Studying the GOP Size Impact on the Performance of a Feedback Channel-based Wyner-Ziv Video Codec

Fernando Pereira[1], João Ascenso[2], Catarina Brites[1]

[1] Instituto Superior Técnico – Instituto de Telecomunicações, Av. Rovisco Pais, 1049-001 Lisboa, Portugal
[2] Instituto Superior de Engenharia de Lisboa — Instituto de Telecomunicações, Rua Conselheiro Emídio Navarro, 1, 1950-062 Lisboa, Portugal
{fernando.pereira, joao.ascenso, catarina.brites}@lx.it.pt

Abstract. Wyner-Ziv video coding has become one of the hottest research topics in the video coding community due to the conceptual, theoretical and functional novelties it brings. Among the many practical architectures already available, feedback channel-based with channel coding, e.g. LDPC and turbo codes, solutions are rather popular. These solutions rely on decoder motion estimation based on periodic Intra coded key frames, setting the so-called GOP size, very much like in conventional video coding. This paper targets the rate-distortion and complexity performance study of this type of Wyner-Ziv coding solution as a function of the GOP size, considering both LPDC and turbo codes.

Keywords: Distributed video coding, Wyner-Ziv video coding, GOP size, LDPC coding, Turbo coding, RD performance, Complexity performance.

1 Introduction

Since the middle eighties, the video coding research community has been developing video codecs where it is the task of the encoder to exploit the data redundancy and irrelevancy to reach the compression factors necessary to deploy video coding needy applications and services. The most popular video codec architecture, the so-called motion compensated hybrid scheme, adopted by all MPEG and ITU-T video coding standards, relies on a combination of efficient motion-compensated temporal prediction and block-based transform coding, where encoders may become rather complex in comparison with decoders. This complexity balance is particularly suitable for asymmetric application topologies, such as digital television, video on demand, digital storage, and video streaming, where the content is typically coded once (and many times offline) and decoded many times or by many decoders.

The beginning of this decade saw the emergence of a new video coding paradigm, the so-called distributed video coding [1], challenging the 'traditional' coding model since it proposes to fully or partly exploit the video data redundancy at the decoder and not anymore at the encoder. This new coding approach is based on the Slepian-Wolf and Wyner-Ziv theorems which basically state, for the lossless and lossy coding cases, that, under certain conditions, the same compression can be achieved with both

the joint-encoding and joint-decoding paradigm and the distributed/independent-encoding and joint-decoding paradigm [1]. Wyner-Ziv (WZ) video coding regards the lossy coding of two correlated sources where the source X is coded without access to the correlated source Y, assumed to be available at the decoder to perform joint decoding. This new paradigm implies the data correlation is mainly exploited at the decoder, enabling low-complexity video encoding, where the core of the computation, notably motion processing, is shifted to the decoder.

Following important developments in the channel coding domain, the first practical implementations of the distributed video coding paradigm, notably Wyner-Ziv video coding solutions, appeared around 2002 [1,2,3]. One of the most popular Wyner-Ziv video coding architectures relies on a feedback channel to perform rate control at the decoder and uses some channel codes, e.g. turbo codes or Low-Density Parity-Check (LDPC) codes, to 'correct' the errors in the frame estimations (designated as side information), created at the decoder after motion estimation, for the frames to be Wyner-Ziv encoded (X source). In this architecture, the full video sequence is divided into the so-called key frames (Y source) and the so-called WZ frames (X source), with the key frames appearing in a periodic way and setting the so-called GOP (Group of Pictures) size, as in conventional video coding with I frames versus P and B frames.

Since the key frames serve for the decoder to create the estimations of the WZ frames based on motion estimation, the GOP size has a very significant impact on the overall performance of the WZ video codec (even more than for traditional video coding) since it strongly determines the goodness of the motion estimation, and thus the quality of the side information, and the number of 'estimation errors' that have to be corrected by requesting (channel coding) parity bits to the WZ encoder through the feedback channel.

While most WZ video coding performance results in the literature regard a GOP size of 2 (the simplest case in terms of decoder motion estimation), it is well know that higher GOP sizes are relevant and interesting since temporal redundancy should be exploited more than once every two frames. In fact, conventional video coding tells that the RD performance limits improve with the GOP size and the overall WZ encoding complexity decreases with the GOP size which are two simultaneously positive trends that have to be exploited notably for some encoder complexity and battery constrained applications. However, longer GOP sizes are particularly challenging for WZ video coding since the motion estimation becomes more difficult, and thus the side information poorer, especially for more active video content.

In this context, the main target of this paper is to evaluate in detail the current performance of an advanced WZ video codec in terms of the GOP size used for the key frames versus WZ frames splitting. This performance evaluation will not only consider the rate-distortion (RD) performance but also the complexity performance since WZ coding is deeply related to additional complexity budget flexibility; in fact, theoretically, the compression efficiency can, at most, be the same as for conventional video coding.

While Section 2 briefly describes the WZ video codec used for this study, Section 3 details the performance evaluation for several relevant metrics always as a function of the GOP size. Finally, Section 4 summarizes and concludes this paper.

