Video Aggregation: Adapting Video Traffic for Transport Over Broadband Networks by Integrating Data Compression and Statistical Multiplexing

Soung C. Liew, *Senior Member, IEEE*, and Chi-yin Tse, *Member, IEEE*

***Abstract*—This paper investigates video aggregation, a concept that integrates compression and statistical multiplexing of video information for transport over a communication network. We focus on the transmission of a group of video sessions as a bundle, the practical examples of which include entertainment-video broadcast and video-on-demand (VoD). In this situation, the advantage of constant bit-rate (CBR) transport (which facilitates simple network management and operation) and the advantage of variable bit-rate (VBR) video compression (which yields smoother image quality) can be achieved simultaneously. We show that it is better to integrate compression and statistical multiplexing before the bundle of video traffic enters the network than performing them as independent processes. We present experimental results which indicate the advantages of video aggregation in terms of superior image quality and efficient bandwidth usage.**

I. Introduction

FUTURE broadband integrated services networks based on the asynchronous transfer mode (ATM) technology are expected to carry information from a large variety of different services and applications. However, video traffic is likely to dominate because of the bandwidth-hungry nature of images. It is therefore important to understand how video traffic might best be multiplexed, transported, and switched.

In ATM networks, data are packetized into fixed-length cells of 53 bytes. Cells are routed in the network based on the routing information contained in their five-byte headers [1]. These cells may be discarded inside the network when traffic congestion occurs.

To reduce the bandwidth needed, video is almost always compressed before transmission. The Moving Picture Experts Group (MPEG) coding scheme [2] has been developed as a standard of video compression. Since data have been highly compressed, cell loss during transmission of MPEG-coded video may cause serious degradation of image quality. Low-cell-loss-rate network operation, or schemes that facilitate such operation, is therefore essential.

This paper focuses on the scenario where information from a group of video sessions are to be delivered as a bundle. We argue that compression and multiplexing of video streams in such a scenario should occur together before packetization (i.e., before the ATM layer [1]). To distinguish this from traditional statistical multiplexing of cells, we call this *video aggregation*. This paper discusses the potential advantages of aggregation from the standpoints of image quality, bandwidth usage, network management, and operation.

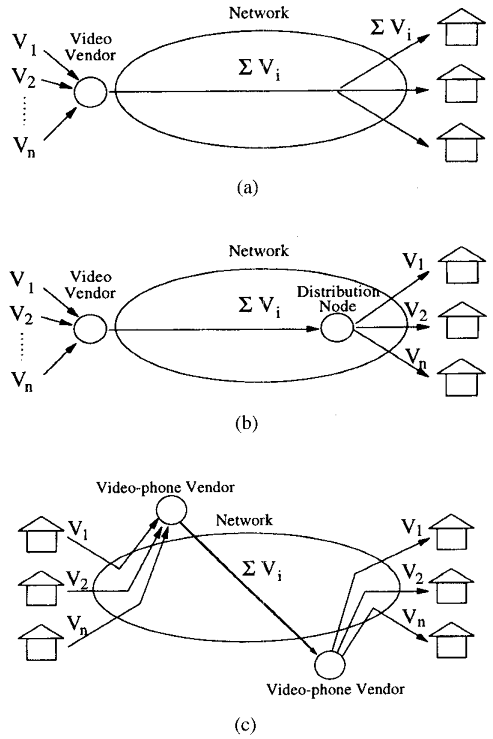


Fig. 1. Applications of video aggregation: (a) video broadcasting, (b) VoD, and (c) long-distance video-phone.

Application areas of aggregation include video broadcast and video-on-demand (VoD). Video programs are transported as a bundle from the video server directly to the subscribers in the former [Fig. 1(a)], and to a distribution node close to the subscribers in the latter [Fig. 1(b)] [3]. Aggregation may also find use in the transport of long-distance video-phone data: video streams from various subscribers targeted for a common remote area may be aggregated at a local central office before being delivered as a bundle to the remote central office serving the area [Fig. 1(c)]. These three application scenarios will be further discussed in Section III after the basic concept of aggregation has been explained in the next section.

Manuscript received May 15, 1995; revised October 3, 1995. This work was supported by the RGC Earmarked Grant of the Hong Kong University and the Polytechnic Grant Council under Grant CUHK 477/95E. This paper was presented in part at the IEEE INFOCOM'95, Boston, MA.

The authors are with the Department of Information Engineering, The Chinese University of Hong Kong (email: soung@ie.cuhk.edu.hk).

Publisher Item Identifier S 0733-8716(96)04888-3.

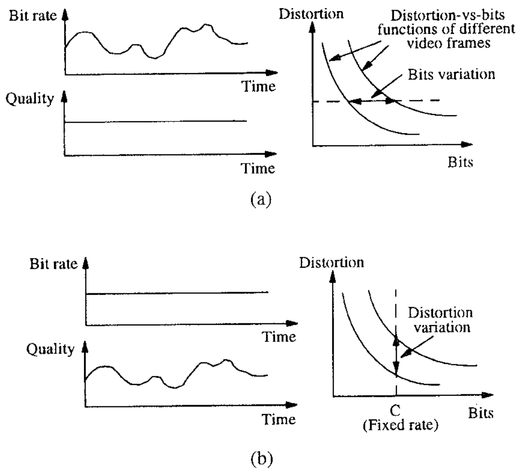


Fig. 2. Characteristics of VBR and CBR video compression schemes: (a) VBR video compression and (b) CBR video compression.

II. Motivations and Basic Concepts of Video Aggregation

Let us first consider a video stream before moving on to a video bundle. Compression methods of a video stream can be divided into two classes: variable bit-rate (VBR) compression and constant bit-rate (CBR) compression. The intrinsic bandwidth requirement of a video stream for a given constant image quality may vary over time, due to the variation of the scene complexity. In VBR compression, the output bit rate of the encoder varies according to the bandwidth requirement of the underlying video sequence. The image quality is more or less constant [Fig. 2(a)] [4]. In CBR compression, the output bit rate of the encoder is forced to be constant. The image quality varies over time since scenes that intrinsically demand high bandwidths may have their bandwidths cut down to maintain the constant output bit rate [Fig. 2(b)]. Compression schemes that lie somewhere between the two extremes are also possible. In general, in the consideration of compression, there is a trade-off between the variations of bit rate and image quality.

CBR and VBR transport, as distinct from compression, refers to using CBR and VBR channels, respectively, for the transport of data. CBR transport has many advantages from the network viewpoint. Since the data rate is constant, bandwidth allocation and tariff for network usage are simple. It is also straightforward for the network to multiplex several CBR channels onto a common communication channel and guarantee the lossless delivery of cells since cells arrive at predictable rates.

It is natural to use CBR transport for CBR-compressed data. Similarly, VBR compression followed by VBR transport [5]–[7] is a natural combination. In the latter, however, it is difficult to statistically multiplex VBR video streams and guarantee the lossless delivery of cells, since the bit rates of the multiplexed streams may peak together. In general, a bandwidth higher than the average bit rate needs to be allocated to a VBR stream to maintain a small cell-loss probability. Lossless cell delivery is not possible unless the peak bandwidth is allocated, in which case the delivery of the VBR stream will be expensive. For public networks, the fact that cells may be dropped due to interference from other streams also complicate the tariff problem and the contractual agreement between the network operator and user.

The other combination that makes sense is CBR transport for VBR-compressed data. The VBR data from the output of the VBR encoder is fed to a smoothing buffer, which forwards data at a constant rate to the network. However, if the buffer size is not large enough, then data may be dropped due to buffer overflow. Furthermore, data may also incur delay jitters in the buffer in addition to those in the network.

One general issue is how to achieve both the advantages of VBR compression (which offers relatively constant image quality) and those of CBR transport (which facilitates simple network operation) simultaneously. It turns out that this is possible when several video streams are to be transported as a bundle. A common CBR channel can be used to transport the VBR-compressed streams as a whole. In other words, as a group, the video bundle is CBR, but individually, the video streams are VBR. The contract between the network and the user is simple: the network is required to guarantee the delivery of all cells so long as the total data rate of the streams does not exceed the reserved CBR-channel bandwidth. It is the user's responsibility to adapt the VBR streams into a CBR stream before pumping the data into the network.

A straightforward adaptation method for the user is to first packetize the output data of the VBR encoders into cells and then multiplex the cells statistically [5]–[7]. The problem with this approach is that we have simply shifted the statistical multiplexing from the network to the user. Instead of the network, the user now faces the problem that cells may be dropped when the bit rates of the VBR streams peak together. When cell loss occurs, and especially when some important data (e.g., header information and grey-level signals of the images) are contained in the discarded cells, serious image-quality degradation may result.

One alternative is to drop data selectively according to their importance when the assigned bandwidth is exceeded. One example is the two-layer video coding and transport strategy that has been widely investigated [4], [8]–[11]. In the two-layer approach, VBR-coded data of a video stream are divided into the base layer that contains the basic-quality-image data and the second layer that contains the image-enhancement data. The base layer is transported as the guaranteed stream (GS), and the second layer is transported as the enhancement stream (ES). That is, the base-layer and second-layer data are contained in separate cells. The idea is that only the cells of the enhancement stream can be dropped. The two-layer scheme has mainly been investigated in the context of multiplexing within the network [4], [8]–[11]. One could, in principle, apply the two-layer approach to the video-bundle scenario in which the user is performing the multiplexing. The GS cells of all video sources are transmitted and they use up a certain amount of the bandwidth of the reserved CBR channel. The remaining bandwidth is then used for the statistical multiplexing of the ES cells.

With the two-layer approach, more processing is needed at the receiver to combine the data from the two layers before decoding. Perhaps the more severe shortcoming of the two-layer approach is its implication for image quality when cells are dropped. Generally, not all ES cells are equally important from the visual-quality standpoint: 1) the ES cells of the same video stream may be of different levels of importance and 2) among the video streams, the ES cells of some streams may be more important than those of other streams because of the higher complexity of the associated scenes. The different levels of importance among the ES cells are not distinguishable during the multiplexing process of the two-layer approach.

