International Workshop on Very Low Bitrate Video Coding (VLBV'99), Kyoto, Japan, October 1999.

ERROR-RESILIENT VIDEO COMPRESSION VIA MULTIPLE STATE STREAMS

John G. Apostolopoulos

Hewlett-Packard Laboratories
Palo Alto, CA, USA
japos@hpl.hp.com

ABSTRACT

Video compression enables a number of applications by reducing the required bit rate needed to represent a video sequence, however the compressed video is much more susceptible to errors, e.g. bit errors or packet loss. Conventional video compression standards employ an architecture which we refer to as single-state systems since they have a prediction loop with a single state (e.g. the previous decoded frame) which if lost or corrupted can lead to the loss or severe degradation of all subsequent frames until the state is reinitialized (the prediction is refreshed). We propose to combat this problem of incorrect state and error propagation at the decoder by coding the video into multiple independently decodable streams, each with its own prediction process and state, such that if one stream is lost the other streams can still be used to produce usable video. The correctly received streams provide improved error concealment and, more importantly, enable faster state recovery for the lost stream. This approach is conceptually similar to multiple description coding, e.g. [1], however it differs in the representation used for each description as well as its use of state recovery.

1. INTRODUCTION

Video communication over bit-rate-limited and error-prone channels such as packet networks and wireless links requires both high compression and high error resilience. Important applications within this context include video streaming over the Internet and wireless video to handheld devices such as with the emerging Third Generation (3G) cellular system. Achieving both high compression and high error resilience is difficult because these are largely conflicting requirements.

This paper begins by briefly describing the basic problems that afflict compressed video in error-prone environments and the modern approaches developed to overcome these problems. We continue by introducing our multiple state streams approach for combating the problem of incorrect state and error propagation at the decoder, and conclude by providing some experimental results that show the effectiveness of the proposed approach.

2. BACKGROUND AND PREVIOUS RESEARCH

Most video compression systems possess a similar architecture based on motion-compensated (MC) prediction between frames, Block-DCT (or other spatial transform) of the prediction error, followed by entropy coding of the parameters. The basic error-induced problems that afflict a system based on this architecture include:

Bitstream Synchronization With the use of entropy coding (e.g. Huffman coding) an error can cause the decoder to lose synchronization with the bitstream, i.e. the decoder may not know what bits correspond to what parameters. Approaches for overcoming this problem include: placing resynchronization markers at strategic locations in the compressed video hierarchy (e.g. picture or slice headers), placing resync markers after every fixed number of bits (variable number of blocks), organizing the variable length coded blocks so that each block starts at a known location in the bitstream, partitioning the data into groups based on importance, and reversible variable length codes for (partial) recovery of lost data [2, 3, 4, 5, 6].

Incorrect State at Decoder Even if the bitstream has been resynchronized, another crucial problem is that the state of the representation at the decoder may not be the same as the state at the encoder. In particular, when using MC-prediction an error causes the reconstructed frame to be incorrect and often leads to significant error propagation to subsequent frames. We refer to this problem as having incorrect (or mismatched) state at the decoder, because the state of the representation at the decoder (the previous coded frame) is not the same as the state at the encoder. This problem also arises in other contexts (e.g. random access or channel acquisition) and a number of approaches have been proposed to overcome it including: independent coding of each frame (all Intra-frame or I-frame coding), periodic I-frames to periodically reinitialize the prediction loop (e.g. MPEG GOP), leakage within the prediction loop, and partial intra-encoding of each frame. While intra coding limits the effect of errors, the high bit rate required for intra coding limits its use in many applications.

The special case of point-to-point transmission with a back-channel facilitates additional approaches including: the decoder notifying the encoder to (1) reinitialize the prediction loop, or (2) which frames where correctly/erroneously received and therefore which frame should be used as the reference for the next prediction (referred to as NewPred in MPEG-4 Version 2 and as Reference Picture Selection (RPS) in H.263 Version 2 [7, 4, 5, 3]). NewPred can be very valuable in the case of a point-to-point link which also has a reliable back channel and with sufficiently short round-trip-delay; otherwise the visual degradation can be quite significant [7].

