Combined Sequential Decoding and Error Concealment of H.264 Video

Claudio Weidmann and Petr Kadlec
ftw. Telecommunications Research Center Vienna
Donau-City-Strasse 1, A-1220 Vienna, Austria
Email: {weidmann,kadlec}@ftw.at

Olívia Némethová and Ameen Al Moghrabi
Vienna University of Technology
Gusshausstrasse 25/389, A-1040 Vienna, Austria
Email: onemeth@nt.tuwien.ac.at

***Abstract*— Data partitioning in H.264 Extended Profile video coding enables unequal error protection. Its performance can be improved if the decoder tries to also decode packets containing errors. We propose a soft-input sequential decoding algorithm for the prediction residuals encoded in low-priority packets. Information from the decoding process is then used to control additional error concealment. This combined technique provides significant PSNR gains compared to a simple packet-loss scenario.**

I. Introduction

The emerging video coding standard H.264 provides an error resilient mode (in the Extended Profile) which partitions data according to its importance [1]. Header data and motion vectors are labeled type A, so that they can be better protected. The *residuals* (prediction differences) of intra frames are labeled type B, while inter-predicted residuals are type C. Both type B and C data can be less protected than type A. All data is encapsulated into network abstraction layer units (NALUs), which are put in RTP packets for transmission over an IP network. Data partitioning does not apply to instantaneous decoding refresh (IDR) pictures, which provide decoder restart anchors and should therefore be heavily protected.

Another key point of the Extended Profile is that the type B/C prediction residuals are encoded with a context-adaptive variable-length code (CAVLC), which has a simpler and potentially more robust structure than the arithmetic encoding available in the Main Profile. This makes it comparatively easier to devise a soft-input decoder for type B/C packets.

In this paper we consider a scenario where all packets are equipped with a CRC to detect errors. Type A packets and IDR packets are protected by a stronger channel code than type B/C, so that one can assume that all type A and IDR packets are received without error. However, the weaker (or absent) protection of type B/C packets causes them to be received with random errors. The classic packet-loss framework simply discards packets that fail their CRC and tries to conceal the resulting decoding errors. We propose a soft-input sequential decoder for the packets containing errors; therefore we assume that the physical layer provides the soft information (log-likelihood ratios, LLR) for the bits in these packets.

The CAVLC for the residuals contains also fixed-length-coded (FLC) fields, in which errors go undetected by the sequential decoder. Therefore we introduce heuristic techniques for spotting macroblocks (MBs) which might contain such FLC errors, as well as for VLC errors. Traditional error concealment is then carried out for these MBs.

II. Sequential CAVLC decoding

Video coding performance depends critically on a variety of temporal and spatial prediction methods. In H.264 all prediction differences (called *residuals*) are encoded with basically the same method, regardless of their origin. We describe the most common case, the 4×4 luma residuals.

A sub-macroblock (SMB) of 4×4 residual pixels is transformed and quantized. The quantization indices are scanned in zig-zag order and the resulting sequence of *coefficients* is encoded using the CAVLC. First, a VLC encodes the number of nonzero coefficients and the number of trailing ones (i.e. up to three coefficients of amplitude one at the end of the sequence, ignoring any zeros in between). The signs of the trailing ones are then coded using one bit per coefficient. The VLC table in this first step is chosen depending on the number of nonzero coefficients in the two SMBs left and above of the current SMB. Second, the values of the remaining nonzero coefficients are encoded, starting with the last coefficient preceding the trailing ones. Values are coded with an Elias-type code that consists of a variable-length prefix and a fixed-length suffix, whose length depends on the current prefix VLC table or the prefix itself (escape mechanism). The VLC table is switched based on the currently decoded coefficient value. Third and fourth, the number of zeros between the nonzero coefficients and the zero run-lengths are encoded. The code is again adaptive; the VLC table used depends on the number of zeros still left to code. These encoding operations are repeated for all the SMBs of a picture slice in order to form a type B/C packet.

In summary, the encoding of residual coefficients depends causally on previous data in the same type B/C packet and the relevant information from type A packets. Given the packet length and the header information, in principle it would be possible to build a trellis that records all valid sequences of SMBs. However, this trellis grows too fast to be of practical interest and hence a method to reduce the number of explored trellis paths is needed.

One such method is sequential decoding, which was originally proposed for convolutional channel codes by Wozencraft

and Reiffen and later refined by Fano [2]. A list of partial decoding paths is kept in memory and each is labeled with a metric that allows comparing paths of different length. Since the list size shall be limited, the decoder needs to decide which paths to explore further based on the path metric. Several strategies exist, one of the simplest involves storing the paths in a stack which is sorted according to the metric. The top path (with the highest metric) is replaced by its extensions (a corresponding number of low-metric paths will be dropped from the stack) and the stack is sorted again. These steps are repeated until the top path has the required length and can be output as the decoded path.

