

# ERROR CORRECTING ARITHMETIC CODING FOR ROBUST VIDEO COMPRESSION

Marco Grangetto, Enrico Magli, Gabriella Olmo

CERCOM - Center for Multimedia Radio Communications  
Dipartimento di Elettronica - Politecnico di Torino  
Corso Duca degli Abruzzi 24 - 10129 Torino - Italy  
E-mail: grangetto (magli, olmo)@polito.it

## ABSTRACT

In this paper we present a novel error correcting arithmetic co/decoder, based on forbidden symbol detection and maximum a posteriori estimation. The error correction goal is obtained by means of a sub-optimal metric-first search technique that exhibits attractive performance in terms of error recovery and decoding complexity. The proposed technique is able to compress the input data and at the same time to provide error correction. In the present work error correcting arithmetic codes are applied to the case of compression and transmission of H.264 motion vectors and the performance is compared with a concatenated scheme based on standard arithmetic coding and forward error correction codes.

## 1. INTRODUCTION

Reliable multimedia communications over wireless links represent one of the most important targets for both the industrial and the academic community, involved in the development of future personal mobile communication systems. For this reason emerging standards for data compression, such as JPEG 2000 [1] for still images and H.264 [2] for video sequences, aim at improving not only the performance in terms of quality at a certain bit rate but also the error resilience and network adaptation capabilities of the compressed stream.

In this paper a novel error correcting entropy coding technique, based on arithmetic coding (AC), is presented. Both the compression standards, mentioned above, comprise an AC stage, as the last coding step; in fact AC is progressively substituting traditional *Huffman* codes because of its superior performance. On the contrary AC is very sensitive to transmission error and even a single flipped bit in the received com-

pressed stream can cause irreversible error propagation throughout the decoded sequence. As a consequence a great interest in the development of error resilience tools for AC has arisen.

In [3] Boyd *et al.* proposed an effective technique for embedding error detection capability into AC, based on the introduction of a forbidden symbol in the coding alphabet. The forbidden symbol allows one to adjust the amount of coding redundancy to be embedded in the coded stream, at the expenses of compression efficiency; on the other hand it permits error detection at the decoder side. The same error detection technique is applied to the case of image transmission in [4] and further studied in [5], where the concept of continuous error detection is introduced. In [6, 7] the coding redundancy is employed for error correction in the case of image transmission.

In this paper we improve the error correction capability of the novel *maximum a posteriori* (MAP) estimator for AC, proposed by Grangetto and Cosman in [8], and present results in the case of H.264 motion vectors (MV) compression and transmission across AWGN channel, which occurs e.g. in the case of data partitioning and prioritized transmission. The obtained results suggest that the proposed technique could be a viable solution in order to provide the entropy coder with error correction capability, competitive with standard forward error correction approaches.

## 2. MAP ESTIMATION OF ARITHMETIC CODES

### 2.1. AC with a forbidden symbol

As already mentioned, AC represents the last coding stage of the two emerging standard JPEG 2000 and H.264. In the first case the MQ coder is used, in the latter one CABAC codec [9] is adopted; these are bi-

nary, adaptive and context-based arithmetic coders. In the following we are going to develop an error resilient entropy coder, based on a simple binary and non adaptive AC. Nevertheless, it is worth pointing out that the proposed approach is general and its introduction in more sophisticated codec will be part of our future developments.

Binary and non adaptive AC is based on a memoryless source model, defined by the probability of “0”,  $P_0$ , and the probability of “1”,  $P_1$ . We consider the encoding of a fixed length binary string  $\mathbf{a}_i$  of  $L$  bits, that will be consequently mapped onto a variable length sequence  $\mathbf{b}_i$  of  $N$  bits. AC allows one to use a number of bits close to the source entropy  $N \approx LH$ , where  $H = -\sum_{i=0}^1 P_i \log(P_i)$ . Both encoding and decoding are iterative tasks. The decoding iterations are very sensitive to bit errors and the resynchronization is far less likely than in Huffman decoding case [4]. A simple mean to obtain error resilience, proposed in [3], is provided by the introduction of a forbidden symbol  $\mu$ , which is never encoded and whose probability is fixed to an arbitrary value  $P_\mu = \epsilon$ . The encoding task is thus performed on the base of the ternary alphabet “0”, “1” and  $\mu$  with probability  $P_0(1 - \epsilon)$ ,  $P_1(1 - \epsilon)$  and  $\epsilon$  respectively. The introduction of  $\mu$  clearly forces a modification of the source model, that becomes less precise as  $\epsilon$  increases. This corresponds to an amount of coding redundancy per encoded bit  $R_\mu = -\log(1 - \epsilon)$  [4], that is forced in the encoded binary sequences at expenses of compression efficiency. At the decoder side the presence of  $\mu$  allows for error detection; if it is decoded, this means that transmission errors have occurred. In [4], it is shown that the probability that the number of erroneously decoded bits before  $\mu$  is detected is greater than  $n$  is  $(1 - \epsilon)^n$ . Therefore, a large value of  $\epsilon$  assures fast error detection, but it greatly reduces the compression efficiency. On the contrary, a small value of  $\epsilon$  does not impact compression efficiency but there will be a large error detection delay.

