OVERLAPPED QUASI-ARITHMETIC CODES FOR DISTRIBUTED VIDEO CODING

Xavi Artigas[1], Simon Malinowski[2], Christine Guillemot[3], Luis Torres[1]

[1]Technical University of Catalonia, {xavi, luis}@gps.tsc.upc.edu
[2]IRISA/University of Rennes, smalinow@irisa.fr
[3]IRISA/INRIA, cguillem@irisa.fr

ABSTRACT

This paper describes Slepian-Wolf codes based on overlapped quasi-arithmetic codes, where overlapping allows lossy compression of the source below its entropy. In the context of separate decoding, these codes are not uniquely decodable: the overlap introduces ambiguity in the decoding process leading to decoding errors. The presence of correlated side information at the decoder is used to remove this ambiguity and achieve a vanishing error probability. The state models and the automata of the overlapped quasi-arithmetic codes are described. The soft decoding algorithm with side information is then presented. The performance of these codes has been assessed first on theoretical sources and integrated in a distributed video coding platform.

Index Terms— Distributed video coding, Wyner-Ziv coding, coding with side information, arithmetic codes

1. INTRODUCTION

Distributed source coding (DSC) refers to the problem of compressing correlated signals captured by different sensors which do not communicate between themselves. DSC finds its foundation in the seminal Slepian-Wolf [1] and Wyner-Ziv [2] theorems establishing lossless rate bounds and rate-distortion bounds respectively. Most Slepian-Wolf and Wyner-Ziv coding systems are based on channel coding principles, using e.g., coset codes [3] or turbo codes [4]. The statistical dependence between two sources is modeled as a virtual correlation channel analogous to binary symmetric channels or additive white Gaussian noise (AWGN) channels. The use of entropy source codes tailored to the conditional distribution between the two sources has also been considered for DSC in [5].

This paper considers the design of Slepian-Wolf codes based on arithmetic and quasi-arithmetic codes. The basic idea is to allow the intervals corresponding to each source symbol to overlap. This procedure can achieve arbitrary compression controlled by the amount of allowed overlap, but, as already stated by Langdon in [6], it leads to a code which is not uniquely decodable. This work is a preliminary exploration of the possibility of using the side information available only at the decoder to disambiguate the overlaps.

It is shown that overlapped arithmetic (OA) and quasi-arithmetic (OQA) codes can be represented with an encoding and a decoding state machine (SM). The method to obtain these state machines is adapted from the ones proposed in [7] and [8]. The representation of OA and OQA codes by SM allows defining a symbol clock based state model for soft decoding of these codes. Finally, the side information available at the decoder is integrated into this state model to help the decoder remove the ambiguity introduced by the overlapping intervals principle.

Multiple reasons motivate the research of source codes in addition to channel codes. For example, the proposed DSC scheme can be easily extended to non-binary sources by just considering non-binary source codes. Moreover, for non uniform sources or sources with memory, the additional information about the distribution of the source is directly exploited by the decoder. Finally, the considered scheme does not require the source to be uniform to perform optimally, as turbo codes do, for example.

Section 2 describes overlapped codes, their construction and the problem posed by their ambiguous decoding. Section 3 then presents the soft decoding scheme used to decode these codes. Section 4 gives simulation results for theoretical sources. Section 5 outlines how these codes can be embedded in a DVC architecture and finally Section 6 extracts some conclusions and sketches the future research lines derived from this preliminary work.

2. THE OVERLAPPING TECHNIQUE

Let $X = X_1, \ldots, X_K$ be a symbol string generated by a binary memoryless source. The alphabet of the source is $\mathcal{A} = \{a, b\}$ and the probability of symbol a is denoted P_a. Only binary memoryless sources are considered in this work but the results can be generalized to Markov sources and to non-binary alphabets.

In classical binary arithmetic coding, the interval $[0, 1)$ is recursively partitioned according to the source probabilities. At each symbol clock instant n, the current interval $\mathcal{I}_n = [L_n, H_n)$ is partitioned into two subintervals $\mathcal{I}_{n+1}^a$ and $\mathcal{I}_{n+1}^b$ whose widths are proportional to P_a and $P_b = 1 - P_a$ respectively. We hence have $\mathcal{I}_{n+1}^a = [L_n, P_a(H_n - L_n))$ and $\mathcal{I}_{n+1}^b = [P_a(H_n - L_n), H_n)$. One of these subintervals is selected according to the value of X_{n+1} and becomes the current interval. Once the last symbol is encoded, the encoder outputs enough bits to distinguish the final interval from all other possible intervals of $[0, 1)$.

