Overview of Fine Granularity Scalability in MPEG-4 Video Standard

Weiping Li, *Fellow, IEEE*

***Abstract*—Streaming Video Profile is the subject of an Amendment of MPEG-4, and is developed in response to the growing need on a video-coding standard for streaming video over the Internet. It provides the capability to distribute single-layered frame-based video over a wide range of bit rates with high coding efficiency. It also provides fine granularity scalability (FGS), and its combination with temporal scalability, to address a variety of challenging problems in delivering video over the Internet. This paper provides an overview of the FGS video coding technique in this Amendment of the MPEG-4.**

***Index Terms*—Bitplane coding, Internet, media, MPEG, scalability, standards, streaming, video.**

I. Introduction

THE OBJECTIVE of video coding has traditionally been to optimize video quality *at a given bit rate*. Due to the network video applications, such as Internet streaming video, the objective is somewhat changed. This change is necessary because network video has introduced a new system configuration, as illustrated in Fig. 1, and the network channel capacity varies over a wide range depending on the type of connections and the network traffic at any given time.

In a traditional communication system, the encoder compresses the input video signal into a bit rate that is less than, and close to, the channel capacity, and the decoder reconstructs the video signal using all the bits received from the channel. In such a model, two basic assumptions are made. The first is that the encoder knows the channel capacity. The second is that the decoder is able to decode all the bits received from the channel fast enough to reconstruct the video. These two basic assumptions are challenged in Internet streaming video applications. First of all, due to the video server used between the encoder and the channel, as shown in Fig. 1, plus the varying channel capacity, the encoder no longer knows the channel capacity and does not know at which bit rate the video quality should be optimized. Secondly, more and more applications use a software video client/decoder that has to share the computational resources with other operations on the user terminal. The video decoder may not be able to decode all the bits received from the channel fast enough for reconstruction of the video signal. Therefore, the objective of video coding for Internet streaming video is changed to optimizing the video quality *over a given bit rate range* instead of *at a given bit rate*. The bitstream should be partially decodable at any bit rate within the bit rate range to reconstruct a video signal with the optimized quality at that bit rate. Fig. 2 illustrates this point.

Manuscript received June 15, 2000; revised December 7, 2000. This paper was recommended by Guest Editors M. R. Civanlar, A. Luthra, S. Wenger, and W. Zhu.

The author is with WebCast Technologies, Inc., Mountain View, CA 94041 USA (e-mail: li@webcasttech.com).

Publisher Item Identifier S 1051-8215(01)02236-4.

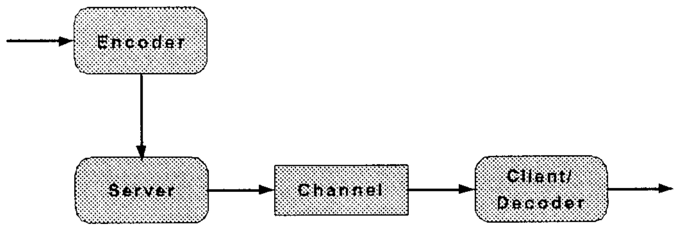


Fig. 1. System configuration for Internet Streaming Video.

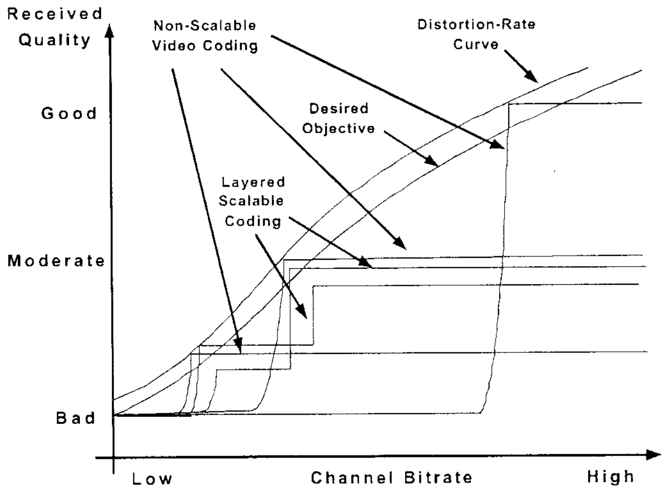


Fig. 2. Illustration of video coding performance.

The horizontal axis in Fig. 2 indicates the channel bit rate, while the vertical axis indicates the video quality received by a user. The distortion-rate curve indicates the upper bound in quality for any coding technique at any given bit rate. The three staircase curves indicate the performance of an optimal nonscalable coding technique. Once a given bit rate is chosen—either low, medium, or high—the nonscalable coding technique tries to achieve the optimal quality indicated by having the upper corner of the staircase curve very close to the distortion-rate curve. If the channel bit rate happens to be at the video-coding bit rate, the received video quality is the best. However, if the channel bit rate is lower than the video coding bit rate, a so-called “digital cutoff” phenomenon happens and the received video quality becomes very poor. On the other hand, if the channel bit rate is higher than the video-coding bit rate, the received video quality does not become any better. Scalable video coding has been an interesting topic. In MPEG-2 and MPEG-4, several layered

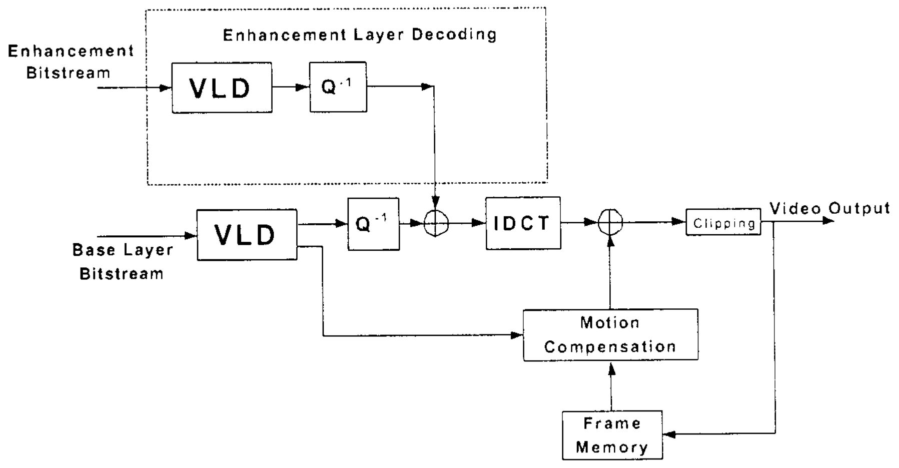


Fig. 3. SNR scalability decoder defined in MPEG-2.

scalability techniques, namely, SNR scalability, temporal scalability, and spatial scalability, have been included. In such a layered scalable coding technique, a video sequence is coded into a base layer and an enhancement layer. The enhancement layer bitstream is similar to the base layer bitstream in the sense that it has to be either completely received and decoded or it does not enhance the video quality at all. As shown in Fig. 2, such layered scalability techniques change the nonscalable single staircase curve to a curve with two stairs. The base-layer bit rate determines where the first stair is and the enhancement layer bit rate determines the second stair. The two curves shown in Fig. 2 for layered scalability have different characteristics. The first one has a poor performance for the base layer and a good performance for the enhanced video. The second is the opposite, namely, poor performance for the enhancement layer and good performance for the base layer. More detailed discussions on some specific layered scalable coding techniques are presented in the next section. As shown in Fig. 2, the desired objective of video coding for Internet streaming video is to achieve the continuous curve that parallels the distortion-rate curve with *a single bitstream*. This is the objective of the fine granularity scalability (FGS) video-coding technique in the Amendment of MPEG-4 [1].

This paper provides an overview on the FGS video coding technique defined in MPEG-4. To fully take advantage of the FGS video-coding technique for streaming video applications, many network-related issues have to be dealt with, which are not addressed in this paper. The next section briefly reviews several layered scalable coding techniques from which one can see the similarities and differences between the FGS video coding technique and the other layered scalable video techniques. In Section III, bit-plane coding of the DCT coefficients is described and compared with conventional run-level coding of the discrete cosine transform (DCT) coefficients, showing that bit-plane coding is more efficient than run-level coding. In Section IV, the basic FGS coding tools based on bit-plane coding are described. In Section V, the performance of the FGS coding technique is compared with multilevel SNR scalability, nonscalable coding, and simulcast in terms of coding efficiency. In Section VI, several advanced features of FGS coding are described. Finally, the paper is completed with some conclusions in Section VII and an Acknowledgment in Section VIII.

II. Brief Review of Layered Scalable Coding Techniques

Before presenting the FGS technique in MPEG-4, other scalable video coding techniques (SNR, temporal, spatial) are briefly reviewed in this section. More detailed descriptions can be found in [2]–[22]. Signal-to-noise ratio (SNR) scalability is a technique to code a video sequence into two layers at the same frame rate and the same spatial resolution, but different quantization accuracy. Fig. 3 shows the SNR scalability decoder defined in MPEG-2 video-coding standard [2]–[4]. The base-layer bitstream is decoded by the base layer variable-length decoder (VLD) first. The inverse quantizer in the base layer produces the reconstructed DCT coefficients. The enhancement bitstream is decoded by the VLD in the enhancement layer and the enhancement residues of the DCT coefficients are produced by the inverse quantizer in the enhancement layer. A higher accuracy DCT coefficient is obtained by adding the base-layer reconstructed DCT coefficient and the enhancement-layer DCT residue. The DCT coefficients with a higher accuracy are given to the inverse DCT (IDCT) unit to produce reconstructed image domain residues that are to be added to the motion-compensated block from the previous frame.

In this SNR scalability decoder, the enhancement-layer information is used in the motion-prediction loop. Therefore, there are the following four possible results depending on the corresponding SNR scalability encoder and whether the enhancement layer information is received by the decoder or not.

1) If the encoder *uses* the enhancement-layer information in the motion-prediction loop and the enhancement-layer information *is received* by the decoder, the enhancement-layer coding efficiency is high.
2) If the encoder *uses* the enhancement-layer information in the motion-prediction loop and the enhancement-layer

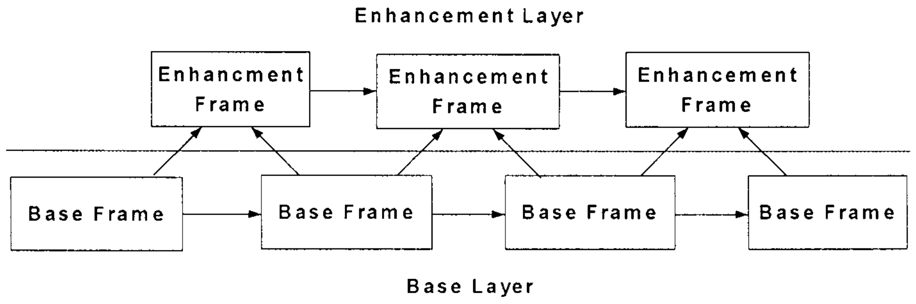


Fig. 4. A temporal scalability structure.

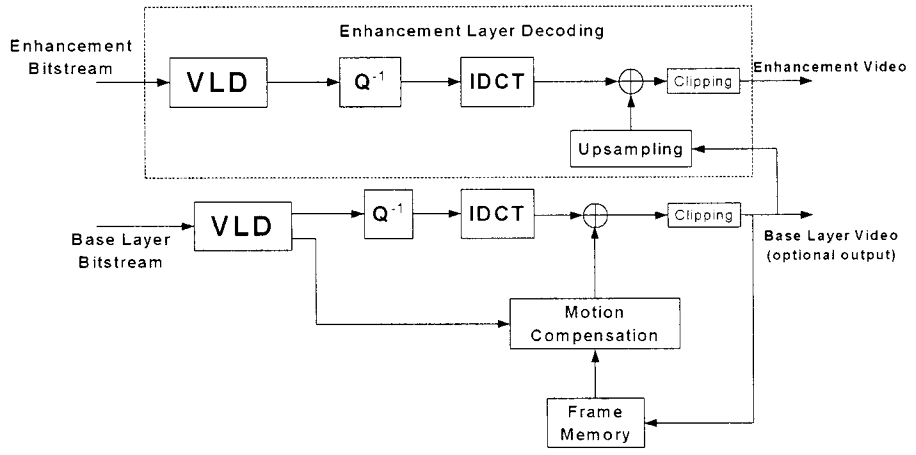


Fig. 5. A single-loop spatial scalability decoder.

information *is not received* by the decoder, drift happens in the base layer and coding efficiency is low.

