AN EFFICIENT REQUEST STOPPING METHOD AT THE TURBO DECODER IN DISTRIBUTED VIDEO CODING

M. Tagliasacchi[*], *J. Pedro*[†], *F. Pereira*[†], *S. Tubaro*[*]

[*]Dipartimento di Elettronica e Informazione, Politecnico di Milano, Milan - Italy
[†]Instituto Superior Técnico, Instituto de Telecomunicações, Lisbon - Portugal

ABSTRACT

Most of the literature on distributed video coding (DVC) is based on a Wyner-Ziv video coding architecture that adopts turbo codes to replace conventional source coding tools. By moving the motion estimation task at the decoder, it achieves light encoding complexity. Nevertheless, this scheme has a serious limitation that hinders its practical application. In fact, the decoder needs an efficient way to estimate the probability of error without assuming the availability of the original video at the decoder. The original frames are used to detect when a sufficient number of parity bits are received, in order to guarantee a residual BER (bit-error rate) below a given threshold, typically set to be equal to 10^{-3}*. In this paper we investigate the use of a practical stopping criterion that can be used to detect successful decoding, without the need to have access to the original frames. We study the robustness of this method for binary i.i.d. sequences transmitted over a BSC channel. We show that there is a tradeoff between residual BER and rate overhead, that can be adjusted by tuning the threshold of the stopping criterion. We apply these concepts to both pixel-domain Wyner-Ziv coding (PDWZ) and transform-domain Wyner-Ziv coding (TDWZ), showing that the rate-distortion performance is only marginally affected by the lack of original frames at the decoder. A maximum loss of* $0.1dB$ *is observed at high bit-rates.*

1. INTRODUCTION

The standards developed by ITU-T and MPEG set the current digital video coding paradigm - hybrid DCT and inter-frame predictive coding with motion compensation - which is largely used in hundred of millions of video codecs deployed around the world. This kind of architecture is well-suited for applications where the video is encoded once and decoded many times, i.e. one-to-many topologies, such as broadcasting or video-on-demand, and the cost of the decoder is more critical than the cost of the encoder. In recent years, with emerging applications such as wireless low-power surveillance, multimedia sensor networks, wireless PC cameras and mobile camera phones, the traditional video coding architecture is being challenged. These applications have rather different requirements than those of broadcasting systems. Distributed video coding (DVC) also know as Wyner-Ziv (WZ) coding [1], fits well these scenarios, since it enables to exploit the video statistics, partially or totally, at the decoder only. This paradigm targets a flexible allocation of complexity between encoder and decoder which may have as a rather important sub-case, low encoding complexity. In the last 3-4 years, there has been a growing interest in developing DVC practical solutions. Some of the most relevant solutions adopt an architecture based on punctured turbo coding and a feedback channel, both for pixel and transform domains [1][2]; the feedback channel is used by the decoder to request more (parity) bits to the encoder and thus successfully correct the errors in the so-called side information. The latter is generated at the decoder as an estimation of the video information to be coded. To control the number of requests to be made through the feedback channel, an estimation of the error probability, e.g. at bit-plane level, is needed at the decoder. This error probability is typically made lower than a certain threshold, usually 10^{-3}, to limit the amount of residual errors. In the first feedback channel based DVC solutions, the error probability computation is made using the original video data at the decoder which is not a realistic situation but gives an "ideal" performance in terms of rate-distortion performance. To obtain a practical DVC solution, a "non-ideal" request stopping criterion is needed which performs as close as possible to the "ideal solution" in terms of residual errors and overall rate.

The paper is organized as follows: Section 2 briefly summarizes the pixel-domain and transform-domain Wyner-Ziv video coding scheme. Turbo encoding and decoding are detailed in Section 3. The proposed stopping criterion is described in Section 4, and experimental results are presented and analyzed in Section 5. Finally, conclusions and some future work topics are presented in Section 6.

2. WYNER-ZIV CODING ARCHITECTURE

The video coding scheme that we adopt in this paper is based on the Wyner-Ziv codec described in [1]. This coding architecture offers an intra-frame encoder and inter-frame decoder with very low computational encoder complexity.

