MULTIMEDIA VIDEO

With advances in computer technology, digital video is becoming more and more common in various aspects of life, including communication, education, training, entertainment, and publishing. The result is massive amounts of video data that already exist in digital form, or will soon be digitized. According to an international survey (1), there are more than six million hours of feature films and video archived worldwide, with a yearly rate of increase about 10%. This would be equal to 1.8 million Gbyte of MPEG-encoded digital video data if all these films were digitized. The unique characteristics of video data (2,3) and its differences from other data types can be summarized in Table 1.

Both video and image data contain much more information than plain textual data. The interpretation of the video and image data is thus ambiguous and dependent on both the viewer and the application. By contrast, textual data usually have a limited and well-defined meaning. Textual data are neither spatial nor temporal and can be thought of as one-dimensional data. Image data, however, contain spatial but not temporal information and can be regarded as two-dimensional data. Video data, on the other hand, have a third dimension—that is, time. Compared to traditional data types like textual data, video and image data do not have a clear underlying structure and are much more difficult to model and represent. One single image is usually of the magnitude of kilobytes of data volume, and 1 min of full motion video data contains 1800 image frames. The data volume of the video data is said to be about seven orders of magnitude larger than a structured data record (3). Relationship operators such as equal defined for textual data are simple and well-defined. However, the relationships between video (or image) data are very complex and ill-defined. This causes many problems for video data indexing, querying, and retrieval. For example, there is no widely accepted definition of a simple similarity operator between two images or video streams.

Identifying the rich information content or features of video data helps us to better understand the video data, as well as to (a) develop data models to represent, (b) develop indexing schema to organize, and (c) develop query processing techniques to access them. The video data content can be classified into the following categories (2,3):

- *Semantic content* is the idea or knowledge it conveys to the user. It is usually ambiguous and context-dependent. For example, two people can watch the same TV program and yet have different opinions about it. Such semantic ambiguity can be reduced by limiting the context or the application.
- *Audiovisual content* includes audio signal, color intensity and distribution, texture patterns, object motions and shapes, and camera operations, among many others.
- *Textual content* provides important metadata about the video data. Examples are the closed caption of a news video clip, the title of the video clip, or actors and actresses listed at the beginning of a feature film.

It should be pointed out that various contents of video data are not equally important. The choice and importance of the features depend on the purpose and use of the video data. In an application such as animal behavior video database management system (VDBMS), the motion and shape information of objects (in this case, animals) is the most important content of the video data. There may also be additional meta information, which is usually application specific and cannot be obtained directly from the video data. Such information is usually added during the annotation step of inserting video data into the video database (VDB)—for example, background information about a certain actor in a feature film video database.

Table 1. Comparison of Video Data with Other Types of Data

Criteria	Textual Data	Image Data	Video Data
Information	Poor	Rich	Very rich
Dimension	Static and nonspatial	Static and spatial	Temporal and spatial
Organization	Organized	Unstructured	Unstructured
Volume	Low	Median	Massive
Relationship	Simple and well-defined	Complex and ill-defined	

VIDEO CODECS

One of the problems faced in using digital video is the huge data volume of video streams. Table 2 shows the data rate of some standard representations of uncompressed digital video data that are obtained by sampling the corresponding analog video stream at certain frequencies.

1. NTSC stands for National Television System Committee; it has image format 4 : 3, 525 lines, 60 Hz, and 4 MHz video bandwidth with a total of 6 MHz of video channel bandwidth. *YIQ* is the color space standard used in NTSC broadcasting television system. The *Y* signal is the luminance. *I* and *Q* are computed from the difference between the color red and the luminance and the difference between the color blue and luminance. The transformation from *RGB* to *YIQ* is linear (5):

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

   NTSC-1 was set in 1948. It increased the number of scanning lines from 441 to 525 and replaced AM-modulated sound with FM. The frame rate is 30 frames/s.
2. PAL stands for phase alternating line and was adopted in 1967. It has image format 4 : 3, 625 lines, 50 Hz, and 4 MHz video bandwidth with a total of 8 MHz of video channel width. The frame rate is 25 frames/s. PAL uses *YUV* for color coding. *Y* is the same as in *YIQ*. $U = 0.493 \times (B - Y)$ is the function of the difference between red and luminance, and $V = 0.877 \times (R - Y)$ is the function of the difference between blue and luminance.
3. SECAM stands for Sequential Coleur à Memoire; it has image format 4 : 3, 625 lines, 50 Hz, and 6 MHz video bandwidth with a total 8 MHz of video channel bandwidth. The frame rate is 25 frames/s.
4. CCIR standards for Comité Consultativ International de Radio, which is part of the United Nations International Telecommunications Union (ITU) and is responsible for making technical recommendations about radio, television, and frequency assignments. The CCIR 601 digital television standard is the base for all the subsampled interchange formats such as source input format (SIF), common intermediate format (CIF), and quarter CIF (QCIF). For NTSC (PAL/SECAM), it is 720 (720) pixels by 243 (288) lines by 60 (50) fields/s where the fields are interlaced when displayed. The chrominance channels horizontally subsampled by a factor of two, yielding 360 (360) pixels by 243 (288) lines by 60 (50) fields/s. CCIR 601 uses YC_rC_b color space which is a variant of YUV color space.

Clearly, video streams must be compressed to achieve efficient transmission, storage, and manipulation. The bandwidth will become even more acute for high-definition television (HDTV) since uncompressed HDTV video can require a data rate of more than 100 Mbyte/s. The term *video codec* is a combination of the words video *co*mpression and *dec*ompression which express its two major functions: (i) compress the video, and (ii) decompress the video during playback. Video codecs can be either hardware or software. Software codecs is slower than hardware codecs, but is more portable and cost-efficient. Video can be compressed both spatially and temporally. Spatial compression applies to a single frame and can be lossy or lossless. Temporal compression is to identify and store the infraframe difference. Both can be used in a video codec.

Most extant video compressions standards are *lossy video compression algorithms.* In other words, the decompressed result is not totally identical to the original data. This is so because the compression ratio of lossless methods [e.g., Huffman, Arithmetic, Lempel-Ziv-Welch (LZWs)] is not high enough for video compression. On the other hand, some lossy video compression techniques such as MPEG and MJPEG can

Table 2. Data Rate of Uncompressed Digital Video

Video Standard	Image Size	Bytes/Pixel	Mbyte/s
NTSC square pixel (USA, Japan, etc.)[a]	640 × 480	3	27.6
PAL square pixel (UK, China, etc.)[b]	768 × 576	3	33.2
SECAM (France, Russia, etc.)[c]	625 × 468	3	22.0
CCIR 601(D2)[d]	720 × 486	2	21.0

[a] NTSC, National Television System Committee.
[b] PAL, phase alternating line.
[c] SECAM, Sequential Coleur à Memoire.
[d] CCIR, Comité Consultative International de Radio.

reduce the video data rate to 1 Mbyte. Lossy compression algorithms are very suitable for video data, since not all information contained in video data is equally important or perceivable to the human eye. For example, it is known that small changes in the brightness are more perceivable than changes in color. Thus, compression algorithms can allocate more bits to luminance information (brightness) than to the chrominance information (color). This leads to lossy algorithms, but people may not be able to see the data loss. One important issue of a video compression scheme are the tradeoffs between compression ratio and video quality. "Higher quality" implies smaller compression ratio and larger encoded video data. The speed of a compression algorithm is also an important issue. A video compression algorithm may have a larger compression ratio, but it may not be usable in practice due to the high computational complexity and because real-time video requires a 25 fields/s decoding speed.

There are many video compression standards including the CCITT/ISO standards, Internet standards, and various proprietary standards based on different tradeoff considerations. Some codecs takes lots of time to compress a video, but decompress very quickly. They are called *Asymmetric video codecs. Symmetric video codecs* take about the same amount of time in both compression and decompression processes. Video codecs should not be confused with *multimedia architectures.* Architectures, such as Apple's QuickTime (6), are operating system extensions or plug-ins that allow the system to handle video and other multimedia data such as audio, animation, and images. They usually support certain codecs for different media including video. We include QuickTime as an example here.

JPEG AND MJPEG

JPEG is a standardized image compression mechanism. JPEG stands for Joint Photographic Experts Group, the original name of the committee that wrote the standard. It is designed for lossy compression of either full-color or gray-scale still images of natural, real-world scenes. It works well on photographs, naturalistic artwork, and similar images; however, it does not work so well on lettering, simple cartoons, or line drawings. JPEG exploits known limitations of the human eye, notably the fact that small color changes are perceived less accurately than small changes in brightness on which MPEG is also based. Thus, JPEG is intended for compressing images that will be viewed by human beings. If images need to be machine-analyzed, the small errors introduced by JPEG may pose a problem, even if they are invisible to the human eye. A useful property of JPEG is that the degree of loss of information can be varied by adjusting compression parameters. This means that the image maker can trade off file size against output image quality. Another important aspect of JPEG is that decoders can also trade off decoding speed against image quality by using fast, though inaccurate, approximations to the required calculations.

JPEG is a symmetric codec ($\approx$1:1), and its image coding algorithm is the basis of spatial compression of many video codecs such as MPEG and H.261. The coding process (shown in Fig. 1) involves the following major steps:

1. RGB to YIQ transformation (optional) followed by DCT (discrete cosine transformation).

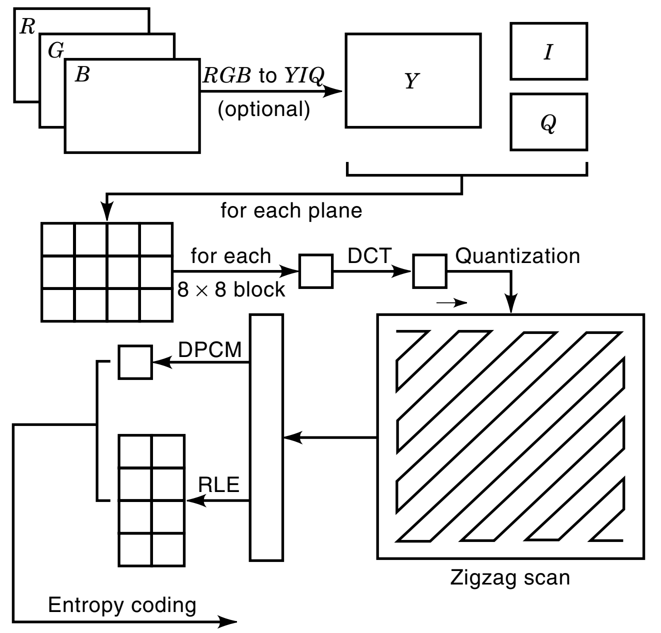

Figure 1. JPEG image coding.

2. Quantization, which is the main source of the lossy compression. JPEG standard defines two default quantization tables, one for the luminance and one for chrominance.
3. Zigzag scan to group the low-frequency coefficients on top of a vector by mapping 8 × 8 to a 1 × 64 vector.
4. DPCM (differential pulse code modulation) on direct-current (dc) component. The dc component is large and varied, but often close to previous values, so we can encode the difference from previous 8 × 8 blocks.
5. RLE (run length encode) on alternating-current (ac) components. The 1 × 64 vector has lots of zeros and can be encoded as (*skip, value*) pairs, where *skip* is the number of zeros and value is the next nonzero component. (0, 0) is used as end-of-block essential value.
6. Entropy coding, which categorizes dc values into SSS (number of bits needed to represent) and actual bits, for example, if dc value is 7, 3 bits are needed. Alternating-current components (skip, value) are encoded as composite symbol (skip, SSS) using Hoffman coding. Hoffman table can be default or custom (contained in the header).

MJPEG stands for motion JPEG. Contrary to popular perception, there is no MJPEG standard. Various vendors have applied the JPEG compression algorithm to individual frames of a video sequence and have called the compressed video MJPEG, but they are not compatible across the vendors. Compared with the MPEG standard, MJPEG is suitable for accurate video editing because of its frame-based encoding (spatial compression only). MJPEG has a fairly uniform bit rate and simpler compression which requires less computation and can be done in real time. The disadvantage of MJPEG is that there is no interframe compression; thus its compression ratio is poorer than that of MPEG (about three times worse).

MJPEG is one of the built-in JPEG-based video codecs in QuickTime.

MPEG

MPEG (7) stands for Moving Pictures Expert Group, which meets under the International Standards Organization (ISO) to generate standards for digital video and audio compression. The official name of MPEG is *ISO/IEC JTC1 SC29 WG11*.

MPEG video compression algorithm is a block-based coding scheme. In particular, the standard defines a compressed bit stream, which implicitly defines a decompressor. However, the choice of compression algorithms is up to the individual manufacturers as long as the bit streams they produce are compliant with the standard, and this allows proprietary advantage to be obtained within the scope of a publicly available international standard. MPEG provides very good compression but requires expensive computations to decompress, which may limit the frame rate that can be achieved with software codec. The current status of MPEG standards is listed in Table 3.

MPEG-1. MPEG-1 (ISO/IEC 11172 standard) defines a bit stream for compressed video and audio optimized to fit into a bandwidth (data rate) of 1.5 Mbit/s which is the data rate of uncompressed audio CDs and DATs. The video stream takes about 1.15 Mbit/s, and the remaining bandwidth is used by audio and system data streams. The data in the system stream provide information for the integration of the audio and video streams with the proper time stamping to allow synchronization. The standard consists of five parts: video (ISO/IEC 11172-1), audio (ISO/IEC 11172-2), systems (ISO/IEC 11172-3), compliance testing (ISO/IEC 11172-4), and simulation software (ISO/IEC 11172-5).

MPEG-1 video coding is very similar to that of MJPEG and H.261. The spatial compression uses a lossy algorithm similar to that of JPEG except that *RGB* image samples are converted to YC_rC_b color space and the C_r and C_b are then subsampled in a 1 : 2 ratio horizontally and vertically. The temporal compression is done using block-based motion compensation with macroblocks (16×16 blocks). Each macroblock consists of 4 Y blocks and the corresponding C_r and C_b blocks. Block-matching techniques are used to find the motion vectors by minimizing a cost function measuring the mismatch between a macroblock and each predictor candidate. MPEG-1 coded video may have four kinds of frames with a typical frame image size of 4.8 kbyte and compress ratio of 27 : 1.

- I frames (intra-coded) are coded without any reference to other frames, i.e., they are only compressed spatially. The typical size of an I frame is 18 kbyte with a compress ratio of 7 : 1.
- P frames (predictive coded) are coded more efficiently by using motion compensation prediction from the previous I or P frame and contain both intra- and forward-predicted macroblocks. The typical size of a P frame is 6 kbyte with a compress ratio of 20 : 1.
- B frames (bidirectionally predictive coded) have the highest compression ratio and require references to both previous and next frames (I and P frames) for motion compensation. B frames have four different kinds of macroblocks: intra-, forward-predicted, and backward-predicted, and bidirectionally-predicted (average). B-frame macroblocks of MPEG-1 can specify both past and future motion vectors indicating the result is to be averaged. The typical size of the B frame is 2.5 kbyte with a compress ratio of 50 : 1.
- D frames (DC-coded) contain only low-frequency information (dc coefficients of blocks) and are totally independent of the rest of data. The MPEG standard does not allow D frames to be coded in the same bitstream as the I/P/B frames. They are intended to be used for fast visible search modes with sequential digital storage media and are rarely used in practice.

MPEG-1 video is strictly progressive—that is, noninterlaced. The quality of the MPEG-1 encoded video is said to be comparable to that of a VHS video. Compared with H.261, MPEG-1 video coding allows larger gaps between I and P frames and thus increases the search range of the motion vectors, which are also specified to a fraction of pixels (0.5 pixel) for better encoding. Another advantage is that the bitstream syntax of MPEG-1 allows random access and forward/backward play. Furthermore, it adds the notion of *slice* for synchronization after loss or corrupted data.

MPEG-2. MPEG-2 (ISO/IEC 11318 standard) includes ISO/IEC 13818-1 (MPEG-2 systems), ISO/IEC 13818-2 (MPEG-2 video), and ISO/IEC 13818-3 (MPEG-2 audio) standards. Approved in November 1994 by the 29th meeting of ISO/IEC JTC1/SC29/WG11 (MPEG) held in Singapore,

Table 3. MPEG Family of Video Codec Standards

Name	Objective	Status
MPEG-1	Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s	ISO/IEC 11172 Standard, completed in October 1992.
MPEG-2	Generic coding of moving pictures and associated audio.	ISO/IEC 13818 Standard, completed in November 1994.
MPEG-3	NA	No longer exists (has been merged into MPEG-2).
MPEG-4	Multimedia applications.	Under development. Expected in 1999.
MPEG-5 and -6	NA	Does not exist.
MPEG-7	Multimedia content description interface.	Under development. Expected in 2000.

MPEG-2 was originally targeted at all-digital broadcasting TV quality video (CCIR 601) at coded bit rates of 4 Mbps to 9 Mbps, but has been used in other applications as well, such as video-on-demand (VoD), digital video disc, personal computing, and so on. Other standards organizations that have adopted MPEG-2 include the DVB (digital video broadcast) in Europe, the Federal Communication Commission (FCC) in the United States, the MITI/JISC in Japan, and the DAVIC consortium.

The MPEG-2 system provides a two-layer multiplexing approach. The first layer, called packetized elementary stream (PES), is dedicated to ensuring tight synchronization between video and audio. The second layer depends on the intended communication medium. The specification for error-free environment such as local storage is called MPEG-2 program stream, and the specification addressing error-prone environment is called MPEG-2 transport stream. However, MPEG-2 transport stream is a service multiplex; in other words, it has no mechanism to ensure the reliable delivery of the transport data. MPEG-2 system is mandated to be compatible with MPEG-1 systems and has the following major advantages over MPEG-1:

- Syntax for efficient coding of interlaced video such as TV programs (e.g., 16×8 block size motion compensation, dual prime, etc.) to achieve a better compression ratio. On the other hand, MPEG-1 is strictly defined for progressive video.
- More efficient coding by including user-selectable DCT dc precision (8, 9, 10, or 11 bits), nonlinear macroblock quantization (more dynamic step size than MPEG-1 to increase the precision of quantization at high bit rate), Intra VLC tables, improved mismatch control, and so on.
- Scalable extensions which permit the division of a continuous video stream into two or more bit streams which represent video at different resolutions, picture quality, and frame rates. Currently, MPEG-2 has four extension modes: *spatial scalability, data partitioning, SNR scalability,* and *temporal scalability.*
- A transport stream suitable to a computer network and wider range of picture frames, from 352×240 up to as large as 16383×16383.
- High-definition TV (HDTV) encoding support at sampling dimension up to $1920 \times 1080 \times 30$ Hz and coded bit rates between 20 Mbit/s and 40 Mbit/s. HDTV was the goal of MPEG-3, but it was later discovered that MPEG-2 syntax also works very well if an optimal balance between sample rate and coded bit rate can be maintained. Now, HDTV is part of the MPEG-2 High-1440 Level and High Level toolkit.

MPEG-2 also allows progressive video sequence, and its decoder can decode MPEG-1 video stream as well. More technical details about differences between MPEG-2 and MPEG-1 video syntax can be found in Section D.9 of ISO/IEC 13818-2.

MPEG-4. *MPEG-4* is a standard for multimedia applications being developed by MPEG and will be ISO/IEC 14496 standard in 1999. MPEG-4 will provide a set of technologies to meet the needs of authors, service providers, and end users of multimedia applications. For authors, MPEG-4 will improve the usability and flexibility of content production and will provide better management and protection of the content. For network service providers, MPEG-4 will provide a set of generic QoS (quality of service) parameters for various MPEG-4 media. For end users, MPEG-4 will enable a higher level of interaction with the content within author defined limits. More specifically, MPEG-4 defines standard ways to do the following:

1. Represent units of aural, visual, and audio–visual content, called *audio–visual objects* (AVOs). The very basic units are called *primitive AVOs.* Primitive AVOs include three-dimensional objects, two-dimensional background, voice, text and graphics, animated human body, and so on.
2. Compose AVOs together to create *compound AVOs* that form audiovisual scenes. MPEG-4's scene composition borrows several concepts from VRML in terms of its structure and the functionality of object composition nodes.
3. Multiplex and synchronize the data associated with AVOs to be transported over network channels with a QoS appropriate for the nature of the specific AVOs. Each AVO is conveyed in one or more elementary streams which are characterized by the requested transmission QoS parameter, stream type, and precision for encoding timing information. Transmission of such streaming information is specified in terms of an *access unit layer* and a conceptual two-layer multiplexer.
4. Interact with the audiovisual scene generated at the receiver's end within author-defined limits. Such interaction includes changing the view or listening point of the scene, dragging an object to a different location and so on.