2 DISCOVER Wyner-Ziv Video Codec

The WZ video codec evaluated in this paper has been improved by the DISCOVER project team based on a first codec developed by the authors of this paper at Instituto Superior Técnico [4]. The DISCOVER WZ video codec architecture, illustrated in Figure 1, is based on the basic WZ video coding architecture proposed in [3], and it is presented in detail in [5]. However, at this stage, the initial architecture has already evolved, e.g. adding a Cyclic Redundancy Check code (CRC) and some encoder rate control capabilities, and most of the tools in the various modules are different (and globally more efficient).

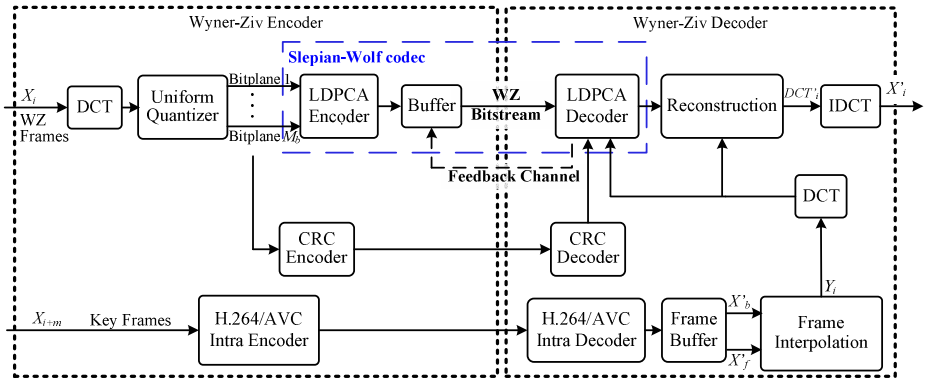


Fig. 1. DISCOVER video codec architecture.

The DISCOVER WZ video codec works as follows:

At the encoder:

1. First, a video sequence is divided into Wyner-Ziv (WZ) frames, this means the frames that will be coded using a WZ approach, and key frames as in the original WZ architecture adopted as the basis of the DISCOVER codec [1, 3]. The key frames are encoded as Intra frames, e.g. using the H.264/AVC Intra codec [6], and may be inserted periodically with a certain Group of Pictures (GOP) size. An adaptive GOP size selection process may also be used. In the latter case, the key frames are inserted depending on the amount of temporal correlation in the video sequence [7]. Most results available in the literature use a GOP size of 2 which means that odd and even frames are key frames and WZ frames, respectively. This paper targets precisely the study of the WZ codec performance impact when changing the GOP size.
2. Over each Wyner-Ziv frame X_{WZ}, a 4×4 block-based Discrete Cosine Transform (DCT) is applied. The DCT coefficients of the entire frame X_{WZ} are then grouped together, according to the position occupied by each DCT coefficient within the 4×4 blocks, forming the DCT coefficients bands.
3. After the transform coding operation, each DCT coefficients band b_k is uniformly quantized with $2M_k$ levels (where the number of levels $2M_k$ depends on the DCT

coefficients band b_k). Over the resulting quantized symbol stream (associated to the DCT coefficients band b_k), bitplane extraction is performed. For a given band, the quantized symbols bits of the same significance (e.g. the most significant bit) are grouped together, forming the corresponding bitplane array which is then independently LDPC (or turbo) encoded [8].

4. The LDPC (or turbo) coding procedure for the DCT coefficients band b_k starts with the most significant bitplane array, which corresponds to the most significant bits of the b_k band quantized symbols. The parity information generated by the LDPC encoder for each bitplane is then stored in the buffer and sent in chunks/packets upon decoder request, through the feedback channel.
5. In order to limit the number of requests to be made by the decoder, and thus the decoding complexity (since each request corresponds to several LDPC decoder iterations), the encoder estimates for each bitplane the initial number of bits to be sent before any request is made [5]. This number should be an underestimation of the final number of bits which means there should be no RD performance losses associated to this step (regarding the case where no initial estimation is made).

At the decoder:

6. The decoder creates the so-called side information for each WZ coded frame by performing a frame interpolation process using the previous and next temporally closer key frames of X_{WZ} to generate an estimate of frame X_{WZ}, Y_{WZ} [9]. The side information for each WZ frame intends to be an estimate of the original WZ frame; the better it is this estimation, the smaller are the number of 'errors' the Wyner-Ziv LDPC (or turbo) decoder has to correct and the bitrate used for that.
7. A block-based 4×4 DCT is then carried out over Y_{WZ} in order to obtain Y_{WZ} DCT, an estimate of X_{WZ} DCT. The residual statistics between correspondent coefficients in X_{WZ} DCT and Y_{WZ} DCT is assumed to be modeled by a Laplacian distribution. The Laplacian parameter is estimated online and at different granularity levels, notably at band and coefficient levels.
8. Once Y_{WZ} DCT and the residual statistics for a given DCT coefficients band b_k are known, the decoded quantized symbol stream q'_{WZ} associated to the DCT band b_k can be obtained through the LDPC decoding procedure. The LDPC (or turbo) decoder receives from the encoder successive chunks of parity bits following the requests made through the feedback channel.
9. To decide whether or not more bits are needed for the successful decoding of a certain bitplane, the decoder uses a request stopping criteria based on the LDPC code parity check equations. If no more bits are needed to decode that bitplane, the decoding of the next band can start; otherwise, the bitplane LDPC decoding task has to proceed with another request and receive another chunk of parity bits.
10. After successfully LDPC (or turbo) decoding the most significant bitplane array of the b_k band, the LDPC (or turbo) decoder proceeds in an analogous way to the remaining M_{k-1} bitplanes associated to that band. Once all the bitplane arrays of the DCT coefficients band b_k are successfully LDPC (or turbo) decoded, the LDPC (or turbo) decoder starts decoding the b_{k+1} band. This procedure is repeated until all the DCT coefficients bands for which WZ bits are transmitted are LDPC (or turbo) decoded.

