The MPEG-4 Fine-Grained Scalable Video Coding Method for Multimedia Streaming Over IP

Hayder M. Radha, *Member, IEEE*, Mihaela van der Schaar, and Yingwei Chen

Abstract—**Real-time streaming of audiovisual content over the Internet is emerging as an important technology area in multimedia communications. Due to the wide variation of available bandwidth over Internet sessions, there is a need for scalable video coding methods and (corresponding) flexible streaming approaches that are capable of adapting to changing network conditions in real time. In this paper, we describe a new scalable video-coding framework that has been adopted recently by the MPEG-4 video standard. This new MPEG-4 video approach, which is known as Fine-Granular-Scalability (FGS), consists of a rich set of video coding tools that support quality (i.e., SNR), temporal, and hybrid temporal-SNR scalabilities. Moreover, one of the desired features of the MPEG-4 FGS method is its simplicity and flexibility in supporting unicast and multicast streaming applications over IP.**

Index Terms—**Author, please supply index terms. E-mail keywords@ieee.org for info.**

I. Introduction

THE transmission of multimedia content over the World Wide Web (WWW) has been growing steadily over the past few years. This is evident from the large number of popular web sites that include multimedia content specifically designed for streaming applications. The growth in streaming audiovisual information over the web has being increasing rather dramatically without any evidence of the previously-feared collapse in the Internet or its global backbone. Consequently, multimedia streaming and the set of applications that rely on streaming are expected to continue growing. Meanwhile, the current quality of streamed multimedia content, in general, and video in particular still needs a great deal of improvement before Internet video can be accepted by the masses as an alternative to television viewing. A primary objective of most researchers in the field, however, is to mature Internet video solutions to the level when viewing of good-quality video of major broadcast television events (e.g., the Super Bowl, Olympics, World Cup, etc.) over the WWW becomes a reality [10]–[15].

To achieve this level of acceptability and proliferation of Internet video, there are many technical challenges that have to be addressed in the two areas of video-coding and networking. One generic framework that addresses both the video-coding and networking challenges associated with Internet video is scalability. From a video-coding point-of-view, scalability plays a crucial role in delivering the best possible video quality over unpredictable "best-effort" networks. Bandwidth variation is one of the primary characteristics of "best-effort" networks, and the Internet is a prime example of such networks [38]. Therefore, video scalability enables an application to adapt the streamed-video quality to changing network conditions (and specifically to bandwidth variation). From a networking point-of-view, scalability is needed to enable a large number of users to view any desired video stream, at anytime, and from anywhere. This leads to the requirement that servers and the underlying transport protocols should be able to handle the delivery of a very large number (hundreds, thousands, or possibly millions) of video streams simultaneously.

Consequently, any scalable Internet video-coding solution has to enable a very simple and flexible streaming framework, and hence, it must meet the following requirements [3].

1) The solution must enable a streaming server to perform *minimal real-time* processing and rate control when outputting a very large number of simultaneous unicast (on-demand) streams.
2) The scalable Internet video-coding approach has to be highly adaptable to unpredictable bandwidth variations due to heterogeneous access-technologies of the receivers (e.g., analog modem, cable mode, xDSL, etc.) or due to dynamic changes in network conditions (e.g., congestion events).
3) The video-coding solution must enable low-complexity decoding and low-memory requirements to provide common receivers (e.g., set-top-boxes and digital televisions), in addition to powerful computers, the opportunity to stream and decode any desired Internet video content.
4) The streaming framework and related scalable video-coding approach should be able to support both multicast and unicast applications. This, in general, eliminates the need for coding content in different formats to serve different types of applications.
5) The scalable bitstream must be resilient to packet loss events, which are quite common over the Internet.

The above requirements were the primary drivers beyond the design of the fine-granular-scalability (FGS) video-coding scheme introduced originally in [1]. Although there are other promising video-coding schemes that are capable of supporting different degrees of scalability, they, in general, do not meet all of the above requirements. For example, the three–dimensional (3-D) wavelet/sub-band-based coding schemes require large

Manuscript received May 31, 2000; revised December 1, 2000. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. K. J. R. Liu.

H. M. Radha is with the Video Communications Research Department, Philips Research Laboratories, Briarcliff Manor, NY 10510 USA, and also with the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48823 USA (e-mail: radha@egr.msu.edu).

M. van der Schaar and Y. Chen are with the Video Communications Research Department, Philips Research Laboratories, Briarcliff Manor, NY 10510 USA.

Publisher Item Identifier S 1520-9210(01)01863-6.

memory at the receiver, and consequently they are undesirable for low-complexity devices [19]–[21]. In addition, some of these methods rely on motion-compensation to improve the coding efficiency at the expense of sacrificing scalability and resilience to packet losses [19], [20]. Other video-coding techniques totally avoid any motion-compensation and consequently sacrifice a great deal of coding efficiency [21], [37].

The FGS framework, as explained further in the document, strikes a good balance between coding efficiency and scalability while maintaining a very flexible and simple video-coding structure. When compared with other packet-loss resilient streaming solutions (e.g., [40], [41]), FGS has also demonstrated good resilience attributes under packet losses [42]. Moreover, and after new extensions and improvements to its original framework,[1] FGS has been recently adopted by the ISO MPEG-4 video standard as the core video-coding method for MPEG-4 streaming applications [4]. Since the first version of the MPEG-4 FGS draft standard [5], there have been several improvements introduced to the FGS framework. In particular, we highlight three aspects of the improved FGS method. First, a very simple residual-computation approach was proposed in [6]. Despite its simplicity, this approach provides the same or better performance than the performance of more elaborate residual-computation methods. (As explained later, yet another alternative approach for computing the FGS residual has been proposed very recently [46]). Second, an "adaptive quantization" approach was proposed in [7], and it resulted in two FGS-based video-coding tools. Third, a hybrid all-FGS scalability structure was also proposed recently [8], [9]. This novel FGS scalability structure enables quality [i.e., signal-to-noise-ratio (SNR)], temporal, or both temporal-SNR scalable video coding and streaming. All of these improvements to FGS (i.e., simplified residual computation, "adaptive-quantization," and the new all-FGS hybrid scalability) have already been adopted by the MPEG-4 video standard [4].

In this paper, we describe the MPEG-4 FGS framework and its new video coding tools which have not been presented outside the MPEG-4 community. The remainder of the paper is organized as follows. Section II describes the SNR FGS framework, its ability in supporting unicast and multicast Internet video applications, and its basic coding tools. Section III presents the "adaptive quantization" approach for the FGS enhancement-layer signal and the related video-coding tools adopted by MPEG-4. Section IV describes the FGS-based hybrid temporal-SNR scalability method. Simulation results will be shown in each section to demonstrate the performance of the corresponding video coding tool. Section V concludes the paper with a summary.

II. SNR FGS Video Coding Method

In order to meet the requirements outlined in the previous section, FGS encoding is designed to cover any desired bandwidth range while maintaining a very simple scalability structure. As shown in Fig. 1, the FGS structure consists of only two layers: a base-layer coded at a bitrate R_b and a single enhancement-layer coded using a fine-granular (or embedded) scheme to a maximum bitrate of R_e. This structure provides a very efficient, yet simple, level of abstraction between the encoding and streaming processes. The encoder only needs to know the range of bandwidth $[R_{\min} = R_b,\ R_{\max} = R_e]$ over which it has to code the content, and it does not need to be aware of the particular bitrate the content will be streamed at. The streaming server on the other hand has a total flexibility in sending any desired portion of any enhancement layer frame (in parallel with the corresponding base layer picture), without the need for performing complicated real-time rate control algorithms. This enables the server to handle a very large number of unicast streaming sessions and to adapt to their bandwidth variations in real-time. On the receiver side, the FGS framework adds a small amount of complexity and memory requirements to any standard motion-compensation based video decoder. These advantages of the FGS framework are achieved while maintaining rather surprisingly good coding-efficiency results (as will be illustrated at the end of this section).

For multicast applications, FGS also provides a flexible framework for the encoding, streaming, and decoding processes. Identical to the unicast case, the encoder compresses the content using any desired range of bandwidth $[R_{\min} = R_b,\ R_{\max} = R_e]$. Therefore, the same compressed streams can be used for both unicast and multicast applications. At time of transmission, the multicast server partitions the FGS enhancement layer into any preferred number of "multicast channels" each of which can occupy any desired portion of the total bandwidth (see Fig. 2). At the decoder side, the receiver can "subscribe" to the "base-layer channel" and to any number of FGS enhancement-layer channels that the receiver is capable of accessing (depending for example on the receiver access bandwidth). It is important to note that regardless of the number of FGS enhancement-layer channels that the receiver subscribes to, the decoder has to decode only a single enhancement-layer as shown in Fig. 2.

The above approach for multicasting FGS [12] is based on the receiver-driven layered multicast framework that is supported by the IP Multicast backBONE (i.e., the MBONE) [36], [37]. Therefore, the IP multicast control and routing protocols needed for multicasting FGS-based streams have already been defined and supported by IP-multicast enabled routers. Moreover, new multicast routing protocols and architectures can further enhance the delivery of FGS-based multicast applications [10], [11], [15]–[17].

After this overview of the FGS framework, below we describe the basic SNR-based FGS encoder and decoder.

A. Basic FGS Encoder and Decoder

As shown in Fig. 1, the FGS framework requires two encoders, one for the base-layer and the other for the enhancement layer. The base-layer can be compressed using any motion-compensation video encoding method. Naturally, the DCT-based MPEG-4 video standard is a good candidate for the base-layer encoder due to its coding efficiency especially at low bitrates. Prior to introducing FGS, MPEG-4 included a very rich set of video coding tools most of which are applicable for the FGS

[1]The original FGS framework was introduced and described in [1] and [3]. Meanwhile, FGS was first introduced to MPEG-4 in [2].

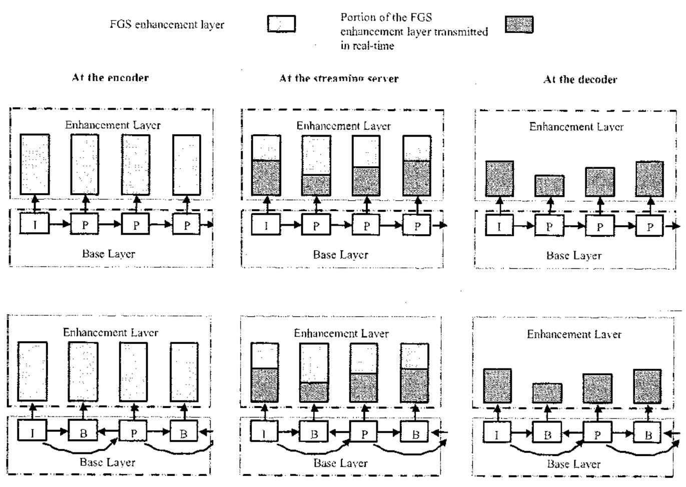

Fig. 1. Examples of the FGS scalability structure at the encoder (left), streaming server (center), and decoder (right) for a typical unicast Inernet streaming application. The top and bottom rows of the figure represent base-layers without and with bidirectional (B) frames, respectively.