The choice of metric is key to the performance of sequential decoding. Massey [3] has shown that the heuristic metric introduced by Fano does indeed minimize the error probability of sequential decoding of variable-length codes, provided the so-called "random tail assumption" holds. Consider a message w that is encoded with the binary variable-length codeword $x_{w,1}x_{w,2}\dots x_{w,\ell(w)}$ and transmitted over a binary-input memoryless channel with transition probabilities $P(y|x)$. The received vector $\boldsymbol{y}$ is assumed to be longer than the codeword $\boldsymbol{x}_w$. Then the random tail assumption states that the bits following the codeword (and belonging to the next codeword) are chosen i.i.d. with some distribution Q. For a good binary source code this is approximately satisfied with $Q=(\frac{1}{2},\frac{1}{2})$. Then the *a posteriori* probability that message w has been sent is

$$\Pr(w|\boldsymbol{y}) = P(w)\prod_{i=1}^{\ell(w)}\frac{P(y_i|x_{w,i})}{P_0(y_i)},$$

where $P(w)$ is the *a priori* probability that w has been sent and $P_0(y_i)=\sum_x P(y_i|x)Q(x)$ is the marginal channel output distribution induced by Q. The metric is now simply the logarithm (usually base two) of $\Pr(w|\boldsymbol{y})$:

$$L(w,\boldsymbol{y}) = \log P(w) + \sum_{i=1}^{\ell(w)}\log\frac{P(y_i|x_{w,i})}{P_0(y_i)}.$$

Using $Q=(\frac{1}{2},\frac{1}{2})$, the argument of the right-hand "channel term" can be directly computed from the soft inputs, e.g. the LLRs $\log\frac{P(y_i|1)}{P(y_i|0)}$. Extending the metric to sequences $w_1^k = w_1w_2\dots w_k$ is straightforward: the *a priori* term $\log\Pr(w_1^k)$ can be decomposed into the sum $\sum_{i=1}^k\log\Pr(w_i|w_1^{i-1})$, which takes care of dependencies on past message symbols, e.g. due to syntax and/or semantics of the H.264 CAVLC. The channel term in $\log\Pr(w_1^k|\boldsymbol{y})$ is clearly additive; its summands will have to be conditioned on w_1^{i-1}, since the choice of VLC codebook for w_i may depend on past symbols.

The *a priori* probabilities $P(w)$ must be known in order to compute this MAP metric. For simplicity and to avoid introducing any bias, we assume that the compression is efficient and hence the probability of emitting a codeword is exponentially related to its length: $P(w)=2^{-\ell(w)}/\sum_w 2^{-\ell(w)}$.

A key difference to sequential decoding of convolutional codes is the fact that not all syntactically valid paths correspond to valid decodings of a packet, since the header information imposes additional constraints. Only paths that have the correct length in bits *and* encode the correct number of SMBs (in the slice) are valid decoder outputs. This yields some error correction capability, since semantically invalid paths can be eliminated from the decoder stack.

III. Combined decoding and concealment

The fixed-length-coded (FLC) fields in the residual packets are mostly due to the suffixes in the coefficient value codes. The decoding metric assigns uniform probability to these fields, that is a hard decision is made and hence errors will go undetected. Our strategy is to compare the number of bits that the sequential decoder has flipped in the VLC parts with the expected total number of errors. If the difference is large, it is likely that errors occurred in the FLC fields. Concealment is then requested for the MB containing the FLC bit with the smallest LLR, that is the bit most likely to be in error.

A more severe decoding error occurs when the sequential decoder outputs a wrong VLC path. This is more likely to occur within I frame slices (since these will utilize the maximal packet size) and results in block-shaped artifacts over several MBs, differing in color and/or intensity from the rest of the picture. Therefore color and intensity differences as well as shape may be used to detect such errors and request concealment for the corresponding MBs.

To detect the artifacts, we first compute the difference between the I frame and the preceding P frame. For every pixel $d_k(i,j)$ of the the difference picture d, with k denoting the luminance y or the chrominance parts u and v, we compute the following metric:

$$M(i,j) = \sqrt{d_y(i,j)^2 + d_u(i,j)^2 + d_v(i,j)^2},$$

which is simply the square norm of the difference picture in yuv space. This metric takes into account both luminance (magnitude) and chrominance of the difference between the I frame and the preceding P frame. To obtain a robust artifact detection method, the metric is thresholded and combined with several other criteria, such as shape and size of the artifact. For example, it is important to avoid detecting as artifacts picture areas with high amounts of motion, which have larger difference picture values and thus also larger metric. This can be avoided by comparing information about slice layout of the I frame with the shape of the artifact under test, since errors will be localized in a slice. If the next I frame is already available (depending on the amount of buffering at the decoder), we can also check whether the artifact under test disappears in the next I frame. If it does, it is more likely to be a true artifact.

