Contemporary Engineering Sciences, Vol. 3, 2010, no. 5, 237 - 248

Unequal Error Protection for Robust Video Transmission Using Multiplexed Codes

O. Serrar [1,2], A. Tamtaoui [2] and F. Ragragui [1]

[1] Mohamed V University - Agdal
Faculty of Sciences, LIMIARF
4 Avenue Ibn Battouta B.P. 1014 RP, Rabat, Morocco

[2] National Institute of Posts and Telecommunications
Images Processing Laboratory
Avenue Allal Al Fassi, Madinat Al Irfane 10100, Rabat, Morocco

e-mail: serrar_ouafae@yahoo.fr

Abstract

An error resilient video coding scheme based on unequal error protection over error prone channels is proposed. Due to the sensitivity of VLC to channel errors, a single bit error can cause loss of synchronization at the receiver, leading to catastrophic degradation of image quality. Multiplexed codes are an effective approach to combat losses of synchronization. In this paper, multiplexed codes appropriate to video coding are designed and then integrated into a block based motion compensated video coder. We compared performances of this structure with one of the most used codes: Huffman code. Results show that the proposed scheme achieves high error resilience at almost no cost in compression efficiency.

Keywords: Entropy coding, multiplexed codes, synchronization losses, unequal error protection, video coding

1. INTRODUCTION

Robustness to transmission errors is a crucial requirement for video communication. Video streams are very sensitive to transmission errors because

these errors hinder decoder from reconstructing desirable video frames. Moreover, due to the spatial-temporal prediction, a single erroneously reconstructed sample can lead to errors in the samples within the same and following frames. It is a challenging problem to design robust video coding schemes that are resilient to transmission errors.

Widely used in modern communications due to their good compression capabilities, variable-length codes (VLCs) have the disadvantage of being very sensitive to errors. in fact, a single bit error can cause synchronization losses at the receiver, making remainder of the bitstream useless. Header extension codes, data partitioning, and variable length coding can be protected during transmission by partitioning the video into portions of varying importance, such as headers, motion vectors, and DCT values. Error correcting codes, such as Reed-Solomon Erasure codes or feedback-based protection schemes (e.g ARQ) can also be added to the video portions to improve robustness. However, since the strength of the error protection is based only on the data partition type, these error protection methods ignore the data actually contained in the data streams. This results in increased overhead for coding non-important portions of the video streams. More, if errors encountered exceed the capacity of the channel encoder the position of symbol boundaries are not properly estimated, leading to dramatic symbol error rates.

To regain synchronization, several error correction tools have been developed .Various techniques include resynchronization makers [6] [1] [14]; the data between the synchronization point prior to the error and the first point where synchronization is re-established are typically discarded. The inserted resynchronization markers and header information are redundant and they lower the coding efficiency. RVLC has been developed for the purpose of data recovery at the receiver [13] [15]. Using this tool, the variable length codes are designed so that they can be read both in the forward and reverse directions. This allows the bitstream to be decoded backwards from the next synchronization marker until the point of error. The increased re-synchronization capability is however often obtained at the expense of redundancy. Error Resilience Entropy Coding (EREC) technique was proposed [8] [10]. In this method, the incoming bit-stream is reordered without adding redundancy such that longer VLC blocks fill up the spaces left by shorter blocks in a number of VLC blocks that form a fixed-length EREC frame. Such fixed-length EREC frames of VLC codes are then used as synchronization units, where only one EREC frame, rather than all the codes between two synchronization markers, will be dropped should any VLC code in the EREC frame be corrupted due to transmission errors. However, much computation power and a large memory buffer are required. Soft VLC decoding ideas, exploiting residual source redundancy as well as the inter-symbol dependency, have also been shown to reduce synchronization losses effect [9][2][12]. The T_codes[3] and HVLC[7] have also been noted for their property of self-synchronization however at the expense of high decoding complexity.

Multiplexed codes [4] are an effective approach to combat synchronization losses of decoder. Their design principle relies on the fact that compression systems of real signals generate sources of information with different levels of priority. Using these codes, the risk of errors propagation is confined to the low priority information. In this paper, multiplexed codes appropriate to video coding are designed and then integrated into a predictive hybrid video coder. The objective is to achieve high error resilience at almost no cost in compression efficiency. We compared performances of this structure with one of the most used codes: Huffman code.