11. Because the estimation of the bitplane error probability is not perfect, a CRC check sum is transmitted to help the decoder detect and correct the remaining errors in each bitplane since they have a rather negative subjective impact. Since this CRC is combined with the developed request stopping criteria, it does not have to be very strong. As a consequence, an 8 bit CRC check sum per bitplane is found to be strong enough for this purpose which only adds minimal extra rate.
12. After LDPC (or turbo) decoding the M_k bitplanes associated to the DCT band b_k, the bitplanes are grouped together to form the decoded quantized symbol stream associated to the b_k band. This procedure is performed over all the DCT coefficients bands to which WZ bits are transmitted.
13. Once all decoded quantized symbol streams are obtained, it is possible to reconstruct the matrix of DCT coefficients, X'_{WZ} DCT. The DCT coefficients bands for which no WZ bits were transmitted are replaced by the corresponding DCT bands of the side information, Y_{WZ} DCT.
14. After all DCT coefficients bands are reconstructed, a block-based 4×4 inverse discrete cosine transform (IDCT) is performed and the reconstructed X_{WZ} frame, X'_{WZ}, is obtained.
15. To, finally, get the decoded video sequence, decoded key frames and WZ frames are mixed conveniently.

It is important to stress that the DISCOVER WZ video codec does not include any of the limitations which are many times present in WZ papers, notably those adopting this type of WZ architecture [1]. This means, for example, that no original frames are used at the decoder to create the side information, to measure the bitplane error probability or to estimate the noise correlation model parameters for LDPC (or turbo) decoding.

2.1 Transform and Quantization

Different RD performances can be achieved by changing the M_k value for the DCT band b_k. In this paper, eight rate-distortion points are considered corresponding to the various 4×4 quantization matrices depicted in Fig. 2. Within a 4×4 quantization matrix, the value at position k in Fig. 2 indicates the number of quantization levels associated to the DCT coefficients band bk; the value 0 means that no Wyner-Ziv bits are transmitted for the corresponding band. In the following, the various matrices will be referred as Qi with i= 1, …, 8; the higher is Qi, the higher are the bitrate and the quality.

2.2 Slepian-Wolf Coding

The DISCOVER WZ video codec has, finally, adopted LDPC codes for the Slepian-Wolf part of the WZ codec after using for a long time turbo codes. The literature states that LDPC codes can better approach the capacity of a variety of communication channels than turbo codes [8].

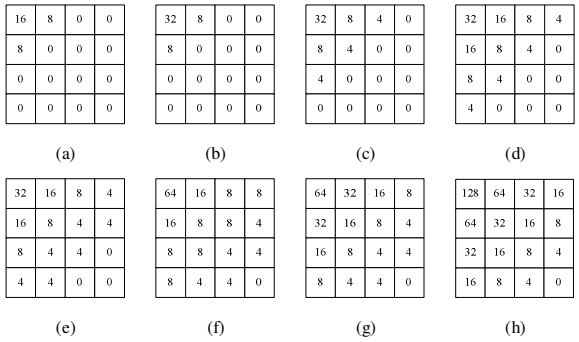


Fig. 2. Eight quantization matrices associated to different RD performances (and qualities).

The DISCOVER codec uses a LDPC Accumulate (LDPCA) codec which consists of an LDPC syndrome-former concatenated with an accumulator [8]. For each bitplane, syndrome bits are created using the LDPC code and accumulated modulo 2 to produce the accumulated syndrome. The Wyner-Ziv encoder buffers these accumulated syndromes and transmits them to the decoder in chunks, upon decoder request.

Previously, the DISCOVER codec has used turbo codes. The turbo encoder enclosed a parallel concatenation of two identical constituent recursive systematic convolutional (RSC) encoders of rate ½ and a pseudo-random L-bit interleaver was employed to decorrelate the L-bit input sequence between the two RSC encoders. The Slepian-Wolf decoder enclosed an iterative turbo decoder constituted by two soft-input soft-output (SISO) decoders. Each SISO decoder was implemented using the Logarithmic Maximum *A Posteriori* (Log-MAP) algorithm. A confidence measure based on the *a posteriori* probabilities ratio is used as error detection criteria to determine the current bitplane error probability P_e of a given DCT band. If P_e is higher than 10^{-3}, the decoder requests for more parity bits from the encoder via the feedback channel; otherwise, the bitplane turbo decoding task is considered successful.