The video sequence is partitioned in group of pictures (GOP) of size G. Let us denote with t the generic time instant within a GOP and with $X(t)$ the original frame at time t. The first frame of each GOP is intra-frame coded, and it is denoted as key frame. The frames $X(t)$, $t \in [1, G-1]$ are called Wyner-Ziv frames. Two coding modes are enabled in [1]: pixel-domain (PDWZ) and transform-domain (TDWZ) Wyner-Ziv coding. In the former case, each pixel in the Wyner-Ziv frame is uniformly quantized. Bit-plane extraction is performed from the entire image and then each bit-plane is fed into a RCPT (Rate Compatible Punctured Turbo) encoder to generate a sequence of parity bits. As for TDWZ coding, a 4 × 4 block DCT transform is applied to the original frames first. Then, DCT coefficients are organized into frequency bands, where each band groups together DCT coefficients of the same index across the whole frame. Each frequency band is uniformly quantized and bit-plane extraction is performed. Each bit-plane of each frequency band is independently fed into a RCPT encoder to generate parity bits. All parity bits are temporarily stored in a buffer at the encoder. Packets consisting of a subset of parity bits are obtained by puncturing, and sent on request to the decoder, as detailed in the following.

At the decoder, the two key frames $\hat{X}(0)$ and $\hat{X}(G)$[1] are used by the motion-compensated frame interpolation module [2] to generate the side information $Y(t)$. For each bit-plane (and for each frequency band in the TDWZ case), the turbo decoder requests packets of parity bits by means of a feedback channel, to "correct" the side information $Y(t)$ into the decoded frame $\hat{X}(t)$. A Laplacian correlation channel is assumed and an error probability estimation method is used to decide when decoding is successful and no more parity bits are needed. When the desired number of bit-planes (and

THE WORK PRESENTED WAS DEVELOPED WITHIN VISNET II, A NETWORK OF EXCELLENCE (HTTP://WWW.VISNET-NOE.ORG), AND DISCOVER, A FUTURE EMERGING TECHNOLOGY PROJECT (HTTP://WWW.DISCOVERDVC.ORG/) BOTH FUNDED BY THE EUROPEAN COMMISSION.

[1] We denote with $\hat{X}$ the decoded version of frame X.

frequency bands) have been decoded, a reconstruction module combines the decoded bit-planes with the side information to obtain the reconstructed frame $\hat{X}(t)$.

3. TURBO ENCODING/DECODING

Turbo encoding/decoding is performed at the bit-plane level. The encoder extracts the bit-planes of the pixel-domain (transform-domain) representation of the original frame X (we disregard the time index t to simplify the notation), organizing the extracted bits in B vectors $\mathbf{x}^b$, $b = 1,\dots,B$, of length N ($B \times K$ vectors $\mathbf{y}^{b_j}$, $b = 1,\dots,B$, $j = 1,\dots,K$, of length N). B is number of decoded bit-planes (typically $B \in [1,4]$) and N is the total number of pixels (blocks) in a frame, while K is the number of DCT frequency bands ($K = 16$).

Since encoding and decoding do not depend on the bit-plane level b, in the following we consider the processing of a generic binary sequence $\mathbf{x}$ of length N. Figure 1 gives an overview of the turbo encoding/decoding process that is detailed in the following.

- *Turbo encoding*: The turbo encoder consists of two parallel Recursive Systematic Convolutional (RSC) encoders with rate 1/2. For each input bit there are two output bits (one parity bit and one systematic bit). The input of the second RSC is interleaved to decorrelate the input sequence between the two constituent RSC encoders. Only parity bits $\mathbf{y}^p$ are buffered to be sent to the decoder, while the systematic bits are discarded.
- *Turbo decoding*: At the decoder, the generated side information represents a noisy version of the original frame. After bit-plane extraction, the binary sequence $\mathbf{y}$ is obtained. Let $p(\mathbf{x},\mathbf{y})$ denote the statistics of the correlation channel. The Wyner-Ziv coding architecture adopted in this paper uses a Rate Compatible Punctured Turbo (RCPT) code in order to transmit the minimum amount of parity bits needed to decode each bit-plane. At each request r, additional N/P parity bits are received, where P denotes the puncturing period.
  - *Initialization*: Set iteration index to $r = 0$
  - *Parity bit request loop*:
    1. *Evaluate request stopping criterion*: The decoder evaluates if the residual bit-error rate at the rth request, BER_r, is below a given threshold, i.e. $BER_r < \tau$. If this is the case, decoding is declared to be successful. Set $\mathbf{u} = \mathbf{u}_r$ and terminate turbo decoding. The residual BER_r is defined as:
    $$BER_r = d_H(\mathbf{u}_r, \mathbf{x})/N \tag{1}$$
    where $D = d_H(\mathbf{u}_r, \mathbf{x})$ denotes the Hamming distance between the original and the decoded binary sequence. In the next section we provide further details about this step.
    2. *Parity bit request*: Request additional N/P parity bits, where P is the puncturing period.
    3. *Log-MAP decoding*: Turbo decoding is performed through an iterative process which makes use of two identical Soft-Input Soft-Output (SISO) decoders. It receives in input the side information $\mathbf{y}$ and a subset of $r(N/P)$ parity bits $\mathbf{y}_r^p$, in order to produce $\mathbf{u}_r$, the decoded version of $\mathbf{x}$.
    4. Increment $r = r + 1$

