EUSIPCO 2013 1569742581

IMPROVING DECODING SPEED FOR PARALLEL DISTRIBUTED VIDEO CODING ARCHITECTURES

Jeffrey J. Micallef, Reuben A. Farrugia and Carl J. Debono

Department of Communication and Computer Engineering, University of Malta, Msida, Malta
jeffrey.micallef@ieee.org, reuben.farrugia@um.edu.mt, c.debono@ieee.org

ABSTRACT

The Distributed Video Coding (DVC) paradigm is suitable for devices which have limited encoding capabilities. However, it is characterized by excessive decoding delays which compromise their application for time constrained services. This limitation can be mitigated by adopting parallel DVC architectures. Yet, the traditional Gray-code or binary-code representations have a non-uniform distribution of mismatch across bit-planes, resulting in uneven decoding times which hinder parallel decoding. This work proposes an alternative indexing scheme, where mismatch is distributed more uniformly amongst bit-planes and thus comparable decoding delays are expected, facilitating parallel implementations. This method reduces decoding time by up to 32% compared to architectures using simple parallel techniques, with a slight loss of 0.06dB in RD performance.

Index Terms— Distributed video coding, index assignment, parallel architecture, source representation, Wyner-Ziv coding.

1. INTRODUCTION

The traditional video coding schemes vest in the encoder to exploit complex Motion Compensation (MC) and Rate-Distortion (R-D) optimization techniques and achieve considerable compression efficiencies. The standard encoders are thus significantly more complex than the decoder [1] and not ideal for mobile devices or miniature cameras which have limited computational resources. Distributed Video Coding (DVC) is a promising encoding strategy which was widely investigated in the last decades. The DVC paradigm proposes to shift most of the complex tasks involved in exploring the source redundancies from the encoder to the decoder. Based on the Wyner-Ziv (WZ) theorem [2], this shift should not affect the theoretical coding efficiencies compared to standard encoding schemes.

The DVC paradigm transfers the computational expensive motion compensation process from the encoder to the decoder, and predicts the WZ frames from the adjacent key-frames. This prediction is then used as Side Information (SI) to aid compression, by employing Turbo-Codes or Low-Density Parity-Check Accumulate (LDPCA) codes which can correct the SI and reconstruct the original WZ frame using only a sub-set of parity information, hence obtaining compression. However, these channel codes use belief propagation algorithms which require a large number of floating point calculations to be repeated iteratively until they converge. Furthermore, since the quality of the SI is unpredictable, the WZ decoder has to run these codes several times for each bit-plane, requesting additional parity bits until it manages to correct the bit-plane successfully. This makes DVC architectures suffer from an excessive decoding delay relative to traditional video coding schemes.

Several techniques have been proposed to alleviate some of the complexity issue for DVC decoding and make it suitable for more practical scenarios. Early stopping criterions for LDPC codes and Turbo Codes were proposed in [3]-[5] and [6]-[7], respectively. These methods could determine, as early as possible, whether the transmitted parity bits are sufficient for the channel decoder to converge successfully. On the other hand, the authors in [8] adopt a hybrid rate control mechanism, where the encoder promptly transmits a minimum number of parity bits to prevent useless attempts of recovering the WZ bits using very few parity bits. Parallelized message-passing decoding algorithm for LDPCA codes, on a General Purpose Graphics Processing Unit (GPGPU) were considered in [9]-[10] to speed up convergence of LDPCA codes. Yet, an optimized decoder implementation which can achieve desirable performance on a multi-core processor, without using specialized hardware like GPGPU, still needs to be addressed [11].

The authors in [12] considered a parallel DVC architecture, where multiple bit-planes are decoded simultaneously on a multi-core CPU. Yet, the Gray-code or binary code representations used for these architectures have a non-uniform probability of mismatch across bit-planes, thus the Lower Significant Bit-planes (LSB) need more parity information and take longer to recover. For parallel DVC architectures, these uneven decoding times lead to a lower utilization since some processors are left idle waiting for the LSB to be decoded, reducing the potentially achievable speed up. This work presents an indexing scheme which can distribute the probability of mismatch more uniformly across bit-planes, with a minimal affect on the coding performance. All bit-planes can thus be recovered at approximately the same time, minimizing decoding delays. Simulation results show that the proposed method can reduce decoding delays by up to 32% compared to other parallel schemes [12], with a slight loss of 0.06dB in RD performance.