The coding of conventional images and video in MPEG-4 is similar to MPEG-1 and -2 coding and involves motion prediction and compensation followed by texture coding. MPEG-4 also provides content-based functionalities which support the separate encoding and decoding of content (i.e., objects in a scene, video objects). This feature provides the most elementary mechanism for interactivity—that is, flexible representation and manipulation of video object content in the compressed domain, without the need for further segmentation or transcoding at the receiver. The content-based functionalities are supported by MPEG-4's efficient coding of arbitrary shapes and transparency information.

MPEG-4 extends the video coding functionalities of MPEG-1 and -2, and it will include provisions to efficiently compress video with different input formats, frame rates, pixel depth, bit rates and supports various levels of spatial, temporal, and quality scalability. MPEG-4 includes a *very-low-bit-rate video core* (VLBV) which provides algorithms and tools for applications with data rates between 5 kbit/s and 64 kbit/s, supporting low spatial resolution (typically up to CIF) and low frame rates (typically up to 15 Hz). MPEG-4 also provides the same basic functionalities for video with higher bit-rates, spatial, and temporal parameters up to the ITU-R Rec. 601 resolution.

On February 11, 1998, ISO MPEG decided to adopt the joint proposal submitted by Apple, IBM, Netscape, Oracle, Silicon Graphics, and Sun Microsystems to use QuickTime File Format as the starting point for the development of a digital media storage format for MPEG-4.

MPEG-7. Similar to MPEG-3, MPEG-5 and MPEG-6 have never been defined. MPEG-7, formally known as *multimedia content description interface,* is targeted at the problem of searching audiovisual content. MPEG-7 will specify a standard set of descriptors that can be used to describe various types of multimedia information. MPEG-7 will also standardize ways of defining other descriptors as well as structures (*description schemes*) for the descriptors and their relationships. The descriptions are associated with the content to build up indexes and provide fast and efficient search. The multimedia material can be video, audio, three-dimensional models, graphics, images, speech, and information about how these elements are combined in a multimedia presentation. MPEG-7 is also required to include other information about multimedia material:

- The *form* which could help the user determine whether the material is readable or not. Examples are the codec used (e.g., JPEG, MPEG-2) or the overall data size.
- *Conditions for accessing the material* which could include copyright information and price.
- *Classification* which could include parental rating and content classification categories.
- *Links to other relevant material* which could help the user search and browse the material.
- The *context* that describes the occasion of the recorded material, such as "1997 Purdue football game against Notre Dame."

As we can see, MPEG-7 is built on other standard audio–visual representations such as PCM, MPEG-1, MPEG-2, and MPEG-4. In fact, one functionality of the standard is to provide references to suitable portions of these standards. MPEG-7 descriptors are independent of the ways in which the content is encoded or stored, but they are determined by the user domains and applications. Such an abstraction feature is very important. First, it implies that the same materials can be described through different types of features. It is also related to the way the features can be extracted: Low-level features such as shape, size, texture, color, and so on, can be extracted automatically, whereas the higher-level features such as semantic content need much more human interaction. Second, content descriptions do not have to be stored in the same data stream or on the same storage system with the multimedia material; rather, they can be associated with each other through bidirectional links. In addition to the similar MPEG-4 capability of attaching descriptions to objects within the scene, MPEG-7 will also allow different granularity in its descriptions, offering the possibility of different levels of discrimination. MPEG-7 is now in the process of accepting proposals, and the final international standard is expected in the year 2000.

H.261

H.261 is the most widely used international video compression standard for videophone and video conferencing over the Integrated Services Digital Network (ISDN). This standard describes the video coding and decoding methods for the moving picture component of an audiovisual service at rates up to 2 Mbit/s, which are multiples (1 to 30) of 64 kbit/s. H.261 was completed and approved by ITU in December 1990. The standard is suitable for applications using circuit-switched (phone) networks as their transmission channels, since both basic and primary rate ISDN access with the communication channel are considered within the framework of the standard.

With a bit rate of 64 kbit/s or 128 kbit/s, ISDN can only be used for videophone. It is more suitable for video conferencing at a bit rate of 384 kbit/s or higher where better-quality video can be transmitted. H.261 approximates entertainment quality (VHS) video at the bit rate of 2 Mbyte/s. H.261 is usually used in conjunction with other control and framing standards such as H.221, H.230, H.242, and H.320, for communications and conference control.

The actual encoding algorithm of H.261 is similar to, but incompatible with, that of MPEG. H.261 has two kinds of frames: I (interframe) and P (interframe) frames which compose the decoded sequence IPPPIPPP . . .. I frames are encoded similar to JPEG. P frames are encoded by estimating the next frame based on the current frame and coding predicted differences using some intraframe mechanism. One important estimator is the motion compensated DCT coder which uses a two-dimensional warp of the previous frame, and the difference is encoded using a block transform (the Discrete Cosine Transform). H.261 needs substantially less CPU power for real-time encoding than MPEG because of the real-time requirements of the applications it was designed for. The H.261 encoding algorithm includes a mechanism that optimizes bandwidth usage by trading picture quality against motion, so that a quickly changing picture will be of lower quality than a relatively static picture. H.261 used in this way is constant-bit-rate encoding rather than constant-quality, variable-bit-rate encoding. H.261 supports two kinds of resolutions, QCIF (Quarter Common Interchange Format 176 $\times$ 144) and CIF (Common Interchange Format, 352 $\times$ 288).

H.263

H.263 is an international video codec standard for low bit rate (may be less than 64 kbit/s) communication approved in March 1996 by ITU. The coding algorithm is similar to that of H.261 with changes to improve performance and error recovery: (a) Half-pixel rather than full-pixel precision is used for motion compensation; (b) some parts of the hierarchical structure of the data stream are now optional, so the codec can be configured for a lower data rate or better error recovery; (c) four optional negotiable options are included to improve performance: *unrestricted motion vectors, syntax-based arithmetic coding, advance prediction,* and *forward and backward frame prediction* similar to MPEG's P and B frames. H.263 supports five resolutions including SQCIF (128 $\times$ 96), QCIF, CIF, 4CIF (704 $\times$ 576), and 16CIF (1408 $\times$ 1152). The support of 4CIF and 16CIF makes H.263 competitive with other higher-bit-rate video codec standards like the MPEG standards. It is expected that H.263 will replace H.261 in many applications.

Cinepak

Cinepak is a standard software video codec which is optimized for 16- and 24-bit video content and CD-ROM-based video playback. Cinepack is based on vector quantization-based algorithms for video compression and decompression, and thus it is capable of offering variable levels of compression quality based on time available for compression and the date rate of the target playback machine. Interframe compression is also used to achieve higher compression ratios. The average compression ratio of Cinepak is 20 : 1 compared to original source video. As a highly asymmetric video codec (≈192 : 1), Cinepak decompresses quickly and plays reasonably well on both low-end machines (486s and even 286s) and high-end ones (Pentiums). Cinepak is implemented in Video for Windows as well as QuickTime, which creates portability across various platforms. It can also constrain data rates to user-definable levels for CD-ROM playback. Cinepak also has some weaknesses. First, Cinepak compression is complex and very time-consuming. Second, Cinepak must always compress video at least 10 : 1, so it is less useful at higher data rates for 4× CD-ROM and above. Furthermore, it was never designed for very low bandwidth and, as a result, does not work very well at a data rate under 30 kbyte/s. Current Cinepak licensees include Apple Computer for QuickTime on both MacOS and Windows, Microsoft for Video for Windows, 3DO, Ataro Jaguar, Sega, NeXT Corporation's NeXTStep, Cirrus Logic, Weitek, Western Digital, and Creative Labs.

DVI, Indeo, and IVI

DVI is the Intel's original name for its Digital Video Interactive codec, which is based on the region encoding technique. DVI has been replaced by Intel's Indeo technology (8,9) for scalable software capture and playback of digital video. Intel licenses Indeo technology to companies such as Microsoft, who then integrate it into products such as Microsoft's Video for Windows. Indeo technology can record 8-, 16-, or 24-bit video sequences and store the sequence as 24-bit for scalability on higher-power PCs.

After introducing Indeo 2 and 3, Intel released Indeo 4 and 5 under the new name of Intel Video Interactive (IVI) with many new capabilities. The IVI codec replaces Indeo 3.2's vector quantization technique with a more sophisticated interframe codec using a hybrid wavelet compression algorithm. Wavelet compression works by encoding the image into a number of frequency bands, each representing the image at a different level of sharpness. By representing the image based on frequency content, it is possible to choose which portion of the video data to keep to achieve the desired compression ratio. For example, high-frequency content, which makes up the fine detail of the frame image, can be reduced to achieve a considerable amount of compression without some of the characteristic blockiness of other codecs.

Other interesting features of IVI include scaling, transparency, local window decode, and access protection. Scaling means that the codec can be used to adapt video playback to the processor power of a particular machine. The transparency feature of the IVI lets video or graphical objects be overlaid onto either a video or a background scene and be interactively controlled, which is ideal for interactive video applications. Local video decoding gives programmers the ability to decode any rectangular subregion of a video image. The size and the location of such subregions can be dynamically adjusted during the playback. Each IVI video can also be password-protected, which is very useful for video developers in controlling video distribution. IVI also allows the developer to place key video frames during the video compression process, and thus it supports rapid access to selected points in the video without giving up the interframe compression. Another useful feature is on-the-fly contrast, brightness, and saturation adjustment.

IVI produces better image quality than other codecs such as Cinepak, but it was designed for the Pentium II chip and MMX technologies and requires lots of processor power. Compared to Cinepak, IVI produces superior image quality. But on the low-end PCs, the quality of Indeo video playback can be very poor. Video files compressed with IVI are usually 1.5 to 2 times larger than those of MPEG-1; however, on a fast Pentium machine with a video accelerated graphic card, Indeo plays back at a quality comparable to software MPEG-1 players. The distinctive advantage of IVI is its support for features necessary to develop multimedia games and applications incorporated with interactive video. Currently, IVI is available for use with Video for Windows, and a QuickTime version is also promised.

QUICKTIME

QuickTime (10,11) is often mistaken as one of the video codecs. In fact, like Microsoft's Video for Windows, it is defined by Apple as an architecture standard for computer systems to accommodate video as well as text, graphics, animation, and sound. Unlike video codecs, multimedia architectures such as QuickTime are more concerned with defining a usable API so that program developers can generate cross-platform multimedia applications quickly and effectively. Thus, neither QuickTime nor Video for Windows specifies a specific video codec. Rather, they assume that all kinds of encodings/decoding will be available through hardware/software codecs, and thus provide meta-systems that allow the programmer to name the encoding and provide translations.

QuickTime is composed of three distinct elements: the *QuickTime Movie file format,* the *Quicktime Media Abstraction Layer,* and a set of built-in *QuickTime media services.* The QuickTime Movie file format specifies a standard means of storing digital media compositions. Using this format, we can not only store media data individually, but also store a complete description of the overall media composition. Such description might include a description of the spatial and auditory relationships between multiple video and audio channels in a more complex composition. QuickTime Media Abstraction Layer specifies how software tools and applications access media support services built into QuickTime. It also defines how hardware can accelerate performance critical portions of the QuickTime system. Finally, the QuickTime Media Abstraction Layer outlines the means by which component software developers can extend and enhance the media services accessible through QuickTime. QuickTime also has a set of built-in media services that application developers can take advantage of to reduce the time and resources needed for the development. QuickTime 3.0 includes built-in support for 10 different media types including video, audio, text, timecode, music/MIDI, sprite/animation, tween, MPEG, VR, and 3D. For each built-in media type, QuickTime provides a set of media specific services appropriate for managing each particular media type.

QuickTime supports a wide variety of video file formats including Microsoft's AVI (Audio/Video Interleaved), open DML (a professional extension of AVI), Avid's OMF (Open Media Framework), MPEG-1 video and audio, DV, Cinepak, Indeo, MJPEG, and many others. For example, QuickTime 3.0 (10) has built-in platform-independent software support (DV codec) for DV. This means that all current QuickTime-enabled application can work with DV without any changes. DV data can be played, edited, combined with other video formats, and easily converted into other formats such as Cinepak or Indeo for CD-ROM video delivery. The QuickTime DV encoder can also encode video of other format into DV which can be transferred back to DV camcorder using Firewire. QuickTime has the potential of becoming a computer-industry standard for the interchange of video and audio quences. According to a recent survey (11), over 50% of all Web video is in QuickTime format. MPEG format is in second place and can also be played in any QuickTime application.

VIDEO DATABASES

It is impossible to cope with the huge and ever-increasing amount of video information without systematic video data management. In the past, a similar need led to the creation of computerized textual and numeric database management systems (DBMSs). These alpha-numerical DBMSs were designed mainly for managing simple structured data types. However, the nature of video data is fundamentally different than alpha-numerical data, and it requires new ways of modeling, inserting, indexing, and manipulating data. A *video database management system (VDBMS)* can be defined as a software system that manages a collection of video data and provides content-based access to users (3). A generic video database management system is shown in Fig. 2. Similar to the issues involved in the traditional DBMS (12), a VDBMS needs to address the following important issues:

- *Video data modeling,* which deals with the issue of representing video data—that is, designing the high-level abstraction of the raw video to facilitate various operations. These operations include video data insertion, editing, indexing, browsing, and querying. Thus, modeling of the video data is usually the first thing done in the VDBMS design process and has great impact on other components. The video data model is, to a certain extent, user- and application-dependent.
- *Video data insertion,* which deals with the issue of introducing new video data into a video database. This usually includes the following steps: (1) key information (or features) extraction from video data for instantiating a data model; the automatic feature extraction can usually be done by using image processing and computer vision techniques for video analysis. (2) Break the given video stream into a set of basic units; this process is often called *video scene analysis and segmentation.* (3) Manually or semiautomatically annotate the video unit; what needs to be annotated is usually within the application domain. (4) Index and store video data into the video database based on the extracted information and annotated information about video data.
- *Video data indexing,* which is the most important step in the video data insertion process. It deals with the organization of the video data in the video database to make user access more efficient. This process involves the identification of important features and computing the search keys (indexes) based on them for ordering the video data.
- *Video data query and retrieval,* which deals with the extraction of video data from the database that satisfies certain user-specified query conditions. Due to the nature of video data, those query conditions are usually ambiguous in that the video data satisfying the query condition are not unique. This difficulty can be partially overcome by providing a graphic user interface (GUI) and video database browsing capability to the users. Such a GUI can greatly help the user with regard to query formulation, result viewing and manipulation, and navigation of the video database.

VIDEO DATA MODELING

Traditional data models like the relational data model have long been recognized as inadequate for representing the rich data structures required by image and video data. In the past few years, many video data models have been proposed; they

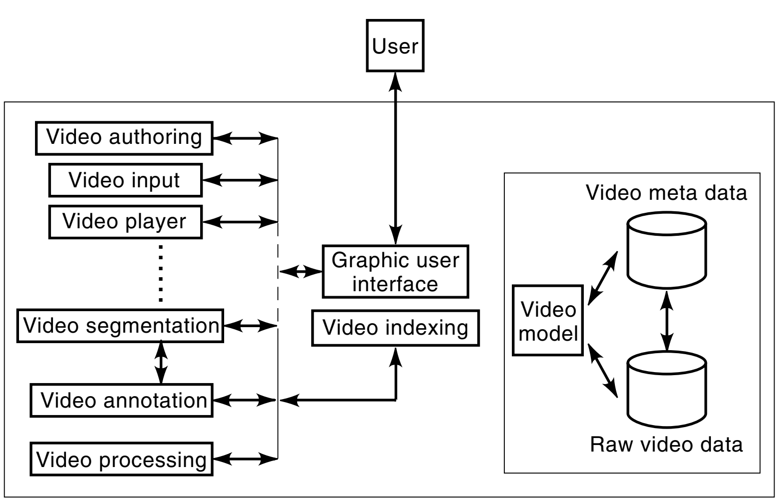

Figure 2. A generic video database system.

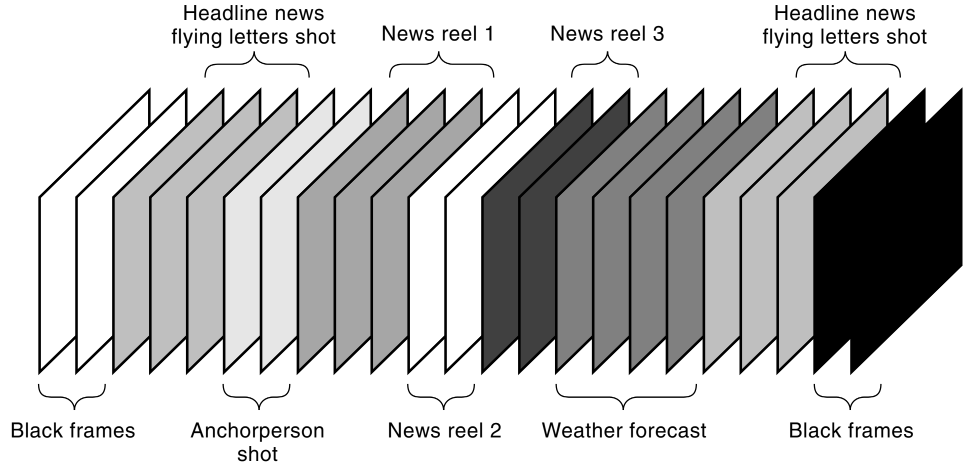

Figure 3. Frames and shots of a CNN "Headline News" episode.

can basically be classified into the following categories (2): segmentation-based models, annotation layering-based models, and video object-based models. In order to provide efficient management, a VDBMS should support video data as one of its data types, just like textual or numerical data. The supporting video data model should integrate both video content and its semantic structure. Structural and temporal relationships between video segments should also be expressed. Other important requirements for a video data model include:

Multilevel Video Structure Abstraction Support. There are two inherent levels of video abstractions: the entire video stream and the individual frames. For most applications, the entire video stream is too coarse as a level of abstraction. On the other hand, a single video frame is too brief to be the unit of interest. Other intermediate abstractions, such as scenes, are required, and thus a hierarchy of video stream abstraction can be formed. At each level of hierarchy, additional information, like shot type, should be allowed to be added. Such multilevel abstraction makes it easier to reference and comprehend video information, as well as being more efficient to index, browse, and store video data.

The *shot* is often considered as the basic structural element for characterizing the video data which can be defined as one or more frames generated and recorded contiguously, representing a continuous action in time and space (13). *Shots* that are related in time and space can be assembled in an *episode* (14), and Fig. 3 is an example representing the CNN "Headline News" episode structure. Another example is the *compound unit-sequence-scene-shot* hierarchical video stream structure in the VideoStar system (15). A *scene* is a set of shots that are related in time and space, and scenes that together give a meaning are grouped into what is called a *sequence.* Related sequences are assembled into a *compound unit* which can be recursively grouped together into a *compound unit* of arbitrary level.

Spatial and Temporal Relationship Support. A key characteristic of video data are the associated spatial and temporal semantics which distinguish video data from other types of data. Thus, it is important that the video model identifies physical objects and their relationship in time and space to support user queries with temporal and spatial constraints. The temporal relationships between different video segments are also very important from the perspective of a user navigating through a video. There are thirteen distinct ways in which any two intervals can be related (16), and they can be represented by seven cases (17) (*before, meets, overlaps, during, starts, finishes,* and *equal*), since six pairs of them are inverses of each other. Those temporal relations are used in formulating queries that contain temporal relationship constraints among the video segments (15,18). For spatial relations, most of the techniques are based on projecting objects on a two- or three-dimensional coordinate system. Very few research attempts have been made to formally represent the spatiotemporal relationship of objects contained in video data and support queries with such constraints.

Video Annotation Support. A video data model should support incremental and dynamic annotation of the video stream. Unlike textual data, digital video does not easily accommodate the extraction of content features because fully automatic image and speech recognition is not yet feasible. Moreover, the structure of a video captures some of the aspects of the video material but is not suited for the representation of every characteristic of the material. It should be possible to make detailed descriptions of the video content that is linked not necessarily directly to structural components but more often to arbitrary frame sequences (18–20). Additionally, video annotations often change dynamically depending on human interpretation and application contexts. Currently, the video annotation process is mostly an off-line and manual process. Because of this, GUIs are often built to help users input descriptions of video data. Video annotation will remain interactive in general until significant breakthroughs are made in the field of computer vision and artificial intelligence.