Instead of the cell-level multiplexing described above, the user can multiplex the data before they have been packetized into cells. The advantage of this is that the relative importance of the data is known to the last detail, and one could choose to drop the least significant data when the reserved bandwidth is exceeded. We can potentially achieve 1) better and smoother image quality for the frames within a video stream and 2) fairness of image quality among the video streams. As such, discarding data can be viewed as a form of compression which is necessitated only when the common CBR bandwidth has been exceeded. This is the basic observation that motivates video aggregation: integration of video compression and multiplexing into a single process prior to data packetization.

In video aggregation, video sequences are compressed collaboratively such that 1) the sum of the bit rates of the video sequences is almost equal to (but not larger than) the reserved bit rate of the CBR channel, 2) within each video stream, data discarded are less important than those retained, and 3) different video streams have roughly the same image quality according to some signal-to-noise or distortion metric. Video aggregation is described in an abstract manner below as a lossy secondary compression process that is applied after a preliminary compression process.

A. Basic Concept of Video Aggregation

In many video compression schemes, the output data can be divided into segments. Each segment has a certain number of bits, some of which can be dropped, if needed, at the expense of image quality. Associated with each segment is a function relating the number of bits retained and the corresponding image quality. Within each segment, bits can be ordered according to their significance so that those of lower significance will be dropped first when necessary.

As an illustration, in MPEG coding (see next section) the segments could be "blocks" and the bits are from codewords representing the nonzero frequency components in the blocks. The bits in a block can be ordered according to frequency because the codewords of low frequencies are generally more significant to image quality.

In aggregation, a number of segments from each video source is collected in each aggregation time unit. Let n be the number of video streams and k be the number of segments taken from each stream for aggregation. Then, $m = nk$ is the total number of segments collected from all sources. Let B_t be the number of bits reserved for sharing among the m segments (i.e., B_t should be proportional to the reserved bandwidth on the CBR channel). When B_t is insufficient to accommodate all bits of the segments, for each segment i, we compute $B_i(D)$, the number of bits that must be retained in order to maintain a distortion level of D. Note that $B_i(D)$ is computed as a function of D. To select a specific but common operating distortion level for all segments, we find a distortion level D' such that

$$B_1(D') + B_2(D') + \cdots + B_m(D') = B_t. \tag{1}$$

For each of segment i, the least-significant bits are then dropped so that the number of bits remaining is $B_i(D')$.

In practice, it may not be possible to achieve absolute equality of distortion levels because of the discrete nature of the bits or groups of bits (e.g., codewords) that are dropped. In this case, the aim is to transport no more than B_t bits and to minimize the difference between distortion levels of any two segments.

B. Related Work

Several other papers have also considered performing the statistical multiplexing and compression of several video streams in a related manner [12]–[15]. Our work focuses on the scenario in which the video streams have been pre-compressed in an independent manner (e.g., stored video) before multiplexing. During multiplexing, the data are selectively discarded only when necessary to meet the CBR transmission-bandwidth constraint. From the encoding viewpoint, aggregation is a secondary compression process applied after a preliminary compression process. From the networking viewpoint, it is an adaptation process before data is pumped into the network. References [12]–[15], in contrast, consider directly modifying the the encoding parameters (e.g., the quantization scale) of the video streams based on the CBR bandwidth constraint. A summary of their approaches and brief comments are given below.

In [12]–[14], a smoothing buffer is used to collect outputs from the video streams. The occupancy level of the buffer is used as the feedback to determine the amount of data the encoders may output in the future. The key idea is that when the buffer level is high, the encoders must encode at a lower image quality to prevent buffer overflow; and when the buffer level is low, the encoders can encode at a higher image quality (so as to make full use of the bandwidth). An issue is the prevention of buffer overflow and underflow.

Generally, there is a trade-off between the prevention of buffer underflow/overflow and the smoothness of image quality in successive frames. Stronger feedback in which the encoders react quickly to buffer occupancy changes tends to prevent buffer underflow and overflow better; however, the image quality also tends to fluctuate more along successive frames. Weaker feedback allows the image quality to change slowly; however, the encoders may sometimes react too slowly to prevent underflow or overflow. Also, as in any feedback system, the judicious choice of the form of feedback (e.g., whether the feedback is in terms of absolute buffer level, the rate of change of buffer level, etc.) and the feedback strength are also important as far as system stability is concerned. We shall address these intricacies in a separate paper.

This paper, in contrast to [12]–[14], focuses primarily on the scenario in which *no* smoothing buffer is used. Since there is no smoothing buffer, sudden video-quality degradation due to buffer overflow can never happen. Also, one would not need to deal with the intricate problem of setting the appropriate feedback parameters during the system design. As will be shown, our experiments indicated that when the number of streams is large, the image quality of all streams can be quite smooth without the need for the smoothing buffer.

Reference [15] does not use the smoothing-buffer feedback mechanism. In each frame period, a certain number of bits R_i is allocated to video stream i, and its encoder will code a picture to satisfy this bit allocation. The complexity measure of MPEG TM5 [16] is used to determine R_i. It states that the complexity X_i of a picture i equals the product R_iQ_i, where Q_i is the average quantization scale used over picture i. For a given bit allocation R_t to all pictures (one from each video stream) and with the aim of achieving the same $Q_i = Q$ for all i, the number of bit allocated to picture i can easily be found to be $R_i = X_iR_t/\sum_i X_i$. The encoder i then attempts to meet the bit allocation R_i when encoding picture i. Given the R_i, the resulting quantization scale, however, is not necessarily $Q_i = Q$ because the TM5 complexity measure is an empirical approximation which may not hold true exactly. Therefore, although the goal is to achieve the same Q for all pictures, this is by no means guaranteed.

For the interested readers, some preliminary results related to the work reported in this paper can be found in the conference paper [17]. The thesis [18] documents this work in more detail than this paper does.

III. Network Application Scenarios of Aggregation

With the basic understanding of aggregation, we now elaborate the various application scenarios depicted in Fig. 1. Fig. 1(a) concerns broadcasting in which all the video streams of a video vendor are to be delivered to all receivers. The video vendor may lease a CBR channel from the network provider. An issue is how to provide acceptable video quality with as little bandwidth as possible. In an ATM network, the CBR channel could be in the form of a permanent virtual circuit (PVC) which is used exclusively by the video vendor. There is no bandwidth sharing among the video vendor and other network users. However, with aggregation, there is bandwidth sharing among the video streams of the video vendor.

Fig. 1(b) depicts a VoD scenario in which the video streams are not all destined for a common destination. In fact, only one video stream is to be delivered to each receiver. This, however, does not preclude the application of aggregation. In a public network, there is typically a distribution node (sometimes called remote node) to which many subscribers in a neighborhood are connected. The video vendor may be located in a central office and is serving an area covered by several distribution nodes. Video streams targeted to the same distribution node (but different subscribers) may be aggregated. At the distribution node, the video streams are separated and forwarded to their respective destinations.

In an ATM public network, the VoD vendor may lease a CBR virtual path (VP) from its location to the distribution node. To save cost, the video vendor is interested in minimizing the bandwidth of the CBR. For networking purposes, the leased VP will carry a group of aggregated video streams; each video stream is in turn carried on one particular virtual channel (VC) within the VP. Thus, the individual VC's may be VBR, although the network does not have to deal with that because the VP is CBR. The distinct virtual-channel identifiers (VCI) of the different VC's allow a local ATM switch at the distribution node to separate the video streams and forward them to their respective destinations.

The reader may have noticed that there is no bandwidth sharing for the links between the distribution node and the receivers. However, this is not a major concern. In a real network, it is likely that the cost charged by the network provider depends more heavily on the bandwidth of the common VP rather than the bandwidths of the individual VC's. This is because the bandwidth between the video vendor and the distribution node, if not used by the video vendor, could have been used by other network subscribers or service providers. In contrast, the bandwidth between the distribution node and a particular subscriber can only be used exclusively by the subscriber: if the subscriber does not use it, it will be wasted anyway.

Fig. 1(c) depicts a long-distance video-telephony scenario in which the sources as well as the receivers are geographically separated. A video-telephony vendor may purchase bandwidths from the network provider in order to provide video-telephony service. The subscribers of a common region may forward their video streams to a nearby server of the vendor. At the server, the cells from the streams are depacketized and data from those streams targeted for a common remote area are aggregated and forwarded over a long distance to another server in the neighborhood of the receivers. At the latter server, the video streams are separated and forwarded to their receivers. In this scenario, the goal of aggregation is to save the expensive long-distance bandwidth.

The background assumption of this paper is shown in the scenarios of Fig. 1(a) and (b), and the reader should keep them in mind as a reference in the rest of the paper.

IV. MPEG Video Aggregation System

We now explain in more detail how the concept of aggregation might be applied with respect to the MPEG coding standard. As a preliminary, let us first review the basics of MPEG coding, as well as the problems of cell-level multiplexing from the viewpoint of image quality.

A. MPEG Coding

The schematic of an MPEG coder is shown in Fig. 3 [8]. In the MPEG coding standard [2], spatial information of a frame is partitioned into four layers: frame, slice, macroblock, and block. A frame is the basic unit of display, and is further divided into slices. A slice is a sequence of macroblocks (MB).

