Indexing Multimedia for the Internet

Brian Eberman, Blair Fidler, Robert Iannucci, Chris Joerg, Leonidas Kontothanassis,
David E. Kovalcin, Pedro Moreno, Michael J. Swain, Jean-Manuel Van Thong

Cambridge Research Laboratory
Compaq Computer Corporation
Cambridge, MA 02139, USA
swain@crl.dec.com

Abstract. We have developed a system that allows us to index and deliver audio and video over the Internet. The system has been in continuous operation since March 1998 within the company. The design of our system differs from previous systems because 1) the indexing can be based on an annotation stream generated by robust transcript alignment, as well as closed captions, and 2) it is a distributed system that is designed for scalable, high performance, universal access through the World Wide Web. Extensive tests of the system show that it achieves a performance level required for Internet-wide delivery. This paper discusses our approach to the problem, the design requirements, the system architecture, and performance figures. It concludes by showing how the next generation of annotations from speech recognition and computer vision can be incorporated into the system.

1 Introduction

Indexing of Web-based multimedia content will become an important challenge as the amount of streamed digital video and audio served continues to grow exponentially. Inexpensive storage makes it possible to store multimedia documents in digital formats with costs comparable to that of analog formats, and inexpensive, higher-bandwidth networks allow the transmission of multimedia documents to clients in corporate intranets and through the public Internet. The very rapid growth in the use of streaming media players and browser plug-ins demonstrates that this is a compelling medium for users, and the ease of use of products such as Microsoft's NetshowServer^{TM} and RealNetworks' RealServer^{TM} will make it possible for even small organizations to distribute their content over the Internet or Intranets.

We built the CRL Media Search system (termed Media Search from now on) to investigate mechanisms for indexing video and audio content distributed via the Web. The Media Search service bears significant resemblance to existing search engines on the World Wide Web. Like search engines it allows the users to perform a text search against a database of multimedia documents and return a list of relevant documents. Unlike standard search engines, it also locates the matches within the documents – search engines are able to leave this task up to the browser.

We completed initial implementation of the system in early March, 1998 and have been running it nearly continuously over Compaq's corporate intranet, with updates, since then. A broad range of content sources has been added to the system. The content includes broadcast material from the Web, technical talks, produced management conferences, and captured broadcast video content. The audio quality of these assets varies widely and has provided a very useful set of tests. Statistical analysis of the system usage has enabled the development of a user model for extensive scalability testing of the system.

This paper summarizes the system design issues and contrasts our solutions with some that have appeared previously in the literature. We then present the basic system architecture and how the system functions, and conclude with discussion of our plans for incorporating additional annotation types into the system.

2 Design Issues

During the past few years a number of video indexing systems have been built. These systems generally take in a video feed from an analog source, digitize the source feed and then perform indexing on the content. Both research and commercial systems have been built.

The News-on-Demand project, part of Informedia, at CMU [4] is a good example of this type of system. In this project CNN and other live feeds were digitized, then image analysis was performed to cut the video into shots. Shots are then grouped into scenes using multiple cues. The speech in text format for each scene is then indexed using a standard text indexing system. Users can then send queries to retrieve scenes. Work since [4] has focused on video summarization, indexing with speech recognition, and learning the names of faces. A similar example is a commercial product from Virage$^{\text{TM}}$. This product can provide a system to do real-time digitization, shot cut detection, and closed-caption extraction. Off-line, the system provides a keyframe summary of the video over a Web browser; by selecting time periods represented by pairs of keyframes, the user can add annotations to the index. Finally, Maybury's [5] system at MITRE focuses on adding advanced summarization and browsing capabilities to a system that is very similar to Informedia.

These previous works, and others, employ the *shot* – an uninterrupted sequence of video from a single camera view – as the basic atomic unit for all additional processing. While shots are clearly important, this type of structuring of the information is not always appropriate. Davenport *et al* [2] introduced the idea of a stream-based representation of video from which multiple segmentation can be generated. In a stream-based representation, the video is left intact, and multi-layered annotations with precise beginning and ending times are stored as associated metadata with the video. Annotations can be a textual representation of the speech, the name of a person, objects in the scene, statistical summaries of a sequence of video, or any other type of data.

A stream-based annotation system provides a more flexible framework and can always be reduced to a shot/scene representation by projecting the time intervals of the other annotations against the shot annotations. In our work, the stream-based approach can be used to produce a text oriented display. For example, if the system has primarily indexed conversations or speech, as ours has, then what is of interest to the user is the structure of the textual representation of the speech. A single keyframe per paragraph could be more appropriate than one determined from image analysis. A second example is content reuse and repurposing. Content companies are very interested in reusing old content for new productions. In this case, the semantic content of the story is not of interest. Instead the basic objects, people, and settings are of value. Annotations should mark their appearance and disappearance from the video. As a final case, consider indexing a symphony video based on instruments and scores. In this case a visually-based temporal segmentation is not appropriate, but one based on the musical structure is. Our system, paired with a Web-based annotation tool we built, can support all these differing styles of annotation.