3) If the encoder *does not use* the enhancement-layer information in the motion-prediction loop and the enhancement-layer information *is received* by the decoder, drift happens in the enhancement layer and coding efficiency is low.
4) If the encoder *does not use* the enhancement-layer information in the motion-prediction loop and the enhancement-layer information *is not received* by the decoder, the result is the same as using the base layer only.

Therefore, the choice is between the two curves illustrated in Fig. 2 for the layered scalable coding techniques, namely, either the base layer has a poor performance to ensure a good performance for the enhancement layer, or the enhancement layer has a poor performance to ensure a good performance for the base layer.

Temporal scalability is a technique to code a video sequence into two layers at the same spatial resolution, but different frame rates. The base layer is coded at a lower frame rate. The enhancement layer provides the missing frames to form a video with a higher frame rate. Coding efficiency of temporal scalability is high and very close to nonscalable coding. Fig. 4 shows a structure of temporal scalability. Only P-type prediction is used in the base layer. The enhancement-layer prediction can be either P-type or B-type from the base layer or P-type from the enhancement layer.

Spatial scalability is a technique to code a video sequence into two layers at the same frame rate, but different spatial resolutions. The base layer is coded at a lower spatial resolution. The reconstructed base-layer picture is up-sampled to form the prediction for the high-resolution picture in the enhancement layer. Fig. 5 shows a single-loop spatial scalability decoder. The advantage of single-loop spatial scalability is its simplicity. If the spatial resolution of the base layer is the same as that of the enhancement layer, i.e., the up-sampling factor being 1, this spatial scalability decoder can be considered as an SNR scalability decoder too. Unlike the SNR scalability decoder in MPEG-2, the above spatial scalability decoder does not include the enhancement-layer information into the prediction loop. Therefore, if the corresponding encoder does not include the enhancement-layer information into the prediction loop either, the base-layer drift does not exist. Coding efficiency of the enhanced video using such an “open-loop” scalable coding method suffers from the fact that the enhancement information of the previous frame is not used in the prediction for the current frame.

The spatial scalability decoders defined in MPEG-2 and MPEG-4 use two prediction loops, one in the base layer and the other in the enhancement layer [2]–[6]. The MPEG-2 spatial scalable decoder uses as prediction a weighted combination of up-sampled reconstructed frame from the base layer and the previously reconstructed frame in the enhancement layer, while the MPEG-4 spatial scalable decoder allows a “bi-directional” prediction using up-sampled reconstructed frame from the base layer as the “backward reference” and the previously

TABLE I
CODING GAIN OF BITPLANE CODING
CCIR601 SEQUENCES

Sequence	Configuration	I-VOP	Frame-rate
CCIR601	I. B, B, P, B, B, P, ...	1 in every 15 frames	30fps
SIF	I, P, P, P, P, P, P, ...	First VOP	15fps
QCIF	I, P, P, P, P, P, P, ...	First VOP	10fps

reconstructed frame in the enhancement layer as the "forward reference".

A common characteristic of the layered scalable coding techniques is that the enhancement layer is either entirely transmitted/received/decoded or it does not provide any enhancement at all. This is why such a layered scalable coding technique has the performance curve of two stairs illustrated in Fig. 2. The major difference between FGS and the layered scalable coding techniques is that, although the FGS coding technique also codes a video sequence into two layers, the enhancement bitstream can be truncated into any number of bits within each frame to provide partial enhancement proportional to the number of bits decoded for each frame. Therefore, FGS provides the continuous scalability curve illustrated in Fig. 2.

III. BIT-PLANE CODING OF THE DCT COEFFICIENTS

In the conventional DCT coding, the quantized DCT coefficients are coded using run-level coding. The number of consecutive zeros before a nonzero DCT coefficient is called a "run" and the absolute value of the nonzero DCT coefficient is called a "level". If a so-called "2-D" VLC table is used, the (run, level) symbol is coded and a separate "EOB" symbol is used to signal the end of the DCT block. If a so-called "3-D" VLC table is used, the (run, level, eob) symbol is coded, where "eob" signals the end of the DCT block. The major difference between the bit-plane coding method and the run-level coding method is that the bit-plane coding method considers each quantized DCT coefficient as a binary number of several bits instead of a decimal integer of a certain value [23], [24]. For each 8×8 DCT block, the 64 absolute values are zigzag ordered into an array. A bit-plane of the block is defined as an array of 64 bits, taken one from each absolute value of the DCT coefficients at the same significant position. For each bit-plane of each block, (RUN, EOP) symbols are formed and variable-length coded to produce the output bitstream. Starting from the most significant bit-plane (MSB plane), 2-D symbols are formed of two components: 1) number of consecutive zeros before a 1 (RUN) and 2) whether there are any ones left on this bit-plane, i.e., end-of-plane (EOP). If a bit-plane contains all zeros, a special symbol ALL-ZERO is formed to represent it.

The following example illustrates the procedure. Assume that the absolute values and the sign bits after zigzag ordering are given as follows:

$$\begin{array}{ll} 10,0,6,0,0,3,0,2,2,0,0,2,0,0,1,0,\ldots,0,0 & \text{(absolute)} \\ 0,x,1,x,x,1,x,0,0,x,x,1,x,x,0,x,\ldots,x,x & \text{(sign bits).} \end{array}$$

The maximum value in this block is found to be 10 and the number of bits to represent 10 in the binary format (1010) is 4. Therefore, the 4 bit-planes are considered in forming the (RUN, EOP) symbols. Writing every value in the binary format, the 4 bit-planes are formed as follows:

$$\begin{array}{ll} 1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,\ldots 0,0 & \text{(MSB)} \\ 0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,\ldots 0,0 & \text{(MSB-1)} \\ 1,0,1,0,0,1,0,1,1,0,0,1,0,0,0,0,\ldots 0,0 & \text{(MSB-2)} \\ 0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,\ldots 0,0 & \text{(MSB-3).} \end{array}$$

Converting the four bit-planes into (RUN, EOP) symbols, we have

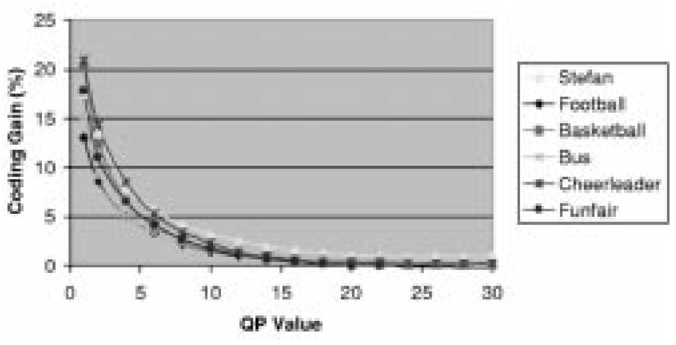
$$\begin{array}{ll} (0,1) & \text{(MSB)} \\ (2,1) & \text{(MSB-1)} \\ (0,0),(1,0),(2,0),(1,0),(0,0),(2,1) & \text{(MSB-2)} \\ (5,0),(8,1) & \text{(MSB-3).} \end{array}$$

Therefore, 10 (RUN, EOP) symbols are formed in this example. These symbols are coded using variable-length code together with the sign bits, as shown at the bottom of the page. Each sign bit is put into the bitstream only once right after the VLC code that contains the MSB of the nonzero absolute value associated with the sign bit. For example, no sign bit follows the second VLC code of the MSB-2 plane because the sign bit has been coded after the VLC code in the MSB-1 plane.

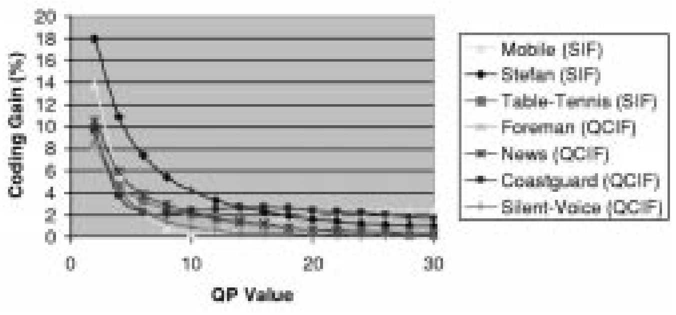
To evaluate coding efficiency of the bit-plane coding technique, we compare it with run-level coding without any scalability. In the experiment, bit-plane coding replaces run-level coding of the quantized DCT coefficients in a motion-compensated DCT coding structure and all the other operations in the coding structure are the same. The comparison is done on several CCIR601 interlaced sequences, as well as several QCIF and SIF sequences. Table I specifies the sequence configurations.

In the experiment, fixed QP values are used, and therefore the PSNR values and subjective quality are exactly the same for both bit-plane coding and run-level coding. The only difference is the number of bits used to code the DCT coefficients. Fig. 6 shows the relative coding gain of bit-plane coding over run-level coding in terms of the total number of bits used for coding each sequence. The experiment shows that bit-plane coding is always more efficient than run-level coding. The coding gain is larger when the QP value is smaller. Up to 20% of bit savings can be achieved by using bit-plane coding. The reason for bit-plane coding being more efficient is that the bit-plane statistics are independent of the QP value used for quantizing the DCT co-

$$\begin{array}{ll} \text{VLC}(0,1),0 & \text{(MSB)} \\ \text{VLC}(2,1),1 & \text{(MSB-1)} \\ \text{VLC}(0,0),\text{VLC}(1,0),\text{VLC}(2,0),1,\text{VLC}(1,0),0,\text{VLC}(0,0),0,\text{VLC}(2,1),1 & \text{(MSB-2)} \\ \text{VLC}(5,0),\text{VLC}(8,1),0 & \text{(MSB-3)} \end{array}$$



(a)



(b)

Fig. 6. Coding gain of bit-plane coding over run-level coding. (a) CCIR601 sequences. (b) SIF and QCIF sequences.

efficients. In run-level coding, a large QP value results in more symbols of long run and small level while a small QP value results in more symbols of short run and large level. Thus, for run-level coding, it is impossible for a single VLC table to be optimal for all QP values. Any VLC table for run-level coding has to be a compromise over the QP values. On the other hand, in bit-plane coding, the statistics are very close, regardless the QP value. For example, the MSB plane of an 8×8 DCT block with $\mathrm{QP} = 8$ is exactly the same as that with $\mathrm{QP} = 16$. Using $\mathrm{QP} = 8$ just results in one more bit plane than using $\mathrm{QP} = 16$. Therefore, it is possible to design the optimal VLC tables for each bit plane. Although the statistics of the bit planes with the same significance are very close, regardless the QP values, the statistics of the bit planes with different significance can be very different. Our studies have shown that the statistics of the first three bit planes (MSB, MSB-1, and MSB-2) are very different from each other and from the lower bit-planes. Therefore, four different VLC tables have been designed for MSB plane, MSB-1 plane, MSB-2 plane, and the other bit planes.

IV. FGS Using Bit-Plane Coding of DCT Coefficients

FGS has been identified in MPEG-4 as a desired functionality, especially for streaming video applications. It is well known that wavelet coding based on zero-tree arithmetic coding can also achieve the FGS functionality [25]–[29]. Initially, three types of techniques were proposed for FGS in MPEG-4, namely, bit-plane coding of the DCT coefficients [30], wavelet coding of image residue [31]–[33], and matching pursuit coding of image residue [34], [35]. After several core experiments [36], [37], bit-plane coding of the DCT coefficients was chosen due to its comparable coding efficiency and implementation simplicity. This section describes some details of using bit-plane coding to achieve FGS. In the first subsection, the overall FGS coding structure used in MPEG-4 is presented. In the second subsection, a few details of FGS coding are discussed. In the third subsection, profile definitions in the Amendment of MPEG-4 are briefly described.

A. *Overall Coding Structure of FGS*

The basic idea of FGS is to code a video sequence into a base layer and an enhancement layer. The base layer uses nonscalable coding to reach the lower bound of the bit-rate range. The enhancement layer is to code the difference between the original picture and the reconstructed picture using bit-plane coding of the DCT coefficients. Figs. 7 and 8 show the FGS encoder and the decoder structures, respectively.

