Towards Rate-decoder Complexity Optimisation in Turbo-coder based Distributed Video Coding

Zouhair M. Belkoura and Thomas Sikora

Communication Systems Group,
Technische Universität Berlin, Germany
{belkoura,sikora}@nue.tu-berlin.de

Abstract. Conventional hybrid video coding such as H.264 is compared to turbo-coder based Distributed Video Coding (DVC) from a complexity point of view. It is shown here that the overall workload in DVC can exceed that of H.264 by a substantial amount. Hence, DVC has the advantage of low-complexity encoding but at the price of high-complexity decoding, exceeding the encoder complexity of H.264. Given a turbo-coder based DVC setup, this work introduces a method to vary and possibly optimise decoder complexity while keeping encoder burden fixed at a low level. Using operational curves of the channel coding tools used in the DVC, a relation between bitrate and decoder complexity is given. It is demonstrated that in certain regions of these operational curves, large reductions in computations can be traded against relatively small increases in bitrate. Moreover, a variation of channel coder memory has influence on decoder complexity. Together with expected prediction error-rates, this permits to select an optimal operation point for given overall constraints.

1 Introduction

Wyner and Ziv showed as an extension of the Slepian-Wolf theorem, that a source X can be transmitted at a Rate $R < H(X)$ [1], when the decoder has access to a source Y that is correlated to X. Actual image and video source coding implementations based on this theorem have only emerged in the last ten years and are also known as distributed video coding (DVC) or Wyner-Ziv coding (WZC). A non-exhaustive overview of the achievements so far is given in [2]. Groundbreaking work was also done by Ramchandran et. al [3].
At present, DVC methods still do not match the rate-distortion performance of well known hybrid block-based video coders such as H.264 [4] (referred to as conventional coders in this work). However, a main advantage of DVC is the possible transfer of the computational burden from the encoder to the decoder side and is therefore an important future trend in video coding [5]. The search for low complexity encoding methods for video compression is the main motivation driving current research on DVC.
All of the DVC approaches presented in [2] make use of forward error correction (FEC) methods. The architecture presented by Stanford University [6] uses punctured turbo codes. Codecs based on the latter have received much attention in recent work on Wyner-Ziv coding e.g. [7], [8], [9].
In DVC, stronger turbo codes allow higher compression rates but come at the cost of increased decoding complexity. In many cases, employing DVC does not simply shift workload from encoder to decoder, but results in massively increased overall decoder computations as we will show below.
It is therefore reasonable to examine how this computational burden can be varied and what this variation can be traded for. Given, for instance, a scenario where many complexity-restricted sources transmit DV coded data to one common receiver, the decoder might face maximum complexity constraints with increasing incoming transmissions. Below, a method to vary decoder complexity is introduced. This method keeps the encoder burden fixed at a low level. Using operational curves of the channel coding tools used in the WZC, a relation between Wyner-Ziv bitrate and decoder complexity is given. It is demonstrated that in certain regions of these operational curves large reductions in computations can be traded against relatively small increases in bitrate. Moreover, with moderate puncturing, weaker channel codes can be employed resulting in yet another decoder complexity decrease.
This paper is organised as follows: In Sec. 2, the DVC architecture is explained and Sec. 3 presents the complexity estimation. The operational curves for turbo codes are shown in Sec. 4, demonstrating the tradeoff possibilities. Calculation of bitplane error probabilities based on estimated reconstruction quality is given in Sec. 5, Sec. 6 concludes the paper.