4. PROPOSED STOPPING CRITERION

In the past literature, the error probability estimation step defined in the previous section assumes ideal error estimation, i.e. the original binary sequence $\mathbf{x}$ is available at the decoder to compute BER_r exactly. In this case, the optimal number of requests r^* is obtained by monitoring the value of BER_r, i.e.

$$r^* = \arg\min_r [BER_r < \tau], \tag{2}$$

where τ is a threshold that indicates the acceptable residual BER. In the distributed video coding literature, τ is typically set to 10^{-3}.

In this section, we propose a simple algorithm that allows to estimate the optimal number of parity bit requests, without having access to the original sequence $\mathbf{x}$. To this end, it is instructive to provide further details about the turbo decoding algorithm.

Each SISO decoder uses the log-MAP (Maximum A Posteriori) algorithm [3] to determine the Logarithm of the A Posteriori Probability (*LAPP*) ratio. Let $LAPP_r[u(i)]$ denote the LAPP ratio computed after the rth parity bit request:

$$LAPP_r[u(i)] = \log \frac{\Pr(u(i) = 1|\mathbf{y}, \mathbf{y}_r^p, p(\mathbf{x},\mathbf{y}))}{\Pr(u(i) = 0|\mathbf{y}, \mathbf{y}_r^p, p(\mathbf{x},\mathbf{y}))}, \tag{3}$$

where $u(i)$ is the ith bit to be decoded. At each iteration, the SISO decoders exchange extrinsic soft information between each other. After the last iteration, the final *LAPP* ratio is determined, and a hard decision is made based on the sign of the *LAPP* ratio, i.e.

$$u(i) = \begin{cases} 1, & \text{if } LAPP[u(i)] > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

At each request, the following statistics can be computed from the *LAPP* ratios:

$$L_r = 1/N \sum_{i=0}^{N-1} |LAPP_r[u(i)]|, \tag{5}$$

which represents the sample mean of the absolute values of the *LAPP* ratios, computed across the N sequence samples. Intuitively, the higher is the absolute value of $|LAPP_r[u(i)]|$, for a given sample i, the more confident is the turbo decoder in making the hard decision in (4).

Figure 2 shows both BER_r and L_r for a sample decoded sequence. In this example, $\mathbf{x}$ is one realization of a Bernoulli 1/2 i.i.d. process and $\mathbf{y}$ is the output of a binary symmetric channel with crossover probability p. BER_r and L_r are computed as in (1) and (5) after turbo decoding at the rth parity bit request. We notice that when BER_r drops to zero, indicating successful decoding, L_r rapidly rises at a much larger value. As more parity bits are received, L_r further increases, but at a lower pace. The qualitative behavior depicted in Figure 2 tends to be general, thus inspiring the following stopping criterion:

$$\boxed{\text{If } L_r > T_L\text{, then stop requesting parity bits.}} \tag{6}$$

The use of L_r for the definition of a stopping criterion has already been investigated in the communications literature [4], although from a different perspective. In fact, in [4] the number of parity bits is considered to be fixed, and the goal is to reduce the number of turbo decoder iterations, thus limiting the computational complexity. Instead, in this paper, we use a fixed number of iterations (equal to 18), but a variable rate.

5. EXPERIMENTAL RESULTS

In the following, we analyze the performance of the proposed stopping criterion. First, we consider the simple case of a binary sequence transmitted over a binary symmetric channel (BSC). Then, we examine the PDWZ and TDWZ scenarios.

5.1 BSC scenario

Let $\mathbf{x}$ denote an i.i.d. Bernoulli 1/2 sequence of length N. The sequence $\mathbf{x}$ is sent through a binary symmetric channel (BSC), characterized by a crossover probability p. Let $\mathbf{y}$ denote the binary sequence received at the decoder, also of length N.