This paper is organized as follows: the transform domain Wyner-Ziv video coding architecture is introduced in Section 2. Section 3 looks at the proposed index assignment scheme, compares the probability of mismatch at each bit-plane with those of the traditional index schemes, and studies the effects of both schemes on decoding speed. Section 4 then considers an algorithm to spread the correlation noise more uniformly across all the quantization intervals. The experimental results are presented in Section 5, whilst Section 6 gives the final comments and conclusion.

2. WYNER-ZIV VIDEO CODING FRAMEWORK

The implemented WZ video coding architecture is based on the DISCOVER Codec [13] and is illustrated in Fig. 1. It considers the odd frames as key frames and encodes them using H.264/AVC Intra coding. Meanwhile, the WZ frames undergo a block-based Discrete Cosine Transform (DCT) and the resulting coefficients are organized into bands having coefficients of the same frequency. Each band is then uniformly quantized into 2^M levels, with a dead-zone quantizer whose intervals adapt with the dynamic range of the coefficient band [13]. The quantized symbol stream q is mapped into a new index representation $\hat{q}$ and the bit-planes of the resulting symbols are fed into an LDPCA encoder [17]. For an architecture with N separate LDPCA decoders, the encoder considers the syndrome bits for the next N bit-planes and stores them into separate buffers. The *Min-rate estimation* module calculates the initial number of bits R_{min}, to be transmitted for each of the N bit-planes, using the Laplacian parameters calculated at the decoder and sent periodically to the encoder via a feedback channel [8].

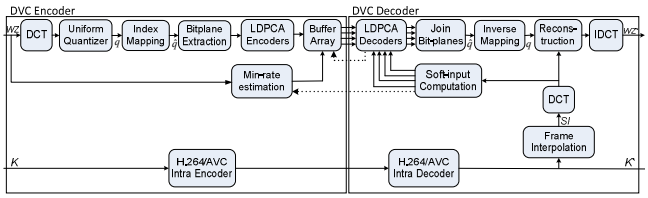


Fig. 1. Proposed transform domain WZ video coding architecture.

The decoder decodes the adjacent key frames and uses them to predict the SI by employing Motion Compensation Temporal Interpolation (MCTI) [16]. The difference between the forward and backward motion compensated frames is then used to model the correlation noise between the original WZ frame and predicted SI [17]. The Laplacian parameters, calculated at band level, are sent to the encoder and used to calculate R_{min}. Conversely, the parameters at coefficient level are used to generate the soft-input values for the next N bit-planes. These values are then fed into N separate LDPCA decoders [15] running on different cores of a multi-core CPU, in this case N is four for a quad-core CPU. Each decoder can request parity bits from the corresponding buffer to correct the initial predictions given by SI. The early stopping criteria adopted in [4] was also included to prevent unnecessary iterations and avoid decoding delays. When the set of N bit-planes are decoded successfully, the next N bit-planes are considered, until all the bit-planes of the WZ frame are recovered successfully. These bit-planes are then joined together to form the indices $\hat{q}$ and mapped back into the quantized coefficient stream q using an inverse mapping scheme. The resulting coefficients are subsequently used, together with the SI, to obtain the best reconstruction of the original WZ coefficients [18]. Finally, an Inverse DCT is applied to recover the WZ frame back in the pixel domain.

3. INDEX REPRESENTATION SCHEME

The soft-input values are calculated by modeling a Laplacian distribution around the value of SI and then sum up the area, under the distribution, enclosed by the quantization intervals whose index has the same bit-value as that of SI at the current bit-plane [12], [19]. The index representation scheme adopted by the DVC architecture will thus affect the accuracy of the soft-input values fed into the LDPCA decoder. Some models, like [20], consider the previously decoded bit-planes as well to improve the accuracy of the soft-input values. Yet, in parallel DVC architectures, where multiple bit-planes are decoded simultaneously, the soft-input values must be computed separately for each bit-plane. The authors in [12] showed that an index representation where the adjacent indices differ by just one bit, like Gray-codes (Fig. 2), can minimize parallelization loss and get results very close to the method presented in [20], even when the previous bit-planes are not exploited in the decoding of the lower order bit-planes. Moreover, the index representation can also affect how the mismatch in SI is distributed amongst the various bit-planes.

