Error-resilient Spectral Fine Granular Scalable (SFGS) Video Coding for Network Streaming Applications

Wen-Nung Lie\textsuperscript{1}, Cheng-Hsiung Tseng\textsuperscript{2}, and Ping-Chang Ju\textsuperscript{2}

\textsuperscript{1}Department of Electrical Engineering
National Chung Cheng University, Chia-Yi, 621, Taiwan, ROC.
E-mail: wnlie@ee.ccu.edu.tw

\textsuperscript{2}Materials & Electro-Optics Research Division, Chung-Shan Institute of Science & Technology
Lung-Tan, Tao-Yuan 325, Taiwan, ROC.

ABSTRACT
As networking grows, video transmission becomes more popular. MPEG4 FGS is a video coding technique designed to cope with variability of bandwidth gracefully. Our previously proposed Spectral FGS (SFGS) coding technique, a variation of traditional FGS, when in combination with a rate-distortion-based rate allocation scheme, was capable of making video quality smoother between pictures in a sequence. However, both FGS and SFGS-coded video streams, when transmitted through networks, may be partially corrupted by channel errors or packet loss. To resist errors, we add sync codes according to the spectrum-grouping property of SFGS to enhance error resilience capability. This scheme of adding sync codes is simple, does not cause heavy loads of the system, and can on average improve video quality by 1–3 dB when errors are present.

I. INTRODUCTION
As the grows rapidly, video transmission applications become popular. MPEG4 FGS [1] is a video coding technique that is capable of coping with variability of bandwidth gracefully. Here, we adopted a variation of FGS, called Spectral FGS (SFGS) [2], to achieve the same purpose. SFGS groups data in each bit-plane of the enhancement layer by the ordering of spectral frequencies before they are fed into the traditional FGS bit-plane coding modular. SFGS has the advantage of evener intra image quality than traditional FGS. With rate allocation [2] based upon rate-distortion indices of each spectral band for each frame, SFGS coding technique is also capable of achieving smoother quality between consecutive frames (i.e., less variation for inter image quality).

When video bit streams are transmitted in the network, they are possibly incurred channel errors or packet losses, leading to a severe degradation in video quality. To protect video data from errors, error resilience mechanisms are often offered for the operation between the encoder and decoder. A number of literature has been emphasized on the error resilience/concealment of the base layer data. The enhancement layer, however, needs a simpler scheme with less overhead and complexity, due to their less contributions to image quality.

In [7], a strategy of adding asymmetrical row and column channel codes (in terms of the well-known Reed-Solomon codes) to protect MPEG4 FGS video data from bit error or packet loss was adopted. The row channel codes are used for resisting bit errors and the column channel codes are for packet losses. The higher bit-planes are generally equipped with more channel codes. In [8], a rate-distortion (R-D) optimization framework was developed to optimally partition the FGS enhancement-layer bit stream into blocks and then a degressive error protection algorithm was applied to blocks by using the RS codes. That is, the higher bit-plane or the priority-encoded bit stream will be protected with more FEC codes.

The above two schemes both utilize RS codes to protect the FGS enhancement layer data. The amount of RS codes added depends on the significance of the data. According to experiences, a full transmission of the enhancement layer can only improve image quality by around 3–5 dB. Hence, a heavy overhead of FEC codes does not necessarily lead to the largest coding efficiency in channel error conditions. Furthermore, the error resilient scheme should be simple so as not to stress too many loads on the FGS decoder.

In these viewpoints, we adopt the insertion of synchronization (sync.) codes to avoid error propagation, rather than adding FEC codes for error recovery. Our method can be integrated with that in [7], that is, sync codes are used to protect data from bit errors, while column channel codes are used to recover from packet loss. Our emphasis is designing a way to insert cost-effective sync codes according to the contribution and property of each bit-plane data. Besides, the adoption of SFGS coding method makes the above proposed error resilience scheme more efficient.

II. REVIEW OF SFGS VIDEO CODING
In the original FGS scheme, each bit-plane of the enhancement layer is coded by scanning macroblocks (MB) in the top-to-bottom and left-to-right order. Bits of each MB are then run-length-encoded. When the available bandwidth is not enough for transmitting a whole bit-plane data, the bitstream is hence truncated. In this way, the video quality of upper-left subimage will be better than that of the lower-right part. This effect of space-varying quality is especially obvious when the bitstream is truncated at the more significant bit-plane.