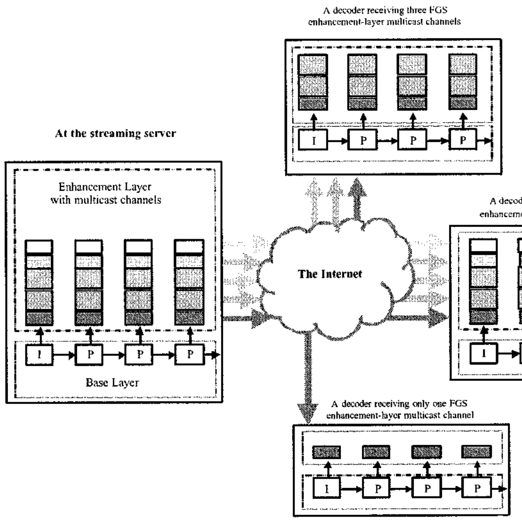
A decoder receiving three FGS enhancement-layer multicast channels
At the streaming server
Enhancement Layer with multicast channels
The Internet
A decoder receiving all five FGS enhancement-layer multicast channels
Base Layer
A decoder receiving only one FGS enhancement-layer multicast channel
I
P

Fig. 2. Example of an FGS-based multicast scenario. (The distribution of the base-layer is implicit and therefore is not shown in the figure.)

base-layer. For a complete description of these tools, the reader is referred to [4], [39].

In principle, the FGS enhancement-layer encoder can be based on any fine-granular coding method. When FGS was first introduced to MPEG-4, three approaches were proposed for coding the FGS enhancement layer: wavelet, DCT, and matching-pursuit based methods [2]. This led to several proposals and extensive evaluation of these and related approaches (see, for example, [22]–[33]). In particular, the performance of different variations of bitplane DCT-based coding [26], [27] and wavelet compression methods were studied, compared, and presented recently in [43]. Based on a thorough analysis of the FGS enhancement-layer (SNR) signal, the study in [43] concluded that both bitplane DCT coding and embedded zero-tree wavelet (EZW) based compression provide very similar results. The same conclusion was reached by the MPEG-4 FGS effort. Consequently, and due to the fact that the FGS base-layer is coded using MPEG-4 compliant DCT coding, employing embedded DCT method for compressing the enhancement layer is a sensible option [43]. Therefore, the basic SNR MPEG-4 FGS coding scheme is built upon: a) the original FGS scalability structure proposed in [1], [2], and b) embedded DCT coding of the enhancement layer as proposed in [26] and [27].

Using the DCT transform at both the base and enhancement layers enables the encoder to perform a simple residual computation[2] of the FGS enhancement-layer as shown in Fig. 3 [6]. Each DCT FGS-residual frame consists of N_{BP} bitplanes:

$$N_{BP} = \lfloor \log_2 (|C|_{\max}) \rfloor + 1$$

where $|C|_{\max}$ is the maximum DCT (magnitude) value of the residual frame under consideration.[3] After identifying $|C|_{\max}$ and the corresponding N_{BP}, the FGS enhancement-layer encoder scans the residual signal using the traditional zig-zag scanning method starting from the most significant bitplane $BP(1)$ and ending at the least significant bitplane[4] $BP(N_{BP})$ as shown in Fig. 3(b). Every bitplane consists of nonoverlapping 16×16 macroblocks (MB's), and each MB includes four 8×8 luminance (Y) blocks and two chroma blocks (U and V). Run-length codes are used for (lossless) entropy-coding of the zeros and ones in each 8×8 bitplane block [4]. This process generates variable length codes that constitute the FGS compressed bitstream. A special "all-zero blocks" code is used when all six bitplane-blocks (within a given bitplane-macroblock) do not have any bits with a value of one.

At the receiver side, the FGS bitstream is first decoded by a variable length decoder (VLD) as shown in Fig. 4. Due to the embedded nature of the FGS stream, the VLD re-generates the DCT residual bitplanes starting from the most significant bitplane toward the least significant one. Moreover, due to the type of scanning used by the FGS encoder [Fig. 3(b)], it is possible that the decoder does not receive all of the bitplane-blocks that belong to a particular bitplane. Any bitplane block not received by the decoder can be filled with zero values.[5] The resulting DCT residual is then inverse-transformed to generate the SNR residual pixels. These residual pixels are then added to the base-layer decoder output to generate the final enhanced scalable video.

In summary, the basic SNR FGS codec employs embedded DCT variable length encoding and decoding operations that re-

[2]It is important to note that there is an alternative approach for computing the FGS residual [46]. This alternative approach is based on computing the residual after clipping the base-layer reference picture in the pixel domain. Therefore, this approach, which is known as the "post clipping" method, computes the FGS residual in the pixel domain, and consequently it requires an additional DCT computation of the FGS residual prior to performing the bitplane coding. As reported in [46], there is no noticeable difference in the performance of both methods. Throughout this document we describe FGS based on the "pre clipping" residual computation approach which eliminates the need for performing DCT computation within the FGS enhancement-layer encoder.

[3]In the FGS MPEG-4 standard, three parameters are used for the number-of-bitplanes variable N_{BP}: $N_{BP}(Y)$, $N_{BP}(U)$, and $N_{BP}(V)$ for the luminance and chroma components of the video signal [4].

[4]Alternatively, the encoder may stop encoding the residual signal if the desired maximum bitrate is reached.

[5]For an "optimal" reconstruction (in a mean-square-error sense) of the DCT coefficients, one-fourth (1/4) of the received quantization step-sized as added. For example, if the decoder receives only the MSB of a coefficient (with a value x, where $x = 0$ or 1), then this coefficient is reconstructed using the value $x01000\cdots$ (i.e., instead of $x0000\cdots$).

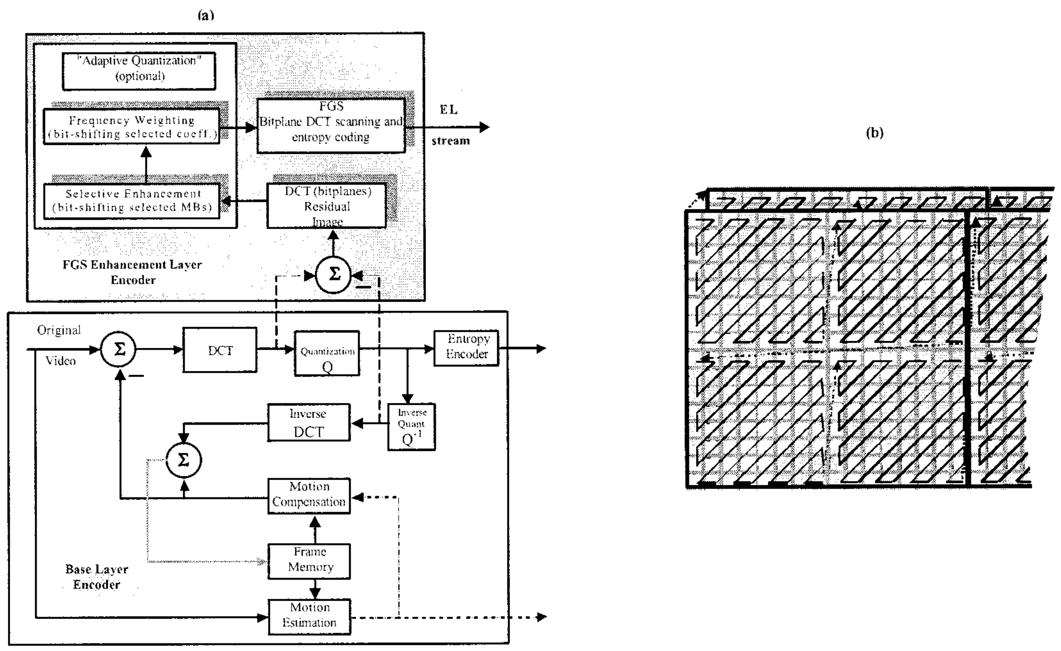
(a)
"Adaptive Quantization" (optional)
Frequency Weighting (bit-shifting selected coeff.)
Selective Enhancement (bit-shifting selected MBs)
FGS Bitplane DCT scanning and entropy coding
EL stream
DCT (bitplanes) Residual Image
FGS Enhancement Layer Encoder
Original Video
DCT
Quantization Q
Entropy Encoder
Inverse DCT
Inverse Quant Q-1
Motion Compensation
Frame Memory
Motion Estimation
Base Layer Encoder
(b)

Fig. 3. (a) Basic (SNR) FGS encoders for the base and enhancement layers. It is clear that the added complexity of the FGS enhancement-layer encoder is relatively small. (b) The scanning order of the FGS enhancement-layer residual DCT coefficients. Scanning starts from the most-significant-bitplane (MSB) toward the least-significant-bitplane. Each 8×8 bitplane-block is scanned using the traditional zig-zag pattern.

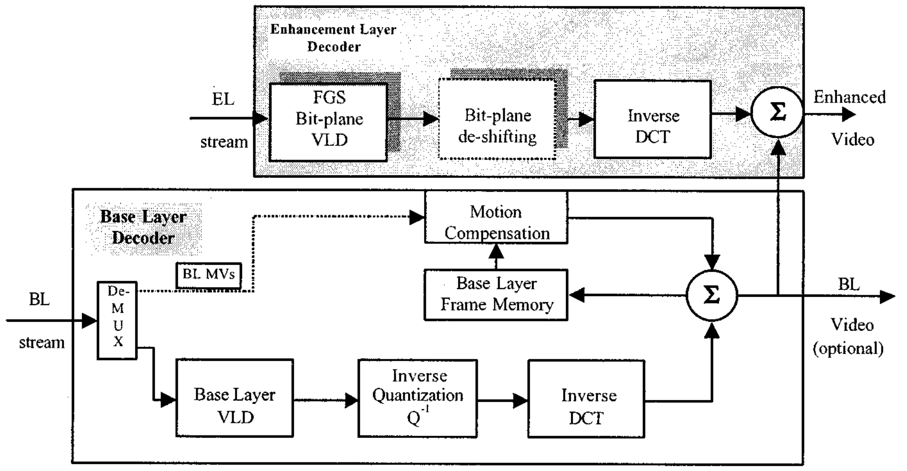

Fig. 4. Basic structure of the FGS SNR decoder. The FGS decoder includes bitplane de-shifting to compensate for the two "adaptive quantization" encoding tools: selective enhancement and frequency weighting.

semble the ones used in typical DCT based standards. In previous standards (including MPEG-4 base-layer), the DCT coefficients are coded with (run-length, amplitude) type of codes, whereas with FGS the bitplane ones-and-zeros are coded with (run-length) codes since the "amplitude" is always one. For more information about the VLC codes used by FGS, the reader is referred to [4].

B. Performance Evaluation of the FGS SNR Coding Method

The performance of FGS has been compared thoroughly with the performance of traditional SNR scalability video coding.[6] In particular, rate-distortion (RD) results of multilayer (discrete) SNR scalable video coding were compared with FGS results over wide ranges of bitrates (e.g., [28]). These results have clearly shown that FGS coding provides the same or better coding efficiency as traditional SNR scalability methods. Fig. 5 shows an example of these results for one of the MPEG-4 video sequences. As illustrated in the figure, FGS outperforms multilayer SNR scalable coding over a wide range of bitrates and for both QCIF and CIF resolution video. (For more extensive data on the comparison between FGS and multilayer SNR coding, the reader is referred to [28].)

[6]Traditional SNR scalability coding methods include the ones supported by MPEG-2 and MPEG-4 (i.e., prior to FGS).