To reduce the coding delay, bits can be output during the encoding process as soon as the current interval is entirely in $[0, 0.5)$ (a bit 0 is output) or entirely in $[0.5, 1)$ (a bit 1 is output). The current interval is rescaled each time a bit is output. The arithmetic principle has been represented in [9] as a stochastic model used for soft decoding of these codes. The number of states is finite if the source probabilities are known but it exponentially increases with the sequence length.

It has been shown in [10] that the number of states of an arithmetic code can be reduced for a slight loss in compression performance. The initial interval is set to $[0, N)$ where N is an integer. The partitioning of the current interval is done as in arithmetic coding but the current intervals are always integer intervals. This principle is called quasi-arithmetic (QA) coding. The main interest of QA coding is that the number of states of the encoder is finite and does not increase with the sequence length.

In the case of overlapped encoding, the current interval is first partitioned according to the source probabilities, but the subintervals are then modified so that they overlap. The parameter ρ controls the width of overlapping between the subintervals. It can be seen in Figure 1 that the overlapping allows wider subintervals, and this leads to higher compression. The subintervals of $\mathcal{I}_n = [L_n, H_n)$ are defined as follows:

$$\begin{cases} \mathcal{I}_{n+1}^a & = [L_n, P_a(H_n - L_n) + \rho N/2) \\ \mathcal{I}_{n+1}^b & = [P_a(H_n - L_n) - \rho N/2, H_n) \end{cases} \quad (1)$$

Note that this partitioning of the current interval can be applied to both arithmetic (by taking $N = 1$) and quasi-arithmetic coding (by taking $N > 1$ and by rounding the bounds of the subintervals to the nearest integers).

Since it has been shown in [10] that the loss of compression efficiency introduced by integer arithmetic coding is very low, overlapped quasi-arithmetic coding will be considered in the following.

A finite state encoder is hence obtained by applying this alternative method of partitioning the current interval. The encoder state machine depends on N, P_a and the overlapping factor ρ. The number of states of the encoder can be reduced by removing the mute transitions (i.e., the transitions that do not output any bit) as explained in [8].

An example of OQA encoder is given in Figure 2. This encoder is not uniquely decodable. For instance, when at state [0, 6), symbol strings $bbaaa$ and $babb$ are encoded with the same bit string 1001.

The overlap principle results in enlarging the current interval, and hence in reducing the number of output bits. Indeed, bits are output as soon as the current interval entirely
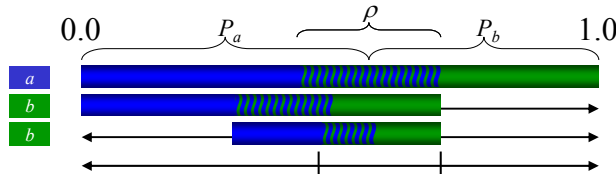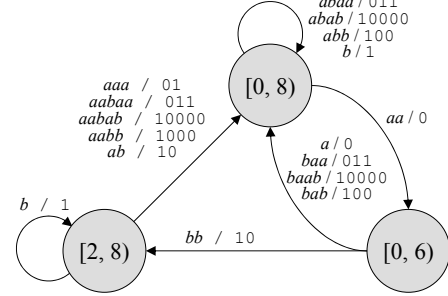
Figure 2. SM for an example binary OQA encoder. N=8, P_a=0.6, P_b=0.4 and ρ=0.1. $\{a, b\}$ are input symbols, $\{0, 1\}$ are output bits.

lies in the first or second half of $[0, N)$. Examples of the compression performance of an $N = 8$ OQA encoder are given in TABLE 1 for different values of ρ and P_a. Observe that for $\rho = 0$ (no overlapping) the compression rate is very close to the entropy, as one would expect from conventional arithmetic codes. The non-monotonic nature of the rows is due to the small N used. As N grows and the performance of the quasi arithmetic codes converges to that of pure arithmetic codes, better compression rates are achieved and the rows become monotonically decreasing.

## 3. SOFT DECODING OF OVERLAPPED CODES

It has been shown in the previous section how to define overlapped arithmetic and quasi-arithmetic encoders. Both kind of encoder can be defined with a state machine. The state machine generates a variable number of bits depending on the current state and the next source symbol(s) that have to be encoded. A decoding SM can be computed by just inverting the inputs and outputs of the encoding SM. Note that this holds only if the encoding state machine does not contain mute transitions.

