Huffman Tree Based Metric Derivation for a Low-complexity Sequential Soft VLC Decoding

Lisa Perros-Meilhac - Catherine Lamy

***Abstract*— This paper considers VLC decoding algorithms based on MAP sequence estimation techniques, using residual source redundancy to provide channel error correction. These algorithms rely on soft values available at the entrance of the VLC decoder. We present here a new soft VLC decoding algorithm based on a sequential decoding technique that is very efficient in terms of decoding complexity.**

The application of the considered soft decoding algorithms to practical decoding of MPEG-4 texture information packets under the assumption of an unequal protection scheme is investigated. Algorithm performance is evaluated on the well-known "Foreman" video sequence. Simulation results show that the proposed algorithm provides approximately the same performance as all existing soft decoding algorithms while exhibiting a significantly lower complexity.

***Index Terms*—sequential decoding, MAP estimation, soft-input decoding, variable length codes**

I. Introduction

Transmission systems operating over bandwidth-limited channels use source encoding to compress the bitstream by reducing the redundancy inherent in the source symbols. One of the most famous compression techniques, commonly used by many existing compression standards, is the variable length coding. It provides highly compressed sources that make the transmission bandwidth-efficient, but unfortunately generate bitstreams very sensitive to channel perturbations. Thus, without any error correction scheme, the quality of the reconstructed source can be dramatically affected by the errors caused by the channel, in particular in the harsh conditions of wireless channels. For this reason, channel coding is generally applied to provide an appropriate level of protection, but at the price of bandwidth expansion.

The new decoding techniques for variable length codes (VLC) considered here provide channel protection without using additional bandwidth. As shown recently by several authors [1], [2], [3], the key point is to use appropriately the residual source redundancy at the decoding part: this redundancy can be considered as a form of implicit channel protection by the decoder, and be exploited as such to provide error correction capability for the variable length coded source. The optimal VLC decoder acts, then, as an estimator of the transmitted information by selecting the sequence maximising the maximum *a posteriori* (MAP) criterion. While valid for both hard and soft input decoders, this sequence estimation technique shows its real strength for soft input. In this last case, the improvement achieved when compared to classical hard decoding is significant, but the complexity is prohibitive. As a consequence, following the works of Demir & Sayood [1], and Park & Miller [2], several different approximate MAP decoders have been proposed [3], [4], [5], [6]. Some of these algorithms reach the optimal soft-input performance but become too complex or even impractical when the VLC table size increases. In this contribution, the goal was set to focus on decreasing significantly the decoding complexity without allowing a noticeable performance degradation.

This work was partially supported by the European Community in the frame of the IST project IST-2000-29511 JOCO. L. Perros-Meilhac and C. Lamy are with Philips Recherche France (PRF), 51 rue Carnot, B.P. 301, 92156 Suresnes Cedex, France (email: catherine.lamy@philips.com)

This paper is organised as follows. In section II, we recall the principle of sequence estimation based on soft VLC decoding, consisting in associating a "metric" to each considered bit sequence, and we consider the problem of metric derivation. In section III, we introduce the use of the sequential stack algorithm to decode VLC sequences and present a new algorithm. In section IV, we achieve comparative simulations in the context of MPEG-4 video sequences. We show that soft decoding provides a very significant visual gain compared to the classical hard one and that the proposed algorithm is the most efficient in terms of decoding complexity.

II. MAP VLC decoding

The communication model considered is presented in Figure 1, and consists of a VLC encoder, a channel and a VLC decoder. The VLC table is of size K and supposed to be tree-structured: in practice it will be determined by following the Huffman construction method. Let $x[t]$ be the t^{th} bit of the transmitted sequence, and use the following notation:

$$\mathbf{x}[t:t+k] = [x[t], x[t+1], \ldots, x[t+k]].$$

The MAP rule, used by the recently proposed soft input decoders, corresponds to the search of the best sequence, *i.e.* the sequence satisfying:

$$\hat{\mathbf{x}}[1:T] = \arg\min_{\mathbf{x}[1:T]} \mathcal{M}(\mathbf{x}[1:T], \mathbf{y}[1:T]),$$

where T is the considered sequence length and

$$\begin{aligned}\mathcal{M}(\mathbf{x}[t:u], \mathbf{y}[t:u]) = &-\log P(\mathbf{y}[t:u]|\mathbf{x}[t:u]) \\ &-\log P(\mathbf{x}[t:u]) + \log P(\mathbf{y}[t:u]).\end{aligned}$$

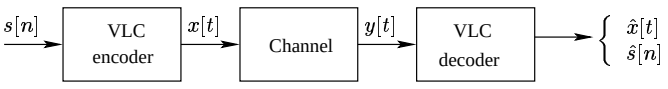


Fig. 1. Communication system model.

