Error Resiliency Measure for RVLC Codes

Liyang Xu, *Student Member, IEEE,* and Sunil Kumar, *Senior Member, IEEE*

***Abstract*—Video signals transmitted over error-prone wireless channels are often corrupted by bit errors. Due to their superior data recovery performance, the reversible variable length coding (RVLC) has been used as an error resiliency scheme in the MPEG-4 and H.263 video coding standards. In order to evaluate and compare the error resiliency performance of RVLC codes, we develop an error resiliency measure in this letter. Using this measure, different RVLC schemes are compared for MPEG-4 video sequences. The effects of propagating and nonpropagating errors are discussed as well.**

***Index Terms*—Error propagation, error resiliency, MPEG-4, reversible variable length codes (RVLCs), video coding.**

I. INTRODUCTION

REVERSIBLE variable length codes (RVLCs) have been used as an error resiliency scheme in the MPEG-4 [1] and H.263 [2] video coding standards. Several RVLC schemes have been recently reported in the literature, such as Toshiba (adopted in MPEG-4) [4], Tsai and Wu [5], and Lakovic and Villasenor [6]. An RVLC codeword usually consists of a variable length prefix and a fixed-length suffix. The bit error(s) in RVLC coded data could be "*propagating*" or "*nonpropagating*" in nature. If the corrupted codeword happens to be another valid codeword of the same length, the error is known as a "*nonpropagating error*" because it does not affect the subsequent codewords. On the other hand, an error that produces a decoded codeword of different length, affects the decoding of subsequent codewords, and causes loss of synchronization is known as a "*propagating error*." For example, an error in suffix bit(s) will not propagate, since the number of suffix bits is fixed. However, an error in the prefix bits of a codeword can propagate.

No generalized measure is available to determine the *error resiliency* performance of an RVLC scheme. A computation-intensive mathematical approach was proposed in [3] to estimate the number of correctly decoded codewords in a data packet. Chujoh and Watanabe [4] briefly introduced a mathematical framework (based on the Markov transition matrix) to determine the error propagation distance (D_{ep}) by computing the average number of bits between the actual location of the first error to where the error is detected. However, they did not distinguish between the propagating and nonpropagating errors for computing D_{ep}. It must be noted that a nonpropagating error cannot be detected in the bit stream unless it introduces a syntax error. Considering the nonpropagating errors for computing D_{ep} can therefore result in an unreasonably large value of D_{ep}. Moreover, the effect of nonpropagating errors is usually not significant, as they do not affect the decoding of subsequent codewords.

Since the RVLC coded data can be decoded in the forward as well as backward directions, better data recovery is achieved when the percentage of propagating errors (as compared to the total number of errors) is small and the value of D_{ep} is short for propagating errors. In this letter, we study the error resiliency of various RVLC schemes in terms of the error propagation distance and the percentage of propagating errors. In Section II, we shall discuss a mathematical framework, in which D_{ep} is defined as the average number of bits between the location of the very first propagating error to where the error is detected. In Section III, we shall evaluate the performance of five RVLC schemes (Golomb-Rice (GR) [3], exponential-Golomb (EG) [3], Toshiba (MPEG-4) [1], [4], Tsai and Wu (T&W) [5], and Lakovic and Villasenor (L&V) [6]) using the proposed measure, which is followed by the conclusions in Section IV.

II. COMPUTATION OF ERROR PROPAGATION DISTANCE

When a certain codeword $x \in X$ (codeword table) is transmitted, three possible situations can arise at the receiver: a correct codeword received, an incorrect codeword with nonpropagating error(s) received, or an incorrect codeword with propagating error(s) received. Let the probabilities of these events be represented by $P_1(y) = \sum_{y=x} P(y|x)P(x)$, $P_2(y) = \sum_{x\in X, y\in X-\{x\}} P(y|x)P(x)$, $P_3 = \sum_{x\in X, y\in Y-X} P(y|x)P(x)$, in that order. Here, $P_1(y) + P_2(y) + P_3(y) = 1$, and Y is the set of all possible codes received by the decoder. Here, $P_2(y)$ is the probability of receiving a valid (but incorrect) codeword, and $P_3(y)$ is the probability of receiving an invalid codeword. In a binary symmetric channel (BSC) with a bit-error rate (BER) $p(0 < p < 1)$, we have