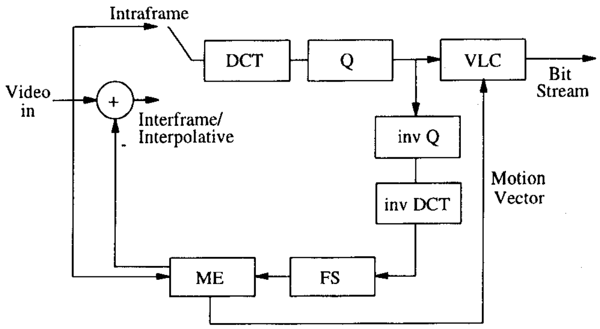


Fig. 3. Schematic MPEG coder. DCT denotes the discrete cosine transform, Q denotes quantization, FS denotes frame storage, VLC denotes variable length coding, inv DCT denotes inverse DCT, inv denotes inverse quantization, and ME denotes motion estimation.

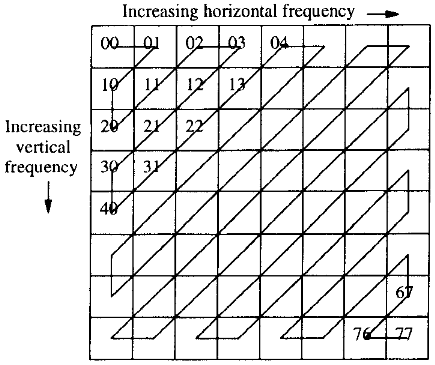

Increasing horizontal frequency
Increasing vertical frequency
00
01
02
03
04
10
11
12
13
20
21
22
30
31
40
67
76
77

Fig. 4. Zigzag scanning order of DCT components.

A 16 × 16 (16 pixels by 16 pixels) macroblock (MB) is the unit for motion compensation, and it consists of 8 × 8 blocks. Discrete cosine transform (DCT) is performed on each 8 × 8 blocks. For color video, an MB consists of four 8 × 8 luminance blocks and two 8 × 8 chrominance blocks. For each frame, there are three choices of coding algorithms: Intraframe, interframe, and interpolative coding.

Intraframe-coded frames (I) are coded independently. The whole I frame undergoes 8 × 8 block-based DCT without referring to other frames. The DCT coefficients are then quantized. The DC coefficients of individual blocks are coded differentially within a slice. For variable-length coding (VLC), each nonzero AC component is first grouped with the run-length of preceding zero components (in zig-zag order, see Fig. 4), and then assigned a codeword from a Huffman table.

For interframe-coded frames (P), temporal redundancy is first reduced by causal MB-based motion compensation, with respect to the preceding I or P frames stored in the Frame Storage. If the motion estimation (ME) error for a MB is less than a threshold (i.e., there is enough redundancy such that interframe coding is worthwhile), then the motion vector (MV) will be differentially and then VLC coded, while the ME error will undergo DCT, coarse quantization, and then VLC. Otherwise, that MB will undergo intraframe coding. Interpolative frames (B) are coded in a way similar to coding P frames; however, the motion compensation is bidirectional with respect to both the preceding and following P (or I) frames. The reader is referred to [2], [8], and [19] for more details on the MPEG standard.

An MPEG coder is characterized by three parameters: q, N, and M. Quantization factor q controls the degree of fineness of quantization. $N-1$ is the number of frames coded between successive I frames, while $M - 1$ is number of B frames coded between successive P frames. A group of frames with $N = 10,\ M = 3$ is as follows:

IBBPBBPBBP

Data in an MPEG-coded video stream are of unequal importance. The header information, MV's and DC components are obviously very important. Among the DCT AC components, those of lower frequencies are more important than those of higher frequencies for two reasons. First, the energy (i.e., amplitude square) of the DCT AC components tends to decrease along the zig-zag scanning order (i.e., energy compaction) [20]. Second, the human vision system is less sensitive to the high frequency signals.

When some data in I and P frames are lost during transmission, the frame contents in the Frame Storages at the coder and decoder become different. Even if no further data is lost, for the following P and B frames, the ME at the coder and decoder will refer to different frame contents as the "baseline" of estimation. Consequently, errors due to data loss of one I or P frame will propagate along the following P and B frames, and this is often referred to as error propagation. The accumulated errors can be cleared by sending an I frame.

B. Shortcomings of the MPEG Video Bundle Scenario with Two-Layer and Cell-Level Multiplexing

In Section II, we have claimed that the shortcoming of the cell-level multiplexing with two-layer coding and transport is that there is no distinguishing between the relative importance of data within an ES and among the separate ES's. Let us now examine its implications for image quality in the context of MPEG coding in detail.

1) Blocky Effects Within a Frame: In multiplexing ES's, the discarding of an ES cell means that those MB's corresponding to this cell (in general, one to ten MB's [9]) will have only their base-layer data transmitted. Therefore, only basic image quality can be provided. In contrast, those MB's having their second-layer data transmitted will provide perfect image quality. Therefore, unless all cells from a frame can be transmitted, the MB's within a frame will have different qualities due to the discarding of some ES cells and the retaining of others. This results in *blocky effects* on the reconstructed image (image appears as clusters).

2) Nonoptimal Image Quality Within a Frame: Although the ES data of a video sequence are of different importance, when they are packetized in cells, there is no further prioritization among them. However, an ES cell is either dropped or transmitted in its entirety. We cannot, say, drop part of an ES cell and part of another ES cell so as to ensure that the missing data are the least significant. As a result, optimality cannot be achieved because some of the dropped data may potentially contribute more to the quality of the reconstructed images than those retained.

3) Fairness of Image Quality Among the Video Sequences: Consider the video streams that are multiplexed. To provide the same image quality, different scene contents may demand different bit rates: sometimes video stream A may need more bandwidth than video stream B, at other times the reverse may be the case. When cells must be dropped at the multiplexer, the multiplexer does not have the knowledge of the significance levels of the ES cells. It is possible that some images (or portions of an image) suffer more visual degradation than others, even if they incur the same cell-loss rate.

One might generalize the two-layer cell-multiplexing approach and set up n layers of different importance where $n > 2$. The cells from layer i will be transmitted only if there is leftover bandwidth after the transmission of all cells from layers below i. The multiplexer must somehow recognize the different levels of importance of cells from different layers. Since there is only one priority bit in the ATM cell header, we cannot use it to distinguish the cells from different layers. A solution is to carry different layers on different VC's and let the multiplexer use the VCI's to distinguish cells from different layers. The receiver, however, now faces the problem of having to assemble data from even more layers before decoding. In short, the n-layer cell multiplexing approach entails additional processing at the receiver.

Let us examine the implications of the n-layer approach for image quality with respect to the three shortcomings of the two-layer approach listed above. As n increases, the n-layer approach should alleviate the blocky effects to the extent that the data from the more important layers have lower loss probabilities.

The problem of not being able to distinguish the different levels of importance among cells of the same layer remains with the n-layer approach. To this extent, we can still say that the second consideration of nonoptimal image quality within a frame with respect to the two-layer approach remains. However, as n increases, distinguishing the different levels of importance among cells of the same layer may not be as visually significant as before.

The fairness of image quality among the video sequences can be achieved as n increases. One way to ensure that is to code the n layers such that the reception of layers up to layer $i \leq n$ will ensure some fixed image quality, say, at D_i. Approximately equal image quality among different video sequences can be achieved when n is large.

Aggregation described below can achieve the same effect as the n-layer cell-multiplexing approach (with very large n) without requiring the receiver to re-assemble data from different layers before decoding.

C. MPEG Video Aggregation

The goal of MPEG video aggregation is to ensure that all MB's contained in the corresponding spatial unit (slice or frame) from all video sequences will provide more or less the same image quality. In our experimental implementation, the video-aggregation process is slotted into slice periods.[1] In every slice period, data for a slice (which are still in the form of VLC codewords and not yet packetized) is collected from every video sequence. A number of bits are allocated for all the slices to be aggregated.

There are at least two approaches to MPEG-based video aggregation. In the bit-plane approach [4], an n-layer strategy (without the cell-level multiplexing described above) is used. For each 8 × 8 block, the DCT coefficients are first coarsely quantized, and these data form the first layer. A distortion metric is then calculated based on the sending of the data. The differences between the original DCT coefficients and the coarsely quantized DCT coefficients are further quantized with a finer quantization step to form the second-layer data. This process is repeated until n layers of data are obtained. Thus, reception of layers up to i corresponds to a certain quantization step ΔQ_i in the corresponding image.

This paper focuses on the frequency-plane approach [4], [8]–[11], [21] in which the lower-frequency codewords in a block are accorded higher importance than the higher-frequency codewords. All the header information, MV's, as well as the first β codewords from every 8 × 8 block are forwarded. This uses up a certain amount of bandwidth. The remaining codewords are then subjected to aggregation with the remaining bandwidth B (note that B may change from aggregation period to aggregation period).

There are two reasons why we might want to exempt the first β codewords from the aggregation process. The first reason is that this will reduce the amount of data to be aggregated and hence the complexity of the process. The second reason, which is more subtle, is that this exemption might be advantageous in a certain variation of aggregation systems (see Section V on partial-reference VAS system). Basically, in that scheme, only the first β codewords are fed back to the frame storage for removing redundancy in later interframe-coded frames, and therefore larger β means better redundancy removal, leading to more efficient compression.

The above argues for a large β. There is, however, a reason that argues for a small β. A smaller β implies a higher degree of bandwidth sharing among video sessions in the aggregation process, and higher bandwidth efficiency can be achieved. When β is large, there is less sharing, and to the extent that β is large enough, there could be no bandwidth sharing at all.

At the beginning of the aggregation process, the distortions of all MB's with only the DC and first β AC components sent are calculated. The MB that has the lowest image quality is identified. If there are remaining bits, the next codeword from all the 8 × 8 blocks contained in this MB will be forwarded. The distortion of that MB will then be updated. Afterwards, the next MB that has the lowest image quality is identified and the step is repeated until all the allocated bits for that slice period have been exhausted.

Note that because the codewords for each 8 × 8 block are arranged with their DCT components in the zig-zag order (see Fig. 4), for each block, the codewords discarded during

[1] In general, the unit of aggregation can be smaller or larger than a slice, depending on the processing capability.

aggregation are of higher frequencies and hence are less important.