The exact calculation of the term $\log P(\mathbf{y}[t:u])$ is complex and can be computationally prohibitive for most systems. Some algorithms *e.g.* [3], [4], [6], [7] rely on the derivation of the metric over different streams corresponding to the same part of received bit sequence. In this case, the term $\log P(\mathbf{y}[t:u])$ is identical for all the possible compared sequences and thus it can be omitted without any suboptimality. But, in the case of the sequential algorithms considered in this paper, the compared sequences do not correspond to the same received bit sequence. Thus, the term $\log P(\mathbf{y}[t:u])$ cannot be neglected without a significant degradation of performance. Park and Miller [4] have suggested to approximate it by $(t-u)\log(2)$, but this remains insufficient. Other authors [9], [5] have applied the metric proposed by Fano [10] for the sequential decoding of convolutional codes and adapted by Massey [11] to the variable length codes context. This consists in replacing $P(\mathbf{y}[t:u])$ by the expression $P_0(\mathbf{y}[t:u])$ where

$$P_0(\mathbf{y}[t:u]) = \prod_{k=t}^{u} P_0(y[k]), \tag{1}$$

and

$$P_0(y[k]) = \sum_{i=0}^{1} P(x[k]=i)P(y[k]|x[k]=i). \tag{2}$$

In most of the cases, $P_0(\mathbf{y}[t:u]) \neq P(\mathbf{y}[t:u])$ and thus the metric $\tilde{\mathcal{M}}(\mathbf{x}[t:u],\mathbf{y}[t:u])$ considering this replacement is only an approximation of the MAP metric $\mathcal{M}(\mathbf{x}[t:u],\mathbf{y}[t:u])$. However, simulation results show that using the Fano-Massey metric provides very close performance to the optimal one. We will consequently use it in our own derivations.

Moreover, only sequences $\tilde{\mathbf{x}}$ corresponding to streams of variable length codewords belonging to the VLC table can be solutions and therefore must be examined. Thus, for each considered sequence $\tilde{\mathbf{x}}[1:T]$, a sequence of R codewords indexed by τ exists, such that $\tilde{\mathbf{x}} = [C_{\tau(1)},\dots,C_{\tau(R)}]$ where C_i denotes the i^{th} codeword, $i=1,\dots,K$. The metric associated to this sequence verifies the following symbol by symbol recursive formulation:

$$\tilde{\mathcal{M}}(\tilde{\mathbf{x}}[1:T],\mathbf{y}[1:T]) = \sum_{i=1}^{R} \tilde{M}\Big(C_{\tau(i)},\mathbf{y}[\sum_{j=1}^{i-1} l_{\tau(j)} : \sum_{j=1}^{i} l_{\tau(j)} - 1]\Big),$$

where l_i denotes the length of C_i. Such a formulation relies on the "codeword metrics", *i.e.* on metrics corresponding to a given codeword at a given bit time. It is easy to see that the metric of the codeword C_i at bit time t is given by:

$$\tilde{M}(C_i,\mathbf{y}[t:t+l_i-1]) = -\log P(\mathbf{y}[t:t+l_i-1]|C_i) - \log P(C_i) + \log P_0(\mathbf{y}[t:t+l_i-1]). \tag{3}$$

Moreover, the codeword metric can be derived at the bit level as [7]:

$$\tilde{M}(C_i,\mathbf{y}[t:t+l_i-1]) = \sum_{k=0}^{l_i-1} \tilde{m}(C_i[k],y[t+k]),$$

where $C_i[k]$ is the k^{th} bit of C_i and

$$\tilde{m}(C_i[k],y[t+k]) = \underbrace{-\log P(y[t+k]|C_i[k])}_{T_1} \underbrace{-\log P(x[t+k]=C_i[k]|\mathbf{x}[t:t+k-1])}_{T_2} + \underbrace{\log P_0(y[t+k])}_{T_3}. \tag{4}$$

In the presence of an additive white Gaussian noise (AWGN),

$$T_1 = \|y[t+k]-C_i[k]\|^2/2\sigma^2 + Q, \tag{5}$$

where Q is a constant term which has no influence for metric comparison and can hence be omitted. In the same way, the term T_3 can be easily derived knowing the *a priori* probabilities of transmitted bits. Finally, the term T_2 can be directly obtained as explained in [7] from the tree representation of the VLC table and the codeword probabilities which are assumed to be known by the decoder. These probabilities come either directly from the source or from an estimation algorithm [1] [8].

