# ERROR-RESILIENT BINARY MULTIPLEXED SOURCE CODES

*Hervé* JÉGOU

IRISA/ENS Cachan, France

*Christine Guillemot*

IRISA/INRIA, France

## ABSTRACT

This paper addresses the issue of robust transmission of VLC encoded sources over error-prone channels. A new family of codes, called multiplexed codes, has been recently introduced in [1]. They exploit the fact that compression systems of real signals generate sources of information with different levels of priority. Multiplexed codes allow to confine the de-synchronization phenomenon to low priority data while allowing to reach asymptotically the entropy bound for both (low and high priority) sources. A multiplexing procedure based on an iterative Euclidian decomposition has been proposed. This paper introduces a variant of multiplexed codes, called binary multiplexed codes, together with a very simple multiplexing algorithm that exploits the structure of variable length codetrees. It is shown analytically and experimentally that this family of codes is more error resilient than fixed length codes while reaching the compression efficiency of classical variable length codes.

## 1. INTRODUCTION

Entropy coding, producing variable length codewords (VLC), is a core component of any data compression scheme. The main drawback of VLCs is their high sensitivity to channel noise: when some bits are altered by the channel, the position of symbol boundaries are not properly estimated, leading to dramatic symbol error rates. This phenomenon has motivated the design of codes with better synchronization properties [2, 3, 4]. Reversible VLCs [5, 6], have also been designed to fight against desynchronization. The above solutions add redundancy in the generated bitstream.

Here, we consider a family of codes that has been recently introduced under the name of “multiplexed codes” [1]. Their design principle relies on the fact that compression systems of real signals generate sources of information with different levels of priority (e.g. texture and motion information for a video signal). Two sources, a high priority source and a low priority source referred respectively as $\mathbf{S_H}$ and $\mathbf{S_L}$ in the sequel are hence considered. These codes are such that the risk of “de-synchronization” is confined to the low priority information. The idea consists in creating a fixed length code (FLC) for the $\mathbf{S_H}$ source, in partitionning the set of FLCs in *classes of equivalence* and in exploiting the redundancy inherent of each class to represent or store information of the low priority source $\mathbf{S_L}$. The fixed length codewords will then repesent jointly a symbol of the high priority source and a set of symbols of the low priority source. Different “multiplexing” algorithms may be considered to construct the joint (or multiplexed) codewords that are eventually sent on the channel. These different “multiplexing” algorithms in turn lead to variants of multiplexed codes. A first approach has been described in [1] : it relies on an iterative Euclidian decomposition. Here, we introduce a subset of these codes, that we call “binary multiplexed codes”, based on a partition of the FLC into *classes of equivalence* whose cardinals are integer powers of 2. A very simple “multiplexing” algorithm is then described. Pushing further this idea, we then introduce a family of “binary multiplexed codes” constructed from VLCs : instead of constructing a FLC for the high priority source, a VLC is considered. The structure of the variable length codetree is then exploited for “multiplexing’ the low priority information. A theoretical and experimental analysis of these codes show that they are more resilient than FLCs, while at the same time allowing to almost reach entropy bounds.

## 2. PROBLEM STATEMENT AND NOTATIONS

Let $\mathbf{S}_H = (S_1, ..., S_t, ..., S_{K_H})$ be a sequence of source symbols of high priority taking their values in a finite alphabet $\mathcal{A}$ composed of $\Omega$ symbols, $\mathcal{A} = \{a_1, ...a_i, ...a_\Omega\}$. Note that we reserve capital letters to random variables, and small letters to values of these variables. Bold face characters will be used to denote vectors or sequences. The stationary probability of the source $\mathbf{S}_H$ is denoted $\boldsymbol{\mu} = (\mu_1, ...\mu_i, ...\mu_\Omega)$, where $\mu_i$ stands for the probability that a symbol of $\mathbf{S_H}$ equals $a_i$. Let $h$ be the stationary entropy of the source per symbol, given by $h = -\sum_{i=1}^{\Omega} \mu_i \, log_2(\mu_i)$. Let $\mathbf{S}_L = (S'_1, ...S'_{i'}, ...S'_{K_L})$ be a sequence of source symbols of lower priority taking their values in a finite alphabet $\mathcal{A}'$. We assume that the realization $\mathbf{s_L}$ of this source has been pre-encoded into a bitstream $\mathbf{b} = (b_1, ...b_r, ...b_{K_B})$ with a VLC coder (e.g. Huffman or arithmetic coder). The problem addressed here is the design of a family of joint codes for the two sources $\mathbf{S_H}$ and $\mathbf{S_L}$ that would guarantee no “de-synchronization” of the high priority source $\mathbf{S_H}$ at almost no cost in terms of compression efficiency.