Another difference in our system is that we treat the Web as not just a delivery mechanism, but as the basic infrastructure on which to build. We believe that the ease with which video and audio content can be placed on the web will soon cause a wide assortment of groups to distribute their multi-media content as ubiquitously as their text documents. We have experience with large corporate and institutional archives, and news and other content-production companies. All of these types of organizations are actively considering or have started to make this move. When this happens, video indexing systems will be needed not to index analog feeds, but to index video that has been placed on multi-media content servers. To investigate this issue, we designed our system so that HTTP-based content servers, which we call Media Servers, could be distributed across the organization, or anywhere on the Internet, and then be indexed from one central location. As a proof of the concept, we indexed audio content on NPR's web site in our system. Users could search this content using our system; when they played one of these clips it was delivered directly from the NPR site.

Since our system is designed to be used by a wide variety of users, we built an HTML-based user interface that would work on all major web browsers. To provide access across a world-wide corporate intranet, the content delivery was based on low-bitrate streaming video.

The system follows the standard search engine user interaction model. This model consists of two steps: First, users search and get pointers to documents, and then they go to the documents and use the browser's *find* command to search for their particular query. Since *find* is not a browser supported function for video, we had to create a way of supporting *find* through HTML pages that was consistent with this model.

Since we worked with a very broad range of content sources, the audio quality of these materials varied broadly and followed many different production formats. Thus our system could not use knowledge of the format for browsing. We further had to develop very robust methods for aligning the available, occasionally inaccurate, transcripts to very long audio segments (often greater than 1 hour in length) containing speech, music, speech over music, speech with background noise, etc. A paper by Moreno *et al* [6] reports on this algorithm.

Finally, this area is rapidly evolving new techniques for automatically annotating content. This, plus our first experience with trying to build a content processing system led us to develop a distributed periodic processing model with a central database as a workflow controller. This approach is discussed more throughly in DeVries [3].

3 System Description

The Media Search system, as shown in Figure 1, is broken into six components: 1) one or more Media Servers, 2) a metadatabase that is a built on a standard relational database, 3) a collection of autonomous periodic processing engines or daemons managed by the metadatabase, 4) an Indexing System which is a modified version of the NI2 index used in the AltaVista engine, 5) a Feeder to synchronize information in the database with the NI2 index, and 6) a Presentation Server that communicates with the other subsystems to construct an HTML response to a user query.

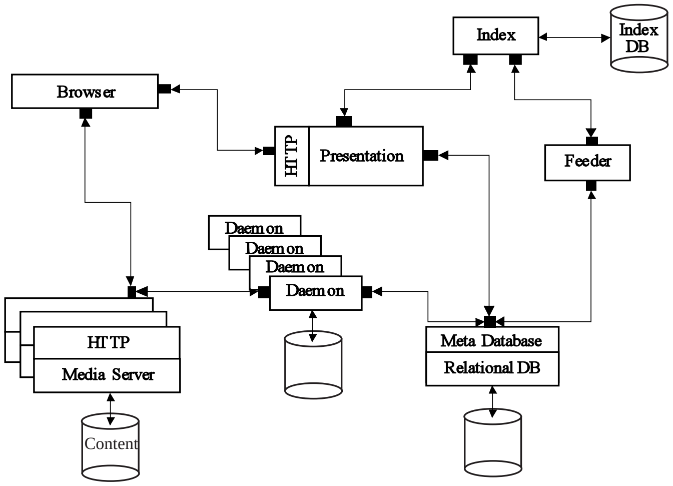


Fig. 1. CRL Media Search Architecture

Media Server A Media Server stores and serves up the actual content and provides a uniform interface to all the media stored in the system. In addition, it handles storage and access control, if required. We use it to store video (MPEG, RealVideo$^{\mathrm{TM}}$), *audio*(RealAudio$^{\mathrm{TM}}$), and images (JPEG). We also implemented on-demand conversion functions to convert from MPEG to other formats. For large-scale Internet applications of the system, on-demand conversion is only used to produce formats accessed by our internal processing daemons. All formats accessed by user queries are pre-produced in order to improve performance. Both the stored files, and the "virtual" files which are computed on-demand are accessed through a uniform URL interface. The URL-based interface to Media Servers provides a layer of abstraction allowing storage versus computation trade-offs, and makes it possible to locate Media Servers anywhere within an intranet or across the Internet.