III. Sequential decoding

In this section, we briefly recall the principle of sequence estimation using the stack algorithm in the context of convolutional code and VLC, then we present our improved version having a reduced-complexity.

A. The stack and the VLC-stack algorithms

The stack or ZJ algorithm was introduced independently by Jelinek [12] and Zigangirov [13] for the decoding of convolutional codes. As all sequential decoding algorithms, this algorithm is based on the tree representation of the codes and relies on the search of the most likely emitted sequence examining the nodes of the considered code tree. The goal is to find such a sequence or "path" in an efficient way, *i.e.* without examining too many nodes of the tree. When compared to Viterbi decoding, the main advantage of sequential decoding is that when there is no error in the received sequence, the correct path is extended immediately and the decoding process is much faster that the Viterbi one. In the stack algorithm, an ordered stack of previously examined path (of different length) is stored and at each time the most likely path in the stack is extended. As paths of different length are compared, the specific branch metric introduced by Fano in [10] is used. The algorithm can be summarised in the following way [14]:

- Step 1. Load the stack with the tree origin node and assign a metric 0 to this path.

- Step 2. Compute the metric of the successors of the top path in the stack.
- Step 3. Delete the top path from the stack.
- Step 4. Insert the new paths in the stack, and reorder paths according to metric values.
- Step 5. If the top path ends stop. Otherwise, return to step 2.

The adaptation of the stack algorithm to VLC decoding, that we will call as VLC-stack algorithm, has been proposed by Buttigieg [9]. In this case, one codeword of each length generates a successor path to the top path in the stack. The approximated Massey metric presented in the previous section is used to compare paths. It has been shown [9] that the size of the stack can be limited to a quite small value without degrading appreciably the performance, preventing an infinite growth of the stack.

B. A reduced complexity VLC-stack algorithm

As most sequence estimation-based VLC soft decoding algorithms, the VLC-stack requires the computation of the metric associated to each codeword at each decoding step, making the metric derivation the most costly part of decoding in term of CPU (central processing unit) requirements. As the size of the stack is limited, successors of the top path having a cumulative metric greater than the one of the worst path in the stack will not be kept. In this section, we show that it is possible to examine and select codewords in the increasing order of their metric value, and thus to reduce the complexity of the VLC-stack by deriving only the useful metrics *i.e.* metrics associated to extended paths which will be inserted in the stack.

The method consists of applying a kind of stack algorithm onto the Huffman tree representing the considered VLC table, in order to select the most likely codewords at the considered time. More precisely, the selection of useful codewords is achieved in the following way:

- Step 1. Load a stack with the origin node of the Huffman tree and assign a metric 0 to this tree-path.
- Step 2. Compute the metric of the (one or two) succeeding branches of the top tree-path in the stack using the branch-metric defined by Equation 4.
- Step 3. Delete the top tree-path from the stack.
- Step 4. Insert the extended tree-paths in the stack.
- Step 5. Test the following stop conditions:
  - at least, one codeword has been selected,
  - the metric of the top tree-path added to the top path of the VLC-stack algorithm is greater than the metric of the worst path of the VLC-stack algorithm.

If the two stop conditions are satisfied, stop. Otherwise, continue to step 6.

- Step 6. If the top tree-path corresponds to a complete codeword, add it to a list of main best codewords, then return to step 2.

In most cases, very few codewords are selected, often even only one. They are employed to generate the successor paths of the top path of the stack.

As in many decoding algorithms, it is possible to improve the performance of the algorithm employing additional *a priori* information (*e.g.* the number of symbols by sequence). In this case, when a top path containing the correct number of bits, but not verifying the additional information, is considered, it is deleted and the decoding continues considering the new top path.

As for all sequential decoding algorithms, the complexity of this algorithm depends mainly on the noise level. Noisy sequences will take more computation time as the algorithm will go forth and back with examining false paths and tree-paths. To avoid excessive forward and backward processing, which is a well-known problem for sequential decoding algorithms [14], we limit the number of decoding steps to a given limit. If this maximum number of decoding steps is reached, the decoding process is stopped and no solution is returned.