The bitstream of the FGS enhancement layer may be truncated into any number of bits per picture after encoding is completed. The decoder should be able to reconstruct an enhancement video from the base layer and the truncated enhancement-layer bitstreams. The enhancement-layer video quality is proportional to the number of bits decoded by the decoder for each picture. The FGS decoder structure shown in Fig. 8 is the one standardized in the Amendment of MPEG-4. There are some possible variations to the standardized structure, which are discussed in the next subsection. The functionality of "bit-plane shift" shown in Figs. 7 and 8 is discussed in Section VI as advanced features.

B. *Some Details of FGS Coding*

The basic description of the bit-plane coding technique in Section III is for one 8×8 DCT block. Some other details of using bit-plane coding are discussed in this subsection.

1) Different Numbers of Bit-Planes for Individual Color Components: In Fig. 7, there is a functional block labeled "Find Maximum". This is to find the maximum number of bit-planes in a frame. As shown in Fig. 9, the three color components (Y, U, V) may have different numbers of bit-planes in a frame. Therefore, there are three syntax values **maximum_level_y**, **maximum_level_u**, and **maximum_level_v** coded in the frame header to indicate the maximum numbers of bit-planes for the Y, U, V components in the frame respectively. In the example shown in Fig. 9, coding the first bit-plane of the frame only involves the Y component since the U or V component has less bit-planes than the Y component.

2) Variable-Length Codes: The "Bitplane VLC" and the "Bitplane VLD" blocks in Figs. 7 and 8 perform encoding and decoding, respectively, of the (RUN, EOP) symbols as discussed in Section 3. Four VLC tables corresponding to the most significant bit (MSB) plane, the MSB-1 plane, the MSB-2 plane, and the other bit-planes are designed based on the statistics shown in Fig. 10.

One important point to be noted is that the MSB plane in the context of using the VLC tables is defined on a block basis. Not every 8×8 DCT block has the same number of bit-planes. The MSB plane of each individual block is the first bit-plane that is not an ALL-ZERO bit plane. This may vary from block to block in terms of its absolute significant position in a frame. The usage of the VLC tables is ac-

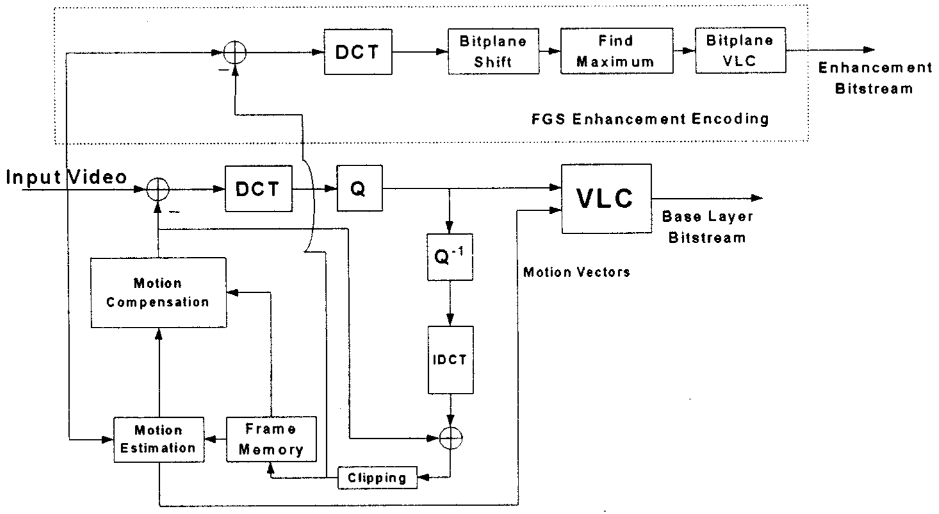


Fig. 7. FGS encoder structure.

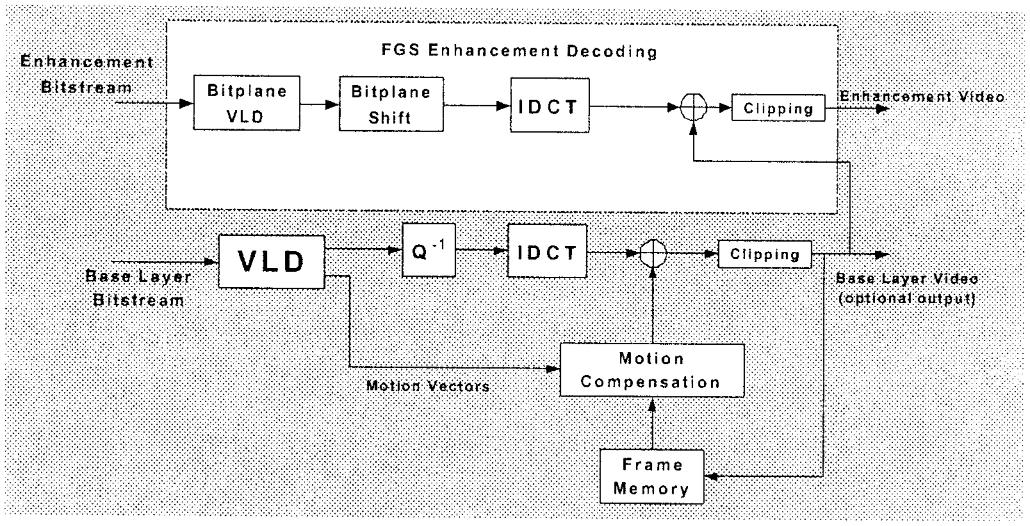


Fig. 8. FGS decoder structure.

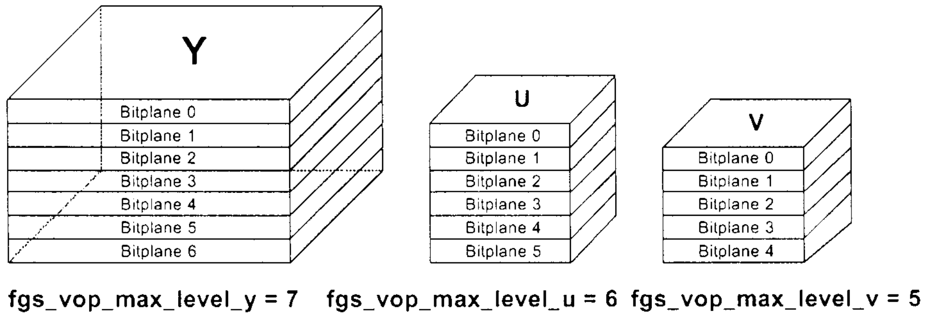


Fig. 9. An example of different numbers of bit planes for different color components.

cording to the individual block. Therefore, by definition, the MSB plane does not have the ALL-ZERO case. Since there are 64 bits in each bit plane, the value of RUN may range from 0 to 62 for EOP =0 and from 0 to 63 for EOP =1. Note that there cannot be a case of 63 consecutive zeros with EOP $=$ 0 (this would mean that there are more nonzero bits following the bit one). Plus, the case of ALL-ZERO, the maximum size of each VLC table is 128 symbols. Statistics of the symbols have shown that the probability of a symbol is smaller if the value of RUN is larger. Because the probabilities of large RUN values are very small, an ESCAPE code is used in each VLC table to signal a symbol with a large RUN value. Following the ESCAPE code, 6 bits are used to code RUN and 1 bit for EOP. In each plot in Fig. 10,

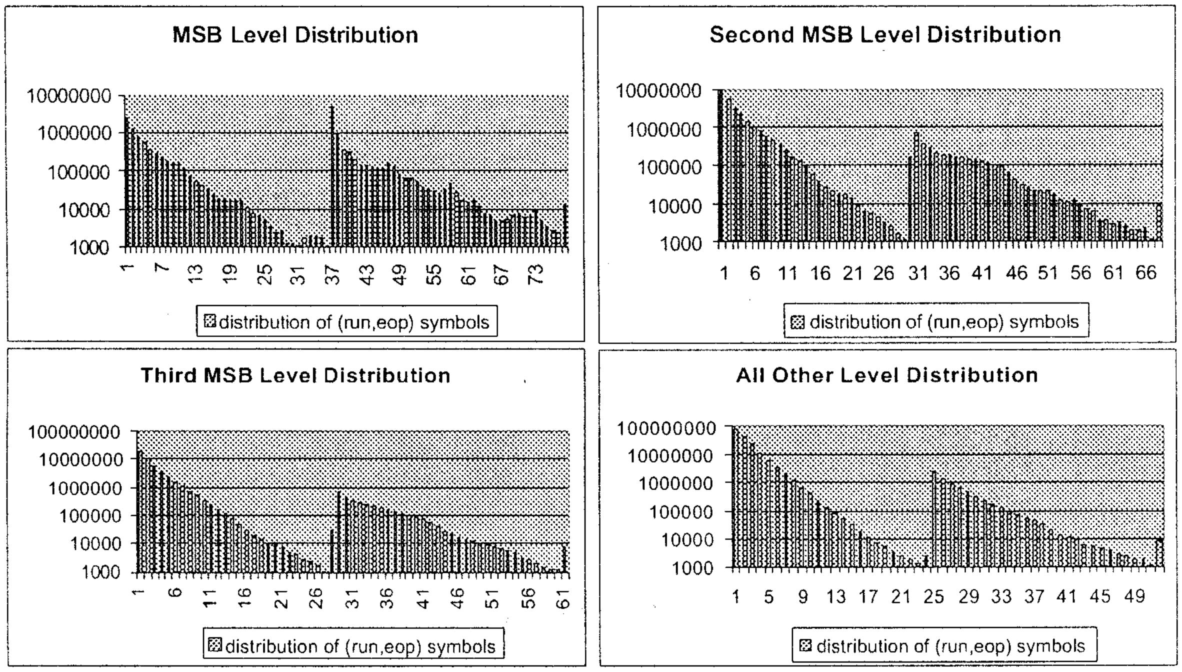
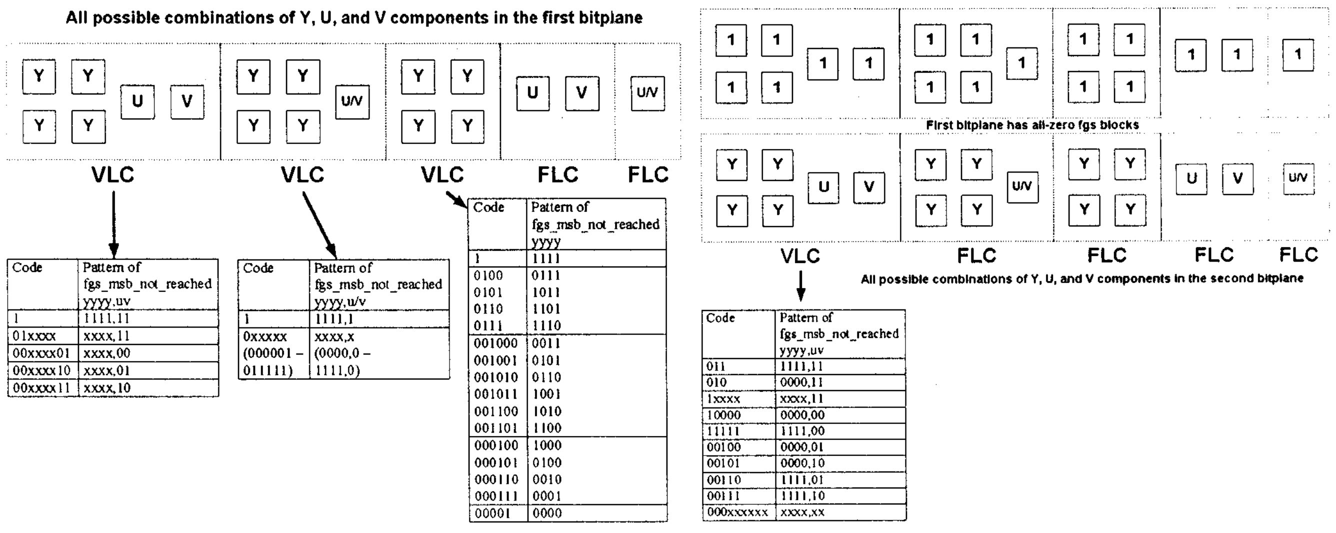


Fig. 10. Statistics of the (RUN, EOP) symbols in the four VLC tables.