2.3 Frame Interpolation or Side Information Generation

The side information creation process is rather complex and central for the performance of the Wyner-Ziv codec since it determines how many 'errors' have to be corrected though LDPC (or turbo) parity bits. This process is described in detail in [9]. In this section, the most important control parameters related to the side information creation process are presented. For the frame interpolation in the side information creation process, two block sizes are used: 16×16 and 8×8. The forward motion estimation works with 16×16 block size and ±32 pixels are used for the search range. In the second iteration, the motion vectors are refined using 8×8 block sizes and an adaptive search range is used. The motion search is performed using the half-pixel precision and the reference frames are first low pass filtered with the mean filter using a 3×3 mask size.

3. GOP Size Dependent Performance Evaluation

As mentioned before, the main purpose of this paper is to evaluate the RD and complexity performance of a rather advanced WZ video codec as a function of the GOP size. For this purpose, it is essential to define first the performance evaluation conditions. As usual in the WZ literature, only the luminance component is coded and thus all metrics in this paper refer only to the luminance.

Although performance results are available for many video sequences, results in this paper will be presented only for two rather different sequences: Foreman (with the Siemens logo), and Coast Guard. For both sequences, 299 frames are coded at QCIF resolution, 15 Hz. The key frames quantization steps have been found using an iterative process which stops when the average quality (PSNR) of the WZ frames is similar to the quality of the Intra frames (H.264/AVC Intra encoded).

3.1 RD Performance

Although many metrics are relevant to evaluate the RD performance, it is recognized that the most used quality metric is the average PSNR (with all limitations it brings) over all the frames of a sequence coded for a certain quantization matrix. When this PSNR metric is represented as a function of the used bitrate – in this case, the overall bitrate which includes all WZ and key frames bits for the luminance component – very important performance charts are obtained since they allow to easily comparing the overall rate-distortion (RD) performance with other coding solutions, including standard coding solutions largely well known and used. In this paper, the RD performance of the DISCOVER codec will be compared with the corresponding performance of three standard coding solutions which share an important property in terms of encoder complexity: the complex and expensive motion estimation task is not performed by any of them. Sections 3.3 and 3.4 will present encoder and decoder complexity evaluations which complement the RD evaluation proposed in this section.

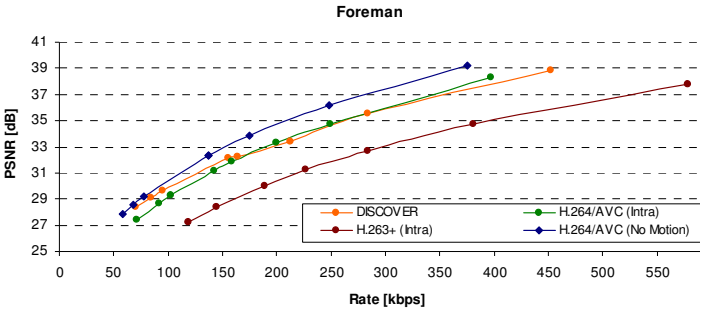
The three standard video coding solutions used here for comparison purposes are:

- **H.263+ Intra** [10] – Coding with H.263+ without exploiting temporal redundancy; this is a rather old codec although still much used in the literature for comparison with WZ codecs because it is 'easier to beat' in comparison to the H.264/AVC Intra codec.
- **H.264/AVC Intra** [6] – Coding with H.264/AVC in Main profile without exploiting temporal redundancy; this type of Intra coding is among the most efficient Intra (video) coding standard solutions available, even more than JPEG2000 in many cases. However, notice that the H.264/AVC Intra codec exploits quite efficiently the spatial correlation (at a higher complexity cost when compared to H.263+ Intra) with several 4×4 and 16×16 Intra modes, a feature that is also (still) missing in the DISCOVER WZ codec.
- **H.264/AVC Inter No Motion** [6] – Coding with H.264/AVC in Main profile exploiting temporal redundancy in a IB… structure but without performing any motion estimation which is the most computationally expensive encoding task. The so-called "no motion" mode achieves better performance than Intra coding

because it can partly exploit temporal redundancy but it requires far less complexity than full motion compensated Inter coding since no motion search is performed. This type of comparison (excluding encoder motion estimation as in WZ coding) is not typically provided in most WZ published papers because its RD performance is still rather difficult to 'beat' with WZ coding solutions.

While Fig. 3 shows the RD performance for the various video codecs tested, Fig. 4 shows a RD performance comparison for various GOP sizes, notably 2, 4 and 8. The main conclusions that can be drawn from these charts are:

- For the sequences here tested (and many others), the WZ codec at GOP 2 can always beat the H.263+ Intra RD performance. The same happens for H.264/AVC Intra with the DISCOVER WZ always beating or equaling the H.264/AVC Intra RD performance (although the H.264/AVC Intra encoding complexity is much higher as it will be shown later). For the Coast Guard sequence, the DISCOVER WZ codec can even beat the H.264/AVC No Motion RD performance which may be explained by the fact the WZ codec is performing motion estimation at the decoder while the H.264/AVC No Motion codec is not doing it at the encoder.
- For the selected test sequences, the highest RD performance always happens for GOP 2 showing the difficulty in getting good side information for longer GOP sizes due to the decreased quality of the frame interpolation (key frames are further away). However, Fig. 4 shows RD performance results for the Hall Monitor sequence, which is a rather stable sequence, where GOP 4 is already more efficient than GOP 2 since motion estimation and frame interpolation is now more reliable.



