LOW COMPLEXITY DECODING OF VARIABLE LENGTH SOURCE-CHANNEL CODES

Hugues Mercier

Université de Neuchâtel
Institute of Computer Science
2000 Neuchâtel, Switzerland

Abdul Wasae, Fabrice Labeau

McGill University
Electrical and Computer Engineering
Montréal (Québec) Canada H3A 0G4

ABSTRACT

Soft source decoders, in conjunction with error correcting channel codes, can be used to improve the error resilience of digital communication systems based on variable length codes. In this paper, we present a novel approach to reduce the complexity of maximum a posteriori variable length decoders implemented on a bit-symbol trellis. The decoding algorithm is implemented in a narrow corridor along the trellis diagonal to reduce the decoder complexity. Furthermore, by periodically adjusting the corridor boundaries, a significant reduction in complexity is achieved at the price of a small degradation in decoding performance.

***Index Terms*—** Exponential Golomb Code, variable length joint source-channel decoder, maximum a posteriori algorithm, bit-symbol trellis, corridor decoding.

1. INTRODUCTION

Most current data compression schemes use variable length codes (VLC). Latest image and video compression standards achieve compression by removing redundancy from the source symbols, however this makes them very sensitive to channel noise. Furthermore, hard decoding results in error propagation and synchronization losses. To counter these issues various schemes can be used that make use of source statistics and soft information at the channel output [1]-[4]. These also include the use of joint source-channel designs exploiting the residual source redundancy to achieve error resilience [2, 3]. All these schemes assume that some a priori information is available at the decoder: either the number of symbols or the number of bits (or both) is sent as side information to the receiver. The classical maximum a posteriori (MAP) decoding algorithm is used either to search for the optimal path through the trellis in case of sequence estimation, or to calculate symbol a posteriori probabilities in case of symbol by symbol estimation. However, when used on variable length codes, MAP decoding is costly due to the lack of synchronization between symbol sequences and coded bit sequences, resulting in algorithms with prohibitive complexity. The problem of reducing this high complexity has been addressed in several works [5]-[7].

In this paper, we use a modified MAP decoding algorithm taking advantage of the soft information from the corrupted received sequences and the residual redundancy of the source code. Inspired by a technique originally proposed in [8] for channel codes against synchronization errors, we adapt it to reduce the complexity of VLC source-channel decoders. Unlike prior work, this technique neither requires any modification of the VLC tables nor does it require merging states. Instead, we use a generic bit-symbol trellis [4, 9] and reduce its size by making use of the source statistics and received sequence.

Our first insight is to discard the nodes of the trellis which are less likely to be followed by the true path. The MAP algorithm is thus effectively implemented on the part of the trellis formed by a narrow corridor along the diagonal. This reduces the trellis size, with possible degradation in decoding performance depending upon the corridor width. Our second insight is to divide the symbol stream into blocks and to implement the MAP decoding algorithm on each block. After every block, the position of the corridor is adjusted towards the most likely location of the true path, resulting in a large portion of the trellis being pruned. This further reduces significantly the size of the memory required by the decoding algorithm. Finally, we show that corridor decoding can work for practical systems by proving that it performs well when the input statistics are not accurately known at the decoder. The code used in this paper is the Exponential Golomb Code (EGC), although we note that the proposed techniques could be used with any variable length code.

2. EXPONENTIAL GOLOMB CODE AND MAP DECODING ALGORITHM

Most data compression methods based on variable length codes employ Huffman or Golomb codes, and in this paper we use Exponential Golomb Codes (EGC), which are used in video compression standards such as H.264/MPEG-4 AVC [10]. Table 1 shows the first three classes of the EGC: Class j has 2^{j-1} codewords, each codeword consisting of a sequence of $j-1$ '0's, followed by a 1, followed by $j-1$ more bits. One of the advantages of EGCs is their low redundancy, but unfortunately, like all good VLCs, they tautologically have poor error-correction capabilities (the minimum distance of EGCs is 1).