2 Wyner-Ziv Codec Architecture

Based on a simple DVC setup similar to that of Aaron and Girod [6], a Wyner-Ziv codec (WZC) is given with most of the processing done at the decoder. Figure 1 shows the functional blocks of the WZC. The odd frames f_{2i-1} of a video sequence $f_1, f_2, \ldots, f_M$ are coded using conventional intraframe coding. The even frames f_{2i} (Wyner-Ziv frames) of size N pixels are split into bitplanes which are then turbo encoded. The two constituent codes of the $(3N, N)$ systematic turbo code are $(2N, N, K)$ systematic recursive convolutional codes, with K being memory length. At the turbo coder output, the N systematic bits are discarded. In addition, the remaining $2N$ parity bits are punctured according to a puncturing rule: for every p_p bits, $p_p - n_p$ bits at known positions are deleted. Thus, only

$$\text{WZ-rate} = \frac{n_p}{p_p} 2N$$

bits are transmitted to the decoder. At the decoder, the reconstructed odd frames $\hat{f}_{2i-1}$ and $\hat{f}_{2i+1}$ are decoded and produce a prediction f'_{2i} for f_{2i}. f'_{2i} is produced using motion compensated interpolation. The bitplanes of f'_{2i} are used as systematic input for turbo decoding. In combination with the received parity bits, the turbo decoder corrects the prediction errors in f'_{2i} and returns a reconstructed version $\hat{f}_{2i}$ as its output.

If the reconstructed odd frames $\hat{f}_{2i-1}$ are not identical to f_{2i-1}, the prediction f'_{2i} will have more errors with respect to f_{2i}. Here, lossless reconstruction of the odd frames is assumed. Omitting bitplanes will reduce transmission bitrate and can be seen as equivalent to a quantisation operation. With bitplanes missing, the reconstruction $\hat{f}_{2i}$ will differ from f_{2i}.

It is important to note that unlike in other source coding algorithms, the bitrate (WZ-rate for this architecture) does not have direct impact on reconstruction quality. For a given number of errors in a bitplane of f'_{2i}, there is a minimal required value $n_{p_{min}}$ below which the turbo decoder cannot correct the prediction error. Increasing $n_p > n_{p_{min}}$ increases the WZ-rate, however it does not improve reconstruction quality. Turbo decoding either corrects all errors in a predicted bitplane or not. Increasing n_p however, has an important impact on decoding complexity as will be shown below.

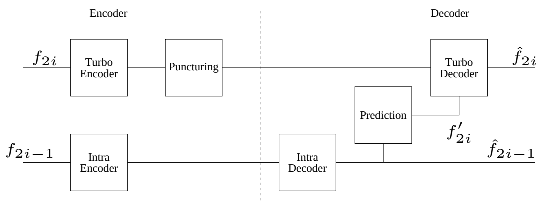

Fig. 1. Wyner-Ziv codec using a turbo coder

3 Workload Estimation

It is clear that the Wyner-Ziv decoder requires more computations than a conventional encoder because error correction (turbo decoding) is performed in addition to motion estimation. Below, the computational burden for motion estimation and that of turbo decoding is derived, and the two quantities are compared.

Not differentiating between operations for the sake of conciseness and simplification, full search motion estimation requires

$$P(2N + N_b)$$

operations (see Table 1), while turbo decoding demands

$$2N_p I_{TC} O_{MAP}$$

operations, with O_{MAP} operations per Maximum-A-Posteriori (MAP) decoding [10], (see Table 2). For turbo decoding, given typical parameters (Table 3), approximately $7.7 \cdot 10^9$ operations are required to decode a Wyner-Ziv coded frame. This can be contrasted with $0.2 \cdot 10^9$ operations for quarter-pel precision motion estimation or $0.01 \cdot 10^9$ operations for the full-pel case. These sample calculations illustrate the increase in overall workload when replacing a conventional video codec with a Wyner-Ziv architecture. Using these tools, moving computational burden from encoder to decoder can result in a hundredfold complexity increase.

In the DVC setup shown in Fig. 1, after choosing a suitable turbo code, the decoding complexity depends only on the number of transmitted bitplanes N_p and the number of required iterations I_{TC}. The latter of the two is not deterministic. I_{TC} depends on the combination of puncturing and interleaver scheme and, more importantly, on the number of transmitted parity bits n_p. With decreasing n_p, I_{TC} increases, as experimental data shows (see below).

