OPTIMAL VLC SEQUENCE DECODING EXPLOITING ADDITIONAL VIDEO STREAM PROPERTIES

Hang NGUYEN, Pierre DUHAMEL, Jérôme BROUET, Denis ROUFFET
Alcatel, Research and Invovation, Route de Nozay, F91460, FRANCE (hang.nguyen@alcatel.fr, jerome.brouet@alcatel.fr, denis.rouffet@alcatel.fr)
CNRS/LSS, Supelec, Plateau de Moulon, F92190, FRANCE (pierre.duhamel@lss.supelec.fr)

***Abstract*—**Joint source–channel decoding of Variable Length Codes (VLC) for image and video streaming transmission over unreliable links, such as wireless networks, is a subject of increasing interest. This paper proposes an optimum Maximum Likelihood (ML) decoder of VLC sequences which exploits additional inherent redundancy in the source information, namely (i) the correlation between bits inside a VLC codeword as well as (ii) the correlation between VLC codewords of a VLC sequence unit (e.g. corresponding to one image block). Performance results for improving video decoding over AWGN channels are then presented and compared to the prefix-based decoder as well as the recently proposed VLC decoders.

I. INTRODUCTION

Most image and video compression standards [1] make heavy use of variable length codes (VLC) which are known to be very sensitive to errors, specially when the decoding is based on the prefix-properties. This motivated the search for new efficient ways of decoding VLC encoded data. The VLC decoding methods proposed in [2-6] are based on the projection of the received sequences on the codewords dictionary, which corresponds to exploiting the relationship between bits inside a VLC codeword (e.g. due to the prefix property). However, we have shown in a previous paper [7] that additional redundancy is also available in the bit stream by exploiting the relationship between the successive VLC codewords of the same sequence. As a result, methods based only on the structure of the VLC, may provide a non-meaningful estimated sequence even if each decoded codeword is valid in the codebook. This paper proposes an algorithm able to take both redundancies into account. In [2-3,5-6], the underlined decoding trellis is based on the nodes taken from the VLC codewords tree. A straightforward modification of these algorithms could hardly take into account the constraints on the whole sequence issued from the source. This paper first proposes a new algorithm (in a list-Viterbi manner) able to take both constraints into account. Then, the gain of the algorithm is shown on real video sequences.

II. SOURCE REDUNDANCY IN VLC SEQUENCE OF IMAGE BLOCK

This section summarizes results from [7], in order to provide motivation for this algorithm, as well as notations used for describing the method. In most image and video compression standards [1], the pixels of each image block are processed by a Discrete Cosine Transform (DCT). The number of DCT coefficients is equal to the number of pixels of the image block. These DCT coefficients are then encoded by a run-length VLC. In the H.263 standard [1], each VLC codeword corresponds to a triplet (*run*, *level*, *last*): "*run*" represents the number of zeros till the next non-zero DCT coefficient, "*level*" represents the value of this non-zero DCT coefficient, and "*last*" indicates if this DCT coefficient is the last non-zero value of the image block. So, each VLC codeword corresponds to ("*run*" + 1) DCT coefficients ("*run*" zero DCT coefficients and one non-zero coefficient). In the H.264 standard [1], codewords are either the couple (*run*, *level*) or the *end of block* indicator *EOB*. Under these assumptions, the following properties are always true:

Property 1: the sum over all the number of DCT coefficients corresponding to the VLC codewords of one compressed image block should be less or equal to $N_{DCTcoef}$ (the total number of DCT coefficients of one image block) [1]. So, the "*run*" values of the VLC codewords of one compressed image block meet the following constraint:

$$\sum_{VLC\ codewords\ of\ one\ image\ block} (run_{vlc} + 1) \le N_{DCTcoef} \qquad (1)$$

Property 2: only the last codeword of the sequence corresponding to one image block has the field "*last*" of the triplet (*run*, *level*, *last*) equals to one [1].