## 2.2. Metric-first MAP estimation

The coding redundancy associated with the forbidden symbol can be exploited by the decoder for error correction. This task can be recognized as a *Joint Source and Channel Coding* (JSCC) approach, where the redundancy at source level can be used as a form of channel protection against transmission errors. Let us consider that the coded binary string  $\mathbf{b}_i$  is transmitted and the receiver observes the bit sequence  $\mathbf{r}$  of length  $N$ , through the channel with transition proba-

bility  $P(\mathbf{r}/\mathbf{b}_i)$ . The objective of the MAP decoder is to pick up the most probable input sting  $\hat{\mathbf{a}}$ ,

$$\hat{\mathbf{a}}|P(\hat{\mathbf{a}} = \mathbf{a}_i/\mathbf{r}) \geq P(\mathbf{a}_j/\mathbf{r}) \quad \forall j \neq i \quad (1)$$

Therefore the estimation is based on the following decoding metric,

$$P(\mathbf{a}_j/\mathbf{r}) = \frac{P(\mathbf{r}/\mathbf{a}_j)P(\mathbf{a}_j)}{P(\mathbf{r})} = \frac{P(\mathbf{r}/\mathbf{b}_j)P(\mathbf{a}_j)}{P(\mathbf{r})} \quad (2)$$

where  $P(\mathbf{a}_j)$  represents the *a priori* source term. In the MAP metric, along with the channel transition probability, it is worth noticing the term  $P(\mathbf{r})$ , which represents the probability of observing a certain sequence at the receiver. Given the received  $N$  bits binary string  $\mathbf{r}$ , we have  $P(\mathbf{r}) = \sum_{\mathbf{j} \in \mathbf{B}_N} \mathbf{P}(\mathbf{r}/\mathbf{b}_j)\mathbf{P}(\mathbf{a}_j)$ , where  $\mathbf{B}_N$  identifies the subset of the coded sequences whose length is equal to  $N$ ; since the evaluation of this term is as complex as the decoding metric (2), it can be approximated by  $2^{-N}$ , assuming that all the received sequences of equal length are equally likely. This approximation does not hold for any variable length code, but still provides satisfactory results, as demonstrated in [8].

The MAP estimator is implemented by means of a suboptimal sequential decoder known as *Stack Algorithm* (SA). In principle, the metric (2) must be evaluated for all the possible transmitted string  $\mathbf{b}_j \in \mathbf{B}_N$ , whose length is compatible with the received sequence. This is practically unfeasible for reasonable values of the input sequence length  $L$ , and therefore suboptimal SA is employed. The decoding metric, in logarithmic form, is decomposed into additive terms per received bit, and the problem is recast as a search along a binary tree, representing all the binary sequences of length  $N$ . The SA is a metric-first technique, which greedily extends the tree branch with the best accumulated metric. During the tree exploration, those paths which do not correspond to valid encoded strings are pruned with a certain probability upon detection of  $\mu$  by the arithmetic decoder. The algorithm sub-optimality stems from the limited memory  $M$ , which represents the maximum number of paths stored for future visit. SA exhibits good performance and reasonable complexity as reported in [8], where it was compared with another sequential approach.

SA allows sequential decoding, since after a certain delay the explored path roots tend to merge in a single one, permitting partial decoding of the initial bits. Finally the correcting capability is deeply affected by the termination rule, adopted in order to

stop the search and claim that the correct decisions have been taken. The termination strategy used in this paper will be discussed in the following section.

