Browsing and Retrieval of Full Broadcast-Quality Video

David Gibbon, Andrea Basso, Reha Civanlar,
Qian Huang, Esther Levin, Roberto Pieraccini
AT&T Labs Research

Abstract

In this paper we describe a system we have developed for automatic broadcast-quality video indexing that successfully combines results from the fields of speaker verification, acoustic analysis, very large vocabulary speech recognition, content based sampling of video, information retrieval, natural language processing, dialogue systems, and MPEG2 delivery over IP. Our audio classification and anchorperson detection (in the case of news material) classifies video into news versus commercials using acoustic features and can reach 97% accuracy on our test data set. The processing includes very large vocabulary speech recognition (over 230K-word vocabulary) for synchronizing the closed caption stream with the audio stream. Broadcast news corpora are used to generate language models and acoustic models for speaker identification. Compared with conventional discourse segmentation algorithms based on only text information, our integrated method operates more efficiently with more accurate results (> 90%) on a test database of 17 one half-hour broadcast news programs. We have developed a natural language dialogue system for navigating large multimedia databases and tested it on our database of over 4000 hours of television broadcast material. Story rendering and browsing techniques are employed once the user has restricted the search to a small subset of database that can be efficiently represented in few video screens. We focus on the advanced home television as the target appliance and we describe a flexible rendering engine that maps the user-selected story data through application-specific templates to generate suitable user interfaces. Error resilient IP/RTP/RTSP MPEG-2 media control and streaming is included in the system to allow the user to view the selected video material.

Introduction

Recent technical advances are enabling a new class of consumer applications involving browsing and retrieval of full broadcast-quality video. Cable modems are bringing megabit bandwidth to the home. Set top boxes include low cost MPEG2 decoders and can also include an HTTP client for web browsing. Disk storage technology is riding a Moore's law curve and is currently at a dollar-per-megabyte point that makes large digital video archives practical. Selective delivery of digital video over IP is less well established than these other technologies, but rapid progress is being made in this area. Systems that build upon these component technologies in novel ways can bring new services that meet consumers needs. In this paper we present an example system for selective retrieval of broadcast television news with full broadcast quality.

Video indexing is one of the areas in which further technology development is needed. To build systems for video database retrieval, we need standard storage and communication protocols at several levels for handling the video program attributes, key frames, associated text streams such as the closed caption, in addition to the usual issues associated with storage and delivery of the bulk media. (Video program attributes include such things as the program title, copyright, etc.) Currently international standards bodies are active in this area. In particular MPEG7 aims to address this as a natural evolution of the MPEG video standards of the past [MPG7.] The IETF is also working on standards focused in the areas where television and the Internet intersect [Hoschka98, Kate98a, and Kate98b.] Meanwhile the industry is moving ahead with implementations that may become de-facto standards in their own right. For example at the system level we have Microsoft ASF [Fleischman98] and Real Networks [Real98] and at the application level there are Virage and ISLIP and others [Virage98, Islip98, FasTV98, Magnifi98, Excalibur98.]

When these indexing components become stable, we will have all the building blocks necessary to create systems for browsing video databases that have been manually created. The major broadcasters will likely begin to generate indexes of each video program as part of the normal production process. However, for a significant time, smaller broadcasters will not have the necessary resources to do this. Further, it will be too costly to manually index the many large (several hundred thousand hour) video news archives that exist. An automated video indexing system is required for these applications. Several such systems have been proposed (for example, at CMU and AT&T [Wactlar98, Shahraray95b].) A common aspect of the successful systems is true multimedia processing in which state of the art techniques from several disciplines are employed. In fact, in many cases it is necessary to extend existing methods in the process of applying them to the domain of video indexing. In this paper we describe the successful combination of results from the fields of speaker verification, acoustic analysis, very large vocabulary speech recognition, content based sampling of video, information retrieval, natural language processing, dialogue systems, and MPEG2 delivery over IP networks.

The display device itself imposes an additional challenge for broadcast-quality video indexing systems intended for consumers. Typical home PC's are not well designed for the display of interlaced video, and the comparatively low resolution of television sets is not capable of rendering the browsing and retrieval user interfaces (UIs) of the aforementioned systems. We will focus on the home television as the display device. Much of the prior work assumes a desktop environment in which high-resolution graphics are available for the UI, and "postage stamp" quality video is sufficient. In these systems QCIF resolution video serves the purpose of identifying a particular video clip from a database, and for verifying that it is relevant to a given query. This quality is not sufficient for actually watching the video in the way to which a television viewer is accustomed.

There are several issues to be addressed when designing user interfaces for television-resolution applications. Even though the addressable resolution is approximately 640x480 for NTSC displays, the usable resolution is further reduced by overscan and interlacing. Fonts must be large, and preferably smoothed to avoid interlacing artifacts. Also, usage paradigms dictate that scrollbars are to be avoided [WebTV98]. In summary, prior systems employ high-resolution UIs to access low-resolution video, we are concerned here with low-resolution UIs for accessing high-resolution video.

Media Processing

To facilitate browsing of large collections of video material on limited resolution terminals, organizing the data in a structured manner, and presenting this structure through natural user interfaces is necessary for a successful system. The process of populating the collection must be automated as much as possible. At the highest level, video programs have attributes such as "network," "title," and "genre." Any finer grained classification or segmentation will also aid the searching and browsing process. For the case of broadcast news video, we can automatically segment video into semantically meaningful units such as stories and summaries, and derive a content hierarchy [Huang99b].

For the remainder of this section, we will focus on broadcast news content, but much of the processing can be applied to other types of content as well (e.g. commercial/program segmentation, closed caption synchronization.) Please refer to Figure 1 for an overview of the media processing. We assume that we are starting with elementary MPEG2 audio and video streams including a closed caption data stream. In a later section we describe the preprocessing required for streaming the media. Also the content based sampling and representative image selection process are covered in the section on "story rendering for browsing."

Audio Classification and Anchorperson Detection

Our method first classifies video into news versus commercials using acoustic features. Current algorithms can reach 97% accuracy on our test data set. Within the news segments, we further identify smaller segments containing the anchorperson. Gaussian mixture model based anchor and background models are trained using 12 mel cepstral and 12 delta cepstral features and then used to compute the likelihood values estimated from the audio signal [Rosenberg98]. With these techniques, we have been able to achieve a 92% correct segment identification rate, with a low false alarm rate on our test data set [Huang99a.] By further integrating audio characteristics with visual information in recognizing the anchor, the detection precision on detected segments are significantly improved. Such recognized anchor segments serve two purposes. First, they hypothesize the news story boundaries because every news story starts with anchor's introduction (vise versa is not true though).