In the following, sequences meeting these constraints are denoted as ***feasible sequences***. Our algorithm allows these properties to be used in order to improve the decoding of the bit stream. For H.263, both conditions hold. For H.264, the first condition is still valid, and the equivalent of the second condition is that the last VLC codeword in a block should correspond to the EOB. In [7], an evaluation of the source redundancy resulting from those properties was evaluated. Two quantities were estimated: the redundancy due to correlation between bits inside a VLC codeword, and, the redundancy due to both types of correlation: between bits of a codeword and between codewords of a sequence. As an example, for a typical value of 60 bits of one compressed image block using H.263, the "number of equivalent redundancy bits" is about 11 bits when both types of correlation are taken into account (i.e. VLC structure and constraints on the whole sequence due to the source semantics). In comparison, it is only about 2 bits when considering only the VLC structure [7].

III. THE PROPOSED VLC DECODER

A. Problem formulation

A VLC codeword sequence of N bits corresponding to one unit of data (e.g. a compressed image block) to be decoded is received. N is assumed to be known. The number of VLC codewords and the corresponding VLC codewords are to be estimated. Besides, errors may have occurred during the transmission, so the received N bits may correspond neither to a Variable Length Code (VLC) sequence nor to a meaningful image block. Hence, the aim is to find out the transmitted Variable Length Code (VLC) sequence which maximizes the likelihood of the estimated sequence.

Let Y be the received sequence of N bits, S the set of all feasible sequences of feasible VLC codewords, and X an element of the set S. If the distribution of the source is uniform, the optimum hard-output sequence in the ML sense is defined by:

$$X_{estimated} = \arg \underset{X \in S}{Max}\ P(Y/X) = \arg \underset{X \in S}{Max}\ P(X/Y) \quad (2)$$

The set S contains all the feasible sequences of VLC codewords of N bits, meeting all constraints of an image block ("*last*"&"*run*" constraints). To get the optimum hard-output solution, all feasible sequences of the set S have to be considered. The trivial enumeration of the set S cannot be used practically because its complexity increases exponentially with N.

B. Basic idea of the new VLC decoder

The proposed decoding algorithm benefits from all the constraints that can be expressed on the whole sequence. It uses the notion of "survivor sequence" as in the Viterbi algorithm (VA) [7], and applies it to variable length code decoding. The survivor selection takes into account the conventional Viterbi metric [7], the VLC structure and the source semantics constraints on the sequence. It is similar to a list-Viterbi [9] and to the VLC decoding method of [4], but with additional sequence constraints included in the decoding process and a different selection of survivors. However, the straightforward application of VA meets two major difficulties: the variable length structure of the code and the additional sequence constraints. The first difficulty is resolved by the recursive construction of the lists, in which, the candidate sequences are classified by their lengths in bits. Only candidate sequences of same length in bits are considered for the survivor selection. Hence, there are one or more survivors for each possible length in bits. The second difficulty is resolved by the additional steps (2.4, 2.5 and 2.6 of the proposed algorithm described in section III-D) related to the number of DCT coefficients r associated to the candidate sequences (refer to section III-C for the definition of r). Only candidate sequences of same length in bits and same associated value r are considered for the survivor selection. There is one survivor for each possible value of r. Hence, for each possible length in bits, there are several survivors corresponding to different possible values of r. More precision is provided in sections III-C, III-D and IV. This new decoder can deliver optimum hard-output values in the maximum likelihood (ML) sense, with linear complexity with the number of VLC codewords. The complexity and processing costs are similar to those of the other existing methods for decoding VLC sequences, which only make use of a partial redundancy. Another advantage of the method is that the decoded sequence is always a feasible VLC sequence of a valid image block.