| class $\mathcal{C}_i$ | codeword $c_{i,q}$ | $a_i$ | $N_i$ | $\mu_i$ | index $q$ |
|---|---|---|---|---|---|
| $\mathcal{C}_1$ | 000 | $a_1$ | 2 | 0.30 | 0 |
| | 001 | | | | 1 |
| $\mathcal{C}_2$ | 010 | $a_2$ | 3 | 0.43 | 0 |
| | 011 | | | | 1 |
| | 100 | | | | 2 |
| $\mathcal{C}_3$ | 101 | $a_3$ | 1 | 0.25 | 0 |
| | 110 | | | | 1 |
| $\mathcal{C}_4$ | 111 | $a_4$ | 1 | 0.02 | 0 |

**Table 1**. An example of multiplexed codes ($c = 3$).

## 3. MULTIPLEXED CODES

### 3.1. Principle

Let $c$ be a fixed number of bits reserved for the representation of any symbol of the alphabet $\mathcal{A}$. The $c$ bits define a set of $N = 2^c$ codewords. This set of codewords is partitioned into $\Omega$ subsets, $\mathcal{C}_i, i = 1 \ldots \Omega$, called *equivalence classes* associated to symbols $a_i$ of the alphabet $\mathcal{A}$, as shown in Table 1. The condition $c \geq log_2(\Omega)$ is required to have at least 1 codeword per subset. Each *equivalence class* $\mathcal{C}_i$ contains a set of codewords $\{c_{i,0}, c_{i,1}, \ldots, c_{i,q}, \ldots, c_{i,N_i-1}\}$, where the integer $N_i$ stands for the number of codewords in the subset (cardinality of the *equivalence class*) and is such that $\sum_{i=1}^{\Omega} N_i = N$.

A symbol $S_t = a_i$ of the flow $\mathbf{S}_H$ can be encoded with any $c$-bit codeword $c_{i,q}$ belonging to the *equivalence class* $\mathcal{C}_i$ (see example of Table 1). Hence, each symbol $S_t$ can be mapped into a pair $(\mathcal{C}_i, Q)$ of two variables denoting respectively the *equivalence class* and the index of the codeword in the *equivalence class* $\mathcal{C}_i$. The variable $Q$ is an $N_i$-valued variable, taking its value between 0 and $N_i - 1$ (see Table 1) and representing the inherent redundancy of the $c$-bits fixed length codes. This redundancy can then be exploited to describe the lower priority flow of data. Therefore, to the realization of the sequence of symbols $\mathbf{S}_H$ one can associate a sequence of $N_i$-valued variables $\mathbf{N} = N_1, \ldots N_i, \ldots N_{K_H}$, which will be used to describe jointly the $\mathbf{S}_H$ and $\mathbf{S}_L$ data flows.

A multiplexed code has been defined in [1] as the function which maps a $c$-bits fixed length codeword $c_{i,q}$ into a pair of variables comprising the symbol value $a_i$ of the alphabet $\mathcal{A}$ (on which the high priority source is quantized) and the value $q$ of a variable $Q$ denoting the index of the codeword in the *equivalence class* associated to $a_i$ as:

$$c_{i,q} \leftrightarrows (\ a_i, q\ ) \qquad (1)$$

For example, let $\mathcal{A} = \{a_1, a_2, a_3, a_4\}$ be the alphabet of the source $\mathbf{S}_H$ with the stationary probabilities given by $\mu_1 = 0.30$, $\mu_2 = 0.43$, $\mu_3 = 0.25$, and $\mu_4 = 0.02$. Table 1 gives an example of partitioning of the set of $N = 2^c$ codewords into the 4 *equivalence classes* associated to the alphabet symbols.