Story Segmentation
The recognized anchor speech segments the synchronized (see below) closed caption stream into blocks of text to which text analysis algorithms are applied to extract individual news stories. Compared with conventional discourse segmentation algorithms based on only text information, our integrated method operates more efficiently with more accurate results (> 90%) on a test database of 17 one half-hour broadcast news programs. The second purpose in identifying anchor segments is to extract condensed news summaries or introductions spoken by the anchor which are separate from the body of the story. By playing back only the news summary, a user can experience "glancing at headline stories", similar to reading only the front page of a newspaper, before deciding which headline story to browse further.

Caption Synchronization
Real-time closed captioning lags behind the audio and video by a variable amount of time from about 1 to 10 seconds. Since the story segmentation uses all three media streams to perform the segmentation, errors can be introduced if the caption is not aligned with the other media streams. We have had success using a large vocabulary speech recognizer to generate very accurate word timestamps (although the word error rate is considerable.) The system was built using broadcast news corpora and has lexicon of over 230K words [Choi99]. The recognizer runs in 5 times real time on a 400Mhz NT machine using the AT&T Watson Engine with about a 40% word error rate. After recognition, a parallel text alignment is performed to import the more accurate timing information from the automatic speech transcription to the more accurate transcription of the closed caption (the method is similar to that described in [Gibbon98].)

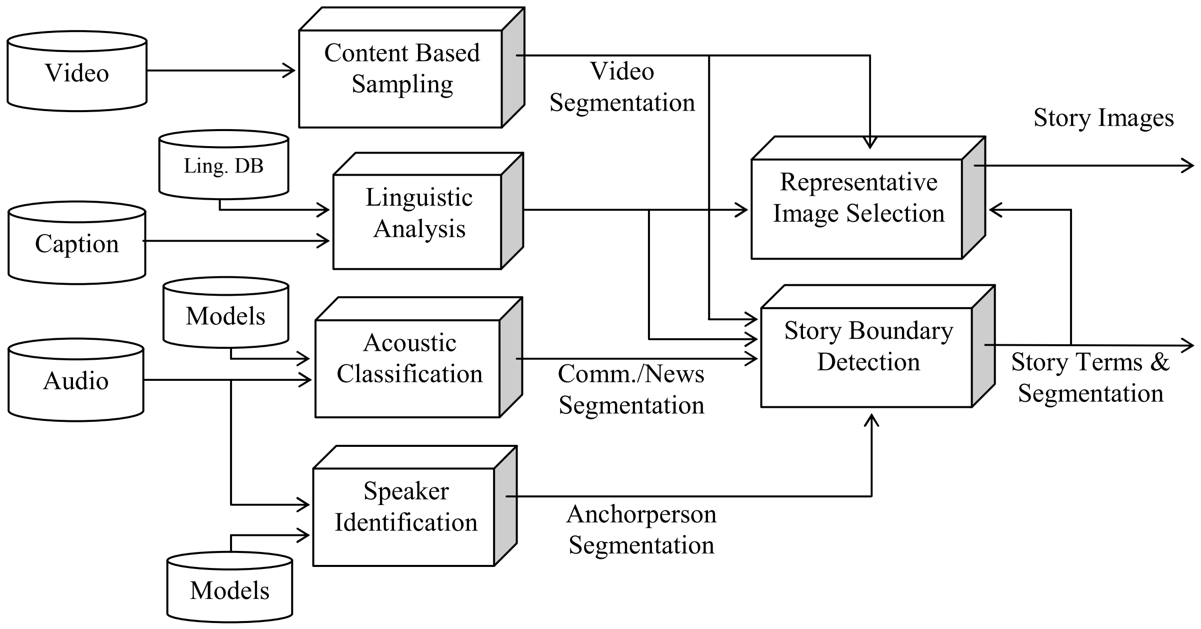


Figure 1: Media processing to determine story segmentation and representative images and terms

Multimedia Database

The story segmentation, keywords, and story image descriptors are stored in a multimedia database along with the full closed caption text and the audio and video media. We have built upon our prior experience with a 4000+ hour heterogeneous multimedia database [Shahraray97.] The database is comprised primarily of broadcast television news programs from NBC, CNN and PBS over the period from 1994 through 1999. Each database entry includes closed caption text and still frames, and may also include 16Kbs audio, 128Kbps video, or 5Mbps MPEG2 video. This large number of items proved useful in designing the dialogue interface described below.

Through a navigation process described below, users select a particular story for display. A rendering engine takes the story data and maps it through templates to generate user interfaces. This template-based architecture allows the content to be used in a variety of applications. Here we use templates to render displays suitable for television display (e.g. black backgrounds, large fonts, etc.) By using a different template set, other interfaces to the multimedia data are produced by the rendering engine. See Figure 2.

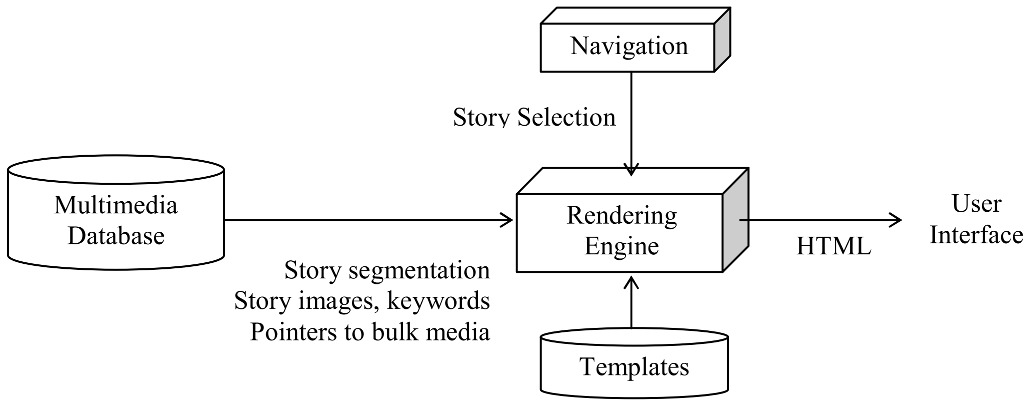


Figure 2: User Interface Rendering

Archive Navigation

When dealing with a large video archive, users engage in three distinct modes of interaction: searching, browsing, and video playback (see Figure 3.) Usually, the searching operation is used first to obtain a small number of documents (or video programs) of interest. This phase is followed by a browsing phase, which deals only with this small set. (Of course there are exceptions to this. For example a large database may be browsed using a calendar-like interface. Also some systems combine browsing with video playback [Smith97].) Users typically traverse many states in each phase, and they may loop back (e.g. from playback to browsing.) For clarity, we do not show this in Figure 3. For low-resolution terminals a natural language dialogue interface is well suited to the searching phase, while a more visually rich interface (e.g. with icon images acting as hypermedia anchors) is a better choice for the browsing phase. In this paper, we will follow this order by first describing a dialogue system for searching and then describing a browsing interface.