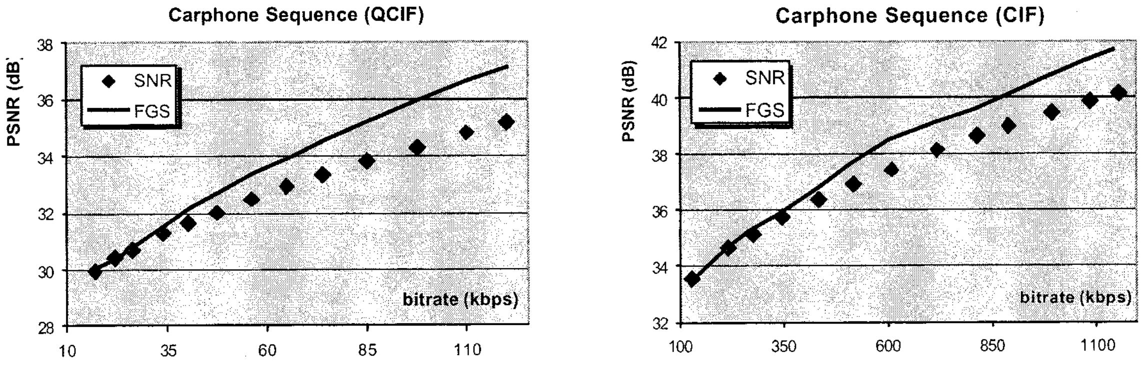

Fig. 5. Performance of FGS coding and "traditional" MPEG-4 SNR coding with multiple layers. It is clear from these plots that FGS outperforms multilayer (discrete) SNR coding over a wide range of bitrates. For more data on the comparison between the performance of FGS and multilayer SNR compression, the reader is referred to [28].

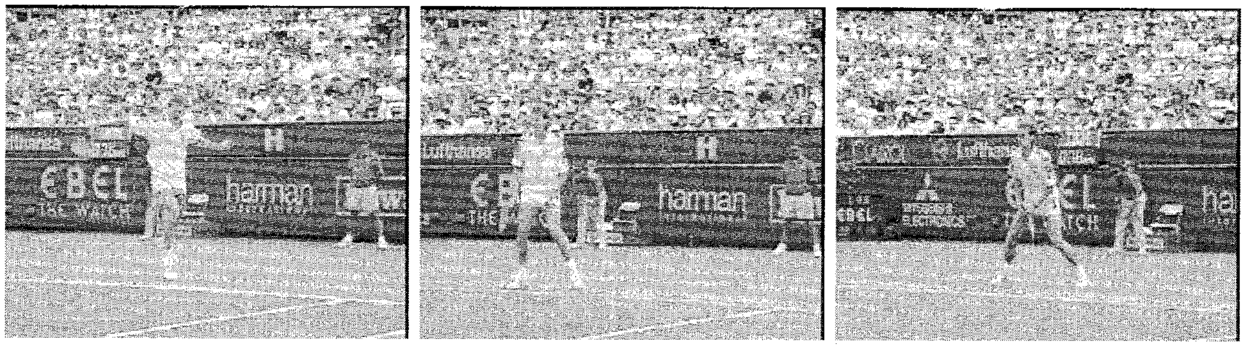

Fig. 6. Three frames from the "Stefan" sequence. This is an example of a sequence that exhibits a high-degree of temporal correlation among successive frames. The pictures shown here are 25-frame apart yet most of the background is very similar from one picture to another.

Here, we focus on two important (yet related) aspects of FGS rate-distortion performance. Before discussing these two aspects, it is important to highlight that one of the key advantages of FGS is its simplicity and flexibility in supporting adaptive streaming applications. Naturally, this flexibility comes, in general, at the expense in video quality. Hence, one important question is: how much penalty is being paid in quality when comparing FGS with a nonscalable stream that is coded at a particular bitrate R? Therefore, one aspect of FGS performance that we would like to address here is how does FGS compare with a set of nonscalable streams coded at discrete bitrates (e.g., $R_1 = R_{\min}, R_2, \cdots R_N = R_{\max}$) covering the same bandwidth range $[R_{\min}, R_{\max}]$? Although this type of comparison may seem to be unfair to FGS—since the (multiple) nonscalable streams are optimized for particular bitrates whereas FGS covers the same range of bandwidth with a single enhancement-layer, this comparison provides an insight into the theoretical (upper) limits of FGS's rate-distortion performance. Moreover, since the (ideal) nonscalable multiple-streams' scenario represents an extreme case of inflexibility, this comparison provides an insight into the level of quality-penalty being paid for FGS's flexibility.

A related aspect to the above question is the impact of the base-layer (coded at a given bitrate R_b) on the overall performance of FGS over the range of bandwidth $[R_b, R_{\max}]$. In this section, we will try to shed some light on these two aspects in a joint manner. To achieve that, we have conducted a very comprehensive evaluation of a large number of sequences with different motion and texture characteristics. Each sequence was coded at multiple (discrete) bitrates $R_1, R_2, \cdots R_N = R_{\max}$ to generate the nonscalable streams at these rates. Then, we used the nonscalable streams (coded with a bitrate R_i, $i = 1, 2, N-1$) to generate corresponding FGS streams that covers the bandwidth range $[R_i, R_{\max}]$.

To illustrate some of the key conclusions of our simulation study, we show here the results of two video sequences coded in the range 100 kbit/s to 1 Mbit/s, at 10 frame/s, and with a CIF resolution.[7] The two selected sequences: "Stefan" (shown in Fig. 6) and "Flying" (Fig. 7). These sequences, "Stefan" and "Flying," represent two types of content: one type with relatively high-temporal correlation and the other content without significant correlation among frames, respectively.

[7]This bandwidth range was selected since it represents the type of bandwidth variation one may encounter over "broadband" Internet access (e.g., cable-modem access technologies [18]) which are suitable for video streaming applications. FGS performance results over lower-bitrate bandwidth ranges were similar to the ones presented here when using lower resolution pictures (e.g., using QCIF resolution for 10 kbit/s to 100 kbit/s bitrates). Moreover, it is important to highlight here that the selected frame-rate (i.e., 10 frames/s) was chosen since it represents the adequate frame rate for a base-layer coded at around 100 kbit/s. By employing the SNR (only) scalability of FGS, the enhancement layer is "locked" to the base-layer frame rate regardless of the bitrate. This issue, which represents an intrinsic limitation of all SNR-based scalability coding methods, is resolved when using the hybrid temporal-SNR FGS scheme discussed later in this paper. In general, using higher frame rates (e.g., 15 frames/s) will lower the PSNR values for both FGS and nonscalable streams. However, and depending on the video sequence, the difference in the performance between FGS and the nonscalable streams may increase with increasing the frame rate of the base-layer stream.

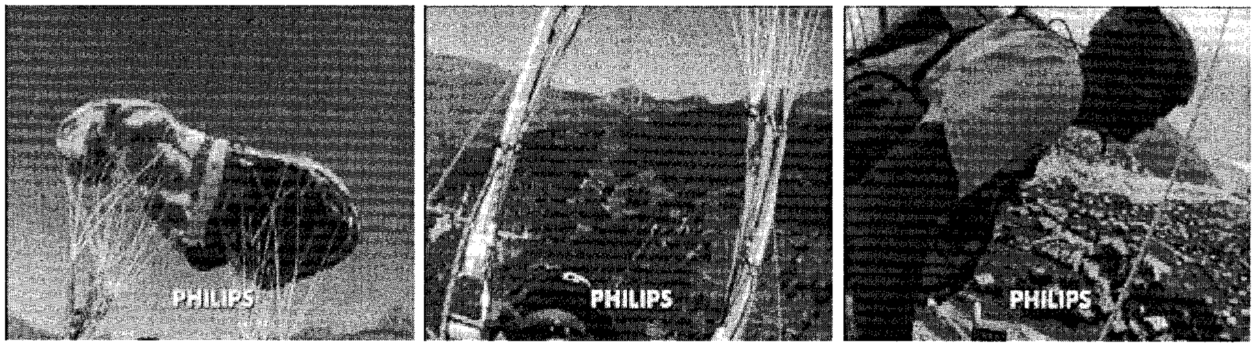

Fig. 7. Three frames from the "Flying" sequence. This is an example of a sequence that exhibits a high-degree of motion and scene changes. The pictures shown here are only 5-frame apart yet most of the visual content is changing from one picture to another.

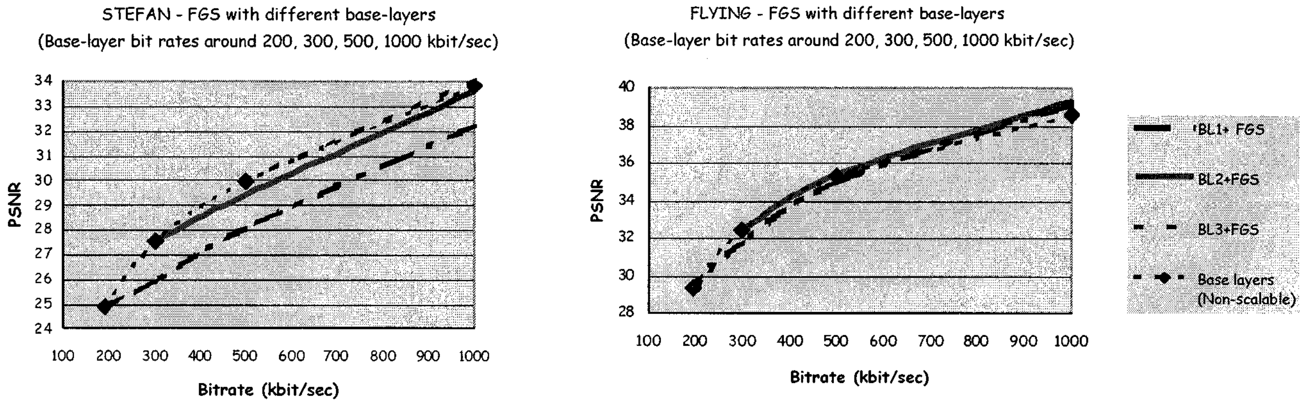

Fig. 8. FGS performance in comparison with multiple nonscalable streams' (ideal) case. The left figure shows the results for a sequence that exhibits a high-degree of temporal correlation among successive frames. For this type of sequences, FGS pay penalty in performance due to the absence of motion compensation within the enhancement-layer. The right figure shows the performance for a sequence with very-high motion and a large number of scene cuts. For this type of sequences, FGS performance is either similar to or even slightly better than the ideal nonscalable case.

The Peak SNR performance numbers are shown in Fig. 8. The left figure shows the results for the sequence "Stefan" (which is characterized by a relatively high-degree of temporal correlation among successive frames). It is clear that for these sequences, FGS pays some penalty in performance when compared with the ideal nonscalable case, due to the absence of motion compensation within the FGS enhancement-layer. This penalty manifests itself in more "blocky" video for FGS coded sequences when compared with the nonscalable streams, in particular, at low bitrates (e.g., around the 300–500 kbit/s bitrate-range for the Stefan sequence). At higher bitrates, the difference in quality is usually less visible. It is also clear that selecting a higher bitrate base-layer could provide rather significant improvement in quality at the expense of decreasing the bandwidth range that FGS covers.

The right plots in Fig. 8 show the performance for the sequence "Flying" which includes very-high motion scenes and a large number of scene cuts. For these sequences, FGS performance is either similar to or even slightly better than the ideal nonscalable case. It is also clear that, here, the impact of selecting a higher bitrate base-layer does not provide significant improvement, and therefore one can still cover the desired (wider) range of bandwidth without paying much penalty in quality. Consequently, based on our study, the following key conclusions can be made:

1) When compared with the (ideal) nonscalable coding case, FGS suffers the most for sequences with high temporal correlation between successive frames.[8] This result is somewhat intuitive since FGS exploits temporal redundancy only at the base layer, and therefore FGS suffers some coding efficiency due to lack of motion compensation at the enhancement-layer. (An example of this type of sequences is shown in Fig. 6.) Other very common examples of such sequences include simple "head-and-shoulder" scenes with static background (e.g., scenes of news anchors, talk shows, etc.).
2) On the other hand, for sequences with a high degree of motion (e.g., with a large number of scene cuts and/or very fast motion), FGS's rate-distortion performance is very good. In these cases, FGS usually (and rather surprisingly) provides similar (sometimes slightly better) coding efficiency when compared with the nonscalable (ideal) streams. (An example of this type of sequences is show in Fig. 7.) Although this type of video content is not as common as the type of sequences mentioned above (i.e., in 1), the presence of high-motion video content is growing in support of many IP streaming applications. Examples of this type of high-motion sequences include "movie trailers" (which usually contain a large number of scene changes), certain commercials, and news clips with high-action content.