RPS can also be applied without a back channel in an approach referred to as Video Redundancy Coding (VRC) [8] where most frames are assigned to one of two or more independently coded threads and at periodic intervals (e.g. 7 or 10 frames for 2 or 3 threads [8]) a single frame is coded redundantly into each of the threads to enable synchronization; the decoder receives multiple copies of each sync frame and decodes one error-free copy while discarding the rest. If one thread is lost because of an error, the next sync frame can be used for recovery. Because VRC codes each sync frame multiple times and because of the reduced prediction accuracy that results from predicting frames spaced further apart in time, there is an extra overhead of approximately 35% and 57% for the 2 and 3 thread cases, respectively [8].

Layered or scalable approaches essentially prioritize data and thereby support intelligent discarding of the data (the enhancement data can be lost or discarded while still maintaining usable video), however the video can be completely lost if there is an error in the base layer. Multiple Description Coding (MDC) attempts to overcome this problem by coding a signal into multiple bitstreams such that any one bitstream can be used to decode a baseline signal, and additional bitstreams will improve the quality of the reconstructed signal [9]. MDC ideas have recently been applied to video coding [1] where the authors extend their multiple description quantizer work to the case of a DPCM loop.

The problem of bitstream synchronization can be largely minimized through an appropriate choice of tools and system design, however currently there is no general approach to overcome the problem of incorrect state at the decoder[1]. Therefore, the goal of this work is to overcome the problem of incorrect state and error propagation at the decoder. Specifically, we assume no back-channel between the decoder and encoder (e.g. broadcast or point-to-point with unreliable backchannel) and that it is not possible to specify different qualities of service (i.e. all bits or packets are equally likely to be lost).

3. PROPOSED SYSTEM ARCHITECTURE

Conventional video compression standards employ a similar architecture which we refer to as single-state systems since they have a single state (e.g. the previous coded frame) which if lost or corrupted can lead to the loss or severe degradation of all subsequent frames until the state is reinitialized (the prediction is refreshed). In our proposed approach we code the video into a number of independently decodable streams, each with its own prediction process and state information, as shown in Figure 1. By having multiple (independently decodable) state streams, if one state is corrupted the other states remain accurate and their respective streams can still be accurately decoded to produce usable video and may also be used to recover the lost state. In particular, the novelty lies in the use of data from the multiple streams to recover the lost state. Specifically, we exploit the redundancy between frames in the different streams to improve the recovery of the lost frames.

3.1. Encoder Portion of System

In the simplest incarnation of the proposed approach, the input video is partitioned into two subsequences of frames (even and odd) which are coded into two separate bitstreams. Specifically, each stream has a different prediction loop and a different state, and is independently decodable from the other. Since in general

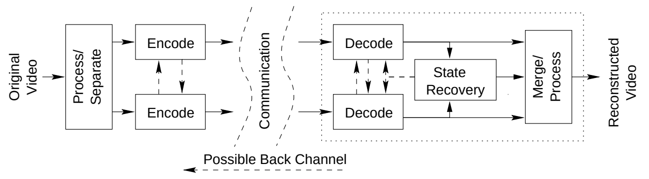

Figure 1: A general two-state stream video communication system.

there can be multiple coded streams each with its own state we refer to this approach as *Multiple State Streams*.

The encoder may consist of two separate conventional encoders, or an encoder which stores the last two previously coded frames (instead of just the last one) and chooses which previously coded frame to use to form the prediction for the current frame to be encoded. Both MPEG-4 and H.263+ support switching prediction among reference frames.

A higher bit rate is required to code the frames in separate subsequences as opposed to a single sequence, since they are spaced farther apart in time and prediction does not perform as well. However, unlike video redundancy coding there are no redundantly coded frames. The proposed approach is conceptually similar to multiple description coding, e.g. [1], however it differs in the representation used for each description and most importantly in its use of state recovery.

The different streams should be transmitted over different channels undergoing independent error effects to minimize the chance that both streams are lost. For example, the bitstreams from the even and odd frames can be sent in different packets over a packet network, so that any lost packet will only affect one of the streams.

3.2. Decoder Portion of System

In a manner similar to the encoder, the decoder may consist of two separate decoders, or a single decoder that alternates which previous decoded frame it uses to perform the prediction. If there are no errors and both the even and odd streams are received correctly, then both streams are decoded to produce the even and odd frames which are interleaved for final display.