The browsing interface allows users to navigate the database within a particular video program. This interface terminates ultimately with a video playback request.

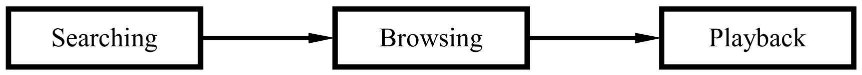


Figure 3: Primary modes of interaction with a video database

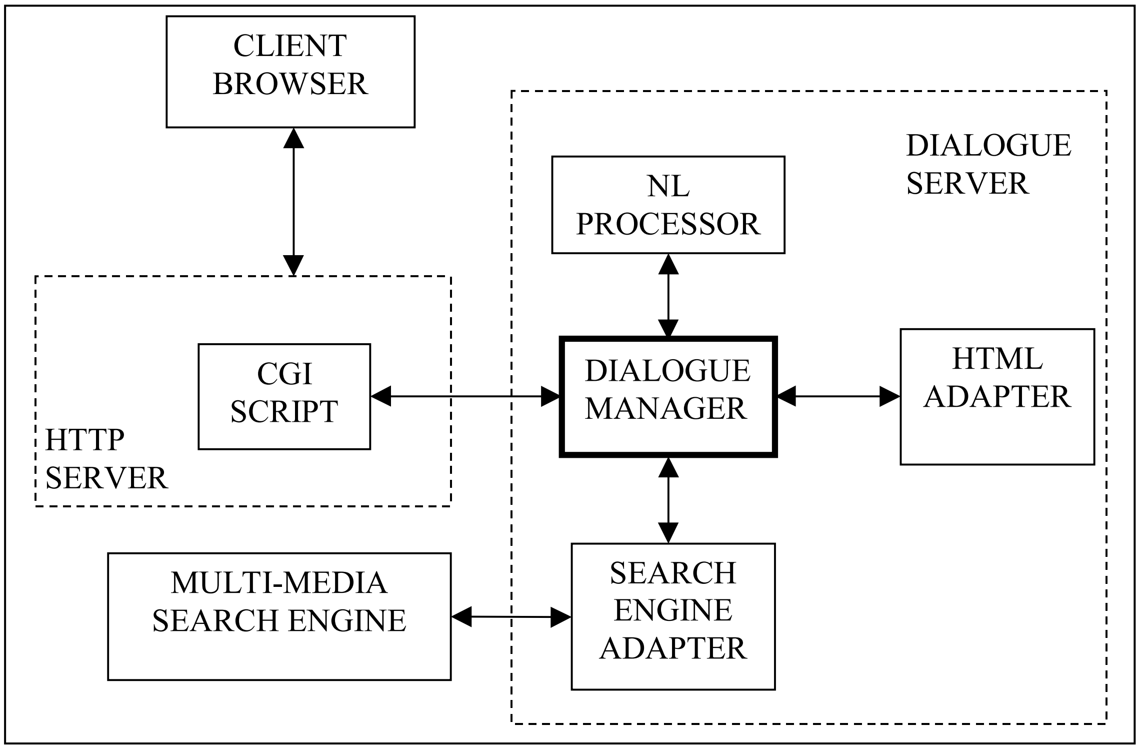


Figure 4: Dialogue system architecture

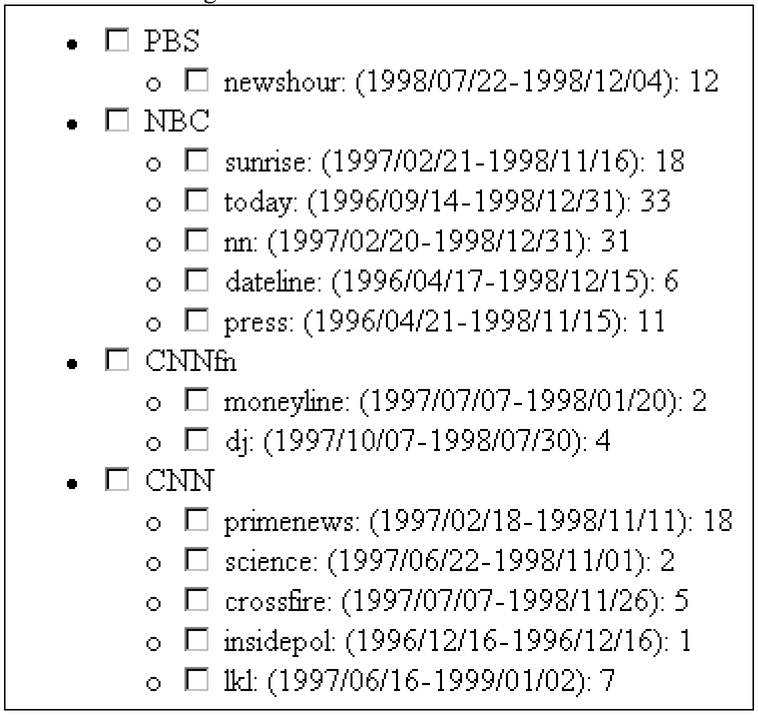
Natural Language Dialogue Interface for Searching
To navigate a large database of video programs, a natural language dialogue system is appropriate. Without this, we must resort to complicated form-filling interfaces that are not well suited to the TV usage paradigm. Figure 4 shows the architecture of the Natural Language Dialogue Interface. A user, through a client browser, sends a natural language request plus some context information from previous interactions (i.e. the dialogue state) to the HTTP server. The HTTP server forks a CGI script that connects (through TCP/IP protocol) to the dialogue server. The dialogue server, after analyzing the sentence and the dialogue state returns the proper answer in HTML format in addition to the updated dialogue state information that will be stored in the client browser as hidden fields.

The dialogue manager executes the strategy of the dialogue [Pieraccini97], i.e. depending on the current dialogue state it invokes the appropriate action out of a finite set of predetermined actions. The list of actions includes getting information from the input, merging information from the current user's input with the context store in the dialogue state, sending information to the output, generating a question for the user, or invoking one of the task specific auxiliary processors, such as the natural language (NL) processor, the HTML adapter or the search engine adapter.

The NL processor takes a sentence in input and generates its semantic description in terms of keyword/value pairs. For instance, given the sentence: *What do you have about John Glenn from NBC during the past three weeks* as input, the NL processor would generate the following semantic representation: **query: LIST, topic: John Glenn, network: NBC, time: -3w.** , The NL processor is based on the stochastic conceptual paradigm [Pieraccini93.]. Stochastic n-gram grammars represent portions of the sentence (i.e. concepts) that have associations to defined conceptual units. Some of them are task independent, such as **query** (e.g. *I'd like to know*, *Please show me*, *What do you have about*, etc.), **time** (e.g. *during the past three months, between January and March this year, yesterday,* etc.*),* and were automatically estimated on corpora collected from different applications. Others are task dependent, such as **network** (e.g. by *NBC*, *on CNN*, etc.), and were initially handcrafted and successively improved as more examples were collected with the system.