$$P(y|x) = \begin{cases} p^{1-d(x,y)}(1-p)^{d(x,y)}, & l(x) = l(y) \\ 0, & l(x) \neq l(y) \end{cases} \qquad (1)$$

where $P(y|x)$ represents the probability of receiving a codeword y with length of $l(y)$ when a codeword x with length of $l(x)$ was sent. The $d(x,y)$ is the Hamming distance between x and y [4].

We use a Markov transition matrix with $(N+1)$ states to compute the probabilities that the decoder will transit from one state to others, in the presence of propagating error(s). We shall use code tree traversal to describe the decoding procedure. There are following four types of decoding states.

1) *Initial state (I or State 1):* The decoder receives the very first codeword with propagating error(s) at this state, and the Markov state transition starts.

Manuscript received May 30, 2005; revised August 23, 2005. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Mauro Barni.

The authors are with the Electrical and Computer Engineering Department, Clarkson University, Potsdam, NY 13699 USA (e-mail: xul@clarkson.edu; skumar@clarkson.edu).

Digital Object Identifier 10.1109/LSP.2005.861594

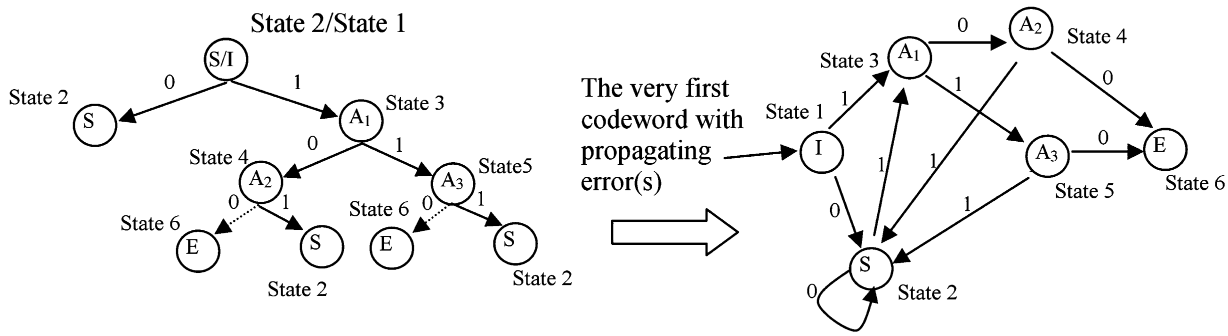

Fig. 1. Code tree for (left) $X = \{0\,101\,111\}$ and (right) its state flow.

2) *Synchronous state (S or State 2):* The decoder stops at a valid leaf after decoding an input codeword, and therefore, one valid codeword is obtained. This state verifies that the decoder has not detected an error so far. It is not necessary to distinguish the leaf at which the decoder stops, since decoding of the next codeword will begin from the root of the tree.
3) *Asynchronous state (A):* After decoding a received codeword, the decoder stops at an internal node of the code tree and does not output any decoded codeword. A propagating error must have occurred before this state. However, the decoder is not aware of the desynchronization and is still waiting for the incoming bits. For example, we have a codeword table $X = \{0\,101\,111\}$, and three codewords 0 0 101 are transmitted. If the first bit changes due to error, the decoder will decode 101 0 1. Here, the decoder is waiting for more incoming bits after decoding the fifth bit, and no error is reported. We use the internal nodes of the code tree to represent different asynchronous states and identify each of the states, as 3, 4, …, or N state.
4) *Error detecting state (E):* The decoder stops at a nonexistent leaf while decoding an input codeword. We denote it as state $(\mathrm{N}+1)$ in this letter.