### 3.2. Conversion of the lower priority bitstream

It appears from above that the design of multiplexed codes relies on the choice of the partition $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_\Omega\}$. In order to be multiplexed with symbols of $\mathbf{S}_H$, the lower priority bitstream $\mathbf{b}$ must be mapped into a sequence of $N_i$-valued variables. Let us consider the mapping of the realization $\mathbf{s}_H = s_1 \ldots s_t \ldots s_{K_H}$ into the sequence $\mathbf{n} = n_1 \ldots n_t \ldots n_{K_H}$. The sequence of $n_t$-valued variables can be seen as a unique $\Lambda$-valued variable $\theta$, where $\Lambda = \prod_{t=1}^{K_H} n_t$. The variable $\theta$ hence verifies $0 \leq \theta \leq \Lambda - 1$. The maximum amount of information that can be stored in the $\Lambda$-valued variable $\theta$ is theoretically $log_2(\Lambda)$ bits. It is shown in [1] that the mean description length (mdl) of $\mathbf{S}_H$ can be made as close as required to the stationnary entropy of the source $\mathbf{S}_H$, by increasing the length $c$ of the codewords.

The quantity $\Lambda$ denotes the number of different multiplexed sequences $\mathbf{m}$ of codewords $c_{i,q}$ that can be used as a coded representation of the sequence $\mathbf{s}_H$. One of these sequences $\mathbf{m}$ can be used as a multiplexed coded description of the two sequences $\mathbf{s}_H$ and $\mathbf{s}_L$ [1]. It depends on the bitstream representation $\mathbf{b}$ of the source $\mathbf{S}_L$.

There are several ways to convert the low priority bitstream into a sequence of $n_t$-valued variables $(q_t)$, where $t = 1, \ldots K_H$. The sequence of pairs $(n_t, q_t)$ provides entries in the multiplexed code table. In [1], two methods have been introduced for this conversion:

- The first approach regards the low priority bitstream as an integer which is then decomposed using an iterative Euclidean decomposition. This approach allows to reach the entropy bound.
- A second approach relies on a constrained partition of the FLCs into *equivalence classes* with cardinals belonging to a subset of integer values. This approach still leads to a good approximation of the high priority source distribution, hence is quasi-optimal.

## 4. BINARY MULTIPLEXED CODES

The conversion of the bitstream $\mathbf{b}$ of lower priority into a sequence of states is the most time consuming step of the coding procedure. This section introduces subsets of multiplexed codes that allow to avoid this conversion.

### 4.1. Constrained partition of FLCs

The partition of the FLC into several equivalence classes can be chosen such that the storage capacity is an integer number of bits. It means that, for each equivalence class $\mathcal{C}_i$, a bijection exists between the interval of indexes $[0, N_i - 1]$ and a varying integer number of bits, hence the following definition:

[1] In fact, only a part of $\mathbf{s_L}$, whose length depends on the multiplexing capacity of the sequence $\mathbf{s_H}$

| class $\mathcal{C}_i$ | $c_{i,q}$ | $a_i$ | $N_i = 2^{D_i}$ | $\mu_i$ | $q$ | $\overline{U_i}$ |
|---|---|---|---|---|---|---|
| $\mathcal{C}_1$ | 000 | $a_1$ | 2 | 0.30 | 0 | 0 |
| | 001 | | | | 1 | 1 |
| $\mathcal{C}_2$ | 010 | $a_2$ | 4 | 0.43 | 0 | 00 |
| | 011 | | | | 1 | 01 |
| | 100 | | | | 2 | 10 |
| | 101 | | | | 3 | 11 |
| $\mathcal{C}_3$ | 110 | $a_3$ | 1 | 0.25 | 0 | $\varnothing$ |
| $\mathcal{C}_4$ | 111 | $a_4$ | 1 | 0.02 | 0 | $\varnothing$ |

**Table 2**. A binary Multiplexed codes ($c = 4$).

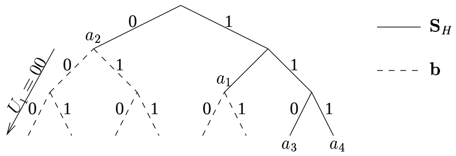


**Fig. 1**. Codetree describing both $\mathbf{S}_H$ and $\mathbf{B}$

**Definition 1:** A binary multiplexed code is a multiplexed code such that $\forall i \in [1..\Omega], D_i = log_2(N_i) \in \mathbb{N}$.