The search engine adapter converts the key/value semantic representation of the current query into the format required by the search engine, submits the query, gets the results and stores them in the current dialogue state in the form of a list of data tuples. The search engine includes an inverted index of the closed caption text from television news programs and a relational database of program attributes such as the network, show title, and broadcast date and time.

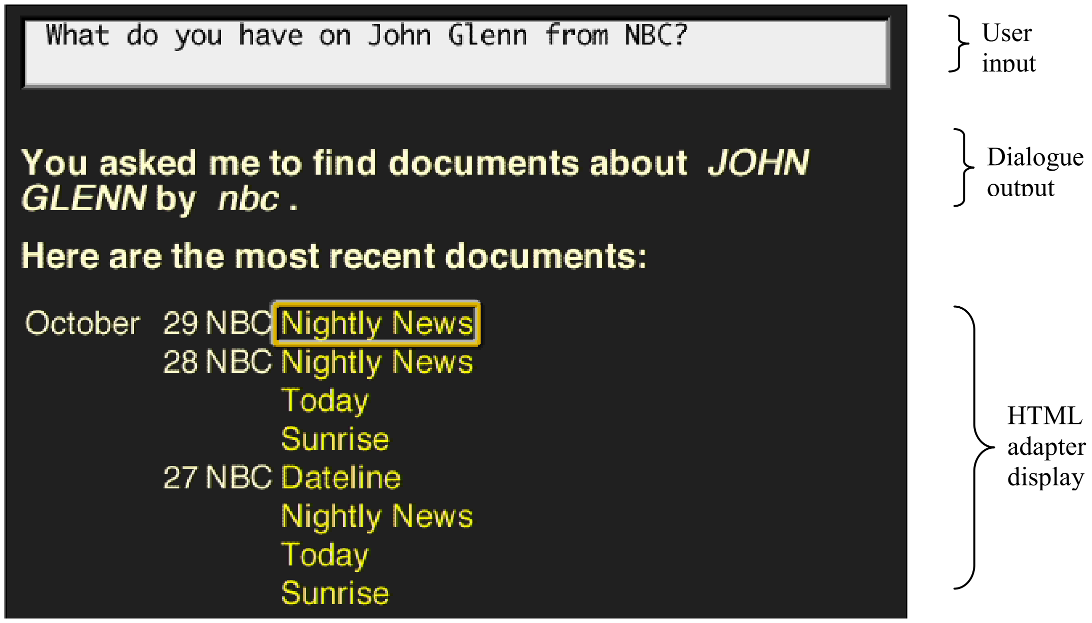
The dialogue strategy has been designed in order to take advantage of several query types supported by the multimedia search engine with the goal of improving performance, and minimize network traffic. For example, for a given user's question, the dialogue manager first builds a query to extract only the total number of matching documents (and not document's attributes). If this number is large, the system prompts the user to restrict the query (for instance by asking to select a particular range of time or a network). Only when the number is reasonable is the full request of document attributes sent to the search engine. We found that a better approach is a compromise in which a limited amount of information describing the query results is returned in the initial stage. This information is obtained by computing the frequency of documents grouped by date, network, and show, and is represented as a tree with selectable nodes as in Figure 5. In the figure, each leaf node includes an indication of the number of items available. The user can either select a subset of the nodes and get a restricted set of documents, or refine the query using a natural language sentence (e.g. *The documents by NBC and CBS between January and February*). When the number of documents returned by a query is reasonable (in our case less than 20), a compact tree representation of the features of the retrieved documents is displayed with hyperlinks to the actual documents, as in Figure 6. The complete user interface is shown in Figure 7.



- ☐ PBS
  - ☐ newshour: (1998/07/22-1998/12/04): 12
- ☐ NBC
  - ☐ sunrise: (1997/02/21-1998/11/16): 18
  - ☐ today: (1996/09/14-1998/12/31): 33
  - ☐ nn: (1997/02/20-1998/12/31): 31
  - ☐ dateline: (1996/04/17-1998/12/15): 6
  - ☐ press: (1996/04/21-1998/11/15): 11
- ☐ CNNfn
  - ☐ moneyline: (1997/07/07-1998/01/20): 2
  - ☐ dj: (1997/10/07-1998/07/30): 4
- ☐ CNN
  - ☐ primenews: (1997/02/18-1998/11/11): 18
  - ☐ science: (1997/06/22-1998/11/01): 2
  - ☐ crossfire: (1997/07/07-1998/11/26): 5
  - ☐ insidepol: (1996/12/16-1996/12/16): 1
  - ☐ lkl: (1997/06/16-1999/01/02): 7

Figure 5: Query results represented as a tree with selectable nodes

1999	January	2	CNN	Larry King Live
1998	December	31	NBC	Nightly News
				Today
		25	NBC	Today
		15	NBC	Dateline
		4	PBS	The News Hour with Jim Lehrer
		3	CNN	Larry King Live
	November	26	CNN	Crossfire
		17	NBC	Today
		16	PBS	The News Hour with Jim Lehrer
			NBC	Nightly News
				Today
				Sunrise
		15	NBC	Meet The Press
		14	NBC	Nightly News
		12	CNN	Larry King Live
			NBC	Today
				Sunrise
		11	CNN	Prime News
		6	NBC	Nightly News


What do you have on John Glenn from NBC?

You asked me to find documents about *JOHN GLENN* by *nbc* .

Here are the most recent documents:

```
October  29 NBC Nightly News
         28 NBC Nightly News
                Today
                Sunrise
         27 NBC Dateline
                Nightly News
                Today
                Sunrise
```

Figure 6: Compact tree with links to actual documents

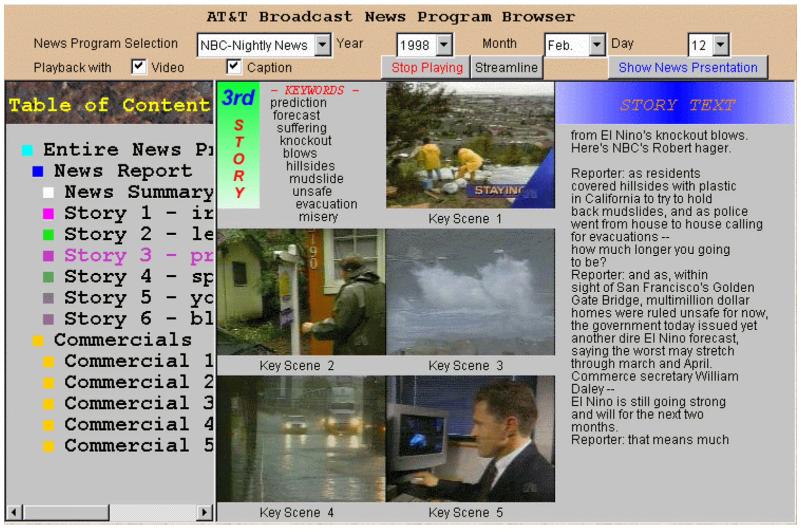
Figure 7: Multimodal Dialogue interface designed for TV resolution displays

