Journal of Electronic Imaging 10(1), 329–338 (January 2001).

MPEG-compliant joint source/channel coding using discrete cosine transform and substream scheduling for visual communication over packet networks

Seong-Whan Kim
IMT-2000 SW-1 Lab. of Cellular Division
LGIC Inc.
An-Yang, Korea
E-mail: ksh@lgic.co.kr

Shan Suthaharan
Tennessee State University
Department of Computer Science
Nashville, Tennessee

Heung-Kyu Lee
Korea Advanced Institute of Science and Technology
Department of Computer Science
373-1 Kusong-Dong
Yusong-Ku
Taejon 305-701, Korea

K. R. Rao
University of Texas at Arlington
Department of Electrical Engineering
Arlington, Texas

Abstract. *Quality of Service (QoS)-guarantee in real-time communication for multimedia applications is significantly important. An architectural framework for multimedia networks based on substreams or flows is effectively exploited for combining source and channel coding for multimedia data. But the existing frame by frame approach which includes Moving Pictures Expert Group (MPEG) cannot be neglected because it is a standard. In this paper, first, we designed an MPEG transcoder which converts an MPEG coded stream into variable rate packet sequences to be used for our joint source/channel coding (JSCC) scheme. Second, we designed a classification scheme to partition the packet stream into multiple substreams which have their own QoS requirements. Finally, we designed a management (reservation and scheduling) scheme for substreams to support better perceptual video quality such as the bound of end-to-end jitter. We have shown that our JSCC scheme is better than two other two popular techniques by simulation and real video experiments on the TCP/IP environment.* © *2001 SPIE and IS&T.* [DOI: 10-1117/1.1311794]

Paper 98032 received July 29, 1998; revised manuscript received July 7, 2000; accepted for publication July 19, 2000.
1017-9909/2001/$15.00 © 2001 SPIE and IS&T.

1 Introduction

Quality of Service (QoS) guarantee in real time communication for multimedia applications is significantly important, because more multimedia services, such as video on demand, video conference, and live video and audio broadcasting, require guaranteed services in the distributed heterogeneous environment. These applications require high bandwidth and real time. The real-time constraint is however not as critical as hard real-time systems and is usually classified as soft real time. However, they impose other requirements which are peculiar to multimedia traffic, for example, bounded delay, bounded jitter, bounded packet loss, and bounded synchronization gap. These requirements are defined as QoS of the real-time multimedia application.[1,2]

The problem is that how to support QoS in the lossy packet network. One group has made a proposal for an architecture for the future global information infrastructure (GII),[3] and we used their architecture for consistency. The architecture is composed of application layer, service layer, and bitway layer. The service layer provides a set of common generic capabilities that are available to all applica-

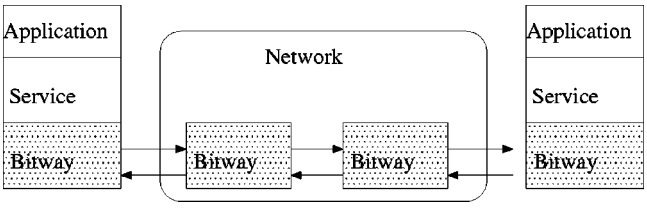


Fig. 1 Architecture of the GII including the network.

tions and conditions data for the bitway. The bitway layer establishes connections between end points, carries data between the end points, and monitors its own performance.[3] In signal processing, source coding and channel coding are used for signal transmission, where source coding removes signal redundancy as well as signal components that are subjectively unimportant, and channel coding adds controlled redundancy so that transmission impairment such as packet loss or bit errors can be reversed. Source coding is used in the service layer to increase the traffic-carrying capacity of transmission links within the bitway layer, and channel coding is used in the bitway layer to increase the traffic capacity of a network. Joint source/channel coding (JSCC) is a way to increase the traffic capacity of a network and also maximize a subjective image quality by exchanging some information between source coding and channel coding.[4–6]

The multimedia data characteristics and QoS parameters can be varied with substreams (which we define as parts of media, for example, subblocks or subband of video/audio signal). If we support application of QoS with further specified substream level QoS, we can achieve more efficient QoS management.[4–6] To support the substream model for an existing frame by frame approach such as the MPEG video coding standard, we designed a simple MPEG transcoder to convert the MPEG coded stream into a series of packets. We also designed a substream scheduling scheme, which interacts with the proposed source coding scheme to maximize the perceptual video quality, and showed by simulation that our JSCC scheme is better than others.

In Sec. 2, we review the basic communication architecture which is used for the packet network model in this paper. In Sec. 3, we review the basic characteristics of the human visual system (HVS) and the basic idea of frameless rendering. In Sec. 4, we designed JSCC, where we used the MPEG transcoder, which packetizes the MPEG coded stream and the JSCC coupler which classifies packets based on motion information. In Sec. 5, we propose substream scheduling to maximize perceptual quality in our JSCC framework, show simulation results for our proposed scheme, and then compare the results with those of other JSCC schemes which use different scheduling schemes in Sec. 6. In Sec. 7, we conclude with future research issues.

2 Reviews on JSCC for Packet Network Architecture

An architecture has been made for the future GII as shown in Fig. 1. The application layer draws on the service layer, which in turn calls upon the bitway layer. The service layer provides a set of common generic capabilities that are available to all applications. One of the functionalities in the service layer is the conditioning of data for the bitway, which means that a source coding scheme can be included in the service layer to compensate for impairment in the bitway by some error resilient schemes for multimedia data.[4]

As shown in Fig. 1, the application layer presents data (e.g., video frames) with application QoS attributes (e.g., resolution, frame rate, etc.) to the service layer. The service layer encodes the video frames, and presents the compressed video stream with service QoS attributes (e.g., average and peak rate) to the bitway, and the bitway communicates with other bitways by exchanging its QoS attributes (e.g., loss, corruption, and delay characteristics).