Let $\mathcal{T} = \{\alpha_0, \ldots, \alpha_d\}$ be the set of states of the decoding SM. As the mute transitions of this SM have been removed, a variable number of transitions leave each state of the SM as it can be seen in Figure 2. For every transition t in the decoding SM, let b_t be the bit string that labels this transition and s_t the symbol string output by this transition. The corresponding length of b_t and s_t are denoted $\overline{b_t}$ and $\overline{s_t}$ respectively.

The overlapping mechanism results in ambiguity being introduced in the encoded bit string, and this means that, even if the received bit string is error-free, more than one symbol sequence will be retrieved by the decoding state machine. In the following, it will be assumed that a side information $Y = Y_1 \ldots, Y_K$ correlated to X is available at the decoder

Figure 1. Example of overlapped arithmetic encoding. P_s is the probability of symbol s and ρ is the overlapping factor. The sequence abb has been encoded into the shown final interval.

TABLE 1. Compression rates obtained for a $N = 8$ overlapped quasi-arithmetic code at different values of P_a and ρ after simulation of 1000 strings of 1000 symbols each.

ρ \ P_a	0.5	0.6	0.7
0.0	1.000	0.992	0.888
0.1	1.000	0.849	0.702
0.2	0.682	0.706	0.628
0.3	0.513	0.558	0.519
H(X)	1.000	0.971	0.881
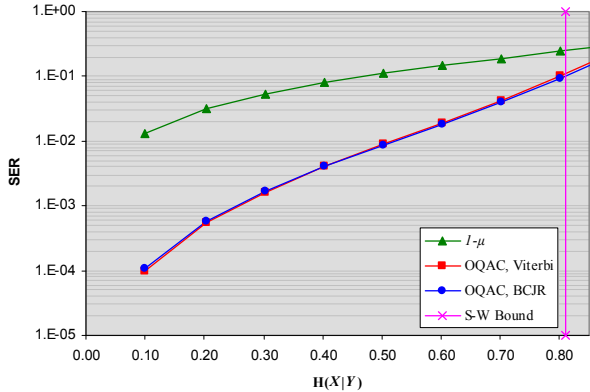
Figure 3. Decoding performance for a compression rate of 0.81 bits/symbol (Source entropy is 0.5)
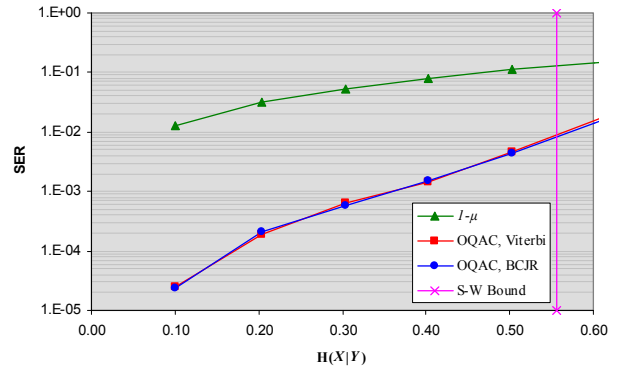
Figure 4. Decoding performance for a compression rate of 0.56 bits/symbol (Source entropy is 0.722)

side. The amount of correlation between X and Y is denoted μ. In other words, for $1 \leq i \leq K$, $\mathbb{P}(X_i = Y_i) = \mu$. In order to optimally exploit this additional information, a symbol clock based state model has to be defined. The state model for soft decoding of overlapped codes is defined by the pairs (N_k, M_k) where N_k is the state of the decoding SM at the bit clock instant k (i.e., $N_k \in \mathcal{T}$) and M_k represents the possible values of the symbol clock at the bit clock instant k. Such a model for soft decoding of arithmetic and quasi-arithmetic codes was proposed in [9] and [7]. The transition probabilities on this model are defined as follows:

$$
\mathbb{P}(N_k = \alpha_i, M_k = m' | N_l = \alpha_j, M_l = m) =
\begin{cases}
\mathbb{P}(N_k = \alpha_i | N_l = \alpha_j) & \text{if } k - l = \overline{b_t} \text{ and } m' - m = \overline{s_t} \\
0 & \text{otherwise}
\end{cases} \quad (2)
$$