IV. Application to the MPEG-4 standard

Having established this new algorithm, we propose to study the feasibility and interest of soft VLC decoding for existing video standards such as MPEG-4 [15]. We first describe the simulation chain and algorithm modifications necessary to adapt the existing soft VLC decoding algorithms to the MPEG-4 encoding characteristics.

A. Description of the simulation chain

The simulation chain is depicted in Fig. 2. The video sequence is MPEG-4 encoded, in data partioning mode. Then, the texture partition, which roughly consists of a sequence of VLC codewords, is extracted from the bitstream and corrupted by an additive white Gaussian noise while no noise is added to the other parts of the MPEG-4 bitstream. This selective addition of the noise is justified by the achievement of an unequal protection scheme (UEP) of the bitstream: we assume here that a very efficient channel code is used to protect headers and motion information and thus no error appears on this part, whereas no channel code is used to protect the texture information. The extracted and corrupted bitstream is then decoded using soft decoding algorithms and put back into the non-corrupted bitstream to reconstruct an MPEG-4 frame. Finally, this frame is decoded and erronous packets are concealed.

Note that the existence of an escape mode in the MPEG-4 syntax for texture encoding prevents the direct application of soft decoding algorithms to the extracted texture partition. This has been dealt with by artificially including escape mode codewords in the VLC table, and adapting the algorithms to automatically treat this fixed length code extension.

As mentioned before, the VLC sequence estimation can be done more accurately by using some additionnal *a priori* information on the considered sequence. A classically considered *a priori* information is the number of symbols by sequence [8] [6] but unfortunately this information cannot be extracted from the MPEG-4 frame except by performing its decoding, whereas it is possible to obtain information on the number of

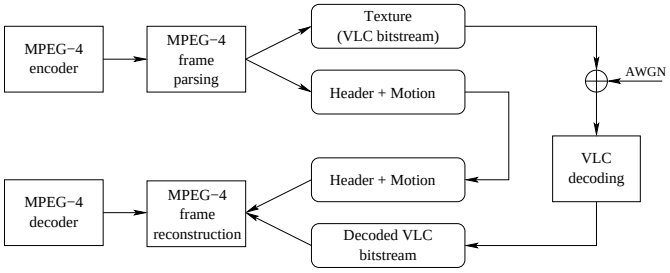


Fig. 2. The chain used in simulating VLC soft decoding of MPEG-4 frames.

blocks contained in the texture partition when decoding headers and motion partition. This information can easily be used by counting in a given sequence of codewords the number of occurrences of a LAST parameter being equal to 1. Thus, the knowledge of the number of blocks by partition is an *a priori* information that can be used as the number of symbols to select a likely sequence, in the sense that it contains the exact number of blocks, and that is available to the user without requiring side information to be transmitted.

Finally, soft VLC decoding algorithms use also the *a priori* knowledge of the probabilities of occurrence of codewords. In practice, these probabilities depend on the video sequence and thus cannot be precisely known by the decoder. In the simulations presented below, we have assumed that the probability of occurrence of a codeword is directly linked to its length, *i.e.* $P(C_i) = (1/2)^{l_i}$.

B. Simulation results

Simulations have been achieved on the so-called "Foreman" sequence encoded with the following parameters: 300 images, CIF resolution, 800 kbit/s, 25 frames/s, one intra image every 12 images and video packets of size 4000 bits. We have compared the proposed sequential algorithm to the *hard* decoding algorithm and to the approximate *bit-trellis* based decoding algorithm proposed in [6]. It has been shown in [6] that this last algorithm outperforms the algorithm of similar complexity presented in [4], and almost reaches the optimal performance.

Figures 3 and 4 give simulation results respectively in terms of Frame Error rate (FER) and Peak Signal to Noise Ratio (PSNR) as a function of the signal to noise ratio over the channel (Eb/N0). The gain achieved though the use of soft values over hard decision in terms of FER is quite small. For a FER of 10^{-1}, we obtain a gain of about 0.4 dB. This corresponds, in terms of PSNR which is the most important criterion, to a very significant gain: for an Eb/N0 of 8 dB we obtain a gain in PSNR of about 8 dB.

Figure 6 illustrates this point presenting the 1^{st} image of the Foreman sequence after respectively a hard and a soft decoding for signal to noise ratio equal to 8 dB. One can see that the visual aspect of the image is clearly better when soft decoding is used.

