EFFICIENT FINE GRANULAR SCALABLE VIDEO CODING

*Christian Buchner, Thomas Stockhammer**

Institute for Communications Eng. (LNT)
Munich University of Technology (TUM)
80290 Munich, Germany

Detlev Marpe, Gabi Blättermann, Guido Heising†

Image Processing Department
Heinrich-Hertz-Institute (HHI)
10587 Berlin, Germany

ABSTRACT

In this work we present an efficient fine granular scalable video compression scheme which supports a fast bit rate adaptation independent of the encoder. The proposed scheme generates an embedded bitstream for each frame or, by appropriate multiplexing, for each group of picture. This rate-scalability is supported by an embedded bitstream which allows decoding at multiple rates, or to be more specific at virtually any rate. Drift removing techniques based on intra refresh or feedback mechanisms are presented. We show the potential of this video codec for variable bit rate channels as well as in combination with an unequal erasure protection scheme for error robust and efficient transmission over packet erasure channels. Possible enhancements of the presented coding schemes are discussed.

1. INTRODUCTION

Scalable video coding has attained great interest recently because of its inherent network friendliness. It is very suited for networking video applications such as video streaming. Scalable coder encode video input into multiple layers. The base layer should be transmitted with very reliability. However, the enhancement layers might be dropped or only transmitted according to the available network bitrate. In addition to conventional scalable coders fine granular scalability (FGS) is even more suited as virtually each bit forms an additional enhancement layer. This allows very fast and accurate network adaptation to variable bit rate channels. Rate-scalable, embedded video coding was first proposed by Taubman and Zakhor [1] using 3-D subband coding. Based on this groundbreaking work, a number of embedded 3-D video coding algorithms such as in [2, 3] were proposed which combine 3-D subband coding with motion compensation. In order to meet more restrictive requirements with respect to delay, implementation memory and computational complexity, McCanne et al. [4] introduced a simple progressive video coding algorithm. An overview of the FGS approaches in the MPEG-4 video standard is provided in [5]. The combination of scalable coding in combination with unequal erasure protection to combat Internet packet losses has recently been studied in [6], [7] and [8] where up to 5 dB gain in PSNR for transmission over packet lossy networks compared to standard anchor streams are reported.

However, especially fine granular scalability generally suffers from reduced coding performance in contrast to single layer approaches when operating at the same bit rate. In this paper we will introduce an efficient fine granular scalable coding system based on progressive texture video coding (PTVC) [9]. We will show the applicability of this system in several network environments and discuss the problem of drift removal in more detail. Starting with a brief description on PTVC in section 2, we continue to highlight FGS features of the PTVC and appropriate drift removing techniques in section 3. The performance of the codec in different network environments is presented in section 4.

* e-mail: {buchner, tom}@lnt.ei.tum.de, Tel: +49 89 28923474
† e-mail: {marpe,blaetter,heising}@hhi.de, Tel.: +49 30 31002619.

2. PROGRESSIVE TEXTURE VIDEO CODING

2.1. Overview

The PTVC is based on the H.26L test model [10] which has been modified to perform motion estimation and compensation only, i.e. coding of transform coefficients has been disabled in the H.26L codec. Instead, we use an embedded bitplane coding to represent I-frames and the displaced frame difference (DFD) resulting from motion compensation. All features of the encoder are shown in figure 1. The resulting video bitstream consists of the combination of the H.26L control and motion component and the progressive texture bitstream. We only give a brief overview over the codec. For a detailed description of the PTVC we refer to [11] and [9].

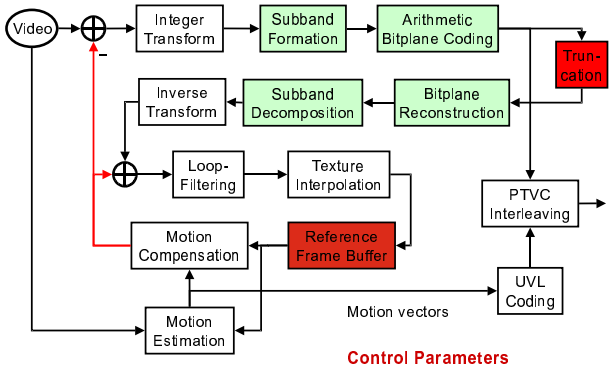

Fig. 1. Architecture of the Progressive Texture Video Codec

The proposed system uses the motion estimation and compensation apparatus of the H.26L codec TML5.0 [10]. H.26L includes a block based motion compensation with variable vector block sizes for each macroblock and 1/4 Pel estimation accuracy. Additionally, multiple reference frames can be used in motion compensation. Greater efficiency in intra-frame coding compared to previous approaches is achieved in the H.26L design using directional prediction in the spatial domain rather than coefficient value prediction in the transform domain. For more details we refer to

the test model description [10]. In the following we do not distinguish between I-frame prediction information and motion vector prediction as both methods result in a DFD which is processed identically in the progressive texture coding. We refer to prediction information as control and motion vector data.

