concept. Of course, programming is an advanced skill and many users will not have the necessary experience or knowledge to utilize the programming features of VideoScheme. In such normal cases, VideoScheme allows a skilled developer to write potentially complex editing functions in the Scheme programming language, while still providing simple capabilities for the naive user. Once a program is developed and tested, it can be mapped to menu or keystroke options for the normal user. Hence, the capabilities of the editor can be easily extended.

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.

The fact that Scheme is an interpreted language makes it idea for a distributed environment. VideoScheme can be considered to have two major components, a graphical front end, and a Scheme back end. These two components need not always run in the same system, as is detailed in the next section. The graphical front end can run on a local machine and the interpreter back end can be run on a remote video server. The common component of the two, the Scheme programming language, is interpreted and, as such, is common to the two environments, even though they may be totally disparate architectures and operating systems. As an example, a project is currently underway to move VideoScheme from its current Macintosh host to a Silicon Graphics workstation. In this workstation the graphics front end will be rewritten for the Motif programming environment while the interpreter and core functions of the program are little changed.

3.2 VideoScheme language features

VideoScheme extends the Scheme language to accommodate digital media. In this section a few of the VideoScheme data objects and functions specific to video and audio editing and manipulation are described in order to illustrate the design of the program. In addition to the standard number, string, list, and array data types VideoScheme supports objects designed for the manipulation of digital video, such as:

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 built-in as well and user developed 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)
Extracts sound samples from an audio track

With this small set of primitive objects, and small number of built-in functions, one 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. Cut-detection is an example of an information retrieval tool for video in that is allows segmentation of video so as to simplify searching.

One point to be made about VideoScheme is that it is a passive editing system. Edits are made by creating new reference lists of existing data or deriving new data. This structure is necessitated by the large volumes of data in a typical video segment. An advantage of this approach is that VideoScheme is an ideal test bed for information retrieval concepts since it does not directly modify files it manipulates.

4. DISTRIBUTED AND REMOTE VIDEO ACCESS

The traditional approaches employed in text databases, such as keyword searching and volume browsing, are inadequate for our purposes because (a) they do not apply to video at all, or (b) they are not practical due to the volume 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 of disk access time 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 wide area networks such as 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.

A distributed approach to video databases is attractive since it is scalable. As an example, the Internet is a system that scales to millions of users. Video databases attached to the Internet can reach large communities. It is important to devise 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 system presented here balances the needs of the novice user against the network and server resource requirements.

4.1 Distributed editing facilities in VideoScheme

One of the main ideas of this system is that VideoScheme's melding of direct manipulation with programmability is a promising approach for manipulating digital video in an information retrieval environment.[20,21] It is even more promising in a distributed environment, where the user is separated from the video data by a limited-capacity network (as described in a later section). 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 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 4 represents the conventional, local processor

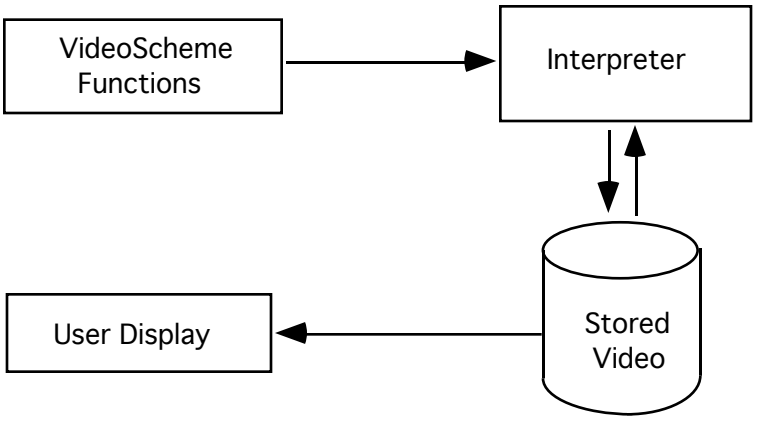

Fig. 4. VideoScheme Single Processing Model

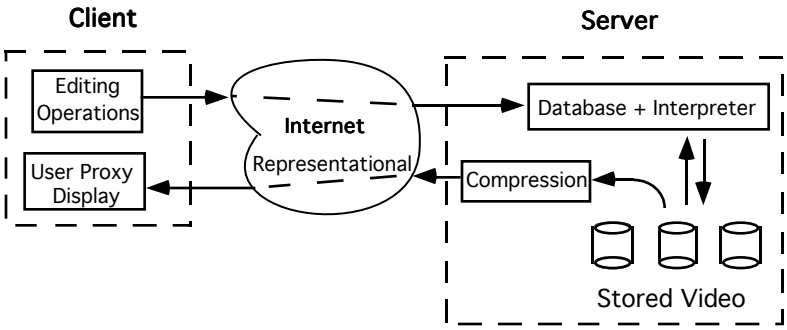

Fig. 5. VideoScheme Client-Server Model

implementation of VideoScheme. The distributed client-server extension of the VideoScheme system is illustrated in the Figure 5.