C. Notations

Sequences of exactly N bits are called ***complete sequences***. Sequences of length strictly smaller than N bits are called ***incomplete sequences***. We note:

- C: the VLC codebook.
- N: the length in bits of the received VLC sequence to be decoded, corresponding to one image block.
- k: the estimated number of estimated VLC codewords in the incomplete sequence. k is progressively incremented up to the maximum number of codewords that a feasible VLC sequence can have.
- L_k : the set of list containing *feasible incomplete* VLC sequences of exactly k VLC codewords.
- F: the list containing the *feasible complete* survivor VLC sequences.
- For any binary sequence S, we define the function denoted ζ such as $\zeta(S)$ takes the value of the length in bits of the sequence S.
- For any VLC sequence S, we define the function denoted τ such as $\tau(S)$ takes the value "1" if the tests of Fig.1 applied on the sequence S are successful.

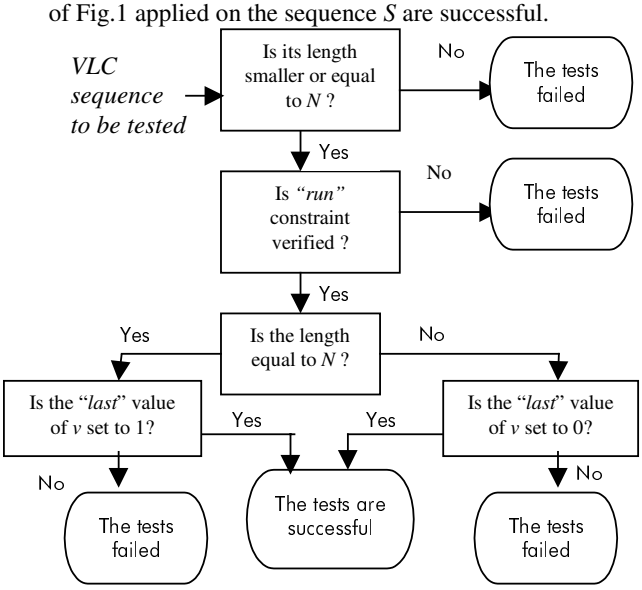

Fig. 1 : Tests on a candidate VLC sequence

- For any VLC sequence S, we define the function denoted ρ such as $\rho(S) = r_S = \sum_{vlc \in S} (run_{vlc} + 1)$

  ρ (S) (or r_S) is denoted as ***the number of DCT coefficients associated to the sequence*** S**.**
- S_k : an element of the list L_k .
- Length of the longest/shortest codeword of the VLC codebook: $\overline{\lambda}(C) = \max_{v \in C}\{\zeta(v)\}$ and $\underline{\lambda}(C) = \min_{v \in C}\{\zeta(v)\}$
- Length of the longest/shortest sequence of the list L_k : $\overline{\lambda}(L_k) = \max_{S_k \in L_k}\{\zeta(S_k)\}$ and $\underline{\lambda}(L_k) = \min_{S_k \in L_k}\{\zeta(S_k)\}$

D. Algorithm description

The algorithm determines the ML VLC sequence of N bits verifying both constraints "*run*"&"*last*". For that, a number of lists of sequences L_k and F are computed recursively.

1) Initialization

The list L_0 and the list F are initialized as empty lists.

2) Recursion

The recursion is on k which is increased by 1 at each step. At each recursion, L_{k-1} is available, and L_k is initialized to an empty list. The list L_k containing feasible incomplete VLC sequences of k codewords is to be constructed at the end of the recursion.

1. Determine the range of possible lengths of sequences of the list L_k by computing $\lambda_{\max}$ et $\lambda_{\min}$:

$\lambda_{\max} = \min\left\{N, \overline{\lambda}(L_{k-1}) + \overline{\lambda}(C)\right\}$; $\lambda_{\min} = \underline{\lambda}(L_{k-1}) + \underline{\lambda}(C)$

2. For λ varying from $\lambda_{\min}$ to $\lambda_{\max}$:

2.1 Compute the set Δ_λ of sequences of length λ, and being the concatenation of one sequence from L_{k-1} and one codeword from the VLC codebook.

$$\Delta_\lambda = \left\{S_j \middle| \zeta(S_j) = \lambda \text{ and } \left(\exists S_{k-1} \in L_{k-1}, \exists v \in C : S_j = S_{k-1}v\right)\right\}$$

Δ_λ may be empty. In this case, the algorithm goes on with the following possible values of λ.