Video Data Independence. Data independence is a fundamental transparency that should be provided by a VDBMS. One of the advantages of data independence is sharing and reuse of video data; that is, the same basic video material may be used in several different video documents. Sharing and reuse is critical in a VDBMS because of the sheer volume and rich semantics of video data, which can be achieved by defining abstract logical video concepts on the top of physical video data (15,18). For example, Hjelsvold et al. (15) define the video content of a video document as a logical concept called *VideoStream,* which can be mapped onto a set of physically stored video data called *StoredVideoSegment.* On the other hand, *logical video streams* and *logical video segments* are proposed by Jiang et al. (18) as higher-level abstractions of *physical video segments.*

Segmentation-Based Video Data Models

Segmentation-based video models (14,21,22) rely heavily on image processing techniques. For a given video stream, scene change detection (SCD) algorithms (23) are usually used to parse and segment the stream into a set of basic units called *shots.* A representing video frame (RFrame) can then be selected from each shot; together they can represent the corresponding video stream. The features of RFrames can be extracted and serves as video indices. In a more sophisticated video model, shots can also be matched or classified against a set of domain specific templates (or patterns or models) in order to extract higher-level semantics and structures contained in the video such as episodes. A hierarchical representation of the video stream can then be built. One example of such a shot model in the CNN news video is the anchor/person shot (14,24) which can be based on locations of the set of features within frames. These features include the "Headline News" icon in the lower right corner and the title of the anchor/person name.

The main advantage of segmentation-based video data models is that the indexing process can be fully automated. However, they also have several limitations (18). First, they lack flexibility and scalability since the video streams are presegmented by the SCD algorithms. Second, the similarity measure between two frame images is often ill-defined and limited, making the template matching process unreliable. Third, they lack applicability for video streams that do not have well-defined structures. For example a class lecture video can have no clear visual structure in terms of shots. Therefore, segmentation using SCD algorithms would be extremely difficult. Finally, limited semantics can be derived from template matching processes, and templates themselves are application-specific.

Annotation Layering-Based Models

Video annotations are often used to record the semantic video (or image) content and provide content-based access in multimedia systems such as Video-on-Demand (VoD) (25). The automatic creation of video annotations may be one of the major limitations of annotation-based models; however, it still can be done by (1) using a closed caption decoder, (2) detecting and extracting text that appears in the video using OCR techniques (26,27), or (3) capturing and transforming audio signals into text through voice recognition techniques (28,29). The basic idea of annotation-based models is to layer the content information on top of a video stream, rather than segment video data into shots. One of the earliest annotation-based models is the stratification model (30). An example of video models that extends this basic idea is the generic video data model in VideoStar system (15). It allows for free text annotation of any arbitrary video frame sequence by establishing an *Annotates* relationship between a *Frame Sequence* and an *Annotation.* The sharing and reuse of the video material is supported by the idea of logical *VideoStream.* This model, however, supports only simple Boolean queries on the video annotations. Nested stratification is allowed in the Algebraic Video model (31); that is, logical video segments can be overlapped or nested. Multiple views of the same raw video segment can be defined, and video algebraic operators are used for the recomposition of the video material. Four kinds of interval relations (*precede, follow, overlap, and equal*) are defined as attributes of a logical video segment. The Smart VideoText model (18,32) is based on multilevel video data abstraction and concept graph (CG) (33) knowledge representation. It not only supports Boolean and all the possible temporal interval (16) constraints, but also captures the semantic associations among the video annotations with CGs and supports knowledge-based query and browsing. VideoText model allows multiple users to dynamically create and share free text video content descriptions. Each annotation is mapped into a logical video segment which can be overlapped in arbitrary ways.

To summarize, annotation layering-based video models have several advantages. First, they support variable video access granularities, and annotations can be made on logical video segment of any length. Second, video annotations can easily be handled by existing sophisticated information retrieval (IR) and database techniques. Third, multiple annotations can be linked to the same logical segment of video data, and they can be added and deleted independently of the underlying video streams. Thus, they support dynamic and incremental creation and modification of the video annotations, as well as users's views. Finally, annotation layering-based video models support semantic content-based video queries, retrieval, and browsing.

Video Object Models

Two prevailing data models used in current DBMSs are the relational and object-oriented models. The object-oriented model has several features that make it an attractive candidate for modeling video data. These features include capabilities of complex object representation and management, object identities handling, encapsulation of data and associated methods into objects, and class hierarchy-based inheritance of attribute structures and methods. However, modeling the video data using the object-oriented data model is also strongly criticized (34,35), mainly for the following reasons:

- Video data are raw data created independently from their contents and database structure, which is described later in the annotation process.
- In traditional data models such as the object-oriented model, the data schema is static. The attributes of an object are more or less fixed once they are defined, and adding or deleting attributes is impossible. However, attributes of the video data cannot be defined completely in advance because descriptions of video data are user- and application-dependent, and the rich information contained in video data implies that semantic meaning should be added incrementally. Thus, a video data model should support an arbitrary attribute structure for the video data as well as incremental and dynamic evaluation of the schemas and attributes.
- Many object-oriented data models only support class-based inheritance. However, for the video data objects, which usually overlap or include each other, support for *inclusion inheritance* (35) is desired. Inclusion inheritance enables sharing of descriptive data among the video objects.

The notion of *video object* is defined in the object-oriented video information database (OVID) (35) as an arbitrary sequence of video frames. Each video object consists of a unique identifier, an interval presented by a pair of starting and ending frame numbers, and the contents of the video frame sequence described manually by a collection of attribute and value pairs. The OVID video data model is *schemaless;* that is, it does not use the class hierarchy as a database schema like in the OODB system. Arbitrary attributes can be attached to each video object if necessary. This enables the user to describe the content of the video object in a dynamic and incremental way. Additionally, interval inclusion inheritance is applied to ease the effort of providing description data when an existing video is composed into new video objects using the generalization hierarchy concept. This approach, however, is very tedious since the description of video content is done manually by users, and not through an automatic image processing mechanism.

VIDEO CUT DETECTION AND SEGMENTATION

One fundamental problem that has a great impact on all aspects of video databases is the content-based temporal sampling of video data (36). The purpose of the content-based temporal sampling is to identify significant video frames to achieve better representation, indexing, storage, and retrieval of the video data. Automatic content-based temporal sampling is very difficult due to the fact that the sampling criteria are not well defined; whether a video frame is important or not is usually subjective. Moreover, it is usually highly application-dependent and requires high-level, semantic interpretation of the video content. This requires the combination of very sophisticated techniques from computer vision and artificial intelligence. The state of the art in those fields, however, has not advanced to the point where semantic interpretations would be possible.

However, satisfying results can still be obtained by analyzing the visual content of the video and partitioning it into a set of basic units called *shots.* This process is also referred to as *video data segmentation.* Content-based sampling thus can be approximated by selecting one representing frame from each shot since a shot can be defined as a continuous sequence of video frames which have no significant interframe difference in terms of their visual contents. A single shot usually results from a single continuous camera operation. This partitioning is usually achieved by sequentially measuring interframe differences and studying their variances—for example, detecting sharp peaks. This process is often called *scene change detection* (SCD).

Scene change in a video sequence can be either abrupt or gradual. *Abrupt scene changes* result from editing “cuts,” and detecting them is often called *cut detection.* Gradual scene changes result from chromatic, spatial, and/or combined video edits such as zoom, camera pan, dissolve and fade in/out, and so on. An example of abrupt scene change and gradual scene change is shown in Fig. 4. SCD is usually based on some measurements of the image frame, which can be computed from the information contained in the images. This information can be color, spatial correlation, object shape, motion contained in the video image, or discrete cosine (DC) coefficients in the case of compressed video data. In general, gradual scene changes are more difficult to detect than the abrupt scene changes and may cause many SCD algorithms to fail under certain circumstances.

Existing SCD algorithms can be classified in many ways according to, among others, the video features they use and the video objects to which they can be applied. Here, we discuss SCD algorithms in three main categories: (1) approaches that work on uncompressed full image sequences; (2) algorithms that aim at working directly on the compressed video; and (3) approaches that are based on explicit models. The latter are also called *top-down approaches,* whereas the first two categories are called *bottom-up approaches* (3).

Preliminaries

We now introduce some basic notations, concepts, and several common interimage difference measurements. It should be noted that those measurements may not work well for scene detection when used separately, and thus they usually are combined in the SCD algorithms.

A sequence of video images, whether fully uncompressed or spatially reduced, is denoted as I_i, $0 \le i < N$, where N is the length or the number of frames of the video data. $I_i(x, y)$ denotes the value of the pixel at position (x, y) for the ith frame. H_i refers to the histogram of the image I_i. The interframe difference between images I_i and I_j according to some measurement is represented as $d(I_i, I_j)$.

DC Images and DC Sequences. A *DC (discrete cosine) image* is a spatially reduced version of a given image. It can be obtained by first dividing the original image into blocks of $n \times n$ pixels each, then computing the average value of pixels in each block which corresponds to one pixel in the DC image. For the compressed video data (e.g., MPEG video), a sequence DC images can be constructed directly from the compressed

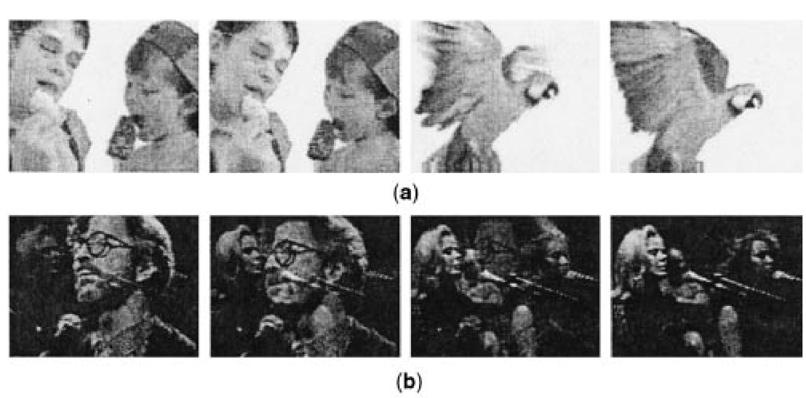

Figure 4. Example of an abrupt scene change (a) and a gradual scene change (b).

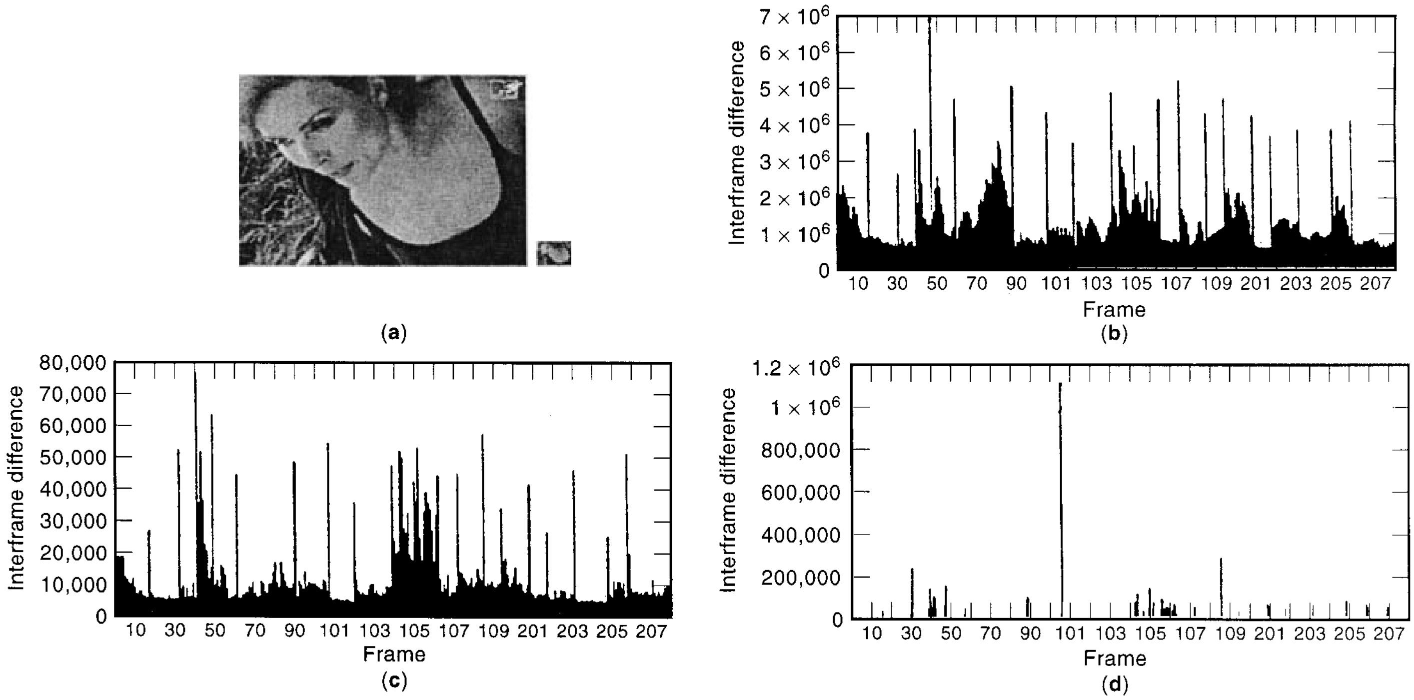

Figure 5. (a) An example of a full image and its DC image, (b) template matching, (c) color histogram, and (d) χ^2 histogram.

video sequence, which is called a *DC sequence.* Figure 5(a) is an example of a video frame image and its DC image.

There are several advantages of using DC images and DC sequences in the SCD for the compressed video (27). First, DC images retain most of the essential global information for image processing. Thus, lots of analysis done on the full image can also be done on its DC image instead. Second, DC images are considerably smaller than the full image frames which makes the analysis on DC images much more efficient. Third, partial decoding of compressed video saves more computation time than full-frame decompression. Extracting the DC image of an I frame from an MPEG video stream is trivial since it is given by its DCT coefficients. Extracting DC images from P frames and B frames requires interframe motion information which may result in many multiplication operations. The computation can be speeded up using approximations (37). It is claimed (27) that the reduced images formed from DC coefficients, whether they are precisely or approximately computed, retain the "global features" which can be used for video data segmentation, SCD, matching, and other image analysis.

Basic Measurements of Interframe Difference

Template Matching. Template matching is done by comparing the pixels of two images across the same location which can be formulated as $d(I_i, I_j) = \sum_{x=0,y=0}^{x<M,y<N} |I_i(x, y) - I_j(x, y)|$ with image size of $M \times N$. Template matching is very sensitive to noise and object movements because it is strictly tied to pixel locations. This can cause false SCD and can be overcome to some degree by partitioning the image into several subregions. Figure 5(b) is an example of interframe difference sequence based on template matching. The input video is the one that contains the first image sequence in Fig. 4.

Color Histogram. The color histogram of an image can be computed by dividing a color space (e.g., RGB) into discrete image colors called *bins* and counting the number of pixels fall into each bin. The difference between two images I_i and I_j, based on their color histograms H_i and H_j, can be formulated as $d(I_i, I_j) = \sum_{k=1}^{n} |H_{ik} - H_{jk}|$, which denotes the difference in the number of pixels of two image that fall into same bin. In the RGB color space, the above formula can be written as $d_{\mathrm{RGB}}(I_i, I_j) = \sum_k^n (|H_i^r(k) - H_j^r(k)| + |H_i^g(k) - H_j^g(k)| + |H_i^b(k) - H_j^b(k)|)$. Using only simple color histogram may not be effective at detecting scene changes because two images can be very different in structure and yet have similar pixel values. Figure 5(c) is the interframe difference sequence of the first video sequence in Fig. 4 measured by the color histogram.

χ^2 Histogram. The χ^2 histogram computes the distance measure between two image frames as $d(I_i, I_j) = \sum_{k=1}^{n} [H_{i(k)} - H_{i(k)})^2/H_{j(k)}]$, which is used in many existing SCD algorithms. Experiments indicate that this method generates better results when compared with other intensity-based measurements, e.g., color histogram and template matching. Figure 5(d) is the interframe difference sequence of the first video sequence in Fig. 4 measured by χ^2 histogram.

Full Image Video Scene Change Detection

Most of the existing work on SCD is based on full image video analysis. The differences between the various SCD approaches are the measurement function used, the features chosen to be measured, and the subdivision of the frame images. The existing algorithms use either intensity features or motion information of the video data to compute the interframe difference sequence. The intensity-based approaches, however, may fail when there is a peak introduction by object or camera motion. Motion-based algorithms also have the drawback of being computationally expensive since they usually need to match the image blocks across video frames. After the interframe differences are computed, some approaches use a global threshold to decide a scene change. This is clearly insuffìcent since a large global difference does not necessarily imply that there is a scene change. In fact, scene changes with globally low peaks is one of the main causes of the failure of the algorithms. Scene changes, whether abrupt or gradual, are localized processes, and should be checked accordingly.

Detecting Abrupt Scene Changes. Algorithms for detecting abrupt scene changes have been extensively studied, and over 90% accuracy rate has been achieved. Following are some algorithms developed specifically for detecting abrupt scene changes without taking gradual scene changes into consideration. Nagasaka and Tanaka (38) presented an approach that partitions the video frames into 4×4 equal-sized windows and compares the corresponding windows from the two frames. Every pair of windows is compared and the largest difference is discarded. The difference values left are used to make the final decision. The purpose of the subdivision is to make the algorithm more tolerant of object movement, camera movement, and zooms. Six different types of measurement functions, namely difference of gray-level sums, template matching, difference of gray-level histograms, color template matching, difference of color histogram, and a χ^2 comparison of the color histograms, have been tested. The experimental results indicate that a combination of image subdivision and χ^2 color histogram approach provides the best results. Akutsu et al. (39) used both the average interframe correlation coefficient and the ratio of velocity to motion in each frame of the video to detect scene change. Their assumptions were that (a) the interframe correlation between frames from the same scene should be high, and (b) the ratio of velocity to motion across a cut should be high also. Hsu and Harashima (40) treated scene changes and activities in the video stream as a set of motion discontinuities which change the shape of the spatiotemporal surfaces. The sign of the Gaussian and mean curvature of the spatiotemporal surfaces is used to characterize the activities. Scene changes are detected using an empirically chosen global threshold. The clustering and split-and-merge approach are then taken to segment the video.

Detecting Gradual Scene Changes. Robust gradual SCD is more challenging than its abrupt counterpart, especially when, for example, there is a lot of motion involved. Unlike abrupt scene changes, a gradual scene change does not usually manifest itself by sharp peaks in the interframe difference sequence and can thus be easily confused with object or camera motion. Gradual scene changes are usually determined by observing the behavior of the interframe differences over a certain period of time. For example, the *twin comparison* approach (41) algorithm uses two thresholds T_b, T_s, $T_s < T_b$ for camera breaks and gradual transition, respectively. If the histogram value difference $d(I_i, I_{i+1})$ between consecutive frames with difference values satisfies $Ts < d(I_i, I_{i+1}) < T_b$, they are considered potential start frames for the gradual transition. For each potential frame detected, an accumulated comparison $A_c(i) = D(I_i, I_{i+1})$ is computed until $A_c(i) > T_b$ and $d(I_i, I_{i+1}) < T_s$. The end of the gradual transition is declared when this condition is satisfied. To distinguish gradual transition from other camera operations such as pans and zooms, the approach uses image flow computations. Gradual transitions result in a null optical flow when there are other camera operations resulting in particular types of flows. The approach achieves good results, with failures occurring due to either (a) similarity in color histograms across shots when color contents are very similar or (b) sharp changes in lighting such as flashes and flickering object.

Shahraray (36), on the other hand, detected abrupt and gradual scene changes based on motion-controlled temporal filtering of the disparity between consecutive frames. Each image frame is subdivided, and image block matching is done based upon image intensity values. A nonlinear order statistical filter (42) is used to combine the image matching values of different image blocks; that is, the weight of an image match value in the total sum depends on its order in the image match value list. It is claimed that this match measure of two images is more consistent with a human's judgment. Abrupt scene changes are detected by a thresholding process, and gradual transitions are detected by the identification of sustained low-level increases in image matching values. False detection due to the camera and object motions are suppressed by image block matching as well as temporal filtering of the image matching value sequence. SCD results can be verified simply by measuring the interframe difference of representing frames resulting from the SCD algorithm; high similarity would likely indicate a false detection.

To improve the result of detecting fades, dissolves, and wipes which most existing algorithms have difficulties with, Zabih et al. (43) proposed an algorithm based on the edge changing fraction. They observed that new intensity edges appear (enter the scene) far from the locations of old edges during a scene change, and that old edges disappear (exit the scene) far from the locations of old edges. Abrupt scene changes, fades, and dissolves are detected by studying the peak values in a fixed window of frames. Wipes can be identified through the distribution of entering and exiting edge pixels. A global computation is used to guard the algorithm from camera and object motion. The experimental results indicate that the algorithm is robust against the parameter variances, compression loss, and subsampling of the frame images. The algorithm performs well in detecting fades, dissolves, and wipes but may fail in cases of very rapid changes in lighting and fast moving objects. It may also have difficulties when applied to video that is very dim where no edge can be detected.