Story Rendering for Browsing

To provide a concise and informative content based browsing interface, we construct visual presentations for the semantics at different levels of abstraction. For example, at a news program level, a table of contents coupled with a timeline representation can be automatically formed [Huang99a.] The former provides a conceptual summary of the content within the news program while the latter offers a physical layout of the content along time. At news story level, a visual presentation is formed for each headline story in a most content revealing manner. A small number of *k* images are automatically selected from a set of key frames derived using content based sampling (see below) based on the histogram of keywords chosen in terms of their importance. This is determined by analyzing term frequencies in a news corpus collected from NBC Nightly News in 1997 [Hunag99b.] Specifically, if *H(t)* denotes the histogram of the chosen keywords within the story boundary and *A(t)* denotes the area within the histogram covered under [*H(t-1), H(t+1)*], *k* images are chosen so that $\sum_{i \in k} A(t_i)$ is maximized. Images belonging to segments containing the anchorperson are not included in the set since they typically do not convey a large amount of visual information.

As mentioned above, a content based sampling algorithm is used to select key frames from the video stream. Several algorithms are available for this purpose (for example [Zhang93].) The goal of these algorithms is to minimize the number of frames retained while maximizing the extent to which the images convey the visual information of the video. We have used an efficient algorithm that detects not only shot cuts, but also detects camera-induced changes (e.g. pan and tilt) as well as gradual transitions such as fades, dissolves and wipes [Shahrary95a]. Another good option for systems designed for MPEG2 indexing is the class of algorithms that operate in the compressed domain (e.g. [Meng95].)

In applications with high-resolution terminals, the retained images, keywords, and closed caption itself are simultaneously displayed to represent the underlying news story (e.g. Figure 8.) While this interface displays a large amount of information, it clearly cannot be reproduced in low-resolution terminal applications.



from El Nino's knockout blows.
Here's NBC's Robert hager.

Reporter: as residents
covered hillsides with plastic
in California to try to hold
back mudslides, and as police
went from house to house calling
for evacuations --
how much longer you going
to be?
Reporter: and as, within
sight of San Francisco's Golden
Gate Bridge, multimillion dollar
homes were ruled unsafe for now,
the government today issued yet
another dire El Nino forecast,
saying the worst may stretch
through march and April.
Commerce secretary William
Daley --
El Nino is still going strong
and will for the next two
months.
Reporter: that means much

Figure 8: Story representation suitable for high-resolution terminals (from [Huang99a].)

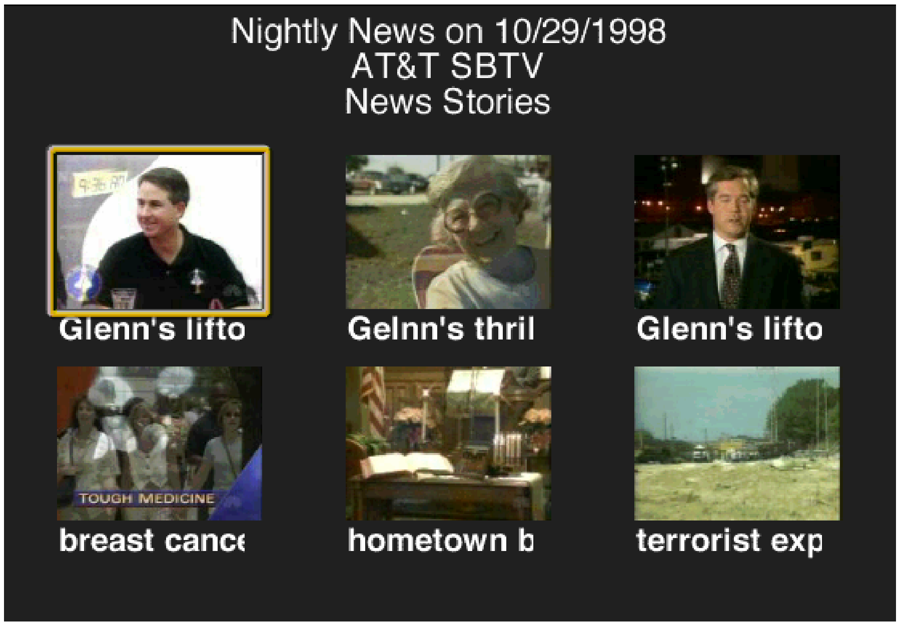


Figure 9: A sample table of contents page

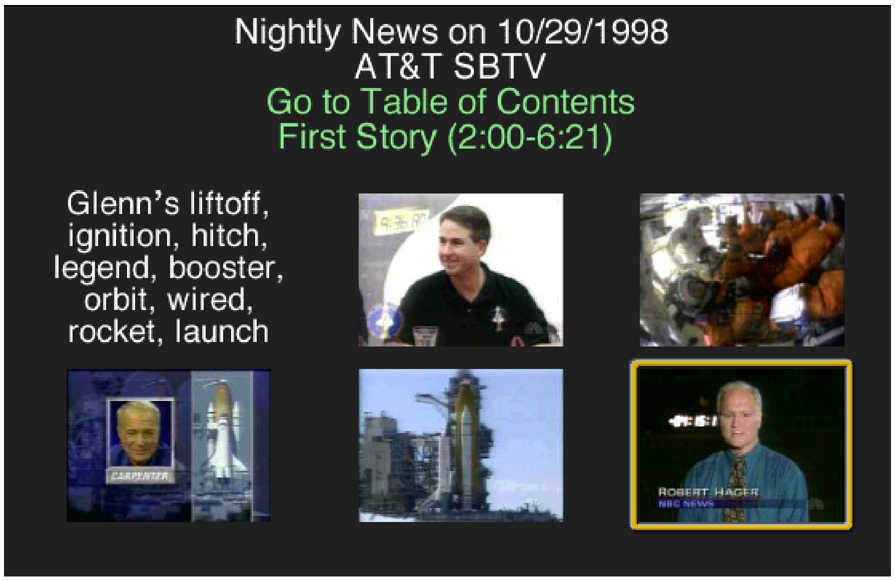

Nightly News on 10/29/1998
AT&T SBTV
Go to Table of Contents
First Story (2:00-6:21)
Glenn's liftoff, ignition, hitch, legend, booster, orbit, wired, rocket, launch
CARPENTER
ROBERT HAGER
NBC NEWS