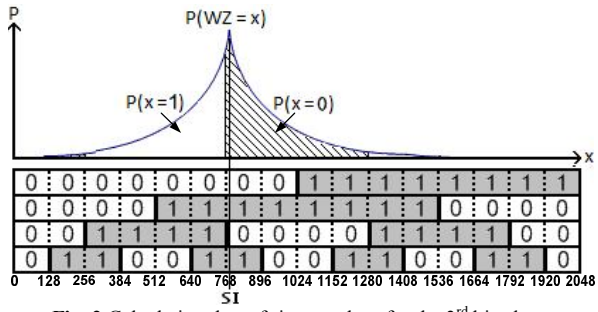


Fig. 2 Calculating the soft-input values for the 3rd bit-plane.

Fig. 2 shows the computation of the soft-input values for the 3rd bit-plane, when a DC coefficient of 766 is predicted with an SI of 771. Using Gray-code representation, the probability $P(x_i=1)$, calculated by summing the area under the shaded regions, is greater than the probability $P(x_i=0)$ and hence the 3rd bit-plane is incorrectly predicted as having a bit-value of 0, despite the very low correlation noise. In general, such mismatch occurs whenever the SI and WZ falls within adjacent intervals, indicated by the grey and white regions. Parity bits are requested from the encoder to

correct such mismatch and to ensure that the reconstructed coefficients are truncated within the quantization interval set by the WZ indices [18]. The probability of mismatch and decoding times, for every bit-plane, are thus correlated to the number of bit-changes within a particular bit-plane level.

Consider the Gray-code representation in Fig. 2, where the number of bit-changes doubles from one bit-plane to the next. The lower significant bit-planes of SI, particularly the Least Significant Bit-plane, have a higher probability of mismatch and require more parity bits (higher rates) to recover, compared to the other bit-planes. The average number of packets required to recover the bit-planes of the DC coefficients, for the first few WZ frames in the *Foreman* sequence, are shown in Fig. 3(a). Here, the coefficients are quantized at 16-levels, to match the index scheme in Fig. 2 and Fig. 4. Clearly, the LSBs take much longer to recover and leave the other decoders, which are concurrently correcting other bit-planes, waiting for them to finish.

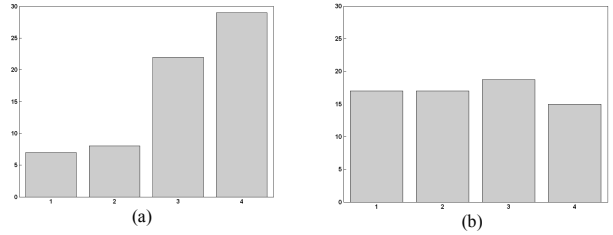


Fig. 3. Rate distribution using (a) Gray-code or (b) proposed index scheme

An alternative indexing strategy can be adopted to ensure that the numbers of bit-changes are spread more uniformly across all bit-planes, whilst having minimal impact on the R-D performance. Fig. 4 illustrates an example of the proposed index representation, set for the different quantization intervals when the coefficients with range $[0, 2^{11}]$ are quantized at 2^L = 16-levels. The proposed index representation scheme can easily be implemented by inserting an *index mapping* module after quantization, to map the quantized DC coefficients into the corresponding index listed in the last row of Fig. 4. The binary representation of the new indices is then considered for encoding and to compute the minimum rate for each bit-plane. With the new index representation, it is clear that the adjacent intervals vary by just one bit, in order to avoid parallelization loss, whilst the different bit-plane levels share the same number of bit-changes.

BINARY	0	0	0	1	1	1	1	0	0	1	1	1	1	0	0	0
	0	0	0	0	0	1	1	1	1	1	0	0	1	1	1	0
	0	0	1	1	1	1	1	1	0	0	0	0	0	0	1	1
	0	1	1	1	0	0	1	1	1	1	1	0	0	0	0	0
Input Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Output Index	0	1	3	11	10	14	15	7	5	13	9	8	12	4	6	2
	0	128	256	384	512	640	768	896	1024	1152	1280	1408	1536	1664	1792	1920	2048

Fig. 4. Proposed index representation for 16-levels

Fig. 3(b) illustrates the average number of packets required to recover the DC coefficients with the proposed index representation, after applying histogram equalization in Section 4. It is clear that the bit-planes, which are worked out simultaneously, are more likely to be recovered at the same rates, taking a much shorter time than that required by the LSBs for Gray code representation thus minimizing the decoding delays.

A similar index scheme is also considered for the possible number of quantization levels 2^L, with $L \in \{1, 2, \ldots 8\}$. The new indices are known a-priori by both the encoder and decoder and kept constant for all input sequences, thus no information needs to be exchanged for synchronization. The decoder uses such index sequence to generate the correct soft-input values and to recover the original DC coefficients after decoding. For the AC bands, the coefficients are first incremented by 2^{M-1}, to obtain indices within the range $[0, 2^M)$, and then mapped into the new representation using the index sequence in Fig. 4.