### 3. PROPOSED CODING SYSTEM

The block diagram of the proposed JSCC system is reported in Fig. 1. The H.264 MV horizontal and vertical components, represented on 8 bits, undergo a binarization stage which aims at creating an unbalanced binary stream ( $P_0 \gg 0.5$ ) for subsequent non adaptive binary AC. 8 bits binary symbols with the least possible number of “1” are used to represent components according to their magnitude, in such a way that the most probable values are mapped on binary words with few “1”. The employed binary mapping is exemplified in Tab. 1, in the case of 4 bits MV components. This kind of binarization is arbitrary; it is evident that a better one could be designed, but this paper peculiarity resides in AC error correction capability, despite of possible improvements from the point of view of compression efficiency. During the binarization the value of  $P_0$ , that is statically employed by the following AC, is evaluated as well.

The obtained 8 bits symbols are organized into fixed length packets, independently encoded by AC with forbidden symbol, based on the memoryless source model  $[P_0(1-\epsilon), P_1(1-\epsilon), \epsilon]$ . In our implementation 512 MV components are placed into each packet, so as to form input sequences  $\mathbf{a}_i$  with length  $L = 4096$  bits. The packet is further terminated with an *End of Packet* (EOP) symbol with probability  $P_{EOP} = \omega$ ; this task is performed by switching to the termination source model  $[P_0(1-\omega), \omega, P_1(1-\omega)]$ . The added redundancy due to EOP symbol is  $R_{EOP} = -\log(\omega)/L$ . Each packet is thus encoded onto a variable length sequence  $\mathbf{b}_i$ . The encoded packet length is transmitted as side information to the receiver. Assuming to protect with a coding rate  $R_C = 1/3$  this vital side information, we have a rate overhead  $R_S = 39/L = 0.0095$ . The value of  $P_0$ ,  $\epsilon$  and  $\omega$  must be sent to the decoder as well, but since their are kept fixed for all the encoded packets the subsequent rate overhead is negligible. The overall compression ratio can be expressed as  $R = H + R_\mu + R_{EOP} + R_S$ .

The encoded packets are then transmitted across the AWGN channel with signal to noise ratio  $E_b/N_0$ , assuming 2-PSK signalling and hard demodulation. We can thus consider the equivalent binary symmetric channel with transition probability  $p = \frac{1}{2} \operatorname{erfc} \left( \sqrt{\frac{E_b}{N_0}} \right)$ . The received packet  $\mathbf{r}$  is finally processed by the MAP

**Table 1.** MVs binarization (4 bits example).

MV	$\mathbf{a}_i$	MV	$\mathbf{a}_i$
0	0000	-4	0110
1	0001	5	1010
-1	0010	-5	1100
2	0100	6	0111
-2	1000	-6	1011
3	0011	7	1101
-3	0101	-7	1110
4	1001	8	1111

estimator, based on metric (2) and SA, that attempts error recovery. The additive metric (2) is sequentially evaluated;  $P(\mathbf{r}/\mathbf{b}_j)$  is computed by comparing the received sequence versus the candidate  $\mathbf{b}_j$ , the a priori source term is accumulated through partial arithmetic decoding, while the denominator is approximated as discussed above.

The decoder termination rules are as follows: SA terminates the search when the best visited path in the tree corresponds to an input sequence  $\hat{\mathbf{a}}$ , which consumes all the received packet bits and ends with the EOP symbol. The role of the termination is crucial in order to allow correct decoding of the final part of the packet; the effect is similar to the termination effect on standard convolutional codes. During the search, paths can be dropped if the candidate can not be decoded with the correct number of bits  $L$  or if the EOP is not revealed; moreover paths are rejected in the case of  $\mu$  symbol detection. The proposed MAP estimator can fail decoding in the case all the  $M$  paths in storage get dropped on the basis of the previous conditions. The proposed dropping rules aim at limiting the search to the subset  $\mathbf{B}_N$ ; a large value of  $\epsilon$  guarantees quick error detection and thus circumscribes the algorithm exploration to a close vicinity of  $\mathbf{B}_N$ , improving performance in term of both error correction and decoding time.

The SA greedy search tends to move as quickly as possible towards the best candidate, according to the defined metric; the random walk followed by the algorithm, and hence the decoding time, are affected by all the terms involved in metric (2), namely the value of  $P_0$  and the channel transition probability  $p$ ; this latter point will be further clarified in the following section by means of experimental results.