If a stream has an error then the state for that stream is incorrect and there will be error propagation for that stream. However, the other independently decodable state stream can still be accurately and straightforwardly decoded to produce usable video. For example, if the bitstream corresponding to the odd frames is lost, the even frames may still be decoded and displayed, recovering the video at half its original frame rate. The error produces a temporary reduction in the frame rate, however there are no other distortions — which may be preferable to the case of conventional (single-state) approaches which are forced to either freeze the video or attempt to estimate the unknown video by performing some form of concealment; either of which can lead to significant distortion, especially if there are many frames before the next I-frame. The drawback of this simple approach is that if there are multiple errors before the next I-frame then both streams may be affected and the situation would be similar to that of the single-state approach.

The novelty in the multiple state streams approach is that it provides improved error concealment and enables state recovery of the lost stream. Conventional single-state approaches only have

[1]In certain special cases, such as a point-to-point link with a backchannel and with sufficiently short and reliable round-trip-delay, NewPred/ReferencePictureSelection can overcome the problem of incorrect state at the decoder. However, many important applications do not have a backchannel, and in other applications the backchannel may be unreliable or has a long RTD, thereby severely limiting NewPred's effectiveness.

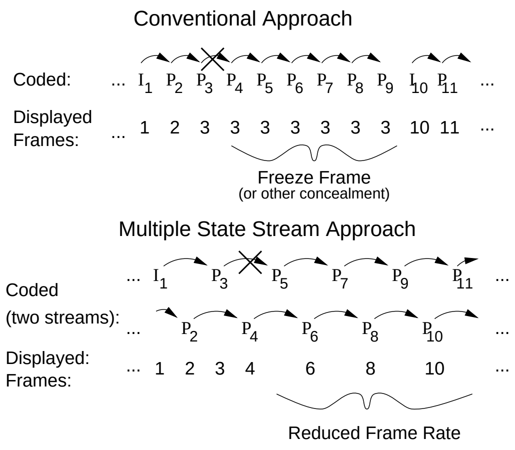

Figure 2: The effects of an error in decoding the frame that depends on frame P_3: In a conventional single-state approach (left) frame 4 is lost and the decoder may freeze frame 3 (or perform other error concealment) until the next I-frame. In a simple two-state stream approach (right) stream #1 is lost, however stream #2 can be accurately decoded – recovering the video at half its original frame rate but without any other distortions. In addition, stream #1 may be recovered by appropriately using stream #2.

access to *previous* frames to use in error concealment. The proposed approach provides access to both *previous and future frames* to conceal the errors, as illustrated in Figure 3. The availability and careful usage of both previous and future frames can greatly assist in error concealment. Furthermore, and most importantly, in many cases it is possible to actually recover the corrupted stream and thereby restore the video to its full frame rate. Specifically, the lost state (the coded frame) may be estimated with sufficient accuracy to be used as a reference for predicting other frames in that stream. As a result, the corrupted stream may be recovered very quickly (and potentially immediately), which is preferable to waiting for the next resynchronization. In the case of very low bit rate communication, I-frames may be spaced many secs or minutes apart (or other resynchronization mechanisms and their corresponding time constants), leading to significant visual quality degradation from an error. Therefore, it is very beneficial to have a mechanism to quickly recover from an error, without having to wait for an I frame.

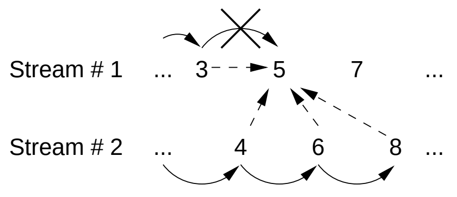

Figure 3: Correctly received streams enable improved error concealment and potential state recovery for the lost stream. Frame 5 may be concealed/recovered by using information from previous and *future* correctly decoded frames as signified by the dashed lines.