2.2 Compute the subset Δ'_λ of Δ_λ of sequences that verify both constraints "*last*"&"*run*" and the length condition: $\Delta'_\lambda = \left\{S_j \middle| S_j \in \Delta_\lambda \text{ and } \tau(S_j) = 1\right\}$.

2.3 Identify the maximum-likelihood (ML) sequence $S^{ML}_{\Delta'_\lambda}$ associated to the Viterbi metrics [8] over the set Δ'_λ, then compute the number of DCT coefficients associated to $S^{ML}_{\Delta'_\lambda}$: $r_{S^{ML}_{\Delta'_\lambda}} = \rho(S^{ML}_{\Delta'_\lambda})$.

However, due to the "*run*" constraint, saving only the sequence $S^{ML}_{\Delta'_\lambda}$ for the next recursion loops is not sufficient. Indeed, at the next recursion loops, the sequences formed from the sequence $S^{ML}_{\Delta'_\lambda}$ may be discarded because they don't verify the "*run*" constraint (even if $S^{ML}_{\Delta'_\lambda}$ does). In this case, if, in addition, the subset of feasible sequences corresponding to $S^{ML}_{\Delta'_\lambda}$ were not empty, this subset would be discarded erroneously. Hence, saving only the sequence $S^{ML}_{\Delta'_\lambda}$ would give a sub-optimal algorithm. More explanation is given in section IV. Therefore, for the optimality of the algorithm, the steps 2.4, 2.5 and 2.6 have been added as follows. More survivor sequences are saved for each length λ.

2.4 Compute the subset Δ''_λ of Δ'_λ of sequences whose associated numbers of DCT coefficients are smaller than the one of the ML sequence:

$$\Delta''_\lambda = \left\{S'_j \middle| S'_j \in \Delta'_\lambda \text{ and } \rho(S'_j) \le r_{S^{ML}_{\Delta'_\lambda}}\right\}.$$

2.5 Organize the sequences of Δ''_λ by the values of their numbers of DCT coefficients *r*.

We note $R_{\Delta''_\lambda}$ the set of possible values of the number of DCT coefficients associated to all the sequences of the set Δ''_λ: $R_{\Delta''_\lambda} = \left\{r \middle| \exists S''_j \in \Delta''_\lambda \text{ and } \rho(S''_j) = r\right\}$.

Subdivide the set Δ''_λ into subsets $\Delta''_{\lambda,r}$ according to the different values of $r \in R_{\Delta''_\lambda}$.

$$\Delta''_{\lambda,r} = \left\{S''_j \middle| S''_j \in \Delta''_\lambda \text{ and } \rho(S''_j) = r\right\}.$$

$\Delta''_{\lambda,r}$ is the subset of Δ''_λ of sequences whose associated numbers of DCT coefficients are equal to *r*.

To summarize, $\Delta''_{\lambda,r}$ is the set of sequences: (i) of length λ, (ii) being the concatenation of one sequence from L_{k-1} and one codeword from the VLC codebook, (iii) verifying both constraints "last"&"run" and the length condition, (iv) whose associated numbers of DCT coefficients are equal to *r* which is smaller than the one of the ML sequence $S^{ML}_{\Delta'_\lambda}$.

$\left\{\Delta''_{\lambda,r}\right\}_{r \in R_{\Delta''_\lambda}}$ is a partition of Δ''_λ.

2.6 Identify and save the maximum-likelihood sequence $S^{ML}_{\Delta''_{\lambda,r}}$ associated to the Viterbi metrics [8] over each subset $\Delta''_{\lambda,r}$.

At this step, a list of survivor sequences for all the possible associated values *r* of $R_{\Delta''_\lambda}$, containing exactly *k* VLC codewords, and of length λ, is obtained.