Packet-based communication networks inevitably introduce three types of impairment: packet loss (failure to arrive), packet corruption (bit errors occurring within the payload), and packet delay. Multimedia services can tolerate some level of loss and corruption without undue subjective impairment, especially if there is an appropriate masking built into the signal decoders.[4] There is some important information which should be transmitted in order to guarantee minimum quality, and it should be transmitted on a more reliable channel. Extending this idea, the substreams model can be developed.[4–6] In the substreams model, the stream of packets is logically divided into substreams which have their own QoS attributes, and each packet is identified as to its substream, which implicitly specifies the QoS objective for that packet. Fortunately, the substream model is consistent with the most important existing protocols. Substreams have been proposed in ST-II, the second generation Internet Stream Protocol.[7] Internet Protocol version 6 includes the concept of a flow, which is similar to a substream, by including a flow label in the packet header.[8,9] ATM networks incorporate virtual circuits (VC), and associate QoS classifications with those VC's, where an application can use multiple VCs.[10]

To support the substream model, the source coder in the service layer should classify source video frames into packets with different QoS objectives, which are based on the HVS knowledge. It should also associate those packets with the appropriate substream. This means that the source coder monitors the traffic it has generated for each substream, and the model can be a form of loosely coupled JSCC.[4–6]

3 Human Visual System and Frameless Rendering

The eye is a globe-shaped object with a lens in the front that focuses objects onto the retina in the back of the eye as shown in Fig. 2(a). The retina contains two kinds of receptors, called rods and cones. The rods are more sensitive to light than cones, and in low light, most of our vision is due to the operation of rods (for example, for detecting motion). We can model the eye as a receptor whose output goes to a logarithmic nonlinearity for spatial and temporal dimensions as shown in Fig. 2(b).[11] We also know that the eye acts as a spatial low-pass filter. From Fig. 2(b), we can elicit the following sensitivity chart (Table 1) for the HVS.[5] In Table 1, we can see that the HVS's sensitivity decreases as temporal frequency gets very high, so that coarse resolution for the scene is enough. We used motion estimation

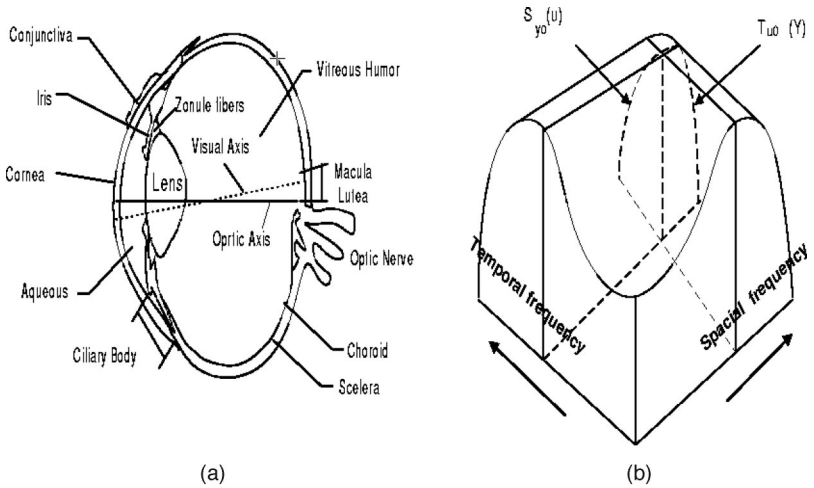


Fig. 2 (a) Cross section of the eye and (b) spatial/temporal sensitivity model.

technique to classify temporal frequency into high, and low, very low motion as will be discussed in Sec. 4.

Visual information is accumulated over an interval of approximately 125 ms. In the retinal image, an integration of this information is blurred or smeared. The newest information will have the greatest photo receptor response. An appropriately weighted integral can model this integration over time. The eye cannot track motion in the following cases:[12,13]

- **Low Predictability of Motion:** If the HVS does not know what to expect in the case of complex (e.g., many objects are moving) or high speed motion, then it cannot easily track the motion.
- **Motion Acceleration:** Object acceleration correlates with an inability of the eye to track the object at the same rate as the motion. Changing velocity is difficult to predict for HVS.

To support the substream model, we used the frameless rendering scheme which reconstructs the frame asynchronously with the sender as it displays any block which arrives at the receiver without frame completion as shown in Fig. 3.[12,13] The existing synchronous frame-by-frame method, which is designed for circuit-switched and wired services, is not satisfactory for the packet-switch driven communication network. It can best adapt to application QoS by separating different QoS needs for each substream which consists of application traffic, but can conflict with eye tracking. Therefore, some spatial degradation is possible. Thus, the frameless rendering scheme can minimize perceptual delay, sacrificing spatial quality, and it has been proven that perceptual delay has a more critical effect on HVS.

Table 1 Spatial-temporal resolution vs required QoS guarantees.

Temporal spatial	High	Low	Very low
High	coarse resol. low delay	normal resol. medium delay	normal resol. high delay
Medium	medium resol. low delay	fine resol. medium delay	fine resol. high delay
Low	coarse resol. low delay	normal resol. medium delay	normal resol. high delay

4 Packetization and JSCC Substream Allocation

There are many ways to make substreams from video frames. We designed an MPEG transcoder to extract image blocks from the MPEG coded stream for packetization and used motion information for each image block to classify those blocks into (high, low, and very low) motion classes as shown in Fig. 4. We classified the motion by its motion magnitude relative to the maximum ± 12 motion vectors. We classified the motion within ± 5 as very low motion, the motion within ± 12 as low motion, and the motion outside ± 12 as high motion.

After extracting the image blocks, each block is packetized with header information. The header gives information on the frame number, the location of the block for this frame, and the timing requirement (i.e., delay and jitter bound) for each discrete cosine transform (DCT) block. There is also one more parameter which specifies how many coefficients in the DCT block must be displayed. We model a block as a sum of (mandatory, level-1-optional, level-2-optional,..., level-3-optional) parts, which is similar to the imprecise computation model in Ref. 14. In this environment, we used reservation techniques for the high motion block to guarantee the application minimum delay, and scheduling techniques for low and very low motion block to enhance the maximum image quality.