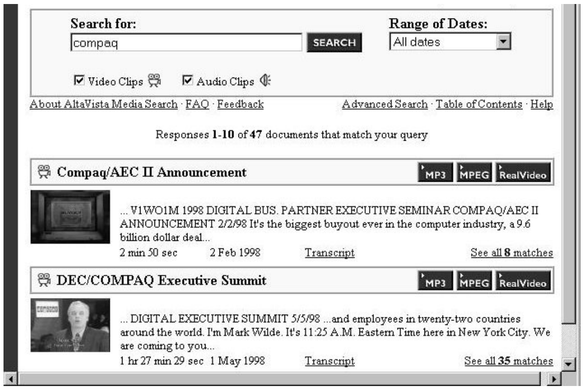
Meta-Database and Daemons The metadatabase, which is built on a relational database, performs three functions. Firstly, it keeps track of the location of all stored formats, or representations, of each multimedia document. Secondly, the metadatabase acts as a central workflow control system for daemon processing. A daemon is a process that performs "work", typically taking one or more representations and computing an output representation. For example, the alignment daemon takes a transcript and an audio file as input and computes an aligned transcript as output. Central workflow processing is enabled by having each daemon type register the format of its required inputs when that type is first installed. Then each daemon instance can request work from the metadatabase and is given the input URL handles that need to be processed. When a daemon completes work on its input representations, it stores the output representation in a Media Server and registers the availability of the new representation with the metadatabase. This simple model leads to a robust, distributed processing method which scales to the large processing systems needed for the Web.

The third role of the metadatabase is to store all the stream-annotation information. Although we physically store the annotations in a way optimized for the presentation user interface, the format of the annotation tables can be thought of as tables giving the type, start time, end time, and value. The system is structured so that it can store arbitrary annotation types in this way. For example, using speech recognition technology we are able to align transcripts to the audio component of the video [6]. The annotation stream then consists of the start and end time of each word in the transcript, and the word itself. The system is sufficiently flexible that we can store a large variety of different forms of annotations, although currently we only store aligned transcripts and images, called keyframes, which represent a video shot.

Index and Feeder One of the most important problems in video and audio indexing is not only to index the collection of words that were spoken during the video, but also to be able to determine where a particular word, phrase, or combination occurred. It was our aim to support this capability while still indexing full documents. We modified the NI2 index used by AltaVista [1] to accomplish this task. While the original NI2 index returns the ID of a document containing the words of the user query, our modified version will return multiple hits per document; one hit for every location in the document that matches the query.

In order to provide within-document indexing, we first had to define what were the within-document match locations to a given query. A match location provides an entry point into the video; the user is then free to play the video for as long as desired. Therefore, a match location naturally defines a subdocument, starting at the match location and ending at the end of the video. The match locations were then defined to be all the locations where terms from the query matched the document, and which defined a subdocument that matched the complete query.

We also extended the model to include rank queries, that is, how to rank certain subdocuments matches more highly than others. A standard *term frequency / inverse document frequency* (tf.idf) metric [7] is used, with each term match multiplied by a position-dependent factor. To enhance the rank of subdocuments where there were many term matches appearing soon after the match location, the position-dependent factor takes the form of an exponential decay with its peak at the beginning of the subdocument. At this point we have not done sufficient information retrieval (IR) testing of this metric to report on the IR performance of within-document ranking.



Search for: compaq

Range of Dates: All dates

☑ Video Clips ☑ Audio Clips

Responses **1-10** of **47** documents that match your query

Compaq/AEC II Announcement

... V1WO1M 1998 DIGITAL BUS. PARTNER EXECUTIVE SEMINAR COMPAQ/AEC II ANNOUNCEMENT 2/2/98 It's the biggest buyout ever in the computer industry, a 9.6 billion dollar deal...

2 min 50 sec 2 Feb 1998 Transcript See all **8** matches

DEC/COMPAQ Executive Summit

... DIGITAL EXECUTIVE SUMMIT 5/5/98 ...and employees in twenty-two countries around the world. I'm Mark Wilde. It's 11:25 A.M. Eastern Time here in New York City. We are coming to you...

1 hr 27 min 29 sec 1 May 1998 Transcript See all **35** matches

SEARCH
About AltaVista Media Search · FAQ · Feedback
Advanced Search · Table of Contents · Help
MP3
MPEG
RealVideo
MP3
MPEG
RealVideo

Fig. 2. CRL Media Search Query Response Page

Presentation Server The Presentation Server communicates with the various services offered by the metadatabase, the Index Server, and the Media Server to allow the user to perform searches, see responses, and view parts of the video stream. A screen shot of the initial response produced by the Presentation Server after a query is shown in figure 2. From this page the user may decide to play the video from the first match within a document, or to go to another page to see all the matches within the video. From this document-specific match page, the user may play the video from any one of the matches within the document, search within the specified document, or progress to yet another page. This third style of page allows the user to browse the video starting at the location of one of the matches.