To overcome this situation, we proposed a new scheme which scans bit-plane data in the order of DCT frequencies [2] (including 1 DC and 63 AC components in the zigzag order), as shown in Fig.1. In each spectral band, the bits are grouped according to Y, U, and V components, while in each component, the MBs are scanned in the raster manner. The main concept is that perceptually important frequencies (often lower frequencies) are encoded and transmitted first. The frequency-rearranged bit-plane data are then treated much like the manner in traditional FGS coding scheme, hence so-called Spectral FGS (SFGS). SFGS changes only the bit-scanning orders in each bit-plane without modifying the coding structure of FGS, so it can also keep the ability of adapting to bandwidth variation.

SFGS coding method has the advantage of evener intra-frame quality when the whole bit-plane is truncated due to insufficient channel bandwidth. It eliminates the disadvantage of traditional FGS and other variations that macroblocks (MBs) in the top-left area will always has a better PSNR than those in the lower-right part, especially at lower bit-rates.
III. BASIC SCHEME OF INSERTING SYNCHRONIZATION CODES IN EACH BIT-PLANE DATA

For scalable video coding, the amount of data in the base layer part is much less than that in the enhancement layer part. We assume that the base layer part is error-free and only the enhancement layer part is considered for the sync code insertion.

Considering the characteristics of SFGS coding, each spectral band, which contains four Y and one U and V components, is associated with one sync code whose length is 40 bits long, as shown in Fig.2. The first 32 bits of each sync code is the start-code prefix in the enhancement layer, and the other 8 bits are used to record the spectral number of this band (from 1 to 64 only). Without the insertion of sync codes, the errors will propagate until next start-code, i.e., the start of the next frame, when channel errors are present. Clearly, the data between the error point and the start of next frame will be discarded, leading to a significant reduction of image quality.

Since a sync code is inserted in front of each spectral band data in each bit-plane, the discarded data will include at most one spectral band in between two sync codes.

At the decoder side, errors are detected via the following possible ways:
1. The decoder encounters a variable-length-decoding (VLD) failure.
2. The segment length after run-length decoding exceeds the standard size.
3. A start code is encountered before the decoding process is completed.

When one of the above three conditions is encountered in decoding the spectral band n, the decoder will skip the data and continue to decode the data of band n+1 by finding the next sync code.

IV. OTHER VARIATIONS ACCORDING TO PROPERTIES OF SPECTRAL BANDS

The basic scheme in Section III does not consider the efficiency of the inserted sync codes. For examples, each sync code guards a different amount of bit stream (a spectral band in a higher bit-plane will normally contain a less amount of bits) and each part of bit stream may have different PSNR contributions even they are of the same length. That is, sync codes inserted at different spectral bands or at different bit-planes may have significantly distinctive efficiency in data protection. To be more efficient, the manner of sync code insertion is investigated in the following.

Hence, we propose to insert sync codes according to the characteristics of spectral bands and bit-planes to get the best error protection capability with the least overheads. First, the MSB is much more important than the LSB; it will be worthy of high error protection. Second, the low frequency band will contribute more in image reconstruction and is thus also worthy of high error protection.

According to the above observations, we designed several schemes for inserting sync codes, as depicted in Table I and Fig.3. Scheme 0 does not insert any sync codes, i.e., error resilience mechanism is not enabled. Scheme 1 inserts one sync code before each spectral band. Scheme 2 inserts a different number of sync codes for each bit-plane, i.e., adding more sync codes in MSB than in LSB. In this scheme, bit-plane 0 (MSB) is enhanced with one sync code for one band and bit-plane 1 with one sync code for two bands. The spectral spacing between two consecutive sync codes becomes larger for lower bit-planes. Scheme 3 inserts sync codes according to the spectral frequency. A threshold band is defined to determine the enabling of adding the sync codes. For spectral bands lower than the threshold band, one sync code is inserted for error protection; for those higher than the threshold band, no sync codes are added. In the following experiments, the threshold band was set to AC-47. The last scheme, Scheme 4, mixes Scheme 2 and 3 to protect data.