2.7 If $\lambda = N$, these survivors are stored in *F*.

If $\lambda < N$, these survivors are stored in L_k.

Finally, the list L_k is obtained. L_k contains all feasible incomplete VLC survivor sequences of *k* codewords, with a bit length strictly smaller than *N*, and meeting both VLC structure constraints and source-induced constraints on the whole sequence.

3) Decision step

The recursion is finished when L_k is empty with $k \ge 1$.

The proposed recursive algorithm will always stop because all VLC codeword lengths are greater or equal to one bit, so the number of VLC codewords in an *N*-bit VLC codeword sequence cannot be greater than *N*.

From the obtained list *F*, the optimal hard-output sequence is the survivor sequence with the smallest metric [8] among all sequences in the list *F*.

IV. OPTIMALITY OF THE HARD-OUTPUT DECODER

This section outlines the proof that the algorithm described above provides the optimum ML estimate of the *N* bits sequence of feasible VLC codewords. The optimality notion here is defined as the maximum likelihood solution of equation (2) when the whole set *S* is considered.

Viterbi algorithm and the VLC decoding methods previously proposed in [7-11] are based on the dynamic programming principle: if the optimal last codeword of the incomplete sequence *S* to be decoded is V_{k+1}, the best choice at iteration *k*+1 is of the form $S_{opt} = S'_{opt}V_{k+1}$ with S'_{opt}, the optimal choice at iteration *k*. If only the "*last*" constraint is used, this dynamic programming principle guarantees the optimality of the proposed algorithm. However, in the proposed algorithm described in section III-D, by adding the "*run*" constraint, this dynamic programming principle does not hold anymore: if the optimal choice for the sequence *S* to be decoded is $S_{opt} = S'V_{k+1}$, *S'* may be different from S'_{opt} that is the optimal choice at iteration *k*. This difference is due to the

following property: even if the "*run*" condition holds for S'_{opt} and V_{k+1}, it may not hold for $S_{new} = S'_{opt} V_{k+1}$. Indeed, the following case may happen:

$R_{S'_{opt}} = \sum_{i=1}^{k} (run_i + 1) \le N$ and $R_{V_{k+1}} = (run_{k+1} + 1) \le N$,

but $R_{S_{new} = S'_{opt} V_{k+1}} = \sum_{i=1}^{k+1} (run_i + 1) > N$. As a result, if $S_{new} = S'_{opt} V_{k+1}$ did not verify the "*run*" condition, S_{new} would be discarded, even if another choice for S'_{opt} would meet this constraint. Some sequences would have been discarded erroneously. Hence, to solve this problem, **the steps 2.3, 2.4 and 2.5** related to *r*, the associated numbers of DCT coefficients of the sequences of Δ''_{λ}, ***have been added in section III-D*** in the proposed algorithm. The proposed algorithm would be much simpler otherwise. The basic idea is to consider only candidate sequences of same length in bits and same associated value *r* for the survivor selection. These additional steps force the storage of the sequences that would have been erroneously discarded, in the list L_k, in order to be sure that they will be considered during the next iteration. Hence, the proposed algorithm in section III-D is optimum for both "*last*" and "*run*" constraints.

V. Simulation Results

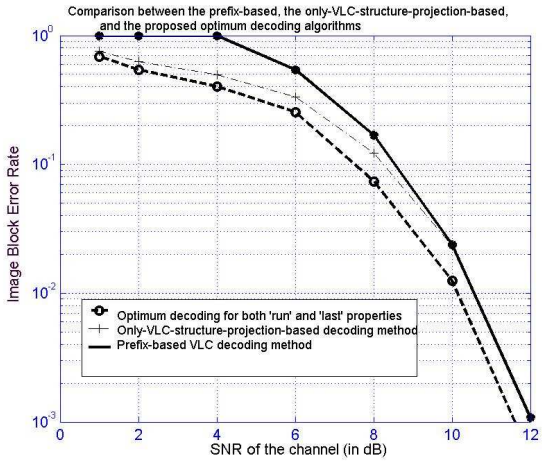

Fig. 2*:* Comparison between the prefix-based, the only-VLC-structure-projection-based and the optimum proposed decoding methods