Figure 10: A sample story page

Two views of the user interface designed for television resolution are shown in Figure 9 and Figure 10. Once the user has navigated down to the story level (Figure 10,) he or she can play the full story, or choose to start the playback from other points by selecting one of the story images. When video playback is initiated, the user's display switches to a full screen playback mode. When the video playback is stopped, the story page is brought back into view, and another segment can be selected or a new navigation session can be initiated. Figure 11 depicts this process. The "media specification generator" takes input parameters via the CGI from the browser indicating the desired media segment. Standards in this area are evolving, but not yet mature [Hoschka98, Kate98a, Kate98b.] The format of the specification can change depending upon the video storage subsystem. For example, in one of the implementation of the system we have used the ASX format with Microsoft's Theater server. The main drawback of this approach is the long start-up delay. We have developed our own system in which SDP is used for the media specification. With this system, we obtain better control of error resilience, and we adopt a more open, standards based architecture.

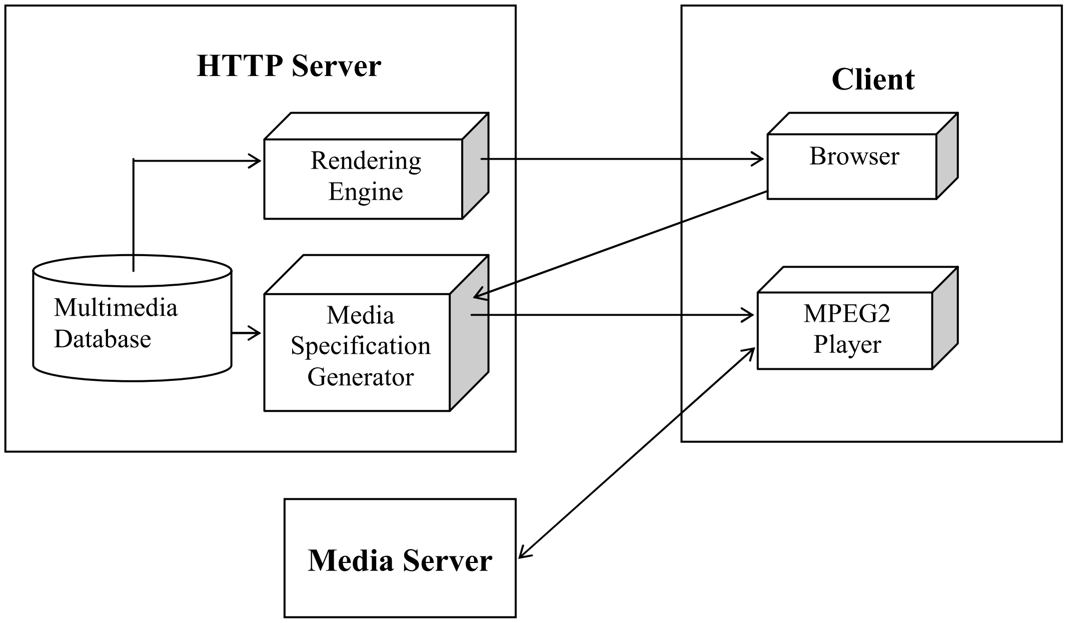


Figure 11: Client and server architecture for media playback

MPEG-2 Media Control and Streaming

Streaming video is a low cost solution that allows bringing live or prerecorded information to customers, employees and business partners worldwide across corporate Intranets and soon over the internet. There are several digital compression formats available today. MPEG-2 is a standard, formally accepted in 1994, developed by ISO/IEC/JTC1/SC29/WG11 and known in literature as ISO/IEC 13118 [ISO94]. It provides a set of methodologies to perform compression as well as the transport and multiplexing of video and audio. Thanks to the MPEG2 home video is currently available on DVD and it is the only standard that can provide comparable and superior quality that consumers and businesses expect to see on their TV, computer screens or during mission critical multimedia business conferences.

Among the recently developed multimedia protocol the IETF Real Time Protocol (RTP) is gaining attention from the network and multimedia community. RTP is designed to deliver various kinds of real time data over packet networks including non-guaranteed quality-of-service (QoS) networks The services provided by the RTP [Schulzrinne96, Yajnik96] include payload type identification, sequence numbering, time stamping and delivery monitoring. RTP typically runs on top of User Datagram Protocol (UDP) [Comer91] to make use of its multiplexing and checksum services. RTP supports data transfer to multiple destinations using multicast distribution if this functionality is supported by the underlying network as in TCP/IP networks. RTP addresses the needs of real-time data transmission only and relies on other well-established network protocols for other communications services such as routing, multiplexing and timing. Additionally, RTP is based on the Application Level Framing (ALF) and Integrated Layer Processing (ILP) [Clark90, Speer96] principles, which dictate using the properties of the payload in designing a data transmission system as much as possible. For example, if we know that the payload is MPEG video, we should design our packetization scheme based on “slices” because they are the smallest independently decodable data units for MPEG video. This approach provides a much more suitable framework for MPEG-2 transmission over networks with high packet loss rates. RTCP is the companion control protocol that offers in the current specification some congestion control and receiver quality metric.

The Real Time Streaming protocol (RTSP) [Shulzrinne98] is an application-level protocol developed by IETF that provides the framework to allow on-demand delivery of real-time data streams, such audio and video, in stored format or live. RTSP allows for the control of multiple data delivery sessions, the selection of the delivery channels such as multicast or unicast UDP, TCP and provides means for the selection of the delivery mechanisms based on RTP.

Media preprocessing: The RTP Packetizer

An offline RTP packetizer compliant with [Hoffman98] packetizes MPEG-2 elementary video and audio streams. It supports different payload filling techniques where the number of slices per packet can be constant or variable to fill the MTU. The packetizer can implement different strategies for the repetition of the RTP’s MPEG specific video header extension in the RTP packets generated from a given frame. Note that the overhead introduced by the use of the RTP specific video header extension can vary from a few to tens of Kb/s depending on the encoding parameters, and the presence of quantization, picture display, and picture temporal and spatial scalable extensions and the packetization strategy used. In this implementation, the MPEG specific header extension is repeated in all the packets of a given frame. The overhead for 2 slices per packet for a 4Mb/s MPEG-2 video stream was approximately 6Kb/s.

Media Server

The video/audio streaming system implements a client/server for the real-time delivery of MPEG-2 encoded audio/video streams over IP based on the Real Time Transport Protocol (RTP) and RTSP [Shulzrinne98]. The entire system is designed using off-the-shelf hardware and commercial operating systems. We emphasized a cost-effective design that can survive under extremely high packet losses and deliver acceptable quality audio and video under packet loss rates as high as 10^{-1}.