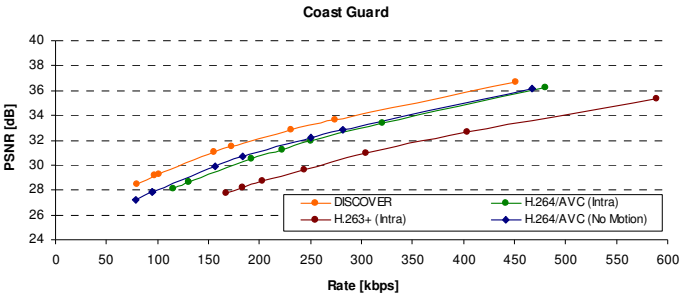
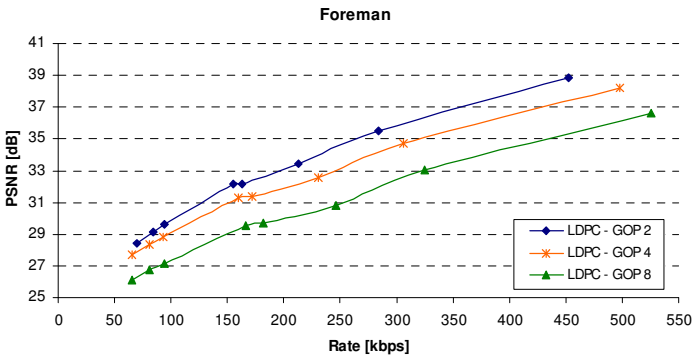
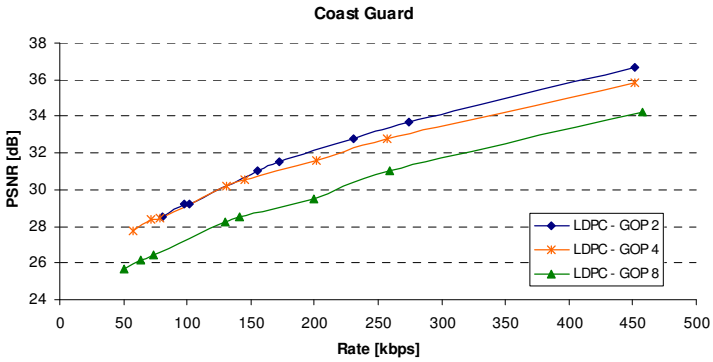


Fig. 3. RD performance for GOP 2: Foreman, and Coast Guard.





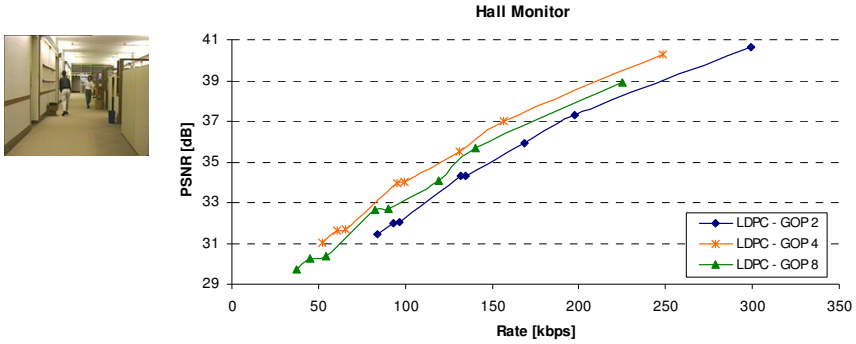


Fig. 4. RD performance comparison between GOP 2, 4 and 8: Foreman, Coast Guard and Hall Monitor.

3.2 LDPC versus Turbo Codes RD Performance

This section compares the RD performances of the DISCOVER codec when using turbo codes and LDPC codes in the Slepian-Wolf codec, for equal conditions in terms of all other modules. Fig. 5 shows the RD performance comparison which highlights the fact that LDPC codes have always better RD performance than turbo codes, for all GOP sizes and all sequences tested. More in detail, the following conclusions can be drawn:

- For lower bitrates, the performance of the turbo and LDPC codes is quite similar since when the correlation between the SI and the (quantized) WZ frames is high, the turbo codes achieve a RD performance similar to the LPDC codes.
- At medium and high bitrates, the LDPC codes have a better RD performance when compared to the turbo codes (with coding gains up to 35 kbps for GOP size equal to 8). The LDPC codes show better performance for the bands/bitplanes which have a lower correlation between the SI and WZ data (i.e. for side information with lower quality regarding the WZ frames).
- When the GOP size increases, the performance gap between the turbo codes and the LDPC codes increases with a clear advantage for the LDPC codes (for GOP size 2, 4, and 8, coding gains up to 10, 27, 35 kbps occur, respectively). One major reason for this effect is that the LDPC codes have always a maximum rate of 1 for any bitplane, i.e. the maximum number of syndrome bits sent cannot exceed the number of bits that represent the original data. This property does not exist for the iterative turbo codes (where rate expansion is possible) and, thus, it is responsible for the turbo codes loss of efficiency, especially at larger GOP sizes, where the correlation between the WZ frames and the side information is lower, and, thus, compression rates higher than 1 happen for successful decoding.