2.2. Progressive Texture Coding

We will briefly discuss the coding of the texture as this feature adds the progressive functionality and therefore extends the H.26L test model. The identical 4×4 block based integer transform as used in the H.26L test model [10] is applied in our embedded texture coder. However, before quantization and entropy coding, the coefficients of the entire frame are rearranged into 16 subbands, such that, for example, all DC coefficients are arranged in the upper left subband as shown in figure 2. The entropy coding applied to the DFD is similar to techniques used in SPIHT [12] and JPEG-2000 [13] based on bitplane and context-based arithmetic coding. Therefore, it allows truncating of the bitstream and reconstruction with reduced quality. For further detail on the entropy coding we again refer to [11] and [9].

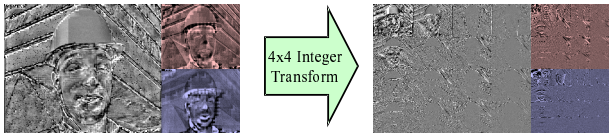

Fig. 2. Conversion of spatially local block based representation of texture into a spatially global subband oriented representation

2.3. Bitstream Properties and Control Parameters

According to figure 1 and the previous description we have specified the bitstream of the motion data and the DFD of each color component. The embedded bitstream for each frame is generated by multiplexing the motion information and the color components of the frame. The arithmetic coder independently operates on each color component (Y, U and V), generating three progressive color component bitstreams for each video frame. According to the weight factor for the color components R_c the texture part for each frame is generated in an embedded way. We take the first R_c bytes from the progressively coded Y-component and place them in the beginning of the message. Then we take the first byte of the U-component followed by the first byte of the V-component and place them sequentially into the message. Then we continue with the next R_c bytes from the Y-component and so on until all symbols from all color components within the frame are distributed.

Figure 3 illustrates the bitstream of one frame. The entire control and motion information is put in the beginning of each frame. Then, the progressively coded and interleaved texture information is appended. According to figure 1 the bitrate for each frame can be controlled very easily by truncating the frame at a certain number of bits. This number of bits is specified by the feedback bitrate r_f. The reconstructed frame is fed back into the prediction loop. Additionally, the information which is not fed back in the prediction loop results in an intra-coded enhancement (see section 3.1). Other means to control quality and data rate are an I/P-frame switch and a method to choose the frames in the reference buffer. This issue will be discussed in section 3.3.

3. FINE GRANULAR SCALABILITY

3.1. Scalable Bitstream Properties

According to figure 3 we obtain three parts of the bitstream for each frame. The control and motion vector data is placed in the beginning followed by the entire progressively coded texture information. However, as parts of the texture information is fed back to the motion compensation loop we can distinguish between a base layer and an intra coded enhancement layer. Denote that the texture is ordered in both layers in such a way that we obtain a fine granular SNR scalability.

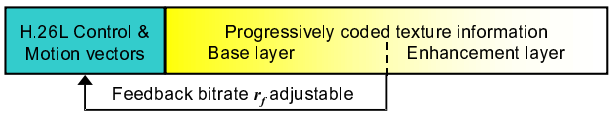

Fig. 3. Bit ordering for one frame

We briefly discuss the consequences removing parts of the bitstream in scalable transmission. Assume that the actual transmission bit rate r_t varies around the feedback bitrate r_f and buffering is not applied. Therefore, if $r_t \geq r_f$, we can improve the quality for this frame by transmitting some parts of the enhancement layer. In contrast, if the transmission bitrate is below the feedback bitrate, i.e. $r_t < r_f$, the later parts of the base layer texture according to figure 3 are removed. Therefore, the quality of the current frame is reduced in a graceful way as we remove parts of the progressively coded texture. Hence, the base layer coding in PTVC can be regarded as an advanced data partitioning scheme where each bit is a separate partition. However, this loss is much more severe for the performance of the codec as this results in a drift. Encoder and decoder reference frames are different in the motion compensation process. In general, this leads to a significant performance reduction [9]. Therefore, we will discuss means to generate independently decodable messages and methods to reduce or completely remove drift effects.