With reference to Figure 2, we notice that the selection of the threshold T_L is conditioned to the following conflicting requirements:

- The residual BER should be kept below a pre-determined threshold τ.

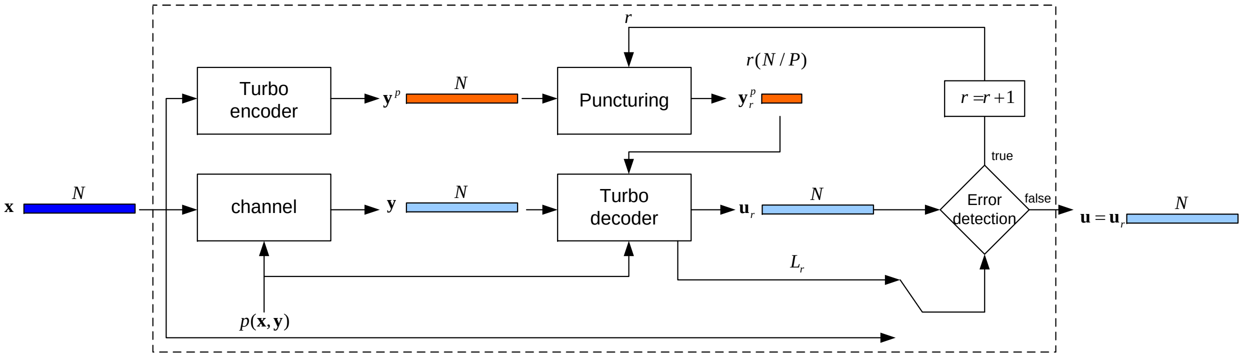

Figure 1: Block diagram of the proposed system.

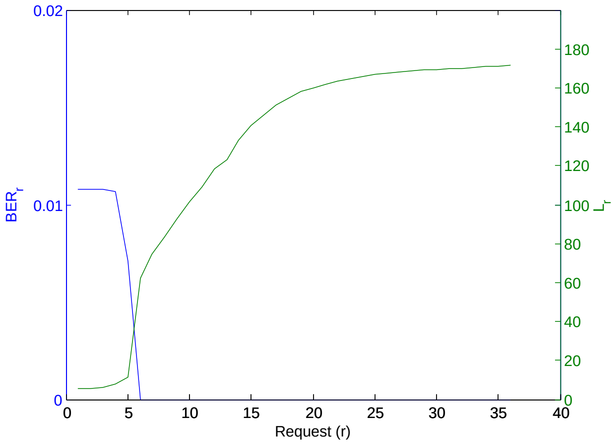

Figure 2: BER vs. r and L_r vs. r. $N = 10000$, $p = 0.01$

- The rate overhead should be minimized. In other words, no additional parity bit requests should be made when the residual BER is below the selected threshold τ.

Increasing T_L decreases the residual BER, thus satisfying the first requirement, but inevitably increases the number or requests and, in some cases, an overhead is introduced. In the following, we analyze how T_L can be adjusted to jointly satisfy both requirements.

In our experiments, we generate $R = 1000$ realizations of $\mathbf{x}$ and $\mathbf{y}$, of length $N = 10000$. For each realization, we perform turbo decoding with ideal error detection, in order to obtain r^*. Then, we set the value of T_L and we record the minimum number of requests $\bar{r}$ needed to satisfy $L_r > T_L$, i.e. $\bar{r} = \arg\min_r [L_r > T_L]$. We define the relative overhead as

$$\Delta r = \frac{\bar{r} - r^*}{r^*} \tag{7}$$

At the same time, we record the residual $BER_{\bar{r}}$.

Figure 3c shows the residual BER as a function of T_L. Recall that L_0 is the $LAPP$ ratio when no parity bit requests are made, i.e. when no parity bits have been received. Therefore, it depends only on the prior information about the source. In fact, it is possible to show that the value of L_0 depends solely on the original BER of the side information, i.e.

$$E[L_0] = \left| \log \frac{1-p}{p} \right| \tag{8}$$

Equation (8) can be obtained evaluating the expectation of (5) when $r = 0$, and assuming transmission over a BSC channel with crossover probability p. For a given value of p, no requests are made when $T_L < E[L_0]$, thus the BER remains equal to p. Increasing T_L above $E[L_0]$ decreases the residual BER. For values of BER lower than $2 \cdot 10^{-5}$, a non-linear behavior is observed, due to the fact that the turbo codec error floor is reached.