All possible combinations of Y, U, and V components in the first bitplane
Y
U
V
U/V
VLC
FLC
1
First bitplane has all-zero fgs blocks
All possible combinations of Y, U, and V components in the second bitplane

Code	Pattern of fgs_msb_not_reached yyyy,uv
1	1111,11
01xxxx	xxxx,11
00xxxx01	xxxx,00
00xxxx10	xxxx,01
00xxxx11	xxxx,10

Code	Pattern of fgs_msb_not_reached yyyy,u/v
1	1111,1
0xxxxx (000001 – 011111)	xxxx,x (0000,0 – 1111,0)

Code	Pattern of fgs_msb_not_reached yyyy
1	1111
0100	0111
0101	1011
0110	1101
0111	1110
001000	0011
001001	0101
001010	0110
001011	1001
001100	1010
001101	1100
000100	1000
000101	0100
000110	0010
000111	0001
00001	0000

Code	Pattern of fgs_msb_not_reached yyyy,uv
011	1111,11
010	0000,11
1xxxx	xxxx,11
10000	0000,00
11111	1111,00
00100	0000,01
00101	0000,10
00110	1111,01
00111	1111,10
000xxxxxx	xxxx,xx

Fig. 11. Coding patterns for syntax element fgs_cbp.

the first half is for EOP $= 0$ and the second half is for EOP $= 1$. The ALL-ZERO symbol is in the middle and the ESCAPE symbol is at the end. The VLC table of the MSB plane does not contain the ALL-ZERO symbol. The ALL-ZERO case above the MSB plane of each block is signaled by a 1-bit syntax element **msb_not_reached**. Since it is very likely that many 8×8 DCT blocks have fewer bit planes than the maximum number of bit planes in a frame, there are many ALL-ZERO cases above the MSB plane of individual DCT blocks. In other words, the first couple of bit-planes of a frame are very likely to have many ALL-ZERO cases. Using 1 bit per block to signal the ALL-ZERO case is not efficient. A more efficient method is to use a macroblock (MB) syntax **fgs_cbp**. The idea is to group the blocks in each MB and to code the ALL-ZERO cases in the MB together.

Fig. 11 shows the block patterns and codes for the first and the second bit-planes of a frame. In the most likely case, in which all the blocks in an MB contain ALL-ZERO cases, a 1-bit code is used to signal it, instead of 6 (5 or 4) bits by coding **msb_not_reached** for each block. In the other extreme, for the first bit-plane of a frame, the very unlikely cases of having U or V block without Y blocks are simply coded using fixed-length

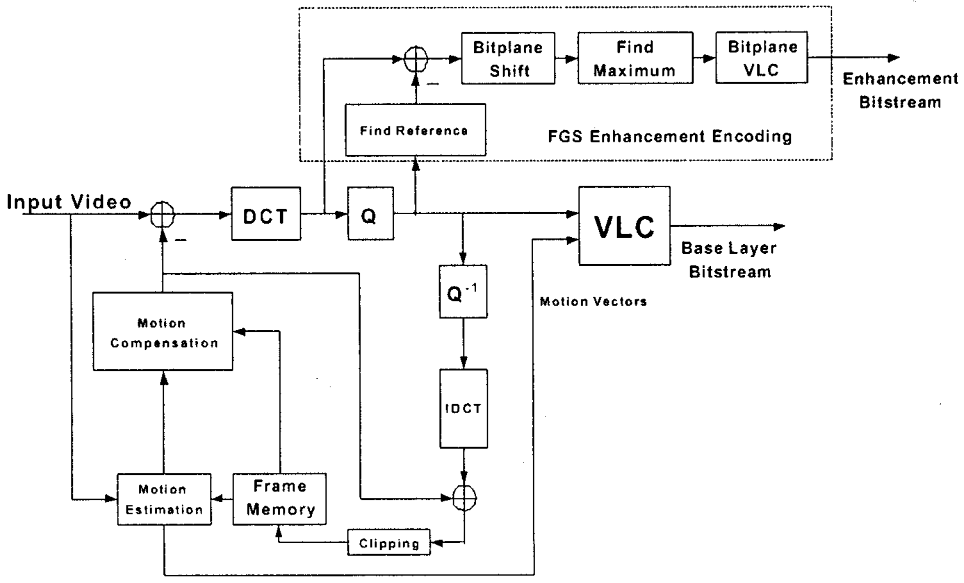


Fig. 12. A possible variation of the FGS encoder structure.

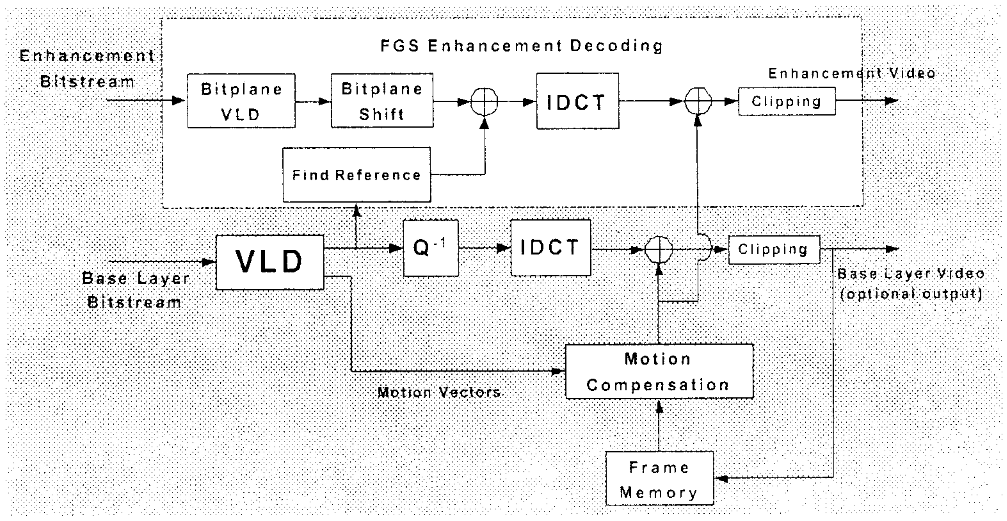


Fig. 13. A possible variation of the FGS decoder structure.

code (FLC) with 1 bit per block. For the second bit-plane of a frame, only one case is coded using VLC. That is, when the first bit-plane of the MB has all ALL-ZERO blocks and the second bit-plane of the MB has all 6 blocks. All other cases in the second bit-plane are coded using FLC.

3) Decoding Truncated Bitstreams: In a typical application of FGS, the bitstream at the input of an FGS decoder is a truncated version of the bitstream at the output of an FGS encoder. It is likely that, at the end of each FGS frame before the next FGS frame start code, only partial bits of the FGS frame are at the input of the decoder due to truncation of the FGS bitstream. Decoding of the truncated bitstream is not standardized in MPEG-4. One possible method for decoding a truncated bitstream of an FGS frame is to look ahead 32 bits at every byte-aligned position in the bitstream. If the 32 bits are the fgs_vop_start_code, the decoder either completes decoding up to the fgs_vop_start_code or discards the bits before the fgs_vop_start_code. If the 32 bits are not fgs_vop_start_code, the first 8 bits of the 32 bits are information bits of the FGS frame to be decoded. The decoder slides the bitstream pointer by one byte and looks ahead another 32 bits to check for fgs_vop_start_code.

4) Variations to the Standardized FGS Coding Structure: As mentioned in the previous subsection, there may be a few variations to the overall FGS coding structure defined in MPEG-4. One of the variations is shown in Figs. 12 and 13 for encoder and decoder, respectively.

Comparing Figs. 12 and 13 with Figs. 7 and 8, one can notice two main differences. One difference is that the variation structure shown in Figs. 12 and 13 processes the residue signal in the DCT domain while the standardized structure requires processing the residue signal in the image domain. The second difference is that the variation structure has a functional block called "Find Reference" that generates the reference signal in the DCT domain to be subtracted in the encoder and added in the decoder. The operation of taking residue is slightly different

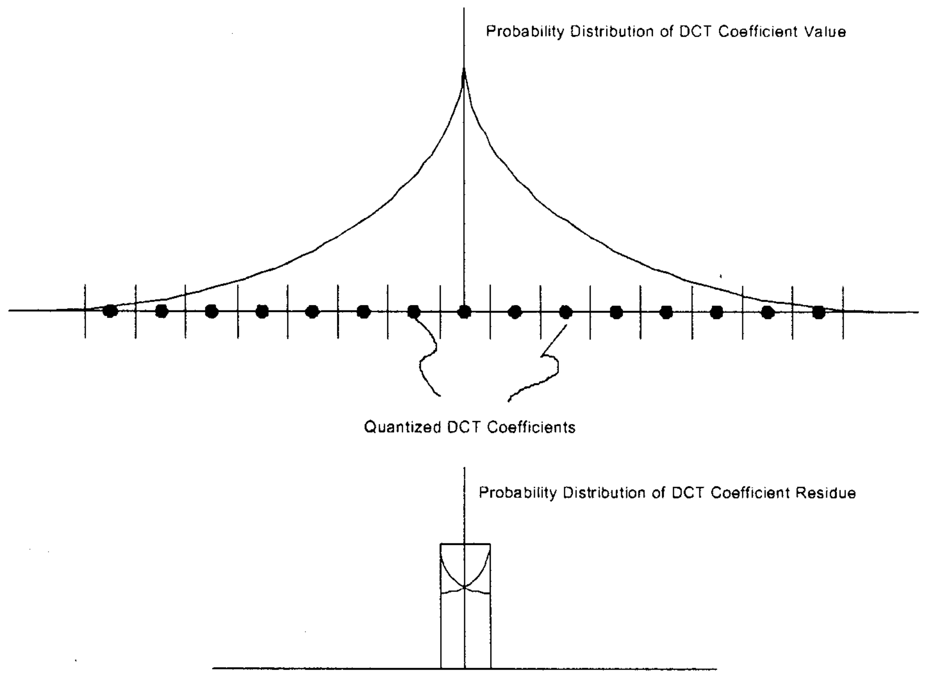

Probability Distribution of DCT Coefficient Value
Quantized DCT Coefficients
Probability Distribution of DCT Coefficient Residue

Fig. 14. Taking Residue between Original and Reconstructed DCT Coefficients.

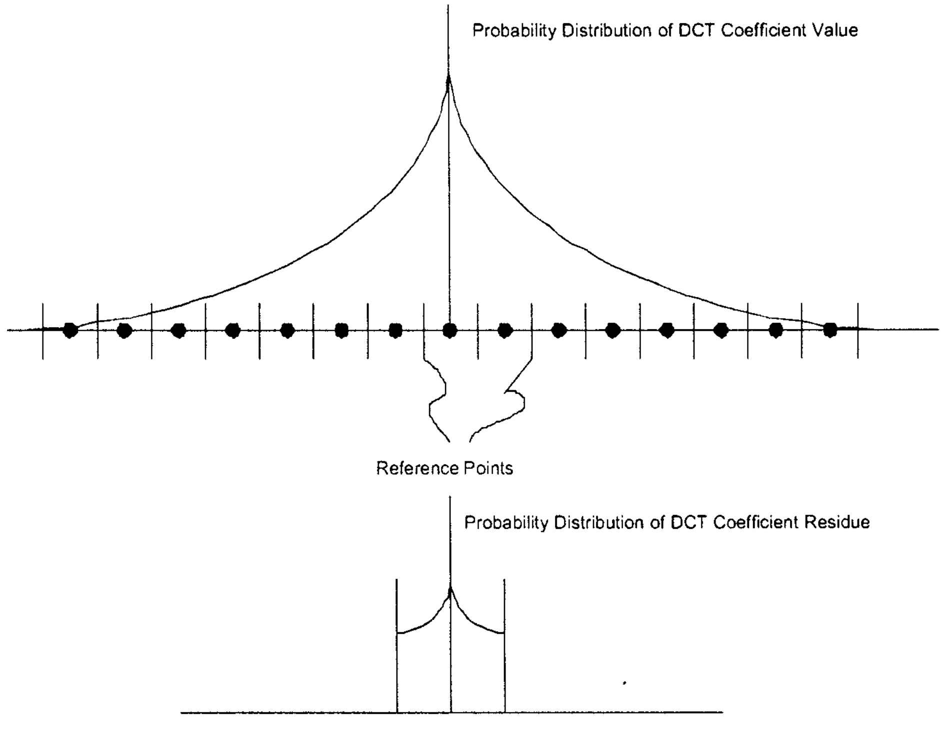

Probability Distribution of DCT Coefficient Value
Reference Points
Probability Distribution of DCT Coefficient Residue

Fig. 15. Taking residue between original and lower boundary point of quantization bin.

in the variation method. Usually, a residue is taken between the original DCT coefficient and the reconstructed DCT coefficient. Fig. 14 shows the distribution of such a residual signal.