The remainder of this paper is organised as follows: Section 2 describes the principle of multiplexed codes. The construction and application of our multiplexed codes are detailed in section 3. Section 4 presents transmission performance results, including experimental burst error results and decoded images. The paper is concluded in Section 5.

2. MULTIPLEXED CODES: PRINCIPLE

In this section we will briefly recall some properties of multiplexed codes. For more details see [4].

Let $S_H = (S_1,...,S_t,...S_{KH})$ be a sequence of source symbols of high priority taking their values in a finite alphabet A composed of Ω symbols. The stationary probability of the source S_H is denoted $\mu = (\mu_1,...,\mu_i,...\mu_\Omega)$ where μ_i stands for the probability that a symbol of S_H equals a_i.

Let $S_L = (S_1^{'},...,S_{i'}^{'},...S_{KL}^{'})$ be a sequence of source symbols of lower priority taking their values in a finite alphabet $A^{'}$. We assume that the realization S_L of this source has been pre-encoded into a bitstream $b = (b_1,...,b_r,...b_{KB})$ with a VLC coder (e.g. Huffman or arithmetic coder). Let c be a fixed number of bits reserved for the representation of any symbol of the alphabet A. The c bits define a set of $N = 2^c$ codewords. This set of codewords is partitioned into subsets C_i , $i = 1...\Omega$ called classes of equivalence associated to symbols a_i of the alphabet A. Each class of equivalence C_i contains a set of n_i codewords $\{c_{i,0}, c_{i,1},..., c_{i,q},..., c_{i,n_i-1}\}$, such that :

$$\sum_{i=1}^{\Omega} n_i = N \qquad (1)$$

A symbol $S_t = a_i$ of the sequence S_H can be encoded with any c-bit codeword $c_{i,q}$ belonging to the class of equivalence C_i. Hence; each symbol S_t can be mapped into a pair (C_i, q) of two variables denoting respectively the class of

equivalence and the index of the codeword in the class of equivalence C_i. The variable q is a $n_i - valued$ variable, taking its value between 0 and $n_i - 1$ and representing the inherent redundancy of the c-bits fixed length codes.

A multiplexed code has been defined in [13] 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 A (on which the high priority source is quantized) and q denoting the index of the codeword in the class of equivalence associated to a_i, as: $c_{i,q} \leftrightarrow (a_i, q)$. The sequence of symbols used to describe jointly S_H and S_L data flows is obtained by the completion of following steps:

1. For each symbol $S_t = a_i$ of S_H, we determine n_t cardinality of the class of equivalence associated to S_t.
2. The quantity $\Lambda = \prod_{t=1}^{KH} n_t$ denotes the number of different multiplexed sequences of codewords that can be used as a coded representation of the sequence S_H.
3. Only $K_B^{'} = \lfloor \log_2(\Lambda) \rfloor$ bits of the lower priority bitstream b can be stored.
4. The last $K_B^{'}$ bits of b are then seen as the binary representation of the integer θ comprised between $0 \; and \; 2^{K_B^{'}} - 1$, that can be expressed as:
$$\theta = \sum_{r=1}^{K_B^{'}} b_{\left(r+K_B-K_B^4\right)} 2^{r-1} \qquad (2)$$
5. The variable θ must then be expanded into a sequence of pairs (n_t, q_t) that will provide entries in the multiplexed codes table codes, hence allows selecting the sequence of c-bits fixed length codewords to be transmitted on the channel.
$$q_t = \left\lfloor \frac{\theta}{\prod_{t'=1}^{t-1} n_{t'}} \right\rfloor \bmod n_t \qquad (3)$$
6. If $K_B^{'} \le K_B$ the $K_B^{'} - K_B$ first bits of the bitstream b are then concatenated to the sequence of multiplexed codewords.

Since all operations are reversible, the decoding proceeds in the reverse order.