When there are many objects with complex or high speed motion, the human eye cannot track the motion in the scene. For such cases, the retinal velocity will be zero (i.e., no perceived motion), and there will be no relative velocity for objects (i.e., the eyes do not follow any object's motion). Therefore, a static guideline such as ± 12 for high motion, should be redesigned to be dynamic to adapt motion content in the scene. But since no previous research work has been done to determine quantitatively the amount of motion there is in a scene, we define new terminology ''motion entropy,'' for this purpose as follows:

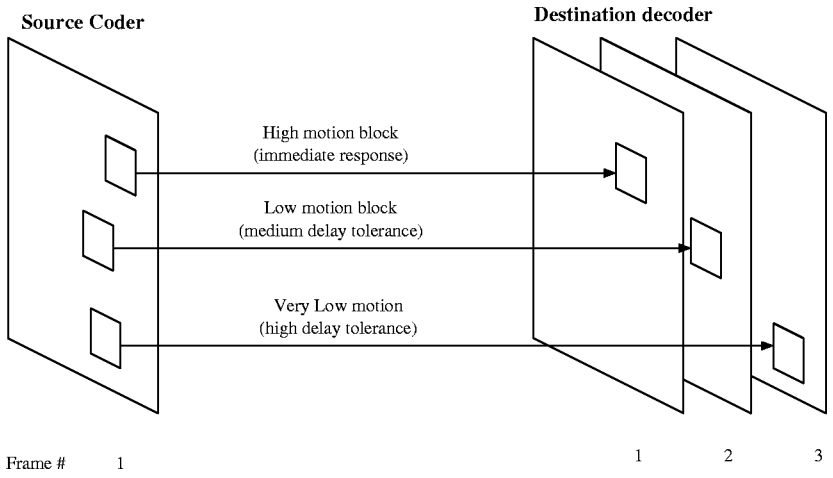


Fig. 3 Frameless rendering of substreams with different QoS.

$$M_b = p(C(b)) \cdot \log \frac{1}{p(C(b))}, \tag{1}$$

where M_b is the motion information of a set $C(b)$ which the block b belongs to, and there are 24 sets for C as shown in Fig. 5. Motion information for each set C is computed [i.e., $p(x)$ is the probability of collective motion x, which classified as a region as shown in Fig. 5] and summed up to make the motion entropy for a frame: $M = \Sigma M_b$.

We used motion vector information to compute the motion entropy for simplicity. We used the motion entropy to model motion change and to determine the required image quality for each scene; after that, the required quality for each substream will be decided. We used the motion entropy to change its classification values, and the motion classification value is changed in a state-by-state manner as shown in Fig. 6. In the figure, we see that high motion entropy (i.e., complex motion) causes transitions from other states to the high motion entropy state, and decreases the motion classification values $(t2,t1,t0)$ which means that most objects in the scene will be considered to be high motion with the new motion classification value. In other words, low motion entropy causes transitions from other very-low/high motion entropy states to the low motion entropy state decreasing/increasing motion classification values.

MPEG assumes synchronous coding, which determines the required coder/decoder prediction memory sizes. We designed asynchronous coding, which supports synchronous coding. The mechanism to use is the transcoder (i.e., JSCC coupler and JSCC decoder). The MPEG decoder will decode each frame at a predetermined frame processing time. The JSCC decoder will gather and schedule blocks for the frame, and gives them to the MPEG decoder before the predetermined processing time for the frame. The JSCC decoder can utilize the previous frame blocks when there are no available blocks. The required memory size for the JSCC decoder can be set as implementation specifics.

5 Substream Scheduling for JSCC

After substream assignment according to their motion contents, video packets arrive at the destination, and are scheduled to maximize the video quality. In this paper, we designed the (m,k,w) substream scheduling scheme extending real-time scheduling algorithms.[14–18]

• *EDF Scheduling*. At any arrival of a new task, EDF immediately computes a new order, that is, it preempts the running task and schedules the new task according to its deadline. Processing of the interrupted task continues later. EDF handles not only periodic tasks, but also tasks with arbitrary requests, deadlines, and service execution times. However, in the arbitrary case of an overload situation, EDF cannot guarantee the processing of any task.[15,16]

• (m,k) *scheduling*. Recently, there has been some research on maximizing uniformness in real-time scheduling of tasks, not just minimizing deadline failure. In (m,k) real-time scheduling. (m,k) specifies a desired QoS for the stream. A stream with (m,k)-firm deadlines experiences a

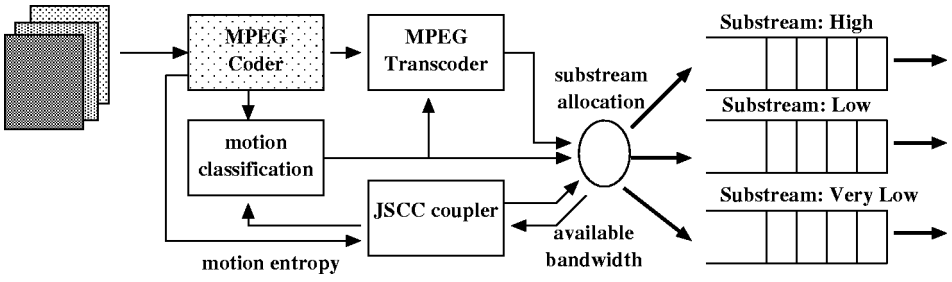


Fig. 4 MPEG compliant JSCC architecture using substream model.

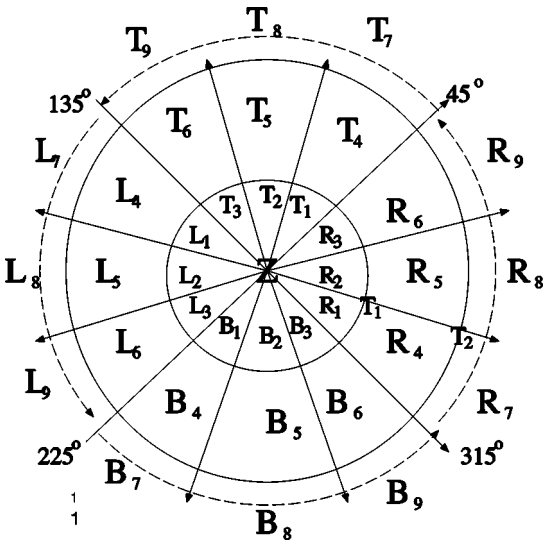


Fig. 5 Motion entropy model.

dynamic failure if fewer than m customers meet their deadlines in a window of k consecutive customers.[17,18] A (1,2)-firm specification corresponds to the constraint that no two consecutive customers from a stream should miss their deadlines. A stream gets closer to a failing state when its customer misses its deadline. The objective is to prevent streams from going to a failing state. The idea is as follows. The closer a stream is to a failing state, the higher the priority assigned to its next customer so as to increase its chances of meeting the deadline and thus move the stream away from a failing state.[18]

As shown in Fig. 4, the source coder uses motion estimation to classify the DCT block to allocate a different substream. We assigned (m,k,w) values for each substream, where (m,k) specifies bounded jitter requirements for each DCT block, and w means the number of DCT coefficients which should be displayed to guarantee the minimum perceptual quality. They are controlled by the JSCC coupler as shown in Fig. 4. The JSCC coupler will assign (m,k,w) values to each substream according to motion class and motion entropy. In the decoder part, the display order of substreams is determined by the (m,k,w) scheduler. The (m,k,w) scheduler chooses a substream among three (high, low, and very low motion) substreams, and gets a packet to display from the chosen substream's first packet. Similar to Ref. 18, the (m,k,w) scheduler has each substreams' deadline miss history, where a deadline miss can be checked by frame number in the video packets. The substream that has the highest probability of dynamic failure is chosen by the (m,k,w) scheduler. Figure 7 shows the state transition diagram for a (2,3,*) substream, where * means any w value. The letters M and m are used to represent a meet and a miss, respectively. States are denoted by three-letter strings. For example, MMm denotes the state where the most recent video packet missed its playout time and the two video packets before that one met their playout time. The edges represent the possible state transitions. Starting from a state, the stream makes a transition to one of two states, depending on whether its next video packet meets or misses its playout time.