Table 1. Number of operations for full search motion estimation

Parameters	
P	No. of search positions per block
N	No. of pixels per frame
N_b	No. of blocks per frame
n_b	No. of pixels per block
For each block b and search position p	
Difference between b and estimate $b'(p)$	n_b operations
Sum of absolute error	n_b operations
Compare error to current minimum	1 operation
Total	$Pn_bN_b + Pn_bN_b + PN_b = P(2N + N_b)$

Table 2. Number of operations for turbo decoding of a WZ coded frame

Parameters	
N_t	block length
K	coder memory
I_{TC}	No. of iterations per bitplane decoding
N_p	No. of bitplanes per frame
Iterative decoding process	
MAP decoding of convolutional code (see [10])	$O_{MAP} = N_t(8 \cdot 2^{2K} + 20 \cdot 2^K + 1)$ operations
One turbo code iteration	2 MAP decodings: $2O_{MAP}$ operations
Total	$N_pI_{TC}2O_{MAP}$

Table 3. Typical parameters used for numerical complexity calculations

Framesize	QCIF 176x144=25344 pixels
turbo blocklength N_t	25344 bits
code memory K	4
decoding interations I_{TC}	8
number of bitplanes N_p	8
motion estimation search region	256 positions
block size (quarter-pel precision)	4x4
block size (full-pel precision	16x16

4 Decoding Complexity vs. Puncturing Rate

The complexity of turbo decoding depends on the number of iterations I_{TC} and , through O_{MAP}, on the coder memory K. I_{TC} depends on the number of unpunctured bits per period n_p and thus on the WZ-rate, however this dependence is nonlinear. This is illustrated by simulation results for a DVC implementation for coding and reconstruction of a video frame for two turbo codes. In Figs. 2 and 3, the number of operations for turbo decoding are plotted against n_p for bitplane 0 (most significant bit) and bitplane 3 with 319 and 2310 bit errors in the prediction f'_{2i}. The leftmost curve point is at the minimum rate $n_{p_{min}}$ needed for successful decoding. It can be seen that for n_p slightly larger than $n_{p_{min}}$, there is a large complexity reduction. Comparison of the two codes in Fig. 4 show that low WZ-rate (e.g. $n_p = 4$) requires a strong code (memory 4), while at $n_p = 6$, the strong code is suboptimal because a weaker, less complex code (memory 3) can accomplish the decoding task. When designing the WZ codec, this complexity examination can be made for varying error-rates and puncturing rates across a number of codes with different strengths. Given overall constraints on bitrate and complexity, an appropriate operation point can thus be selected during encoding by choosing puncturing scheme and turbo code. Leaving memory issues aside, the encoder complexity is not changed by this variation.

5 Estimating Bitplane Error Rates

To choose an operating point, the encoder needs to be able to estimate the error rate in a bitplane. Given an expected prediction quality (in PSNR/MSE), this can be done as follows: The turbo decoder assumes a zero-mean Laplacian distribution

$$p_X(x) = \frac{\alpha}{2}e^{-\alpha|x|}$$

for the prediction error

$$e_p = f_{2i} - f'_{2i},$$

with the mean square error being

$$E[e_p^2] = \frac{2}{\alpha^2}.$$

Let y_i be the pixel values of frame f_{2i} and

$$y'_i = y_i + e_{p_i}$$

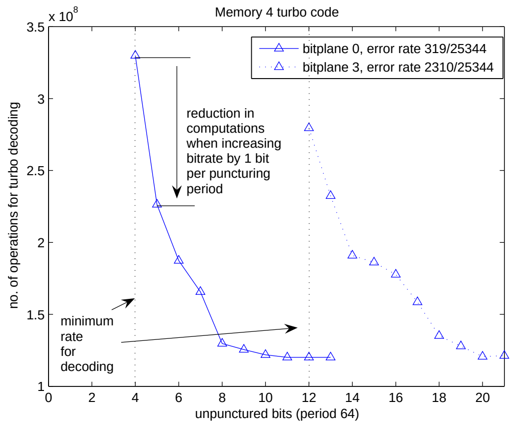

Fig. 2. Required computations versus unpunctured bits (code memory 4) - With increasing unpunctured bits (i.e. increasing WZ-rate), the number of required operations for decoding decreases. Depending on error rates, there is a minimal required rate