The upper half of Fig. 14 illustrates the probability distribution of a DCT coefficient. Small values have higher probabilities and large values have smaller probabilities. The intervals along the horizontal axis are the quantization bins. The dot in the center of each interval is the reconstructed DCT coefficient. Taking residue between the original and the reconstructed DCT coefficient is equivalent to moving origin to the reconstruction point. Therefore, the probability distribution of the residue becomes that shown in the bottom half of Fig. 14. The residue from the positive side has a higher probability of being negative than positive, and the residue from the negative side has a higher probability of being positive than negative. The result is that the probability distribution of the residue becomes almost uniform.

In the variation method, the difference is taken between the original and the lower boundary point of the quantization bin, except the zero-bin, as shown in Fig. 15. In this method, the residue from the positive side remains positive and the residue from the negative side remains negative. Taking the residue is equivalent to moving the origin to the reference point. Therefore, the probability of the residue becomes the one shown in the lower half of Fig. 15, which preserves the shape of the original nonuniform distribution.

To compare coding efficiency of the two different ways of generating DCT residues, Fig. 16 shows the results of using the two methods to code five different video sequences.

It shows that using the boundary reference to generate the DCT residues is always more efficient than using the reconstructed DCT coefficient. The coding gain ranges from 0.25 to 0.5 dB. Comparing the implementation complexity of the variation method shown in Figs. 12 and 13 with that of Figs. 7 and 8, one can notice that the encoder shown in Fig. 7 requires one more DCT units than the encoder shown in Fig. 12, while the variation method requires a unit of "Find Reference" in both the encoder and the decoder. Therefore, from computational counts, it seems that the two methods have about the same level of complexity if the variation method is not more efficient. However, comparing the two decoder structures shown in Figs. 8 and 13, one can notice that the variation method requires a close coupling of the enhancement-layer decoding with the base-layer decoding while the decoder structure shown in Fig. 8 has the enhancement-layer decoding completely separated from the base-layer decoding until the final addition of the base-layer picture and the enhancement-layer residue. Such a separation of the enhancement-layer decoding from the base-layer decoding makes practical implementation easier and more flexible, especially the base-layer decoder design does not have to be touched if one has a nonscalable decoder design already. Therefore, the decoder structure shown in Fig. 8 was chosen to be standardized in the Amendment of MPEG-4.

An interesting point to be made is that, because the IDCT is a linear operation, the residue calculation in the image domain is equivalent to that in the DCT domain if the reconstructed DCT coefficients are used as the reference. The only nonlinear operation is the clipping unit that sets any number less than zero to zero and any number larger than 255 to 255. Since the encoder structure is not standardized in MPEG-4, one may use the encoder structure shown in Fig. 17, where the residue calculation is performed in the DCT domain and the DCT unit in the enhancement layer can be saved. This is equivalent to calculating the image domain residue before the clipping unit. Because the standardized decoder structure adds the base-layer picture back after the clipping unit, there is a mismatch between the encoder and the decoder. Since the enhancement-layer picture is not used in the prediction loop, the mismatch only affects the individual pictures without error propagation from one picture to the next. To understand how much error the mismatch causes, an experiment was performed. The results are shown in Fig. 18 [38], [39].

As shown in Fig. 18, the most mismatch errors are with a magnitude of 1. Large errors occur very infrequently. Visually, the errors do not make a noticeable difference. Therefore, one may

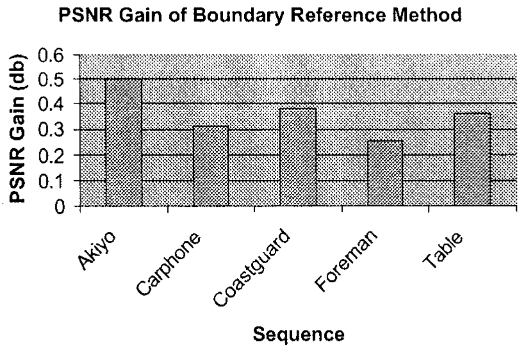


Fig. 16. Comparison of two different ways of generating the DCT residues.

make a design tradeoff by eliminating the DCT unit in the enhancement layer of the encoder and accepting the consequence of having mismatch errors when the bitstream is decoded by a standardized FGS decoder.

C. *Profile Definitions in the Amendment of MPEG-4*

MPEG-4 is a multimedia coding standard that contains many powerful coding tools. For any given application, only a subset of the coding tools will be used. Therefore, in order to limit the implementation complexity, MPEG-4 defines profiles and levels. Each profile contains a subset of tools. Within each profile, there usually are several levels defined to further limit the complexity by having different values for different parameters of the coding tools. Looking at the available video-coding standards and the MPEG-4 profiles, a need for defining a separate profile for the base layer of FGS was identified in [43]. As a result, there are two profiles defined in the Amendment of MPEG-4. One is called Advanced Simple Profile, which contains a subset of nonscalable video-coding tools to achieve high coding efficiency at any given bit rate within a wide range of bit rates. The base-layer coding tools defined in the Advanced Simple Profile include both P-VOP (forward prediction only) and B-VOP (bi-directional prediction) for coding motion-compensated residues. It includes the option of using error resilience tools (resynchronization marker, reverse VLC, and data partitioning). It provides backward compatibility with baseline H.263 by including the short header option. It allows two types of quantization methods, one with a single quantizer for all DCT coefficients and the other with a quantization matrix for the DCT coefficients. It includes tools for efficient coding of interlaced video (field/frame motion compensation and field/frame DCT). Using the Advanced Simple Profile as the base layer, the FGS profile is defined to meet the requirement of optimizing video quality over a given bit rate range. The coding tools in the FGS Profile include the basic bit-plane coding technique and the advanced features to be discussed in Section VI.

V. Coding Efficiency Performance of FGS

To evaluate the coding efficiency of the FGS technique, extensive experiments have been performed to compare FGS with multilayer SNR scalability, nonscalable coding, and simulcast. The test conditions are set up to cover a wide range of sequences,

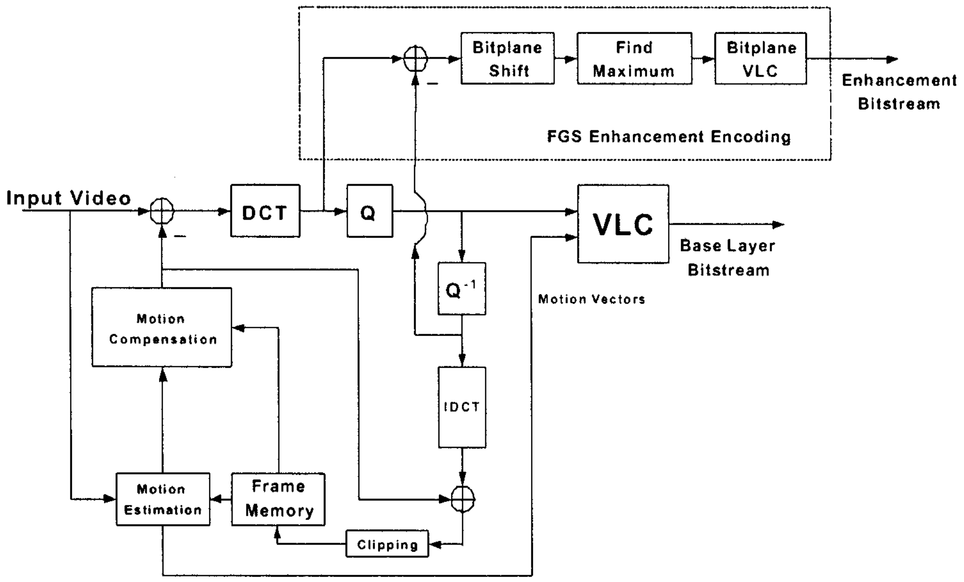


Fig. 17. Encoder structure without the DCT unit in the enhancement layer.

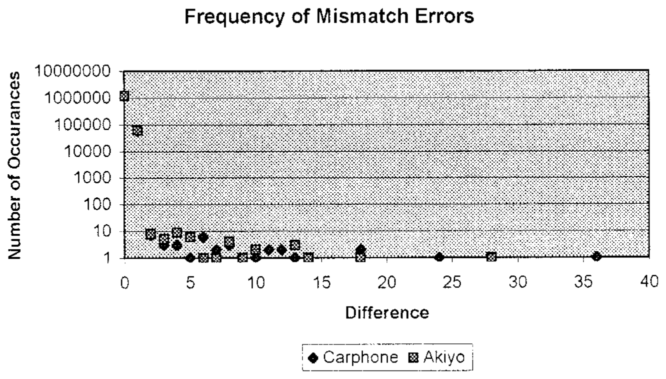


Fig. 18. Experiment results on mismatch errors.

bit rates, frame rates, and spatial resolutions. The number of bits for each FGS frame is truncated according to the channel bit rate and frame rate as follows:

$$\text{bits/frame} = \text{bit rate/frame rate}.$$

A. FGS versus Multilayer SNR Scalability

An alternative approach, which is close to the open-loop enhancement structure of FGS, is to use the spatial scalability as SNR scalability multiple times to create a multilayer SNR scalable bitstream. As shown in Fig. 5 of Section II, if the up-sampling factor is chosen to be 1, the spatial scalability becomes open-loop SNR scalability. This SNR scalability technique takes the difference between the original picture and the reconstructed picture, performs the DCT, quantizes the DCT coefficients, and codes the quantized DCT coefficients using run-level coding. The multilayer SNR scalability performs this procedure multiple times with a progressively better quality of reconstructed picture. Although this approach does not provide FGS, it is a possible candidate for comparison with FGS.

Since bit-plane coding of the DCT coefficients has better coding efficiency than run-level coding as shown in Section III, it is expected that FGS has better coding efficiency than multilayer SNR scalability. Extensive experiments have confirmed this point [41]. Fig. 19 shows the results. On each plot, the X axis is the total bit rate, and the Y axis is average PSNR of reconstructed sequences. As the bit rate increases, the FGS curve rises faster than the multilayer SNR scalability curve. FGS is about 2-dB better at the high bit rates.

B. FGS versus Non-Scalable Coding

In order to measure the coding efficiency of FGS, it is compared with nonscalable coding that is the upper bound of any scalable coding techniques [41]. Fig. 20 shows the results. As the total bit rate increases, nonscalable curve increases faster than the FGS curve, and the difference reaches about 2 dB at the highest bit rate.

C. FGS versus Simulcast

Another approach is to generate multiple bitstreams of multiple bit rates for the same content at the encoding time. For multicast on the network, the multiple bitstreams are transmitted by the server, and the total bit rate out of the server is the sum of the multiple bit rates. Each individual receiver takes one of the bitstreams according to its connection bit rate and decodes it to reconstruct the video content. In case of channel bandwidth variation due to local network traffic, the receiver may switch to a different bitstream. For unicast on the network, the multiple bitstreams are stored on the server and one of them is transmitted to a receiver according to the channel connection of the receiver. In case of a channel bit-rate change, a feedback signal is sent to the server and the server switches to a different bitstream. This method of generating multiple bitstreams of multiple bit rates for the same content is called simulcast.