The Markov transition matrix that represents state transition during binary code tree traversal is

$$\begin{array}{ccccccc} & I & S & A_1 & \dots & A_{N-2} & E \\ I & 0 & \theta_{12} & \theta_{13} & \dots & \theta_{1N} & \theta_{1N+1} \\ A_1 & 0 & \theta_{22} & \theta_{23} & \dots & \theta_{2N} & \theta_{2N+1} \\ \dots & 0 & \dots & \dots & \dots & \dots & \dots \\ A_{N-2} & 0 & \theta_{N2} & \theta_{N3} & \dots & \theta_{NN} & \theta_{NN+1} \\ E & 0 & 0 & 0 & 0 & 0 & 0. \end{array} \quad (2)$$

For each row (identified by i) of Markov transition matrix, we have $\sum_{j=1}^{N+1} \theta_{ij} = 1$, where θ_{ij} represents the probability of decoder transition from state i to j. The decoder transition is dependent on the current state and the received codeword. Please note that the "I" state precedes all other states, i.e., no other state exists until the "I" state appears. As shown in (2), the decoder will not return to the state "I" after decoding the first codeword corrupted by a propagating error. When decoder reaches the error detecting state "E," it stops further decoding.

Fig. 1 shows a code tree and its state flow, with the corresponding state number, for the codeword table $X = \{0, 101, 11\}$. All possible codewords received by the decoder are $Y = \{0, 1\,000\,001\,010\,011\,100\,101\,110\,111\}$. Please note that states 1 and 2 share the same node S/I in the code tree. A similar Markov transition matrix construction is used in [9].

The state transition probability $\theta_{ij} = \sum_{y \in Y'} P(y)$, where Y' is the set of all possible decoded codewords, which drives the decoder from state i to j. Here, $P(y) = \sum_{x \in X} P(y|x)P(x)$, where X is the codeword table. For transitions from the starting state $(i = 1)$ where the first codeword with propagating error(s) occurs, some events [such as receiving a correct codeword or another valid codeword (with nonpropagating error)] cannot happen. Therefore, the probability(ies) of receiving a codeword with a propagating error can be represented as

$$s = \sum_{j=1}^{N+1} \theta_{ij} = \sum_{x \in X, y \in Y'} P(y|x)P(x) - \sum_{x \in X, y \in X} P(y|x)P(x)$$
$$= \sum_{x \in X, y \in Y' - X} P(y|x)P(x) < 1.$$

We redefine $\theta_{1j} = \sum_{y \in Y' - X} P(y)/s$ to guarantee that $\sum_{j=1}^{N+1} \theta_{1j} = 1$. This satisfies the property of Markov transition matrix that sum of transition probabilities of every row should be 1. From (2), we get Θ as

$$\Theta = \begin{pmatrix} \theta_{11} & \theta_{12} & \dots & \theta_{1N} \\ \dots & \dots & \dots & \dots \\ \theta_{N1} & \theta_{N2} & \dots & \theta_{NN} \end{pmatrix}. \quad (3)$$

Let $C(q)$ denote the probability of error(s) being detected q codewords away from the codeword where the first propagating error occurred. From the Markov transition matrix in (2), we have

$$C(q) = R\Theta^{q-1}W \quad (4)$$

where $R = (1, 0, 0, \dots, 0)$ is a vector with N elements, and $R\theta^{q-1}$ fetches the first row of θ^{q-1}. Each element in the first row of θ^{q-1} represents the probability of the transition from initial state to another state j (state: $2, \dots, N$) after decoding $q-1$ codewords. $W = (\theta_{1N+1}, \theta_{2N+1}, \dots, \theta_{NN+1})^T$ represents the probability of the transition from state j to the error detecting state E (i.e., state $N+1$) after decoding the qth codeword.