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

Class	Value	Codeword
C_1	0	1
C_2	1	010
	2	011
C_3	3	00100
	4	00101
	5	00110
	6	00111

Table 1. Exponential Golomb Code (EGC)

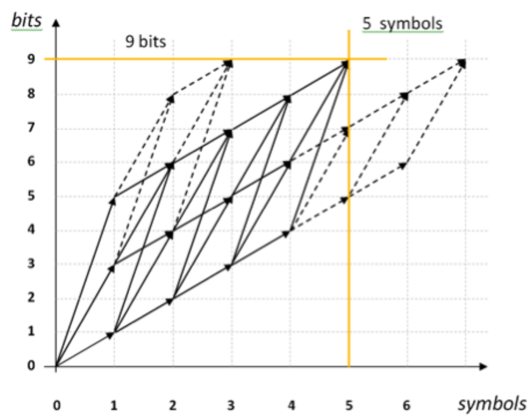
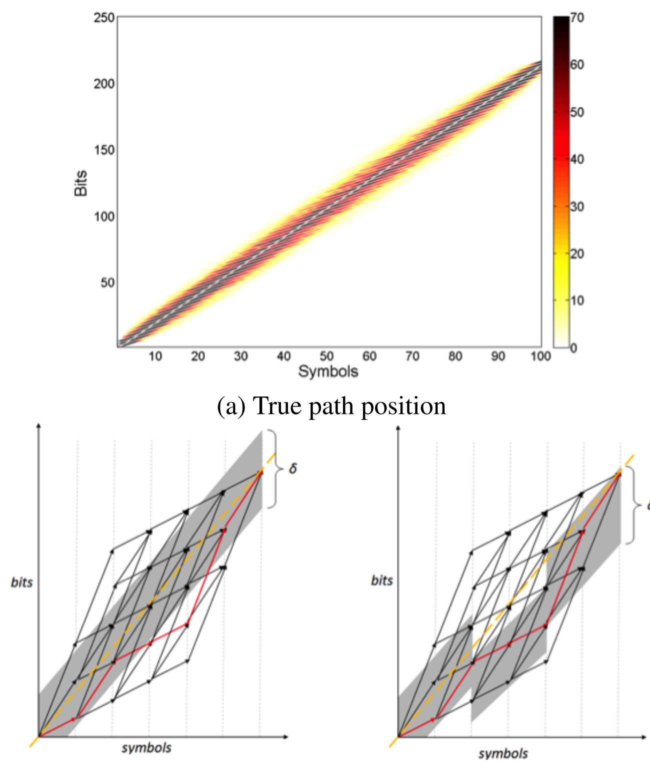


Fig. 1. Bit-symbol trellis of the first 7 codewords of the EGC. Knowledge of the number of bits (9) and symbols (5) eliminates paths not containing 9 bits and 5 symbols (dotted lines).

The correspondence between symbols, codewords, and corrupted received bit sequences can be represented using a bit-symbol trellis, as shown in Figure 1. Since it is assumed that the number of transmitted bits and symbols is known at the receiver, the trellis termination point is known and the possible paths in the trellis are bounded by a parallelogram. In Figure 1, the receiver knows that 5 symbols corresponding to 9 bits were transmitted, thus the trellis is represented by the solid lines. A BCJR-like [9, 11, 12] maximum a posteriori (MAP) decoding algorithm is used to decode the EGC on the bit-symbol trellis. For variable length codes, the BCJR algorithm must be modified, but for space reasons we do not describe it here and point the readers to the extended version of this work for additional details [13].

3. CORRIDOR DECODING

The bit-symbol trellis has a prohibitive decoding complexity in $\Theta(k^2)$, where k is the number of symbols in a packet. Our first contribution to reduce the complexity is to decode along a narrow corridor of width δ along the trellis diagonal, as shown in Figure 2(b). This is supported by the fact that the deviation of the true path from the diagonal is limited by the probability distribution of the VLC and the number of symbols. To illustrate this, Figure 2(a) shows the color map of 600 paths taken at random in a bit-symbol trellis. The code used is the EGC with optimal probability distribution [14]. The colors represent the frequency with which nodes were traversed. For instance, the black nodes were traversed by the true path more than 70 times. The white line in the diagonal represents the average length of the variable length code in bits per symbol. It is evident from the figure that for fixed channel conditions the true path almost always stays within a narrow corridor along the diagonal of the trellis. In the following subsection, we show how the proper width for this corridor can be determined.



(a) True path position

(b) Fixed corridor

(c) Adjustable corridor

Fig. 2. Decoding in a narrow corridor around the most likely path in the trellis.

3.1. Corridor Width

The path taken by the decoder is expected to deviate from the trellis diagonal depending on the source statistics and the number of symbols in a packet. Consider an EGC with N classes and $2^N - 1$ codewords, with Codeword i having n_i bits and occurring with probability p_i. The average codeword length $\overline{n}$ and its variance σ_c^2 are given by $\overline{n} = E[n] = \sum_{i=0}^{2^N-1} n_i p_i$ and $\sigma_c^2 = Var[n] = \sum_{i=0}^{2^N-1} n_i^2 p_i - \left(\sum_{i=0}^{2^N-1} n_i p_i\right)^2$.