Initially, we assigned $(m=9,k=10,w=1)$, $(m=5,k=10,w=6)$, and $(m=1,k=10,w=15)$ values to high, low, and very low motion, respectively. Consequently, the high motion substream gets a higher priority than the low and very low motion substreams. In effect, this scheduling will minimize the jitter that causes critical impairment in perceptual quality. The w parameter can be used to manage the congestion that occurs in the client and network. In case of congestion, we drop the optional part of the received packet to guarantee the mandatory part of other prioritized packets, and the deopped part can be serviced later when there are resources available to service it.

Choosing (m,k,w) parameters in (m,k,w) scheduling is very important because the improper choice can cause an unpleasant effect in image quality. For example, if we choose (m,k,w) parameters for high motion with an extraordinarily high value, image impairment can result as shown in Fig. 8. The blocks with high motion can overwhelm the whole image area in an extreme case.

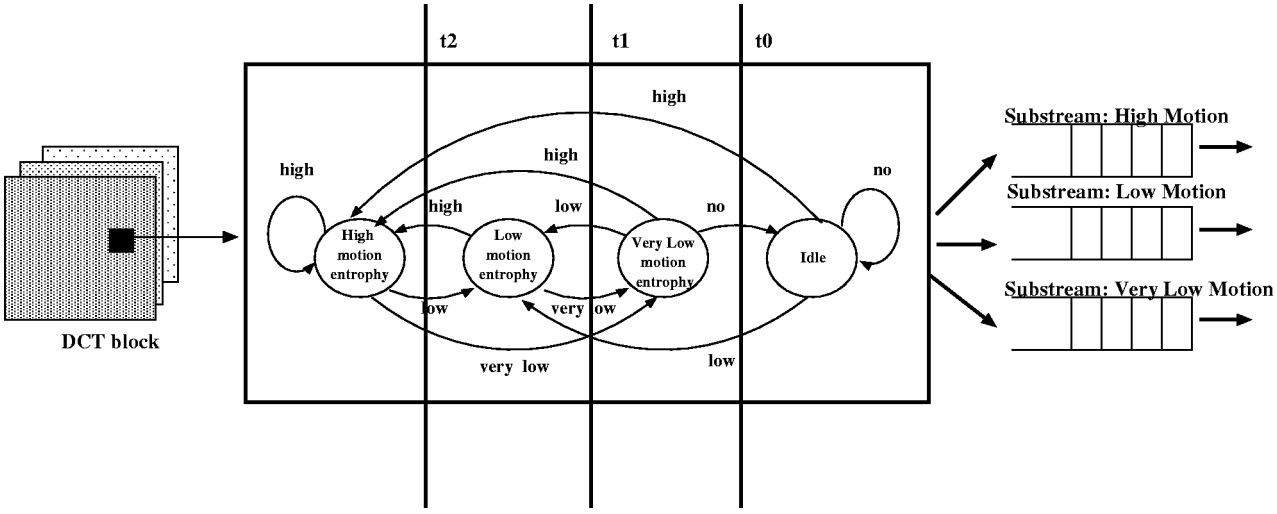


Fig. 6 Packet classification for substream assignment using motion entropy.

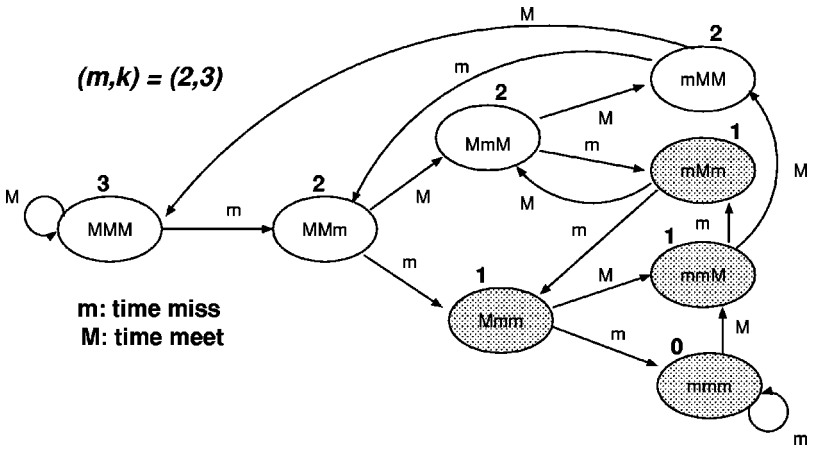


Fig. 7 (*m,k,w*) scheduling system model.

6 Performance Evaluation

In this section, we will show our simulation results for JSCC using the EDF, (*m,k*), and (*m,k,w*) scheduler for randomly generated video traffic, and also show simulation results for real (REAL) video test sequences. In REAL, the source code is provided, so we modified REAL to simulate our scheduling algorithms. We created the source and sink node functions, which are used to simulate JSCC, and also inserted the EDF, (*m,k*), and (*m,k,w*) scheduling algorithms into the scheduler node.