Figure 3a shows the overhead Δr as a function of T_L. As expected, the overhead grows by increasing T_L, as more requests are needed on average to satisfy $L_r > T_L$. For $p = 0.10$, the overhead is kept within 10^{-3} for $T_L < 80$. For $p = 0.01$, the same overhead is attained for $T_L < 23$. Since the BER decreases by increasing T_L, we can conclude that, for the same overhead, a larger residual BER is to be expected for smaller values of p.

Figure 3b combines Figure 3a and Figure 3c showing BER vs. Δr. We can observe that, for $N = 10000$, it is possible to achieve a residual $BER < 10^{-5}$ with an overhead Δr as little as 10^{-4}.

We repeated the same experiment for a smaller block length $N = 1000$. We observe the same kind of behavior with one important exception: for the same overhead as before ($\Delta r = 10^{-4}$), we attain a larger upper bound for the BER, i.e. $BER < 2 \cdot 10^{-3}$. This can be explained by considering that the performance of the turbo codec tends to decrease at shorter block sizes, especially when the code rate approaches 1, as it is the case for small values of p. This is due to the fact that the error floor is higher for smaller block lengths.

5.2 PDWZ scenario

In this section we provide results for the pixel-domain Wyner-Ziv coding architecture described in Section 2. The key frames were encoded with H.264/AVC Intra (Main Profile) with the quantization steps chosen through an iterative process to achieve an average Intra frame quality (PSNR) similar to the average WZ quality. As usual for WZ coding, only luminance data has been coded.

We show results for the *Foreman* and *Coastguard* sequences, at QCIF resolution and 15 fps. The GOP size is set to 2 frames. At QCIF resolution, the block length is equal to $N = 176 \times 144 = 25344$. Table 1 shows the initial BER, i.e. when no parity bits are received, for the first B bit-planes:

$$BER(B,b) = d_H(\mathbf{x}_B^b, \mathbf{y}_B^b)/N, \tag{9}$$

computed comparing the bth bit-plane of the original frame with the corresponding bit-plane of the side information. $B \in [1,4]$ is the total number of encoded bit-planes, thus determining the target distortion, while $b \in [1,B]$ is the index of the specific bit-plane. As expected, the BER is lower for smaller values of b. We notice that, for a given b, the tabulated values depend also on B. This is due to the fact that the key frames are quantized at different distortion levels depending on B.

The reader should be careful when comparing the numbers in Table 1 with the parameter p of the BSC channel defined previously. In fact, for the case of turbo coding applied to DVC, the