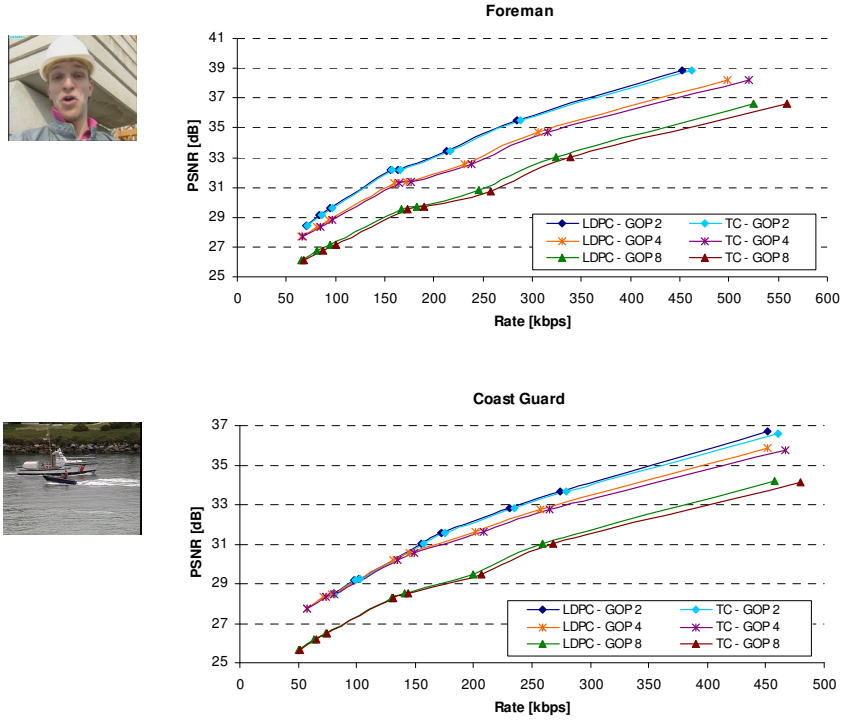


Fig. 5. RD performance comparison for LDPC versus turbo codes for various GOP sizes.

3.3 Encoding Complexity Performance

In this paper, the encoding (and decoding) complexity will be measured by means of the encoding (and decoding) time for the full sequence, in seconds, under controlled conditions. For the present results, the hardware used was an x86 machine with a dual core Pentium D processor, at 3.4 GHz, with 2048 MB of RAM. Regarding the software conditions, the results were obtained with a Windows XP operative system, with the C++ code written using version 8.0 of the Visual Studio C++ compiler, with optimizations parameters on, such as the release mode and speed optimizations. Since the encoding (and decoding) time results are highly dependent on the used hardware and software platforms, they have a relative and comparative value, in this case allowing comparing the DISCOVER codec with alternative solutions, e.g. H.264/AVC based, running in the same hardware and software conditions.

Fig. 6 and Table 1 and Table 2 show the encoder complexity results for GOP 2, 4 and 8, measured in terms of encoding time distinguishing between key frames (blue) and WZ frames (red) encoding times. The results allow concluding that:

- For the DISCOVER video codec, the WZ encoding complexity is negligible when compared to the key frames coding complexity, even for GOP 2. For longer GOP sizes, the overall encoding complexity decreases with the increase of the share of WZ frames regarding the key frames; in this case, the number of key frames decreases, although their encoding complexity is still the dominating part.
- The DISCOVER encoding complexity is always much lower than the H.264/AVC encoding complexity, both for the H.264/AVC Intra and H264/AVC no Motion solutions. While the H.264/AVC Intra encoding complexity does not vary with the GOP size and the H.264/AVC no Motion encoding complexity is also rather stable with a varying GOP size, the DISCOVER encoding complexity decreases with the GOP size. If encoding complexity is a critical requirement, the results in this section, together with the RD performance results previously shown, indicate that the DISCOVER monoview codec with GOP 2 is already a credible practical solution since it has a rather low complexity and 'beats' H.264/AVC Intra in terms of RD performance for most cases. This is a rather important result …
- Another important result is that the WZ encoding complexity does not increase significantly when the Qi increases (i.e. when the bitrate increases); on the other hand, for H.264/AVC Intra and H264/AVC no Motion, the complexity increases when higher bitrates are targeted.
- The encoding complexity is rather similar for the LDPC and turbo coding alternatives for all GOP sizes.

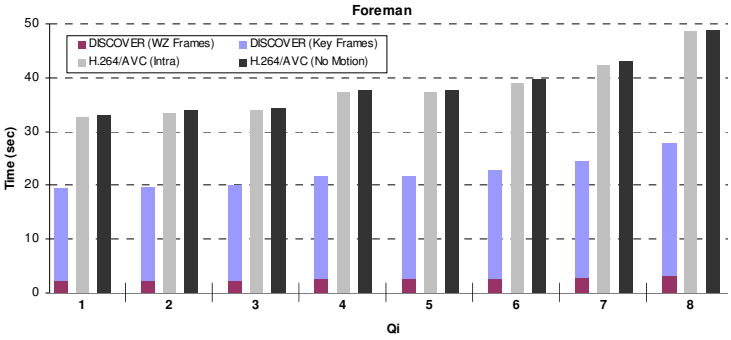


Fig. 6. Encoder complexity measured in terms of encoding time for GOP 2: Foreman sequence.