In simulcast, one needs to decide, under a fixed total bit rate, how many bitstreams at what bit rates should be generated. Another constraint is to make the quality change between different bitstreams not noticeable by the user. With the two constraints,

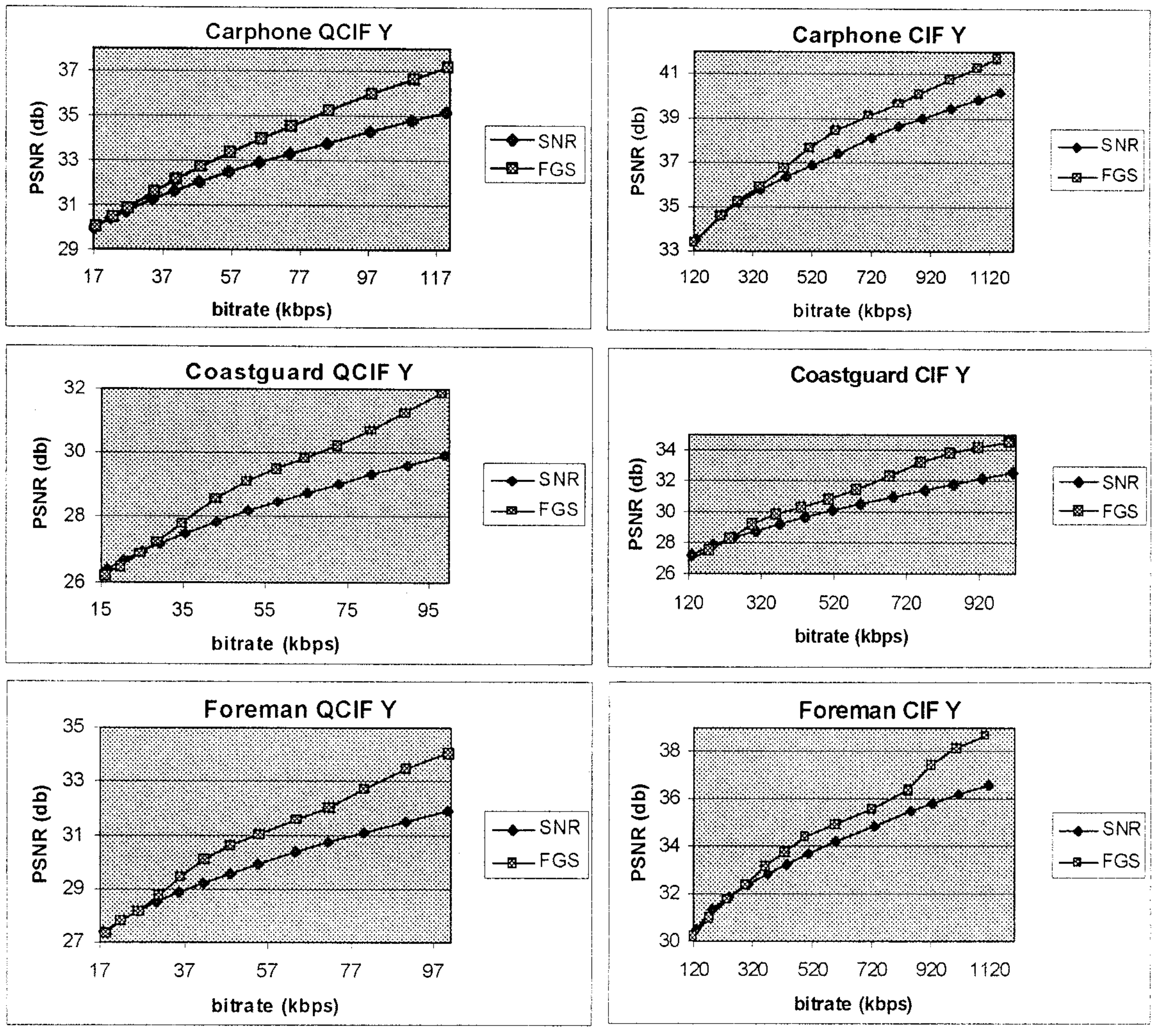


Fig. 19. Compare FGS with multilyer SNR scalability.

the choices for the number of bitstreams and the bit rates are very limited. If two bitstreams are generated, the most efficient choice of the two bit rates is one at lower bound and the other at (upper bound–lower bound). However, such a choice would not satisfy the constraint of having the quality change unnoticeable. To meet this constraint, one has to make two bit rates closer to each other. This naturally leads to the choice of three bitstreams. The highest and the second bit rates are related by a 75% ratio, and the lowest bit rate is the lower bound. This leads to the following set of relations between the three bit rates:

$$R_h + R_m + R_l = \text{total bit rate}$$
$$R_m = 0.75 R_h$$
$$R_l = \text{lower bound}.$$

Depending on the bit rate range, more bitstreams may or may not be feasible. For example, one of the test conditions in the core experiments in MPEG-4 is to have a bit-rate range from 128 to 512 kbits/s. With such a bit-rate range, it does not make any sense to have four or more simulcast bitstreams because {512 kbits/s}/4=128 kbits/s, i.e., all four bitstreams would have the same bit rate of 128 kbits/s. Fig. 21 shows the results comparing FGS with simulcast of both two and three bitstreams [42]. The horizontal axis is the user channel bit rate at which one can receive a video bitstream. The vertical axis is the average PSNR of the reconstructed video received at the decoder through the given channel bit rate. For example, if the user channel bit rate is lower than 384 kbits/s, in the case of simulcast of two bitstreams, the user can only receive the 128 kbits/s bitstream and therefore the reconstructed video quality stays the same as that of 128 kbits/s. The FGS curve continuously rises as the user channel bit rate increases. The simulcast curves are staircase as only two or three bit rates are allowed. Within each staircase, simulcast keeps the same quality. FGS is more efficient than simulcast of three bitstreams at the high end and more efficient than simulcast of two bitstreams at the low end.

VI. Advanced Features in FGS

To further improve visual quality of FGS enhancement video, two advanced features are included: frequency weighting and selective enhancement. Frequency weighting uses different weighting for different frequency components, so that more bits of the visually important frequency components are put in the bitstream ahead of that of other frequency components.

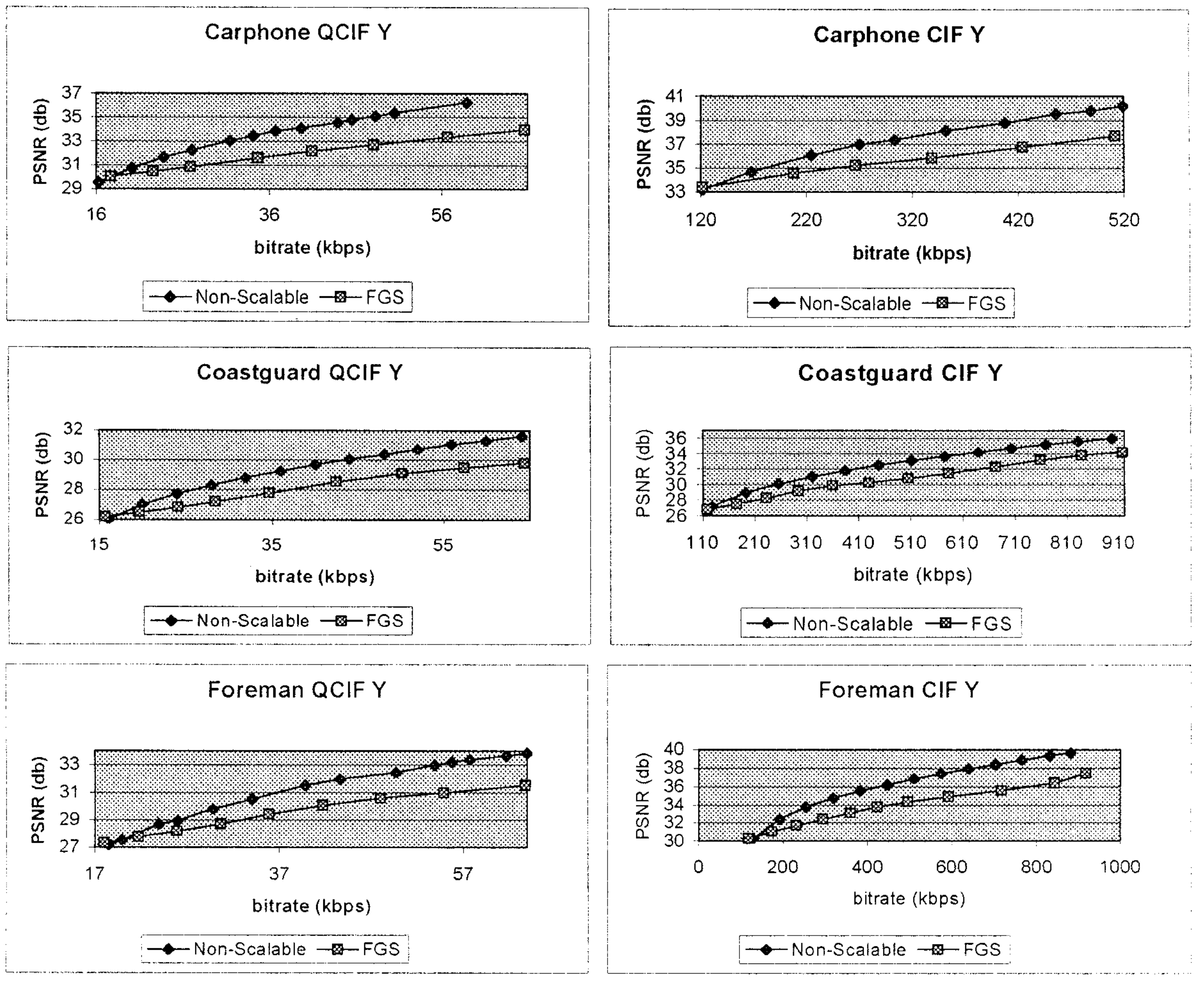


Fig. 20. Compare FGS with non-scalable coding.

Selective enhancement uses different weighting for different spatial locations of a frame so that more bit-planes of some parts of a frame are put in the bitstream ahead of that of other parts of the frame. Another advanced feature is to include resynchronization markers in the enhancement layer to make the FGS bitstream more error resilient for wireless applications where random burst errors may occur. Yet another advanced feature is a combination of FGS with temporal scalability to increase the bit-rate range a scalable bitstream covers.

A. Frequency Weighting

It is well known that different DCT coefficients affect the visual quality differently. Usually, the accuracy of the low frequency DCT coefficients is more important than that of the high frequency DCT coefficients. Therefore, it enhances visual quality better if the bits of the low-frequency DCT components are put into the enhancement bitstream earlier so that they are more likely to be included in a truncated bitstream. A frequency-weighting technique is included in FGS to achieve this objective [44], [45].

Since frequency weighting usually shifts the low-frequency DCT coefficients up, the bit-plane statistics may change. For example, there are more short RUN values with $\mathrm{EOP} = 1$ after frequency weighting. Therefore, two more sets of VLC tables are needed. The choice of VLC tables depends on the first value of the frequency weighting matrix (shifted bits for the DC component). If it is 0 or 1, the original VLC tables are used. If it is 2 or 3, the median-shift VLC tables are used. If it is 4 to 7, the high-shift VLC tables are used. A syntax element, **frequency_weighting_enable**, is used to indicate if frequency weighting is used or not. If frequency weighting is used, **load_frequency_weighting_matrix**, another syntax element, is used to indicate if a frequency weighting matrix is included in the bitstream or not. If it is included, **frequency_weighting_matrix**, a list of 2 to 64 three-bit unsigned integers is included in the bitstream. The integers are in zigzag order. A value of zero indicates that no more values follow and the remaining values are set to zero. Each value in this matrix is used to indicate the number of bits to be shifted up (down) for the corresponding DCT coefficient in encoding (decoding) as shown in Fig. 22.