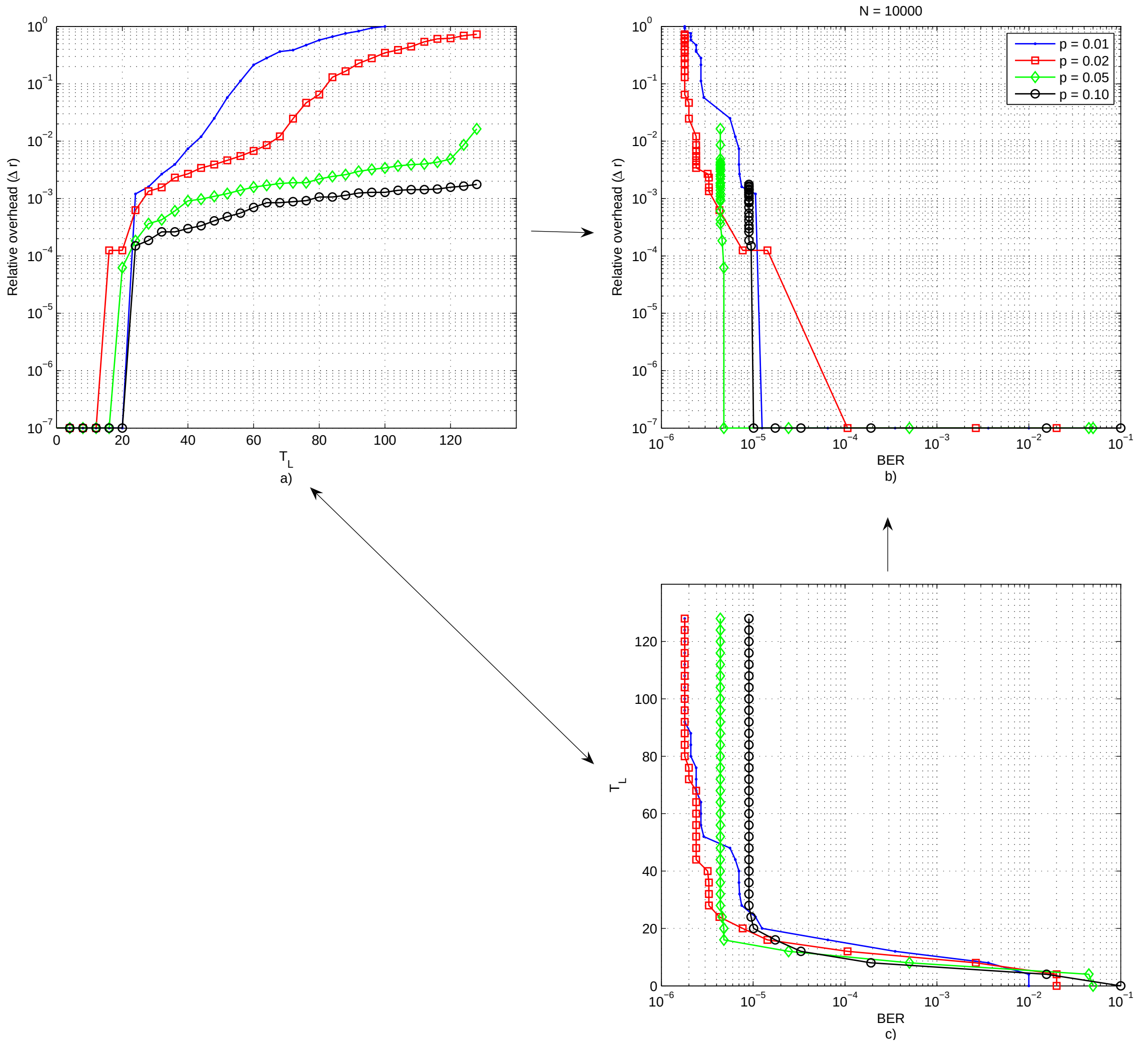

Figure 3: BER vs. Overhead. $N = 10000$, Stopping criterion: *C1*.

	Foreman				*Coastguard*			
B/b	1	2	3	4	1	2	3	4
1	0.060	-	-	-	0.030	-	-	-
2	0.044	0.055	-	-	0.027	0.055	-	-
3	0.035	0.057	0.082	-	0.023	0.050	0.085	-
4	0.035	0.056	0.081	0.073	0.020	0.044	0.084	0.083

Table 1: Initial BER for the first B encoded bit-planes.

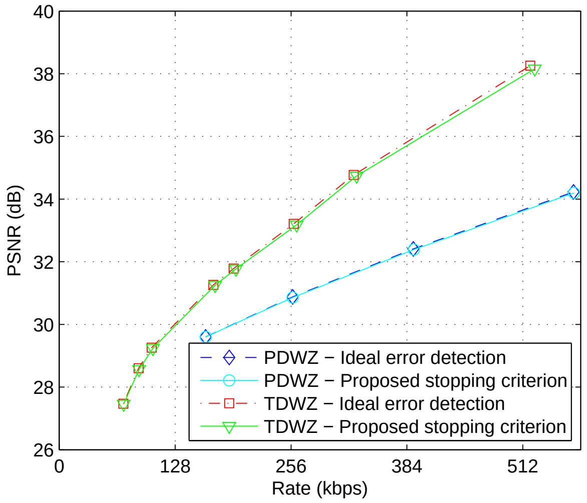

Figure 4: Rate-distortion curve of the *Foreman* sequence.

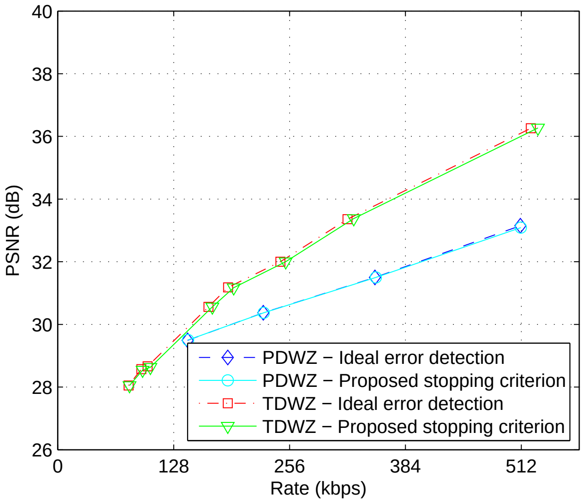

Figure 5: Rate-distortion curve of the *Coastguard* sequence.