Table I Different schemes of inserting sync codes for error protection in SFGS video transmission

V. EXPERIMENTAL RESULTS

To evaluate the basic error protection capability of the proposed scheme for SFGS video coding, experiments were conducted based on two standard CIF sequences: “Foreman” and “Mobile”. The encoding frame rate is fixed at 10 fps. For each sequence, only the first frame is coded as I-frame and all the others are coded as P-frames. The bit rate of the base-layer bit stream is 128 kbps with TM 5 rate control. The streaming server can truncate the enhancement layer bit stream to fit the varying channel capacity of network. In the following experiments, we truncate the enhancement layer bit stream to...
128 kbps, 256 kbps, ..., until 1024 kbps with a step of 128 kbps. We basically assume an error-free base-layer data and randomly add error bits (Bit error rate, BER = 0.01, 0.001 and 0.0001) in the enhancement layer to test the capability of our proposed error resilience mechanism.

Table II. The PSNR improvements relative to the original SFGS without error resilience design at various BER conditions enhancement layer bit-rates (the base layer bit-rate is 128 kbps)

<table>
<thead>
<tr>
<th>BER</th>
<th>Forman EL (bps)</th>
<th>Mobile EL (bps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>128 k 512 k 1024 k</td>
<td>128 k 512 k 1024 k</td>
</tr>
<tr>
<td>0.001</td>
<td>0.36 1.17 1.76 0.29 1.03 1.52</td>
<td></td>
</tr>
<tr>
<td>0.0001</td>
<td>0.09 1.53 3.29 0.1 1.67 3.60</td>
<td></td>
</tr>
<tr>
<td>Error-free</td>
<td>-0.20 -0.25 -0.27 -0.13 -0.21 -0.27</td>
<td></td>
</tr>
</tbody>
</table>

From Table II, we found that at low BER (0.001 and 0.0001) and high bit rate (512 or 1024 kbps), our error resilience design can effectively prevent error propagation and increase video quality by about 1~3 dB. At a higher BER (e.g., 0.01), the number of attacked spectral bands is large, leading to a vain error protection.

We also calculate the overhead percentages at different bit-rates as follows.

\[
\text{overhead\_percentage} = \frac{\text{bits}_{\text{sync}}}{\text{bits}_{\text{EL}}} \times 100\% \quad (1)
\]

In Table III, the overhead percentage of the sync codes becomes smaller when the bit rate of the enhancement layer rises. At enhancement layer bit-rates of 512 and 1024 kbps, the overheads is less than 10%, but at the lower bit-rates (e.g., 128 kbps), it is much more.

In summary, in views of overhead percentage and PSNR improvement, sync codes had better be inserted only when video data are transmitted at high bit-rates under low BER conditions.

Table III. Overhead percentages of the sync codes at the base layer bit-rate of 128 kbps

<table>
<thead>
<tr>
<th>Overhead (%)</th>
<th>Forman</th>
<th>Mobile</th>
</tr>
</thead>
<tbody>
<tr>
<td>128 kbps</td>
<td>20.29%</td>
<td>15.81%</td>
</tr>
<tr>
<td>512 kbps</td>
<td>8.76%</td>
<td>5.80%</td>
</tr>
<tr>
<td>1024 kbps</td>
<td>5.96%</td>
<td>4.38%</td>
</tr>
</tbody>
</table>

We also conducted experiments to evaluate the performances of various schemes proposed in Table I. Most of the encoding parameters are similar to those in the prior basic evaluation, except that the encoding frame rate is 20 fps, due to the adoption of SFGST (Temporal SFGS, like the FGST proposed in [3]) and the base-layer-bit-rate is 256 kbps.

From Table IV, we found that at high bit rates (512, 1024 kbps), Scheme 3 has the best error resilience performance; at low bit rates (256 kbps), Scheme 3 and Scheme 4 are better than others. We average the resulting \(\Delta\text{PSNR}\) at all BERs (assuming equal probabilities) and find that Scheme 3 outperforms others.

It seems reasonable to insert sync codes according to the significance of bit-planes, as proposed in Scheme 2. However, the less amount of data in LSB does not lead to high contribution to PSNR. On the contrary, LSB has a larger PSNR contribution in spite of its low significance. Hence, our strategy is to provide equal error protection (i.e., same amount of sync codes) to each bit-plane, but different protection to spectral bands in a bit-plane by adjusting the threshold band.