The signal-to-noise ratio (SNR) is commonly used as an objective measurement for image quality. However, the actual signal energy of an MB from P or B frames can be found only after it has been decoded (with respect to the reference frame) back into the spatial domain. That is, the actual signal energy of a P- or B- frame MB (after decoding) cannot be obtained by simply summing the energies of the encoded codewords in the MB. Therefore, unless the signal energy in each MB is provided by the MPEG encoders, using SNR as the metric for image quality during aggregation is not feasible (unless, of course, the aggregator decodes the MPEG sequences to find out the signal energies of MB's). Alternatively, we may use noise energy as the metric. Since the amount of energy carried by a codeword (whether in I, P, or B frame) is equal to the amplitude square of its nonzero DCT component, the noise energy in an MB during the aggregation process is equal to the sum of the energies of the discarded (or not-yet-sent) codewords.

Note that using the noise energy as the image-quality metric during aggregation is equivalent to using the peak-SNR (or PSNR), which is also commonly used in video signal processing literature. For PSNR, the ratio of a fixed "peak" signal energy to the noise energy is calculated. That is, the same fixed signal is used for the SNR calculation for all MB's, regardless of the actual signal energies. When performing aggregation, the exact value of this fixed signal energy is not important: if the noise energy of an MB is smaller than of another MB, then the PSNR of the former is larger than the PSNR of the latter, and vice versa.

D. MPEG Video Aggregation System Architecture

We now look at the overall architecture of the MPEG video aggregation system (VAS). An MPEG VAS comprises a group of MPEG video sources, a VAS server, and the ATM adaptation layer (AAL) [1] (Fig. 5).

Video sequences are pre-encoded independently by separate MPEG coders with high quality. The coded data are then forwarded (either directly or from a video storage system) to the VAS server. The VAS server is responsible for aggregating the video sequences, as well as re-assembling the forwarded data block-by-block after aggregation. If the codewords in the pre-coded video sequences have been Huffman-coded, as in the standard MPEG encoding, the VAS server should also Huffman-decode them first before performing aggregation. This is so that the individual codewords can be recovered (note: the boundaries of codewords in an Huffman-coded bit sequence are not known unless Huffman-decoding is performed) and the signal energies associated with them derived. After aggregation is performed, only the codewords selected for transmission are re-Huffman-coded. At the AAL, data of the same video sequence are packetized into cells.

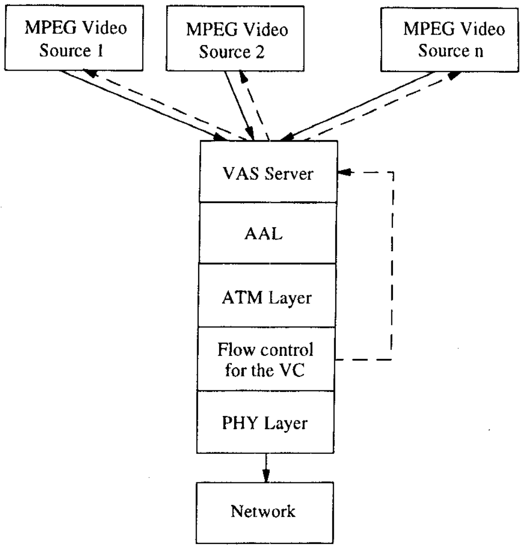


Fig. 5. Schematic diagram of a MPEG VAS. Solid arrows show the flow of data, while dotted arrows show feedback (if any).

In principle, the allocated number of bits for a slice period can either be fixed or varied. In the first case, the output from the VAS enters the CBR channel of the network directly. Temporal statistical multiplexing (i.e., smoothing of traffic generated at different time instants) is confined to a slice period only. In the second case, the output enters a buffer which, in turn, outputs data at a constant rate to the network; the allocated number of bits to a slice (rate control) varies according to the state of the buffer occupancy. The second case allows for smoothing of traffic over a longer time period as compared to the first case, at the expense of more complicated operation and additional delay jitters at the buffer. Our implementation assumes the first case.

Note that aggregation is transparent to MPEG decoders. The forwarded data can be easily put into the standard MPEG format after aggregation. Therefore, at the receivers, standard MPEG decoders can be used without the need for any add-on equipment (at least with two of the schemes to be described in Section V). Note that with the two-layer cell-multiplexing approach, the receivers must combine the two layers before decoding and therefore standard MPEG decoders cannot be used. The property that standard MPEG decoders can be used is an especially attractive feature considering that in many video-distribution systems there are many more receivers than transmitters.

V. Variations of MPEG Video Aggregation System

During aggregation, some codewords of I or P frames may be discarded because of bandwidth shortage. This may cause error propagation. According to how error propagation is dealt with, MPEG VAS's can be categorized into three classes.

For the partial-reference VAS, only the data of the first β codewords, which are not subjected to aggregation, are put back into the Frame Storages of the coder and decoder as the reference for interframe and interpolative coding/decoding. Since the delivery of these data is guaranteed, error propagation will not occur. However, unless β is large, less temporal redundancy can be removed by interframe and interpolative coding this way, and compression becomes less efficient. A judicious choice of β is important because large β also means lesser degree of aggregation, and hence potentially lesser degree of bandwidth sharing among different video streams.

The feedback-reference VAS sends feedback information to the MPEG sources as to which codewords have been chosen for delivery during aggregation, so that their respective encoders can put all delivered components into their Frame Storages. Since the delivery of all forwarded data in the aggregated stream is guaranteed by the network, error propagation will not occur. Compared with the partial-reference VAS, the feedback mechanism here increases the encoders' compression efficiency.

A disadvantage of the feedback-reference VAS is that real-time control of the MPEG encoders is required, which is cumbersome under certain situations. For instance, when the video sources are pre-compressed and stored in the disks for future display, this VAS requires complete decoding and then re-coding of the video sequences during the aggregation process. With the partial-reference VAS, on the other hand, the pre-compressed stored video could be coded in a compatible way such that only the first β codewords of each block are put in the Frame Storages as references. In this way, it is not necessary to perform inverse DCT and DCT in real-time.

Avoidance of error propagation in the above two classes of VAS reduces compression efficiency. A full-reference VAS simply ignores, rather than avoids, error propagation. Thus, at the encoders, all data of reference frames will be put into the Frame Storages (regardless of whether they will be transported). At the receiver side, all the received data of reference frames will be stored at the decoder's Frame Storage. In general, for a given bandwidth, more higher-frequency components can be sent with this approach as more redundancy can be removed. However, the received signals may contain propagated errors due to discrepancies of the data in the sender's and receiver's Frame Storages.

It is difficult to compare the full-reference and partial-reference VAS's from the viewpoint of image quality, as this involves the comparison between degradation due to error propagation and less efficient compression, which depends to a large extent on the scene contents. Nevertheless, when the texture complexity of a video sequences is rather steady (e.g., in video-conferencing), we expect the full-reference VAS to provide better image quality. This is because when successive frames are strongly correlated, the ME error and hence degradation due to error propagation is small. By the same token, the partial-reference VAS should be better when successive frames are not strongly correlated (e.g., video with fast motions). Because a full-reference VAS requires no modification on the frame-storage mechanism of the standard MPEG encoder and decoder, we used it for our experiments described in the next section. Note that for a partial-reference, the encoder is nonstandard in that only the first β codewords of each block are stored even if more that β codewords are produced by the encoder and transmitted. Similarly, the decoder is nonstandard because only β codewords are stored regardless of the number of codewords received. Although the modification required is minimal, in practice, this means that off-the-shelf MPEG encoders and decoders could not be used if the partial-reference scheme is to be adopted.

TABLE I
BITS PER FRAME OF THE SEQUENCES (IN kb)

Sequence name	Bits per frame	
	Mean	Variance
JP1	131.7	28.0
JP2	35.3	20.3
JP3	61.4	32.9
JP4	74.0	20.8
JP5	142.1	31.6
JP6	112.9	31.6
JP7	92.2	22.8
JP8	106.3	24.3

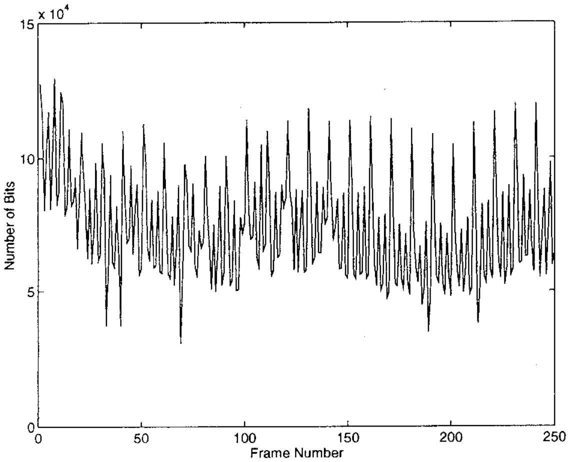


Fig. 6. Bits per frame for the sequence JP4.

VI. EXPERIMENTAL RESULTS

This section presents experimental results that show the performance of video aggregation as compared to two-layer cell-level multiplexing. In addition, effects of varying the amount of allocated bandwidth and the number of aggregated streams are studied.

The video sequences used in the experiments are 8 s in duration. The resolution and frame rate are 320 × 240 and 30 frames per second, respectively (i.e., quarter size of the NTSC standard). All of them were captured from unrelated scenes in the movie *Jurassic Park*, and were coded by an MPEG encoder with $N = 10$, $M = 3$ (see Section IV-A). The traffic in terms of bits per frame of one of the sequences is shown in Fig. 6. Note that sharp peaks occur periodically because of intraframe coding of MPEG coding scheme. In addition, the local average rate of the traffic (say, averaged over 30 frames) varies over time, due to the changes of scene complexity. Some traffic statistics of all the sequences are tabulated in Table I. As can be seen, the traffic of all the sequences is rather bursty.