correlation channel between the source and the side information is modeled as an additive Laplacian noise. At the bit-plane level, this means B parallel non-stationary BSC channels, one for each of the B bit-planes. The non-stationarity of the channels can be explained as follows: even if the variance of the Laplacian model is kept fixed, the a priori probabilities $Pr\{x_B^b(i) = 0|Y\}$ and $Pr\{x_B^b(i) = 1|Y\}$ depend on the full precision non-binary value of the side information Y, as well as on the quantizer step size, through b. Since the value of Y varies on a pixel-by-pixel basis, so do $Pr\{x_B^b(i) = 0|Y\}$ and $Pr\{x_B^b(i) = 1|Y\}$. Therefore, the values in Table 1 can be loosely interpreted as $E[BER(B,b)] = E[Pr\{x_B^b \neq y_B^b\}] \approx p$.

In the previous section we observed that for a target overhead, the residual BER is larger for smaller values of p. Therefore, Table 1 suggests that, if the proposed criterion is to be sub-optimal with respect to ideal error detection, a larger penalty in rate-distortion sense is to be expected for the most significant bit-plane ($b = 0$) at high bit-rates (i.e. $B = 2,3$). Figure 4 shows the rate distortion curve obtained for the *Foreman* sequence, confirming our observation. In this experiment, the threshold T_L has been set equal to 105. The value of T_L has been empirically selected based on simulations, in order to obtain the best rate-distortion performance. The lack of the originals at the decoder does not lead to a coding efficiency loss at low bit-rates. In addition, the loss is bounded to less than 0.1dB at high bit-rates, i.e. for $B = 3$. The same results are confirmed by the test on the *Coastguard* sequence in Figure 5. Also in this case, the loss is less than 0.1dB for $B = 3$.

5.3 TDWZ scenario

The TDWZ codec was configured to use GOP length of 2, and DCT quantization matrices as defined in [5]. The block size is 4×4, and therefore the sequence length N used in turbo coding is $176 \times 144/16 = 1584$. In Section 5.1 we observed that even for a shorter N, i.e. $N = 1000$, it was possible to achieve a residual BER in the 10^{-3} range at a negligible overhead. This fact is confirmed by Figure 4 and Figure 5, which show the rate-distortion curves for the TDWZ scenario both with and without ideal error detection. In this case the threshold T_L has been empirically set equal to 75. Besides the coding efficiency gain due to the exploitation of spatial redundancy, in the TDWZ case we observe the same behavior as in the PDWZ scenario. The coding efficiency loss due to the lack or originals at the decoder is kept below 0.1dB at high bit-rates.

6. CONCLUSIONS

In this paper we describe a practical algorithm that can be used to detect the correct number of parity bit requests in a Wyner-Ziv video coding architecture, without access to the original frames at the decoder. Results obtained for both the PDWZ and TDWZ coding architectures demonstrate that the coding efficiency loss is negligible at low bit-rates, while it is below 0.1dB at high bit-rates. Although in this paper we test the proposed method only in a feedback based architecture, ongoing work is investigating its applicability when the feedback channel is unavailable, i.e. in DVC-based error resilience applications.

REFERENCES

[1] Bernd Girod, Anne Aaron, Shantanu Rane, and David Rebollo Monedero, "Distributed video coding," *Proceedings of the IEEE*, vol. 93, pp. 71–83, January 2005.

[2] João Ascenso, Catarina Brites, and Fernando Pereira, "Interpolation with spatial motion smoothing for pixel domain distributed video coding," in *EURASIP Conference on Speech and Image Processing, Multimedia Communications and Services*, Slovak Republic, July 2005.

[3] W. Ryan, "A turbo code tutorial," 1997, http://citeseer.ist.psu.edu/ryan97turbo.html.

[4] Fengqin Zhai and I.J. Fair, "Techniques for early stopping and error detection in turbo decoding," *IEEE Transactions on Communications*, vol. 51, pp. 1617–1623, October 2003.

[5] Catarina Brites, Joao Ascenso, and Fernando Pereira, "Improving transform domain wyner-ziv video coding performance," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, Toulouse, France, May 2006.