The building blocks of our architecture are shown in Figure 12. The server is composed of two separate blocks, the media control and the streamer modules. The media control module is based on RTSP and has the task of setting up the connection and the media streams in response to a request from a client.

The streamer instead once instantiated, reads from a file the media data pre-assembled in RTP packets, it schedules and delivers the RTP packets over the network. The scheduling algorithm assigns a delivery time to every packet according to a scheduling policy. The policy we used targets matching the encoder's average rate. Furthermore, the scheduler measures the delivery time jitter due to the timer and rearranges the packet scheduling accordingly, see [Basso97] for further details.

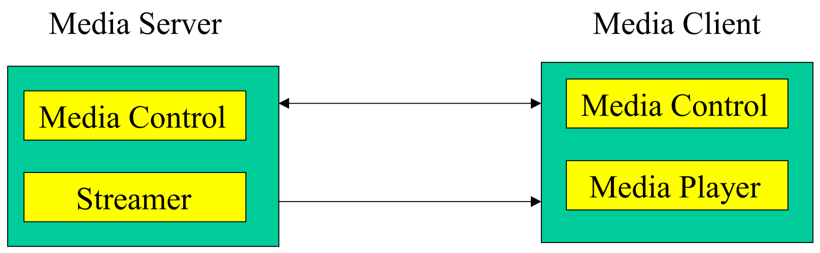


Figure 12: Media Server and Client Architecture

Media Client
The media client is composed of two modules: the Media Control and the Media Player module. The media control module is based on RTSP and operates as a remote VCR control that allows to setup, start, stop and teardown of the audio/video streams. For the representation of a given clip we use RTSP container files which are a storage entities in which multiple continuous media types pertaining to the same end-user presentation are present. Container files are a widely used means to store such presentations. Typically the media components of the presentation are transported as independent streams and the container file maintains a common context for those streams at the server end. Thus the server can keep a single storage handle open. In this application all the streams of a given clip are controlled via a single control message using an aggregate URL [Shulzrinne98].

The following is an example of an RTSP session between the Media Server `M` and the Media Client `C` to control MPEG-2 video and MPEG-1 audio streams of the presentation of a video/audio clip on J. Glenn. The clip description is stored in a container file. The client has obtained an RTSP URL to the container file from the HTML page. (For illustrative purposes we assume here that the video is stored in clip files matched to story boundaries, and that the container files are static. For maximum flexibility, we actually store larger video files, and generate container files dynamically as described above and shown in Figure 11.) In a first phase the client obtains a description in SDP [Handley98] format of the clip from the server.

```
C->M: DESCRIBE rtsp://sbtv.research.att.com/glenn RTSP/1.0
       CSeq: 1
 M->C: RTSP/1.0 200 OK
       CSeq: 1
       Content-Type: application/sdp
       Content-Length: 164
       v=0
       o=- 2890844256 2890842807 IN IP4 135.207.130.108
       s=SBTV RTSP Session
       i= J.Glenn Clip
       a=control:rtsp://sbtv.research.att.com/glenn
       t=0 0
       m=audio 0 RTP/AVP 14
       a=control:rtsp://sbtv.research.att.com/glenn/audio
       m=video 0 RTP/AVP 32
       a=control:rtsp://sbtv.research.att.com/glenn/video
```

The client then issues a SETUP and specifies the client ports to use for the delivery of every media involved in the clip. In our case an MPEG-2 video and MPEG-1 audio.

```
C->M: SETUP rtsp://sbtv.research.att.com/glenn/audio RTSP/1.0
      CSeq: 2
      Transport: RTP/AVP;unicast;client_port=8000-8001
M->C: RTSP/1.0 200 OK
      CSeq: 2
      Transport: RTP/AVP;unicast;client_port=8000-8001;
                 sbtv.research.att.com_port=9000-9001
      Session: 12345678
C->M: SETUP rtsp://sbtv.research.att.com/glenn/video RTSP/1.0
      CSeq: 3
      Transport: RTP/AVP;unicast;client_port=8002-8003
      Session: 12345678
M->C: RTSP/1.0 200 OK
      CSeq: 3
      Transport: RTP/AVP;unicast;client_port=8002-8003;
                 sbtv.research.att.com_port=9004-9005
      Session: 12345678
```

The client finally issues a request to the server to stream the video and audio starting from the beginning (Range: npt=0-)

```
C->M: PLAY rtsp://sbtv.research.att.com/glenn RTSP/1.0
      CSeq: 4
      Range: npt=0-
      Session: 12345678

M->C: RTSP/1.0 200 OK
      CSeq: 4
      Session: 12345678
      RTP-Info: url=rtsp://sbtv.research.att.com/glenn/video;
        seq=9810092;rtptime=3450012
```

During the presentation the user pauses the stream to look for something else:

```
C->M: PAUSE rtsp://sbtv.research.att.com/glenn RTSP/1.0
      CSeq: 6
      Session: 12345678
M->C: RTSP/1.0 200 OK
      CSeq: 6
      Session: 12345678
```

The media player module is based on a multithreaded architecture and it is responsible for converting the RTP packets into MPEG-2 elementary streams, handling the delay jitter and packet losses, inter- and intra-media synchronization, timely delivery of the bitstreams to the audio and video decoders, which are PC plug-in boards.

The packet capture is performed via asynchronous notification as in the server. At startup, all incoming RTP packets are discarded until the start of an I frame is detected by checking the RTP Picture Type field in the video specific header. In our implementation, the sequence header is repeated for every I frame so, the decoding is started as soon as an I frame is detected. The packet losses are detected from the discontinuities in the RTP sequence numbers. It is possible to distinguish between losses that effect just a part of the same picture and packet losses covering more than one picture by comparing the timestamp of the previous packet with the current one. If they are equal, the loss is contained within a frame. If not, the loss spans across one or more pictures. When the encoder is operating at a fixed frame rate, it is also possible to estimate the number of lost pictures by computing the difference between the last and the current timestamp. The GOP and picture header loss detection is performed as discussed in [Hoffman98, Basso97]. Picture headers and corresponding extensions are regularly stored in the client memory and properly updated.

The concealment of lost slices is performed by inserting dummy slices in their place that instruct the hardware decoder to display the corresponding slices from the previous picture. The consistency of the reconstructed frame is checked before it is delivered to the hardware decoder. The status of the hardware decoder is constantly monitored to provide support for potential decoding errors that can arise after error concealment also. The performance of the error detection and concealment techniques have been tested by simulating bursty and isolated losses according to the experimental data presented in [Yajnik96] for the MBONE. We were able to conceal packet loss rates up to 10^{-1} with visible degradation while rates of 10^{-3} seemed to be acceptable for both audio and video.