Then, to each symbol $a_i$ of the source $\mathbf{S}_H$, one can associate, a class of equivalence $\mathcal{C}_i$, hence a set of fixed length codewords, which can in turn be indexed by a sequence $\overline{U_i} = U_i^1 \dots U_i^j \dots U_i^{D_i}$ of $D_i$ bits. A codeword representing jointly a symbol of the source $\mathbf{S}_H$ and a segment of the low priority bitstream $\mathbf{b}$ can thus be selected without requiring any conversion of the bitstream $\mathbf{b}$.

*Example 1:* Let $\mathbf{s}_H = a_1 a_2 a_2 a_3 a_2 a_1 a_2 a_4$ be the source of high priority and $\mathbf{b}$ =0101010101 a pre-encoded bitstream. The number of bits that can be multiplexed with each symbol realization of the sequence $\mathbf{S}$ is given by $d_i = (1, 2, 2, 0, 2, 1, 2, 0)$. This leads to segment $\mathbf{b}$ into $(\overline{u_1}, \dots \overline{u_t}, \dots \overline{u_{K_H}}) = (0, 10, 10, \varnothing, 10, 1, 01, \varnothing)$. Then, for each high priority symbol index $t$, the couple $(a_t, \overline{u_t})$ indexes a codeword in the multiplexed table. The resulting multiplexed bitstream is 000 100 100 110 100 001 011 111.

### 4.2. Binary multiplexed codes construction from VLCs

Instead of designing a FLC codebook for the high priority source $\mathbf{S}_H$, one can alternately consider a variable length codetree. This VLC codetree can then be completed to form a binary FLC codetree (see Fig. 4.2). The symbols of the alphabet of the source $\mathbf{S}_H$ will then be associated either to leaves of the FLC or to intermediate nodes. When they correspond to a node, all the corresponding leaves will constitute the *class of equivalence* associated to the given symbol of $\mathbf{S}_H$. The cardinals of the *classes of equivalence* are the integer powers of 2. Hence, one can create a multiplexed code such that codewords of the same equivalence class have the same prefix. The suffix $\overline{U_i}$ of the codewords can then be used to "store" the bits of the bitstream $\mathbf{b}$ (see Fig. 4.2).

**Definition 2:** A Binary multiplexed code derived from a VLC is a multiplexed code constructed such as (1) every prefix of a codeword is the representation of a symbol in a VLC tree. (2) every suffix corresponds to the multiplexed data of the low priority bitstream.

*Example 2:* Considering the same sequence as in example 1, the VLC-based multiplexed code depicted in Fig. 4.2 leads to the multiplexed bitstream $\mathbf{m}$ below.

| $s_t$ | $a_1$ | $a_2$ | $a_2$ | $a_3$ | $a_2$ | $a_1$ | $a_2$ | $a_4$ |
|---|---|---|---|---|---|---|---|---|
| prefix | 10. | 0.. | 0.. | 110 | 0.. | 10. | 0.. | 111 |
| $\overline{u_t}$ | ..0 | .10 | .10 | ... | .10 | ..1 | .01 | ... |
| $m_t$ | 100 | 010 | 010 | 110 | 010 | 101 | 001 | 111 |

The compression efficiency for the source $\mathbf{S}_H$ is obviously given by the efficiency of the VLC considered. Similarly, since the low priority source before multiplexing has been pre-encoded, the corresponding compression efficiency depends on the VLC code considered for this source. The low priority source (e.g high frequencies of a wavelet decomposition) can be pre-encoded with an arithmetic coder.