[8]Here, "temporal correlation" is based on subjective observations rather than an objective measure.

3) As expected, the base-layer (and its corresponding bitrate) *could* have a major impact on the overall performance of FGS. In particular, this observation is prevalent for the sequences with high-level of temporal correlation (i.e., case 1 above).
4) There is an inherit trade-off between the overall performance and the amount of bandwidth range $[R_i,\ R_{\max}]$, $i = 1, 2, N-1$, one needs/desires to cover. For example, the average performance of FGS over a bandwidth range $[R_2,\ R_{\max}]$ could be significantly better than the average performance over the wider range $[R_1,\ R_{\max}]$ when the nonscalable streams coded at R_2 and R_1 are used as base-layers, respectively. This is usually due, in part, to the fact that the nonscalable (base-layer) stream coded at R_2 has a better quality than the lower bitrate stream R_1, and therefore, starting form a higher-quality base-layer naturally improves the overall quality. Again, this observation was only clear for sequences with high-level of temporal correlation.

In summary, FGS provides fairly acceptable to very good results even when compared with the multiple (ideal) nonscalable streams scenario. In addition, the high-level of flexibility and simplicity that FGS provides makes it an attractive solution for IP streaming applications. Moreover, FGS are further enhanced by two important video coding tools and features as described in the following two sections.

III. FGS Coding with Adaptive Quantization

Adaptive quantization is a very useful coding tool for improving the visual quality of transform-coded video. It is normally achieved through a quantization matrix that defines different quantization step sizes for the different transform coefficients within a block (prior to performing entropy coding on these coefficients). For example, the dc coefficient and other "low frequency" coefficients normally contribute more to the visual quality and consequently small step sizes are used for quantizing them. Adaptive quantization can also be controlled from one macroblock to another through a quantization factor whose value varies on a macroblock-by-macroblock basis. These adaptive quantization tools have been employed successfully in the MPEG-2 and MPEG-4 (base-layer) standards.

Performing "adaptive quantization"—AQ—on bitplane signals consisting of only ones and zeros has to be achieved through a different (yet conceptually similar) set of techniques. We first introduced the notion of adaptive quantization for the FGS bitplane signal in [7]. FGS-based AQ is achieved through *bitplane shifting* of a) selected macroblocks within an FGS enhancement layer frame, and/or b) selected coefficients within the 8×8 blocks. Bitplane shifting is equivalent to multiplying a particular set of coefficients by a power-of-two integer. For example, let assume that the FGS encoder wishes to "emphasize" a particular macroblock k within an FGS frame. All blocks within this selected macroblock k can be multiplied[9] by a factor $2^{n_{se}(k)}$

[9]Throughput the remainder of this section we will use the words "shifted," "multiplied," and "up-shifted" interchangeably.

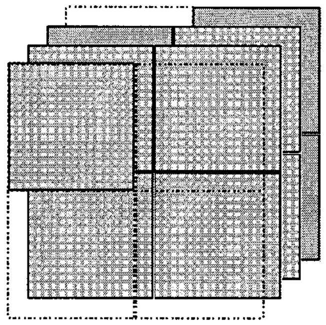

Fig. 9. Example illustrating the use of the Selective Enhancement AQ tool. In this case, a "selected" macroblock is emphasized (relative to the surrounding macroblocks) by up-shifting all coefficients within that macroblock. This generates a new bitplane when compared to the original number of bitplanes.

therefore, the new value $c'(i;\ j;\ k)$ of a coefficient i of block j (within macroblock k) is

$$c'(i,\ j,\ k) = 2^{n_{se}(k)} \cdot c(i,\ j,\ k)$$

where $c(i,\ j,\ k)$ is the original value of the coefficient. This is equivalent to *up-shifting* the set of coefficients $[c(i;\ j;\ k), i = 1,\ 2, \cdots 64\ ]$ by $n_{se}(k)$ bitplanes relative to other coefficients that belong to other macroblocks. An example of this is illustrated in Fig. 9. This type of adaptive-quantization tool is referred to as *Selective Enhancement* since through this approach selected macroblocks within a given frame can be enhanced relative to other macroblocks within the same frame.

In addition to performing bitplane shifting on selected macroblocks, FGS allows bitplane shifting of selected DCT coefficients [Fig. 10(a)]. Therefore, one can define a *frequency weighting* matrix where each element of the matrix indicates the number of bitplanes $n_{fw}(i)$ that the ith coefficient should be shifted by. Again, this is equivalent to multiplying the DCT coefficients by an integer

$$c'(i,\ j,\ k) = 2^{n_{fw}(i)} \cdot c(i,\ j,\ k).$$

Naturally, one can use both "adaptive quantization" techniques (i.e., selective enhancement and frequency weighting) simultaneously [Fig. 10(b)]. In this case, the values of the resulting coefficients can be expressed as follows:

$$c'(i,\ j,\ k) = 2^{n_{se}(k)} \cdot 2^{n_{fw}(i)} \cdot c(i,\ j,\ k).$$

It is important to note the following points regarding the MPEG-4 FGS AQ tools.

1) While selective enhancement can be employed and controlled on a macroblock-by-macroblock basis, the same frequency weighting matrix is applied to all macroblocks in the FGS frames.[10]
2) Selective enhancement is a relative operation in nature. In other words, if a large number of macroblocks are selected for enhancement, there may not be any perceived

[10]Based on the current draft standard [4], the frequency weighting matrix is applied to all FGS frames within a video-object-layer (VOL) which is equivalent to a video sequence in MPEG-2. However, one can download a new matrix by transmitting a new VOL header. In addition, two matrixes are used in FGS: one for the SNR frames and the other for the FGS temporal frames described later in this paper.

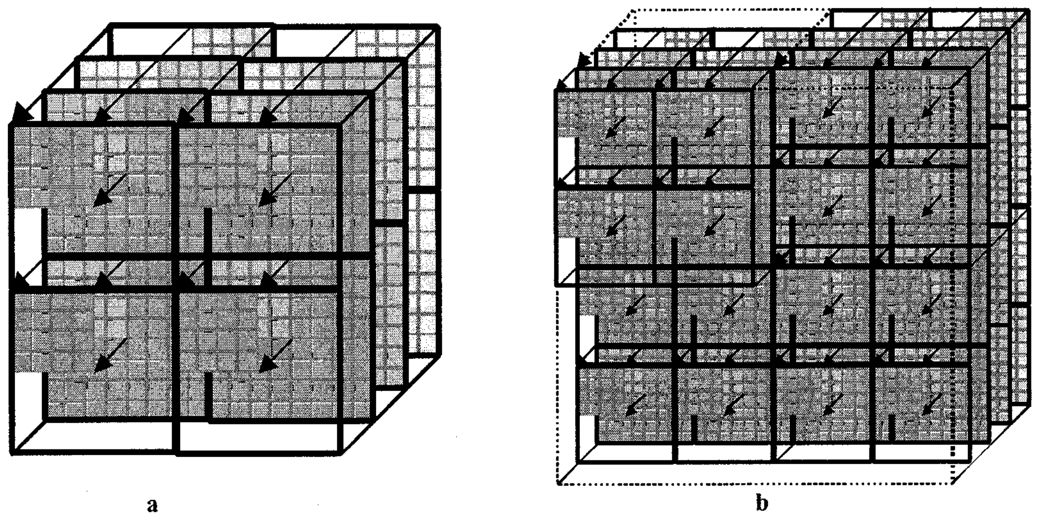

Fig. 10. (a) Example illustrating the use of the FGS Frequency Weighting AQ tool. In this example, one new bitplane is generated due to the bitplane up-shifting used by this AQ tool. (b) Example illustrating the use of both of the FGS "Adaptive Quantization" tools: Selective Enhancement and Frequency Weighting. Here, two new bitplanes are generated due to the two AQ tools employed. In this example, the original number of bitplanes is three, and only four macroblocks are shown. One bitplane is generated due to up-shifting (dotted arrows) the upper-left macroblock by using one-bitplane Selective Enhancement. The other bitplane is generated (the front dotted) due to frequency-weighting by one-bit-plane shifting of the 4×4 lowest-frequency DCT coefficients (which is applied to all blocks of all macroblocks).

improvement in quality. However, selective-enhancement of a large number of macroblocks may be used as a simple tool to de-emphasize an undesired (relatively small) region of the frame.

3) When both selective enhancement and frequency weighting are used, the up-shifting operation does not guarantee that a particular "selected-for-enhancement" macroblock gets scanned earlier[11] as compared with its original scanning order (i.e., prior to AQ). To clarify this point, let assume that, prior to AQ, the number of bitplanes $n_{bp}(k)$ needed for representing macroblock k is

$$n_{bp}(k) = \left\lfloor \log_2 \left(\max_{i,j} |c(i,j,k)| \right) \right\rfloor + 1.$$

Let assume that the coefficients within the (selected) macroblock k are subjected to $n_{se}(k)$ bitplanes' up-shifting (due to selective enhancement) and each ith coefficient is multiplied by $2^{n_{fw}(i)}$ (due to frequency weighting). Therefore, the (new) number of bitplanes $n'_{bp}(k)$ needed for representing the coefficients of macroblock k can be expressed as follows:

$$n'_{bp}(k) = n_{se}(k) + \left\lfloor \log_2 \left(\max_{i,j} \left[2^{n_{fw}(i)} \cdot |c(i,j,k)| \right] \right) \right\rfloor + 1.$$

Now let $C_{\max}$ represents the maximum (global) DCT coefficient value within the FGS frame under consideration (prior to performing any AQ operation). This leads to the number of bitplanes needed for representing the FGS frame:

$$\begin{aligned} N_{bp} &= \lfloor \log_2(C_{\max}) \rfloor + 1 \\ &= \left\lfloor \log_2 \left(\max_{i,j,k} |c(i,j,k)| \right) \right\rfloor + 1 = \max_k n_{bp}(k). \end{aligned}$$

Therefore, the number of bitplanes needed for representing the FGS frame after applying AQ is

$$\begin{aligned} N'_{bp} &= \max_k \\ &\cdot \left\{ n_{se}(k) + \left\lfloor \log_2 \left(\max_{i,j} \left[2^{n_{fw}(i)} \cdot |c(i,j,k)| \right] \right) \right\rfloor + 1 \right\} \\ &= \max_k n'_{bp}(k). \end{aligned}$$

Hence, if $[N'_{bp} - n'_{bp}(k)] > 0$, then the resulting[12] *most-significant-bitplane* of macroblock k will still be coded after some other macroblocks' MSB coefficients. In this case, the selected macroblock k coefficients will be coded using the "all-zero bitplanes" code for the first $[n'_{zero}(k) = N'_{bp} - n'_{bp}(k)]$ scanned bitplanes (i.e., instead of using this code for the first $[n_{zero} = N_{bp} - n_{bp}(k)]$ scanned bitplanes prior to AQ). Therefore, if $[n'_{zero}(k) - n_{zero}(k)]$ is positive, the (resulting) MSB of the selected-for-enhancement macroblock k will actually be scanned later when compared with the bitplane-scanning order of the (original) MSB of same macroblock k.