Figure 9 is our simulation model. The source node generates DCT block packets at periodic intervals. We assume the CIF (352×288) frame size to be 20 frame per second (fps) video traffic. The generated packets are categorized into three motion groups: high, low, and very low motion. We assume that the line speed between the source and transport nodes is infinitely fast (i.e., no coding delay). Transport nodes are used to simulate the transport layer in the packet network. The transport nodes of each motion group take data packets from the source node and send them to the scheduler node with different delayed times to simulate that high motion blocks are sent through a fast simulated line, low motion blocks through a medium rate simulated line, and very low motion blocks through a slow simulated line. Each packet is delivered from the transport nodes to the scheduler node with an end-to-end deadline and (*m,k,w*) value. As in Ref. 18, each substream is queued in a separate first in first out order, and the first packet in each queue is selected and transported by a specific scheduler. We implemented the EDF scheduler, (*m,k*) scheduler, and our (*m,k,w*) scheduler. Each scheduling policy leads packets to have the different queueing delays, and results in a difference in the number of display-failed packets in the sink node. The sink node gathers the simulator results. It checks each packet received from the scheduler node for bookkeeping. If a packet arrives later than its play-out time, it is considered a failed packet. The play-out time of each frame is set to 1/fps. The number of failed packets for a frame is gathered for each schedule policy. We varied the service rate of the scheduler node to observe scheduling efficiency in a congestion situation. We simulated the work load defined as source rate/service rate. This means light overloaded is 4,055,040/3,891,942 (1.05) and heavy overloaded is 4,055,040/2,413,714 (1.68).

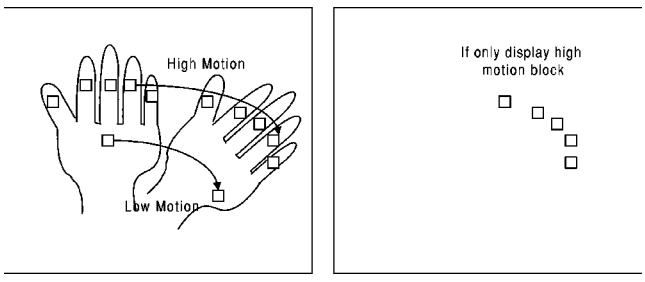


Fig. 8 Problem of incorrect (*m,k,w*) setting.

We define spatial quality (sq) as the weighted average of DCT coefficients that compose a specified DCT block. For this purpose, we used the inverse of the MPEG quantization table for our weighting factors to adapt the human visual characteristics of the DCT coefficients as shown in Fig. 10(a). The temporal quality (tq) is a function of packet display time. tq has a various value due to the motion class. For example, a high motion block sharply drops its value because high motion is less tolerable to packet delay, while low motion sustains some value for some duration because low motion is tolerable to packet delay. In Fig. 10(b), high motion (9,10,1) sharply drops the tq to zero after 2 frame delays, while low motion (5,10,6) sustains its value until 5 frames delay time. Both sq and tq are in the range [0 . . . 1], where 1.0 means maximum quality, and 0 means minimum quality.

Figure 11 shows the simulation result for sq over 200 frames of video sequences under an overload of 1.05 and 1.68, respectively. It shows that EDF is the worst one among three scheduling algorithms: it is saturated fastest to the zero quality value. It is known that EDF is optimal in a dynamical-controlled environment but functions nondeterministically under transient overload.[15] Actually, EDF has no priority ordering between high, low, and very low motion blocks; consequently, they interfere with each other under overload. From Fig. 11, we also see that the (*m,k*) scheduling shows severe fluctuation behavior. In (*m,k*) scheduling, the high motion block will tend to monopolize the bandwidth, and many other low motion blocks will

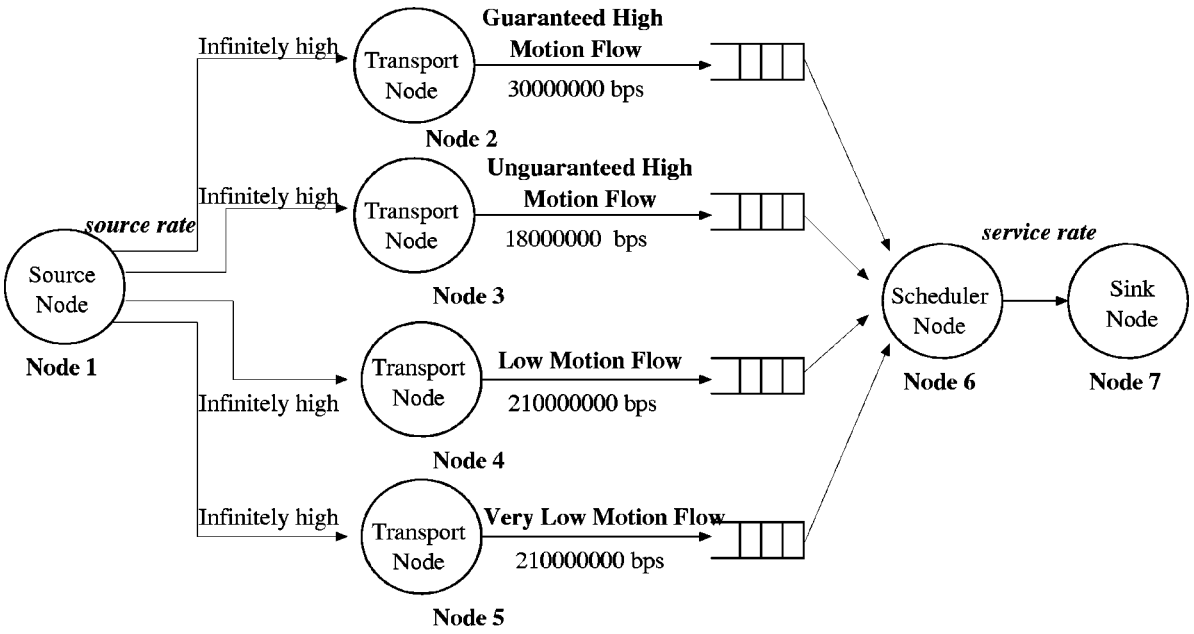


Fig. 9 Simulation model for JSCC scheduler.

starve sometimes, but the mandatory part of low motion has a heavy impact on the overall image quality. For this reason, (*m*,*k*) shows fluctuations, which leads to the real effect shown in Fig. 8. The (*m*,*k*,*w*) scheduling decreases this monopoly artifact by sharing some optional part of the high motion bandwidth with the mandatory part of low motion to increase the overall image quality during frame sequences. Figure 12 shows the simulation result for tq over 200 frames of video sequences under overload of 1.05 and 1.68, respectively. The figure shows that the three algorithms follow almost the same trace with that of the spatial quality.