## 5. PERFORMANCE ANALYSIS

In this section, we analyze the impact of channel errors (considering a binary symmetric chanel, BSC) on the distortion of the source $\mathbf{S_H}$. Let us consider first general multiplexed codes. It has been shown in [1] that the mean square error $MSE_{mul,S_H}$ of the reconstructed $\mathbf{S_H}$ stream in presence of bit errors can be expressed as

$$\sum_{a_i \in \mathcal{A}} \mu_i \sum_{a_{i'} \in \mathcal{A}} (a_i - a_{i'})^2 \underbrace{\frac{1}{N_i} \sum_{c_{i,j} \in \mathcal{C}_i} \sum_{c_{i',j'} \in \mathcal{C}_{i'}} P(c_{i',j'}/c_{i,j})}_{P(a_{i'}/a_i)}, \quad (2)$$

where $P(c_{i',j'}/c_{i,j})$ stands for the probability to receive the codeword $c_{i',j'}$, if $c_{i,j}$ has been transmitted, and $P(a_{i'}/a_i)$ is the probability to decode the symbol $a_{i'}$ if $a_i$ has been transmitted. Therefore, the choice of the partition $\mathcal{C}$ has a major impact on the final distortion. Eqn. (2) shows that the respective error resilience of FLCs and multiplexed codes are quite close, depending on the statistical characteristics of the source $\mathbf{S_H}$.

Let us now make a similar analysis for the binary multiplexed codes. The above expression of the MSE obtained for the high priority source $\mathbf{S_H}$ still applies. We will try now to derive measures of symbol error rates (SER) and compare the SER performances for FLCs and multiplexed codes.

The probability to decode $\hat{S}_t = a_i$ having emitted $S_t = a_i$ is function of the length $l_i$ of the codeword $\overline{b_i}$: $Pr(\hat{S}_t = a_i / S_t = a_i) = (1-p)^{l_i}$. The SER for the high priority source $\mathbf{S}_H$ is hence given by

$$1 - \text{SER}_{S_H} = \sum_{a_i \in \mathcal{A}} \mu_i (1-p)^{l_i} \quad (3)$$

Let $h_{\mathcal{C}}$ denotes the mdl of a symbol of $\mathbf{S}_H$. For low bit error rates (BER), the approximation $(1-x)^n = 1 - nx + O(x^2)$ leads to

$$\begin{aligned} \mathrm{SER}_{S_H} &= 1 - \sum_{a_i \in \mathcal{A}} \mu_i (1 - p\, l_i + O(p^2)) \quad (4) \\ &= p \sum_{a_i \in \mathcal{A}} \mu_i\, l_i + O(p^2) = p\, h_{\mathcal{C}} + O(p^2). \end{aligned}$$

For FLCs, the SER is given by $1 - \mathrm{SER}_{\mathrm{FLC}} = (1-p)^l$, which leads to the approximation $\mathrm{SER}_{\mathrm{FLC}} = p\, l + O(p^2)$, where $l$ denotes the length of the codewords. Then, it appears that

$$\mathrm{SER}_{S_H} = \frac{h_{\mathcal{C}}}{l}\, \mathrm{SER}_{\mathrm{FLC}}. \quad (5)$$

This approximation is accurate for BER lower than $0.1$ ($p < 0.1$). Since $h_{\mathcal{C}} \leq l$, Eqn. 5 shows analytically that multiplexed codes provide better SER than FLCs, for BER lower than $0.1$. It also shows that better is the compression efficiency of the code, higher is its resiliency to bit errors. This has also been verified experimentally on real sources (see Fig. 1). If we consider codewords on an integer number of bits, the Huffman code is known to lead to the lowest mdl $h_{\mathcal{C}}$. This observation together with the approximation made in Eqn. 4 (valid for small values of the bit error rate $p$) leads to the following property: For small values of $p$, among all the VLC codes, with codewords represented on an integer number of bits, that can be considered as the basis of construction of the multiplexed codes, the Huffman code is the code leading to the lowest SER for $\mathbf{S}_H$.