Scene Change Detection on the Compressed Video Data

Two approaches can be used to detect scene changes on compressed video streams. The video stream can be fully decompressed, and then the video scene analysis can be performed on full frame image sequence. However, fully decompressing the compressed video data can be computationally intensive. To speed up the scene analysis, some SCD algorithms work directly on compressed video data without the full decompression step. They produce results similar to that of the full-image-based approach, but are much more efficient. Most of SCD algorithms in this category have been tested on the DCT-based standard compressed video since DCT (discrete cosine transformation)-related information can be extracted directly and doesn't require full decompression of video stream. Some algorithms operate on the corresponding DC image sequences of the compressed video (27,44,45), while some use DC coefficients and motion vectors instead (46–49). They all need only partial decompression of the video.

DC Image Sequence-Based Approach. Yeo and Liu (27,44,45) propose to detect scene changes in the DC image sequence of the compressed video data. Global color statistic comparison (RGB color histogram) is found to be less sensitive to the motion but more expensive to compute. Although template matching is usually sensitive to the camera and object motion and may not produce good results as the full frame image case, it is found to be more suitable for DC sequences because DC sequences are smoothed versions of the corresponding full images. Yeo's algorithm uses template matching measurement. Abrupt scene changes were detected by first computing the interframe difference sequence and then applying a slide window of size m. A scene change is found if the difference between two frames is the maximum within a symmetric window of size $2m - 1$ and is also n times the second largest difference in the window. The second criteria is for the purpose of guarding false SCD because of fast panning, zooming, or camera flashing. The window size m is set to be smaller than the minimum number of frames between any scene change. The selection of parameters n and m relates to the trade-off between missed detection rate and false detection rate; typical values can be $n = 3$ and $m = 10$. Gradual scene changes can also be captured by computing and studying the difference of every frame with the previous kth frame—that is, checking if a "plateau" appears in the difference sequence. Experimental results indicate that over 99% of abrupt changes and 89.5% of gradual changes can be detected. This algorithm is about 70 times faster than on full image sequences, which conforms to the fact that the size of the DC images of a MPEG video is only $\frac{1}{64}$ of their original size. Although there may exist situations in which DC images are not sufficient to detect some video features (27), this approach is nonetheless very promising.

DC Coefficients-Based Approach. Arman et al. (46) detect scene changes directly on MJPEG video by choosing a subset of the DC coefficients of the 8×8 DCT blocks to form a vector. The assumption is that the inner product of the vectors from the same scene is small. A global threshold is used to detect scene changes; and in case of uncertainty, a few neighboring frames are then selected for further decompression. Color histograms are used on those decompressed frames to find the location of scene changes. This approach is computationally efficient but does not address gradual transitions. Sethi and Patel (47) use only the DC coefficients of I frames of a MPEG video to detect scene changes based on luminance histogram. The basic idea is that if two video frames belong to the same scene, their statistical luminance distribution should be derived from a single statistical distribution. Three statistical tests used are Yakimovsky's likelihood ratio test, the χ^2 histogram comparison test, and the Kolmogorov–Smirnov test. Experiments show that the χ^2 histogram comparison seems to produce better results. DCT blocks and vector information of a MPEG video are used by Zhang et al. (48) to detect scene changes based on a count of nonzero motion vectors. It is observed that the number of valid motion vectors in P or B frames tended to be low when such frames lie between two different shots. Those frames are then decompressed, and full-image analysis is done to detect scene changes. The weakness of this approach is that motion compensation-related information tends to be unreliable and unpredictable in the case of gradual transitions, which might cause the approach to fail. Meng et al. (49), use the variance of DC coefficients I and P frames and motion vector information to characterize scene changes of MPEG-I and MPEG-II video streams. The basic idea is that frames tend to have very different motion vector ratios if they belong to different scenes and have very similar motion vector ratios if they are within the same scene. Their scene-detection algorithm works in the following manner. First an MPEG video is decoded just enough to obtain the motion vectors and DC coefficients. Inverse motion compensation is applied only to the luminance microblocks of P frames to construct their DC coefficients. Then the suspected frames are marked in the following ways: (a) An I frame is marked if there is a peak interframe histogram difference and the immediate B frame before it has a peak value of the ratio between forward and backward motion vectors; (b) a P frame is marked if there is a peak in its ratio of intracoded blocks and forward motion vectors; and (c) a B frame is marked if its backward and forward motion vector ratio has a peak value. Final decisions are made by going through the marked frames to check whether they satisfy the local window threshold. The threshold is set according to the estimated minimal scene change distance. Dissolve effect is determined by noticing a parabolic variance curve.

It should be pointed out that the above algorithms also have following limitations. First, current video compression standards like MPEG are optimized for data compression rather than for the representation of the visual content and they are lossy. Thus, they do not necessarily produce accurate motion vectors. Second, motion vectors are not always readily obtainable from the compressed video data since a large portion of the existing MPEG video has I frames only. Moreover, some of the important image analysis, such as automatic caption extraction and recognition, may not be possible on the compressed data.

Model-Based Video Scene Change Detection

It is possible to build an explicit model of scene changes to help the SCD process (3,50,51). These model-based SCD algorithms are sometimes referred to as top-down approaches, whereas algorithms discussed above are known as bottom-up approaches. The advantages of the model-based SCD is that a systematic procedure based on mathematical models can be developed, and certain domain-specific constraints can be added to improve the effectiveness of the approaches. For example, the *production model-based classification* approach (3,51) is based on a study of the video production process and different constraints abstracted from it. The *edit effect model* contains both abrupt and gradual scene changes. Gradual scene changes such as fade and dissolve are modeled as chromatic scaling operations; for example, fade is modeled as a chromatic scaling operation with positive and negative fade rates. The algorithm first identifies the features that correspond to each of the edit classes to be detected and then classifies video frames based on these features by using both template matching and χ^2 histogram measurements. Feature vectors extracted from the video data are used together with the mathematical models to classify the video frames and to detect any edit boundaries. This approach has been tested with cut, fade, dissolve, and spatial edits, at an overall 88% accurate rate. Another example is an SCD algorithm based on a differential model of the distribution of pixel value differences in a video stream (50). The model includes: (1) a small amplitude additive zero-centered Gaussian noise that models camera, film, and other noises; (2) an intrashot change model for pixel change probability distribution constructed from object, camera motion, angle change, focus, or light change at a given time and in a given shot; and (3) a set of shot transition models for different kinds of abrupt and gradual scene changes that are assumed to be linear. The SCD algorithm first reduces the resolution of frame images by undersampling to overcome the effects of the camera and object motion and make the compensation more efficient in the following steps. The second step is to compute the histogram of pixel difference values and count the number of pixels whose change of value is within a certain range determined by studying above models. Different scene changes are detected by checking the resulting integer sequence. Experiments show that the algorithm can achieve 94% to 100% detection rate for abrupt scene changes and around 80% for gradual scene changes.

Evaluation Criteria for SCD Algorithm Performance

It is difficult to evaluate and compare existing SCD algorithms due to the lack of objective performance measurements. This is mainly attributed to the diversity in the various factors involved in the video data. There are, however, still some common SCD performance measurements (23) given a set of benchmark video: (1) speed in terms of the number of frames processed per time unit; (2) average success rate or failure rate which includes both false detection and missed detection (a 100% scene change capture rate does not imply that the algorithm is good since it may have very high false change alarms); (3) accuracy in terms of determining the precise location and type of a scene change; (4) stability, that is, its sensitivity to the noise in the video stream (flashing of the scene and background noises often trigger the false detection); (5) types of the scene changes and special effects that the algorithm can handle; and (6) generality in terms of the applications it can be applied to and kinds of video data resources it can handle.

Further improvement on existing SCD algorithm can be achieved in the following ways (23). First, use additional available video information such as closed caption and audio signal. Some initial efforts (29,52) on using audio signal have been made for video skimming and browsing support. Second, develop adaptive SCD algorithms that can combine several SCD techniques and self-adjust various parameters for different video data. Third, use a combination of various scene-change models. Different aspects of video editing and production process can be individually modeled for developing detectors for certain scene changes. Another idea is to develop new video codecs that include more information about the scene content (53). Current motion-compensated video codec standards like MPEG complicate the scene analysis task by partitioning the scene into arbitrary tiles, resulting in a compressed bitstream that is not physically or semantically related to the scene structure. A complete solution to the SCD problem, however, may require information available from psychophysics (54) and understanding the neural circuitry of the visual pathway (55). Techniques developed in computer vision for detecting motion or objects (56–58) can also be incorporated into SCD algorithms.

VIDEO INDEXING

Accessing video data is very time-consuming because of the huge volume and complexity of the data within the video databases. Indexing of video data is needed to facilitate the process, which is far more difficult and complex compared to the traditional alpha-numerical databases. In traditional databases, data are usually selected on one or more key fields (or attributes) that can uniquely identify the data itself. In video databases, however, what to index on is not as clear and easy to determine. The indexes can be built on audio–visual features, annotations, or other information contained in the video. Additionally, unlike alpha-numerical data, content-based video data indexes are difficult to generate automatically. Video data indexing is also closely related to the video data model and possible user queries. Based on how the indexes are derived, existing work on video indexing can be classified into three categories: annotation-based indexing, feature-based indexing, and domain-specific indexing.

Annotation-Based Indexing

Video annotation is very important for a number of reasons. First, it fully explores the richness of the information contained in the video data. Second, it provides access to the video data based on its semantic content rather than just its audio–visual content like color distribution. Unfortunately, due to the current limitations of machine vision and image-processing techniques, full automation of video annotation, in general, still remains impossible. Thus, video annotation is a manual process that is usually done by an experienced user, either as part of the production process or as a post-production process. The cost of manual annotation is high and thus not suitable for the large collection of video data. However, in certain circumstances, video annotation can also be automatic captured from video signals as we discussed in the section entitled "Video Data Modeling." Automatic video semantic content extraction using computer vision techniques with given application domain and knowledge has also been studied. One example is the football video tracking and understanding (59). Another example is the animal behavior video database (4). In addition, video database systems usually provide a user-friendly GUI to facilitate video annotation creation and modification.

One of the earliest ideas for recording descriptive information about the film or video is the *stratification model* (30). It approximates the way in which the film/video editor builds an understanding of what happens in individual shots. A *data camera* is used during the video production process to record descriptive data of the video including time code, camera position, and voice annotation of who–what–why information. This approach is also called *source annotation* (3). However, the model doesn't address the problem of converting this annotation information into textual descriptions to create indexes of the video data. It is common to simply use a preselected set of keywords (31) for video annotation. This approach, however, is criticized for a number of reasons (2,60). First, it is not possible to use only keywords to describe the spatial and temporal relationships, as well as other information contained in the video data. Second, keywords cannot fully represent semantic information in the video data and do not support inheritance, similarity, or inference between descriptors. Keywords also do not describe the relations between descriptions. Finally, keywords do not scale; that is, the greater the number of keywords used to describe the video data, the lesser the chance the video data will match the query condition. A more flexible and powerful approach is to allow arbitrary free text video annotations (18,32,6) which are based on logical data abstractions. Jiang et al. (18) also further address the problem of integrating knowledge-based information retrieval systems with video database to support video knowledge inferencing and temporal relationship constraints. Another way to overcome the difficulties of keyword annotations is suggested by an annotation system called *Media Stream* (60,62). Media Stream allows users to create multilayer, iconic annotations of the video data. The system has three main user interfaces: Director's Workshop, icon palettes, and media time lines for users to annotate the video. Director's Workshop allows users to browse and compound predefined icon primitives into iconic descriptors by cascading hierarchical structure. Iconic descriptors are then grouped into one or more icon palettes and can be dropped into a media time line. The media time line represents the temporal nature of the video data, and the video is thus annotated by a media time line of icon descriptors. The creation of video indices however, is not discussed.

The spatiotemporal relationships between objects or features in a video data can also be symbolically represented by *spatial temporal logic (STL)* (63). The spatial logic operators include *before, after, overlaps, adjacent, contained,* and *partially intersects.* Temporal logical operators include *eventually* and *until.* Standard boolean operators are also supported including *and, or,* and *not.* The symbolic description, which is a set of STL assertions, describes the ordering relationships among the objects in a video. The symbolic description is created for, and stored together with, each video data in the database and serves as an index. The symbolic description is checked when a user query is processed to determine matches.

Feature-Based Indexing

Feature-based indexing techniques depend mainly on image processing algorithms to segment video, to identify representing frames (RFrames), and to extract key features from the video shots or RFrames. Indexes can then be built based on key features such as color, texture, object motion, and so on. The advantage of this approach is that video indexing can be done automatically. Its primary limitation is the lack of semantics attached to the features which are needed for answering semantic content-based queries. One of the simplest approaches is to index video based upon visual features of its RFrames. A video stream can first be segmented into shots which can be visually represented by a set of selected RFrames. The RFrames are then used as indices into these shots. The similarity comparison of the RFrames can be based on the combination of several image features such as object shapes measured by the gray level moments and color histograms of the RFrames (64). This approach can be a very efficient way of indexing video data; however, types of query are limited due to the fact that video indexing and retrieval are completely based on the computation of image features.

Video data can also be indexed on the objects and object motions which can be either interactively annotated or automatically extracted using motion extraction algorithms such as optical flow methods and block matching estimation techniques (65). Object motions can be represented by different combinations of primitive motions such as north and rotate-to-left, or motion vectors (65). Motion vectors can then mapped by using spatiotemporal space ($x - y - t$) and are aggregated into several representative vectors using statistical analysis. Objects and their motion information are stored in a description file or a labeling record as an index to the corresponding video sequence. Notice that each record also needs to have a time interval during which the object appears.

Multiple image features can be used simultaneously to index video data. They are often computed and grouped together as multidimensional vectors. For example, features used in MediaBENCH (66) include average intensity, representative hue values which are the top two hue histogram frequencies of each frame, and camera work parameters created by extracting camera motions from video (67). These values are computed every three frames in order to achieve real-time video indexing. Indexes are stored along with pointers to the corresponding video contents. Video data can be segmented into shots using a SCD algorithm based on index filtering by examining indices frame by frame and noticing the inter-frame differences. Thus, a structured video representation can be built to facilitate video browsing and retrieval operations.

Domain-Specific Indexing

Domain-specific indexing approaches use the logical (high-level) video structure models, such as the anchorperson shot model and CNN "Headline News" unit model, to further process the low-level video feature extraction and analysis results. After logical video data units have been identified, certain semantic information can be attached to each of them, and domain specific indices can be built. These techniques are effective in their intended domain of application. The primary limitation is their narrow range of applicability, and limited semantic information can be extracted. Most current research uses collections of well-structured video such as news broadcast video as input. One of the early efforts with domain-specific video indexing was done by Swanberg et al. (14) in the domain of CNN news broadcasting video. Several logical video data models that are specific to news broadcasting (including the anchorperson shot model, the CNN news episode models, and so on) are proposed and used to identify these logical video data units. These models contain both spatial and temporal ordering of the key features, as well as different types of shots. For example, the anchorperson shot model is based on the location of a set of features including the icon "Headline News" and the titling of the anchorperson. Image-processing routines, including image comparison, object detection, and tracking, are used to segment the video into shots and interactively extract the key elements of the video data model from the video data.

Hampapur et al. (21) proposed a methodology for designing feature-based indexing schemes which uses low-level image-sequence features in a feature-based classification formalism to arrive at a machine-derived index. A mapping between the machine-derived index and the desired index was designed using domain constraints. An efficacy measure was proposed to evaluate this mapping. The indexing scheme was implemented and tested on cable TV video data. Similarly, Smoliar et al. (22,68,69) used an *a priori* model of a video structure based on domain knowledge to parse and index the news program. A given video stream is parsed to identify the key features of the video shots, which are then compared with domain-specific models to classify them. Both textual and visual indexes are built. The textual index uses a category tree and assigns news items to the topic category tree. The visual index is built during the parsing process, and each news item is represented as a visual icon inside a window that provides an unstructured index of the video database. A low-level index that indexes the key frames of video data is also built automatically. The features used for indexing include color, size, location, and shape of segmented regions and the color histograms of the entire image and nine subregions that are coded into numerical keys.

VIDEO QUERY, RETRIEVAL, AND BROWSING

The purpose of a video database management system (VDBMS) is to provide efficient and convenient user access to a video data collection. Such access normally includes query, retrieval, and browsing. The video data query and retrieval process typically involves the following steps. First, the user specifies a query using facilities provided by a GUI; this query is then processed and evaluated. The value or feature obtained is used to match and retrieve the video data stored in video database. In the end, the resulting video data is presented to the user in a suitable form. Video query is closely related to other aspects of VDBMS, such as video data indexing, since features used for indexing are also used to evaluate the query, and the query is usually processed by searching the indexing structure. Unlike alpha-numerical databases, video database browsing is critical due to the fact that a video database may contain thousands of video streams with great complexity of video data. It is also important to realize that video browsing and querying are very closely related to each other in the video databases. In a video database system, a user's data access pattern is basically a loop of the "query-browse" process in which video queries and video browsing are intermingled. Playing video data can be thought of as the result of the process.

User video browsing usually starts with a video query about certain subjects that the user is interested in. This makes the browsing more focused and efficient since browsing a list of all of the video streams in a video database is time and network resources consuming. Such initial queries can be very simple because the user isn't familiar with the database content. On the other hand, a video query normally ends with the user browsing through the query results. This is due to the ambiguous nature of the video query; that is, it results in multiple video streams, some of which are not what the user wanted. Browsing is a efficient way of excluding unwanted results and examining the contents of possible candidates before requesting the playing of a video.

Different Types of Queries

Identifying different classes of user queries in a VDBMS is vital to the design of video query processing. The classification of the queries in a video database system can be done in many ways depending on intended applications and the data model they are based on, as well as other factors (3).

A video query can be a *semantic information query, meta information query,* or *audio–visual query*. A semantic information query requires an understanding of the semantic content of the video data. For an example, "find a video clip in which there is an angry man." This is the most difficult type of query for a video database. It can be partially solved by semantic annotation of the video data, but its ultimate solution depends on the development of technologies such as computer vision, machine learning, and artificial intelligence (AI). A meta information query is a query about video meta data, such as who is the producer and what is the date of production. In most cases, this kind of query can be answered in a way that is similar to the conventional database queries. Meta data are usually inserted into the video database along with the corresponding video data by video annotation that is currently manually or semi-manually done off-line. An example of a query could be to find video directed by Alan Smithee and titled "2127: A Cenobite Space Odyssey." This class also includes *statistical queries,* which are used to gather the information about video data without content analysis of the video. A typical example is to find the number of films in the database in which Tom Cruise has appeared. An audiovisual query depends on the audio and visual features of the video and usually doesn't require understanding the video data. An example of such a query would be to find a video clip with dissolve scene change. In those queries, audio and visual feature analysis and computation, as well as the similarity measurement, are the key operations as compared to the textual queries in the conventional DBMS.

Video queries can also based on the spatiotemporal characteristics of the video content as well. A video query can be about spatial, temporal, or both kind of information. An example of a *spatial query* is "retrieve all video clips with a sunset scene as background," and the query "find all video clips with people running side by side" is a *spatiotemporal query* example.

Depending on how a match is determined in the query evaluation, a video query can be classified as either an *exact match-based query* or a *similarity match-based query*. An exact match-based query is used to obtain an exact match of the data. One example is to find a CNN "Dollars and Sense" news clip from the morning of March 18, 1996. Similarity match-based queries actually dominate the VCBMS because of the ambiguity nature of the video data. One example is "find a video clip that contains a scene that is similar to the given image."

A video query can have various *query granularity* which can be either video frames, clips, or streams. A *frame-based query* is aimed at individual frames of video data that are usually the atomic unit of the video database. A *clip-based query* is used to retrieve one or more subsets of video streams that are relatively independent in terms of their contents. A *video stream-based query* deals with complete video streams in the database. An example query is "find a video produced in 1996 that has Kurt Russell as the leading actor." Queries can also be categorized according to *query behavior*. A *deterministic query* usually has very specific query conditions. In this case, the user has a clear idea what the expected result should be. A *browsing query* is used when a user is uncertain about his or her retrieval needs or is unfamiliar with the structures and types of information available in the video database. In such cases, the user may be interested in browsing the database rather than searching for a specific entity. The system should allow for the formulation of fuzzy queries for such purpose.

There are many ways that a user can specify a video query. A *direct query* is defined by the user using values of features of certain frames, such as color, texture, and camera position. A *query by example,* which is also called *query by pictorial example* (QBPE) or *Iconic Query* (IQ), is very useful since visual information is very difficult to describe in words or numbers. The user can supply a sample frame image as well as other optional qualitative information as a query input. The system will return to the user a specified number of the best-match frames. The kind of query methodology is used in IBM's QBIC system (70) and JACOB system (71). In an *iterative query,* the user uses a GUI to incrementally refine their queries until a satisfying result is obtained. The JACOB system is a practical example of this approach.