4. HISTOGRAM EQUALIZATION

Fig. 5 depicts the histogram of coefficients in the DC band, and those in the first AC band, of all the frames in the *Foreman* sequence. Quantizing the bands at 16-levels clearly shows that the histograms are not uniformly distributed across the whole dynamic range. They have peaks over certain intervals, especially for the AC band which is heavily skewed towards the 9th interval (index 0). This non-uniformity implies that some indices, and their neighboring bit-changes, have a higher probability of occurrence, creating an uneven probability of mismatch amongst bit-planes. Consider the histogram in Fig. 5(b) where the 9th interval has the highest probability of mismatch. Using the index representation in Fig. 4, the 1st and the 3rd bit-plane will still have a higher probability of mismatch compared to the rest.

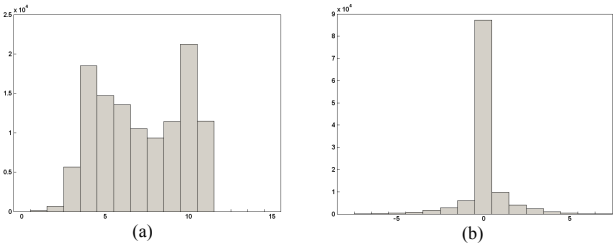
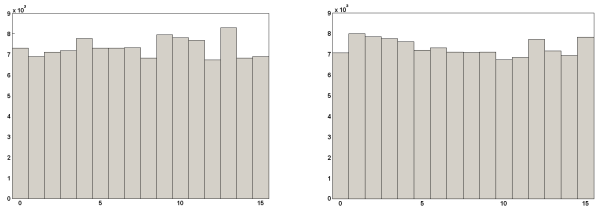


Fig. 5. Histograms for *(a)* DC or *(b)* AC coefficients in *Foreman* sequence.

Since the histogram is sequence dependent, the index representation cannot be optimized to give a uniform distribution of bit-changes over any pre-defined region. It is therefore proposed to ensure that the resulting indices are uniformly distributed across the whole dynamic range instead. This can be achieved by rotating the index sequence in Fig. 4 by a pseudo-random number for each coefficient, and use the new sequence for mapping. Each occurrence of the same coefficient, mainly those occurring very frequently, will thus be assigned a different index representation, equalizing the histogram. The required pseudo-random numbers are obtained by generating a random number within the range [0, 128), for each coefficient location, at the design stage, and known a-priori for both sides of the codec.

The same random sequence is then used for the coefficient of different bands, quantization parameters (QPs) and sequences, to allow synchronization without extra overheads. The histograms in Fig. 6 show the occurrence of the new indices, when using the proposed algorithm for the DC and the first AC band of the *Foreman* sequence. The histograms are better distributed and thus even the number of bit-changes among bit-planes and decoding times. Such uniformity was seen for all bands of the *Foreman* and other sequences.



(a) (b)

Fig. 6. Histograms of indices assigned for *(a)* DC and *(b)* AC coefficients.

5. EXPERIMENTAL RESULTS

The *Akiyo, Hall Monitor, Coastguard* and *Foreman* sequences, all encoded at 15 Hz with a QCIF resolution, were considered to study the effects of the proposed scheme on the R-D performance and decoding speed. All the four sequences were compressed using the parallel architecture in Fig. 1, with a GOP size of 2. The decoding times required when considering both Gray-codes [12] and the proposed index representation are listed in the third and fourth column of Tables I-IV, whilst the fifth column lists the percentage reduction in decoding time between them. For comparison purposes, the 1st column of the tables include the decoding times of the DISCOVER codec [13]. During all experiments, the WZ frames were quantized with the 4×4 quantization matrices in [13], whilst the key frames were encoded using H.264/AVC Intra with QPs chosen so that the key frames and WZ frames have a similar average quality.

The timings in Tables I-IV were evaluated as the total time required to recover all the WZ frames of the sequence, when considering the early stopping criterion in [4] and start decoding with a minimum number of packets given in [8]. All measurements were performed on an Intel® Core™ i5-750 Processor at 2.66 GHz with 12 GB RAM, using four separate cores to simulate the LDPCA modules at the decoder. The code was written using Visual Studio C++ 9.0 and nothing was running on the machine, beside the operating system, when gathering decoding times (as in [21]).