Table 1. Encoding time (full sequence in seconds) comparison for the Foreman sequence.

	H.264/AVC				DISCOVER						Ratios		
	Intra	No Motion			Using LDPC			Using Turbo Codes			H.264/AVC Intra vs DISCOVER LPDC		
QP		GOP 2	GOP 4	GOP 8	GOP 2	GOP 4	GOP 8	GOP 2	GOP 4	GOP 8	GOP 2	GOP 4	GOP 8
40	32.67	33.14	33.64	32.97	19.44	11.58	8.02	19.14	11.26	7.58	1.68	2.82	4.08
39	33.30	33.81	34.41	33.72	19.75	11.80	8.09	19.47	11.42	7.69	1.69	2.82	4.11
38	33.78	34.34	34.88	34.15	20.05	11.95	8.22	19.73	11.57	7.79	1.69	2.83	4.11
34	37.11	37.60	38.21	37.34	21.67	12.94	8.79	21.39	12.48	8.33	1.71	2.87	4.22
34	37.13	37.61	38.29	37.34	21.68	13.00	8.80	21.39	12.53	8.34	1.71	2.86	4.22
32	38.94	39.55	40.18	39.23	22.78	13.66	9.37	22.38	13.14	8.79	1.71	2.85	4.15
29	42.33	43.11	43.73	42.72	24.49	14.58	9.94	24.13	14.07	9.34	1.73	2.90	4.26
25	48.56	48.89	49.54	48.17	27.72	16.61	11.23	27.36	15.98	10.55	1.75	2.92	4.32

Table 2. Encoding time (full sequence in seconds) comparison for the Coast Guard sequence.

	H.264/AVC				DISCOVER						Ratios		
	Intra	No Motion			Using LDPC			Using Turbo Codes			H.264/AVC Intra vs DISCOVER LPDC		
QP		GOP 2	GOP 4	GOP 8	GOP 2	GOP 4	GOP 8	GOP 2	GOP 4	GOP 8	GOP 2	GOP 4	GOP 8
37	39.30	39.27	39.55	38.58	23.05	13.71	9.19	22.85	13.21	8.62	1.70	2.87	4.28
36	39.70	40.00	40.25	39.42	23.31	13.86	9.41	23.15	13.37	8.81	1.70	2.87	4.22
36	39.73	40.08	40.26	39.44	23.34	13.86	9.42	23.17	13.38	8.82	1.70	2.87	4.22
33	42.22	42.66	43.06	42.17	24.65	14.72	9.95	24.54	14.22	9.39	1.71	2.87	4.24
33	42.29	42.77	43.16	42.20	24.73	14.72	9.97	24.57	14.23	9.41	1.71	2.87	4.24
31	44.30	44.83	44.97	44.02	25.89	15.52	10.50	25.82	14.90	9.83	1.71	2.85	4.22
29	46.22	46.91	47.41	46.38	26.96	16.12	10.94	26.71	15.50	10.28	1.71	2.87	4.23
24	52.58	53.50	54.09	53.09	30.32	18.17	12.50	29.97	17.49	11.59	1.73	2.89	4.21

3.4 Decoding Complexity Performance

Table 3 and Table 4 show the decoder complexity results for GOP 2, 4 and 8, measured in terms of decoding time. The results allow concluding that:

- For the DISCOVER video codec, the key frame decoding complexity is negligible regarding the WZ frames coding complexity, even for GOP 2 (when there are more key frames). This confirms the well known WZ coding trade-off where the encoding complexity benefits are paid in terms of decoding complexity. Contrary to the encoding complexity, the longer is the GOP size, the higher is the overall decoding complexity since the higher is the number of WZ frames.
- The DISCOVER decoding complexity is always much higher than the H.264/AVC decoding complexity, both for the H.264/AVC Intra and H264/AVC no Motion solutions. While the H.264/AVC Intra decoding complexity does not vary with the GOP size and the H.264/AVC no Motion decoding complexity is also rather stable with a varying GOP size, the DISCOVER decoding complexity increases with the GOP size.

- The WZ decoding complexity increases significantly when the Qi increases (i.e. when the bitrate increases) since the number of bitplanes to LDPC (or turbo) decode is higher and the LDPC (or turbo) decoder (and the number of times that is runs) is the main responsible for the decoding complexity.
- Regarding the decoding complexity comparison between LDPC and turbo codes, the results seem to say that while LDPC wins for more quiet sequences, e.g. Coast Guard (and Hall Monitor) for GOP 2, turbo codes win for sequences with more motion, e.g. Foreman (and Soccer).

Table 3. Decoding time (full sequence in seconds) comparison for the Foreman sequence.