The FGS encoder and decoder with the AQ tools described above are illustrated in Figs. 3 and 4, respectively. At the encoder side, bitplane shifting due to selective-enhancement and

[11]To be precise, the scanning order of a macroblock is defined here by its nonzero most-significant-bitplane (MSB). Therefore, although all the macroblocks are actually scanned in the same order from one bitplane to another, the times at which the macroblocks' MSB's get scanned and coded differ from one macroblock to another.

[12]It is important to note that the resulting MSB from the bitplane shifting operation is not the same as the original MSB due to frequency weighting. However, applying selective-enhancement based bitplane shifting only (i.e., without frequency weighting) preserves the original MSB pattern of ones and zeros.

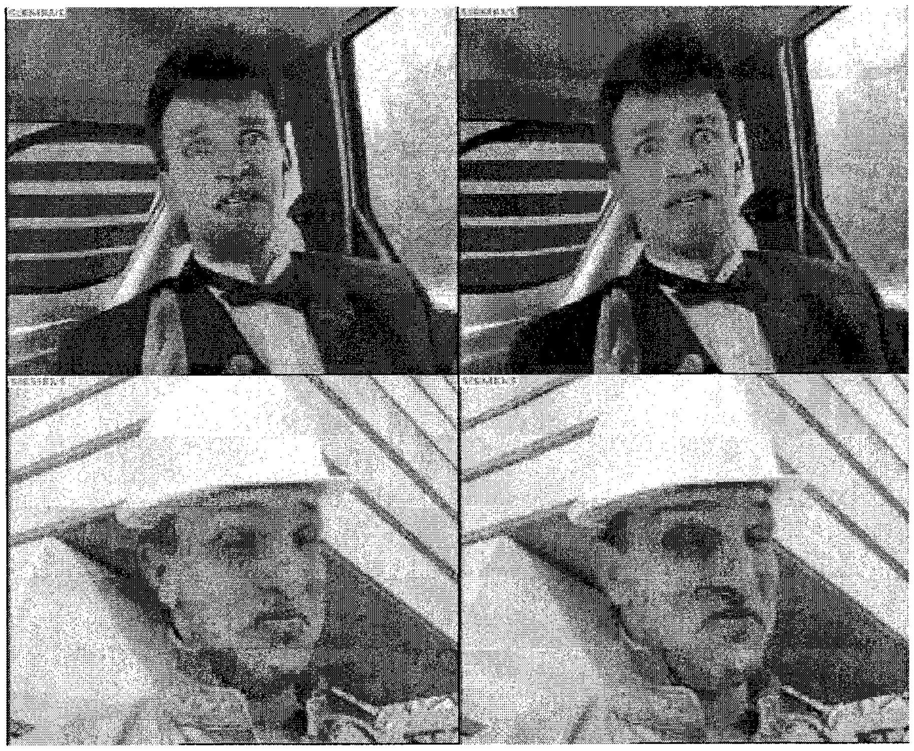

Fig. 11. Impact of FGS AQ through selective enhancement; The left images are without AQ, and the right images are with AQ.

frequency-weighting are performed on the residual FGS signal prior to the scanning and entropy coding of the bitplanes. Bitplane de-shifting is performed at the decoder side after the entropy decoding process and prior to the computation of the inverse DCT of the FGS residual signal.

A. Evaluation of FGS AQ

The aim of FGS AQ is not to improve the rate-distortion performance, but rather to improve the visual quality of the resulting video. In general, the rate-distortion performance of an FGS coder that uses AQ may actually degrade due to the overhead needed for transmitting the AQ parameters (i.e., n_{se} and n_{fw}). In particular, the transmission overhead of the Selective Enhancement shifting factors (i.e., n_{se}) may be significant since they are sent for each macroblock. On the other hand, the overhead of sequence-level (or even frame-level) transmission of the frequency-weighting matrix [i.e., the $n_{fw}(i)$] is relatively small. Below,[13] we show the performance of FGS AQ in the context of Selective Enhancement, and we briefly illustrate the impact of employing Frequency Weighting.

In order to reduce the bitrate overhead of the Selective Enhancement shifting factors n_{se}, a low overhead mechanism must be carefully designed to encode them. First, the range of allowed shifting factors is determined. Since every increment of the shifting factor corresponds to a factor-of-two decrease in the equivalent quantization step-size,[14] a moderate maximum shifting factor suffices to provide the encoder with enough space for maneuvering local control of quantization. Another consideration in determining the maximum shifting factor is the resulting maximum number of bit-planes. Without shifting, DCT coefficients are 11 bits inclusive of sign, which is less than two bytes. It is important that with selective enhancement, DCT coefficients are kept within this range. Based on the above two considerations, the maximum shifting factor is chosen to be 5.

Variable length coding is used to entropy code the shifting factors, taking advantage of the observation that smaller shifting factors are more often employed. For more information about the VLC codes used for the Selective Enhancement tool, the reader is referred to [4].

To illustrate the visual quality improvement that selective enhancement can provide, we present here simulation results for two sequences (Fig. 11): "carphone" and "foreman" in CIF (352×288) format. The sequences are encoded using an FGS reference software (known as MoMuSys version 2.1). The base layer bitrate target is 128 kbps, and the frame rate is 10 fps. The enhancement layer (cutoff) bitrate is also 128 kbps. A fixed shifting factor mask that shifts the center portion of the video by 3 bitplanes, but leaves the remaining part unchanged is used. Images from the compressed video, both with and without selective enhancement, are shown Fig. 11.

From the images shown, it can be clearly seen that this type of FGS AQ does "selectively" enhance the visual quality of chosen macroblocks. This obviously results in some quality degradation to the "unselected" parts of video. For example, notice the tree in the background of the "carphone" sequence. Although the face has been enhanced due to selective enhancement, the tree has been distorted. On the other hand, while the improve-

[13]It is important to highlight here that the visual impact of Selective Enhancement and Frequency-Weighting AQ is best evaluated through an actual viewing of a real-time playing video sequence.

[14]Up-shifting by a single bitplane can be viewed as *either* doubling the range of possible values while preserving the quantization step-size *or* dividing the quantization step-size by two while preserving the range of possible values.

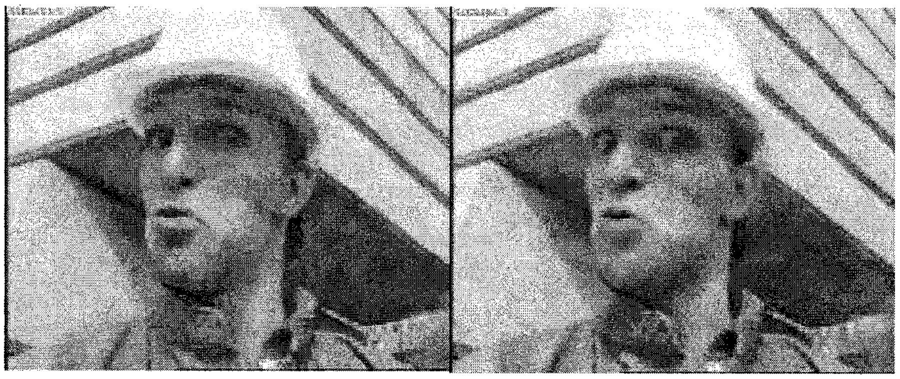

Fig. 12. Impact of employing frequency weighting on an FGS coded sequence. The left image is without frequency weighting while the right image is achieved by emphasizing low DCT frequencies.

ment to the "foreman" image is clear, there is not any salient distortion in the background.

While selective enhancement can be used to enhance a particular region of the video pictures, frequency weighting can be employed effectively to reduce some of the "blockiness" artifacts throughout an FGS coded video frame. This can be achieved by emphasizing the low frequencies, and this in turn is achieved by bitplane-shifting these frequencies. Table I shows a simple frequency weighting matrix that emphasizes low frequencies by "up shifting" them two bitplanes in the DCT domain. Fig. 12 shows the impact of using this matrix when coding an FGS video stream. It is clear that, even with such a simple matrix that emphasizes low frequencies, some of the "blockiness" artifacts can be significantly reduced. However, this enhancement may come at the expense of softening some of the sharp edges and other fine details within the video. (More simulation results for the impact of frequency weighting are reported in [34] and [35].)

In summary, utilizing the AQ tools can improve the overall perceived *visual-quality* of FGS-coded video. This can be achieved by carefully designing an effective algorithm that adjusts the shifting factors (for selective enhancement) and frequency weighting matrix. Therefore, a key challenge here is to design an algorithm that adapts to different video sequences, different scenes within the same sequence, and to different regions within a video frame. This is left as a task for FGS encoder optimization and a topic of further research.

IV. Hybrid Temporal-SNR Scalability with an All FGS Structure

Temporal scalability is an important tool for enhancing the motion smoothness of compressed video. Typically, a base-layer stream coded with a frame rate f_{BL} is enhanced by another layer consisting of video frames that do not coincide (temporally) with the base layer frames. Therefore, if the enhancement layer has a frame rate of f_{EL} then the total frame of both the base- and enhancement-layer streams is $f_{BL} + f_{EL}$.

Based on the SNR FGS scalability structure described above, the frame rate of the transmitted video is "locked" to the frame rate of the base-layer regardless of the available bandwidth and corresponding transmission bitrate. At the same time, one of the design objectives of FGS is to cover a relatively wide range of bandwidth variation over IP networks (e.g., 100 kbit/s to 1 Mbit/s). Consequently, it is quite desirable that the SNR enhancement tool of FGS be completed with a temporal scalability tool. It is also desirable to develop a framework that provides the flexibility of choosing between temporal scalability (better motion smoothness) and SNR scalability (higher quality) at transmission time. This, for example, can be used in response to a) users' preferences and/or b) real-time bandwidth variations at transmission time. For typical streaming applications, both of these elements are not known at the time of encoding the content.

Consequently, we introduced a novel framework for supporting hybrid temporal-SNR scalabilities building upon the SNR FGS structure [8], [9]. The proposed framework provides a new level of abstraction between the encoding and transmission processes by supporting *both* SNR and temporal scalabilities through a *single* enhancement-layer. As mentioned above, this abstraction is very important since the transmission bandwidth and user preferences are not known at encoding time and thus, the optimal tradeoffs between motion smoothness and quality (SNR) improvements cannot be *a-priori* made. With the proposed solution, which employs a fine-granular single-layer for both SNR and temporal scalabilities, these decisions can be easily performed at transmission time depending on the user, decoder or server requirements. Another advantage of the proposed framework is its reduced decoder complexity, requiring minimal addition to the basic MPEG-4 FGS encoder and decoder described above.