For both the aggregation and cell-level multiplexing experiments, only the transmission of the header information, MV's and DC components was guaranteed (i.e., $\beta = 1$), while the AC codewords could be discarded when the allocated bits were not enough to accommodate all data. For aggregation, the metric used for comparing the image qualities among the MB's was noise energy (see Section IV-C).

Both aggregation and cell-level multiplexing operations were slotted into slice periods. In each slice period, a fixed number of bits corresponding to the reserved CBR bandwidth were allocated. As a simple means to reduce burstiness of the traffic to be aggregated/multiplexed, the I frames of the eight sequences were disaligned: the first sequence started with frame 1 (the I frame), the second sequence with frame 2, and so on. For simplicity in aggregation, the effect of error propagation was simply ignored (i.e., full-reference VAS was used). For cell-level multiplexing, the ES cells of the streams are taken one-by-one in a round-robin fashion from stream to stream until the bandwidth is exhausted.

In Section VI-A, we compare the performance of video aggregation with cell-level multiplexing from the viewpoint of image quality for a given bandwidth. The effects of varying the amount of bandwidth reserved and the number of video sequences are discussed in Sections VI-B and VI-C.

A. Comparison of Video Aggregation and Cell-Level Multiplexing

Eight video sequences were used in this set of experiments. For simplicity and as an arbitrary choice, the reserved bandwidth of the CBR channel was fixed to be the sum of the mean bit rates of the sequences, where the mean bit rate of a sequence was obtained by averaging over all frames within the eight-second duration of the sequence. Also for simplicity, we assumed that all the 48-byte payload of the ATM cells could be used to carry data from the video streams. In practice, some overhead fields may be necessary. For example, a sequence number may be necessary for the receiver to detect loss cells (note: given that the CBR channel is not shared with other network users, we expect the network to guarantee that there will be no cell loss due to buffer overflow in the network; however, cell loss may still occur due to transmission bit errors in the ATM header, although this should be very rare given the one-byte CRC protection in the ATM header). For interested readers, Ghanbari and Hughes [22] consider the details of introducing overhead during the packing of video data into cells.

For our video sequences, the sum of the mean rates corresponds to 132 cells per slice period. With such bandwidth usage, the average percentage of data lost in aggregation and cell-level multiplexing are 6.19% and 7.70%, respectively. That the latter has a higher percentage loss is due to our implementation, which assumes aggregation/multiplexing (including the cell packing process) in successive slices is independent. Some bandwidth is wasted due to nonfully packed cells at the end of a slice period. There is more loss for the multiplexing case because of layering of data: as each cell can contain data from one layer only, for each sequence, there can be two nonfully packed cells (one from each layer) during a slice period. A more efficient alternative from the bandwidth-usage viewpoint would be to fully pack all cells: in case there was a nonfully pack cell at the end of a slice, some data from the next slice would be packed into the same cell.

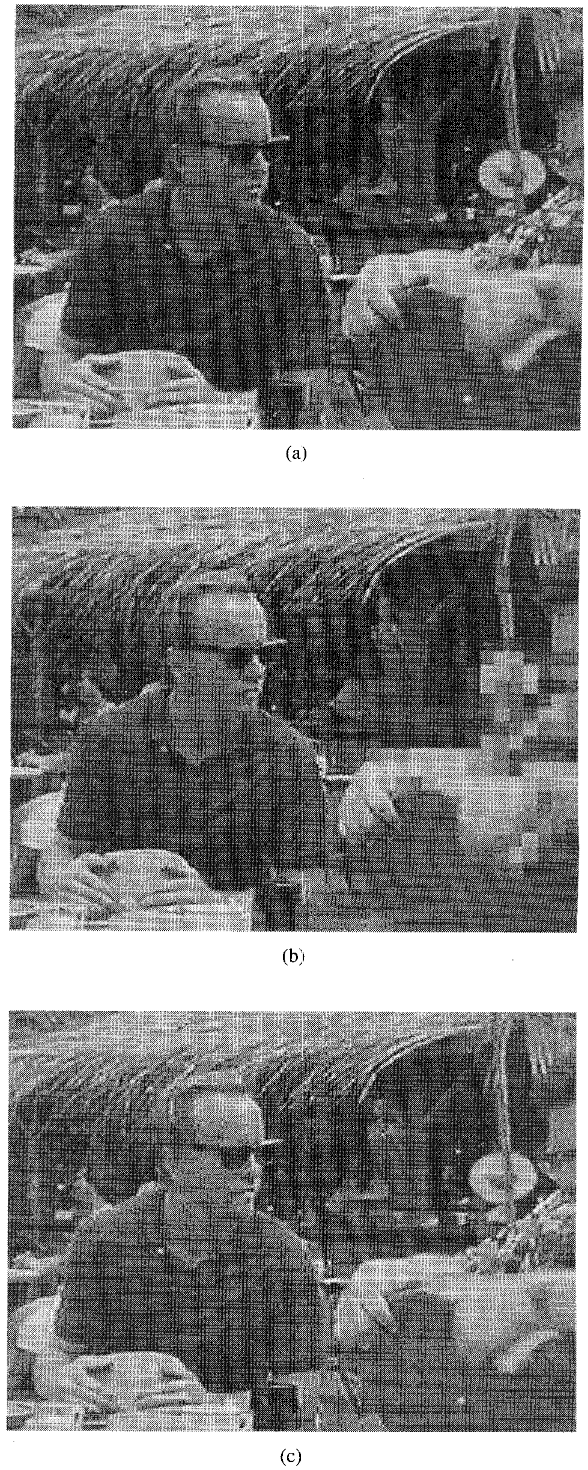


Fig. 7. A frame in the sequence JP1: (a) before MPEG coding, (b) reconstructed from the cell-level multiplexing scenario, and (c) reconstructed from the aggregation scenario.

1) Smoothness of Quality Within a Frame: The original and the reconstructed images after multiplexing and aggregation for a randomly chosen frame are shown in Fig. 7. Compared with the original image [Fig. 7(a)], the post-aggregation image [Fig. 7(c)] is a little "misty," as some of the high frequency signals have been discarded. Note that, however, the quality is smooth within the whole frame. For the post-multiplexing

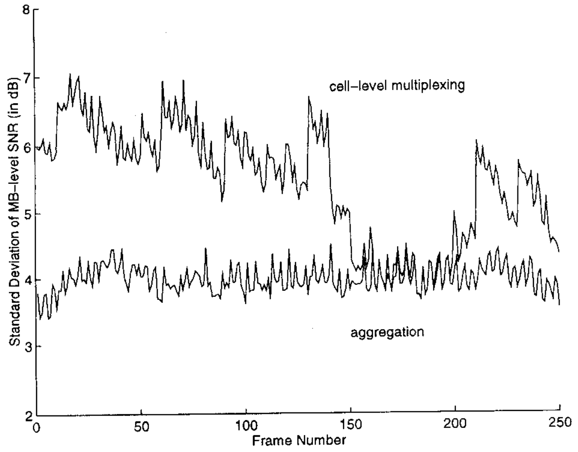


Fig. 8. Standard deviation of SNR of the MB's in a frame, σ_{MB}, along the sequence JP1.

image [Fig. 7(b)], although the left side is very well reconstructed, serious degradation and blocky effects can be easily seen on the right. Although not shown, the other seven images being multiplexed also have blocky effects on the right. The blocky effects cluster on the right because of the round-robin cell multiplexing and the exhaustion of bandwidth at the ends of slices. Instead of taking ES cells from a slice from left to right during multiplexing, interleaving can be performed. The net effect is that the blocky effects will be distributed more sparsely but throughout the whole screen rather than concentrated on the right region.

For a frame, let us define the SNR of an MB j (expressed in dB) as

$$\mathrm{SNR}_{\mathrm{MB}_j} = 10 \log_{10} \frac{\sum_l s_l^2}{\sum_l \left(s_l - s_l'\right)^2} \tag{2}$$

where s_l is the original (pre-MPEG compressed) value of pixel l, s_l' is the pixel value after aggregation or multiplexing, and the summations are taken over all pixels l in the MB. The smoothness of the image quality of a frame can be measured objectively by the standard deviation of $\mathrm{SNR}_{\mathrm{MB}_j}$ over all MB's in the frame,

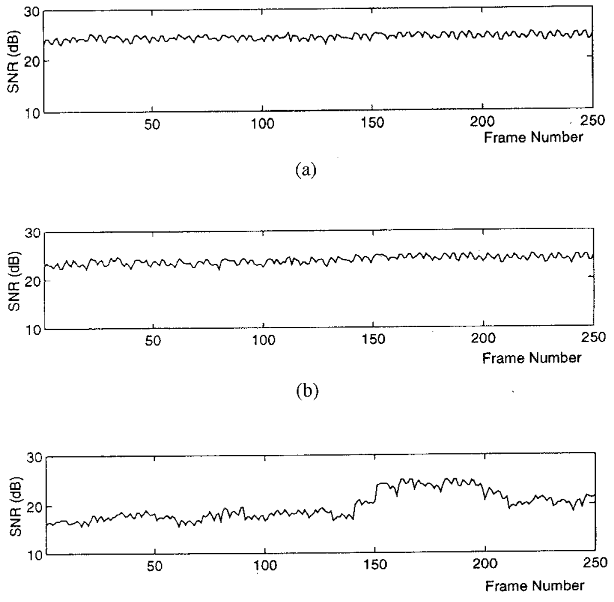
$$\sigma_{\mathrm{MB}} = \sqrt{\frac{\sum_j (\mathrm{SNR}_{\mathrm{MB}_j} - \overline{\mathrm{SNR}_{\mathrm{MB}}})^2}{\text{Number of MB's in frame}}} \tag{3}$$

where the summation is over all MB's in the frame and

$$\overline{\mathrm{SNR}_{\mathrm{MB}}} = \frac{\sum_j \mathrm{SNR}_{\mathrm{MB}_j}}{\text{Number of MB's in frame}}. \tag{4}$$

The larger the σ_{MB} of a frame, the less smooth is the image of the frame. Fig. 8 plots σ_{MB} along one of the sequences after aggregation/multiplexing. As can be seen, the aggregated sequence has much lower σ_{MB} for most of the frames.