Table I: An example of multiplexed codes [4]: $A = \{a_1, a_2, a_3, a_4, a_5\}$ and c=3

Class C_i	Codeword x	Symbol a_i	$_i$	Probability μ_i	Index q_i
C1	000 001 010	a1		0.40	0 1 2
C2	011 100	a2		0.20	0 1
C3	101	a3		0.20	0
C4	110	a4		0.10	0
C5	111	a5		0.10	0

3. CONSTRUCTION OF MULTIPLEXED CODES APPROPRIATE TO VIDEO CODING

The most effective video compression standards use the motion-compensated technique to achieve high degrees of compression at acceptable levels of picture quality. The coder compresses video into a bitstream composed of packets. Each packet begins with a header, which is followed by motion information, texture information, and stuffing bits. The aim of our work is to construct a multiplexed code appropriated to video coding such that the motion vectors are considered as priority source S_H, while the texture, converted into binary using Huffman code , represents the lowest priority source S_L. Constructing multiplexed code consists of creating fixed length codes (FLC) that represent S_H (motions vectors) and exploit the inherent redundancy to store information of the low priority source S_L(texture). For this purpose, we first calculated μ_i of each motion vector by proceeding to a statistical study, then we determinated n_i of different class of equivalence, and finally we integrated the obtained multiplexed codes into video coding scheme.

3.1 Motion vector statistics

S_H takes its values in the set of motion vector's alphabet $MV = \{mv_1, mv_2, mv_3, ..., mv_{30}, mv_{31}\}$. Each motion vector mv_i corresponds to one class of equivalence. Since all elements of MV should be classified in decreasing order of probabilities $\mu = (\mu_1 .. \mu_{31})$, the first step was providing a probabilistic characterization of different motion vectors. For this purpose, we extract the motion vectors from different types of well-known standard test sequences. The sequences we tested comprise three different groups according to their content: objects are moving at slow, moderate or high speed. With the resulting vectors, frequency of occurrence of vector values are obtained, then

different probabilities $\mu = (\mu_1 .. \mu_{31})$ are calculated.

3.2 Class of equivalence

Once we had μ_i of each mv_i, the second step was to determine n_i the number of code in each class of equivalence associated to motions vectors. 6 bits are used to represent any motion vector of MV. So, in order to have a bite rate close to the entropy bound of the source S_H, we had to choose an efficient partitioning of 2^6 FLCs in classes of equivalence [5] such as:

$$(n_1, \ldots, n_i, \ldots, n_{31}) = \arg\min(\hat{h}) \quad (4)$$

$$\hat{h} = -\sum_{i=1}^{31} \mu_i \log_2 \frac{n_i}{64} \quad (5)$$

$\hat{h}$ is the expected description length (EDL)

Table III: Class C1 corresponding to vm=0

Class C_i	Codeword x	Symbol a_i	$_i$	Probability μ_i	Index q_i
C1	000000	0		0.1250	0
	000001				1
	000010				2
	000011				3
	000100				4
	000101				5
	000110				6
	000111				7

Finally, we obtained the partitioning described in Table II. To condense this table we mention in third column only the number of codes per class of equivalence n_i. However, as an example, the class C1 is detailed in Table III.

3.3 Application of multiplexed codes into video coding scheme

In [11] we applied a first version of multiplexed codes to scalable video coder. In this paper, we combined multiplexed codes with predictive hybrid video coder with motion compensation and block-based transform (Figure 1). Such coder uses motion estimation to predicting motion from frame to frame in the temporal direction, and then applies DCTs (Discrete Cosine Transform) to organize any redundancies in the spatial directions. An attempt was made to protect the motion vectors better than the residual information. The reason is that the motion vector

information provides significantly more information regarding the structured block than the residual does. It is important to note, however, that the B motion vectors are not useful for our purpose, so we are sub-sampling the video in a sense. We only have motion vectors for P frames, thus for our purpose, 10 frames per second will suffice.