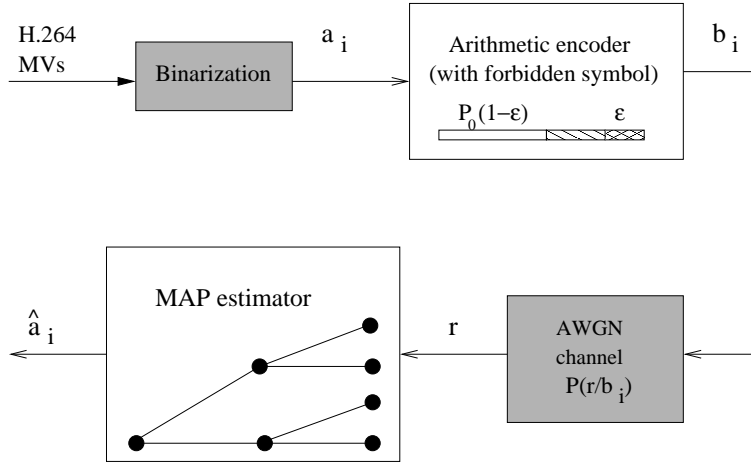


Figure 1. System block diagram.

#### 4. SIMULATION RESULTS

Simulations have been run on the MVs obtained by means of the H.264 encoder (JM 2 codec, ver. 2.1) on the *Foreman* sequence with a temporal resolution of 15 and 30 fps respectively.

The proposed MAP decoding scheme has been tested in the following conditions: packets of length  $L = 4096$  bits are considered and the SA algorithm with a memory of  $M = 2048$  is employed. The value of  $M$  is selected as a compromise between performance and computational complexity. The EOP symbol probability is  $\omega = 10^{-4}$ , corresponding to a rate overhead  $R_{EOP} = 0.003$ . The characteristic of the encoded data are summarized in Tab. 2, where the number of obtained MVs, the value of  $P_0$  after the binarization stage and the overall compression ratio  $R$ , in the case  $\epsilon = 0$ , are reported. It is evident that the sequence at 15 fps is characterized by larger magnitude MV components and, as a consequence, it is slightly less compressible.

The JSCC scheme presented in this paper has been compared with a traditional separated approach, based on *Rate Compatible Punctured Convolutional* (RCPC) codes. In this case each packet undergoes binary AC without forbidden symbol, still terminated by EOP in order to recognize decoding failure. The packet is then protected against transmission errors by means of RCPC codes, reported in [10], with memory 6 and non punctured rates 1/3, so as to obtain the same rate as the the proposed AC.

In Fig. 2 the packet error rate (PER) as a function of the signal to noise ratio  $E_b/N_0$  over the channel, obtained with our MAP estimator (star markers) and traditional RCPC based scheme (triangle) on Fore-

man at 30 fps, are reported. The values of  $E_b/N_0$  are in the range from 4.3 dB to 6.8 dB, corresponding to a transition probability on the channel from  $p = 10^{-2}$  to  $p = 10^{-3}$ . In Fig. 2 we set  $\epsilon = 0.035$  and a RCPC coding rate  $R_C = 8/9$ , both yielding a compression ratio  $R = 0.43$ . It is worth noticing that, in the simulated conditions, the proposed JSCC decoder achieves a considerable coding gain of more than 0.5 dB over the separated approach. In Fig. 3 similar results are obtained in the case  $R = 0.47$  when  $\epsilon = 0.065$  and  $R_C = 4/5$  respectively. The results presented above clearly underline the advantage of the JSCC over a traditional separated scheme; besides its superior performance in presence of MAP decoding, the proposed scheme presents a high degree of flexibility. First of all the coding rate is a simple function of  $\epsilon$  and it is not limited to a finite set of puncturing rates as in the RCPC case. Moreover, the coding engine can be designed adaptively, where both the source model and the added redundancy can be adjusted according to the statistic of the data and the required level of error protection respectively. Finally, a decoder with reduced computational complexity can be designed, which gives up error correction and simply detects transmission error by means of the forbidden symbol.

In Tab. 3 and 4 we report results in terms of PER, along with the average frame decoding time (T) in ms, obtained on a Pentium IV with 512MB RAM. The channel transition probability is fixed to  $p = 10^{-3}$ , and different values of coding rate are investigated; results in Tab. 3 are worked out on *Foreman* MVs at 15 fps, while those in Tab. 4 are obtained with the same video sequence at 30 fps.