In summary, we use two methods to request concealment: for P/B frames, the sequential decoder will request concealment of individual MBs with possible FLC bit errors. This

Fig. 1. Example of sequential decoding and error concealment: original picture, picture with lost slices (packets), picture with residual errors after sequential decoding, difference picture, artifact detection metric, picture after concealment.

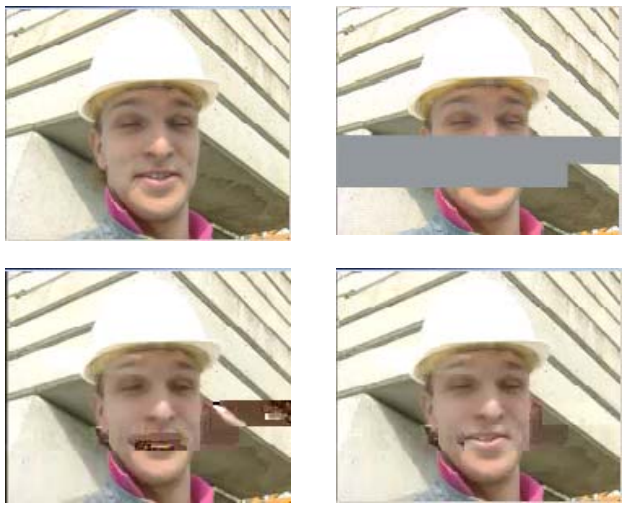

Fig. 2. Original picture, picture with lost slices (packets), picture with residual errors after sequential decoding, picture after concealment (slight errors remain).

does not always improve the PSNR or the perceived quality, especially if the bit in question has low significance (however, a more refined detection method could take this into account). For I frames, we use more traditional robust artifact detection, adapting it to the failure behavior of the sequential decoder. Since the decoder is more likely to fail in I frames, its own requests for concealment will be less trustworthy than for P/B frames. Conversely, the detection used for the I frames is less robust for P/B frames.

Factors that make concealment more difficult, such as scene changes, were not taken into account. It probably makes sense to have the video encoder detect scene changes and use an IDR frame to start a new scene, so this is a lesser issue.

Figures 1 and 2 show examples of the decoding and error concealment procedure. The pictures are taken from the simulations described in Section IV (channel SNR 7.5 dB).

A. Error Concealment Methods

There is a vast choice of error concealment strategies [4]. To keep the proposed method as generally applicable as possible, we did not want to use object-based methods. Our simulations focused on QCIF video (176×144 pixels or 11×9 MBs). At such low resolutions, a lot of visual information is contained in a single MB and therefore temporal interpolation provides in most cases a better and simpler basis for concealment than spatial or frequency domain interpolation. Hence motion compensated temporal prediction was used for P and B frames. Error concealment in I frames is more important, because the remaining errors propagate within the slice (H.264 uses spatial prediction to compress I frames) and also into the following P and previous B frames. We have chosen temporal interpolation with boundary matching for I frame concealment. If there was an error in a frame preceding the current I frame, and the area is smooth or contains clearly identifiable edges, spatial interpolation (possibly with smoothing along the edges) may be used to avoid error propagation.

IV. Simulation results

All simulations are based on the Foreman QCIF sequence, which was encoded with H.264 (joint model encoder, $QP_I = 28$, $QP_P = 30$, I9P GOP, maximal packet size 500 Bytes) and then transmitted over a binary-input additive white Gaussian noise (AWGN) channel. The total file size was 306'000 Bytes, of which 113'946 were in type A packets, which were assumed to be received error-free. The remaining 192'054 Bytes were in type B/C packets and were decoded with the sequential decoder. (The proportion of type A vs. B/C data varies a lot with video content.)

Figure 3 shows the bit error rate performance of the new decoder. Over a wide range of channel SNR, there is a coding gain of about 0.5 dB compared to binary hard decision decoding (thresholding). If only VLC bits are considered, the gain is even larger, since FLC errors cannot be corrected.

Figure 4 plots the YUV-averaged PSNR over the channel SNR. The additional gain obtained with error concealment