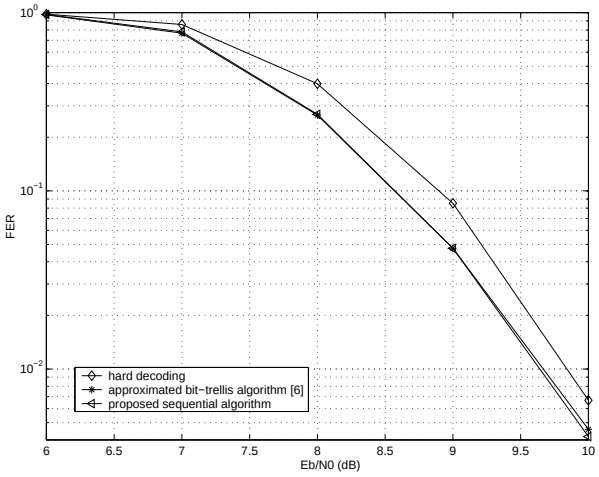


Fig. 3. Frame Error Rate vs. channel noise performance.

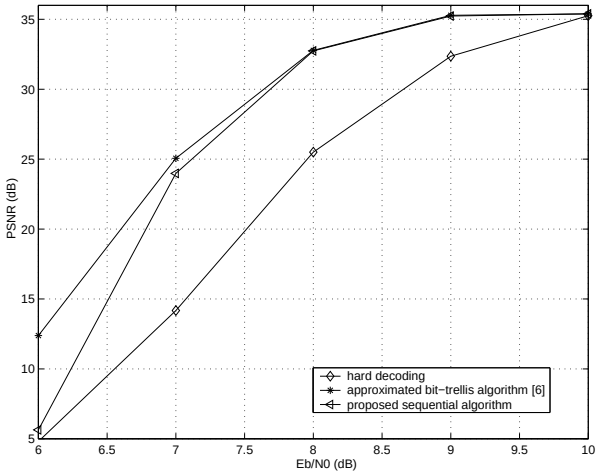


Fig. 4. PSNR vs. channel noise performance.

Finally, Figure 5 summarises the result of a complexity study in terms of number of operations by second as a function of the number of bits by packet. For each algorithm, a complexity region is defined, two curves giving the lower and the upper bound of number of operations. This figure shows that in the MPEG-4 context the proposed modified VLC-stack algorithm is approximatively 10 times less costly than the existing VLC-stack.

V. CONCLUSIONS

We have considered the problem of a soft VLC decoding based on sequential sequence estimation techniques. We have described an improved version of the stack algorithm for VLC decoding. This algorithm achieves approximately the same performance as trellis based algorithms but it is significantly less complex than both trellis based and existing stack algorithms.

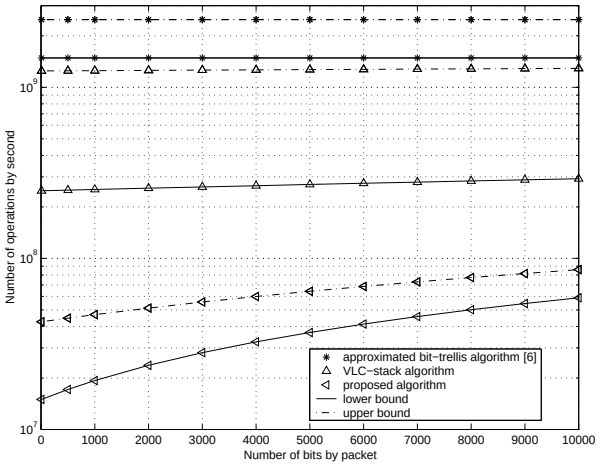
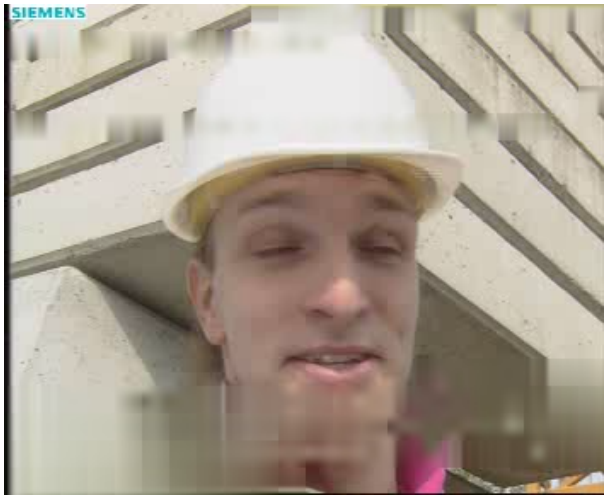
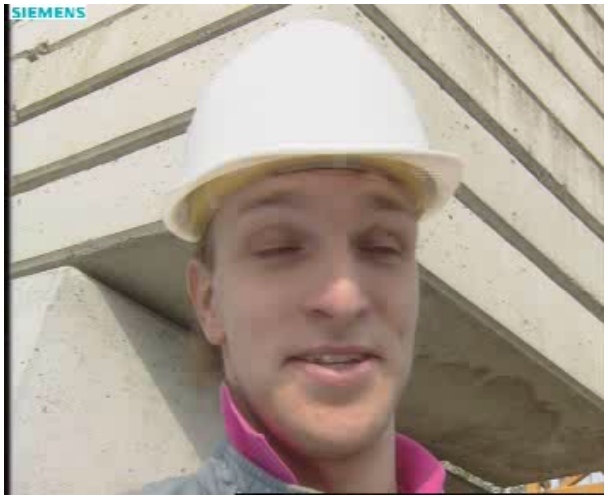