3.3. State Recovery Using Multiple State Streams

The problem of state recovery is similar to that of MC-interpolation (MC-I) where a frame is estimated using both previous and future frames in the sequence. MC-I has received considerable attention over the years and many of the algorithms and results developed for MC-I can be used in our context for state recovery. However, while there are similarities between the problems of state recovery and MC-I, there are also differences. First, state recovery and MC-I have a subtle but important difference in goals. MC-I is designed to produce video explicitly for display and therefore they should be visually very pleasing. The goal of state recovery is to produce an accurate estimate of the state (coded frame) so that it can be used to form an accurate prediction of the subsequent frames. Therefore, prediction accuracy and not visual quality is the most important criterion for state recovery. Second, state recovery attempts to recover a coded (distorted) frame from previous and future coded (distorted) frames. This contrasts with MC-I which tries to interpolate a clean frame from other clean frames. This has a number of implications, e.g. the conventional MC-I approach of estimating the motion between the previous and future frames may often be inappropriate. Third, it is often desirable for the decoder to perform state recovery in real time with low complexity. And fourth, the decoder has access to coded motion vectors and other information that may be useful for performing the state recovery.

There are a variety of possible approaches for estimating the lost frame. These include low-complexity approaches such as simply replacing the lost frame by a correctly decoded frame, or a MC correctly decoded frame, or a more sophisticated MC-I algorithm, e.g. compute the motion field across a subset of correctly decoded past and future frames from the corrupted and uncorrupted streams, and apply appropriate linear or nonlinear filtering along the motion trajectories. The recovery (interpolation) should also account for covered and uncovered areas within the frame by appropriately choosing to use only future or previous frames to estimate the appropriate areas. An adaptive method which selects the appropriate recovery method as well as the appropriate subset of past and future frames based on the specific video context would be most effective. The coded information within the bitstreams (e.g. motion vectors, inter/intra decisions) can be used instead of, or in addition to, performing motion estimation on the coded frames. The use of coded information can significantly reduce the complexity of state recovery at the decoder and may in certain cases improve its effectiveness.

4. EXPERIMENTAL RESULTS

The effectiveness of low-complexity state-recovery methods was examined for the bus (240×352 pixels/f, 30 f/s) and carphone (144×176 pixels/f, 30 f/s) sequences. Each sequence was coded into two streams (containing the even and odd frames) at 15 f/s each and at a constant quality of 29.5 dB (PSNR) for bus and 31.7 dB for carphone. The bits/P-frame when coding the bus sequence using two streams is approximately 53 kb/P-frame; 12 % larger than the corresponding single-state system (47.2 kb/P-frame). The bits/P-frame when coding the car sequence using two streams is approximately 2.4 kb/P-frame; 22 % larger than the corresponding single-state system (2.0 kb/P-frame). A conventional single-state constant quality (30.3 dB) coding of bus was also tested at about the same bitrate (55.3 kb/P-frame) as the two-state case for comparison.

The tests assume that the even sequence has an error which corrupts an entire frame while the odd sequence is received correctly, therefore the odd frames and correctly received even frames are used to estimate the lost even frame. Four low-complexity state-recovery methods are examined. InplaceMC avoids computing motion by using the coded motion vectors between the previous and next odd frames, scaling them by 1/2 and applying them inplace to the previous odd frame to estimate the current even frame. The other methods examined were averaging the previous and next odd frames, using the previous odd frame, and using the previous even frame. Of course these are very simple recovery methods, more sophisticated methods are likely to provide significant improvements.

The effectiveness of recovering a lost frame using each of the simple recovery methods is illustrated in Figure 4. The horizontal axis specifies the even frame that was lost and the various plots illustrate the accuracy for which that lost frame was estimated using each method. This figure illustrate the variability in the recovery accuracy for different frames within the same sequence, i.e. most frames of the bus sequence can be recovered with approximately the same accuracy for a given recovery method, while the recovery accuracy for carphone varies significantly depending on the specific frame that is lost. Note that the PSNRs' in Figure 4 are with respect to the lost coded frame, and not to the original frame, since the goal is to recover the coded frame. The PSNR of the MC-prediction of the lost even frame is also plotted to provide an indication of how well the lost frame can be estimated. InplaceMC works reasonably well for the bus sequence, followed by simple frame averaging (approximately 2-2.5 dB lower). InplaceMC performs worse than averaging for carphone, because carphone contains very few coded motion vectors.