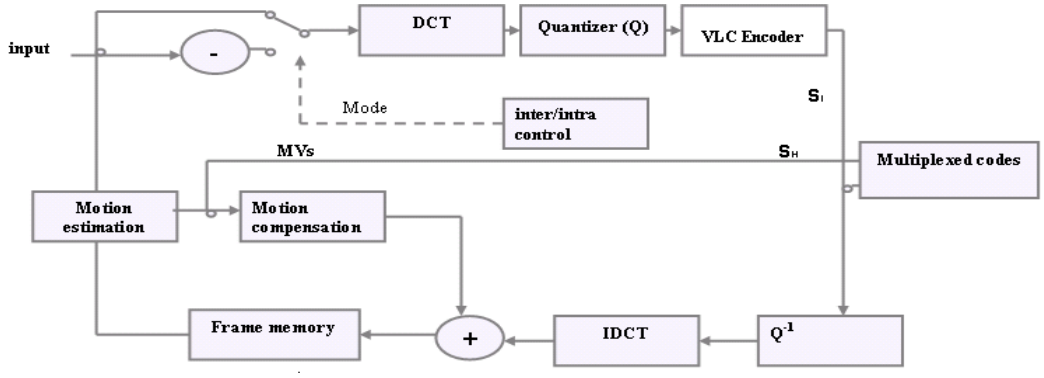

Figure 1 - Block based motion compensated video coder, combined with multiplexed codes

4. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed scheme, when the bitstream transmitted is corrupted and causes losses of synchronization of decoder, three tests have been performed on the first 100 frames of Foreman sequence and Carphone sequence. First test measured the General performance, in terms of PSNR, of multiplexed codes and compared the result with that of standard Huffman codes. Frames are simulated with different bit error rates (BERs) to consider various channel conditions. The transmission error is either a random single-bit error or includes successive error bits. Some simulated performances are listed in Table IV. Figure2 shows that even with errors injected into the bitstream, multiplexed codes do not suffer from any losses of synchronization. While the Huffman code can decode only less than 30% of the data received because of its high sensitivity to errors. The sequences were decoded with no error concealment technique in order to illustrate the effect of losses synchronization on reconstructed image quality. Such effect can be seen in "Huffman decoded images" in figure 3. However, the visual degradation of "multiplexed codes decoded images" is due just to the propagation of prediction errors over P images.

In the second test we have examined the compression efficiency. Results are shown in Table V. *SL_bits* represent the numbers of bits from S_L bitstream multiplexed with S_H. It can be seen that the higher compression is obtained with

multiplexed codes since SH_bits value corresponds to S_H and part of S_L (SL_bits) .

The last test compared the encoding and decoding durations of multiplexed codes with those of Huffman code. Figure 4 shows that multiplexed codes encoding is faster than Huffman encoding. However, in the case where there has no loss of synchronization, decode Huffman bitstream takes less time than decode multiplexed codes bitstream (figure 5).

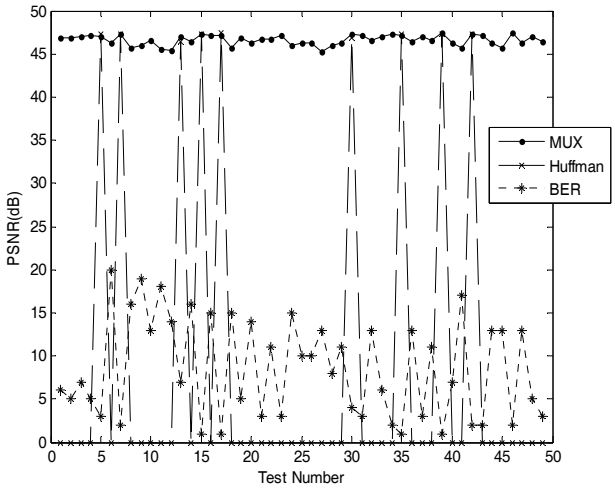

Figure 2: Simulations results, PSNR (dB), for multiplexed codes (MUX) and Huffman code when errors are injected into bitstream.

5. CONCLUSION

An error-resilient video coding scheme based on multiplexed codes is proposed. It exhibits low redundancy, low complexity, and high error tolerance. It can avoid synchronization losses of decoder. Hence, video quality can be well maintained.