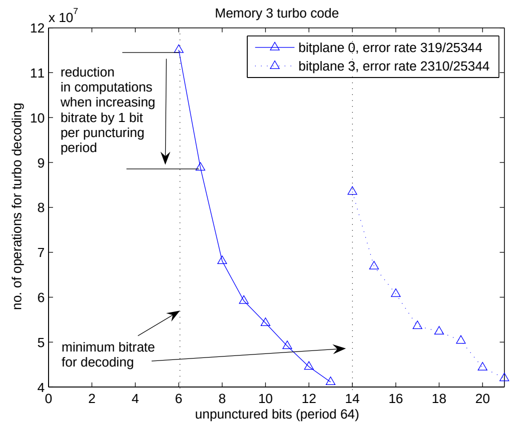

Fig. 3. Required computations versus unpunctured bits (code memory 3) - With increasing unpunctured bits (i.e. increasing WZ-rate), the number of required operations for decoding decreases. Depending on error rates, there is a minimal required rate

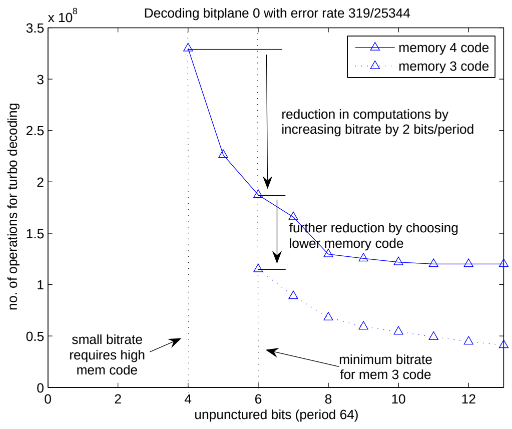

Fig. 4. Comparison of decoding complexity across different codes for 319 errors per block - Small WZ-rate (4 unpunctured bits) requires a strong code, a higher rate enables optimisation by selecting a weaker code.

the corresponding prediction. y_i contains 8 bitplanes $y_i^0, .., y_i^7$ with y_i^0 being the most significant bit.

$$y_i^0 = \begin{cases} 0 \text{ for } y_i < 128 \\ 1 \text{ otherwise} \end{cases}$$

The probability of prediction error in bitplane 0 for a pixel value y_i is thus

$$p(e_p > d^0(y_i)) \quad \text{with} \quad d^0(y_i) = |y_i - 127.5|.$$

Given a probability distribution $p_Y(y)$ over the possible pixel values, the bitplane error probability is thus

$$p_e^0 = \sum_{i=0}^{255} p(y=i) \int_{d^0(y)}^{\infty} \frac{\alpha}{2} e^{-\alpha e_p} de_p.$$

For N pixels per image, $p_e^0 N$ errors in bitplane 0 can be expected. The calculation for the remaining bitplanes is similar.
Figures 5 and 6 show the number of actual and estimated bit errors in bitplanes 0 and 3 for the first 25 Wyner-Ziv coded frames of the foreman sequence. With the formulae above, reasonable estimates for the number of bitplane errors can be made. With this quantity available at the encoder, an appropriate code and puncturing rule can be chosen during encoding to meet overall decoder complexity constraints. With the possible operating points stored in memory, the encoder burden increases only due to error estimation. The number of additional computations is negligible.