B. Selective Enhancement

For some frames of a video sequence, a part of a frame may be visually more significant than other parts. Therefore, we may use a bit-plane shifting method to put the bit-planes of the MBs of interest earlier in the bitstream so that they are more likely to be included in a truncated bitstream [40]. A syntax element,

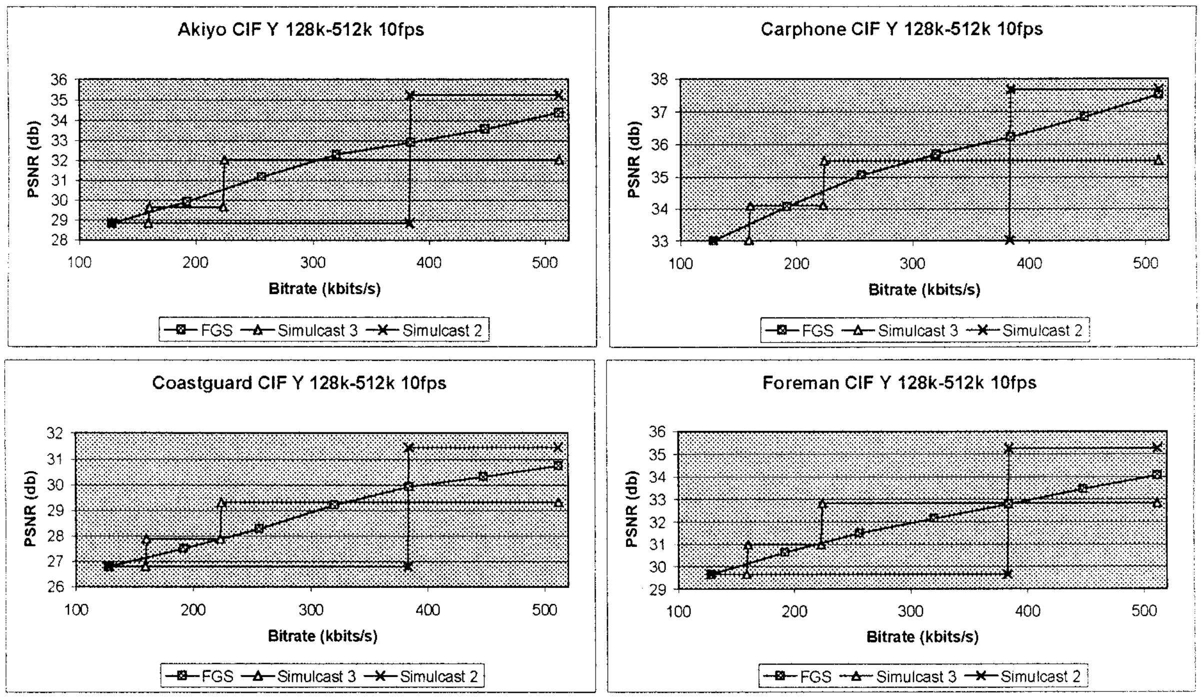


Fig. 21. Compare FGS with simulcast.

shifted_bit_planes, (a number between 0 to 4) is specified to indicate how many bit-planes are shifted up for the FGS macroblock. A flag, **fgs_vop_selective_enhancement_enable**, is set to 1 when selective enhancement is enabled or 0 otherwise. Fig. 23 illustrates how this technique is used to shift up the bit-planes in the center part of a frame.

C. Error Resilience

In wireless applications, random burst errors may occur in the FGS bitstream, too. In order to quickly isolate the errors, resynchronization markers are used in the enhancement-layer bitstream [46]. Fig. 24 shows the FGS bitstream structure with the resynchronization markers. A syntax element, **fgs_resync_marker_disable**, may be set to 1 to disable the usage of the resynchronization markers.

With or without the resynchronization markers, there is a start code, **fgs_bp_start_code**, to separate the bit-planes in each frame. This start code serves two purposes. One is to be used as a resynchronization marker for the error-resilience purpose. The other is to help the server and/or the decoder to identify the beginning of a bit plane quickly without fully decoding all the VLC codes.

D. FGS Temporal Scalability

In order to cover a wide range of bit rate with a scalable bitstream, there is a need to combine FGS with temporal scalability so that not only quantization accuracy can be scalable, but also temporal resolution (frame rate) can be scalable [47]. Since the temporal prediction for the temporal enhancement frame is restricted to the base layer, the quality of each temporal enhancement frame does not affect the quality of any other frames. Therefore, it is possible to use bit-plane coding for the entire

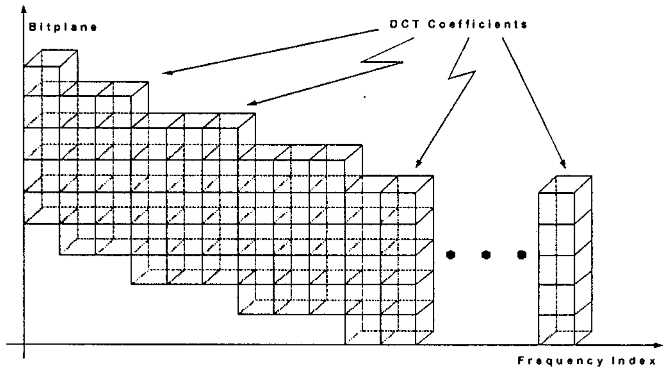


Fig. 22. Illustration of bit shifting for frequency weighting.

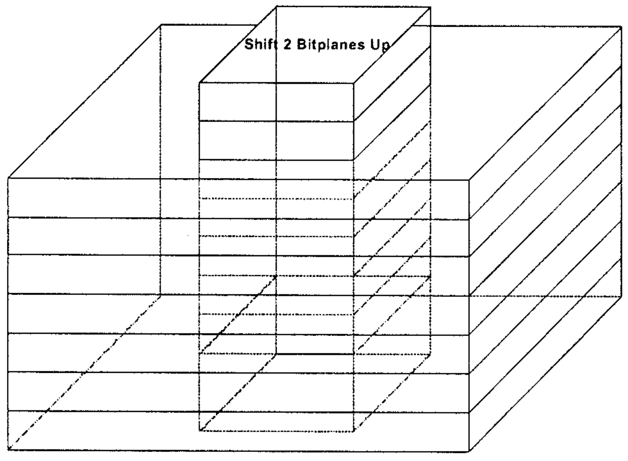


Fig. 23. Illustration of selective enhancement.

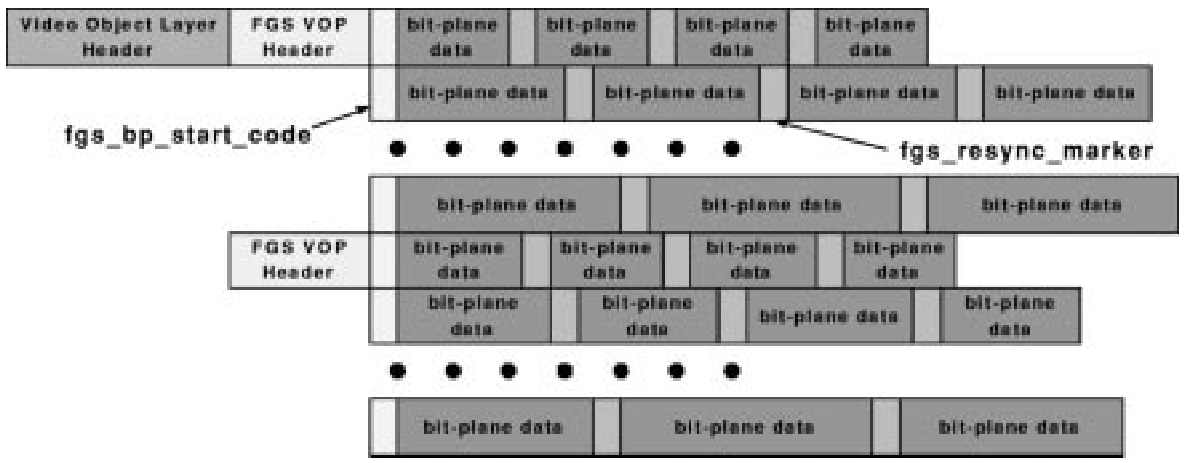


Fig. 24. FGS bitstream structure with resynchronization markers.

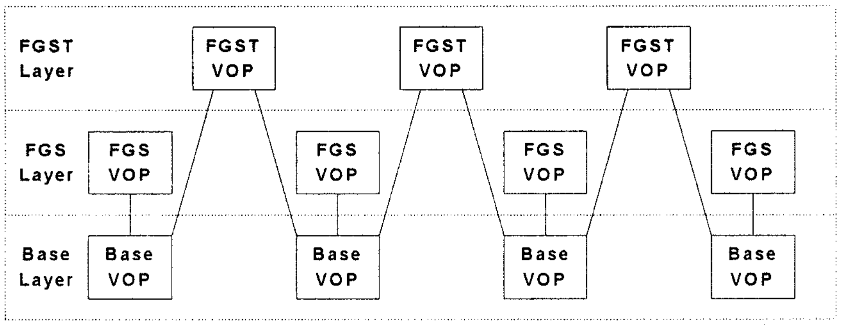


Fig. 25. FGST organized into a separate layer from FGS.

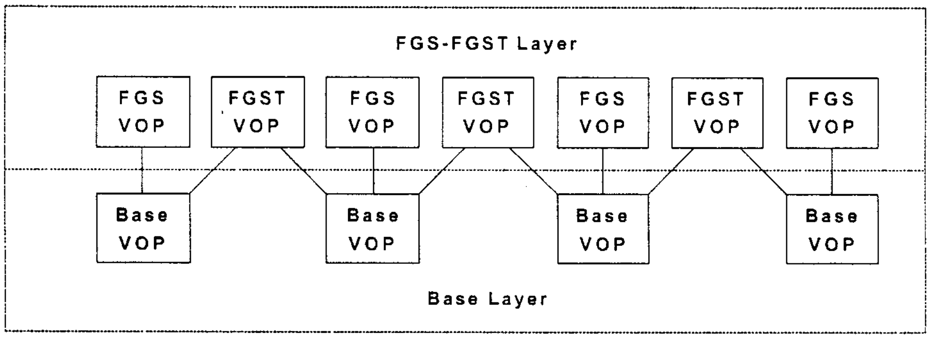


Fig. 26. FGST and FGS organized into a single enhancement layer.

DCT coefficients in the temporal enhancement frame. This is called FGS temporal scalability (FGST) because not only the temporal enhancement frames can be dropped as in the regular temporal scalability, but also the quantization accuracy is scalable within each temporal enhancement frame. As already shown in Section III, coding efficiency of bit-plane coding is actually higher than that of run-level coding. Therefore, using all bit-plane coding in the temporal enhancement frame has higher coding efficiency than regular temporal scalability for coding the DCT coefficients. This compensates its potential loss of coding efficiency by not allowing predictions in the enhancement layer.

From an implementation point of view, this approach actually saves an IDCT operation in the decoder compared with coding the DCT coefficients in both the base layer and the enhancement layer. From a syntax point of view, there are two ways to organize the temporal enhancement frames, as shown in Figs. 25 and 26. One way is to organize the them into a separate layer from the FGS frames (Fig. 25). This requires dealing with "two enhancement layers". It provides more flexibility for servers and decoders to make the tradeoff between the picture quality and temporal resolution without decoding all the frames.

Another way is to organize the FGST VOPs and FGS VOPs into a single enhancement layer (Fig. 26). This requires transmission of only one enhancement layer, but needs to decode frame headers to determine whether a frame is FGS or FGST. There is a syntax element, **fgs_layer_type**, that signals which configuration is used in an enhancement layer. This is a 2-bit code indicating whether this layer is FGS only (the middle enhancement layer in Fig. 25), FGST only (the top enhancement layer in Fig. 25), or a combination of FGS and FGST (the enhancement layer in Fig. 26). Each FGST VOP can be coded using forward prediction or the bi-directional prediction from the base layer. The syntax element, **fgs_vop_coding_type**, in the frame header identifies whether a frame is an FGS VOP, predictive-coded FGS VOP, or bi-directionally predictive-coded FGS VOP. The syntax element **ref_select_code** indicates the reference for the P or B frames in FGST. In P-type coding, the reference can be either the past or the future frame in the base layer. In B-type coding, both past and future frames in the base layer are used for prediction.