Query Specification and Processing

Video Query Language. Most textual query languages such as SQL have limited expressive power when it comes to specifying video database queries. The primary reason is that the visual, temporal, and spatial information of the video data can not be readily structured into fields and often has a variable-depth, complex, nested character. In a video database, queries about visual features can be specified, for examples, by using an iterative QBPE mechanism; and spatiotemporal queries can be expressed, for example, by TSQL or spatial temporal logic (STL).

Queries dealing with the relationships of video intervals can be specified using a temporal query language like *TSQL* (TSQL 2, Applied TSQL2) (72,73). TSQL2 has been shown to be upward compatible with SQL-92 and can be viewed as an extension of SQL-92. However, not all SQL-92 relations can be generated by taking the time slices of TSQL2 relations, and not all SQL-92 queries have a counterpart in TSQL-92. The completeness and evaluation of the TSQL2 are discussed by Bohlen et al (72). STL (74) is proposed as a symbolic representation of video content, and it permits intentional ambiguity and detail refinement in the queries. Users can define a query through an iconic interface and create sample dynamic scenes reproducing the contents of the video to be retrieved. The sample scenes are then automatically translated and interpreted into STL assertions. The retrieval is carried out by checking the query STL assertions against the descriptions of every image sequence stored in the database. The description of a video sequence is used to define the object-centered spatial relationship between any pair of objects in every frame and created manually when the sequence is stored in the database.

The VideoSTAR system uses a video query algebra (15,75) to define queries based on temporal relationships between video intervals. A GUI is developed to assist users interactively define queries with algebra operations include (a) normal set operations (AND, OR, and DIFFERENCE), (b) temporal set operations, (c) filter operations that are used to determine the temporal relationships between two intervals, (d) annotation operations that are used to retrieve all annotations of a given type and have nonempty intersections with a given input set, (e) structure operations that are similar to the above but on the structural components, and (f) mapping operations that map the elements in a given set onto different contexts that can be basic, primary, or video stream. VideoSQL is the video query language used in OVID (35), which allows users to retrieve video objects that satisfying certain conditions through SELECT-FROM-WHERE clauses. Video SQL does not, however, contain language expressions for specifying temporal relations between video objects.

Other Video Query Specifications. Despite its expressive power and formalism, defining and using certain video query language can often become very complex and computationally expensive. Some researchers simply combine important features of the video data to form and carry out queries. In these cases, the types of queries that can be defined and processed are usually limited. For an example, the MovEase system (76) includes motion information as one of the main features of the video data. Motion information, together with other video features (color, shape, object, position, and so on), is used to formulate a query. Objects and their motion information (path, speed) can be described through a GUI in which objects are represented as a set of thumbnail icon images. Object and camera motions can be specified by using either predefined generic terms like pan, zoom, up, down, or user input-defined motion descriptions, such as zigzag path. The query is then processed and matched against the preannotated video data stored. Results of the query are displayed as icons, and users can get meta information or the video represented by each icon image simply by clicking on it.

Query Processing. Query processing usually involves query parsing, query evaluation, database index search, and the returning of results. In the query parsing step, the query condition or assertion is usually decomposed into the basic unit and then evaluated. After that, the index structure of video database is searched and checked. The video data are retrieved if the query assertion is satisfied (74) or if the similarity measurement (65) is maximum. The result video data are usually displayed by a GUI in a way convenient to the user [such as iconic images (76)]. One example is an on-line object-oriented query processing technique (77) which uses generalized n-ary operations for modeling both spatial and temporal contents of video frames. This enables a unified methodology for handling content-based spatial and spatiotemporal queries. In addition, the work devises a unified object-oriented interface for users with a heterogeneous view to specify queries. Another example is the VideoSTAR system (75) which parses the query and breaks it into basic algebraic operations. Then, a query plan is determined and many be used to optimize the query before it is computed. Finally, the resulting video objects are retrieved.

VIDEO AUTHORING AND EDITING

Digital video (DV) authoring usually consists of three steps: *video capture and digitization, video editing,* and *final production.* In the video capture step, raw video footage can be captured or recorded in either analog format or digital format. In the first case, the analog video needs to be digitized using a *video capture board* on the computer. The digital video is usually stored in a compressed format such as MPEG, MJPEG, DV, and so on. Analog recording, digitization, and editing using video capture cards and software tend to suffer from information loss during the conversion. However, this approach is very important since the majority of the existing video materials are on video tapes and films. According to an international survey (1), there are more than 6 million hours of feature films and video archived worldwide with a yearly increase of about 10%. With the appearance of the DV camcorder, especially those with the Firewire interface, a superior digital video authoring and editing solution finally comes into reality.

DV editing refers to the process of rearranging, assembling, and/or modifying raw video footage (or clips) obtained in the video capture step according to the project design. The raw video clips which may not come from the same resources can be trimmed, segmented, and assembled together on a time line in the video construction window. Possible edits also include transitions and filters, as well as many other operations such as title superimposition. Special effect transitions are commonly used for assembling video clips, which include various wipes and dissolves, 3-D vortex, page peel, and many others. Filters including video and audio filters can be used to change the visual appearance and sound of video clips. The examples of filters are *Gaussian sharpen, ghosting, flip, hue, saturation, lightness,* and *mirror*. During the video editing process, the user can preview the result in the software window on the computer screen or on an attached TV monitor. Digital video editing can be classified as linear or nonlinear, described in more detail later. In the final production step, the final editing results can be recorded back on a video tape or a CD. The final format of the video production depends on the intended application, for example, One should choose MPEG-1 video compression for CD application and MPEG-2 for TV quality video playback. In any case it is a good idea to keep the original DV tape or analog (Hi-8 or VHS) tape.

Video capture board mentioned above is one of the key components of a video editing system and is responsible for digitizing analog video input into digital ones for desktop digital video editing. It is also widely used in other applications such as video conferencing. Some of the common or expected features of a video capture board are listed in the following. The actual features depend on each individual card and can make the card very expensive.

- Real-time, full-screen (640×480 NTSC, 768×576 PAL), true color (24 bits), and full motion (30 frames/s, 25 frames/s PAL) capture and playback of NTSC, PAL, or SECAM analog video.
- Analog output in NTSC, PAL, or SECAM in composite or S-video. This feature can be used to output the editing result back on to the video tape or preview the editing result on a TV monitor.
- Support for multiple sampling rate audio data, along with the ability to record and play audio from voice grade to CD/DAT stereo quality. It also need to support the sychronization of the video and audio channels
- Hardware support for video compression standards such as MJPEG, MPEG, and ITU H.261. It needs to also support audio compression standards (G.711, G.722, and G.728) and be compatible with QuickTime or AVI.
- Software and developing tools for video editing and video conferencing, and so on.

Linear Digital Video Editing

Linear video editing systems are usually hardware-based and require edits to be made in a linear fashion. The concept behind linear editing is simple: The raw video footage which may be recorded on several tapes is transferred segment by segment from *source machine*(*s*) onto a tape in another video recorder. In the process, the original segments can be trimmed and rearranged, unwanted shots can be removed, and audio and video effects can be added. An *edit recorder,* controlled by an *editing controller,* is used to control all of the machines and make final *edit master.* The edit controller can be used to shuttle tapes back and forth to locate the beginning and ending points of each needed video segment. These reference points are entered as either control track marks or time code numbers into the edit controller to make edits.

There are two types of linear edits. *Assemble editing* allows video and audio segment to be added one after another, complete with their associated control track. However, the control track is difficult to record without any error during video edits. For example, any mis-timing during this mechanical process results in a glitch in the video. *Insert editing* requires a stable control track to be established first for stable playback. Video and audio segments can then be inserted over the prerecorded control track. Linear video editing is generally considered slow and inflexible. Although video and audio segments can be replaced within the given time constraints of the edited master, it is impossible to change the length of segments or insert shots into the edited master without starting all over again. This can be easily done with the more flexible and powerful nonlinear video editing.

Despite its limitations, linear video editing is nonetheless an abandoned solution and it is still used even for DV editing (78) for a number of reasons. First, when editing long video programs, linear editing may actually save time when compared to the nonlinear editing. This is because, for example, there is no need to transfer video data back and forth between the video tapes and computer. Second, long digital video programs occupy a huge amount of disk space. A 1 h DV, for example, fills about 13 Gbyte-space. The file size constraints of the computer operating system limit the length of the video footage that can be placed on the disk and operated by nonlinear editors. So, the choice of linear or nonlinear editing is really application-dependent. The best solution may be a combination of both.

Nonlinear Video Editing

Nonlinear video editing (*NLE*) is sometimes called *random-access video editing,* which is made possible through digital video technologies. Large-capacity and high-speed disks are often used as the recording medium and video footage are stored in either compressed or uncompressed digital format. NLE supports random, accurate, and instant access to any video shot or frame in a video footage. It also allows the video segments to be inserted, deleted, cut, and moved around at any given point in the editing process. Nonlinear video editing supports a much wider range of special effects such as fades, dissolves, annotation, and scene-to-scene color corrections. It also supports many audio enhancement including audio filters and sound effects.

Most NLE systems have multiple *time lines* to indicate the simultaneous presence of multiple audio and video sources. For example, one could have background music, the original sound track of the raw footage, and the voice of the narrator at the same time. One can instantly preview and make adjustments to the result at any give point of the NLE process. The video and audio segments can be clicked and dragged to be assembled on a designated time line. Video segments are often represented by thumbnail icons of its video frames with adjustable temporal resolutions (one icon per 100 frames, for example). The results of nonlinear video editing can be converted into analog video signals and output back to a video tape, or stored in any given digital video format.

Digital Video Camcorder and Digital Video Editing

Using a DV camcorder, video is digitally captured and compressed into DV format before it is recorded onto the DV tape. There are two ways that the DV footage can be edited. One can still connect the analog output of the DV camcorder/VCR to the video capture board on the computer and edit the video as previously discussed. However, this approach is not recommended due to the quality loss in A/D (analog-to-digital) conversions and lossy codecs used in DV equipment. True end-to-end high-quality digital video editing can be done using DV equipment (VCR or camcorder) and the Firewire (IEEE 1394). A single Firewire cable can carry all the DV data between DV devices and the computer including video, audio, and device control signals. It eliminates multiple cables required in the traditional digital video authoring and editing system. Sony first introduced the DV camcorders with the Firewire connector. This approach has no generation loss and is not necessarily more expensive than the first method. One may need to purchase a Firewire interface board; however, its price may be cheaper than many video capture boards.

Firewire—IEEE 1394. Firewire (79), officially known as IEEE 1394, is a high-performance digital serial interface standard. Originated by Apple for desktop local area networks (LANs), it was later developed and approved in December 1995 by IEEE. IEEE 1394 supports data transfer rates of 12.5, 25, 50, 100, 200, and 400 Mbit/s which can easily meet the requirements of DV data transportation or even uncompressed digital video data at 250 Mbit/s. Data rate over 1 Gbit/s is under design. Other key advantages of IEEE 1394 include the following:

- It is supported by 1394 Trade Association which has over 40 companies including Apple, IBM, Sun, Microsoft, Sony, and Texas Instruments. For example, Apple is the first to support Firewire in its operating system (Mac OS 7.6 and up) and provide Firewire API 1.0 in Mac OS 8.0.
- It is a digital interface; there is no A/D conversion and data integrity loss.
- It is physically small (thin serial cable), easy to use (no need for terminator and device ID, etc.), and hot pluggable. Hot pluggable means that 1394 devices can be added to or removed from the IEEE 1394 bus at any time, even when the bus in full operation.
- It has scalable architecture which allows for the mixture of data rates on a single bus.
- It has flexible topology which supports daisy chaining and branching for true peer-to-peer communication. Peer-to-peer communication allows direct dubbing from one camcorder to another as well as sharing a camcorder among multiple computers.
- It supports asynchronous data transport which provides connectivity between computers and peripherals such as printers and modems and provides command and control for new devices such as DV camcorders.
- It also supports isochronous data transport guarantees delivery of multiple time-critical multimedia data streams at predetermined rates. Such just-in-time data delivery also eliminates the need for costly buffering.

The current standard allows Firewire cable up to 4.5 m per hop; but with repeaters or bridges, over 1000 bus segments can be connected and thus can reach thousands of meters. Each firewire cable contains two power conductors and two twisted pairs for data signaling. Signal pairs are shielded separately; additionally, the entire cable is also shielded. The Firewire cable power is specified to be from 8 V dc to 40 V dc at up to 1.5 A. It is used to maintain a device's physical layer continuity when the device is powered down or malfunctions and provide power for the devices connected to the bus. However, some manufacturers may have sightly different cables; for example, the Sony camcorder Firewire cable only has four wires with two power wires removed.

Firewire is widely used for attaching DV camcorders to computers and as a high-performance, cost-effective digital interface for many other audio/video applications such as digital TV and Multimedia CDROM (MMCD). IEEE 1394 has been accepted as the standard digital interface by the Digital VCR Conference (DVC) and has been endorsed by European Digital Video Broadcasters (DVB) as their digital TV interface as well. The EIA (Electronic Industries Association) has also approved IEEE 1394 has the point-to-point interface for digital TV and the multipoint interface for entertainment systems. In the future, IEEE 1394, as a high-speed, low-cost, and user-friendly interface, is expected to improve existing interfaces such as SCSI. In fact, the American National Standards Institute (ANSI) has already defined Serial Bus Protocol (SBP) to encapsulate SCSI-3 for IEEE 1394.

Various DV Video Format. DV is a digital video format (80) developed by DVC and adopted by over 50 manufacturers including Sony, Panasonic, JVC, Philips, Toshiba, Hitachi, Sharp, Thomson, Sanyo, and Mitsubishi. The DV specification was approved in September 1993 and is intended primarily for prosumer, eventually consumer applications. The DV format offer two tape cassette sizes: the standard 4 h (125 mm $\times$ 78 mm $\times$ 14.6 mm) and the mini 1 h (66 mm $\times$ 48 mm $\times$ 12.2 mm). Most of the DV VCRs will play both.

The DV video compression algorithm is DCT-based and very similar to that of MPEG and MJPEG. First, RGB video is converted to a YUV digital component video. The luminance signal (Y) is sampled at 13.5 MHz, which provides a 5.75 luminance bandwidth for both the NTSC and PAL systems. For NTSC video, the R–Y (U) and B–Y (V) color difference signals are digitized at 3.375 MHz sampling rate, which provides a 1.5 MHz bandwidth for each chroma component. The result is 4 : 1 : 1 digital video. The PAL DV system samples each chroma component at 6.775 MHz yields there by a 3.0 MHz bandwidth per chroma component. However, PAL DV uses a 4 : 2 : 0 sampling schema that yields only half of the vertical chroma resolution of the NTSC DV format.

Before compression, digital video frames are stored in a 720 $\times$ 480 pixel buffer where the correlation between two fields are measured. Two fields are compressed together unless the correlation is low, which indicates too much interfield motion. Each DCT macroblock consisting of four 8 $\times$ 8 blocks has its own quantization table (Q-table), which enables dynamic intraframe compression. The DV formation has a standard set of Q-tables. DV video compression ratio is 5 : 1. DV provides two digital audio record modes: 16-bit and 12-bit.

The 16-bit mode uses a sampling frequency of 48 kHz and 12-bit mode operates at 32 kHz.

DV format uses Reed–Solomon error correction on the buffered video data to prevent frame loss. Each DV track consists of four sectors: subcode, video, audio, and ITL. *Subcode sector* records timecode, an index ID for quick searches for specific scenes, and the PP-ID for Photo Mode recording and playback. *Video sector* records not only the video data but also the auxiliary data such as data and time, focus mode, AE-mode, shutter speed F-stop, and gain setting. *ITI sector* stores data for the DV device itself, such as tracking signal for audio dubbing. The separation of audio and video signals makes video-only insert editing possible.

DVCPRO is a professional variant of the DV by Panasonic. The main differences are the doubled tape speed needed for dropout tolerance and general recording robustness. It is also capable of 4× normal speed playback which can be used to accelerate data transfer. DVCAM is Sony's DV variation. DV and DVCAM uses 4 : 2 : 0, and DVCPRO uses 4 : 1 : 1 sampling rates for PAL. They all use 4 : 1 : 1 for NTSC and have a data rate of 25 Mbps. Panasonic also has *DVCPRO-50* for the studio-quality video. Unlike DV, DVCPRO, and DVCAM which sample at 4 : 1 : 1, DVCPRO-50 provides a 4 : 2 : 2 sampling which is consistent with ITU-R BT.601-4 (CCIR-601) digital video standard. Such a sampling rate is sometimes preferred since it provides more color information and better compositing. The data rate of DVCPRO-50 is 50 Mbps, which is twice that of DV, and it supports lightly compressed picture (3.3 : 1) with a high signal-to-noise ratio. JVC's Digital-S is another 50 Mbps video format. Together, they are known as DV422 and are compatible with each other. The 4 : 1 : 1 DV tapes can be played on the CV422 decks which can bump the output to 4 : 2 : 2 for post-production uses. Another advantage of DV422 is that it is closer to the MPEG-2 standard which samples at 4 : 2 : 0. Sony's Betacam SX is yet another DV video format targeted at professional market. Betacam SX is similar to MPEG-2 and uses adaptive quantization and MPEG-2's IB frame (IBIB. . .) compression to achieve a constant data rate of 18 Mbps with 4 : 2 : 2 sampling. Betacam SX thus has a higher compression ratio of 9.25 : 1.

DV Board. DV boards are sometimes referred to as Firewire interface boards. This is because the Firewire interface is the most important component on the board since it enables the fast DV data transmission between the computer and DV equipment. Besides Firewire interface, a DV board usually contains the following:

- DV codecs. Some DV boards come with software codecs that use the computer processor to decompress DV files for preview and editing. Software codecs are cost-effective, flexible, and easy to be upgraded. Other DV boards have a DV codec chip which frees the CPU from the compression/decompression procession and can be fast enough for full-motion, real-time playback. However, they are also much more expensive. Notice that a software DV codec can also make use of the hardware codec in the DV equipment connecting to the DV board.
- Analog video/audio I/O ports. They are especially useful for previewing the DV on an analog TV monitor and mixing the analog video footage with DV files or converting analog video footage to DV format. The DV board may also contain a chip that can compress the analog video to the DV or MPJEG digital video format. In this case, the DV board functions like the video capture board previously described. It usually supports full resolution, true color, and real-time compression of analog video.
- Additional Firewire ports for connecting other Firewire peripherals such as a printer. They can also be used for synchronized video/audio playback and VCR control with time code for accurate video editing.

End-to-End Digital Video Editing Using DV and Firewire. Editing DV with Firewire (78,81) requires DV equipment such as DV camcorder or DVCR with Firewire I/O port. A Firewire cable is used to connect the equipment to the computer which has a DV board with Firewire interface. Editing is done by using a nonlinear video editing software such as Adobe Premiere. The computer needs to have sufficient processor power and large amount of disk space. The data rate of DV is usually 3.7 Mbit/s, which means 222 Mbyte space per minute and 20 Gbyte for a 90 min DV footage. The disk drive also needs to be fast enough to accommodate the steady DV stream of 3.7 Mbit/s. Digital video authoring and editing using DV with Firewire consists of the following steps:

Step 1. Shoot the video footage using a DV camcorder. As the video is being shot, it is compressed by the DV codec chips in the DV camcorder and recorded digitally on a DV tape which can also be played by the DVCR.

Step 2. DV footage can be then transferred into the computer and stored on a hard disk through the Firewire which is connected to the Firewire interface of computer's DV board. During the transferring process, the DV data is usually encapsulated into certain multimedia systems such as AVI or QuickTime. DV codecs are not involved during the transfer.

Step 3. Video editing software such as Adobe Premiere can be used to work with DV data which are now encapsulated in some multimedia system format. Notice that the DV data in the computer so far are identical to what is on the DV tape; that is, no information is lost. The DV codec is only for decompressing the DV data when filters and/or transitions are to be added. Otherwise, the DV data are simply copied to the target file. The DV codec can be software or hardware on the DV board. During the editing process, the video can be previewed either on the computer screen or on a monitor. Monitor preview is usually supported by the analog port on DV board, DV camcorder, or DVCR. Such a Firewire interface board needs to have a DV codec hardware which increases the cost considerably.

Step 4. After all the edits are done, the resulting DV file can be transferred back to the DV equipment via Firewire. It is obvious that the whole editing process has no generation loss. The result can also be transcoded into other digital formats such as MPEG, or outputs to Hi-8 or VHS tapes. The latter can be done through the analog I/O port of the DV board, DV camcorder, or DVCR.