To show real video experiments, we implemented the three algorithms over the TCP/IP environment. As a transport node which simulates congestion, a traffic simulator program which delays video traffic put inbetween the JSCC source and destination is used. It models delay/jitter which is usually shown in the TCP/IP environment. We tested our JSCC using the Susie test sequence with default MPEG coding parameters. In this paper, we designed a joint source channel coding scheme, which uses source coding information for the improvement of channel coding. Therefore, the bit rate and coding parameters are basically independent of our experimentatal results. Our experimental results can be

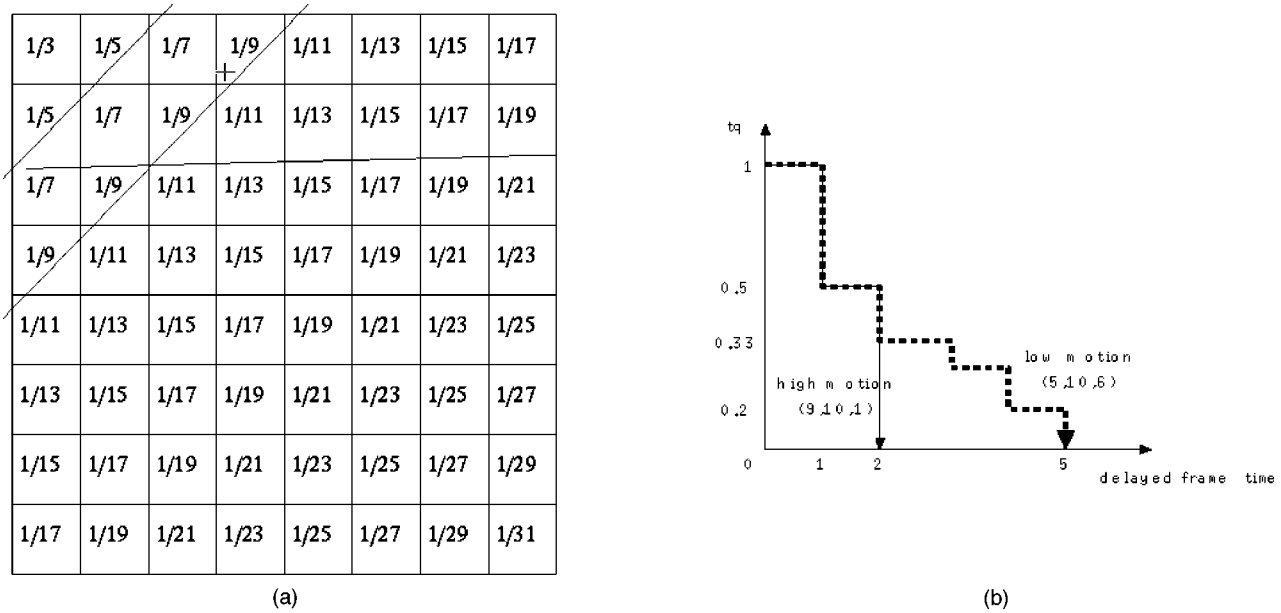
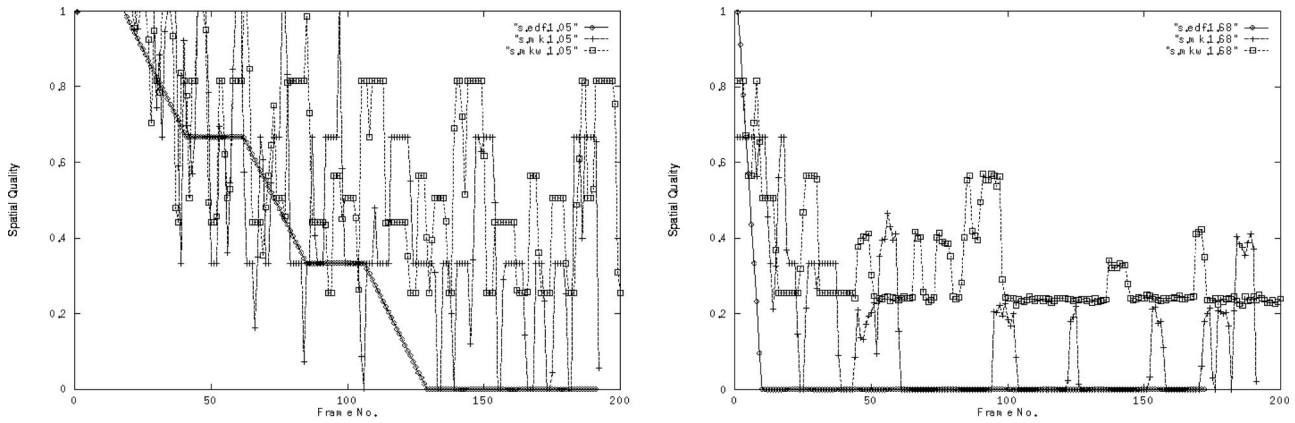


Fig. 10 Weighting factor for (a) sq and (b) tq.



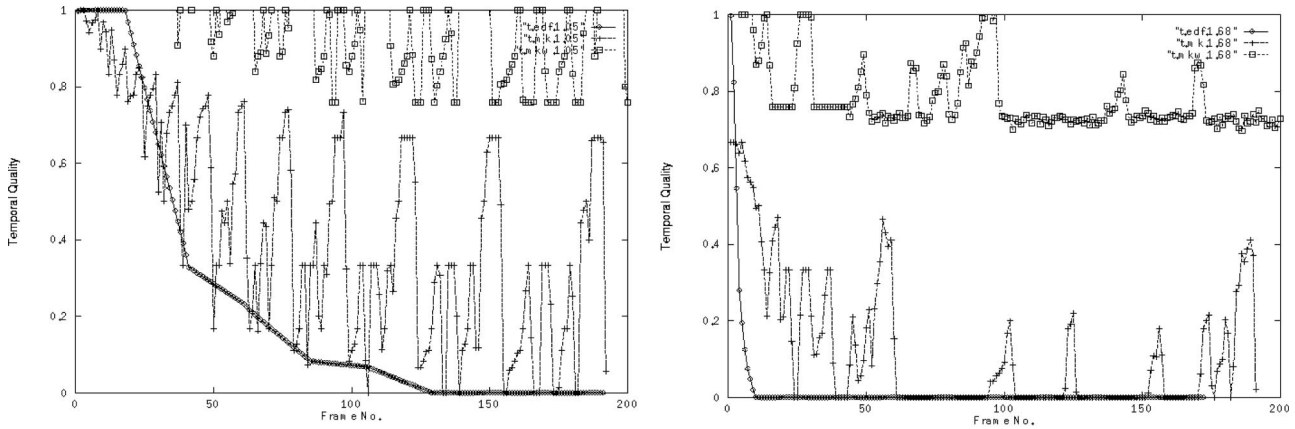
(a) (b)

Fig. 11 sq in (a) light overloaded (1.05) and (b) heavy overloaded (1.68) workload.

obtained in our REAL network simulation with a video sequence demonstration to interpret the physical meaning. It shows that (m,k,w) scheduling is better than (m,k) scheduling in perceptual video quality as shown in Fig. 13. As shown in Fig. 13(a), a scene can consist of low motion, very low motion, high motion, and very high motion block areas. For all three schedulings, very low motion block areas are kept well in image quality because they will keep their image values for a long time.