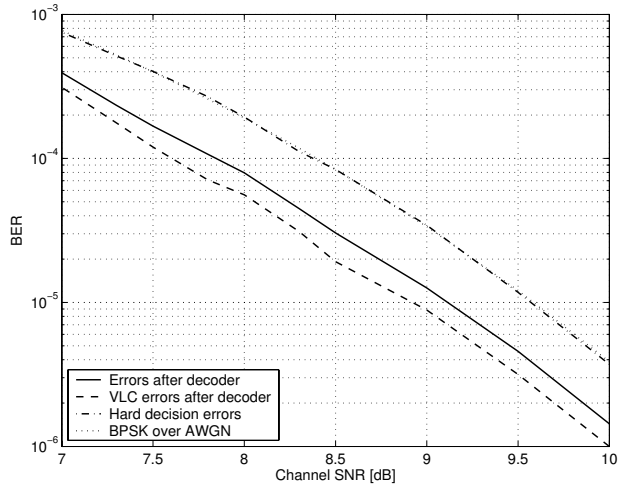

Fig. 3. Bit error rates vs. channel SNR

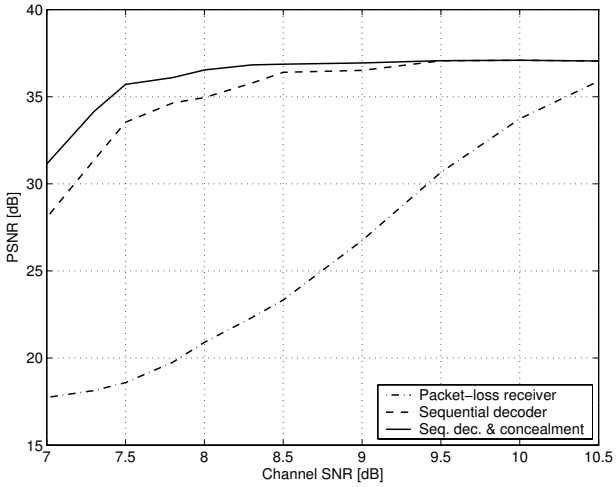

Fig. 4. Average PSNR vs. channel SNR

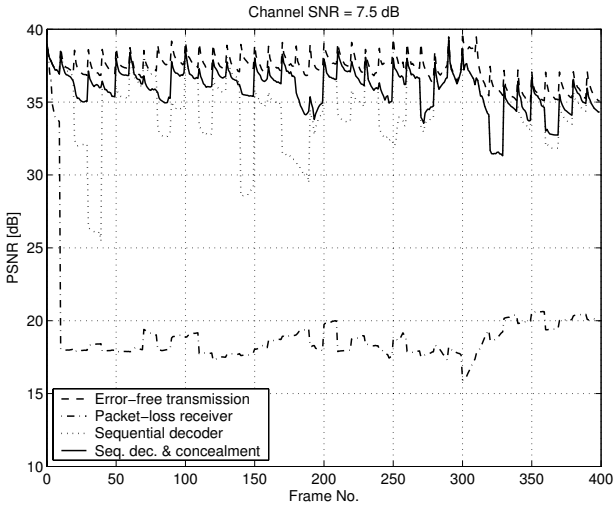

Fig. 5. Average per-frame PSNR

after sequential decoding can be seen clearly. The packet-loss scenario does not employ any concealment beyond what is inherent in the H.264 prediction mechanisms.

Finally, Figure 5 shows the per-frame PSNR when the channel SNR is 7.5 dB (hard decision: $P_{err,bit} \approx 4 \times 10^{-4}$, $P_{err,packet} \leq 0.8$). The first frame is an IDR assumed to be error-free. After that frame, the quality for packet loss degrades rapidly. The PSNR gains from concealment are often small, although the corresponding perceptual quality improvement can be quite remarkable.

V. Conclusion

Recent years have seen growing interest in error-resilient source decoding methods, ranging from simple error detection with repeat request to elaborate iterative joint source-channel decoders. Many of these methods are unrealistically complex or cannot be used with unmodified standard multimedia encoders. Sequential decoding clearly fills a gap in this respect, in particular for well-structured codes such as the H.264 CAVLC, which makes the implementation quite straight-forward. The decoder complexity essentially depends only on the stack size.

The presented simulation results show how residual source redundancy can be exploited in situations where the classic packet-loss model fails almost completely or would require very sophisticated error concealment. This is particularly interesting for advanced video coding standards such as H.264, which use a lot of prediction, hence making concealment more difficult. In contrast, standard concealment techniques are adequate to improve the already good performance of the sequential decoder.

Acknowledgment

The authors would like to thank Prof. Markus Rupp from Vienna University of Technology for supporting their research collaboration.

References

[1] J. V. T. of ITU-T and I. J. 1, "Draft ITU-T recommendation and final draft international standard of joint video specification (ITU-T rec. H.264 — ISO/IEC 14496-10 AVC)," March 2003, document JVT-G050 available on http://bs.hhi.de/˜wiegand/JVT.html.

[2] R. M. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. Inform. Theory*, vol. IT-9, pp. 64–73, April 1963.

[3] J. L. Massey, "Variable-length codes and the Fano metric," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 196–198, January 1972.

[4] Q.-F. Zhu and Y. Wang, "Error concealment for video communication," in *Compressed Video over Networks*, M.-T. Sun and A. R. Reibman, Eds. Marcel Dekker, Inc., 2001.