DV footage can be easily mixed with analog footage during above NLE process. If the DV camcorder or DVCR has analog input, the analog footage can be transferred into the DV camcorder and then digitized, compressed, and recorded in the DV format. Another way is to make use of the analog I/O port on the DV board with hardware DV codec. Such a board is capable of converting analog video to DV, but costs more. The third approach is to use the video capture board to digitize the analog video into MJPEG digital video clips. Such video clips can then be transcoded by video editing software into DV format when a DV codec is presented.

The advantages of video authoring and editing using DV and Firewire are obvious. First, the video is of high quality and free of noises. Experiments (78) show that DV video still has high quality (better than Betacam SP video digitized at highest quality) even after being decompressed and recompressed 10 times. Second, the high-quality video also tends to be compressed better and is more tolerable to lower data rate. Third, DV has a steady data rate of 3.7 Mbit/s which is easier to handle and results good playback. This approach is also cost-effective since codec is hardwared inside the DV camcorder and DVCR. There is simply no need for an expensive video capture board.

VIDEO CONFERENCING

Video conferencing refers to the interactive digital video and audio communication between a group of parties who may be remotely located through the use of computers over computer networks (82). Video conferencing is generally considered one type of data conferencing which also includes text and graphics, and so on. Video conferencing has many important applications, such as tele-medicine and distance learning. Video conferencing requires real-time capture, sampling, coding, and transmission of both audio and video. Compression is critical to video conferencing due to the huge data volume involved. For example, an uncompressed full motion CIE frame size video stream needs a bandwidth of 30 $frame \times (352 \times 288)\ pixel/frame \times 8\ bit/pixel = 24\ Mbit/s$. Some important video codecs are described in the section entitled "Video Codecs." The audio analog signal is usually sampled at a rate range from 8 kHz to 48 kHz. This is based on the Nyquist theory since human hearing range is 20 Hz to 20 kHz, and human voice ranges from 40 Hz to 4 kHz. Sampled values are then quantized into a number of discrete levels (256 for 8-bit representation, or 65536 for 16-bit representation) and then coded using following methods:

- PCM (pulse code modulation), which includes uniform PCM, mu-law PCM, and A-law PCM. Uniform PCM uses equally spaced quantizer values and is an uncompressed audio encoding. Au-law and A-law PCMs use logarithmic quantizer step spacing which can represent larger value range using the same number of bits. Mu-law and A-law PCMs can achieve a compression ratio of 1.75 : 1, and they are formally defined in the IUT-T Recommendation G.711.
- ADPCM (adaptive difference pulse code modulation) encodes the difference between each sample and its predicted value based on the previous sample. The quantizing and prediction parameters of ADPCM are adaptive to the signal characteristics, and ADPCM typically can achieve a compression ratio of 2 : 1. There are several ITU-T recommendations which specify different ADPCM audio encoding algorithms, including G.721, G.722, G.723, G.726, and G.727.

Unipoint and Multicast Video Conferencing

Video conferencing can be categorized in several ways. Depending on the number of parties involved, a video conference can be either *point-to-point* or *multipoint*. Point-to-point (circuit-switched) or unicast (packet-switched) video conferencing is the simplest form of video conference which involves only two sites. Both parties of a point-to-point must use the same video/audio coding algorithms and operate at the same speed. Multipoint (circuit-based) or multicast (packet-switched) video conference involves multiple parties. In the circuit-based multipoint video conferencing, each party talks to an MCU (multipoint control unit). For the packet-based video conference, a somewhat analogous software tool called the MSB (multisession bridge) is needed. MCU uses the following methods to switch between each video conferencing participant:

1. *Potting.* MCU switches between participants at certain time interval.
2. *Voice Active Switching or Picture Follows Voice.* MCU switches to the participant who has highest audio level.
3. *Continuous Presence.* MCU divides the window into several subwindows, one for each participant.
4. *Chair Control.* MCU always presents the picture of the participant who is designated as the chair of the conference.

ITU-T Recommendation H.231 is a standard that covers MCU and defines how several H.320-compatible video conferencing system can be linked together. H.243 defines the MCU protocols. In multipoint video conferencing, all codecs must be mutually compatible, and the MCU must be compatible with the codecs. The video conference operates at the smallest frame size (FCIF or QCIF) and the lowest bandwidth of any of the nodes and node-MCU links.

Packet-Switched and Circuit-Switched Video Conferencing

Video conferencing can also be distinguished by the way the data are transmitted over the network: packet-switched or circuit-switched.

Packet-Switched Video Conferencing. Packet-switched communication is a method of data transfer where the information is divided into packets, each of which has an identification and destination address. Packets are sent individually through a network and, depending on network conditions, may take different routes to arrive at their destination at different times and out of order. Unlike circuit-switched communication, bandwidth must be shared with others on the same network. In the packet-switched video conferencing, the data can be transmitted over the Internet (e.g., using MBONE or Multicast BackbONE). The general bandwidth requirement is 192 kbit/s, in which 128 kbit/s is for video and 64 kbit/s is for audio.

An advantage of packet-switched communication for video conferencing is the capability to more easily accommodate multipoint conferences. A disadvantage is the unpredictable timing of data delivery, which can cause problems for delay-sensitive data types such as voice and video. Video packets that are received out of order may have to be discarded. Audio packets can be buffered at the receiver, reordered, and played out at a constant rate; however, this induces a delay which can be detrimental to interactive communication.

Circuit-Switched Video Conferencing. Circuit-switched communication is a method of data transfer where a path of communication is established and reserved for the duration of the session. A dedicated amount of bandwidth is allocated for exclusive use during the session. When the session is completed, the bandwidth is freed and becomes available for other sessions. Advantages of circuit-based communication for video conferencing include the availability of dedicated bandwidth and predictability of data delivery. A disadvantage is that the session is primarily point-to-point and requires expensive MCUs to accommodate multipoint conferences. Also, the dedicated bandwidth tends to be wasted during periods of limited activity in a conference session. The general bandwidth requirement for a circuit-based video conferencing over POTN (Plain Old Telephone Network) is 128 kbit/s (video 108 kbit/s, audio: 16 kbit/s, overhead: 4 kbit/s).

Video Conferencing Over Various Networks

Video conferencing can be classified based on the communication network it uses.

POTS-Based Video Conferencing. POTS (Plain Old Telephone Service) is the basic telephone service that provides access to the POTN. POTS is widely available but has very low bandwidth (the total bandwidth of a V.34 modem is only 36.6 kbit/s). ITU-T Recommendation H.324 is an interoperability standard for video conferencing operating over V.34 modem (33.6 kbit/s). H.324 uses H.263 for video encoding and G.723 for audio codec (please refer to the section entitled "H.323").

ISDN-Based Video Conferencing. ISDN (integrated services digital network) is a digital service over the public switched network. ISDN has two access rates: basic rate interface (BRI) and primary rate interface (PRI). BRI provides two data channels of 64 kbit/s (B-channels) and one signaling channel of 16 kbit/s (D-channel). ISDN PRI provides 23 or 30 B channels of 64 kbit/s and one D-channel of 64 kbit/s, but is much more expensive. ITU-T H.320 is the interoperability standard for ISDN-based video conferencing. It uses H.261 as the video codec and G.711 and G.728 for audio codec (please refer to the section entitled "H.320").

B-ISDN-Based Video Conferencing. B-ISDN (broadband ISDN) is the high-speed and broadband extension of the ISDN. It is a concept as well as a set of services and developing standards for integrating digital transmission services over the broadband network of fiber-optic and radio media. B-ISDN provides bandwidth range from 2 Mbit/s to 155 Mbit/s and up. It uses a fast cell-switching protocol called *Asynchronous Transfer Mode* (*ATM*) (83,84) as the underlying data link layer protocol. ATM has many advantages for video conferencing: (a) high bandwidth available instantly on demand; (b) more efficient than circuit switch with statistical multiplexing which can combine many virtual circuits into one physical channel; (c) low cell delay variation which is good for real-time video and audio; (d) high resilience with dynamic alternative routing.

BISDN can also be used to interconnect LANs together to provide wide area video conferencing. The Integrated Service Working Group of IETF developed a best-effort, real-time Internet service model which includes RTP (Real-Time Transportation Protocol), RSVP (Resource Reservation Protocol), and RTCP (Real-Time Control Protocol). The interconnected LANs need to have these protocols and must be able to interwork with BISDN's access protocols such as AAL5.

ITU-T Recommendations H.321 and H.310 are the interoperability standards for BISDN-based video conferencing. H.321 (Adaptation of H.320 Visual Telephone Terminals to B-ISDN Environments, adopted in March 1996) describes technical specifications for adapting barrow-band visual telephone terminals defined by H.320 to B-ISDN. H.310 (Broadband Audiovisual Communication Systems and Terminals, adopted in November 1996) specifies technical requirements for both the unidirectional and bidirectional broadband audiovisual systems and terminals. With such high bandwidth, B-ISDN video conferencing uses MPEG-2/H.261 as the video codec and MPEG-1/MPEG-2/ITU G series for audio coding. Therefore, B-ISDN video conferencing can achieve very high video and audio quality. B-ISDN and ATM show great promise for video conferencing applications, but their deployment is currently limited.

LAN-Based Video Conferencing. The physical layer of LANs (local area networks) usually consist of 10 Mbps Ethernet, 100 fast Ethernet, or 4 or 16 Mbit/s Token Ring segments. With much more bandwidth available than ISDN, LAN video conferencing can achieve picture quality similar to that of television. However, bandwidth management and scalability for a large number of users becomes a problem since the network bandwidth is shared among all the participants and users in a LAN.

H.323 is the ITU-T recommendation for LAN-based video conferencing. It defines terminals, equipment, and services for multimedia conferencing over a network without a Quality-of-Service (QoS) guarantee such as a LAN. LAN-based video conferencing can also use UDP, RTP for point-to-point transmission of real-time video and audio, and RSVP, which works together with RTP. RSVP allows the router to reserve bandwidth for the smooth transmission of time-sensitive data such as video and audio.

Internet-Based Video Conferencing. The Internet uses IP (Internet Protocol) and two transportation layer protocols: TCP and UDP. TCP (Transmission Control Protocol) provides a reliable end-to-end service by using error recovery and reordering. UDP (User Datagram Protocol) is an unreliable service without error recovery capability (83). Internet video conferencing applications primarily use UDP for video and audio data transmission. TCP is not practical because of its error recovery mechanism. If lost packets were retransmitted, they would arrive too late to be of any use. TCP is used by video conferencing applications for other non-time-sensitive data such as whiteboard data and shared application data. Notice that UDP is an unreliable data transportation protocol; in other words, packets may be lost, duplicated, delayed, or out of order. All these may not be a problem for highly reliable and low-delay LANs, but will cause serious problems for wide area Internet video conferencing.

The above challenges of transmitting video and audio over the Internet has led to the development of a new transport protocol called Real-Time Transport Protocol (RTP) proposed by the IETF-AVT (Audio/Video Transport Working Group). RTP (RFC 1889) provides support for sequencing, time stamp, and QoS feedback. RTP is used in ITU-T Recommendation H.323. Most of the commonly used MBONE tools as well as video conferencing products on the market have implemented some version of RTP.

MBONE-Based Video Conferencing. MBONE (Multicast BackbONE) (85,86) is a virtual network that sits on top of the Internet and uses software multicast routers. Using the MBONE, it is possible to transmit video, audio, and other data in real time to multiple destinations throughout the global Internet. MBONE originated from the first two experiments to multicast live audio and video from meetings of the IETF (Internet Engineering Task Force) to other sites. Multicast has been implemented over LANs such as Ethernet and Fiber Distributed Data Interface (FDDI) and an Internet extension has been defined in RFC 1112 in 1989 (87). Basically, MBONE consists of "islands" supporting IP multicast such as multicast LANs like Ethernet, connected by point-to-point links called "tunnels."

With IP multicast, data are transmitted to a *host group* (83,87) which includes all the participating hosts. Each host group is specified by a class D IP address in the range of 244.0.0.0 to 239.255.255.255. Multicast routers are responsible for delivering the sender's data to all receivers in the destination group. The Internet Group Management Protocol (IGMP) is used by multicast routers to determine what groups are active on a particular subnet. There are several routing protocols that multicast routers can use to efficiently route the data packets, including Distance Vector Multicast Routing Protocol (DVMRP), Multicast Open Shortest Path First (MOSPF), and Protocol-Independent Multicast (PIM). If a router is not equipped with these routing protocols, it can use the *tunneling* technique, which means to encapsulate the multicast packet inside a regular IP packet and set the destination to another multicast router. Most major router vendors now support IP multicast.

Interoperability Standards

Interoperability standards are required for the video conferencing products from different vendors to work together. There are several organizations, including ITU (International Telecommunication Union), IMTC (International Multimedia Teleconferencing Consortium), and PCWG (Personal Conferencing Working Group) that are working toward prompting and producing standards for video conferencing. Many standards have been proposed such as the ITU-T G series standards for audio coding, H.261/H.263 for video coding, H.221/H.223 for multiplexing, and so on. Core standards of video conferencing are ITU-T T.120, H.320, H.323, and H.324 series of standards. T.120 addresses the real-time data conferencing, H.320 is for ISDN video conferencing, H.323 standard addresses video conferencing over the LAN without QoS guarantee, and H.324 is for low-bit-rate multimedia communication over the telephone network using V.34 modems. We are going to concentrate on these four major standards in the following.

T.120. (Data Protocols for Multimedia Conferencing) is a series of ITU-T recommendations for multipoint data communication service in multimedia conferencing environment. It was adopted in July 1996 and has been committed to by over 100 key international vendors including Microsoft, Apple, Intel, IBM, Cisco, MCI, AT&T, and so on. The T.120 defines a hierarchical structure (Fig. 6) with defined protocols and service definitions between the layers (88). T.122 and T.125 define a connection-oriented service that is independent of the T.123 transport stacks operating below it. The lower-level layers (T.122, T.123, T.124, and T.125) specify an application-independent mechanism for providing multipoint data communication services. The upper-level layers (T.126 and T.127) define protocols for specific conferencing applications, such as shared whiteboarding and multipoint file transfer. T.120 covers the document (file and graphics) sharing portion of a multimedia teleconference and can be used within H.320, H.323, and H.324 or by itself. Other T.120 series recommendations are summarized as follows.

T.121. (generic application template), which was adopted in July 1996, provides guidance for application and applications protocol developers on the correct and effective use of the T.120 infrastructure. It supplies a generic model for an application that communicates using T.120 services and defines a Generic Application Template specifying the use of T.122 and T.124 services.

T.122. (multipoint communication service for audiographics and audiovisual conferencing service definition) was adopted in March 1993. It defines network connection independent services, including multipoint data delivery (to all or a subset of a group), uniformly sequenced data reception at all users, resource control by applications using a token mechanism, and multiapplication signaling and synchronization.

T.123. (network specific data protocol stacks for multimedia conferencing) was adopted in October 1996. The networks currently include ISDN, CSDN, PSDN, B-ISDN, and LAN. Communication profiles specified provide reliable point-to-point connections between a terminal and an MCU, between a pair of terminals, or between MCUs.

T.124. (generic conference control), which was adopted in August 1995, provides a high-level framework for conference management and control of multimedia terminals and MCUs. It includes Generic Conference Control (GCC) and other miscellaneous functions including conference security.

T.125. (multipoint communication service protocol specification) was adopted in April 1994 and specifies a protocol to implement the Multipoint Communication Service (MCS) defined by T.122.

T.126. (multipoint still image and annotation protocol), which was adopted in August 1995, supports multipoint exchanges of still images, annotations, pointers, and remote events. The protocol conforms to the conference conductship model defined in T.124 and uses services provided by T.122 (MCS) and T.124 (GCC). T.126 includes components for creating and referencing archived images with associated annotations.

T.127. (multipoint binary file transfer protocol) was adopted in August 1995. It defines a protocol to support the interchange of binary files within an interactive con-

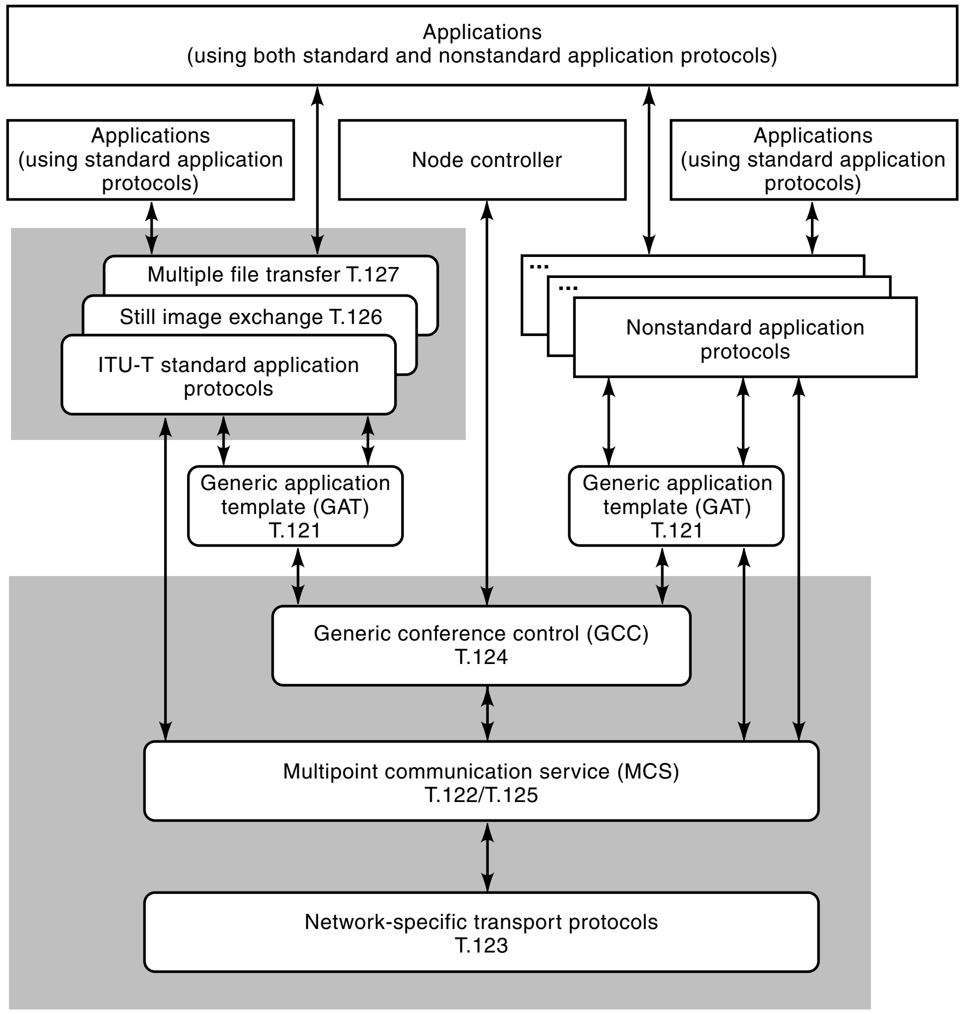

Figure 6. Architecture of ITU-T T.120 Series Recommendation.

ferencing or group working environment where T.120 recommendation series are used. T.127 supports simultaneous distribution of multiple files, selective distribution of files to a subset of participants, and retrieval of files from a remote site.

H.320. H.320 (Narrow-Band Visual Telephone System and Terminal Equipment) is the ITU recommendation adopted in March 1996. Narrow-band refers to the bit rate ranging from 64 kbit/s to 1920 kbit/s (64 kbit/s $\times$ 30). H.320 specifies video conferencing over circuit switched networks like ISDN and

Table 4. ITU H.320 Recommendations

Video codec:	H.261	Video codec for audiovisual service at $p \times 64$ kbps. Please refer to section entitled "H.261."
Audio codec:	G.711	PCM (pulse code modulation) of voice frequencies. 8 kHz, 8-bit encoding and requires 64 kbit/s bandwidth.
	G.722	7 kHz audio-coding within 64 kbit/s.
	G.728	Coding of speech at 16 kbit/s using low-delay code excited linea prediction.
Frame structure:	H.221	Frame structure for a 64 kbit/s to 1920 kbit/s channel in audiovisual teleservices. It supports a variety of data rates from 300 bit/s up to 2 Mbit/s. H.221 uses double error correction for secure transmission and can be used in multipoint configurations. It allows the synchronization of multiple 64 kbit/s or 384 bit/s connections and the control of the multiplexing of audio, video, data, and other signals within the synchronized multiconnection structure in the case of multimedia services such as video conferencing.
Control and indication:	H.230	Frame-synchronous control and indication signals for audiovisual systems.
Communication procedure:	H.242	System for establishing communication between audiovisual terminals using digital channel up to 2 Mbit/s. This recommendation describes all the point-to-point procedures involving the BAS codes in each frame which the control channel within the multiplexing structure specified in H.221.