It is worth pointing out that the MAP scheme ex-

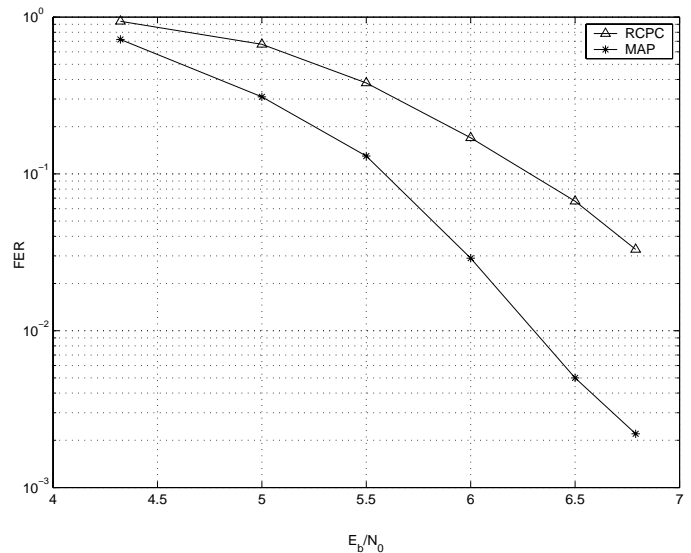
hibits a gain of at least one order of magnitude in terms of PER in all simulated cases. It can be noticed that its performance depends on the input data statistics as well; in fact both the compression efficiency and the PER performance are improved in the case of the 30 fps sequence. In this latter condition the reduced magnitude of MVs yields a higher value of  $P_0$ , that affects both source and channel coding: from one hand the coding rate can be reduced, given a certain value of coding redundancy; from the other hand the MAP estimation task is eased by the more accurate a priori knowledge, because of the reduced number of the most likely input sequences. On the contrary the RCPC performance is dominated only by the correction capacity at each coding rate and PER results in Tab. 3 and 4 are the same.

Another crucial point is represented by the SA computational complexity. It was previously stated that the algorithm complexity depends on the metric (2) and in particular the decoding time is affected by both the value of  $P_0$  and  $p$ , characterizing the source and the channel statistics respectively. On top of that the decoding task is affected by the value of coding redundancy  $\epsilon$ ; in fact a large value of  $\epsilon$  guarantees fast error detection and allows quick pruning of erroneous paths in the tree, speeding up the random search performed by SA. This dependency on the value of  $\epsilon$  is clearly noticeable in the first two rows of Tab. 3 and 4, where the average decoding time exhibits a reduction by an order of magnitude switching from  $\epsilon = 0.035$  to  $\epsilon = 0.065$ . By comparing the two tables, it is worth noticing the effect of the source statistics as well; a slight increase of 0.01 in the value of  $P_0$  is able to reduce  $T$  by almost a factor 2. In Tab. 4 the best performance is obtained decoding MVs at 30 fps in the case  $\epsilon = 0.065$  where the average decoding time is as low as 27 ms; this result is really surprising if compared with the cost of standard binary arithmetic decoding that would take 14 ms. For sake of comparison, the average decoding time of the RCPC scheme is also reported, and it is about 70 ms. Finally, MAP decoding delay is affected by the channel state as well; in the worst simulated case ( $p = 10^{-2}$ ) the JSCC becomes quite prohibitive in terms of computational complexity, taking tens of seconds per packet.

To summarize, the proposed MAP technique exhibits an excellent performance in terms of error resilience and decoding complexity. The JSCC approach exhibits attractive performance especially in the case of highly compressed data with limited redundancy

**Table 2.** *Foreman* MVs characteristics at 15 and 30 fps respectively; number of available MVs, value of  $P_0$  and achieved compression rate in absence of the forbidden symbol.

Frame rate	MVs	$P_0$	$R$
15	170249	0.92	0.41
30	179519	0.93	0.38



**Figure 2.** Packet Error Rate (PER) obtained by MAP estimator (star markers) and RCPC scheme on *Foreman* at 30 fps; compression ratio  $R = 0.43$ .

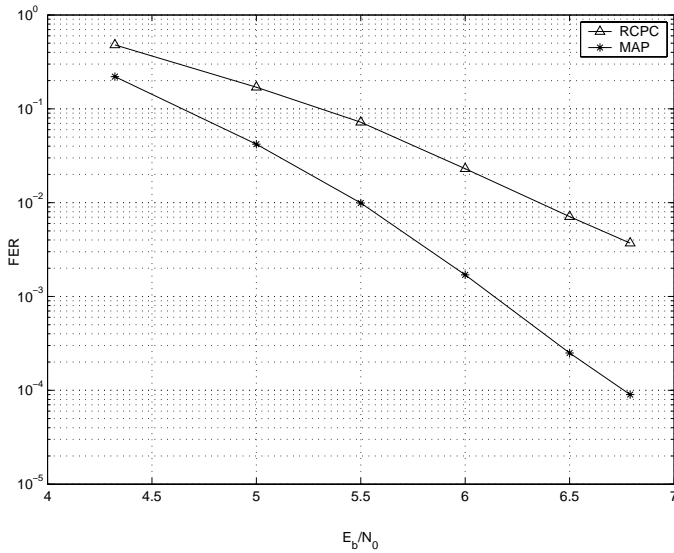
for error correction purpose.