H.264/AVC					DISCOVER					
	Intra	No Motion			Using LDPC			Using Turbo Codes		
QP		GOP 2	GOP 4	GOP 8	GOP 2	GOP 4	GOP 8	GOP 2	GOP 4	GOP 8
40	1.55	1.53	1.53	1.50	664.06	1150.11	1486.70	590.47	983.92	1237.19
39	1.55	1.55	1.53	1.52	729.45	1237.47	1605.34	680.80	1124.59	1402.70
38	1.58	1.55	1.58	1.53	848.45	1482.45	1904.23	768.25	1280.75	1606.08
34	1.64	1.66	1.66	1.64	1362.45	2536.06	3293.84	1219.88	2105.47	2663.94
34	1.66	1.67	1.69	1.64	1541.00	2824.58	3641.94	1346.77	2319.48	2930.91
32	1.72	1.73	1.77	1.70	2041.53	3640.48	4586.58	1765.06	3030.48	3808.64
29	1.83	1.83	1.84	1.81	2352.56	4273.28	5551.36	2148.23	3738.67	4728.45
25	1.92	1.97	1.98	1.94	3254.92	5901.63	7640.66	3207.05	5535.03	6974.56

Table 4. Decoding time (full sequence in seconds) comparison for the Coast Guard sequence.

H.264/AVC					DISCOVER					
	Intra	No Motion			Using LDPC			Using Turbo Codes		
QP		GOP 2	GOP 4	GOP 8	GOP 2	GOP 4	GOP 8	GOP 2	GOP 4	GOP 8
38	1.55	1.56	1.56	1.53	430.27	709.75	986.66	440.89	691.36	890.94
37	1.58	1.61	1.56	1.56	485.89	776.86	1048.20	512.61	796.70	994.88
37	1.61	1.64	1.58	1.57	531.17	894.77	1226.23	579.48	908.95	1150.61
34	1.64	1.66	1.70	1.66	796.69	1525.11	2168.36	874.58	1463.97	1885.91
33	1.70	1.75	1.77	1.69	824.23	1628.22	2329.80	931.80	1564.95	2024.33
31	1.72	1.81	1.78	1.80	1144.45	2254.47	3193.66	1229.36	2093.36	2708.59
30	1.75	1.88	1.81	1.81	1497.25	2802.48	3856.28	1566.00	2668.41	3375.45
26	1.92	2.05	2.03	1.94	2461.73	4462.38	5872.11	2500.95	4309.33	5428.16

4. Final Remarks

This paper presents a detailed evaluation of the RD and complexity performance of an advanced feedback-channel based Wyner-Ziv video codec as a function of the GOP size. While longer GOP sizes increase the RD performance theoretical upper bounds, as shown by conventional video coding, due to the more intensive exploitation of the temporal redundancy (even if at the cost of higher encoder complexity), this is still not happening for the tested WZ video codec since the longer is the GOP size the more difficult and less reliable is the motion estimation process at the decoder,

reducing the quality of the side information estimate. This fact asks for the development of more sophisticated frame interpolation methods a the decoder, eventually combined with having the encoder sending to the decoder some auxiliary data for the more motion critical parts of the frame, e.g. local hash codes [11], even at the cost of some encoder complexity. It is, however, important to stress that WZ video coding already proposes competitive solutions for application scenarios where encoding complexity is the main critical requirement since it RD performs the best for the lowest encoder complexity in comparison with standard alternative solutions.

Acknowledgments. The work presented here was developed within DISCOVER, a European Project (http://www.discoverdvc.org), funded under the European Commission IST FP6 programme.

References

1. B. Girod, A. Aaron, S. Rane and D. Rebollo Monedero, "Distributed Video Coding", Proceedings of the IEEE, vol. 93, nº 1, pp. 71 - 83, January 2005.
2. R. Puri and K. Ramchandran, "PRISM: A New Robust Video Coding Architecture Based on Distributed Compression Principles", 40th Allerton Conference on Communication, Control and Computing, Allerton, USA, October 2002.
3. A. Aaron, R. Zhang and B. Girod, "Wyner-Ziv Coding of Motion Video", Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, November 2002
4. C. Brites, J. Ascenso, F. Pereira, "Improving Transform Domain Wyner-Ziv Video Coding Performance", International Conference on Acoustics, Speech, and Signal Processing, Toulouse, France, May 2006.
5. X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov, M. Ouaret, "The DISCOVER Codec: Architecture, Techniques and Evaluation", Picture Coding Symposium, Lisbon, Portugal, November 2007.
6 ISO/IEC 14496-10:2003, "Coding of Audio-Visual Objects – Part 10: Advanced Video Coding", 1st Edition, July 2003 (also ITU-T: 2003, H.264).
7 J. Ascenso, C. Brites, F. Pereira, "Content adaptive Wyner-Ziv Video Coding Driven by Motion Activity", Int. Conf. on Image Processing, Atlanta - USA, October 2006.
8. D. Varodayan, A. Aaron and B. Girod, "Rate-Adaptive Codes for Distributed Source Coding", EURASIP Signal Processing Journal, Special Issue on Distributed Source Coding, pp. 3123 - 3130, vol. 86, nº 11, Nov. 2006.
9. J. Ascenso, C. Brites, and F. Pereira, "Improving Frame Interpolation with Spatial Motion Smoothing for Pixel Domain Distributed Video Coding", 5th EURASIP Conf. on Speech, Image Processing, Multimedia Communications and Services, Smolenice, Slovak Republic, July 2005.
10. ITU-T: 1998, H.263+, "Video Coding for Low Bitrate Communication", ITU-T Recommendation H.263, Version 2, January 1998.
11. J. Ascenso, F. Pereira, "Adaptive Hash-based Side Information Exploitation for Efficient Wyner-Ziv Video Coding", Int. Conf. on Image Processing (ICIP'2007), San Antonio, TX, USA, September 2007.