TABLE I
Example of a Frequency Weighting Matrix

2	2	2	2	2	2	0	0
2	2	2	2	2	0	0	0
2	2	2	2	0	0	0	0
2	2	2	0	0	0	0	0
2	2	0	0	0	0	0	0
2	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

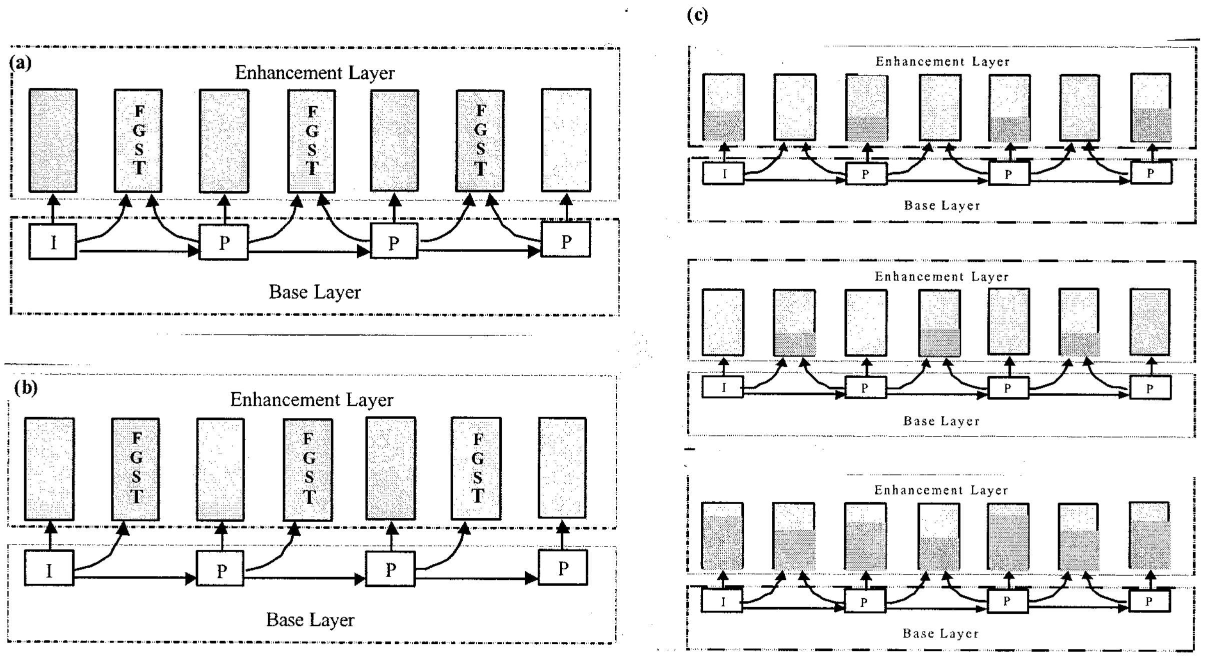

Fig. 13. Examples of the all-FGS hybrid temporal-SNR scalability structure. Both (a) bidirectional and (b) forward-prediction FGST picture types are supported by the MPEG-4 FGS standard. (c) Examples of supporting (top) SNR-only, (middle) temporal-only, or (bottom) both temporal-and-SNR scalable streaming based on the hybrid all-FGS temporal-SNR scalability structure shown in (a).

Fig. 13 shows the proposed hybrid scalability structure. In addition to the standard SNR FGS frames, this hybrid structure includes motion-compensated residual frames at the enhancement layer. We refer to these motion-compensated frames as the FGS temporal (FGST) pictures. As shown in the figure, each FGST picture is predicted from base-layer frames that do not coincide temporally with that FGST picture, and therefore, this leads to the desired temporal scalability feature. Moreover, the FGST residual signal is coded using the same fine-granular video coding method employed for compressing the standard SNR FGS frames.

Consequently, each FGST picture includes two types of information: a) motion vectors (MV's) which are computed in reference to temporally adjacent base-layer frames and b) coded data representing the bitplanes' DCT signal of the motion-compensated FGST residual. The MV's can be computed using standard macroblock-based matching motion-estimation methods. Therefore, the motion-estimation and compensation functional blocks of the base layer can be used by the enhancement-layer codec as explained below.

The two sets of information (i.e., MV's and bitplane-DCT residual signals) of an FGST picture are coded and transmitted using a data-partitioning strategy. Under this strategy, after the FGST video-object-plane (VOP)[15] header, all motion-vectors are clustered and transmitted first and then the coded representation of the DCT bitplanes' residual signal. This strategy provides a useful packet-loss resilience tool by enabling the transmission of the MV data in designated packets (i.e., separate from the residual-DCT signal packets of both SNR and temporal FGS frames). These MV designated packets can then be provided with a higher level of protection than other packets, and this, consequently, reduces the negative impact of packet losses on the motion-compensated FGST frames.

As mentioned above, the hybrid temporal-SNR FGS structure enables the server to support SNR-only, temporal-only, or both temporal-SNR scalabilities. An example of this is shown in Fig. 13. It is important to note that although the FGST frames can be transmitted in a fine-granular way, at minimum *all* of the MV data of a given (transmitted) FGST picture should be sent first. This enables the decoder to perform the motion compensation first and then perform the bitplane DCT decoding of the FGST frames. Therefore, the decoder can first reconstruct the motion-compensated ("reference") frame that represents the base-signal over which all residual-data is added. This can be advantageous, for example, in cases when the decoder is experiencing some computational-power limitations. Consequently, in addition to the packet-loss resilience benefit mentioned above, this highlights another advantage of the data-partitioning strategy used for generating the FGST frames' bitstream.

Fig. 14 shows a functional architecture for the hybrid temporal-SNR FGS encoder. It is important to note that although the SNR FGS residual can be computed directly in the DCT domain, the FGST residual is computed in the pixel domain. Therefore, the FGST residual frames have to be DCT transformed prior to their bitplane-based entropy coding. In addition, and as mentioned above, the FGST residual is computed based on a motion-compensation approach from base-layer pictures.

Despite these additional computations[16] needed for FGST frames' coding when compared with the standard SNR FGS picture coding, one can realize these additional computations

[15]Using MPEG-4 terminology, VOP is the set of words used for a picture.

[16]In other words, computing the motion-compensation residual in the pixel domain and then computing the DCT transform of this motion-compensated residual for every FGST picture.

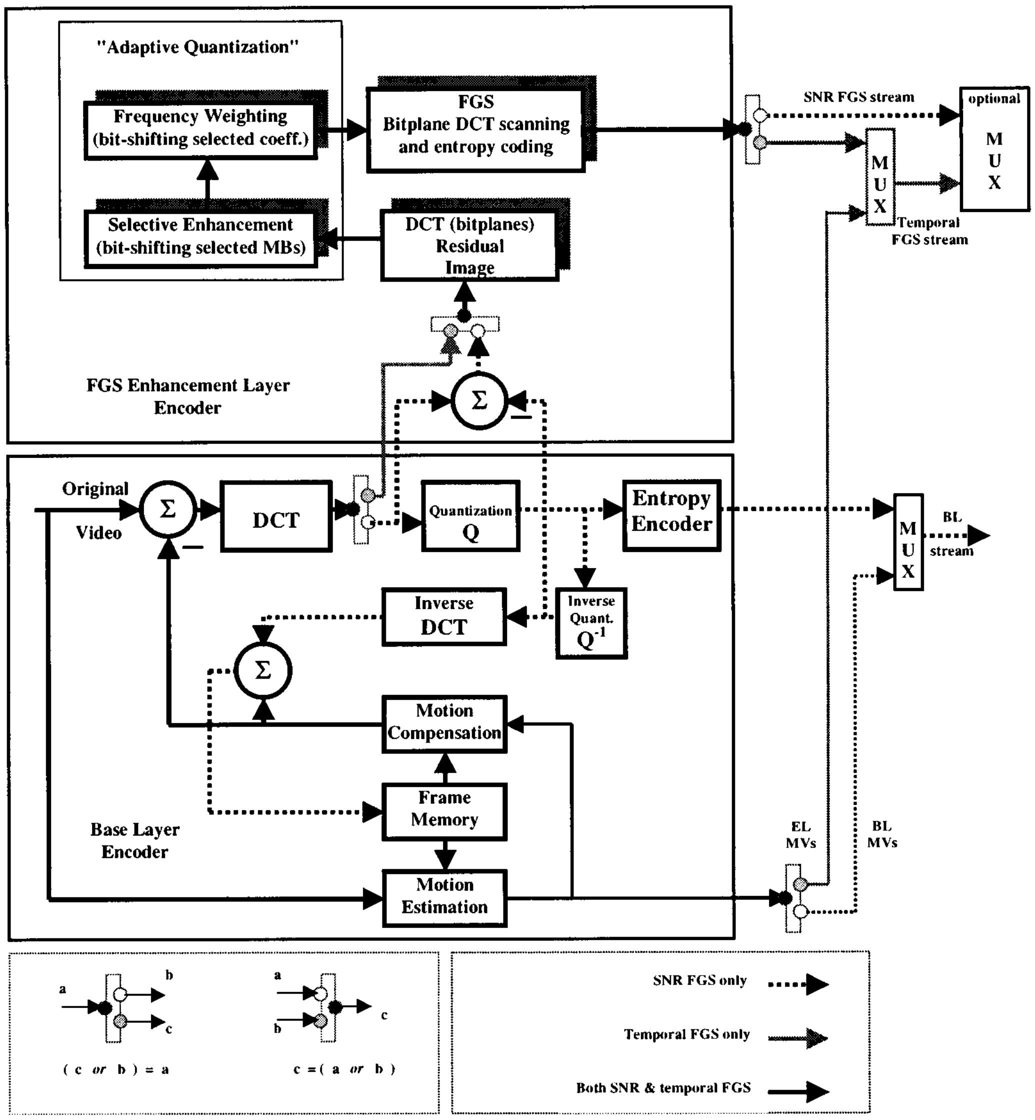

Fig. 14. Architecture for the all-FGS hybrid temporal-SNR scalability encoder. As shown in the figure, all of the new functions needed for the support of the temporal scalability feature can be shared by already existing functional blocks from the SNR FGS encoder (Fig. 3).

without an extra implementation complexity overhead. As shown in Fig. 14, the DCT, motion-estimation, motion-compensation, and frame-memory functional blocks from the base-layer encoder all can be utilized when computing the FGST DCT residual signal. This can be achieved through a novel (yet simple) data-flow control of the data within the FGS codec. What makes this sharing of resources feasible is the fact that the encoder never compresses a base-layer frame and an FGST frame at the same instance. Similarly, the SNR FGS entropy-encoder can be shared between the SNR FGS and FGST frames since both of these picture types are never compressed at the same instance of time.

As shown in Fig. 14, the motion estimator outputs two sets of motion vectors: one set for the base-layer pictures and the other for the FGST frames. The MV's associated with FGST frames are multiplexed with the enhancement layer bitstream using the data-partitioning strategy explained above. Moreover, the two FGS enhancement-layer streams can be either multiplexed to generate a single stream (which consists of both SNR and temporal FGS pictures) *or* stored/transmitted in two separate streams.