Table IV: Average PSNR of 50 tests,

PSNR_MUX: Average PSNR for multiplexed codes (MUX) when errors are injected into bitstream.
PSNR_HUFF: Average PSNR for Huffman codes when errors are injected into bitstream.
PSNR_HUFF_NE: Average PSNR for Huffman codes when no errors are injected into bitstream.
R_NREJ_B: Rate blocs correctly decoded by Huffman codes

PSNR_MUX (dB)	PSNR_HUFF (dB)	PSNR _HUFF_NE (dB)	R_NREJ_B
46.66	16.42	47.08	26%
46.90	16.39	46.94	26%
46.86	13.67	46.98	22%
46.78	12.32	47.12	20%
46.77	10.93	46.96	18%
46.75	09.57	47.02	16%
46.78	10.94	47.05	18%
46.78	05.49	47.25	10%
46.88	08.20	47.01	14%

Figure 3: (1a): Foreman frame coded using Multiplexed codes. (1b) Foreman frame coded using Huffman code; (2a) Carphone frame coded using Multiplexed codes; (2b) Carphone frame coded using Huffman code;

Table V: Total number of bits generated for 2x99 motion vectors

Huffman_bits (bits)	1053	983	952
SH_bits (bits)	1188	1188	1188
SL_bits (bits)	1112	1134	1155
*SL_bits/(Huffman_bits + SL_bits)*100*	51.3%	53.5%	54.8%
*(SL_bits / SH_bits)*100*	93.6%	95.4%	97.2%

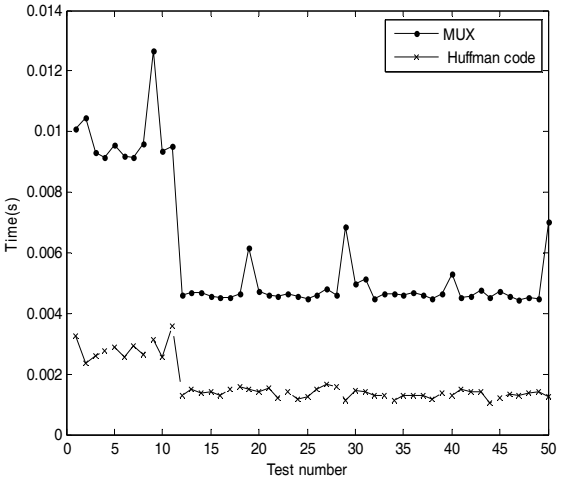

Figure 4: Encoding durations of multiplexed codes and Huffman code

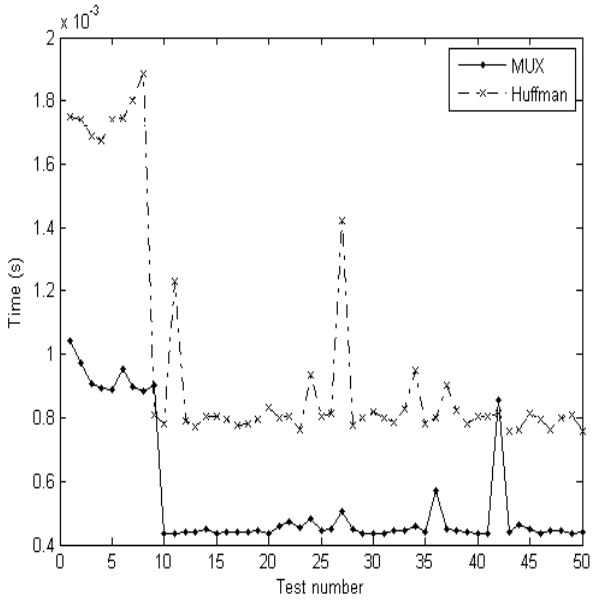

Figure 5: Decoding durations of multiplexed codes and Huffman code when no errors are injected into bitstream.