## 6. SIMULATION RESULTS

The multiplexed codes have been experimented with real sources. We have considered a simple image coding system based on a 2-stage wavelet transform followed by a simple scalar quantizer. High subbands have been multiplexed into low subbands. All the simulations have been performed assuming a BSC with varying BER. For high values of BER, the reconstruction is better when only the lowest subband is used for the reconstruction. Significant improvements, both in PSNR and visual quality, can be observed in Fig. 2 when using multiplexed codes in comparison with FLCs and Huffman codes, and this for a compression factor equal to the one obtained with Huffman codes. Notice that the noise resulting from the bit errors is of a *salt and pepper* nature. Therefore, further quality improvement can be easily obtained by applying an adequate post-processing filter (de-noising filter) on the decoded image.

## 7. CONCLUSION

We have introduced a new family of "multiplexed codes", called binary multiplexed codes and constructed from VLCs. Theory and simulations have evidenced very high error resilience at almost no cost in terms of compression efficiency. Another advantage is that they allow to make use of simple decoding techniques. They hence appear to be excellent alternatives to reversible variable length codes (which suffer from some penalty in terms of compression efficiency, while not avoiding completely error propagation) or to classical VLCs for which robust decoding make often use of computationally expensive estimation techniques. Soft decoding techniques can also be applied on these codes, in order to further reduce the residual bit error rate, however at the expense of additional decoding complexity.

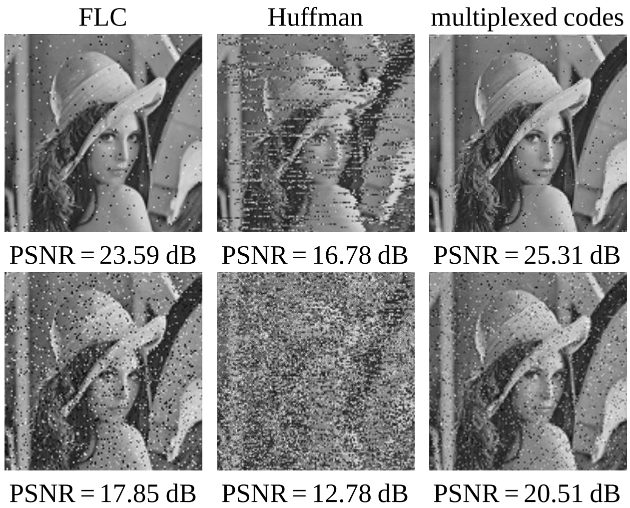


**Fig. 2**. PSNR performance and visual quality obtained respectively with FLCs ($3.25bpp$), Huffman codes ($1.36bpp$) and binary multiplexed codes ($bpp = 1.36$). The channel bit error rates are 0.01 (top images) and 0.05 (bottom images).

## 8. REFERENCES


[1] H. Jégou and C. Guillemot, "Robust multiplexed codes for compression of heterogeneous data," *submitted to IEEE Trans. Inform. Theory*, July 2002.

[2] J.P. Robinson J.C. Maxted, "Error recovery for variables length codes," *IEEE Trans. Inform. Theory*, vol. IT-31, no. 6, pp. 794–801, Nov. 1985.

[3] J.H. Rabinowitz T.J. Ferguson, "Self-synchronizing huffman codes," *IEEE Trans. on Inform. Theory*, vol. IT-30, no. 4, pp. 687–693, July 1984.

[4] W.M. Lam and A.R. Reibman, "Self-synchronizing variable-length codes for image transmission," in *Proc. IEEE Int. Conf. on Accoustic Speech and Signal Processing, ICASSP'92*, Sept. 1992, vol. 3, pp. 477–480.

[5] Y. Takishima, M. Wada, and H. Murakami, "Reversible variable length codes," *IEEE Trans. Commun.*, vol. 43, no. 2/3/4, pp. 158–162, Feb. 1995.

[6] J. Wen and J. D. Villasenor, "Reversible variable length codes for efficient and robust image and video coding," in *Proc. IEEE Data Compress. Conf., DCC*, Apr. 1998, pp. 471–480.