where the probabilities $\mathbb{P}(N_k = \alpha_i | N_l = \alpha_j)$ are deduced from the source statistics and the decoding SM. A Viterbi [11] or BCJR [12] algorithm can be applied on this state model in order to estimate X. The symbol clock based model defined above allows us to integrate the information brought by Y in the decoding trellis. To take into account this additional information, the branch metric of a transition t, denoted $\gamma_t(N_k, M_k | N_l, M_l)$, used for the BCJR or Viterbi algorithm are defined as follows:

$$
\gamma(N_k, M_k | N_l, M_l) = \mathbb{P}(Y_{m'} \ldots Y_m = s_t) \\
\times \mathbb{P}(N_k = \alpha_i, M_k = m' | N_l = \alpha_j, M_l = m) \quad (3)
$$

where $\mathbb{P}(Y_{m'} \ldots Y_m = s_t)$ is deduced from the correlation between X and Y. The Viterbi algorithm selects the most probable sequence given the side information, whereas the BCJR algorithm computes the a-posteriori marginal probabilities of each bit or each symbol.

4. RESULTS WITH THEORETICAL SOURCES

1000 symbol strings of 1000 binary symbols each have been simulated and the results averaged. The obtained Symbol Error Rate (SER) is shown in Figure 3 (for N=16, P_a=0.5 and ρ=0.1) and Figure 4 (for N=16, P_a=0.8 and ρ=0.1), along with the Slepian-Wolf bound (equal to the compression rate). $H(X|Y)$ is the binary entropy of μ. As an upper bound, the SER for a trivial decoding mechanism is also shown: the side information is directly regarded as the decoded output, so the probability of symbol error is $1 - \mu$. Please note that Figure 3 and Figure 4 use N=16 whereas TABLE 1 used N=8.

It can readily be seen that in both set-ups the OQA codes work better than the trivial solution, and the SER decreases when the Slepian-Wolf bound is crossed, although not very sharply.

5. APPLICATION TO DVC

In a complete DVC architecture OQA codes can be used in the place of the Slepian-Wolf codec. The final performance will probably depend on many design decisions like: *i*) how the source statistics are found, *ii*) how the target compression is selected (this is the rate control problem every DSC codec faces) and *iii*) the relationship between the target compression and the overlapping factor has to be studied (it depends on the overlapping mechanism).

Since not all of these issues have been solved yet, experiments have been conducted on data extracted from video sequences and only the performance of the Slepian-Wolf code has been assessed instead of the whole codec.
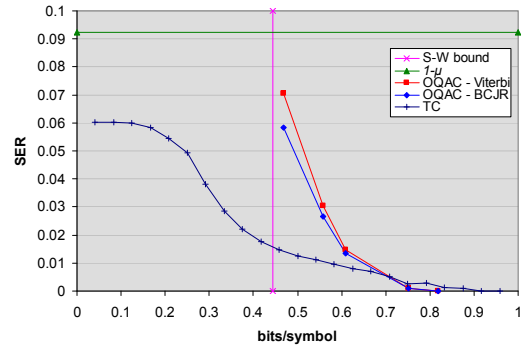
400 frames of the Foreman test sequence at QCIF resolution have been introduced into the DVC system[1] described in [13] to obtain strings of binary symbols. These strings come from different DCT bands and different bit planes, so they all have different statistics and have memory. They

---
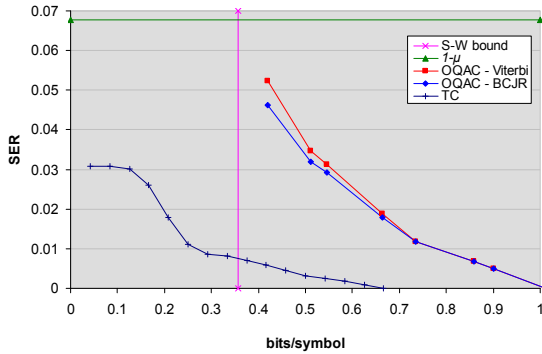
Figure 5. Results for $N = 16$, Band 0, bit plane 3.
$P_a = 0.54$, H($X|Y$) = 0.36

Figure 6. Results for $N = 16$, Band 3, bit plane 0.
$P_a = 0.75$, H($X|Y$) = 0.44

have then been separately coded using the turbo codec already present in the system and OQA codes. The side information has been generated by interpolating neighboring key frames inserted in between every Wyner-Ziv frame (GOP size = 2). A 4x4 DCT transform is used, so each bitplane in a frame produces a string of 1584 binary symbols. 200 such symbol strings from all available Wyner-Ziv frames have been simulated and the results averaged.