C_i	a_i	n_i	μ_i
C_1	0	8	0.1250
C_2	1	7	0.1054
C_3	-1	7	0.1039
C_4	-2	4	0.0600
C_5	2	4	0.0589
C_6	-3	3	0.0423
C_7	3	3	0.0409
C_8	4	3	0.0329
C_9	-4	3	0.0326
C_{10}	5	1	0.0262
C_{11}	-5	1	0.0258
C_{12}	6	1	0 .0231
C_{13}	-6	1	0.0216
C_{14}	-8	1	0.0198
C_{15}	8	1	0.0192
C_{16}	15	1	0.0189
C_{17}	-7	1	0.0187
C_{18}	-15	1	0.0186
C_{19}	7	1	0.0181
C_{20}	-9	1	0.0178
C_{21}	10	1	0.0173
C_{22}	9	1	0.0173
C_{23}	11	1	0.0167
C_{24}	-10	1	0.0166
C_{25}	14	1	0.0154
C_{26}	12	1	0 .0149
C_{27}	-12	1	0.0146
C_{28}	-13	1	0.0146
C_{29}	13	1	0.0145
C_{30}	-11	1	0.0144
C_{31}	-14	1	0.0141

Table II: Multiplexed codes applied to motion vectors (Ω=31, c = 6 bits)

References

[1] G.Cote, S.Shirani, and F.Kossentini, Optimal mode selection and synchronization for robust video communications over error-prone networks, IEEE J. Selected Areas Commun., vol. 18, no. 6,(2000) 952–965.

[2] N. Demir and K. Sayood, Joint source-channel coding for variable length codes, in Proc. IEEE DCC'98 (Mar 1998) 139-148.

[3] U.Günther P.Hertling R.Nicolescu and M.R Titchener, representing variable-length in fixed-length T-depletion format in encoders and decoders, journal of universal computer science (november 1997) 1207-1225.

[4] H.Jegou and C.Guillemot, Robust multiplexed codes for compression of heterogeneous data, Information Theory, IEEE Transactions, Vol. 51, Issue 4 (April 2005) 1393 - 1407.
[5] H.Jegou and C.Guillemot, Error-resilient first-order multiplexed source codes : Performance bounds, design and decoding algorithms, IEEE transactions on signal processing , vol. 54, Issue 4 (April 2006) 1483-1493.
[6] W.-M. Lam and A. R. Reibman, An error concealment algorithm for images subject to channel errors, IEEE Trans. Image Processing, Vol. 4, No.5 (May 1995) 533-42.
[7] Y.Lee, C.Yu and C.Lee, Error resilient hybrid variable length codec with tough error synchronization for wireless image transmission, Circuits and Systems, 2001. ISCAS 2001. The 2001 IEEE International Symposium on, Volume: 4 (2001) 326 -329.
[8] Y.Lee, C.Yu, H.Wei, Yen-Hsu Shih and Chen-Yi Lee, A novel DCT-based bit plane error resilient entropy coding scheme and codec for wireless image communication, Circuits and Systems,ISCAS 2002. IEEE International Symposium on , Volume: 5 (2002) 121 -124
[9] T. Liu and C. Lee, A low-complexity soft VLC decoder using performance modeling, Image Processing, ICIP '04, International Conference on Volume 5 (Oct. 2004)3233 - 3236.
[10]D. Redmill and N. Kingsbury, The erec: An error resilient technique for coding variable-length blocks of data, IEEE Trans. Image Processing, vol. 5 (Apr 1996) 565-574.
[11]O. Serrar, F.Regragui, A. Tamtaoui , Scalable video coder combined with multiplexed codes, Multimedia Computing and Systems, ICMCS '09. International Conference on, (April 2009) 309-312.
[12]S.Sun, T.Liu and C. Lee, A Self-Grouping and Table-Merging Algorithm for VLC-Based Video Decoding System, The 2006 IEEE Asia-Pacific Conference on Circuits and Systems, APCCAS'06, Singapore 2006.
[13]Y. Takishima, M. Wada, and H. Murakami, Reversible variable length codes, IEEE Trans. Commun., vol. 43, no. 2/3/4, (Feb. 1995)158-162.
[14] A.Vetro, Xin, J.; Sun, H., Error Resillence Video Transcoding for Wireless Communications, IEEE Wireless Communications, Vol. 12, Issue 4, (August 2005)14-21
[15]J. Wen, J. D. Villasenor, Reversible variable-length codes for robust image and video transmission, Proc. 31st Asilomar Conf. on Signals, Systems, and Computers, Vol. 2, (Nov. 1997) 973-979.

Received: April, 2010