In EDF, high motion blocks will get the highest priority, and will prevent all the low motion blocks from arriving in the scene until all the high motion blocks arrive. For example, it will create regions like $L1$ and $L2$ where low motion blocks cannot be displayed as shown in Fig. 13(b). Some high motion blocks will also starve to display in case there are so many high motion blocks. In that case, it will result in the display of past image blocks with respect to the current image blocks as shown in H and VH as shown in Fig. 13(b), which shows an ugly face because some blocks are new and the other blocks are old. In (m,k), we can decrease the monopoly of high motion blocks which makes the starvation of low motion blocks by allocating allowable timing requirements instead of fixed priority for each high motion and low motion stream. It will result in better image quality as shown in $L1$, $L2$, $H1$, $H2$, and $H3$ of Fig. 13(c). But some high motion blocks, for example VH, still can be delayed by other high motion blocks in the case there are so many high motion blocks in a scene. Some low motion blocks can also cause the delay of high motion blocks, but low motion blocks usually make the whole image quality visually better.[19]

In (m,k,w), we can make VH as shown in Fig. 13(c) visually better if we allocate smaller mandatory display parts for high motion blocks. Almost all the high motion blocks will be displayed and it will not make an unsynchronized face as shown in Fig. 13(b), because all the delayed high motion blocks will switch to other recent high motion blocks very quickly as shown in VH, $H2$, and $H1$ of Fig. 13(d). However, it will show that there are some blocks which contain high frequency noise as shown in $H3$ of Fig. 13(d). It is almost invisible, and it can be postfiltered in the receiver side. Image quality in high motion blocks as shown in Fig. 13(d) can be enhanced by allocating more



(a) (b)

Fig. 12 tq in (a) light overloaded (1.05) and (b) heavy overloaded (1.68) workload.

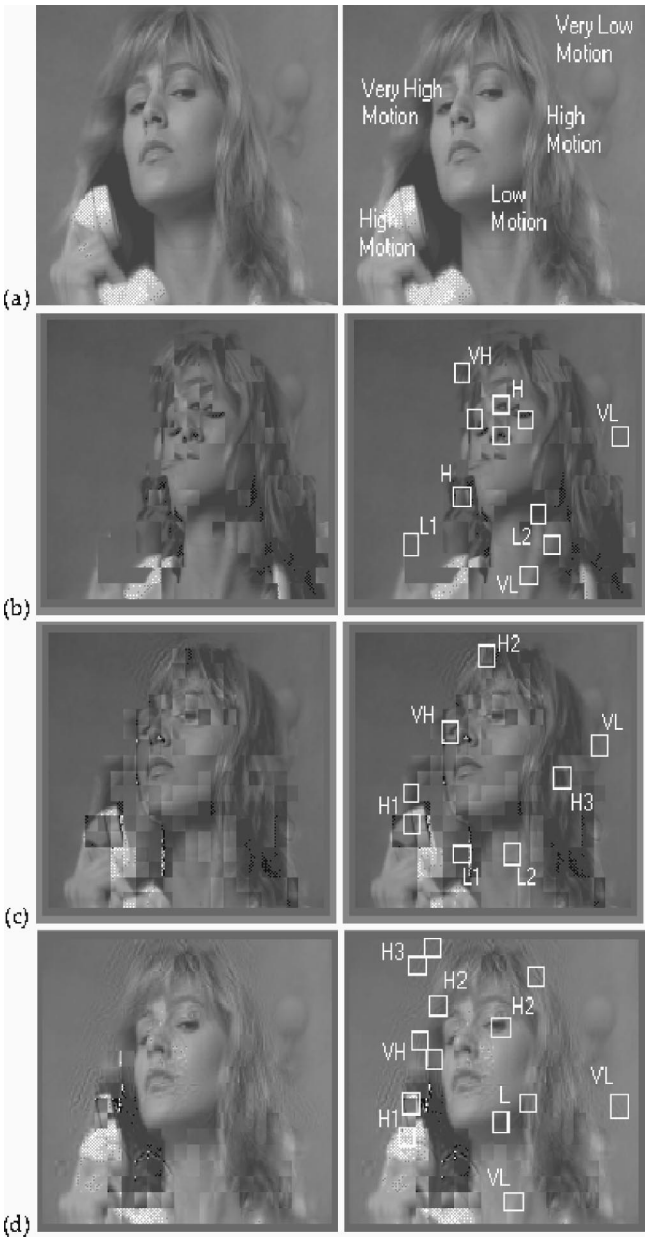


Fig. 13 JSCC for Susie test sequences: (a) original image; (b) EDF scheduling; (c) (m,k) scheduling; and (d) (m,k,w) scheduling.

DCT coefficients to the mandatory part of the high motion blocks, and can be improved by using a wavelet block if they are not compatible with the DCT based code such as the MPEG.[20]

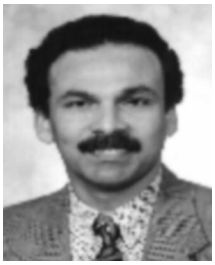
7 Conclusions

Real time multimedia communication requires QoS guaranteeing requirements such as bounded delay, bounded jitter, and bounded packet loss. To support these requirements, a loosely coupled JSCC, which is based on the substream concept, is designed. We have designed a simple MPEG transcoder scheme to extract packets from the existing MPEG coded stream for our JSCC front end. The JSCC then classifies each image packet into multiple substreams which have their own QoS requirements. To maximize perceptual video quality such as the bound of end-to-end jitter at the decoder side, we have designed a scheduling scheme for substreams. We have shown that our JSCC scheme is better than other schemes by simulation and real video experiments in the TCP/IP environment.