A typical response to a user query is computed as follows: The presentation system first sends a message to the Index Server; then, using a handle returned by the index, finds information in the metadatabase. In this way the presentation system uses the two systems together to compute responses to queries, in effect making the NI2 index an extension of the relational database. Although this type of extension can in principle be achieved with object relational databases or in extended relational databases by putting the index within the database, our specialized external text index that is synchronized with the database offers a higher performance solution.

4 System Performance

The system was tested running on two Digital AlphaServer 4100's. One machine had four Alpha 21164 processors running at 466 MHz, 2 gigabytes (GB) of memory, and a disk array with 600 GB of data storage. This machine ran the database, index, and Media Server components of Media Search. The second machine had four 400MHz Alpha 21164 processors and 1.5 GB of memory and ran the presentation server.

The test harness was a set of client processes probabilistically emulating the user model derived from the system deployed on Compaq's internal network. The server side used Oracle 8.0.5 as the relational database with a custom developed interface between the presentation layer and the Oracle engine, and the NI2 index engine with the extensions described in section 3. The presentation server used the Apache web server with the presentation CGI scripts developed in Perl. To avoid the high startup overhead of Perl for every invocation of a CGI script we used the FastCGI extensions to CGI which allow Perl processes to become servers and converts CGI calls to socket communication between the Web server and the resident Perl processes.

Our system has a throughput of 12.5 user sessions per second (as defined by the user model), resulting in 36 pages served per second, or approximately 3.2 million pages over a 24-hour period. We also achieved an average latency of less than 0.5 seconds per page served. The Presentation System, written in Perl, was the bottleneck in these tests, even with the FastCGI optimizations. Performance improvements to the presentation system could be obtained, for example, by rewriting it in C/C++ instead of Perl.

We have also conducted performance tests of the metadatabase as a standalone component since it is the only component of our system whose performance can not be improved simply by replicating the data and using more machines to service requests. All other components can be distributed/replicated among multiple machines fairly easily. The test of the metadatabase component alone running on one AlphaServer 4100 with four Alpha 21164 processors at 466MHz measured a throughput of 29 user sessions per second, equivalent to 88 pages per second or 7.5 million pages over a 24-hour period.

5 Conclusions and Future Work

We implemented the CRL Media Search system to explore the issues that arise when building a distributed system for indexing multimedia content for distribution over the World Wide Web. The system uses the Internet as the basic platform for both organizing the computation and distributing content to users. We have tested the performance of the system and found that it scales well and can provide an indexing service at a cost comparable to indexing text (HTML) documents.

We are investigating adding other meta types of annotation information to the system. For instance, the meta-information extracted by a face detector/recognizer and speaker spotter can be placed directly into the metadatabase. This information can then either be indexed in the current NI2 index or in a second NI2 index. In addition, we are researching indexing crawled multimedia from the World Wide Web, where we do not have transcripts or closed captions. Sound classification and speech recognition are key technologies for this area of research.

Acknowledgements

The Media Search project has benefited from the insights, support, and work of many people, including: Mike Sokolov, Dick Greeley, Chris Weikart, Gabe Mahoney, Bob Supnik, Greg McCane, Katrina Maffey, Peter Dettori, Matthew Moores, Andrew Shepherd, Alan Nemeth, Suresh Masand, Andre Bellotti, S. R. Bangad, Yong Cho, Pat Hickey, Ebrahim Younies, Mike Burrows, Arjen De Vries, and Salim Yusufali. Thanks to Beth Logan for comments on drafts of this paper.

References

1. Burrows, M. Method for indexing information of a database. *U.S. Patent 5745899*, (April 1998).
2. Davenport, G., Aguierre-Smith, T. G., and Pincever, N. Cinematic primitives for multimedia. *IEEE Computer Graphics and Appplications 11*, 4 (July 1991), 67–75.
3. de Vries, A. P., Eberman, B., and Kovalcin, D. E. The design and implementation of an infrastructure for multimedia digital libraries. In *International Database Engineering Applications Symposium* (July 1998).
4. Hauptmann, A., Witbrock, M., and Christel, M. News-on-demand - an application of informedia technology. In *D-LIB Magazine* (Sept. 1995).
5. Mani, I., House, D., and Maybury, M. *Intelligent Multimedia Information Retrieval.* MIT Press, 1997, ch. 12: Towards Content-Based Browsing of Broadcast News Video.
6. Moreno, P. J., Joerg, C., Van Thong, J.-M., and Glickman, O. A recursive algorithm for the forced alignment of very long audio segments. In *International Conference on Spoken Language Processing* (1998).
7. Salton, G., and McGill, M. J. *Introduction to Modern Information Retrieval* McGraw-Hill, 1983.

This article was processed using the LaTeX macro package with LLNCS style