Table 5. ITU-T H.323 Recommendations

Video codec:	H.261	Video codec for audiovisual service at $p \times 64$ kbit/s. Please refer to section entitled "H.261."
	H.263	Video coding for low bit rate communication. Please refer to section entitled "H.263."
Audio codec:	G.711	Pulse code modulation (PCM) of voice frequencies.
	G.722	7 kHz audio-coding within 64 kbit/s (48, 56, and 64 kbit/s).
	G.723	Dual-rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s modes.
	G.728	Coding of speech at 16 kbit/s using low-delay code excited linear prediction (3.1 KHZ).
	G.729	Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear-prediction (CS-ACELP)
Control:	H.245	Control protocol for multimedia communication. H.245 defines syntax and semantics of terminal information messages and procedures for in-band negotiation at the beginning and during communication. The messages include receiving and transmitting capabilities as well as mode preference from the receiving end, logical channel signaling, and Control and Indication. Acknowledged signaling procedures are specified for reliable audiovisual data communication.
Packet and Synchronization:	H.225	Media stream packetization and synchronization and nonguaranteed quality of service LANs. H.225 specifies messages for call control including signaling, registration, and admissions, as well as packetization/synchronization of media streams.

Switched 56. H.320 was designed primarily for ISDN, as ISDN BRI offers two 64 kbps (B-channel) data bandwidth for video conferencing. H.320 video conferencing system can also work over 3 ISDN BRI service (6 B-channels or 384 kbps) which are combined together using an inverse multiplexer (1-MUX). This yields better picture quality since more bandwidth is allocated for video data, but it costs a lot more. H.320 includes a series of recommendations which are summarized in Table 4.

H.323. H.323 (Visual Telephone Systems and Equipment for Local Area Networks Which Provide a Nonguaranteed Quality of Service) is a series of recommendations by ITU-T adopted in November 1996. H.323 extends H.320 to incorporate Intranet, LANs, and other packet-switched networks. It describes terminals, equipment, and services for multimedia communication over LANs which do not provide a guaranteed quality of service. H.323 terminals and equipment may carry video, audio, data, or any combination including videotelephone, and support for voice is mandatory. They may interwork with H.310/H.321 terminals on B-ISDN, H.320 terminals on N-ISDN, H.322 terminals on Guaranteed Quality of Service LANs, H.324 terminals in GSTN, and wireless networks. H.323 series recommendations are summarized in Table 5.

H.324. (Terminal for Low-Bit-Rate Multimedia Communication) is a series of recommendations by ITU-T adopted in March 1996. H.324 describes terminals for low-bit-rate multimedia communication over V.34 modems (total bandwidth of 36.6 kbps) on the Global Standard Telephone Network (GSTN). H.234 terminals may carry real-time voice, data, and video or any combination, including videotelephony. H.324 recommendation series are summarized in Table 6. H.324 allows more than one channel of each type to be in use and uses the logical signaling procedures. The content of each logical channel is described when it is opened, and procedures are provided for the expression of receiver and transmitter capabilities. This limits transmissions to what receivers can decode, and receivers may request a particular mode from transmitters. H.324 terminals may be used in multipoint video conferencing and interwork with H.230 terminals on the ISDN as well as terminals on wireless network. Compared with H.320 (ISDN) and H.323 (LAN), H.324 specifies multimedia teleconferencing over the most pervasive commu-

Table 6. ITU-T Recommendation H.324

Video codec:	H.263	Video coding at data rate less than 64 kbit/s. Please refer to section entitled "H.263."
Audio codec:	G.723	Audio codec for multimedia telecommunication at 5.3 or 6.4 kbit/s. It has a silence suppression mode so that the audio bandwidth can be used for other data when no audio is being transmitted.
Control:	H.245	Control protocol for multimedia communication.
Multiplexing:	H.223	Multiplexing protocol for low-bit-rate multimedia communications. H.223 specifies a packet-oriented multiplexing protocol which can be used for two low-bit rate multimedia terminals or between a low-bit-rate multimedia terminal and a MCU or an interworking adapter. The protocol allows the transfer of any combination of digital voice, audio, image, and data over a single communication link. The control procedures necessary to implement H.223 are defined in H.245.

nication network (GSTN) today. As a result, H.324-based video conferencing products are prominent in the market.

VIDEO-ON-DEMAND

Video-on-Demand (*VoD*) is an interactive digital video system that works like a cable television that allows subscribers to choose and view a movie from a large video archive at their own leisure. VoD is sometimes referred to as *Interactive TV* (*ITV*), and it is one of the most important client/server applications of digital video. VoD involves video servers which contain a large collection of digital video titles and deliver selected ones in stream mode to the subscribers over the network. The client then decompresses the stream and plays back at a good quality (at least comparable to standard VHS). A VoD system must support VCR-like functions including pause, rewind, fast-forward, play, and so on. Such commands are issued by the subscribers, processed by set-top boxes, and sent to video servers. Some of the key applications of VoD are video or film on demand, interactive games, distance learning, home shopping, and so on. VoD needs to be cost-effective in order to compete with the existing video services such as video rental and cable TV.

Depending on the interactive capabilities they provide, VoD can be classified into the following categories (89,90): *Broadcast* (*No-VoD*), *Pay-per-view* (*PPV*), *Quasi VoD* (*Q-VoD*), *Near VoD* (*N-VoD*) and *True VoD* (*T-VoD*). No-VoD service is similar to broadcast TV, in which the user is a passive viewer and has no control over the session. PPV service is similar to the existing PPV offered by cable TV companies in which the subscriber can sign up and pay for certain programs. No-VoD and PPV subscribers have no control over the program viewing and have to receive the program at a predetermined schedule by the service provider. Q-VoD service allows limited user control of viewing by grouping users based on a threshold of interest, and users can switch between different viewing groups. N-VoD service provides staggered movie start times. The additional sessions allow viewers to jump from session to session to gain access to a different portion of the feature presentation. T-VoD service dedicates an entire session to a single user and provides the individual user control over the presentation. The user can select the program at any time and has full-function virtual VCR capabilities such as fast-forward. T-VoD is the most difficult service to provide. A VoD system mainly consists of three components: *set-top boxes, video servers,* and *data delivery network* as shown in Fig. 7.

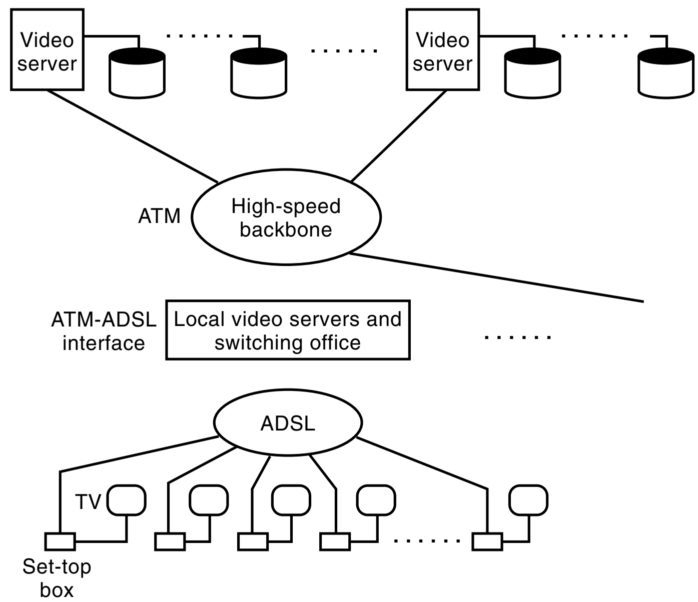

Figure 7. VoD system architecture.

Set-Top Boxes

Set-top boxes interface TV equipment with the VoD services. Set-top boxes must contain the video decoder to decode the compressed video stream delivered from the server and convert into a standard TV transmission format. It also needs to provide VCR-like functionalities by allowing upstream (from the subscriber to the service provider) user commands. A set-top box may consist of the following components: (1) a powerful CPU, a RAM buffer for reducing network jitters, and a graphic chip for screen overlays; (2) a 1 GHz tuner for cable delivery of VoD programs, or an ADSL modem for ADSL delivery; (3) an error correction chip; (4) a hardware MPEG-2 decoder for real-time video data decompression and audio hardware; (5) an RGB color converter and a radio-frequency (RF) demodulator or baseband demodulator for telephone line delivery of VoD programs; (6) an infrared receiver for remote control; and finally (7) a security chip to prevent theft.

Set-top boxes should be of low cost. It is suggested that set-top boxes should be sold at around $150 to make VoD widely acceptable. It also needs to be open and interoperable so that users can subscribe to several different VoD services.

Video Servers

Video servers store and provide user access to large collections of video titles. Its main functionalities include video storage, admission control, request handling, video retrieval, guaranteed stream transmission, video stream encryption, and support of virtual VCR functions. Designing a cost-effective, scalable, and efficient video server is a very challenging task. A video server should have the capacity to hold hundreds of terabytes of digital video and other information on different media such as magnetic tapes, optical write/read (W/R) disks, hard disks, or random access memory (RAM) buffer. It also must support simultaneous and real-time access to hundreds of different video titles by hundreds or even thousands of subscribers.

Real-Time Disk Scheduling. Disk scheduling and admission control algorithms are needed for guaranteed real-time video storage access. A common approach of real-time disk scheduling is to retrieve disk blocks for each stream in a round-robin fashion and keep the block size to the proportion of the stream's playback rate. Thus, this approach is known as *quality proportional multisubscriber servicing* (*QPMS*) (6), *rate conversion* (91), or *period transformation technique* (92). Other real-time disk scheduling algorithm include:

- The *elevator disk scheduling* algorithm (93), which scans the disk cylinders from the innermost to the outermost and then scans backwards. This algorithm is widely used because of its nearly minimal seek time and fairness.

Taking the priorities of the requests into consideration, the elevator disk scheduling algorithm can be easily extended to the real-time disk scheduling. Tasks can be grouped into different *priority classes,* and their priorities are determined based on factors such as tasks' deadlines. Each disk access request will be assigned a priority, and the highest priority class with pending disk accesses is serviced using the elevator algorithm.

- The *group sweeping scheme* (*GSS*) (94), which minimizes both the disk access time and the buffer space. This algorithm assigns each request to a group. Groups are served in a round-robin fashion and the elevator scheduling algorithm is used within each group. The algorithm behavior can be adjusted by changing the group size and the number of groups. It approximates to the elevator algorithm as the number of groups decreases, but approximates to the round-robin algorithm if the number of requests in each group increases.
- The *prefetching disk scheduling algorithm,* which can be extended for the real-time disk scheduling to reduce the memory requirement of the media server. Examples of such extensions are the *love page prefetching* and *delayed prefetching* algorithms which are used in the SPIFFI VoD system (95). Love page prefetching is a buffer pool page replacement algorithm that extends the *global LRU algorithm* (93) by distinguishing *prefetched pages* and *referenced pages.* Love page prefetching makes use of the fact that the video is usually accessed in a strictly sequential manner (for example, watching a movie), and the probability of a data block in the RAM buffer being referenced again is not high (96). It uses two LRU chains: one for referenced pages and one for prefetched pages. When a new page is needed, the referenced page chain is searched first and a page from prefetched chain is taken if there are no available pages in the referenced page chain. Delayed prefetching algorithm delays the data prefetching until the last minute, thus reducing the size of the RAM buffer needed to store the prefetched video data.

CPU Admission and Scheduling Algorithms. The purpose of CPU admission control and scheduling algorithms is to ensure that a feasible scheduling exists for all the admitted tasks. One example of CPU admission control and scheduling algorithm is as follows (97). Isochronous tasks (also known as *periodic tasks*) are periodic network transmissions of video and audio data. These tasks need performance guarantees—that is, throughput, bounded latency, and low jitter. Their priorities can be determined using a rate-monotonic basis (98); that is, a task with a higher frequency has a higher priority. A preemptive fixed-priority scheduling algorithm is used for isochronous tasks. Other real-time and non-real-time tasks can be scheduled using a weighted round robin, which can be preempted by isochronous tasks. General-purpose tasks have the lowest priorities, but they need to have minimum CPU quantum to avoid starvation.

Video Storage Strategies. The video storage subsystem consists of control units, disk/tape storage, and access mechanism. The video titles must be stored in compressed digital format. MPEG-2 is often used since it is the video codec for broadcast and HDTV video and is widely accepted by the cable and TV industry. Real-time video playback imposes strict delay and delay-variance requirements on the retrieval of video data from the storage subsystem.

Video titles can be stored on many different media such as RAMs, hard disks, optical R/W disks, and magnetic tapes. RAMs provide the fastest data access but are prohibitively expensive. On the other hand, magnetic tapes are very cost-effective, but too slow for the multisession and real-time requirement of VoD. Thus, a video server normally uses a hybrid and hierarchical storage structure (96) in which disk arrays are used to store the video retrieved from tertiary storage and deliver the video at users' requests. If we assume the capacity of one disk to be 1 Gbyte and assume the transfer bandwidth to be 4 Mbyte/s, a 1000-disk system is large enough to store 300 MPEG-2 movies of 90 min each and support 6500 concurrent users (99). In order to deliver smooth, continuous, and real-time video streams, a RAM buffer can be used to cache the popular portion of videos.

The arrangement of video titles across different storage media depends on the relative usage, the available bandwidth, and the level of interactivity supported. Such arrangements are often referred to as *video data placement policy* with the goal of balancing the storage device load and maximizing the utilization of both bandwidth and space. One example is the *Bandwidth-to-Space Ratio* (*BSR*) *policy* (100). BSR policy characterizes each storage device by its BSR, and each video stream by the ratio of its required bandwidth to the space needed to store it. The policy then dynamically determines how the video stream needs to be replicated and on which storage devices; this is done according to changes in users' demands. Another algorithm is the *Dynamic Segment Replication* (*DSR*) policy (101) which uses partial replication of the video streams to balance the load. DSR is based on the observation that a group of consecutive requests of a popular video stream can share the partial replication of the video stream generated by the previous request on the same video. Video placement can also be combined with video encoding to create multiresolution replications of the same video stream (102,103). Experiments show that such a schema can satisfy more user requests (with different QoS) than the one resolution approach.

Several basic techniques including *striping, declustering,* and *replication* can also be used to increase the video disk storage performance by interleaving a video title on multiple disks. *Striping* interleaves portions of disk blocks on multiple disks. The aim is to reduce the block access latency by parallel reading of the complete blocks. *Declustering* distributes blocks of files on several disks thus allowing parallel block access from the same file and increasing the data rate of the video stream. Video titles can also be replicated files among video servers based on the user demand and access pattern (e.g., time/day of peak access, average number of simultaneous viewers) to balance the load.

Disk Failure Tolerance. Real-time, continuous video streams of VoD require storage media with very high availability and reliability. Although a single disk may be very reliable, a large disk array used in a media server system may have an unacceptable high failure probability. For example, if the mean time to failure (MTTF) of a single disk is on the order of 300,000 h, the MTTF of a 1000-disk array system will be

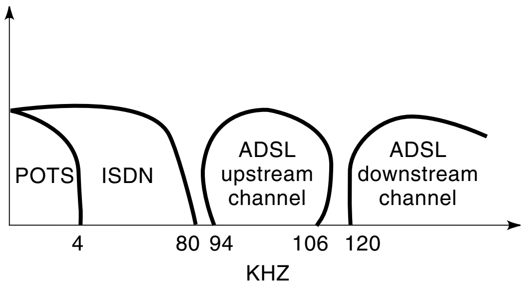

Figure 8. Frequency spectrum of ADSL.

just 300 h (99). Thus it is often necessary to sacrifice some of the disk space and bandwidth to improve the reliability and availability of the media server system. Usually, several parity (99,104,105) and mirroring (106) schemas can be used. For example, the streaming RAID schema (105) can effectively increase the MTTF of the disk array in the above example to 1100 years (99).

Data Delivery Network

Data delivery network connects subscribers and video servers, which includes backbone network, community network (or subscriber network), and switch office. It delivers video streams and carries control signals and commands. Due to the cost consideration, the subscriber network is usually based on twisted copper line or coax cable, whereas the backbone network is based on fiber or coax cable. Network technologies suitable for VoD are ADSL and ATM. Although ISDN is suitable for video conferencing, it does not meet the bandwidth requirement of VoD because its highest bandwidth is under 2 Mbit/s. To date, VoD trails have been conducted extensively on ADSL and ATM, with ATM forming the backbone from video serves to the switch office, and ADSL linking the switch office to individual homes. Switch office is responsible for distributing video signals to individual subscribers (e.g., through ADSL).

ADSL. Asymmetric Digital Subscriber Line (ADSL) refers to the two way capability of a twisted copper pair with analog to digital conversion at the subscriber end (e.g., through ADSL modem) and an advanced transmission technology. ADSL coexists with POTS (lower 4 kHz) and ISDN (lower 8 kHz) service over the same twisted copper line by using higher frequencies in the spectrum for data transmission (see Fig. 8). They can be separated from each other by the ADSL modem at the subscriber's side by using filtering such as *passive filtering*. This ensures the POTS service in case of ADSL modem failure. The ADSL upstream and downstream channels can be separated frequency division multiplexing (FDM) or can overlap each other. In the latter case, a technique called *local echo cancellation* is used to decode the resulting signal.

ADSL can provide asymmetric transmission of data up to 9 Mbit/s downstream to the customer and 800 kbps upstream depending on the line length and line and loop conditions. Table 7 lists some of the ADSL data rates (107). The actual ADSL downstream capacity also depends on the length of the copper loops (see Table 8) (108) and many other factors including wire gauge, bridged taps, and cross-coupled interfaces. Line attenuation increases with loop length and frequency, and it decreases as wire diameter increases (107). The asymmetric bandwidth characteristics of ADSL fit interactive video services such as VoD very well since they need much higher bandwidth for the downstream data transmission (e.g., broadcasting quality MPEG-2 video needs 6 Mbit/s bandwidth) than the upstream user signaling (e.g., rewind command). ADSL is usually used to provide dedicated asymmetrical megabit access for interactive video and high-speed data communication over a single telephone line such as Internet access. Another huge advantage of ADSL is that it can run over POTS (Plain Old Telephone Service) and thus can reach vast amount of customers. This is very important since the full deployment of boardband cable or fiber will take decades and enormous investment. In other words, ADSL helps make digital video services such as VoD marketable and profitable for the telephone company and other service suppliers.

Table 7. Different Data Rates of ADSL Channels

Downstream Bear Channels		Upstream (Duplex) Bear Channels	
$n \times 1.536$ Mbit/s	1.536 Mbit/s	c channels:	16 kbit/s
	3.072 Mbit/s		64 kbit/s
	4.608 Mbit/s	Optional channels:	160 kbit/s
	6.114 Mbit/s		384 kbit/s
$n \times 2.048$ Mbit/s	2.048 Mbit/s		544 kbit/s
	4.096 Mbit/s		576 kbit/s

There are two modulation methods for ADSL, namely, *DMT* (*Discrete Multitone*) and *CAP* (*Carrierless Amplitude/Phase modulation*). DMT is usually preferred because of its higher throughput and greater resistance to adverse line conditions. It can effectively compensate for widely varying line noise conditions and quality levels. The basic idea of DMT is to divide the available bandwidth into large numbers of subchannels or carriers using the discrete fast Fourier transform (FFT). The data are then distributed over these subchannels so that the throughput of every single subchannel is maximized. If some of the subchannels cannot carry any data, they can be turned off to optimize the use of the available bandwidth. DMT is used in the ANSI ADSL standard T1.413. ADSL transmits data in *superframes* which consist of 68 ADSL frames and one additional frame for synchronization. Each ADSL frame contains two parts: the *fast data* and *interleaved data*. The fast data may contain CRC error checking bits and forward error correction bits. The interleaved data contains only the user data. Notice that the error correction can be used to reduce the impulse noise on the video signal, but it also introduces delay. Whether to employ error correc-

Table 8. Relationship Between the Loop Length and the ADSL Bandwidth

Length	Downstream
Up to 18,000 ft	1.544 (T1) Mbit/s
16,000 ft	2.048 (E1) Mbit/s
12,000 ft	6.312 (DS2) Mbit/s
9,000 ft (average line length for US customers)	8.448 Mbit/s

Table 9. Some VoD User Trials

Location	Company	Technology	Service
Fairfax, VA (Stargazer)	Bell Atlantic	nCube/Oracle video server; ADSL (1.5 Mbps/64 Kbps); MPEG-1 and MPEG-2	VoD, home shopping, etc.
Orlando, FL	Time Warner etc.	SGI Challenge video server; ATM over fiber/coax at 45 Mbps with customer side at 3.5 Mbps for MPEG video	VoD, home shopping, games, etc.
Helsinki, Finland	Helsinke Telephone Co.	ADSL (2.048 Mbps/16 Kbps), ATM as backbone	VoD
Singapore	Singapore Telecom	ATM/ADSL (5.5 Mbps/168 kbps) over fiber/copper	VoD
Yokosuka, Japan	NTT, Microsoft	ATM/ADSL over fiber/copper; MPEG-2	VoD
Suffolk, England	British Telecom	nCube/Oracle Media Server; ATM/ ADSL (2 Mbps) over fiber/copper; MPEG-1, MPEG-2	VoD, home shopping, games, etc.
Germany	Deutsche Telekom	ATM/ADSL over fiber/coax, satellite	PPV, NVoD, etc.

tion or not depends on the network and type of data ADSL transmits.