Fig. 5. Comparison of algorithms complexity.

Then, we have investigated the possibility of applying these techniques to improve the decoding of MPEG-4 frames. Simulations show that the gain of soft algorithms compared to the classical hard decoding method is of about 8 dB in PSNR.



(a) 1^{st} image - Hard decoding



(b) 1^{st} image - Soft decoding

Fig. 6. Visual comparison results at Eb/N0=8dB.

REFERENCES

[1] N. Demir and K. Sayood, "Joint source/channel coding for variable length codes," in *Proceedings of the Data Compression Conference*, pp. 139–148, Snowbird, USA, March-April 1998.

[2] M. Park and D. J. Miller, "Decoding entropy-coded symbols over noisy channels by MAP sequence estimation for asynchronous HMMs," in *Proceedings of the Conference on Information Sciences and Systems*, pp. 477–482, Princeton, USA, March 1998.

[3] A. H. Murad and T. E. Fuja, "Robust transmission of variable-length encoded sources," in *Proceedings of the Wireless Communications and Networking Conference*, vol. 2, pp. 968–972, New Orleans, USA, Sept. 1999.

[4] M. Park and D. J. Miller, "Joint source-channel decoding for variable-length encoded data by exact and approximate MAP sequence estimation," *IEEE Transactions on Communications*, vol. 48(1), pp. 1–6, Jan. 2000.

[5] S. Kaiser and M. Bystrom, "Soft decoding of variable-length codes," in *Proceedings of the IEEE International Conference on Communications*, vol. 3, pp. 1203–1207, New Orleans, USA, June 2000.

[6] C. Lamy and O. Pothier, "Reduced complexity Maximum A Posteriori decoding of variable-length codes," in *Proceedings of the IEEE Global Telecommunications Conference*, vol. 2, pp. 1410–1413, San Antonio, USA, Nov. 2001.

[7] L. Guivarch, J.-C. Carlach, and P. Siohan, "Joint source-channel soft decoding of Huffman codes with turbo-codes," in *Proceedings of the Data Compression Conference*, pp. 83–91, Snowbird, USA, March 2000.

[8] J. Wen and J. D. Villasenor, "Utilizing soft information in decoding of variable length codes," in *Proceedings of the Data Compression Conference*, pp. 131–139, Snowbird, USA, March 1999.

[9] V. Buttigieg, *Variable-length error-correcting codes*, PhD thesis, University of Manchester, Manchester, United Kingdom, 1995.

[10] R. M. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Transactions on Information Theory*, vol. 64(9), pp. 64–73, April 1963.

[11] J. L. Massey, "Variable-length codes and the Fano metric," *IEEE Transactions on Information Theory*, vol. 18(1), pp. 196–198, Jan. 1972.

[12] F. Jelinek, "A fast sequential decoding algorithm using a stack," *IBM Journal Reseach and Development*, vol. 13, pp. 675–685, Nov. 1969.

[13] K. Zigangirov, "Some sequential decoding procedures," *Probl. Peredachi Inf.*, vol. 2, pp. 13–25, 1966.

[14] S. Lin and D. J. Costello, *Error control coding: fundamentals and applications*, Prentice-Hall, Englewood Cliffs, 1983.

[15] Rob Koenen, "Overview of the MPEG-4 Standard," Final Maui version ISO/IEC JTC1/SC29/WG11 N3156, International Organization for Standardization, http://drogo.cselt.stet.it/mpeg/standards/mpeg-4/mpeg-4.htm, March 2000.