Now we try to investigate the optimal threshold band for better error resilience and coding efficiency. We made experiments for three sequences: “Flower”, “Foreman” and “Mobile”. The threshold band is set to be 0, 4, ….., or 64 with a spacing of 4.

Table IV. The PSNR difference (dB) between Scheme 1 and Scheme 0 for the “Foreman” sequence coded at different enhancement layer (EL) bit-rates and BECs. A positive value represents a better error resilience.

For the “Flower” sequence (Fig.4), it is found that the error resilience capability is often monotonically increasing (higher PSNR) when the threshold band increases, especially at high bit-rates. At low bit-rates (e.g., EL=256 kbps), the error resilience capability first increases and then slightly decreases when the threshold band increases. That is, the PSNR is optimized at an optimal threshold band, denoted as \(t_{\text{optt}}\). We try to estimate \(t_{\text{optt}}\) by analyzing the PSNR contribution \(\Delta\text{PSNR}\) of each spectral band. Fig.4(b)(d)(f) show the \(\Delta\text{PSNR}\) of each spectral band at different EL bit-rates. Generally, \(\Delta\text{PSNR}\) is small for high spectral bands, meaning that we do not need to spend too many overheads to protect data of high bands. The accumulation of \(\Delta\text{PSNR}\) from band-0 to band-n can be easily obtained in the process of SFGS video coding [2] and used for good decision. For example, \(t_{\text{optt}}\) can be estimated as the band that accumulates 90% of \(\Delta\text{PSNR}\) contributions. Only for bands lower than \(t_{\text{optt}}\), sync codes are inserted. Table V shows the deviation of the estimated \(\hat{t}_{\text{optt}}\) from the true \(t_{\text{optt}}\). The resulting PSNR difference for \(t_{\text{optt}}\) and \(\hat{t}_{\text{optt}}\) is actually ignorable (0.003~0.05dB), as illustrated in Table V. For “Foreman” and “Mobile” sequences, the performances are similar to “Flower” and will not be shown here.

VI. CONCLUSIONS

From the experimental results, it was found that adding sync codes by thresholding the spectral bands is the most effective among several designs. We utilize the distribution of PSNR contribution \(\Delta\text{PSNR}\) of each spectral band to estimate the optimal threshold band. The experimental results proved the feasibility of our proposed method. However, the optimal threshold band is estimated based on a bit stream that is not added with sync codes. This theoretically has a slight deviation with that can be obtained when the bit stream is added with sync codes.
codes and rate-controlled. To take the rate control into account, the processing architecture in Fig.5 was adopted. That is, after SFGS video encoding without sync codes, the bit stream is first roughly truncated after rate control process, the optimal threshold band is then estimated and the sync codes are added accordingly, and finally the bit stream (with syncs) is truncated to fit the channel bandwidth. The advantage of this architecture is its simplicity of the server. Surely, this will sacrifice some advantages of the rate control process (e.g., video smoothness in its simplicity of the server. Surely, this will sacrifice some advantages of the rate control process (e.g., video smoothness in [2]). However, this makes a tradeoff between video smoothness and error resilience. 

The proposed error resilience scheme by inserting sync codes in front of each spectral band actually takes advantage of the characteristic of SFGS video coding. Since SFGS groups bits of the same spectral band from all the MB together, lose of one spectral band data leads to an even quality degradation in the whole frame. However, for the traditional FGS or other variations, similar sync codes to protect groups of MBs might result in an uneven intra-frame quality, thus undervaluing the inserted sync codes.

Table V Flower sequence: the selection of threshold band.

<table>
<thead>
<tr>
<th>Threshold band (d)</th>
<th>I_s</th>
<th>I_e</th>
<th>∆PSNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>128 kbps</td>
<td>24</td>
<td>36</td>
<td>0.003</td>
</tr>
<tr>
<td>512 kbps</td>
<td>56</td>
<td>45</td>
<td>0.029</td>
</tr>
<tr>
<td>1024 kbps</td>
<td>56</td>
<td>43</td>
<td>0.050</td>
</tr>
</tbody>
</table>

Fig. 5 The processing architecture of adding the sync codes by considering the rate control process.

REFERENCES