TABLE I: DECODING TIMES FOR THE *AKIYO* SEQUENCE

QP	DISCOVER	Parallel [Gray]	Parallel [Proposed]	Δ time
1	71.98 s	42.67 s	31.10 s	27.12 %
2	111.64 s	78.19 s	53.24 s	31.91 %
3	145.64 s	102.11 s	70.82 s	30.64 %
4	190.56 s	131.83 s	95.16 s	27.82 %
5	179.86 s	120.56 s	92.62 s	23.18 %
6	261.17 s	169.70 s	135.83 s	19.96 %
7	382.25 s	227.63 s	187.81 s	17.49 %
8	734.82 s	436.35 s	375.10 s	14.04 %

TABLE II: DECODING TIMES FOR THE *HALL MONITOR* SEQUENCE

QP	DISCOVER	Parallel [Gray]	Parallel [Proposed]	Δ time
1	128.83 s	81.01 s	61.58 s	23.98 %
2	153.27 s	96.19 s	77.19 s	19.88 %
3	161.64 s	103.67 s	77.07 s	25.54 %
4	402.53 s	221.64 s	177.20 s	20.05 %
5	325.88 s	181.83 s	154.37 s	15.10 %
6	525.62 s	306.68 s	253.04 s	17.49 %
7	687.80 s	387.95 s	312.18 s	19.53 %
8	1022.22 s	561.10 s	468.29 s	16.54 %

TABLE III: DECODING TIMES FOR THE *COASTGUARD* SEQUENCE

QP	DISCOVER	Parallel [Gray]	Parallel [Proposed]	Δ time
1	281.89 s	139.06 s	105.01 s	24.49 %
2	359.75 s	198.02 s	137.14 s	29.68 %
3	393.63 s	219.75 s	159.07 s	28.49 %
4	727.34 s	387.82 s	329.18 s	19.07 %
5	739.03 s	392.29 s	343.52 s	17.37 %
6	1164.08 s	612.01 s	532.91 s	18.87 %
7	1582.64 s	755.57 s	658.01 s	17.85 %
8	2860.61 s	1381.92 s	1194.07 s	18.74 %

TABLE IV: DECODING TIMES FOR THE *FOREMAN* SEQUENCE

QP	DISCOVER	Parallel [Gray]	Parallel [Proposed]	Δ time
1	584.86 s	274.68 s	206.27 s	24.91 %
2	687.17 s	326.11 s	241.84 s	25.84 %
3	803.64 s	394.51 s	322.62 s	18.22 %
4	1501.76 s	723.44 s	580.84 s	19.71 %
5	1709.35 s	824.50 s	674.55 s	18.19 %
6	2385.18 s	1108.63 s	912.22 s	17.72 %
7	2970.25 s	1352.52 s	1094.98 s	19.04 %
8	4021.01 s	1767.23 s	1475.23 s	16.52 %

Since both architectures generate identical SI and the same WZ indices, the quality of the resultant videos is the same. However, the proposed index assignment scheme distributes the rates more uniformly amongst the bit-planes, reducing decoding delays by up to 32% relative to the parallel architecture using Gray-codes [12]. The quality of the soft-input values is slightly degraded, with a marginal loss of 0.02dB, 0.03dB, 0.05dB and 0.06dB in the R-D performance of the respective sequences. These consider the numerical average difference, in terms of Bjøntegaard-Delta metric [22], between the R-D curves of the parallel architectures using either the Gray-code or the proposed index scheme. Due to space limitations, Fig. 7 shows only the R-D performance of the *Foreman* sequence, which suffers the highest degradation. This verifies that the loss is very negligible.

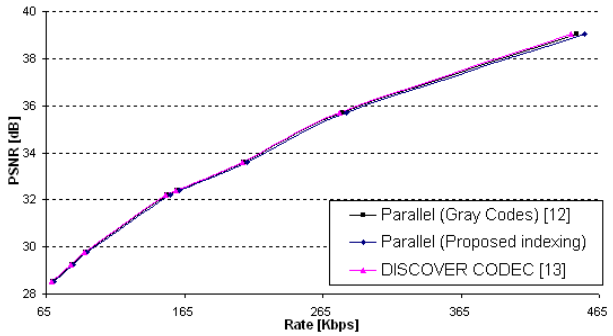


Fig. 7. Rate-Distortion performance for the *Foreman* sequence.