VII. Conclusion

The Amendment on Streaming Video Profile of MPEG-4 is established in response to a growing need for a video-coding standard to deliver video over a channel, such as the Internet, with a wide range of bit rates and a wide range of bit rate variations. The base-layer coding tools included in the Amendment, which are also forming the Advanced Simple Profile, are carefully chosen to provide high coding efficiency with low implementation complexity. Bit-plane coding of the DCT coefficients is a very natural and easy way to meet the requirement of graceful changes in video quality while user channel bandwidth changes. The features in FGS coding are chosen to balance coding efficiency and implementation complexity. Extensive experiments have been performed during the process of establishing the Amendment. The experiment results have shown that FGS technique is much more efficient than multilayer SNR scalability. Compared with nonscalable coding, which is the upper bound for any scalable coding techniques, FGS is about 2-dB worse at the high end of the bit-rate range. Compared with simulcast of two and three bitstreams, FGS has better coding efficiency at the low and high ends of a bit rate range respectively. The reference software implementation and the core experiment process have ensured the robustness of the FGS technique. There are many advantages of using FGS for Internet streaming video applications: it allows separation of encoding and transmission, the server can transmit enhancement layer at any bit rate without transcoding, it enables video broadcast on the Internet to reach a large audience, and it provides a solution to the video server overload problem. It is expected that the Amendment of MPEG-4 will be used for many applications. Of course, there are many network related issues to be addressed in order to take full advantage of FGS video coding, which is out of the scope of this paper.

Acknowledgment

The work for this Amendment of MPEG-4 Visual Standard has truly been a team effort. Many people have made contributions to it. Specifically, the author would like to thank F. Ling, X. Chen, K. Panusopone, and A. Luthra of General Instrument (now Motorola), A. Vetro and H. Sun of Mitsubishi Electric, Z. Wu, T. Kanamaru, Y. Ueda, and Y. Yamada of OKI Electric Industry Co., Ltd., H. Sun of Vector Vision, K. Kiser, J. Kasner, and J. Hill of Optivision, H. Jiang and G. Thayer of Intel, F. Wu, S. Li, and Y.-Q. Zhang of Microsoft Research China, Y. Chen, M. van der Schaar, and H. Radha of Philips Research, B. Thebault of France Telecom, M. Zhou of Texas Instruments, and B. Schuster of Alcatel for their support and contributions. The author would especially like to thank Dr. J.-R. Ohm of HHI for his leadership as the Ad-Hoc Group Chair for the FGS work in MPEG-4. Finally, the author thanks the reviewers for providing excellent comments on the paper.

References

[1] *Coding of Audio-Visual Objects, Part-2 Visual, Amendment 4: Streaming Video Profile*, ISO/IEC 14 496-2/FPDAM4, July 2000.

[2] B. G. Haskell, A. Puri, and A. N. Netravali, *Digital Video: An Introduction to MPEG-2*. New York: Chapman & Hall, Sept. 1996.

[3] *Generic Coding of Moving Pictures and Associated Audio, Part-2 Video*, Nov. 1994. ISO/IEC 13 818-2.

[4] R. Aravind, M. R. Civanlar, and A. R. Reibman, "Packet loss resilience of MPEG-2 scalable video coding algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 426–435, Oct. 1996.

[5] A. Puri and T. Chen, Eds., *Multimedia Systems, Standards, and Networks*. New York: Marcel Dekker, Mar. 2000.

[6] *Coding of Audio-Visual Objects, Part-2 Visual*, Dec. 1998. ISO/IEC 14 496-2.

[7] B. G. Haskell, P. G. Howard, Y. A. LeCun, A. Puri, J. Ostermann, M. R. Civanlar, L. Rabiner, L. Bottou, and P. Haffner, "Image and video coding—Emerging standards and beyond," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 814–837, Nov. 1998.

[8] M. Ghanbari, "Two-layer coding of video signals for VBR networks," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 771–781, June 1989.

[9] H. Sun, W. Kwok, and J. W. Zdepski, "Architectures for MPEG compressed bitstream scaling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 191–199, Apr. 1996.

[10] H. Gharavi and M. H. Partovi, "Multilevel video coding and distribution architectures for emerging broadband digital networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 459–469, Oct. 1996.

[11] M. Domanski, A. Luczak, and S. Mackowiak, "Spatial-temporal scalability for MPEG video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 1088–1093, Oct. 2000.

[12] H. Katata, N. Ito, and H. Kusao, "Temporal-scalable coding based on image content," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 52–59, Feb. 1997.

[13] G. J. Conklin and S. S. Hemami, "A comparison of temporal scalability techniques," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, pp. 909–919, Sept. 1999.

[14] D. Wilson and M. Ghanbari, "Exploiting interlayer correlation of SNR scalable video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, pp. 783–797, Aug. 1999.

[15] J. F. Arnold, M. R. Frater, and Y. Wang, "Efficient drift-free signal-to-noise ratio scalability," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 70–82, Feb. 2000.

[16] R. Mathew and J. F. Arnold, "Layered coding using bitstream decomposition with drift correction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 882–891, Dec. 1997.

[17] M. Ghanbari and J. Azari, "Effect of bit rate variation of the base layer on the performance of two-layer video codecs," *IIEEE Trans. Circuits Syst. Video Technol.*, vol. 4, pp. 8–17, Feb. 1994.

[18] U. Benzler, "Spatial scalable video coding using a combined subband-DCT approach," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 1080–1087, Oct. 2000.

[19] J.-R. Ohm, "Advanced packet-video coding based on layered VQ and SBC techniques," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, pp. 208–221, June 1993.

[20] T. Naveen and J. W. Woods, "Motion compensated multiresolution transmission of high definition video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, pp. 29–41, Feb. 1994.

[21] T. K. Tan, K. K. Pang, and K. N. Ngan, "A frequency scalable coding scheme employing pyramid and subband techniques," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, pp. 203–207, Apr. 1994.

[22] R. Mokry and D. Anastassiou, "Minimal error drift in frequency scalability for motion-compensated DCT coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, pp. 392–406, Aug. 1994.

[23] W. Li, F. Ling, and H. Sun, "Bitplane coding of DCT coefficients,", ISO/IEC JTC1/SC29/WG11, MPEG97/M2691, Oct. 22, 1997.

[24] F. Ling, W. Li, and H. Sun, "Bitplane coding of DCT coefficients for image and video compression," in *Proc. SPIE Visual Communications and Image Processing (VCIP)*, San Jose, CA, Jan. 25–27, 1999.

[25] K. Shen and E. J. Delp, "Wavelet based rate scalable video compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, pp. 109–122, Feb. 1999.

[26] Q. Wang and M. Ghanbari, "Scalable coding of very high resolution video using the virtual zerotree," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 719–727, Oct. 1997.

[27] P.-Y. Cheng, J. Li, and C.-C. J. Kuo, "Rate control for an embedded wavelet video coder," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 696–702, Aug. 1997.

[28] S. A. Martucci, I. Sodagar, T. Chiang, and Y.-Q. Zhang, "A zerotree wavelet video coder," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 109–118, Feb. 1997.

[29] D. Taubman and A. Zakhor, "A common framework for rate and distortion based scaling of highly scalable compressed video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 329–354, Aug. 1996.

[30] W. Li, "Bit-plane coding of DCT coefficients for fine granularity scalability," ISO/IEC JTC1/SC29/WG11, MPEG98/M3989, Oct. 1998.

[31] B. Schuster, "Fine granular scalability with wavelets coding," ISO/IEC JTC1/SC29/WG11, MPEG98/M4021, Oct. 1998.

[32] Y. Chen, H. Radha, and R. A. Cohen, "Results of experiment on fine granular scalability with wavelet encoding of residuals," ISO/IEC JTC1/SC29/WG11, MPEG98/M3988, Oct. 1998.

[33] J. Liang, J. Yu, Y. Wang, M. Srinath, and M. Zhou, "Fine granularity scalable video coding using combination of MPEG4 video objects and still texture objects," ISO/IEC JTC1/SC29/WG11, MPEG98/M4025, Oct. 1998.

[34] S.-C. S. Cheung and A. Zakhor, "Matching pursuit coding for fine granular video scalability," ISO/IEC JTC1/SC29/WG11, MPEG98/M3991, Oct. 1998.

[35] M. Bénetière and C. Dufour, "Matching pursuits residual coding for video fine granular scalability," ISO/IEC JTC1/SC29/WG11, MPEG98/M4008, Oct. 1998.

[36] W. Li, "Fine granularity scalability using bit-plane coding of DCT coefficients," ISO/IEC JTC1/SC29/WG11, MPEG98/M4204, Dec. 1, 1998.

[37] F. Ling and X. Chen, "Report on fine granularity scalability using bit-plane coding," ISO/IEC JTC1/SC29/WG11, M4311, Nov. 1998.

[38] H. Jiang, "Experiment on post-clip FGS enhancement," ISO/IEC JTC1/SC29/WG11, MPEG00/M5826, Mar. 2000.

[39] W. Li, "Verification of post-clipping addition results," ISO/IEC JTC1/SC29/WG11, MPEG00/M5967, Mar. 2000.

[40] ——, "Syntax of fine granularity scalability for video coding," ISO/IEC JTC1/SC29/WG11, MPEG99/M4792, July 1999.

[41] W. Li and Y. Chen, "Experiment result on fine granularity scalability," ISO/IEC JTC1/SC29/WG11, MPEG99/M4473, Mar. 1999.

[42] W. Li, F. Ling, and X. Chen, "Comparison of FGS with simulcast," ISO/IEC JTC1/SC29/WG11, MPEG99/M5083, Sept. 1999.

[43] A. Luthra, "Need for simple streaming video profile," ISO/IEC JTC1/SC29/WG11, MPEG00/M5800, Mar. 2000.

[44] H. Jiang and G. M. Thayer, "Using frequency weighting in FGS bit-plane coding for natural video," ISO/IEC JTC1/SC29/WG11, MPEG99/M5489, Dec. 1999.

[45] W. Li, "Frequency weighting for FGS," ISO/IEC JTC1/SC29/WG11, MPEG99/M5589, Dec. 1999.

[46] R. Yan, F. Wu, S. Li, and Y.-Q. Zhang, "Error resilience methods in the FGS enhancement bitstream," ISO/IEC JTC1/SC29/WG11, MPEG00/M6207, July 2000.

[47] M. van der Schaar, H. Radha, and Y. Chen, "An all FGS solution for hybrid temporal-SNR scalability," ISO/IEC JTC1/SC29/WG11, MPEG99/M5552, Dec. 1999.

Weiping Li (F'00) received the B.S. degree from the University of Science and Technology of China (USTC) in 1982 and the M.S. and Ph.D. degrees from Stanford University, Stanford, CA, in 1983 and 1988, respectively, all in electrical engineering.

In 1987, he joined the Faculty at Lehigh University, where he is currently a Professor in the Department of Electrical Engineering and Computer Science. Since 1998, he has taken a leave from Lehigh University to work on network streaming video in Silicon Valley, CA. He was with Optivision, Inc., Palo Alto, CA, from 1998 to 1999, as the Director of R&D. He is currently with WebCast Technologies, Inc., Mountain View, CA, a start-up company that he co-founded.

Dr. Li is currently the Editor-in-Chief for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and previously served as an Associate Editor from 1995 to 1999. He was one of the Guest Editors for the Special Issue on Image and Video Compression of IEEE PROCEEDINGS (February 1995). He served as the Past Chair (1998–1999) and Chair (1996–1998) of the Technical Committee on Visual Signal Processing and Communications of the IEEE Circuits and Systems Society. He is the Program Chair of the MPEG-4 Workshop and Exhibition (2001) and was a Co-Chair (1999) and Chair (1997) of Technical Track on Multimedia and Communications for the IEEE International Symposium on Circuits and Systems. He served as the Chair of Best Student Paper Award Committee for the 1999 SPIE Visual Communications and Image Processing Conference, and from 1997 to 1998, he served as the Chair of Working Group on reaffirmation of IEEE Standard 1180 (Specifications for the Implementation of 8×8 Inverse Discrete Cosine Transform). He is currently an Editor for the Amendment-4 on Streaming Video Profile of MPEG-4 International Standard. Since 1995, he has been a member of the MPEG of the International Standard Organization. He received the Spira Award for Excellence in Teaching in 1992 from Lehigh University and the Guo Mo-Ruo Prize for Outstanding Student in 1980 from USTC. He has been elected to Fellow of IEEE for contributions to image and video coding algorithms, standards, and implementations.