4.2 Applying VideoScheme on a large-scale video retrieval system

One of the numerous applications for multimedia digital video editing is in education. An easily accessed video library allows an educator to create exciting and relevant presentations for the classroom. Such presentations might show important historical events, illustrate recent scientific achievements, or show a class the process of dismantling an engine block. With access to a large, easily-searched video database the teacher can use clips that are tightly related to the course material of the moment.

Another important application is in research. The research community could benefit from having tools that allow it to easily access and manipulate data from central video resources. Video data such as satellite imagery, historic film 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.

4.3 Basic scheme for remote video querying

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 to video querying, (b) the image processing approach and (c) advanced information retrieval approaches.

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.

The *image processing* approach to video queries attempts 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, 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. Of course, this entails reading all the frame data which is to be searched.

Advanced approaches search metadata created once. These approaches include searching 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 network 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.

4.4 Manual query enhancement

Since query technology is a long way from perfection, the query process can be enhanced by bringing the user into the system. Figure 5 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. In this paper, this process is referred to 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 this model differs from the traditional, simple approach of simply sending the video.

If the complete query results are sent to the user the network will be flooded with traffic, most of which the user does not want, and the user will endure significant delays. The solution to this problem is *representational video.*

Representational video is video that has been decimated in spatial and temporal 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 1 or 2 frames per second. The user can then quickly discard unneeded query responses and edit the desired video into a "rough cut," using a video editor such as the VideoScheme system.[20,21]

Figure 6 illustrates how users edit and retrieve clips 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. Decimated versions of the clips are 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.

4.5 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.

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 composes two

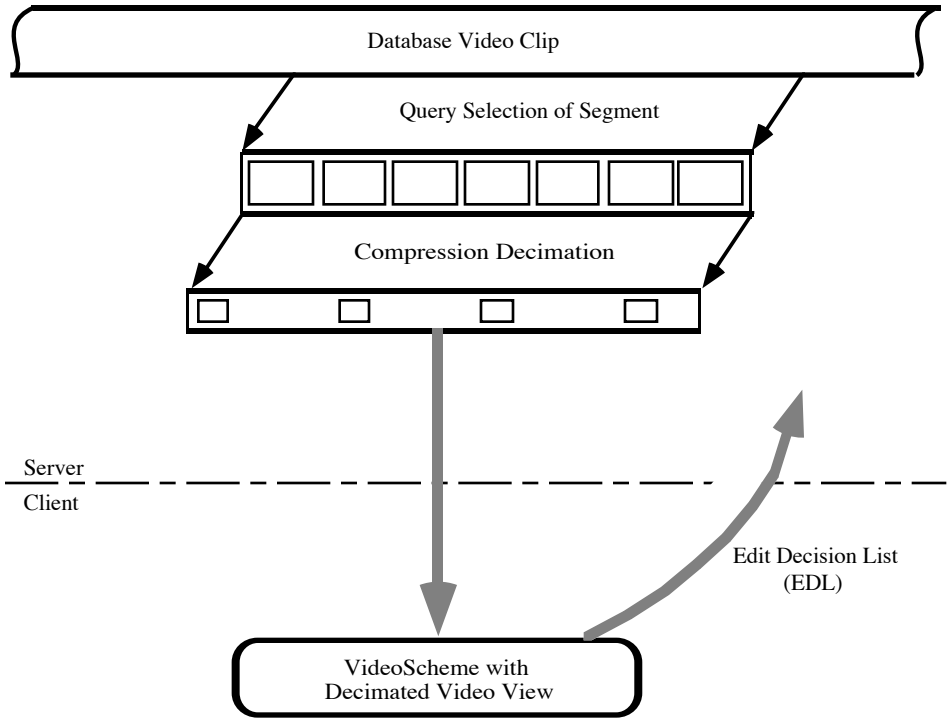

Fig. 6. The Retrieval Model

uncorellated compression processes, which can degrade the video 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 and/or temporal) destination. It has the advantage that information from the original compression process, such as motion estimates and information frame insertion points, 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.