Thus, both subjectively and objectively, we have shown that aggregation provides much smoother quality within a frame than cell-level multiplexing does.



(a)

(b)

(c)

Fig. 9. SNR of the frames in the sequence JP1: (a) before aggregation/multiplexing, (b) after aggregation, and (c) after multiplexing.

2) Quality Degradation Due to Aggregation and Multiplexing: Subjectively, we can see from Fig. 7 that the quality of the post-aggregation image is superior to that of the post-multiplexing image.

For objective comparison among frames, frame-level SNR defined as follows is used to measure the image quality of a frame

$$\mathrm{SNR} = 10 \log_{10} \frac{\sum_l s_l^2}{\sum_l \left(s_l - s_l'\right)^2} \tag{5}$$

which is like (2) except that the summations are over all pixels in the frame.

The SNR for all frames along the sequence JP1 before and after aggregation/multiplexing are plotted in Fig. 9. Comparing the post-aggregation sequence [Fig. 9(b)] with the post-multiplexing one [Fig. 9(c)], the former has higher and more steady SNR.

The post-aggregation sequence has more steady image quality because even when the allocated bandwidth is not sufficient to accommodate all the data from all the eight video sequences (e.g., the first 100 frames in Fig. 10), the image quality degradation of the frames is minimized by dropping the least important data. As a result, the image quality of these frames is not much different from the perfect quality received when the bit rate of the total traffic is lower than the allocated bandwidth (e.g., frames 150 to 200 in Fig. 10). In contrast, with cell-level multiplexing, when the allocated bandwidth is insufficient, some important data in the ES cells may be dropped, resulting in more serious degradation.

In our experiment, the SNR results of all the eight sequences are qualitatively similar to that of the JP1 shown here. It does not yield additional insight and information to present all these results. In the following, we present some processed

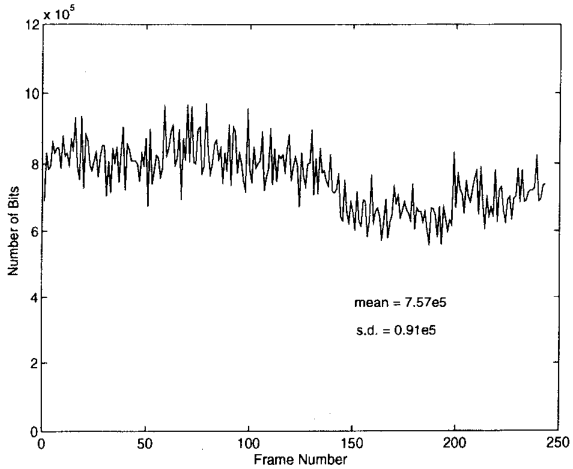


Fig. 10. Bits per frames for the total traffic of $n = 8$.

SNR statistics (i.e., mean and standard deviation) of these sequences.

To focus on the image quality *degradation* resulting from aggregation/multiplexing, let us define the SNR difference of a frame, $\Delta\mathrm{SNR}$, to be the difference between the SNR immediately before and after aggregation/multiplexing. Therefore, higher $\Delta\mathrm{SNR}$ means more signal energy has been dropped during aggregation/multiplexing, and therefore more serious degradation in image quality.

For notational clarity, let us use index i to refer to a frame position and index k to refer to the sequence. Thus, SNR_{ik} is the SNR of frame i of sequence k, and $\Delta\mathrm{SNR}_{ik}$ is the SNR difference of frame i of sequence k.

For each of the sequence k, let us define $\Delta\mathrm{SNR}_k$ and $\sigma_{\Delta\mathrm{SNR}_k}$ to be the mean and standard deviation, respectively, of the SNR difference over all frames in the sequence

$$\Delta\mathrm{SNR}_k = \frac{\sum_i \Delta\mathrm{SNR}_{ik}}{\text{Number of frames in sequence } k} \tag{6}$$

and

$$\sigma_{\Delta\mathrm{SNR}_k} = \sqrt{\frac{\sum_i (\Delta\mathrm{SNR}_{ik} - \Delta\mathrm{SNR}_k)^2}{\text{Number of frames in sequence } k}}. \tag{7}$$

We use $\Delta\mathrm{SNR}_k$ as the metric for measuring the average image-quality degradation of sequence k, while $\sigma_{\Delta\mathrm{SNR}_k}$ for the steadiness of the image quality of the sequence.

The $\Delta\mathrm{SNR}_k$ and $\sigma_{\Delta\mathrm{SNR}_k}$ for aggregation and multiplexing for the eight sequences are given in Tables II and III, respectively. We see that video aggregation provides better and more steady image quality than multiplexing does for all but one sequence.

TABLE II
MEAN SNR DIFFERENCES, $\Delta\mathrm{SNR}_k$, OF THE SEQUENCES (IN db)

Sequence name	$\Delta\mathrm{SNR}_k$	
	Aggregation	Multiplexing
JP1	0.40	4.64
JP2	0.42	0.09
JP3	0.56	3.06
JP4	0.68	3.16
JP5	0.43	5.15
JP6	0.35	2.18
JP7	0.39	1.92
JP8	0.46	2.45
Mean ($\overline{\Delta\mathrm{SNR}_k}$)	0.46	2.83

TABLE III
STANDARD DEVIATION OF SNR DIFFERENCES, $\sigma_{\Delta\mathrm{SNR}_k}$, OF THE SEQUENCES (IN db)

Sequence name	$\sigma_{\Delta\mathrm{SNR}_k}$	
	Aggregation	Multiplexing
JP1	0.38	2.53
JP2	0.38	0.27
JP3	0.54	3.49
JP4	0.55	1.72
JP5	0.40	2.61
JP60.41	0.41	2.18
JP72.04	0.38	2.04
JP8	0.45	2.12
Mean ($\overline{\sigma_{\Delta\mathrm{SNR}_k}}$)	0.44	2.12

In order to compare the overall image quality of the sequences provided by aggregation and multiplexing, we further look into the mean of $\Delta\mathrm{SNR}_k$ and $\sigma_{\Delta\mathrm{SNR}_k}$ across the eight sequences, defined as

$$\overline{\Delta\mathrm{SNR}} = \frac{\sum_k \Delta\mathrm{SNR}_k}{n} \tag{8}$$

and

$$\overline{\sigma_{\Delta\mathrm{SNR}}} = \frac{\sum_k \sigma_{\Delta\mathrm{SNR}_k}}{n} \tag{9}$$

where n is the number of sequences being aggregated/multiplexed (i.e., $n = 8$ in this experiment). As both $\overline{\Delta\mathrm{SNR}}$ and $\overline{\sigma_{\Delta\mathrm{SNR}}}$ are lower in the aggregation scenario, we conclude that overall, aggregation can achieve better and more steady image quality for the sequences than multiplexing can.

3) Fairness Among the Sequences: As a metric for evaluating the fairness of image quality across the n sequences, let us define $\sigma_{\Delta\mathrm{SNR}_i}$ to be the standard deviation of $\Delta\mathrm{SNR}_{ik}$ over the n sequences in frame period i

$$\sigma_{\Delta\mathrm{SNR}_i} = \sqrt{\frac{\sum_k (\Delta\mathrm{SNR}_{ik} - \Delta\mathrm{SNR}_i)^2}{n}} \tag{10}$$

where

$$\Delta\mathrm{SNR}_i = \frac{\sum_k \Delta\mathrm{SNR}_{ik}}{n} \tag{11}$$

is the mean SNR difference across the n sequences in the frame period i. Larger $\sigma_{\Delta\mathrm{SNR}_i}$ means that in frame period i, the image quality degradation of the sequences is less uniform; in other words, there is a lower degree of fairness among n the sequences.

$\sigma_{\Delta\mathrm{SNR}_i}$ for all frame periods for both aggregation and multiplexing scenarios are plotted in Fig. 11. Note that for all frame periods, $\sigma_{\Delta\mathrm{SNR}_i}$ is much lower in the aggregation scenario. This verifies that aggregation can achieve better fairness among the sequences than multiplexing can.

TABLE IV
MEAN SNR DIFFERENCE ACROSS ALL FRAMES AND ACROSS THE EIGHT SEQUENCES, $\overline{\Delta\text{SNR}}$ (IN db) AND PERCENTAGE OF DATA LOSS FOR DIFFERENT AMOUNT OF BANDWIDTH ALLOCATED (NORMALIZED BY THE MEAN RATE)

Allocated Bandwidth	$\overline{\Delta SNR}$		*Mean Percentage of Data Loss*	
	Aggregation	Multiplexing	Aggregation	Multiplexing
1.0	0.46	2.83	6.19	7.70
0.9	0.86	4.28	11.67	13.69
0.8	1.48	5.75	19.94	21.48
0.75	1.79	6.49	23.93	26.13
0.6	3.10	8.26	38.96	40.22
0.5	4.04	9.10	47.85	49.70

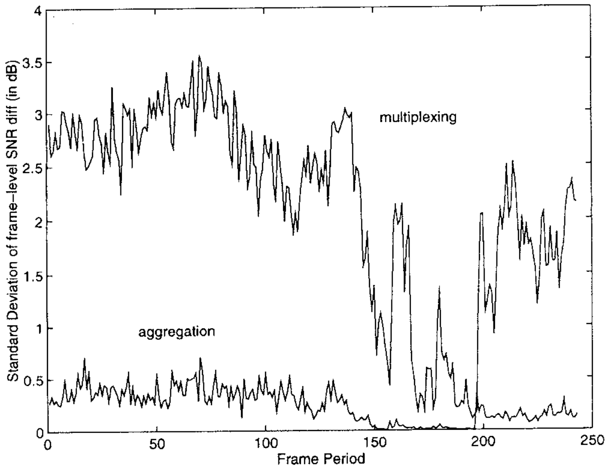


Fig. 11. Standard deviation of SNR difference across the sequences, $\sigma_{\Delta\text{SNR}_i}$ (in dB).