With this set-up, the source probabilities (P_a) and the conditional entropy (the correlation μ with the side information) are fixed by the sequence, the interpolation algorithm and the examined DCT band and bit plane. Varying the ρ parameter (the overlapping factor) produces different compression ratios and different SER, depicted in Figure 5 and Figure 6. The Slepian-Wolf bound is also plot, and, again, the trivial decoder used in the previous subsection is used as an upper bound ($1 - \mu$). The correlation of the side information with the original string is independent of the compression ratio, so this bound is now a straight line.

It can be seen that the SER increases as the compression increases and approaches the Slepian-Wolf bound, and that the OQA codes perform better than the trivial solution. Also, BCJR decoding works better than Viterbi as expected. It is also apparent that the performance of the OQA codes is still far from that of the turbo codes, which achieve comparable SER with higher compression. The next section points out some possible lines of action to bridge the gap with turbo codes.

6. CONCLUSIONS AND FUTURE WORK

This work is a preliminary exploration of the possibility of using Overlapped Quasi-Arithmetic Codes as a proposal for the Slepian-Wolf coding approach. Simulation results indicate that OQA codes perform better than just trusting the side information, but are still far from the performance of state-of-the-art turbo codes. However, the authors believe that the approach is very promising and worth exploring.

These results can still be improved by designing overlapping methods other than the one described in Section 2 and by taking source memory into account, for example.

Regarding encoding complexity, it is comparable to that of turbo codes, since in practice turbo codes are also implemented as state machines (plus an interleaver).

7. REFERENCES

[1] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources", *IEEE Trans. Inform. Theory*, vol. 19 pp. 471-480, July 1973

[2] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder", *IEEE Trans. Inform. Theory*, vol. 22, pp. 1-10, January 1976

[3] R. Puri and K. Ramchandran. "PRISM: A new robust video coding architecture based on distributed compression principles". *Proc. of 40th Allerton Conf. on Comm., Control, and Computing*, Allerton, IL, Oct. 2002

[4] B. Girod, A. Aaron, S. Rane and D. Rebollo-Monedero, "Distributed video coding", *Proc. of the IEEE*, vol. 93, no. 1, January 2005

[5] Qian Zhao, Effros, M., "Optimal code design for lossless and near lossless source coding in multiple access networks", *Proc. of Data Compression Conference* (DCC 2001). Page(s):263 – 272, 27-29 March 2001, Snowbird, Utah, USA

[6] G. G. Langdon, Jr., "An introduction to arithmetic coding", *IBM Journal of Research and Development*, vol. 28, no. 2, pp. 135-149, March 1984

[7] T. Guionnet and C. Guillemot, "Soft decoding and synchronisation of arithmetic codes: application to image transmission over noisy channels", *IEEE Trans. on Image Processing*, vol. 12, pp. 1599–1609, 2003.

[8] S. Ben-Jamaa, C. Weidmann, M. Kieffer, "Asymptotic Error-Correcting Performance of Joint Source-Channel Schemes based on Arithmetic Coding", *IEEE International Workshop on Multimedia and Signal Processing* (MMSP 2006), Victoria, Canada, October 3-6, 2006

[9] T. Guionnet and C. Guillemot, "Soft and joint source-channel decoding of quasi-arithmetic codes", *EURASIP Journal on applied signal processing*, vol. 3, pp. 394-411, Mar. 2004

[10] P. Howard and J. Vitter, "Practical implementations of arithmetic coding", *Image and Text Compression*, J.A. Storer, ed. Kluwer Academic Publishers, Norwell, MA, pp. 85-112, 1992

[11] A. Viterbi, "Error bounds for convolution codes and an asymptotically optimum decoding algorithm", *IEEE Trans. on Inform. Theory*, no. 13, pp. 260–269, 1967.

[12] L.Bahl, J.Jelinek, J.Raviv, and F.Raviv, "Optimal Decoding of Linear Codes for minimising symbol error rate", *IEEE Trans. on Inform. Theory*, vol. IT-20, pp. 284-287, March 1974.

[13] C. Brites, J. Ascenso and F. Pereira, "Improving Transform Domain Wyner-Ziv Video Coding Performance", *IEEE ICASSP*, Toulouse, France, May 14-19, 2006