Let us assume that an EGC randomly generates k symbols according to probabilities p_i. The total length $\hat{n}_{1:k}$ of the packet in bits is given by $\hat{n}_{1:k} = \sum_{j=1}^{k} n_j$ with mean $E[\hat{n}_{1:k}] = E\left[\sum_{j=1}^{k} n_j\right] = \sum_{j=1}^{k} E[n_j] = k \cdot \sum_{i=0}^{2^N-1} n_i p_i$ and variance $Var[\hat{n}_{1:k}] = Var[n_1 + n_2 + \cdots + n_k]$. Since all the symbols are assumed to be independent it follows that $Var[\hat{n}_{1:k}] = k\left(\sum_{i=0}^{2^N-1} n_i^2 p_i - \left(\sum_{i=0}^{2^N-1} n_i p_i\right)^2\right)$. Therefore, the standard deviation σ in bits of the true path from the diagonal in terms of the codeword length n_i, probability distribution p_i and number of symbols k is given by

$$\sigma = \sqrt{k\left(\sum_{i=0}^{2^N-1} n_i^2 p_i - \left(\sum_{i=0}^{2^N-1} n_i p_i\right)^2\right)} = \sqrt{k}\sigma_c.$$

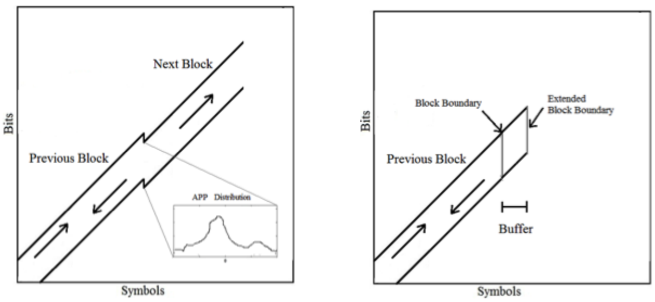
We therefore select the corridor width as $\delta \triangleq c\sqrt{k}\sigma_c$, where c is a scaling factor. Only paths within this corridor are considered during the decoding process, which reduces the computational cost of the decoder from $\Theta(k^2)$ to $\Theta(k\sqrt{k})$.

4. ADJUSTABLE CORRIDOR DECODING

Our second contribution is to further decrease the decoding complexity by making the corridor adjustable, as shown in Figure 2(c). The decoder periodically adjusts the corridor in the direction of the most likely path, and by doing this the corridor can be made much narrower. The use of an adjustable corridor was first proposed in [8]. In the previous section, we showed that the number of symbols per packet is one of the factors affecting the corridor width. If the packet size is reduced, then the corridor width required to keep the most likely path within the corridor is also reduced. The adjustable corridor further exploits this idea by dividing the packets into smaller blocks. The corridor width becomes a function of the block size rather than the packet size, hence the decoding complexity is decreased to $\Theta(k)$.

The forward and backward recursions of the MAP decoding algorithm are initially executed on the first block. In order to adjust the corridor, the node with the highest a posteriori probability (APP) at the end of the block is used as the center of the corridor for the next block, as shown in Figure 3(a). The process is repeated for the following blocks, and the corridor position is updated after each block until the end of the symbol stream is reached. The corridor thus closely follows the most likely path with high probability. When implementing the MAP decoding algorithm on a block, we need to be careful of the way we set the initial conditions for the forward and backward recursions. For the forward recursion, the APP distribution of the nodes at the end of the previous block is taken as the a priori probability distribution at the beginning of the next block. For the backward recursion, the probability distribution of the nodes at the end of forward recursion is taken as the initial condition.

We add a buffer at the end of each block to form an extended block when performing the forward/backward recursions, as shown in Figure 3(b). This provides a better estimate of the APPs of the nodes at the block boundaries. The forward recursion of the next block starts at the original block boundary rather than at the extended block boundary.



(a) Adjusting the corridor (b) End of block buffer

Fig. 3. Decoding at block boundaries.

5. SIMULATION RESULTS

In this section, we apply our MAP decoding algorithm on a bit-symbol trellis using the proposed fixed and adjustable corridor decoding techniques. The simulations are done over an AWGN channel using BPSK modulation. Only a variable length source code is used without any other error-correcting mechanism. The source is memoryless and generates random symbols from the EGC codebook with an experimental probability distribution generated from four standard video sequences [15]: Stefan, News, Football and Foreman, with QCIF size (176x144) each having 100 frames encoded using the H.264 standard. More details regarding the simulation parameters can be found in the extended version of this work [13]. The number of symbols and bits in a packet are assumed to be known at the receiver, and each packet has 1000 symbols. The results show the SNR against the average symbol error rate of 5000 simulation runs. The Levenshtein distance [16] is used as the error rate metric; this is more suitable than the Hamming distance due to the fact that the decoding of variable length codes suffers from synchronization errors.