## 5. CONCLUSIONS

A JSCC technique in order to embody error correction capability into AC has been proposed. The implemented decoder, based on a MAP metric and sequential decoding, exhibits excellent performance in terms of packet recovery and its computational complexity turned out to be very attractive. Moreover the

**Table 3.** Packet Error Rate (PER) and average packet decoding time (T) for *Foreman* sequence at 15 fps, in the case  $p = 10^{-3}$ .

alg.	$\epsilon$	R	PER	$T$ (ms)
MAP	$3.5 \cdot 10^{-2}$	0.46	$3.4 \cdot 10^{-3}$	360
MAP	$6.5 \cdot 10^{-2}$	0.5	$3.6 \cdot 10^{-4}$	40
RCPC	n.d.	0.46	$3.6 \cdot 10^{-2}$	70
RCPC	n.d.	0.50	$3.8 \cdot 10^{-3}$	72



**Figure 3.** Packet Error Rate (PER) obtained by MAP estimator (star markers) and RCPC scheme on *Foreman* at 30 fps; compression ratio  $R = 0.47$ .

**Table 4.** Packet Error Rate (PER) and average packet decoding time ( $T$ ) for *Foreman* sequence at 30 fps, in the case  $p = 10^{-3}$ .

alg.	$\epsilon$	R	PER	$T$ (ms)
MAP	$3.5 \cdot 10^{-2}$	0.43	$2.2 \cdot 10^{-3}$	200
MAP	$6.5 \cdot 10^{-2}$	0.47	$9 \cdot 10^{-5}$	27
RCPC	n.d.	0.43	$3.3 \cdot 10^{-2}$	70
RCPC	n.d.	0.47	$3.7 \cdot 10^{-3}$	72

proposed algorithm presents an high degree of flexibility, being rate compatible and open to future developments such as joint source and channel adaptive arithmetic coding.

## 6. REFERENCES

- [1] ISO/IEC FCD 15444-1, *JPEG2000 image compression standard*, 1999.
- [2] Joint Video Team JVT of ISO/IEC MPEG and ITU-T VCEG, *Joint Committee Draft (CD)*, May 2002.
- [3] C. Boyd, J. Cleary, S. Irvine, I. Rinsma-Melchert, and I. Witten, "Integrating error detection into arithmetic coding," *IEEE Trans. Commun.*, vol. 45, no. 1, pp. 1–3, Jan. 1997.
- [4] J. Chou and K. Ramchandran, "Arithmetic coding-based continuous error detection for efficient ARQ-based image transmission," *IEEE J. Select. Areas Commun.*, vol. 18, no. 6, pp. 861–867, June 2000.
- [5] R. Anand, K. Ramchandran, and I. V. Kozintsev, "Continuous error detection (CED) for reliable communication," *IEEE Trans. Commun.*, vol. 49, no. 9, pp. 1540–1549, Sept. 2001.
- [6] B.D. Pettijohn, M.W. Hoffman, and K. Sayood, "Joint source/channel coding using arithmetic codes," *IEEE Trans. Commun.*, vol. 49, no. 9, pp. 1540–1548, Sept. 2001.
- [7] C. Demiroglu, M.W. Hoffman, and K. Sayood, "Joint source channel coding using arithmetic codes and trellis coded modulation," in *Proc. of DCC 2001*, Mar. 2001, pp. 302–311.
- [8] M. Grangetto and P. Cosman, "MAP decoding of arithmetic codes with a forbidden symbol," in *Proc. of ACIVS 2002*, Ghent, Belgium, Sept. 2002.
- [9] D. Marpe, H. Schwarz, G. Heising, and T. Wiegand, "Context-based adaptive binary arithmetic coding in jvt/h.261," in *Proc. of ICIP 2002*, Sept. 2002.
- [10] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Trans. Commun.*, vol. 36, no. 4, pp. 389–400, Apr. 1988.