B. Varying Amount of the Allocated Bandwidth

Table IV summarizes the results when the bandwidth allocated (normalized by the mean rate, i.e., 132 cells per slice period) to the eight sequences being aggregated or multiplexed is varied. Specifically, $\overline{\Delta\text{SNR}}$ and the corresponding percentage of data lost are given.

As expected, for both aggregation and cell-level multiplexing, when less bandwidth is allocated, more data is dropped and hence $\overline{\Delta\text{SNR}}$ increases. Nevertheless, for the same bandwidth allocation, aggregation always provides better image quality than cell-level multiplexing does. Alternatively, for the same image quality requirement, aggregation uses less bandwidth. Note that if the tolerable image degradation is up to 2.8 dB, the reduction of bandwidth usage by aggregation is more than 25% with respect to the mean bandwidth of the MPEG video streams, which is needed by cell-level multiplexing with two-layer coding.

C. Varying Number of Sequences

We now present results related to varying the number of video sequences, n, in video aggregation. In all cases, the allocated bandwidth of the CBR channel is equal to the sum of the mean rates of the n sequences.

Table V gives both $\overline{\Delta\text{SNR}}$ and $\overline{\sigma_{\Delta\text{SNR}}}$ (defined in (8) and (9), respectively) over all the n sequences as n is varied.

TABLE V
IMAGE QUALITY OF THE SEQUENCES TRANSMITTED BY VIDEO AGGREGATION WITH DIFFERENT n

n	$\overline{\Delta\text{SNR}}$ in (dB)	$\overline{\sigma_{\Delta\text{SNR}}}$ in (dB)
8	0.46	0.44
4	0.89	0.73
2	3.96	1.32
1	6.16	1.54

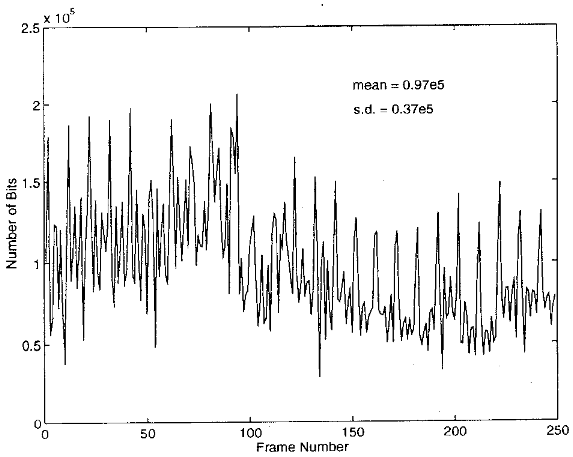


Fig. 12. Bits per frames for the total traffic of $n = 2$.

As n is reduced, both $\overline{\Delta\text{SNR}}$ and $\overline{\sigma_{\Delta\text{SNR}}}$ increase. In other words, when fewer sequences are aggregated, the average degradation and steadiness of image quality also decrease. This is because when n is reduced, the total input traffic becomes more bursty. Compare, for example, Fig. 10, where $n = 8$, with Fig. 12, where $n = 2$. When n is small, sharp peaks occur whenever one of the sequences outputs at high bit rate (e.g., due to I frames or scene changes). At other times, the total bit rate remains lower than the mean rate. As long as aggregation is slotted into slice periods, data will be dropped when the peaks occur. Meanwhile, all data can be transmitted (even with unused excess bandwidth) at other times. As a result, the degradation becomes severe at peak times and the image quality is not steady over time. When n is large, the sharp peak of a sequence can be absorbed by a larger number of other sequences that do not need that much bandwidth at that moment in time. Consequently, the degradation is less severe and the image quality more steady.

The above observation suggests that when n is small (e.g., $n < 4$), it is better to have temporal statistical multiplexing in addition to the bandwidth sharing among the sequences. As mentioned before, temporal statistical multiplexing can be achieved by having a smoothing buffer at the output of the VAS server (the flow control block in Fig. 5). Bandwidth allocated to each slice period is time varying and it depends on two factors: the current buffer occupancy and the the intrinsic bandwidth demand for the slices being aggregated, which could be measured using the bit-versus-distortion function employed in the aggregation process. The exact algorithm for bandwidth allocation within this framework is out of the scope of this paper. Although better image quality can be obtained with temporal smoothing, more complicated operation in the VAS server will be needed. Furthermore, the delay and delay jitter introduced at the buffer must also be dealt with at the receiving end.

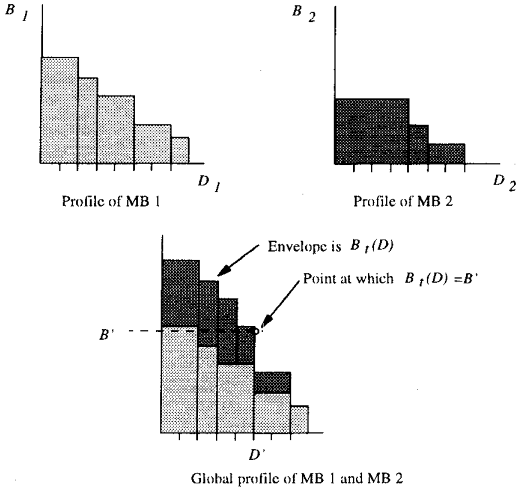
Another point that should be noted with temporal smoothing is that it is only effective in smoothing out traffic within a time window that corresponds to the buffer size. In real movies, high bit rate may occur for a sustained period of time [23] (in the order of several minutes) which could be longer than the buffer would allow. Therefore, there is a limit on what temporal smoothing can do. Fortunately, when n (say, $n \geq 8$) is sufficiently large, smoothing the input traffic solely across the n sequences with aggregation and without a smoothing buffer is likely to provide good-quality images, since the likelihood of all n sequences peaking together is small. This obviates the need for temporal smoothing.

For cell-level multiplexing, as we have already seen, smoothing across the sequences without the smoothing buffer is not enough even when n is as large as eight. Thus, we expect that temporal smoothing with a buffer will have a more beneficial effect for cell-level multiplexing. Nevertheless, as indicated in Fig. 10, besides the bit rate variation from frame to frame, the average bit rate (say averaged over 30 frames) also varies on a larger time scale. This can not be smoothed out with a buffer size that is reasonably small (say smaller than 30 frames) to avoid large delay and delay jitter. Therefore, we would still expect data to be dropped during the peaks of the average bit rate. When this occurs, aggregation will be superior to cell-level multiplexing.

VII. Complexity of Video Aggregation

The previous section concerns the performance of video aggregation. In this section, we discuss the complexity of implementing video aggregation. We shall discuss two software implementations that have been tested. The preceding experiments were based on an implementation that employs sorting and merging. In this implementation, at any point during the execution of the program, we keep track of the remaining bandwidth, the codewords that have been selected for transmission for each MB, and the associated noise or distortion level of each MB if the not-yet-selected codewords were to be dropped.

Initially, the remaining bandwidth is set to the number of bits allocated to the slices being aggregation subtracts the bits used by the headers, the first β codewords, etc. The distortion levels of all MB's are calculated based on the transmission of only the first β codewords. The MB's being aggregated are then sorted according the magnitudes of their distortion levels.



Profile of MB 1

Profile of MB 2

Global profile of MB 1 and MB 2

Fig. 13. The profile approach for implementing video aggregation.

After the above initialization, the MB with the largest distortion level is selected and its next codeword will be chosed for transmission if there is sufficient bandwidth left. Then, the new distortion level of the MB is calculated and it is merged with the rest of the sorted MB's. This process is iterated until the inclusion of an additional codeword will violate the bandwidth constraint, at which point the aggregation process for that slice period is complete.

Using heap sort, the order of complexity of the aggregation algorithm is $mc \log m$, where m is the total number of MB's from all the video sequences in a slice period, and c is the number codewords in each MB. Recall that the video streams are standard MPEG video data which have previously been Huffman-coded, it is necessary to also Huffman-decode them before aggregation is performed in order to find out the boundaries between codewords. With the hash-table lookup approach for Huffman-decoding, the complexity is of order mc.

When the above implementation was run on a Sun Sparc20 dual-processor workstation, a total of about 90 s was needed for eight 8-s video sequences. The initialization of computer memory, the I/O of video data, and other overhead functions took about 10 s. The Huffman-decoding took about 30 s, and the sort/merge aggregation process took about 50 s (the sorting and merging took about 30 s, and the computation of distortion levels during the iterations took about 20 s).

The complexity of the sort-and-merge algorithm is nonlinear in m. In case we want to increase the aggregation unit (say, frame level rather than slice level), the processing time will increase more than linearly. An alternative approach, based on the construction of the rate-distortion profiles of the MB's, has also been implemented. Fig. 13 illustrates this approach.

The bits-versus-distortion profile of each MB is constructed as follows. First, the number of bits required to transmit all the codewords in the MB is calculated. This corresponds to a distortion level of zero. Next, we compute the number of bits required as well as the associated distortion if the codeword with the highest nonzero DCT coefficient were to be dropped. This is repeated until the possibility of dropping all but the first

β codewords has been considered. Fig. 13 shows the profiles of two hypothetical MB's constructed this way. Notice that profiles are staircase functions.