3.2. Independent Decoding of FGS Coded Video Messages

The loss of prediction information results in a mismatch between encoder and decoder reference frames for the PTVC as for most hybrid video codecs. Therefore, we introduce the concept of independently decodable group of pictures (GOP) also referred to as message in the following. Each GOP starts with a non-predictively coded intra frame or with a frame which is encoded to ensure that the drift between encoder and decoder is removed. Note that the first image within a GOP gets assigned more data rate than the residual image. The ratio is denoted as R_{IP}.

The progressively coded frame messages within one GOP are multiplexed in such a way that the most important information precedes less important information. To achieve this, the bitstreams of the individual frames of the entire GOP are interleaved to a progressive bitstream for the entire GOP similar to the procedure of color component multiplexing based on R_{IP} instead of R_c. As the feedback bitrate for each frame is adjusted to the R_{IP} within a GOP, the number of bits of the first frame fed back in the prediction loop is R_{IP} times higher than the bits fed back for the residual frames. Therefore, with this multiplexing and feedback bitrate adaptation the bit position $b_f = r_f R_{IP} / f_r$ within the GOP message specifies the boundary between base and enhancement layer. The number of transmitted bits for the GOP is specified

by $b_t = r_t R_{IP}/f_r$. If $b_t \geq b_f$ no drift within the GOP occurs whereas $b_t \geq b_f$ in general results in a drift problem within the GOP.

3.3. Feedback Based Drift Removing Methods

In [9] drift compensation by introducing regular I-frames has been introduced. However, as coding of I-frames requires in general more data rate than predictively coded frames, the overall performance of the codec decreases. A loss of about 2 dB in PSNR can be observed in figure 5 if the I-frame period is reduced from 100 to 10.

In general, in a variable bit rate transmission environment, not the entire information for one GOP will be lost and therefore, the encoder can usually rely on the decoder having decoded parts of the previous message. Figure 4 shows the framework of the investigated system. Assume that the encoder codes the video with a feedback bitrate r_f. The message is transmitted over a variable bitrate (VBR) channel with actual bitrate r_t and at the decoder this message is decoded at the transmission bitrate r_t. Additionally, we assume that there is an estimation r_e of the transmission bitrate r_t of this message available at the encoder site before encoding the next GOP. This information might be accessible by network feedback or statistical estimations. At the encoder site we can now produce reference frames by decoding the sent message at bitrate r_e, or if $r_e > r_f$, at the video feedback bitrate r_f. These reference frames are used by the encoder to predict the new message. The prediction bitrate r_e for each message is assumed to be known to both, the encoder and decoder by some small side information transmitted in the video bitstream. Denote that for $r_e = 0$ each message is decoded independently, I-frames are used in the beginning of each GOP, whereas $r_e = r_f$ results in the case where regular P-frames are transmitted. For $0 < r_e < r_f$ we obtain frames, which are predicted from drift prone reference frames.

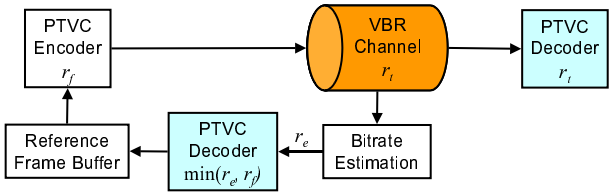

Fig. 4. Feedback based drift removing system

Note that if the estimated bitrate is below the transmission bitrate for all transmitted messages, i.e. $r_e \leq r_t$, drift effects occurring within one GOP are always removed by this scheme. However, if there exists just one message with $r_e > r_t$, the reference frames in encoder and decoder are different. This drift can be removed completely by a very sporadic introduction of I-frames, i.e. $r_e = 0$.

4. PTVC IN NETWORK ENVIRONMENTS

4.1. Rate-Distortion Performance

We compare the performance of the PTVC scheme to two non-scalable conventional hybrid coders for the case of *a priori* known transmission bitrate r_t by adjusting the feedback bitrate $r_f = r_t$. All simulations are carried out using the QCIF test sequence *Foreman* (30 Hz, 300 frames, 176×144 pels) at a constant frame rate $f_r = 10$ Hz, 5 reference frames have been used. As reference systems we used TML5 [10] of H.26L and TMN9 of the ITU-T Rec. H.263+, the latter with advanced options of Annexes D, F, I, J and T. To achieve a given target rate, a simple off-line rate control mechanism was used for both reference schemes, whereas for PTVC the rate control based on bitwise texture truncating was employed with $R_{IP} = 6:1$ and $R_c = 10:1:1$. These values are similar to the one of the reference codecs. Figure 5 shows the results of our experiments using two different I-frame periods $P_I = 10, 100$. As can be seen from the graph, our new texture coding is only little inferior to the coding method of H.26L, at least in the case where only one I-frame ($P_I = 100$) for the whole sequence is coded. Due to an efficient but inherently non-scalable spatial prediction scheme as part of its I-frame coding method [10], the H.26L test model provides a more distinctive gain when more I-frames are inserted ($P_I = 10$). However, even in this scenario, PTVC has a rate-distortion performance similar to or better than H.263+ with advanced coding options.