Fig. 15 shows the corresponding functional architecture for the hybrid temporal-SNR FGS decoder. Similar to the encoder architecture described above, the decoding of the FGST frames can be realized with minimal complexity overhead. This is accomplished by sharing the motion-compensation functional block with the base-layer and sharing the standard SNR FGS decoding path. It is worth pointing out that, at the receiver side, the inverse-DCT of the FGST residuals can be computed (at least from a functional perspective) using the inverse DCT block of either the enhancement-layer or the base-layer decoder. This is the case since it is possible that the base-layer IDCT block is not in use (i.e., by the base-layer) when the receiver is decoding an FGST frame. However, both IDCT blocks are in use when decoding an FGS SNR picture (one for computing the

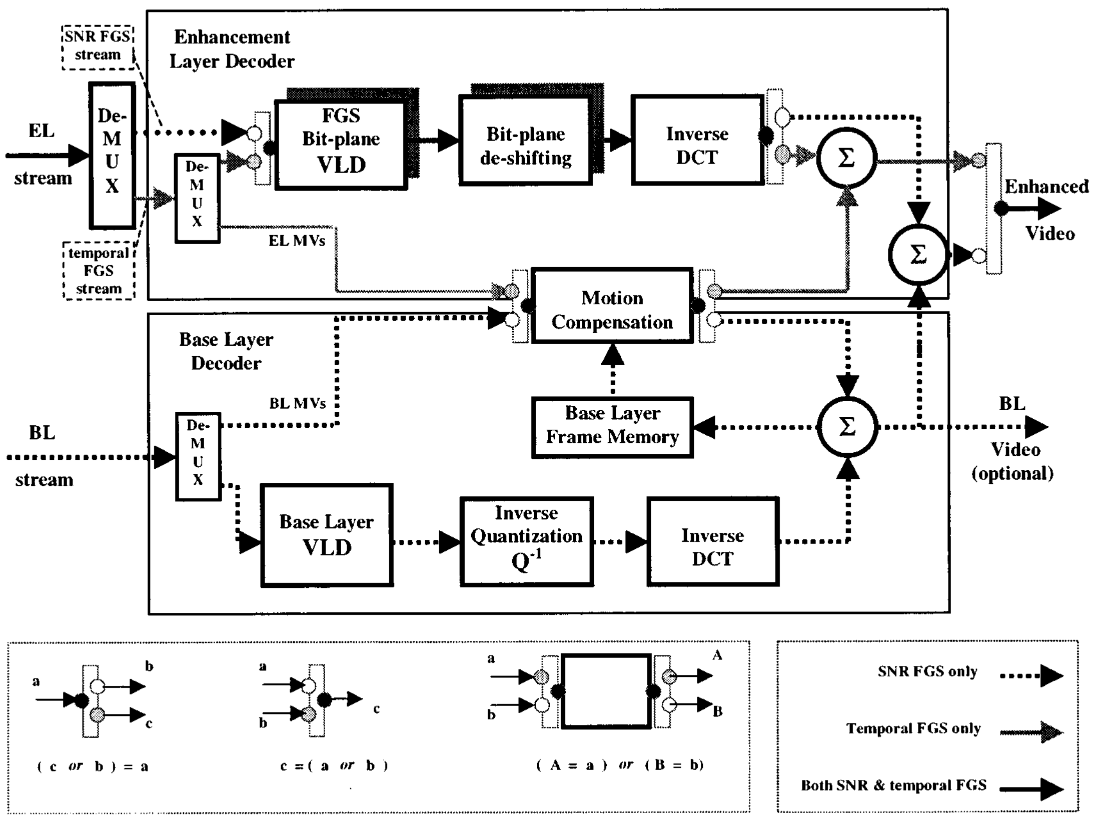

Fig. 15. Architecture for the all-FGS hybrid temporal-SNR scalability decoder. Similar to the encoder case (Fig. 14), all of the new functions needed for the support of the temporal scalability feature can be shared by already existing functional blocks from the SNR FGS decoder (Fig. 4).

IDCT of the FGS residual and the other for computing the IDCT of the corresponding base-layer frame).

As shown in Fig. 15, the FGST compressed stream is de-multiplexed to separate the MV's data from the coded residual information. The FGST MV's are used by the motion-compensation block to compute the FGST predicted frame while the compressed residual information is decoded and inverse transformed by the enhancement-layer decoder. The two signals are added together to generate the FGST frame which can be sent directly to the display device. For the SNR FGS compressed frames, the decoded signal has to be added to the corresponding base-layer frames before the display operation.

A. Performance Evaluation of the Hybrid FGS Scalability Method

To determine the efficiency of the FGST method, comparisons have been performed with a multilayer FGS-temporal scalability structure. This multilayer scheme is depicted in Fig. 16. In this scheme, the FGS layer is coded on top of both the base and temporal enhancement layers and therefore enhances the SNR quality of all frames. For simplicity, this implementation is referred to in the remainder of the paper as multiple-layer FGS-temporal scalability, since one base-layer and two-enhancement-layers are employed for its realization.

In the multilayer FGS-temporal scalability implementation depicted in Fig. 16, the bit-rate of the temporal scalability layer is predetermined at encoding time. Since the temporal enhancement layer is not fine-granular, it needs to be entirely decoded

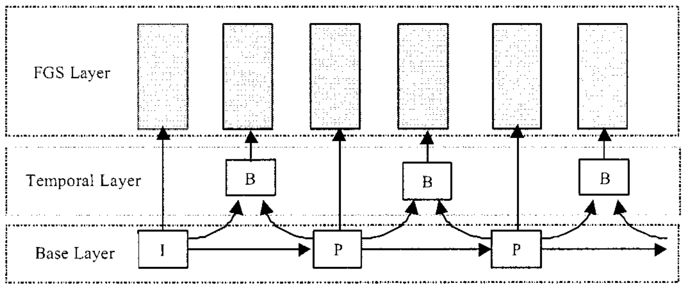

Fig. 16. Multilayer FGS-temporal scalability structure.

in order to improve the temporal resolution of the decoded sequence, requiring a discrete bit-rate of $R_{BL} + {R_{EL}}^{T}$. Another disadvantage of this solution resides in its increased implementation complexity, since two residuals need to be computed for the temporal-frames (MC and FGS-residuals) and two algorithms need to be employed for the decoding of the enhancement-layer texture.

The results of FGST and the multilayer FGS-temporal scalability are presented in Table II for the sequence *Foreman*. The experiments have been performed for a base-layer frame-rate f_{BL} of 5 fps and an enhancement-layer frame-rate f_{EL} of 5 fps. The base-layer contains Group-Of-VOP's (GOV's) with only I and P-frames ($M = 1$) which last for 2.4 s ($N = 12$) and employs TM5 for rate-control. For the coding of the temporal layer B-frames in the multiple-layer implementation, a fixed $QP = 28$ has been employed. To provide a fair comparison,

TABLE II
PERFORMANCE RESULTS FOR THE MULTI-LAYER FGS-TEMPORAL SCALABILITY AND ALL-FGS

R_{BL} (kb/s)	R_{EL}^{T} (kb/s)	R_{EL}^{FGS} (kb/s)	PSNR (Y/U/V) for multi-layer FGS temporal scalabilities - Figure 16)	PSNR (Y/U/V) for all-FGS (FGST and SNR FGS - Figure 13a)
100	40	0	30.57,37.07,37.68	30.54,37.02,37.59
100	40	100	32.94,38.28,39.27	32.95,38.28,39.27
100	40	200	34.73,39.57,40.73	34.74,39.57,40.74
100	40	300	36.01,40.41,41.40	36.02,40.41,41.41
100	40	400	37.54,41.32,42.62	37.55,41.32,42.63
100	40	500	38.75,42.29,43.60	38.75,42.29,43.61
100	40	600	39.53,42.99,43.99	39.54,43.00,44.00
100	40	700	40.44,43.67,44.55	40.45,43.67,44.57
100	40	800	41.51,44.30,45.24	41.53,44.31,46.26

the temporal/FGS (FGST) frames in the all-FGS implementation have been coded with the same amount of bits as employed for the B-frames in the temporal scalability layer of the multilayer scalability implementation. This bit-rate adjustment is easily performed due to the embedded-stream property of FGS. In Table II, R_{BL} represents the base-layer rate, R_{EL}^{T} represents the temporal-layer rate for the multiple-layer implementation and R_{EL}^{FGS} is the SNR FGS-layer bit-rate. For the FGST frames in the all-FGS implementation, the rate is $R_{EL}^{T} + R_{EL}^{\mathrm{FGS}}$. As can be seen from Table II, the rate-distortion performance of the previously two described implementations of hybrid SNR/temporal scalability is very similar. In other words, there is no penalty associated with the proposed single-layer scalability solution (i.e., FGST).

V. CONCLUSIONS AND RELATED WORK

In this paper, we provided a comprehensive description of the MPEG-4 FGS framework and its new video coding tools which have not been presented outside the MPEG-4 community. We highlighted the SNR FGS framework features, its ability in supporting unicast and multicast Internet video applications, and its basic coding tools. We also presented two important aspects of FGS: 1) "adaptive quantization" and its FGS related video-coding tools that have been adopted by MPEG-4, and 2) the FGS-based hybrid temporal-SNR scalability method which has also been adopted by the standard.

Besides the methods described in this paper, several other techniques have been recently proposed for the MPEG-4 FGS standardization. These algorithms are mainly targeted at improving the FGS performance both objectively (PSNR) and subjectively (visually).

The most promising approaches for improving FGS performance is based on supporting some level of prediction or motion compensation within the FGS enhancement layer [45], [50]. These methods have shown improvements of up to 2 dB in PSNR (when compared with the current FGS framework).

In [44], a method has been proposed that employs the information and coding parameters used for the base-layer to improve the compression efficiency of the FGS bit-plane coding scheme. For example, whenever MPEG-2 (type) quantization is used, the quantization step and the weighting matrix of the base-layer can be employed to predict the range of the residual DCT coefficients (i.e., maximum number of significant bit-planes for each residual coefficient, N_{BP}) and to avoid the unnecessary transmission of certain zero-valued bit-planes of the DCT.

Also, in the current FGS scheme, the transmission of the macroblocks is predetermined and fixed to the scanning order for all the bit-planes. However, within a bit-plane, not all the macroblocks are necessarily transmitted depending on the available channel capacity. To improve the coding efficiency, an algorithm has been proposed in [47] for reordering positions of the macroblocks in the enhancement layer such that the macroblocks with larger residue values are transmitted/coded first. Another proposal has been presented recently which is based on scanning the FGS bitplanes starting from a desired macroblock within the picture and then scanning the surrounding macroblocks in a "water ring" fashion [49].

Further, the combination of FGS with arbitrary shape objects has also been proposed [48]. An important benefit of this proposal is that the FGS rate-control algorithm can significantly benefit from knowing the shape and position of the various objects within the scene. For example, the adaptive quantization method described in Section III can be employed to enhance a particular object which is either visually important or is selected as significant by the user (e.g., the face of the person in the "foreman" sequence).

It is important to note, however, that although some of the above (proposed) improvements for FGS are being investigated, non of these techniques have been adopted (yet) by the standard. Therefore, for the latest supported video coding tools within FGS, we encourage the reader to acquire the final version of the MPEG-4 FGS standard which is anticipated to be completed before the end of year 2001.

ACKNOWLEDGMENT

The authors would like to thank all the people who contributed to the MPEG-4 FGS standardization effort. In particular, special thanks go to J. Ohm (HHI) for his effort in chairing the FGS work, to W. Li (Webcast), and F. Ling (GI) for their major contributions to the bitplane DCT coding method, and to H. Jiang (Intel) and B. Thebault (France Telecom) for their collaborations with the authors on the temporal FGS and Selective Enhancement cross-checking experiments, respectively. The authors would also like to thank C. Dufour (Philips Research—LEP) for her major contributions to the FGS evaluation study, and to S. Peng, D. Loguinov, M. Balakrishnan (Philip Research) for their constructive review of an earlier version of this paper. Last but least, the authors sincere thanks to four anonymous reviewers who provided excellent and thorough feedback which improved the quality of this paper, and to J. Liu for being the editor of the paper.

REFERENCES

[1] H. Radha and Y. Chen, "Fine-granular-scalable video for packet networks," in *Packet Video '99*. New York, NY, April 1999.

[2] Y. Chen, C. Dufour, H. Radha, R. A. Cohen, and M. Buteau, "Request for fine granular video scalability for media streaming applications," in *Contribution to 44th MPEG Meeting*, Dublin, Ireland, July 1998.

[3] H. Radha, Y. Chen, K. Parthasarathy, and R. Cohen, "Scalable internet video using MPEG-4," *Signal Process.: Image Commun.*, vol. 15, pp. 95–126, Sept. 1999.