6. CONCLUSION

The non-uniform distribution of rates amongst bit-plane, characterized by the binary or Gray-code representations, and their influence on the performance of the parallel DVC architectures, were considered. A new index representation scheme, which could distribute the rates more uniformly amongst bit-planes, was then proposed to improve decoding speed. Experimental results show that the proposed method can reduce decoding time by up to 32% compared to previous architectures, incurring negligible loss in R-D performance.

ACKNOWLEDGMENTS

The research work disclosed in this publication is partially funded by the Strategic Educational Pathways Scholarship Scheme (Malta). The scholarship is part-financed by the European Union – European Social Fund. (ESF 1.25).

REFERENCES

[1] T. Wiegand, G. Sullivan, G. Bjøntegaard and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560-576, July 2003.

[2] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Transactions on Information Theory*, vol. 22, no. 1, pp. 1-10, January 1976.

[3] J. Ascenso and F. Pereira, "Complexity efficient stopping criterion for LDPC based distributed video coding," in *Proc. International Conference on Mobile Multimedia Communications*, September 2009.

[4] J. Li, X.H. You and J. Li, "Early stopping for LDPC decoding: convergence of mean magnitude (CMM)," *IEEE Communications Letter*, vol. 10, no. 9, September 2006.

[5] Z. Cui, L. Chen and Z. Wang, "An efficient early stopping scheme for LDPC decoding," in *Proc. NASA symposium on VLSI design*, June 2007.

[6] F. 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.

[7] M. Tagliasacchi, J. Pedro, F. Pereira and S. Tubaro, "An efficient request stopping method at the turbo decoder in distributed video coding," in *Proc. EURASIP European Signal Processing Conference*, September 2007.

[8] D. Kubasov, K. Lajnef and C. Guillemot, "A hybrid encoder/decoder rate control for a Wyner–Ziv Video codec with a feedback channel," in *Proc. IEEE International Workshop on Multimedia Signal Processing*, October 2007.

[9] Y. S. Pai, Y. C. Shen and J. L. Wu, "High efficient distributed video coding with parallelized design for LDPCA decoding on CUDA based GPGPU," *Journal of Visual Communications Image Representation*, vol. 23, no. 1, pp. 63-74, January 2012.

[10] S. Wang, S. Cheng and Q. Wu, "A parallel decoding algorithm of LDPC codes using CUDA," in *Proc. Asilomar Conference on Signals, Systems and Computers*, October 2008.

[11] F. Dufaux, W. Gao, S. Tubaro and A. Vetro, "Distributed video coding: Trends and perspectives," *EURASIP Journal on Image and Video Processing*, vol. 2009, Article ID 508167.

[12] Y. Tonomura, T. Nakachi and T. Fujii, "Efficient index assignment by improved bit probability estimation for parallel processing of distributed video coding," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing,* March 2008.

[13] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov and M. Ouaret, "The DISCOVER codec: architecture, techniques and evaluation," in *Proc. Picture Coding Symposium*, November 2007.

[14] C. Brites, J. Ascenso and F. Pereira, "Improving transform domain Wyner-Ziv video coding performance," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2006.

[15] D. Varodayan, A. Aaron and B. Girod, "Rate-adaptive codes for distributed source coding," *EURASIP Signal Processing Journal*, vol. 86, no. 11, November 2006.

[16] J. Ascenso, C. Brites and F. Pereira, "Improving frame interpolation with spatial motion smoothing for pixel domain distributed video coding," in *Proc. EURASIP Conference on Speech and Image Processing*, July 2005.

[17] C. Brites and F. Pereira, "Correlation noise modeling for efficient pixel and transform domain Wyner-Ziv video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 9, September 2008.

[18] D. Kubasov, J. Nayak and C. Guillemot, "Optimal reconstruction in Wyner-Ziv video coding with multiple side information," in *Proc. IEEE International Workshop on Multimedia Signal Processing,* October 2007.

[19] M. Dalai, R. Leonardi and F. Pereira, "Improving turbo codec integration in pixel-domain distributed video coding," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2006.

[20] S. Cheng and Z. Xiong, "Successive refinement for the Wyner-Ziv problem and layered code design," *IEEE Transaction on Signal Processing*, vol. 53, no. 8, pp. 3269-3281, August 2005.

[21] DISCOVER Project [Online]. Available: www.discoverdvc.org.

[22] G. Bjøntegaard, "Calculation of average PSNR differences between RD curves," in *VCEG Meeting*, April 2001.