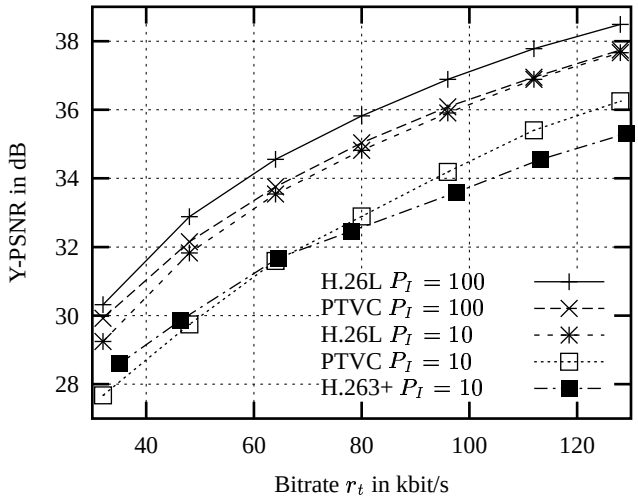

Fig. 5. Performance of PTVC for $r_f = r_t$ compared to standard test model implementations.

4.2. Varying or Unknown Bitrate Scenarios

In addition to the good R-D performance the PTVC shows further advantages in varying and unknown bitrate environments. Due to the flexibility of the codec we highlight only a few selected scenarios. Assume that we want to transmit a sequence not knowing the transmission bitrate r_t in advance. Therefore, neither the rate in a regular video codec nor the feedback bitrate r_f for the PTVC can be adjusted properly. However, due to the progressive texture coding a reduced transmission rate can be compensated by transmitting just the first part of the message. If $r_t \geq r_f$ we drop parts of the intra-coded enhancement layer similar to the MPEG-4 FGS approach whereas for $r_t < r_f$ base layer texture is dropped which results in drift problems.

Performance simulations of the PTVC for different scenarios have been carried out. The parameters (sequence, resolution, frame rate, etc.) are equivalent to the one presented in subsection 4.1. The transmission bitrate r_t is varied between 32 and 128 kbit/s and it is assumed to be fixed but unknown such that each transmission packet is truncated at bit position r_t/f_r. The

ratio between the size of the first frame of the GOP and residual frames R_{IP} was selected appropriately. Additionally, we assume that there is a drift removing strategy every 1 sec, i.e. $P_I = 10$. The results of different experiments are shown in figure 6. For comparison purpose, curve 1 shows the performance if regular intra updates are introduces and the transmission rate is known at the encoder in advance such that the feedback rate can be adjusted, i.e. $r_f = r_t$ and $r_e = 0$. Curve 2 and 4 show the performance having a base layer with bitrate $r_f = 32$ kbit/s and a pure intra coded enhancement layer. For 2 a regular I-frame update is introduced, i.e. $r_f = 32$ kbit/s and $r_e = 0$ and for 4 only P-frames are transmitted in the base layer, i.e. $r_f = r_e = 32$ kbit/s. A very slow increase of quality with increasing bitrate is visible. Similar performance is reported by the MPEG-4 FGS approach [5]. Curve 3 and 5 show the performance at a base layer bitrate $r_f = 128$ kbit/s and, therefore, drift occurs within one GOP. In curve 3, regular intra-frame updates are introduced to completely remove drift, i.e. $r_e = 0$. In curve 5 the performance is shown assuming that the encoder receives the transmission bitrate at the encoder *a posteriori* such that the reference frames can be adjusted, i.e. $r_e = r_t$.

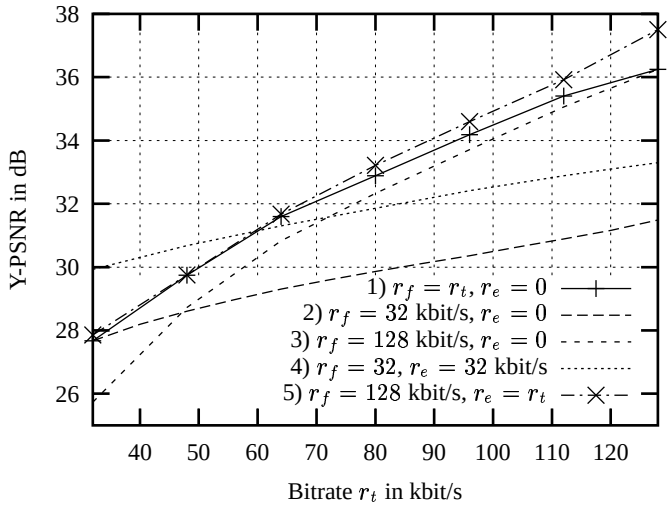

Fig. 6. Performance of PTVC in varying and unknown transmission bitrate conditions.