[4] ISO/IEC, 14 496-2, "MPEG-4 Video FGS v. 4.0,", Noordwijkerhout, The Netherlands, Proposed Draft Amendment (PDAM), March 2000.

[5] ISO/IEC, 14 496-2, "FGS Amendment WD v. 1.0,", Vancouver, BC, Canada, 48th MPEG Meeting, July 1999.

[6] Y. Chen, M. van der Schaar, H. Radha, and B. Schuster, "Residual computation for fine granularity scalability," in *Contribution to 48th MPEG Meeting*, Vancouver, BC, Canada, July 1999, m4937.

[7] M. van der Schaar, Y. Chen, and H. Radha, "Adaptive Quantization Modes for Fine-Granular Scalability," in *Contribution to 48th MPEG Meeting*, Vancouver, BC, Canada, July 1999, m4938.

[8] ——, "An all FGS solution for hybrid Temporal-SNR scalability," in *Contribution to 50th MPEG Meeting*, CITY?, HI, USA, Dec. 1999, m5552.

[9] M. van der Schaar and H. Radha, "A novel MPEG-4 based hybrid temporal-SNR scalability for Internet video," in *Proc. ICIP 2000*, Vancouver, BC, Canada, Sept. 2000.

[10] H. Holbrook and D. Cheriton, "IP multicast channels: Express support for large-scale single-source applications," in *ACM SIGCOMM '99*.

[11] S. McCanne, "Enabling internet TV with intelligent network infrastructure," in *IP Multicast Summit 2000 (mcast2000)*, San Francisco, CA, Feb. 2000.

[12] H. Radha, "Fine granular scalability: A new framework for unicat and multicast video streaming over the internet," in *IP Multicast Summit 2000 (mcast2000)*, San Francisco, CA, Feb. 2000.

[13] ——, "Broadcast TV and the internet," in *IEEE Symp. Computers & Communications*, July 1999.

[14] W. ten Kate and H. Radha, "Bringing the web to the television: Convergence scenario," in *W3C Workshop on the Television and the Web*. France, June 1998.

[15] T. Pusateri, "Large scale multicast with PIM source-only," in *IP Multicast Summit 2000 (mcast2000)*, San Francisco, CA, Feb. 2000.

[16] M. Luby *et al.*, "Asynchronous layered coding a scalable reliable multicast protocol,", http://www.ietf.org/internet-drafts/draft-ietf-rmt-pi-alc-00.txt, Mar. 2000.

[17] M. Luby, "Reliable multicast transport building block: Forward error correction codes,", http://www.ietf.org/internet-drafts/draft-ietf-rmt-bb-fec-00.txt, Mar. 2000.

[18] W. Davis and J. W. Irza, "Cable modems take the early lead," *Byte*, vol. 23, no. 1, January 1998.

[19] S.-T. Hsiang and J. W. Woods, "Embedded video coding using invertible motion compensated 3-D subband/wavelet filter bank," in *Packet Video 2000*, Sardinia, Italy, May 2000.

[20] S.-J. Choi and J. W. Woods, "Motion-compensated 3-D subband coding of video," *IEEE Trans. Image Processing*, pp. 155–167, Feb. 1999.

[21] W. Tan and A. Zakhor, "Real-time internet video using error resilient scalable compression and TCP-friendly transport protocol," *IEEE Trans. Multimedia*, vol. 1, pp. 172–186, June 1999.

[22] Y. Chen, R. Cohen, H. Radha, C. Dufour, W. Li, A. Zakhor, S. Cheung, B. Schuster, and J. Liang, "Description of experiment on fine granular video scalability," in *Contribution to 45th MPEG Meeting*, Atlantic City, NJ, Oct. 1998, m3881.

[23] Y. Chen, "Summary of mini experiment on fine granular video scalability," in *Contribution to 45th MPEG Meeting*, Atlantic City, NJ, Oct. 1998, m3987.

[24] Y. Chen, H. Radha, and R. A. Cohen, "Philips research USA results of ME on fine granular video scalability," in *Contribution to 45th MPEG Meeting*, Atlantic City, NJ, Oct. 1998, m3988.

[25] Y. Chen, M. van der Schaar Mitrea, H. Radha, B. Schuster, J. Liang, and E. Francois, "Fine granular video scalability by combining video object and still texture coding," in *Contribution to 46th MPEG meeting*, Rome, Italy, December 1998, m4331.

[26] W. Li, "Bit-plane coding of DCT coefficients for fine granularity scalability," in *Contribution to 45th MPEG Meeting*, Atlantic City, NJ, Oct. 1998, m3989.

[27] F. Ling and X. Chen, "Report on fine granularity scalability using bitplane coding," in *Contribution to 46th MPEG Meeting*, Rome, Italy, Dec. 1998, m4311.

[28] W. Li *et al.*, "Experiment result on fine granular scalability," in *Contribution to 46th MPEG Meeting*, Seoul, Korea, Mar. 1999, m4473.

[29] S. Cheung and A. Zakhor, "Matching pursuit coding for fine granular video scalability," in *Contribution to 45th MPEG Meeting*, Atlantic City, NJ, Oct. 1998, m3991.

[30] M. Benetiere and C. Dufour, "Matching pursuits residual coding for video fine granular scalability," in *Contribution to 45th MPEG Meeting*, Atlantic City, NJ, Oct. 1998, m4008.

[31] B. Schuster, "Fine granular scalability with wavelets coding," in *Contribution to 45th MPEG Meeting*, Atlantic City, NJ, Oct. 1998, m4021.

[32] 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," in *Contribution to 45th MPEG Meeting*, Atlantic City, NJ, October 1998, m4025.

[33] J.-S. Shin, S.-H. Son, D.-S. Cho, and Y.- S. Seo, "Preliminary results on fine granular video scalability for arbitrary shaped object," in *Contribution to 45th MPEG Meeting*, Atlantic City, NJ, Oct. 1998, m4042.

[34] H. Jiang and G. M. Thayer, "Using frequency weighting in fine-granularity-scalability bit-plane coding for natural video," in *Contribution to 50th MPEG Meeting*, CITY?, HI, December 1999, m5489.

[35] W. Li, "Frequency weighting for FGS," in *Contribution to 50th MPEG Meeting*, CITY?, HI, Dec. 1999, m5589.

[36] S. McCanne, V. Jackobson, and M. Vetterli, "Receiver-driven Layered Multi-cast," in *Proc. SIGCOMM'96*, Stanford, CA, Aug. 1996, pp. 117–130.

[37] S. McCanne, M. Vetterli, and V. Jacobson, "Low-complexity video coding for receiver-driven layered multicast," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 983–1001, Aug. 1997.

[38] V. Paxson, "End-to-end internet packet dynamics," in *Proc. ACM SIGGCOM*, vol. 27, Oct. 1997, pp. 13–52.

[39] "Information technology—coding of audio-visual objects: Visual,", International Standard, ISO/IEC JTC1/SC29/WG11, ISO/IEC 14 496-2, March 2000.

[40] B. Girod *et al.*, "Packet loss resilient internet video streaming," in *VCIP'99, Proc. SPIE*, vol. 3653, Jan. 1999, pp. 833–844.

[41] G. Carle and E. W. Biersack, "Survey of error recovery techniques for IP-based audio-visual multicast applications," *IEEE Network*, vol. 11, no. 6, Nov./Dec. '97.

[42] M. van der Schaar, H. Radha, and C. Dufour, "Scalable MPEG-4 video coding with graceful packet-loss resilience over bandwidth-varying networks," in *Proc. ICME 2000*, New York, July 2000.

[43] M. van der Schaar, Y. Chen, and H. Radha, "Embedded DCT and wavelet methods for fine granular scalable video: Analysis and comparison," in *IVCP 2000, Proc. SPIE*, vol. 2974, Jan. 2000, pp. 643–653.

[44] ——, "Proposal for experiment on coefficient prediction for FGS enhancement layer coding," in *Contribution to 49th MPEG Meeting*, Melbourne, Australia, Oct. 1999, m5259.

[45] S. Li, F. Wu, and Y.-Q. Zhang, "Experimental results with progressive fine granularity scalable (PFGS) coding," in *Contribution to 51st MPEG Meeting*, Noordwijkerhout, The Netherlands, Mar. 2000, m5742.

[46] H. Jiang, "Experiment on post-clip FGS enhancement," in *Contribution to 51st MPEG Meeting*, Noordwijkerhout, The Netherlands, Mar. 2000, m5826.

[47] C. S. Lim and T. K. Tan, "Macroblock reordering for FGS," in *Contribution to 51st MPEG Meeting*, Noordwijkerhout, The Netherlands, Mar. 2000, m5759.

[48] ——, "FGS for arbitrary shaped objects," in *Contribution to 51st MPEG Meeting*, Noordwijkerhout, The Netherlands, Mar. 2000, m5760.
[49] G. H. Park *et al.*, "Water ring scan order for MPEG-4 fine granular scalable coding," in *Contribution to 52nd MPEG Meeting*, Beijing, China, July 2000.
[50] M. van der Schaar and H. Radha, "Motion compensation based fine granular scalability," in *Contribution to 54th MPEG Meeting*, France, Oct. 2000, m6475.

Hayder Radha (M'92) received the B.S. degree in electrical engineering (honors) from Michigan State University (MSU), East Lansing, in 1984, the M.S. degree in electrical engineering from Purdue University, West Lafayette, IN, in 1986, and the Ph.M. and Ph.D. degrees from the Center for Telecommunications Research (CTR), Columbia University, New York, in 1991 and 1993, respectively.

He joined the faculty at MSU in 2000 as an Associate Professor in the Department of Electrical and Computer Engineering. He is also a Philips Research Fellow, and an Adjunct Professor at the City University of New York (CUNY). He joined Philips Research, Briarcliff Manor, NY, in 1996 and worked in the area of video communications and high definition television. He initiated the internet video research program at Philips Research and led a team of researchers working on scalable video coding, networking, and streaming algorithms. Prior to joining Philips, he was a Distinguished Member of Technical Staff at Bell Laboratories. He started his career at Bell Labs in 1986, where he worked in the areas of digital communications, signal and image processing, and broadband multimedia communications. His research interests include image and video coding, multimedia communications and networking, and the transmission of multimedia data over wireless and packet networks. He has more than 20 patents in these areas (between granted and pending).

Dr. Radha served as a Co-Chair and an Editor of the ATM and LAN Video Coding Experts Group of the ITU-T between 1994–1996.

Mihaela van der Schaar graduated in electrical engineering from Eindhoven University of Technology, The Netherlands, in April 1996.

She then joined Philips Research Laboratories, Eindhoven, where she became a member of the TV Systems Department. From 1996 to 1998, she was involved in several projects which investigate low-cost very high quality video compression techniques and their implementation for TV, computer and camera systems. Since 1998, she is an expatriate at Philips Research, Briarcliff Manor, NY, where she is a member of the Video Communications Department involved in the research of video coding techniques for internet video streaming. Since 1999, she is an active participant to the MPEG-4 video standard, contributing to the "Fine Granularity Scalability" tool. Her research interests include video and graphics coding and video streaming over unreliable channels and she coauthored more than 20 research papers in this field and holds several patents.

Yingwei Chen received the B.E. degree from Tsinghua University, Beijing, China, in 1992 and the M.S. degree from Rensselaer Polytechnic Institute, Troy, NY, in 1995, both in electrical engineering.

Since 1996, she has been with Philips Research, Briarcliff Manor, NY, where she is engaged in research into video compression and processing for Internet streaming and various DTV applications. She initiated and participated actively in the standardization of Fine Granularity Scalability in MPEG-4.