In Fig.2, the proposed optimum decoder using the VLC structure projection and intrinsic image properties has been evaluated and compared with the conventional prefix-based VLC decoder, and the existing decoder using only the projection on the VLC structure[2-6]. The performance metric is chosen as the image block error rate, defined as follows:

$$IBER = \frac{number\ of\ blocks\ that\ are\ erroneously\ decoded}{number\ of\ transmitted\ blocks}$$

which is more significant in terms of application, since a single bit in error can result in a loss of the whole image block. Our intent here is only to show how, by using constraints due to the source, the improvement at the decoder translates to a lower number of image blocks in errors. A very simple BPSK modulation transmission chain over Gaussian channel has been simulated. A set of image blocks from three classical video sequences ("Mother-daughter", "Foreman", "Irene") is used for simulation. The video encoder under consideration is H.263. This set of image blocks are transmitted over the Gaussian (AGWN) channel, and then decoded with the algorithms to be compared. The input of the VLC sequence decoder are soft values. Only source encoder and decoder are used in the simulation chain. No channel coding is used. The conventional prefix-based VLC decoder is only a source decoder: it can not correct the transmission errors in the bit stream. The new VLC decoder is also a source decoder, but it has in addition a channel decoding behavior: it can correct some of the transmission errors in the bit stream. Another advantage of this new decoder is that it always provides a feasible VLC sequence corresponding to a valid image block. Fig. 2 depicts the image block error rate as a function of the SNR of the channel. A gain of 0.5 to 2 dB in terms of IBER is obtained compared to the prefix decoding. Our algorithm outperforms both the prefix based decoder and the VLC-based decoder on the whole range of SNR. The complexity of this algorithm can be experimentally evaluated by storing the number of sequences in each list L_k. Our simulation shows that each list L_k contains on average 20 sequences. Video sequences will be played at the conference, demonstrating that the proposed algorithm increases the SNR range in which video sequences can be reliably streamed.

VI. Conclusion

This paper proposes a new decoding process for VLC sequences which exploits both the VLC structure and the intrinsic structure of feasible VLC sequences of image blocks. This method improves significantly the source decoding performance by diminishing the number of erroneously decoded image blocks. Further work is currently being undertaken to evaluate this scheme in real situation and by incorporating channel coders.

References

[1] *ITU-T Recommendation H.263 :* 03-96, *H263+ :* 02-98, *H264*.

[2] R.Bauer, J. Hagenauer, " On Variable Length Codes for Iterative Source-Channel Decoding", *Proc. of IEEE DCC*, 2001.

[3] Bauer, Hagenauer,"Iterative Source-Channel Decoding based on a Trellis representation for Variable Length Codes",*ISIT* 2000.

[4] J.Wen, J.D.Villasenor, " Utilizing soft information in decoding variable length codes", *Proc. IEEE DCC*, March 1999.

[5] S.Kaiser, M.Bystrom, "Soft decoding of variable length codes", IEEE ICC, 2000, volume 3.

[6] L.Guivarch, J.C. Carlach, P. Siohan, "Joint source-channel soft decoding of Huffman codes with Turbo-codes*"*, *DCC* 2000.

[7] H. Nguyen, P. Duhamel, "Estimation of Redundancy in Compressed Image and Video Data for Joint Source-Channel Decoding", *IEEE Globecom,* Dec 2003.

[8] Forney, "The Viterbi Algorithm*"*, *Proc. of the IEEE*, 03/1973.

[9] Hashimoto, "A list-type reduced-constraint generalization of the Viterbi algorithm*"*, *IEEE Trans. On Inf. Theory*, 11/1987

[10] Hoeher, "Advances in Soft-Output Decoding", *IEEE GLOBECOM*, vol.2, Page(s): 793 –797, 29 Nov-2 Dec 1993.