4.6 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 produced ten times as much video as was required. Using a compression decimation ratio of 100:1, the total network traffic is reduced by 89%. The query

overhead on the final video selection is only 10% as opposed to 900% for the naive approach.

Representational video need not be simply linear decimations of the original video. Tools such as VideoScheme combined with learning from the compression process allows for more intelligent representational video. An example would be representational video which is aware of scene changes, as detected using a cut detection algorithm, and presents the decimated video with varying frame rates so as to maximally transmit the scene information. This is one goal of programmable video editing, the processing of video beyond simple cut and paste. It also illustrates the research capabilities of VideoScheme, which allow such algorithms to be easily implemented.

5. FUTURE ISSUES IN MULTIMEDIA DEVELOPMENT

Future research in multimedia development will require computer scientists and engineers to work together with other disciplines in order to draw problems as well as solutions. It is our belief that the ultimate criterion of a successful multimedia system is its usability and for this to succeed, a very complex synergism of various experts must occur. The automation of this process alone is at it infancy. Some example applications where synergism would be essential, are listed below:

(a) Electronic publishing involves real issues of copyright laws, economics and marketing. For efficient multimedia system production in this field, it is necessary for computer scientists to understand certain real-world problems before they can proceed with the development of tools that automate, for example, multimedia object composition. On the other hand, large scale electronic publishing over the Internet will require the expertise of efficient computational methods for searching, pattern recognition, compression, network communication, etc.

(b) The creation of multimedia documents needs to satisfy certain aesthetic thresholds in the visual presentation and composition of information. This is uncharted territory for computer scientists and engineers. The cooperation with artists and designers is imperative.

(c) The development of multimedia systems for learning, training or teaching applications is certainly a big market. However, there are many open questions which are hard to answer, such as: "what is the effectiveness of multimedia systems in teaching topic X to audience Y," or, "How is the learning best achieved with multimedia materials Z?" Educators, cognitive psychologists and course designers must work together with the multimedia document developers, a process that is very unrealistic for mass production.

(d) Scientific applications such as multimedia authoring tools to manipulate satellite image databases require the interdisciplinary expertise of the scientists

and indexing such data efficiently for fast and accurate retrieval is a very hard problem.

The above is only a small sample of the complexity of multimedia development systems. Another important future direction is how to automate the integration and multimedia document development process. This is an issue that has already been discussed in this paper. An array of tools and technologies must come together, as well as an understanding of the commercial, industrial and real world issues behind a given multimedia application.

Mechanisms and network technologies that allow remote query processing must be improved so that the appropriate amount of video to be delivered in chosen. One approach to this balancing of resources is illustrated in this paper. This process places some unusual demands on an 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.

6. ACKNOWLEDGMENTS

This research was supported in part by the National Science Foundation (NSF grants 5-34251, 5-34294, 5-34332), the New England Consortium for Undergraduate Science Education (NECUSE), and the Dartmouth Institute for Advanced Graduate Studies.

Many people have contributed to the ideas and work presented here. Special thanks go to Donald Johnson, P. Takis Metaxas, Jun Zhang, Peter Gloor, Qin Zhang, and Grammati Pantziou.

7. REFERENCES

1. J. F. K. Buford (contributing editor), *Multimedia Systems,* ACM Press: New York, NY, 1994.

2. V. Bush, "As we may think," *Atlantic Monthly*, 1945.

3. A. Califano 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.

4. M. Cheyney, P. A. Gloor, D. B. Johnson, F. Makedon, J. W. Matthews, and P. Metaxas, "Conference on a disk: A successful experiment in hypermedia publishing (extended abstract)," *Educational Multimedia and HyperMedia*, AACE: Charlottesville, VA, 1994.

5. M. Cheyney, P. A. Gloor, D. B. Johnson, F. Makedon, J. W. Matthews, and P. Metaxas, "Towards multimedia conference proceedings" *Communications of the ACM* (Invited paper, to appear).

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

7. S. B. Dynes and P. A. Gloor, "Using hierarchical knowledge representation for an animated algorithm learning environment," *MIT LCS/TNS Technical Report*, 1993.

8. S. T. Dumais, "Enhancing performance in latent semantic indexing (LSI) Retrieval," *Bellcore Technical Report TM-ARH-017527*, Bellcore, 1990.