We now present a small sample of our results. First, Figure 4 shows the performance of the fixed corridor decoder with different corridor widths. It should be pointed out that decoding over the full trellis was indistinguishable from a corridor width of 6 standard deviations for all our simulations. The decoder complexity is determined by measuring the average time taken for a simulation. For a stream of 1000 symbols and a corridor width of 4 standard deviations from the trellis diagonal, the fixed corridor decoder is 4.7 times faster than the full trellis decoder, with no performance degradation at SNR below 9dB. At low SNR, the corridor width can be reduced further since the channel becomes dominated by the errors caused by the poor correction capabilities of the codes instead of the probability of leaving the decoding corridor. Simulation results for the adjustable corridor decoder can be found in [13], although we mention that the adjustable corridor decoder is approximately 15 times faster than the full trellis decoder with a corridor width of 4 standard deviations from the trellis diagonal.

The performance comparison of fixed and adjustable corridor decoding, shown in Figure 5, is done by setting the cor-

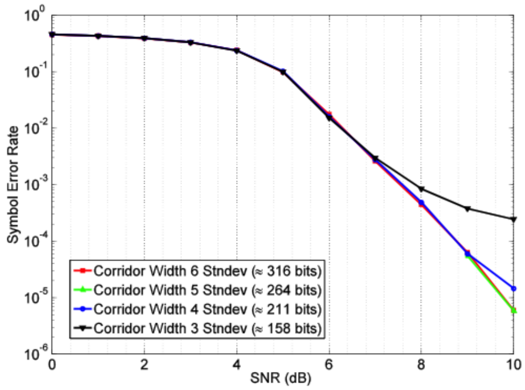


Fig. 4. SER versus SNR performance of the fixed corridor decoder for several corridor widths. Corridor widths are shown as multiples of the standard deviation of the EGC from the expected path.

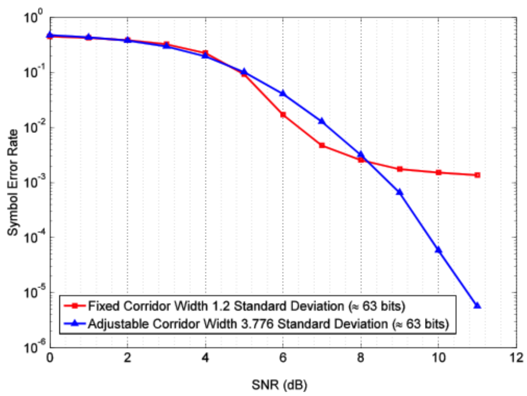


Fig. 5. SER versus SNR performance of the adjustable and fixed corridor decoders of equal complexities with 1000 symbol packets. The block size of the adjustable corridor decoder is 100 and the buffer size is 1.

ridor width such that both methods have equal decoding complexities. It shows that the adjustable corridor significantly outperforms the fixed corridor at high SNR. This is due to the fewer synchronization errors introduced at the block boundaries at high SNR, which allow very narrow adjustable corridors.

5.1. Matched vs Mismatched Input Statistics

The probability distribution over the codewords used in the previous section, which we denote as $\mathcal{P}_1$, was taken from standard video sequences [15]. However, since the input statistics are not always known at the decoder, a comparison is made between matched and mismatched input-output statistics to compare the decoder performance in both cases. The matched scenario is further divided into two cases: the first case uses the practical distribution $\mathcal{P}_1$, whereas the second case uses the theoretical EGC distribution, denoted $\mathcal{P}_2$, which minimizes the residual code redundancy and given by $p_i = \frac{2^{-(2i-1)}}{1-2^{-N}}$.

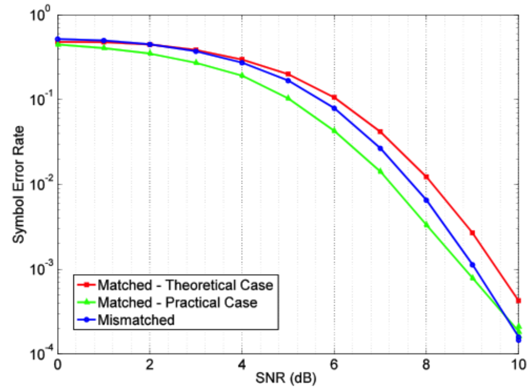


Fig. 6. Matched versus unmatched input-output statistics: SER versus SNR performance of the adjustable corridor decoder.