It can be observed that for a transmission bit rate reduction up to 50 % of the expected bit rate, i.e $r_t = 0.5 r_f$, all drift schemes (curve 3,5) outperform the schemes with intra coded enhancement layers (curve 2,4). Comparing curve 1 and 3 it is also obvious that the drift within one GOP does not effect the performance significantly as the PSNR with known transmission bit rate in curve 1 only increases slightly compared to curve 3. Finally, comparing curve 3 and 5 a gain of about 1 dB can achieved if the transmission bit rate is fed back to the encoder. Further investigations to evaluate this schemes in network environments with feedback or with statistical decoder state estimation is ongoing work. Additionally, the excellent performance of the PTVC in combination with unequal loss protection reported in [8] can be improved further with this feedback based drift compensation.

5. CONCLUSIONS

We presented a new video coding scheme combining the ITU-T H.26L test model with a progressive texture coder utilizing context based arithmetic encoding of bitplanes in the frequency domain. It has been shown that due to the progressive nature of our approach moderate quality degradation occurs when only parts of the bitstream are decoded. In this case drift prediction is introduced which can be considerably reduced by a periodic I-frame update or periodic drift removal using network feedback information. Experimental results indicate the usefulness of our approach for streaming video applications over networks with time-varying bitrate. Our proposed scheme can also be viewed as a means for data partitioning and the applicability in the combination with unequal loss protection for Internet packet losses has been shown [8]. Additionally, the subjective quality in varying bitrate conditions is significantly increased as the partial loss of data does not result in block artifacts nor in reduction of temporal resolution. Instead, the partial loss reduces the sharpness of the moving image as we have a coarser quantization. Future work will focus on rate distortion optimization and network adaptation aspects as well as further improvements of coding efficiency.

6. REFERENCES

[1] D. Taubman and A.Zakhor, "Multirate 3-D subband coding of video," *IEEE Transactions on Image Processing*, vol. 3, pp. 572–584, 1994.

[2] B.-J. Kim, Z. Xiong, and W. A. Pearlman, "Low bit-rate scalable video coding with 3D set partitioning in hierarchical trees (3D SPIHT)," *IEEE Transactions on Circuits and Systems for Video Techn.*, December 2000.

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

[4] S. McCanne, M. Vetterli, and V. Jacobson, "Low-complexity video coding for receiver-driven layered multicast," *IEEE Journal on Selected Areas in Communication*, vol. 15, pp. 983–1001, August 1997.

[5] W. Li, "Overview of fine granular scalability in mpeg-4 video standard," *IEEE Transactions on Circuits and Systems in Video Technology*, vol. 11, no. 3, pp. 385–398, March 2001.

[6] A.E. Mohr, E.A. Riskin, and R.E. Ladner, "Unequal loss protection: Graceful degradation of image quality over packet erasure channels through forward error correction," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 819–828, June 2000.

[7] U. Horn, K. Stuhlmüller, M. Link, and B. Girod, "Robust internet video transmission based on scalable coding and unequal error protection," *IEEE Transactions on Image Processing*, vol. 15, no. 1-2, pp. 77–94, September 1999.

[8] C. Buchner and T. Stockhammer, "Progressive texture video streaming for lossy packet networks," in *Proc. Packet Video Workshop 2001*, Kyongju, Korea, May 2001.

[9] C. Buchner, T. Stockhammer, D. Marpe, G. Blättermann, and G. Heising, "Progressive texture video coding," in *Proc. ICASSP 2001*, Salt Lake City, UT, May 2001.

[10] Gisle Bjontegaard, *H.26L Test Model Long Term Number 5 (TML-5) draft 0*, ITU-T Standardization Sector, Oct. 2000, Doc. Q15-K-59d1.

[11] Christian Buchner, "Progressive video coding for error-prone channels," M.S. thesis, Institute for Communications Engineering, Munich University of Technology, October 2000.

[12] A. Said and W.A. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, pp. 243–250, June 1996.

[13] ISO/IEC CD 15444-1, "JPEG-2000 image coding system," Tech. Rep., Committee Draft, Version 1.0, December 2000.