ADSL is often viewed as the transition technology used before existing copper lines can be converted to the fiber or coax cables. A higher-speed variant of it, called *VDSL,* is under development. VDSL would provide 12.96 Mbit/s to 51.84 Mbit/s downstream and 1.6 Mbit/s to 2.3 Mbit/s upstream data rates with the compromise of the line length (4500 ft to 1000 ft) (108).

ATM. Asynchronous transfer mode (ATM) uses a fixed 53-byte cell (packet) for dynamic allocation of bandwidth. The cells have characteristics of both circuit-switch and packet-switch networks. A virtual path is set up through the involved switches when two end points wish to communicate. This provides a bit-rate-independent protocol that can be implemented on many network media such as twisted pair, coax, and fiber. ATM operates at very high speed; for example, SONET (Synchronous Optical Net) operates at 155 Mbit/s and ATM could potentially operate up to 2.2 Gbps over a cell-switched network. However, ATM requires broadband fiber and coax cables to fully achieve its capacity. ATM is ideal for VoD applications because of its high bandwidth and cell switching capability, which is a compromise between delay-sensitive and conventional data transmissions. ATM AAL1 protocol was designed for constant bit-rate services such as the transmission of MPEG video. The ATM Forum also proposed a standard for constant bitrate AAL5 which can be used for both VoD and fast Internet access.

The ATM backbone network can interlink with the ADSL network through an interface. Such an interface demultiplexes ATM signals and regenerates many 1.5 or 2 Mbit/s signals to feed to the ADSL lines. The drawback of ATM is its availability and the high cost of related equipment. Thus, ATM is often used as the backbone network of the VoD systems. In the future, ATM is expected to replace ADSL in a VoD system once copper twisted lines are upgraded to fiber lines or broadband coax cables.

VoD Trials

VoD is an emerging technology that still needs further product development and refinement before it can be accepted by the average consumers. Many VoD user trials have been done or are being conducted around the world since the early 1990s, which cost billions of dollars in investments. Some examples of VoD trials are listed in Table 9.

Despite the failure of early user trials in the early 1990s due to unacceptable high cost, people have gathered valuable information for the statistical analysis of the related technologies and on the overall economic value of VoD, which include:

- Feasibilities of various VoD system architectures and networking technologies such as ADSL.
- Customer expectations about VoD services. For example, the usage by category data gathered during the Bell Atlantic VoD market trial (Stargazer) in 1995 supports the view that customers desire diversified product offering.
- Customer acceptance and satisfaction. For example, according to the results of the Stargazer VoD trial, the buy rate of VoD subscribers is significantly higher than that of cable PPV and video rental.

With the experience gained in the early phases of various VoD trials and recent advances in the technology and standardization, VoD is becoming more and more affordable especially when set-top boxes are getting cheaper. It is expected that VoD will become a reality in the near future; in other words, VoD will become not only commercially viable, but also ready for the market.

BIBLIOGRAPHY

1. D. E. Gibson, Report on an International Survey of 500 Audio, Motion Picture Films and Video Archives, talk given in the annual FIAT/IASA Conf., Bogensee, Germany, September 1994.
2. A. K. Elmagarmid et al., *Video Database System: Issues, Products and Applications,* Norwell, MA: Kluwer, 1997.
3. A. Hampapur, Design video data management systems, PhD thesis, Univ. Michigan, 1995.
4. H. Jiang and J. W. Dailey, Video database system for studying animal behavior, *Proc. Multimedia Storage and Archiving Syst.,* Vol. 2916, 1996, pp. 162–173.

5. A. Watt, *Fundamentals of Three-dimensional Computer Graphics,* Reading, MA: Addison-Wesley, 1989.
6. H. M. Vin and P. V. Rangan, Designing a multiuser HDTV storage service, *IEEE J. Selected Areas Commun.,* **11** (11): 153–164, 1993.
7. D. L. Gall, MPEG: A video compression standard for multimedia applications, *Commun. ACM,* **34** (4): 46–58, 1991.
8. N. Johnson, Indeo video interactive: Back with a vengeance, *Dig. Video Mag.,* **February**: 46–54, 1996.
9. J. Ozer, Indeo sets the standard for video quality, but some features may be ahead of their time, *PC Magazine,* **January**: 1996.
10. QuickTime technology brief—QuickTime 3.0, Apple Computer, Inc., 1997.
11. C. Wiltgen, The QuickTime FAQ [Online] Available http://www.QuickTimeFAQ.com.
12. C. J. Date, *An Introduction to Database Systems,* Reading, MA: Addison-Wesley, 1975.
13. G. Davenport, T. G. A. Smith, and N. Pincever, Cinematic primitives for multimedia, *IEEE Comput. Graphics Appl.,* **11** (4): 67–74, 1991.
14. D. Swanberg, C.-F. Shu, and R. Jain, Knowledge guided parsing in video database, *Proc. IS&T/SPIE Symp. on Electron. Image Sci. & Technol.,* 1993, pp. 13–24.
15. R. Hjelsvold and R. Midtstraum, Modeling and querying video data. *Proc. 20th Int. Conf. on Very Large Data Bases,* September 1994.
16. J. F. Allen, Maintaining knowledge about temporal intervals, *Commun. ACM,* **26** (11): 832–843, 1983.
17. T. D. C. Little and A. Ghafoor, Interval-based conceptual model for time-dependent multimedia data, *IEEE Trans. Knowl. Data Eng.,* **5**: 551–563, 1993.
18. H. Jiang, D. Montesi, and A. K. Elmagarmid, VideoText database systems, *Proc. 4th IEEE Int. Conf. on Multimedia Computing and Syst.,* 1997, pp. 344–351.
19. R. Hjelsvold, Video information content and architecture. *Proc. 4th Int. Conf. on Extending Database Technol.,* Cambridge, UK, March 1994.
20. T. G. A. Smith, If you could see what I mean . . . descriptions of video in an anthropologist's notebook, Master's thesis, MIT, 1992.
21. A. Hampapur, R. Jain, and T. Weymouth, Digital video indexing in multimedia systems, *Proc. Workshop on Indexing and Reuse in Multimedia Syst.,* 1994.
22. H. J. Zhang et al., Automatic parsing of news video, *Proc. 1st IEEE Int. Conf. on Multimedia Computing and Syst.,* 1994.
23. H. Jiang et al., Scene change detection techniques for video database systems, *ACM Multimedia Syst.,* **6** (3): 186–195, 1998.
24. D. Swanberg, C.-F. Shu, and R. Jain, Architecture of multimedia information system for content-based retrieval, *Proc. Audio Video Workshop,* 1992.
25. T. D. C. Little et al., A digital ondemand video service supporting content-based queries, *Proc. 1st ACM Int. Conf. on Multimedia,* 1993, pp. 427–436.
26. R. Lienhart, Automatic text recognition for video indexing, *Proc. 4th ACM Int. Multimedia Conf.,* 1996, pp. 11–20.
27. B.-L. Yeo, Efficient Processing of Compressed Images and Video, PhD thesis, Princeton Univ., January 1996.
28. T. Kanade et al., Informedia digital video library system: Annual progress report. Technical report, Carnegie Mellon Univ., Comput. Sci. Dept., Pittsburgh, PA, February 1997.
29. M. A. Smith and A. Hauptmann, Text, speech, and vision for video segmentation: The informedia project, *AAAI Fall 1995 Symp. on Computational Models for Integrating Language and Vision,* 1995.
30. T. G. A. Smith and G. Davenport, The stratification system: A design environment for random access video, *Workshop on Networking and Operating Syst. Support for Digital Audio and Video,* 1992.
31. R. Weiss, A. Duda, and D. Gifford, Content-based access to algebraic video, *Proc. 1st IEEE Int. Conf. on Multimedia Computing and Syst.,* 1994.
32. F. Kokkoras et al., Smart VideoText: An intelligent video database system, TR 97-049, Dept. Comput. Sci., Purdue Univ., IN, 1997.
33. J. F. Sowa, *Conceptual Structures: Information Processing in Minds and Machines,* Reading, MA: Addison-Wesley, 1984.
34. J. Banerjee and W. Kim, Semantics and implementation of schema evolution in object-oriented database, *Proc. ACM SIGMOD '87,* 1987, pp. 311–322.
35. E. Oomoto and K. Tanaka, OVID: Design and implementation of a video-object databse system, *IEEE Trans. Knowl. Data Eng.,* **5**: 629–643, 1993.
36. B. Shahraray, Scene change detection and content-based sampling of video sequences, *Proc. Digital Video Compression: Algorithms and Technol.,* Vol. 2419, 1995, pp. 2–13.
37. B.-L. Yeo and B. Liu, On the extraction of DC sequence from MPEG compressed video, *Int. Conf. on Image Processing,* 1995.
38. A. Nagasaka and Y. Tanaka, Automatic video indexing and full-video search for object appearances, *Proc. 2nd Working Conf. on Visual Database Syst.,* 1991, pp. 119–133.
39. A. Akutsu et al., Video indexing using motion vectors, *Proc. Visual Commun. and Image Processing,* 1992.
40. P. R. Hsu and H. Harashima, Detecting scene changes and activities in video databases, *Proc. ICASSP '94,* Vol. 5, 1994, pp. 33–36.
41. H. J. Zhang, A. Kankanhalli, and S. W. Smoliar, Automatic parsing of full-motion video, *Multimedia Syst.,* **1**: 10–28, July 1993.
42. H. G. Longbotham and A. C. Bovic, Theory of order statistic filters and their relationship to linear FIR filters, *IEEE Trans. Acoust. Speech Signal Process.,* **ASSP-37** (2): 275–287, 1989.
43. R. Zabih, J. Miller, and K. Mai, Feature-based algorithms for detecting and classifying scene breaks, *Proc. 4th ACM Conf. on Multimedia,* 1995.
44. B.-L. Yeo and B. Liu, Rapid scene analysis and compressed video, *IEEE Trans. Circuits Syst. Video Technol.,* **5**: 533–544, 1995.
45. B.-L. Yeo and B. Liu, A unified approach to temporal segmentation of motion JPEG and MPEG compressed video. *Proc. 2nd IEEE Int. Conf. on Multimedia Computing and Syst.,* 1995.
46. F. Arman, A. Hsu, and M. Chiu, Image processing on compressed data for large video database, *Proc. ACM Multimedia '93,* 1993, pp. 267–272.
47. I. K. Sethi and N. Patel, A statistical approach to scene change detection, *Proc. Storage and Retrieval for Image and Video Database III,* Vol. 2420, 1995, pp. 329–338.
48. H. J. Zhang et al., Video parsing using compressed data, *Proc. Image and Video Processing II,* Vol. 2182, 1994, pp. 142–149.
49. J. Meng, Y. Juan, and S. F. Chang, Scene change detection in a mpeg compressed video sequence, *Proc. Storage and Retrieval for Image and Video Database III,* Vol. 2420, 1995.
50. P. Aigrain and P. Joly, Automatic real-time analysis of film editing and transition effects and its applications, *Comput. Graphics,* **18** (1): 93–103, 1994.
51. A. Hampapur, R. Jain, and T. Weymouth, Digital video segmentation, *Proc. ACM Multimedia '94,* 1994.

52. M. A. Smith and M. G. Christel, Automating the creation of a digital video library, *Proc. ACM Multimedia '95,* 1995, pp. 357–358.

53. V. M. Bove, What's wrong with today's video coding?, *TV Technol.,* **February**: 1995.

54. M. R. W. Dawson, The how and why of what went where in apparent motion, *Psychol. Rev.,* **98**: 569–603, 1991.

55. M. Livingstone and D. O. Hubel, Segregation of form, color, movement and depth: Anatomy, physiology and perception, *Science,* **240**: 740–749, 1988.

56. C. Cedras and M. Shah, Motion-based recognition: A survey, *Image Vision Comput.,* **13** (2): 129–155, 1995.

57. F. Arman and J. K. Aggarwal, Model-based object recognition in dense-range images—a review, *ACM Comput. Surv.,* **25** (1): 5–43, 1993.

58. M. S. Telagi and A. H. Soni, 3-D object recognition techniques: A survey, *Proc. 1994 ASME Design Tech. Conf.,* Vol. 73, 1994.

59. S. S. Intille, Tracking using a local closed-worlds assumption: Tracking in the football domain, Tech. Rep. 296, MIT Media Laboratory, Perceptual Computing Section, August 1994.

60. M. Davis, Media streams: An iconic visual language for video annotation, *Proc. Int. Symp. on Visual Languages,* 1993, pp. 196–202.

61. R. Hjelsvold, VideoSTAR—A database for video information sharing, PhD thesis, Norwegian Inst. Technol., November 1995.

62. M. Davis, Knowledge representation for video, *Proc. 12th Nat. Conf. on Artificial Intell.,* Vol. 1, Cambridge, MA: AAAI Press, 1994, pp. 120–127.

63. A. D. Bimbo, E. Vicario, and D. Zingoni, Sequence retrieval by contents through spatiotemporal indexing, *Proc. Int. Symp. on Visual Languages,* 1993.

64. F. Arman et al., Content-based browsing of video, *Proc. ACM Multimedia '94* , 1994.

65. M. Ioka and M. Kurokawa, Estimation of notion vectors and their application to scene retrieval, Tech. Rep. 1623-14, IBM Res., Tokyo Res. Laboratory, Shimotsuruma, Yamato-shi, Kanagawa-ken 242, Japan, 1993.

66. Y. Tonomura and A. Akutsu, A structured video handling technique for multimedia systems, *IEICE Trans. Inf. Syst.,* **E78-D**: 764–777, 1994.

67. A. Akutsu and Y. Tonomura, Video tomography: An efficient method for camerawork motion vectors, *Proc. ACM Multimedia '94,* 1994.

68. S. W. Smoliar and H. J. Zhang, Content-based video indexing and retrieval, *IEEE Multimedia,* **1** (2): 62–72, 1994.

69. S. W. Smoliar, H. J. Zhang, and J. H. Wu, Using frame technology to manage video, *Proc. Workshop on Indexing and Reuse in Multimedia Systems,* 1994.

70. M. Flickner et al., Query by image and video content: The QBIC system, *Computer,* **28** (9): 23–31, 1995.

71. E. Ardizzone and M. Cascia, Automatic video database indexing and retrieval, *Multimedia Tools Appl.,* **4** (1): 29–56, 1997.

72. M. H. Bohlen et al., Evaluating and enhancing the completeness of TSQL2, Tech. Rep. TR95-05, Comput. Sci. Dept., Univ. Arizona, 1995.

73. R. T. Snodgrass, The temporal query language Tquel, *ACM Trans. Database Systems,* **12**: 299–321, 1987.

74. A. D. Bimbo and E. Vicario, A logical framework for spatio temporal indexing of image sequence, in S. K. Chang (ed.), *Spatial Reasoning,* Berlin: Springer-Verlag, 1993.

75. R. Hjelsvold, R. Midtstraum, and O. Sandst, *Searching and Browsing a Shared Video Database,* chapter Design and Implementation of Multimedia Database Management Systems, Norwell, MA: Kluwer, 1996.

76. G. Ahanger, D. Benson, and T. D. C. Little, Video query formulation, *Proc. IS&T/SPIE Symp. on Electronic Image Sci. & Technol.: Storage and Retrieval for Image and Video Database III,* Vol. 2420, 1995, pp. 280–291.

77. Y. F. Day et al., Spatio-temporal modeling of video data for online object-oriented query processing, *Proc. IEEE ICDE,* 1995.

78. B. Doyle and J. Sauer, Digital in and digital out—digital editing with Firewire, *New Media Magazine,* **7** (11): 46–55, 1997.

79. 1994-1995 IEEE Standard for a High Performance Serial Bus, Inst. of Electr. and Electron. Engineers.

80. S. Mullen, The DV format [Online], 1996. Available www: http://members.aol.com/dvcnysteve/DVpage.html

81. B. Schmitt and A. Gerulaitis, How to edit your DV footage? [Online], 1997. Available www: http://www.computervice.com/dvl/howedit.html

82. L. A. Rettinger, Desktop video conferencing: Technology and use for remote seminar delivery, Master's thesis, North Carolina State Univ., July 1995.

83. D. E. Comer, *Internetworking with TCP/IP,* Vol. I, Englewood Cliffs, NJ: Prentice-Hall, 1995.

84. D. Miloli and R. Keinath, *Distributed Multimedia Through Boardband Communications,* Norwood, MA: Artech House, 1994.

85. V. Kumar, *MBone: Interactive Multimedia on The Internet,* New York: Macmillan, 1995.

86. K. Savetz, N. Randall, and Y. Lepage, *MBone: Multicasting Tomorrow's Internet,* Foster City, CA: IDG Books, 1986.

87. S. Deering, Host extensions for IP multicasting, Internet Request for Comment (RFC) 1112, 1989.

88. A primer on the T.120 series standard, DataBeam Corp. [Online], 1997. Available www:http://www.databeam.com/ccts/t120primer.html

89. A. D. Gelman et al., A store-and-forward architecture for Video-On-Demand service, *Proc. IEEE ICC,* 1991, pp. 27.3.1–27.3.5.

90. T. D. C. Little and D. Venkatesh, Prospects for interactive Video-on-Demand, *IEEE Multimedia,* **1** (3): 14–24, 1994.

91. P. Lougher and D. Shepherd, The design of a storage server for continuous media, *Computer,* **36**: 32–42, 1993.

92. S. J. Daigle, Disk scheduling for continuous media data streams, Master's thesis, Carnegie Mellon Univ., 1992.

93. A. Silberschatz and P. B. Galvin, *Operating System Concepts,* 4th ed., Reading, MA: Addison-Wesley, 1994.

94. P. S. Yu et al., Design and analysis of a grouped sweeping schema for multimedia storage management, *Proc. 3rd Int. Workshop on Network and Operating Syst. Support for Digital Audio and Video,* 1992, pp. 44–45.

95. C. S. Freedman and D. J. DeWitt, The SPIFFI scalable video-on-demand system, *Proc. ACM SIGMOD'95,* 1995, pp. 352–363.

96. B. Özden et al., A low-cost storage server for movie on demand databases, *Proc. 20th VLDB Conf.,* 1994, pp. 594–605.

97. K. K. Ramakrishnan et al., Operating system support for a Video-on-Demand file service, *Multimedia Syst.,* **3**: 53–65, 1995.

98. C. L. Liu and J. W. Layland, Scheduling algorithms for multiprogramming in hard-real-time environment, *J. ACM,* **11**: 46–61, 1973.

99. S. Berson, L. Golubchik, and R. R. Muntz, Fault tolerant design of multimedia servers, *Proc. ACM SIGMOD '95,* 1995, pp. 364–375.

100. A. Dan and D. Sitaram, An online video placement policy based on bandwidth to space ration (BSR), *Proc. ACM SIGMOD '95,* 1995, pp. 376–385.

101. A. Dan, M. Kienzle, and D. Sitaram, Dynamic segment replication policy for load-balancing in video-on-demand servers, IBM Res. Rep. RC 19589, 1994.

102. T. Chieuh and R. Katz, Multiresolution video representation for parallel disk arrays, *Proc. ACM Multimedia '93,* 1993, pp. 401–409.

103. K. Keeton and R. H. Katz, Evaluating video layout strategies for a high-performance storage server, *Multimedia Syst.,* **3**: 43–52, 1995.

104. D. A. Patterson, G. Gibson, and R. H. Katz, A case for redundant arrays of inexpensive disks (RAID), *Proc. ACM SIGMOD '88,* 1988, pp. 109–116.

105. F. Tobagi et al., Streaming RAID—a disk array management system for video files, *ACM Multimedia '93,* 1993, pp. 393–400.

106. D. Bitton and J. Gray, Disk shadowing, *Proc. VLDB '88,* 1998, pp. 331–338.

107. ADSL tutorial: Twisted pair access to the information highway, ADSL Forum [Online]. Available www:http://www.adsl.com/adsl-tutorial.html

108. General introduction to copper access technology, ADSL Forum [Online]. Available www:http://www.adsl.com/general-tutorial.html

AHMED K. ELMAGARMID
HAITAO JIANG
Purdue University