In the matched cases both the encoder and decoder use the same codeword probability distribution. On the other hand, in the mismatched case the input data is based on $\mathcal{P}_1$ and the decoding is based on $\mathcal{P}_2$. Figure 6 shows the performance of the adjustable corridor decoder for the three cases mentioned above. The matched case with the practical distribution $\mathcal{P}_1$ performs slightly better. This is simply because $\mathcal{P}_1$ has a lower entropy than $\mathcal{P}_2$. The performance degrades for the mismatched case, however this degradation is small considering that the exact input distribution is not available at the decoder. This provides evidence that corridor decoding can work for real-life practical systems.

6. CONCLUSION

We presented a novel approach to reduce the complexity of a joint source-channel MAP decoding algorithm implemented on a bit-symbol trellis. We showed that a random path taken through the trellis has a high probability of staying within a narrow corridor along the trellis diagonal. We used this behavior to show that pruning the trellis by using a corridor of reasonable width, fixed or adjustable, significantly reduces the computational cost of the decoder with a marginal effect on its accuracy. We also showed that the adjustable corridor performs better at high SNR and is therefore the clear decoding choice when lower complexity is required.

The MAP decoding algorithm implemented in this paper uses only a VLC code. The performance could be improved by incorporating an inner error-correcting code for both the fixed and adjustable corridor decoders. Furthermore, the decoder takes soft information as inputs and returns soft outputs in the form of the a posteriori probabilities of the symbols. This soft output could further be utilized in iterative decoders, where information is passed between two or more decoders in an iterative process to improve performance.

7. REFERENCES

[1] S. Kaiser and M. Bystrom, "Soft decoding of variable-length codes," in *Proceedings of the 2000 IEEE International Conference on Communications (ICC)*, 2000, pp. 1203–1207.

[2] A. H. Murad and T. E. Fuja, "Robust transmission of variable-length encoded sources," in *Proceedings of the 1999 Wireless Communications and Networking Conference (WCNC)*, 1999, pp. 968–972.

[3] M. Park and D. Miller, "Joint source-channel decoding for variable-length encoded data by exact and approximate MAP sequence estimation," *IEEE Transactions on Communications*, vol. 48, no. 1, pp. 1–6, 2000.

[4] R. Bauer and J. Hagenauer, "Symbol by symbol MAP decoding of variable length codes," in *Proceedings of the 3rd ITG Conference on Source and Channel Coding*, 2000.

[5] Tsu ming Liu, Sheng zen Wang, Wen hsiao Peng, and Chen yi Lee, "Memory efficient and low complexity scalable soft VLC decoder for the video transmission," in *Proceedings of the 2005 IEEE Asia-Pacific Conference on Circuits and Systems*, 2005, pp. 673–676.

[6] G. Mohammad-Khani, C. Lee, M. Kieffer, and P. Duhamel, "Simplification of VLC tables with application to ML and MAP decoding algorithms," *IEEE Transactions on Communications*, vol. 54, no. 10, pp. 1835–1844, 2006.

[7] H. Nguyen and P. Duhamel, "Reduced complexity VLC sequence decoder," in *Proceedings of the 2004 Data Compression Conference*, 2004.

[8] Matthew C. Davey and David J. C. MacKay, "Reliable communication over channels with insertions, deletions and substitutions," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 687–698, 2001.

[9] P. Duhamel and M. Kieffer, *Joint Source-Channel Decoding: A Cross-Layer Perspective with Applications in Video Broadcasting over Mobile and Wireless Networks*, pp. 123–128, EURASIP and Academic Press Series in Signal and Image Processing, 2010.

[10] D. Marpe, T. Wiegand, and G. Sullivan, "The H.264/MPEG–4 advanced video coding standard and its applications," *IEEE Communications Magazine*, vol. 44, no. 8, pp. 134–143, 2006.

[11] Shu Lin and Daniel J. Costello, *Error Control Coding*, Paerson Prentice Hall, second edition, 2004.

[12] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284–287, 1974.

[13] Abdul Wasae, "Low complexity decoding of variable length codes," M.Eng. Report, McGill University, 2011.

[14] Pierre Siohan, "Exponential golomb codes and their application to JSCD," Private communication, 2005.

[15] Available at `http://media.xiph.org/video/derf/`.

[16] Vladimir I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics-Doklady*, vol. 10, no. 8, pp. 707–710, Feb. 1966, Translated from *Doklady Akademii Nauk SSSR*, 163(4):845-848, 1965 (in Russian).