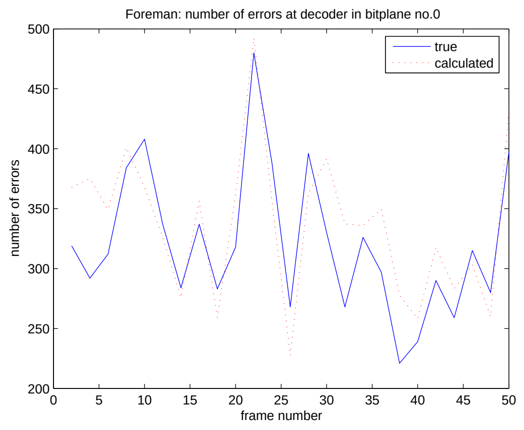

Fig. 5. Actual and estimated number of erroneous bits in bitplane 0 for the first 25 Wyner-Ziv frames of Foreman

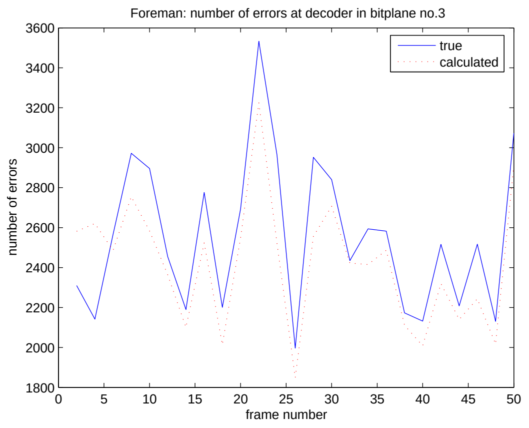

Fig. 6. Actual and estimated number of erroneous bits in bitplane 3 for the first 25 Wyner-Ziv frames of Foreman

6 Conclusions

It has been shown that Wyner-Ziv coding of video can be an order of magnitude more complex than a conventional hybrid video codec. The decoder complexity of a turbo-coder based architecture can be varied by increasing the transmission bitrate while keeping both the encoder complexity and the reconstructed signal quality constant. Using this relation in combination with operational curves for different turbo codes and puncturing rules, is becomes possible to optimise rate vs. decoder complexity for a given application/transmission/network requirement.

References

1. Wyner, A.D., Ziv, J.: The rate-distortion function for source coding with side information at the decoder. IEEE Transactions on Information Theory **IT-22** (1976) 1–10
2. Girod, B., Aaron, A., Rane, S., Rebollo-Monedero, D.: Distributed video coding. Proceedings of the IEEE **93** (2005) 71 – 83
3. Puri, R., Ramchandran, K.: Prism: A 'reversed' multimedia coding paradigm. In: Proceedings of the IEEE International Conference on Image Processing, Barcelona, Spain. (2003)
4. Misc.: Special issue on the h.264/avc video coding standard. IEEE Transactions on Circuits and Systems for Video Technology **13** (2003)
5. Sikora, T.: Trends and perspectives in image and video coding. Proceedings of the IEEE (2005)
6. Aaron, A., Zhang, R., Girod, B.: Wyner-ziv coding of motion video. In: Proc. Asilomar Conference on Signals and Systems. (2002)
7. Artigas, X., Torres, L.: Iterative generation of motion-compensated side inforrmation for distributed video coding. In: Proceedings of the IEEE International Conference on Image Processing, Genoa, Italy. (2005)
8. Natario, L., Brites, C., Ascenso, J., Pereira, F.: Extrapolating side information for low-delay pixel-domain distributed video coding. In: Int. Workshop on Very Low Bitrate Video Coding, Sardinia, Italy,. (2005)
9. Ascenso, J., Brites, C., Pereira, F.: Motion compensated refinement for low complexity pixel based distributed video coding. In: IEEE International Conference on Advanced Video and Signal-Based Surveillance, Como - Italy. (2005)
10. Bossert, M.: Kanalcodierung. Teubner (1998)