A global profile of all the aggregated MB's are constructed by adding the individual profiles of the MB's in the vertical direction. That is, the total number of bits required for a given distortion level for all the MB's is obtained. Fig. 13, for illustration, assumes that only the two MB's are to be aggregated and show the result of the above steps. Each point (B, D) of the the envelope, $B_t(D)$, represents the number of bits B required in order that none of the MB's has a distortion more than D.

After the global profile has been constructed, we find the point at which the envelope $B_t(D)$ intersects with the horizontal line $B_t = B'$, where B' is the number of bits shared by all MB's. The corresponding distortion D' is the operating distortion level for all MB's during this aggregation period. The number of codewords to be selected from each MB for transmission can then be obtained with this D' from the local profile of the MB.

The complexity of the profile-aggregation algorithm is of order mc. When tested using the same hardware platform and assumptions as in the sort-and-merge approach, a total of 60 s were needed. Of the 60 s, 10 s were for I/O and other overhead functions, 30 s for Huffman-decoding, and 20 s for aggregation. Essentially, the profile approach eliminates the 30 s of sorting and merging time, and computation related to aggregation is restricted to calculation of the distortion levels. The advantage of the profile approach over the sort-and-merge approach will be more pronounced if aggregation is performed at the frame level. Essentially, the linearity of the complexity mc means that there is no penalty associated with increasing the aggregation time unit in the profile approach.

Note that parallel processing can be employed to further reduce the run-time of the VAS server. For instance, we could devote *one processor to each sequence*. A processor will Huffman-decode the MB's of its associated sequence and construct their profiles. It then merges the profiles into a "subglobal" profile by adding up the bits in the vertical direction as described above. A coordinating processor (this could be a separate processor or one of the processors devoted to individual sequences) will then gather the subglobal profiles from all processors, compute the global profile and the operating distortion level D'. The value of D' is fed back to the individual processors, which then discard codewords and perform Huffman coding independently. With this parallel design and with the advancement of computer technology, we believe that a software VAS server can aggregate a bundle of video in real time. Of course, hardware implementation, although not discussed here, may also be considered for real-time operation.

VIII. Conclusion

This paper has investigated video aggregation, a concept that integrates compression and multiplexing of video information. In video aggregation, a bulk of fixed bandwidth is allocated to a group of video sessions, and it is up to the video sessions to adapt their traffic to the fixed bandwidth. With the fixed CBR output, video aggregation frees the network operator from the complicated bandwidth-allocation and tariff problems. It has been shown experimentally (based on the objective SNR measure and subjective observation of image quality) that video aggregation can provide better image quality than multiplexing at the cell level. In particular, two important goals are achieved: 1) smooth and good image quality for the frames of each video session and 2) fairness of image quality among the video sessions.

An issue that has not been addressed in this paper is the determination of the CBR bandwidth required by a group aggregated video streams. We briefly discuss this issue below, leaving the details for further studies. The experiments we performed were over short sequences of eight seconds in duration only. The bit rates over the short sequences may not be indicative of the bit-rate requirements over a longer time horizon. One reason we chose the short sequences for the SNR study is that it is meaningless to talk about the average SNR over a very long sequence: two long sequences may have the same average SNR, but one may have highly fluctuating SNR and the other not, and the former is certainly less desirable.

As shown in Table IV, for aggregation, very high loss rate can be tolerated without very significant SNR degradation in a small time window of 8 s. There are two ways to address the bandwidth-allocation problem. The first is to allocate bandwidth based on the aggregate data rate of the aggregated streams as follows. A fixed but small time window (e.g., 8 s) is used to calculate short-term mean rate of the combined streams; find the peak short-term mean rate over time; and then allocate a fraction of this peak mean rate as the CBR bandwidth such that the corresponding loss rate is tolerable using a empirically obtained table (e.g., one like Table IV, but constructed over many experiments).

Another possibility, which is simpler, is to forget about allocating bandwidth with such detailed considerations. The same amount of CBR bandwidth is given to all groups of aggregated streams, hoping that the streams within each group do not all peak together. If they do, aggregation (as opposed to cell multiplexing) is used to alleviate the SNR degradation. This simple bandwidth allocation may work because aggregation is highly tolerant of loss. This approach can also be motivated from the angle of image-quality improvement rather than bandwidth efficiency. Consider a video stream that is coded with 1.5 Mb/s CBR rate. Because of the CBR nature, there is bound to be image-quality fluctuations. If we were to VBR-encode eight video streams and then aggregate them with a CBR bandwidth of 8 × 1.5 Mb/s, the image-quality fluctuations could be reduced.

Acknowledgment

The authors thank H. Chan and W. H. Lee for writing the video aggregation software. The anonymous reviewers have provided numerous useful suggestions on how the initial draft of this paper could be improved.

References

[1] M. De Prycker, *Asynchronous Transfer Mode: Solution for Broadband ISDN*. London, U.K.: Ellis Horwood, 1993.

[2] D. Le Gall, "MPEG: A video compression standard for multimedia applications," *Commun. ACM*, vol. 34, pp. 47–58, Apr. 1991.

[3] D. Deloddere, W. Verbiest, and H. Verhille, "Interactive video on demand," *IEEE Comm. Mag.*, May 1994.

[4] N. Ohta, *Packet Video: Modeling and Signal Processing.* Norwood, MA: Artech, 1994, p. 164.

[5] D. Reininger, D. Raychaudhuri, B. Melamed, B. Sengupta, and J. Hill, "Statistical multiplexing of VBR MPEG compressed video on ATM networks," in *Proc. IEEE INFOCOM'93*, pp. 919–926.

[6] S. S. Dixit and P. Skelly, "Video traffic smoothing and ATM multiplexer performance," in *Proc. IEEE GLOBECOM'91*, pp. 239–243.

[7] R. Coellco and S. Tohme, "Video coding mechanism to predict video traffic in ATM network," in *Proc. IEEE GLOBECOM'93*, pp. 447–450.

[8] P. Pancha and M. El Zarki, "MPEG coding for variable bit rate video transmission," *IEEE Commun. Mag.*, May 1994.

[9] M. Ghanbari and V. Seferidis, "Cell-loss concealment in ATM video codecs," *IEEE Trans. Circuit Syst. Video Technol.*, vol. 3, no. 3, June 1993.

[10] M. Ghanbari, "Two-layer coding of video signals for VBR networks," *IEEE J. Select. Areas Commun.*, vol. 7, no. 5, June 1989.

[11] F. Kishino, K. Manabe, Y. Hayashi, and H. Yasuda, "Variable bit-rate coding of video signals for ATM networks," *IEEE J. Select. Areas Commun.*, vol. 7, no. 5, June 1989.

[12] T. Koga, Y. Iijima, Iinuma, and T. Ishiguro, "Statistical performance analysis of an interframe encoder for broadcast television signals," *IEEE Trans. Commun.*, vol. COM-29, no. 12, pp. 1868–1876, Dec 1981.

[13] B. G. Haskell and A. R. Reibman, "Multiplexing of variable rate encoded streams," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, no. 4, pp. 417–424, Aug. 1994.

[14] A. Guha and D. J. Reininger, "Multichannel joing rate control of VBR MPEG encoded video for DBS applications," *IEEE Trans. Consumer Electron.*, vol. 40, no. 3, Aug. 1994.

[15] G. Keesman and D. Elias, "Analysis of joint bit-rate control in multiprogram image coding," in *Proc. SPIE Visual Communications Image Processing*, 1994, pp. 1906–1917.

[16] MPEG-2 Test Model 5, *Document ISO/IEC JTC1/SC29/WG11/93-400*, Test Model Editing Committee, Apr. 1993.

[17] C.-Y. Tse and S. C. Liew, "Video aggregation: An integrated video compression and multiplexing scheme for broadband networks," in *Proc. IEEE INFOCOM'95*, pp. 439–446.

[18] C.-Y. Tse, "Adaptation of variable-bit-rate compressed video for transport over a constant-bit-rate communications channel in broadband networks," M.Phil. thesis, Chinese University of Hong Kong, 1995.

[19] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Trans. Consumer Electron.*, vol. 38, no. 1, Feb. 1992.

[20] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithm, Advantages, and Applications.* New York: Academic, 1990, p. 170.

[21] A. Eleftheriadis and D. Anastassiou, "Optimal data partitioning of MPEG-2 coded video," in *Proc. 1st Int. Conf. Image Processing*, Nov. 1994.

[22] M. Ghanbari and C. J. Hughes, "Packing coded video signals into ATM cells," *IEEE/ACM Trans. Networking*, vol. 1, no. 5, Oct. 1993.

[23] M. W. Garrett, "Contributions toward real-time services on packet switched networks," Ph.D. dissertation, Columbia University, New York, May 1993.

Soung C. Liew (S'84–M'87–SM'92) received the S.B., S.M., E.E., and Ph.D. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, in 1984, 1986, 1986, 1988, respectively.

He joined Bellcore, NJ, in March 1988. Since July 1993, he has been Associate Professor at the Chinese University of Hong Kong. He has diverse research interests and has published actively in various areas related to broadband communications, including fast packet switching, broadband network control, video transport, and WDM optical networks. His recent research efforts center around the issue of processing and adapting video information for efficient network transport. He also recently (November 1995) launched a project to explore how advanced multimedia and networking technologies can be used to improve university education. The goal is to build a Web-like multimedia lecture-on-demand system. He initiated and is coordinating an ATM testbed network which interconnects three major universities in Hong Kong. He holds two U.S. patents in routing and packet switching.

Dr. Liew is a member of Sigma Xi and Tau Beta Pi.

Chi-yin Tse (S'94–M'95) received the B.Sc. degree in physics and computer science and the M.Phil. degree in information engineering from the Chinese University of Hong Kong, in 1993 and 1995, respectively.

He is currently a Research Assistant in the Broadband Communications Laboratory of the Chinese University of Hong Kong, working on various areas related to video transmission over broadband networks.