References

1. C. M. Aras, J. F. Kurose, D. Reeves, and H. Schulzrinne, ''Real-time communication in packet-switched networks,'' *Proc. IEEE* **82**, 122–138 (1994).
2. A. Campbell, G. Coulson, F. Garcia, and D. Huchison, ''Integrated quality of service for multimedia communications,'' *Proc. IEEE Infocom*, pp. 732–739 (1993).
3. G. M. Parulkar and J. S. Turner, ''Towards a framework for high-speed communication in a heterogeneous networking environment,'' *IEEE Netw.* **4**(2), 19–24 (1990).
4. P. Haskell, D. G. Messerschmitt, and L. C. Yun, ''Signal processing in wireless multimedia networks,'' in *Signal Processing for Wireless Communications*, G. Wornak, Ed., Prentice-Hall, Englewood Cliffs, NJ, 1997.
5. J. Reason, L. C. Yun, Al. Y. Lao, and D. G. Messerschmitt, *Asynchronous Video: Coordinated Video Coding and Transport for Heterogeneous Networks With Wireless Access, Mobile Wireless Information Systems*, Kluwer Academic, Dordrecht, 1995.
6. Y. C. Chang and D. G. Messerschmitt, ''Improving network video quality with delay cognizant video coding,'' *Proc. IEEE International Conf. Image Proc.* Vol. 3, pp. 27–31 (1998).
7. C. Topolcic, ''Experimental internet stream protocol, Version 2(ST-II),'' *Internet RFC 1190*, 1990.
8. C. Bradner and A. Mankin, ''The recommendation for the IP next generation protocol,'' *Internet Draft NRL*, 1994.
9. L. Zhang, S. Deering, D. Estrin, S. Shenker, and Daniel Zappala, ''RSVP: A new resource reservation protocol,'' *IEEE Netw.*, **7**(5), 8–18 (1993).
10. P. Pancha and M. E. Zarki, ''MPEG coding for variable bit rate video transmission,'' *IEEE Commun. Mag.* **32**, 54–66 (1994).
11. C. A. Burbeck and D. H. Kelly, ''Spatiotemporal characteristics of visual mechanisms: excitatory-inhibitory model,'' *J. Opt. Soc. Am.* **70**, 1121–1126 (1980).
12. E. Zagier, ''Motion blur and frameless rendering,'' *IEEE Crossroads*, 8–12 (1997).
13. G. Bishop, H. Funchs, L. McMillan, and E. J. S. Zagier, ''Frameless rendering: double buffering considered harmful,'' in *Proc. SIGGRAPH'94*, pp. 175–176 (1994).
14. J. W. S. Liu, K. Lin, C. L. Liu, and W. Shih, ''Imprecise computations: A means to provide scheduling flexibility and enhancement dependability,'' *Readings in Real-Time Systems*, Y. Leeabd and C. Krishna, Eds., pp. 81–97, IEEE, New York, 1993.
15. R. Steinmetz, ''Analyzing the multimedia operating system,'' *IEEE Multimedia* **2**(1), 1–10 (1995).
16. C. L. Liu and J. W. Layland, ''Scheduling algorithms for multiprogramming in a hard real-time environment,'' *J. ACM* **20**, 46–61 (1973).
17. C. Han, K. Lin, and C. Hou, ''Distance-constrained scheduling and its applications to real-time systems,'' *IEEE Trans. Comput.* **45**, 814–826 (1996).
18. M. Hamdaoui and P. Ramanathan, ''A dynamic priority assignment technique for streams with (m,k)-firm deadlines,'' *IEEE Trans. Comput.* **44**, 1443–1451 (1995).
19. Technical Report 88/472, REAL User Manual, Computer Science Department, University of Califonia at Berkely (1988).
20. S. W. Kim, S. Suthaharan, G. H. Lee, H. K. Lee, and K. R. Rao, ''Joint source/channel coding to maximize perceptual video quality on packer networks,'' *J. Real-Time Imaging* **5**, 329–339 (1999).

Seong-Whan Kim received the BS in computer science from the Dongguk University, Seoul, Korea in 1991, and the MS and PhD degrees from the Korea Advanced Institute of Science and Technology, Taejon, Korea, in 1993 and 1999, respectively. He was a visiting scholar in the Department of Electrical Engineering, University of Texas at Arlington, Texas, from January to March 1998. His research interests include wavelet transforms, digital watermarking, and joint source channel coding for mobile multimedia. He is a member of SPIE and IEEE.



Shan Suthaharan received his BSc (Honors) degree in statistics (specialization in computer science) from the University of Jaffna, Sri Lanka, in 1981, MSc degree in computer science from Dundee University, United Kingdom, in 1988, and earned his PhD degree in computer science from Monash University, Australia, in 1995. He is currently the head of the Department of Computer Science at Tennessee State University and research professor at the NASA/NSF funded Center of Automated Space Science. He has published over 80 research papers in reputed journals and international conferences. His research interests include image/video coding and compression, image restoration, multimedia computing and communication and computational intelligence. He is an associated editor of the *Journal of Real Time Imaging*, published by Academic Press. He is a senior member of IEEE.

Heung-Kyu Lee received the BE degree in electronics from the Seoul National University in 1978. He received the MS and PhD degrees in computer science in 1981 and 1984, respectively, from the Korea Advanced Institute of Science and Technology, Taejon, Korea. From 1985 to 1986 he was a research scientist at the University of Michigan. Since 1986, Dr. Lee has been a professor of computer science at the Korea Advanced Institute of Science and Technology, where he directed research in video compression and its implementation in DSP and taught courses in fault-tolerant systems and real time systems. His research interests include real-time processing, image processing, and digital watermarking. He is a member of IEICE and IEEE.

K. R. Rao received the PhD degree in electrical engineering from The University of New Mexico, Albuquerque, in 1966. Since 1966, he has been with the University of Texas at Arlington where he is currently a professor of electrical engineering. He, along with two other researchers, introduced the discrete cosine transform in 1975 which has since become very popular in digital signal processing. He is the co-author of the books *Orthogonal Transforms for Digital Signal Processing* (Springer-Verlag, 1975), *Fast Transforms: Analyses and Applications* (Academic Press, 1982), *Discrete Cosine Transform-Algorithms, Advantages, Applications* (Academic Press, 1990). He has edited a benchmark volume, *Discrete Transforms and Their Applications* (Van Nostrand Reinhold, 1985). He has coedited a benchmark volume, *Teleconferencing* (Van Nostrand Reinhold, 1985). He is co-author of the book, *Techniques and Standards for Image/Video/Audio Coding* (Prentice Hall, 1996). He has conducted workshops/tutorials on video/audio coding/standards worldwide. He has published extensively in refereed journals and has been a consultant to industry, research institutes and academia.