Figure 5 (left) illustrates the performance of each method (except previous even) for bus when frame 6 is lost. Figure 5 (right) plots the interleaved recovered even and odd frames when frame 6 is lost, as well as a conventional single-state approach when frame 6 is lost. The single-state approach estimates the lost frame as the last correctly decoded frame. Figure 6 shows the recoverd even frames for carphone when frames 4 and 6 are lost. Notice from Figure 4 (right) that frame 4 can be recovered more accurately than frame 6, therefore when frame 6 is lost the consequences are much more severe then when frame 4 is lost.

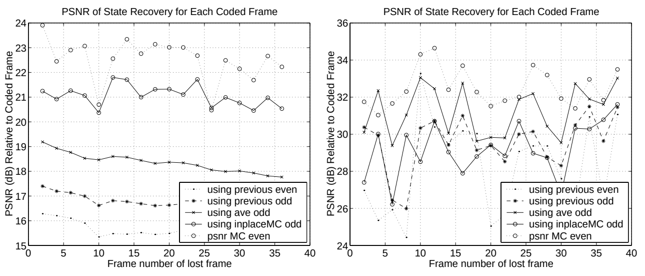

Figure 4: Accuracy of state-recovery as a function of lost frame for the bus sequence (left) and car sequence (right).

5. REFERENCES

[1] V. Vaishampayan and S. John, "Interframe balanced-multiple-description video compression," *Preprint*, 1999.

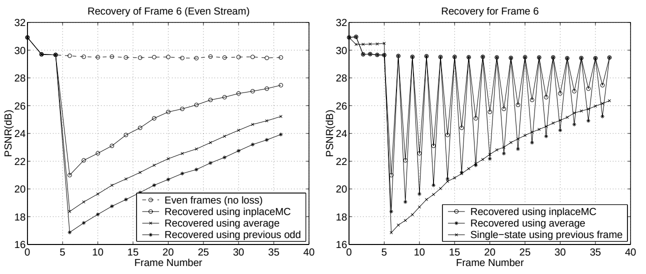

Figure 5: Recovery of even frames when frame 6 is lost (left) and interleaved recovered even and odd frames (right) for bus.

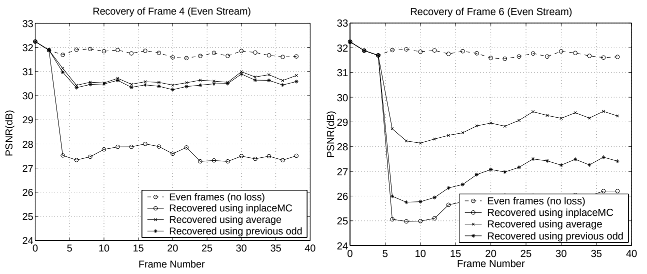

Figure 6: Recovery of even frames when frame 4 is lost (left) and when frame 6 is lost (right) for car sequence.

[2] R. Talluri, "Error-resilient video coding in the ISO MPEG-4 standard," *IEEE Communications Magazine*, pp. 112–119, June 1998.

[3] International Organization for Standardization, MPEG Committee, *Information Technology – Generic Coding of Audio-Visual Objects: Visual, Working Draft Version 2, N2553*, December 1998.

[4] International Telecommunication Union, *Video Coding for Low Bit Rate Communication, ITU-T Draft Recommendation H.263 Version 2*, September 26, 1997.

[5] N. Farber, B. Girod, and J. Villasenor, "Extension of ITU-T Recommendation H.324 for error-resilient video transmission," *IEEE Communications Magazine*, pp. 120–128, June 1998.

[6] D. Redmill and N. Kingsbury, "The EREC: An error-resilient technique for coding variable-length blocks of data," *IEEE Transactions on Image Processing*, vol. 5, pp. 565–574, April 1996.

[7] S. Fukunaga, T. Nakai, and H. Inoue, "Error resilient video coding by dynamic replacing of reference pictures," *Proceedings of GLOBECOM*, vol. 3, pp. 1503–8, November 1996.

[8] S. Wenger, G. Knorr, J. Ott, and F. Kossentini, "Error resilience support in H.263+," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, pp. 867–877, November 1998.

[9] Y. Wang, M. Orchard, and A. Reibman, "Multiple description image coding for noisy channels by pairing transform coefficients," *IEEE Workshop on Multimedia Signal Processing*, pp. 419–424, June 1997.