9. S. T. Dumais, "Improving the retrieval of information from external sources. Behaviour research methods," *Instruments and Computers*, vol. 23 (2), pp. 229-236, 1991.

10. R. Garrett, "Digital libraries, the grand challenges," *Educom Review*, 28(4) (Aug, 1993), 17-21.

11. J. Gemmell. and S. Christodoulakis, "Principles of delay-sensitive multimedia data storage and retrieval," *ACM Transactions on Information Systems*, 10(1) pp. 51-90, January 1992.

12. S. Gibbs, C. Breiteneder, and D. Tsichritzis, "Data modeling of time-based media," in *Visual Objects,* Université de Genève, 1993, 1-21.

13. P. A. Gloor, "Cybermap, yet another way of navigation in hyperspace." *Proc. ACM Hypertext 91,* pp. 107–121, San Antonio, Tx, 1991.

14. P. A. Gloor, F. Makedon, and J.W. Matthews (editors), *Parallel Computation: Practical Implementation of Algorithms and Machine,* Springer-Verlag, 1993.

15. L. Hardman, G. van Rossum, and D. C. A. Bulterman, "Structured multimedia authoring," *ACM Multimedia '93*, Anaheim, CA, 1993.

16. D. B. Johnson and F. Makedon, "Building a digital library with extensible indexing and interface: A discussion of research in progress," *Digital Libraries Workshop*, Rutgers Univ., Princeton, 1994.

17. D. Johnson, J. Ford, F. Makedon, C. Owen, G. Pantziou, S. A. Rebelsky, and Q. Zhang, "Automation of multimedia publishing for educational applications", Dartmouth College Dept. of Computer Science technical report. (In preparation), 1994.

18. F. Makedon and C. Owen, "A digital library proposal for support of multimedia in a campus educational environment," *EdMedia '94 Poster Presentation*, 1994.

19. F. Makedon, S. A. Rebelsky, M. Cheyney, C. Owen, and P. A. Gloor, "Issues and obstacles with multimedia authoring," *Educational Multimedia and HyperMedia*, AACE, Charlottesville, VA, 1994.

20. J. Matthews, P. A. Gloor, and F. Makedon, "VideoScheme: A programmable video editing system for automation and media recognition," *ACM Multimedia '93*, Anaheim, CA, 1993.

21. J. Matthews, F. Makedon, and P. A. Gloor, "VideoScheme: a research, authoring and teaching tool for multimedia," *Educational Multimedia and HyperMedia*, AACE, Charlottesville, VA, 1994.

22. R. Rada (prod. chair), *Proceedings CD-ROM of the First ACM International Conference on Multimedia*, Anaheim, California, 1993.

23. S. A. Rebelsky, F. Makedon, P. T. Metaxas, J. Matthews, C. Owen, L. Bright, K. Harker, and N. Toth, "Building multimedia proceedings: the roles of

video in interactive electronic conference proceedings," submitted to *ACM Transactions on Information Systems special issue on Digital Video in Multimedia Systems*, 1995.

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

25. I. Rigoutsos 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.

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

27. G. Salton, "Automatic text indexing using complex identifiers," *ACM Conference on Document Processing Systems*, Santa Fe, pp. 135–145, 1988.

28. G. Salton and C. Buckley, "Automatic text structuring and retrieval - experiments in automatic encyclopedia searching," TR 91-1196, Cornell U., Ithaca NY, 1991.

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

30. D. Tennenhouse, J. Adam, C. Compton, A. Duda, D. Gifford, H. Houh, M. Ismert, C. Lindblad, W. Stasior, R. Weiss, D. Wetherall, D. Bacher, D. Carver, T. Chang, *The Viewstation Collected Papers*, MIT Laboratory of Computer Science Technical Report TR-590, Release 1, 1993.

31. F. A. Tobagi, and J. Pang, "StarWorks -- A video applications server," *Digest of Papers: IEEE COMPCON Spring '93,* pp. 4-11, 1993.

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

33. C. J. van Rijsbergen, "A non-classical logic for information retrieval," The *Computer Journal*, vol. 29, No. 6, 1986.

34. C. J. van Rijsbergen, "Towards an information logic" *SIGIR '89*, pp. 77-86, 1989.

35. J. Zalewski, Review of "Parallel Computation: Practical Implementation of Algorithms and Machines." *IEEE Computer*, July 1994.