Conclusions and Future Work

We have presented a method and system for searching, browsing, and retrieval of broadcast quality video over IP networks. The target appliance for this application is the advanced home television. We have described methods to work around the limitations imposed by these low-resolution devices. These include a dialogue system and a story segmentation method that facilitate searching and browsing of large multimedia databases. We have presented a system for delivery of MPEG2 streams over IP networks that is standards compliant (e.g. it used RTP, RTSP, and SDP.)

While the user interface presented is serviceable, there is room for improvement. In the future, we hope to include more media elements (such as icon images) during the dialogue-based searching phase. Our work on story segmentation must be extended to include domains other than broadcast news. We are investigating dynamic training for speaker segmentation to improve upon our model-based anchorperson identification.

References

[Basso97] A. Basso G. L. Cash, M.R. Civanlar "Transmission of MPEG-2 Streams over non-Guaranteed quality of Service Networks", PCS-97 10-12 Sept 97.

[Choi99] Spoken Document Retrieval. Proceedings of the Broadcast News Transcription and Understanding Workshop (BNTUW'98) (to appear).

[Clark90] Clark, D., and Tennenhouse, D. " Architecture Considerations for a New Generation of Protocols," Proc. of ACM SIGCOM '90, Sept. 90,pp.201-208.

[Comer91] D. E. Comer, "Internetworking with TCP/IP," ISBN 0-13-468505-9, Prentice-Hall, NJ, 1991.

[Excalibur98] Excalibur, http://www.excalibur.com

[FasTV98] FasTV, http://www.fastv.com

[Fleischman98] Eric Fleischman, "Advanced Streaming Format (ASF) Specification," *Internet Draft*, February 26, 1998. http://www.microsoft.com/asf/resources/draft-ietf-fleischman-asf-01.txt

[Gibbon98] D. Gibbon, "Generating Hypermedia Documents from Transcriptions of Television Programs Using Parallel Text Alignment", in *Handbook of Internet and Multimedia Systems and Applications,* Edited by Borko Furht, CRC Press, December 1998, ISBN 0849318580.

[Handley98] Handley V. Jacobson " SDP: Session Description protocol", RFC 2327 April 1998

[Hoffman98] D. Hoffman, G. Fernando, V. Goyal and M. R. Civanlar, "RTP Payload Format for MPEG1 / MPEG2 Video " RFC2055, Internet draft, 1998.

[Hoschka98] Philipp Hoschka, W3C TVWeb Interest Group, http://www.w3.org/TV/TVWeb/

[Huang99a] Qian Huang, Zhu Liu, Aaron Rosenberg, "Automated semantic structure reconstruction and representation generation for broadcast news," in *Storage and Retrieval for Image and Video Databases,* Proc. SPIE 3656, (1999).

[Huang99b] Qian Huang, Zhu Liu, Aaron Rosenberg, David Gibbon, Behzad Shahraray, "Automated Generation of News Content Hierarchy By Integrating Audio, Video, and Text Information," *Proc. IEEE International Conference On Acoustics, Speech, and Signal Processing,* Phoenix, Arizona, March 15-19, 1999.

[Islip98] Informedia, http://www.islip.com

[ISO94] ISO/IEC International Standard 13818, "Generic coding of moving pictures and associated audio information," November 1994.

[Kate98a] Warner ten Kate, "TV URI Schemes – Requirements," November 10, 1998. http://www.w3.org/TV/TVWeb/TVWeb-URI-Requirements-19981110

[Kate98b] Warner ten Kate, G. Thomas, C. Finseth, "Requirements TV Broadcast URI Schemes," *Internet Draft*, November, 1998. http://search.ietf.org/internet-drafts/draft-tenkate-tvweb-uri-reqs-00.txt

[Magnifi98] Magnifi, Inc., http://www.magnifi.com

[MPG7] MPEG-7 Context and Objectives, Requirements Group, ISO/IEC JTC1/SC29/WG11 http://drogo.cselt.stet.it/mpeg/standards/mpeg-7/mpeg-7.htm

[Meng95] J. Meng, Y. Juan, and S.F. Chang, "Scene Cange Detection in a MPEG Compressed Video Sequence," in *Digital Image Compression: Algorithms and Technologies,* Proc. SPIE 2419, 1995.

[Pieraccini97] R. Pieraccini, E. Levin, and W. Eckert, "AMICA, the AT&T Mixed Initiative Conversational Architecure," *Proc. EUROSPEECH 97*, September 1997, Rhodes, Greece.

[Pieraccini93] R. Pieraccini and E. Levin, "A learning approach to natural language understanding," NATO-ASI, *New Advances & Trends in Speech Recognition and Coding,* Springer-Verlag, Bubion (Granada), Spain, 1993.

[Real98] Real Networks, http://www.real.com

[Rosenberg98] A. E. Rosenberg, I. Magrin-Chagnolleau, S. Parthasarathy, and Q. Huang, "Speaker detection in broadcast speech databases," in *Proc. of International Conference on Spoken Language Processing*, Sydney, November 1998.

[Shahraray95a] B. Shahraray, "Scene Change Detection and Content-Based Sampling of Video Sequences," in *Digital Image Compression: Algorithms and Technologies,* Proc. SPIE 2419, 1995.

[Shahraray95b] B. Shahraray and D. Gibbon Automated Authoring of Hypermedia Documents of Video Programs, *Proc. Third Int. Conf. on Multimedia (ACM Multimedia'95)*, San Francisco, CA, November, 1995.

[Shahraray97] B. Shahraray and D. Gibbon, Pictorial Transcripts: Mulitmedia Processing Applied to Digital Library Creation, *IEEE Workshop on Multimedia Signal Processing*, Princeton, NJ, 1997, pp. 581-586.

[Schulzrinne96] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, January 1996.

[Schulzrinne98] Schulzrinne, et al "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998.

[Smith97] Smith, M., Kanade.T., "Video Skimming and Characterization through the Combination of Image and Language Understanding Techniques." Computer Vision and Pattern Recognition Conference, San Juan, Puerto Rico. Pp. 775-781, June 1997. http://informedia.cs.cmu.edu/documents/CVPR97_smith.pdf

[Speer96] M. F. Speer, S. McCanne, "RTP Usage with Layered Multimedia Streams," Internet Draft, draft-speer-avt-layered-video-02, December 1996.

[Virage98] Virage, http://www.virage.com

[Wactlar98] H. Wactlar, Informedia, Researching Digital Video Libraries, 1998. http://www.informedia.cs.cmu.edu

[WebTV98] WebTV Design Guide, WebTV Networks, 1998. http://developer.webtv.net/docs/designguide

[Yajnik96] M. Yajnik, J. Kurose, D. Towsley "Packet Loss Correlation in MBONE Multicast network" *IEEE Global Internet mini-conference* GLOBECOM'96, 1996.