Using (4), the expected error propagation distance (in the number of codewords) can be represented as

$$\sum_{q=1}^{\infty} qC(q) = \sum_{q=1}^{\infty} qR\Theta^{q-1}W = R\left(\sum_{q=1}^{\infty} q\Theta^{q-1}\right)W. \quad (5a)$$

We have

$$\sum_{q=1}^{\infty} q\Theta^{q-1} - \Theta \sum_{q=1}^{\infty} q\Theta^{q-1} = (I-\Theta)\sum_{q=1}^{\infty} q\Theta^{q-1}. \quad (5b)$$

Using series expansion in the left-hand side of (5b), we can show that

$$\sum_{q=1}^{\infty} q\Theta^{q-1} - \Theta \sum_{q=1}^{\infty} q\Theta^{q-1} = \sum_{q=1}^{\infty} \Theta^{q-1} = (I-\Theta)^{-1}. \quad (5c)$$

Using the right-hand side from (5b) and (5c), we can show that $\sum_{q=1}^{\infty} q\Theta^{q-1} = (I-\Theta)^{-2}$, where I is the identity matrix. From (5a), the expected error propagation distance (D_{ep}) in bits can be expressed as

$$D_{ep} = \left(\sum_{q=1}^{\infty} qC(q)\right)\left(\sum_{x\in X}(P(x)l(x))\right) = R(I-\Theta)^{-2}W\left(\sum_{x\in X}(P(x)l(x))\right) \quad (6)$$

where $\sum_{x\in X}(P(x)l(x))$ is the expected codeword length (in bits) [4]. We use (6) to mathematically compute the error propagation distance of different RVLC schemes in Section III.

III. Performance Evaluation

In this section, we shall study the error resiliency performance of five RVLCs for coding the MPEG-4 discrete cosine transform (DCT) coefficients and the English alphabet table [5]. It must be noted that the T&W and L&V RVLCs can adapt better to many different types of sources because they use a Huffman code as the reference to match the source distribution. We have used the generalized Gaussian sources [8] parameterized by ν (0.3, 0.5, and 0.7) and σ (1) to generate the T&W and L&V RVLC tables. An RVLC scheme can also be parameterized by choosing a suitable suffix length (K) to match the source distributions. The parameters of each tested RVLC scheme are optimized for MPEG-4 encoding in terms of their coding efficiency.

As shown in Fig. 2, the EG $(K = 3)$ and GR $(K = 3)$ schemes have significantly lower percentage of propagating errors due to the use of more (i.e., 3) suffix bits. Fig. 3(a) shows the error propagation distance for the RVLC schemes for coding MPEG-4 DCT coefficients at different BERs, by applying the mathematical relation in (6). For $BER < 10^{-2}$, L&V ($\nu = 0.3$, $K = 1$) RVLC has the shortest error propagation distance of approximately 65 bits. The error propagation distances for EG $(K = 3)$, T&W ($\nu = 0.3$, $K = 1$), GR $(K = 3)$, and MPEG-4 RVLC (Toshiba scheme) are 107, 109, 1200, and 5100 bits, in that order, for a video packet. We observed the similar results for the percent of *propagating errors* and *error propagation distance* for coding the English alphabet table.

Table I[1] shows the relative coding efficiency [using the efficiency of MPEG-4 RVLC as the 100% and average peak signal-to-noise ratio (PSNR) performance of the RVLCs]. We have chosen parameters for each RVLC scheme that achieve the best coding efficiency. The standard RVLC decoding and error recovery scheme proposed in MPEG-4 Annex E [10] has been used. The reconstructed frame quality at BER $= 5\times 10^{-4}$

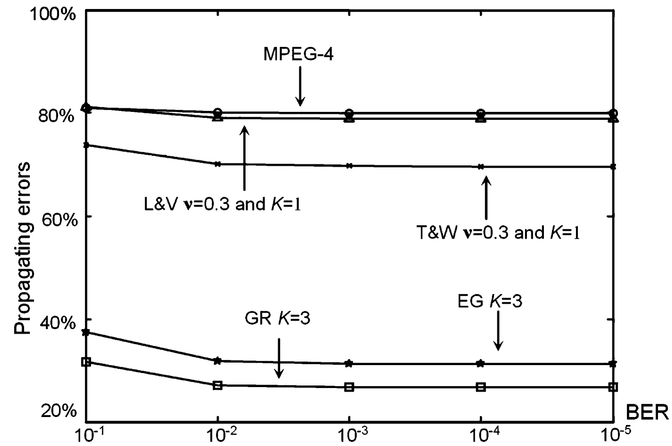

Fig. 2. Percentage of propagating error of five RVLCs for coding MPEG-4 Table B-23 coefficients.

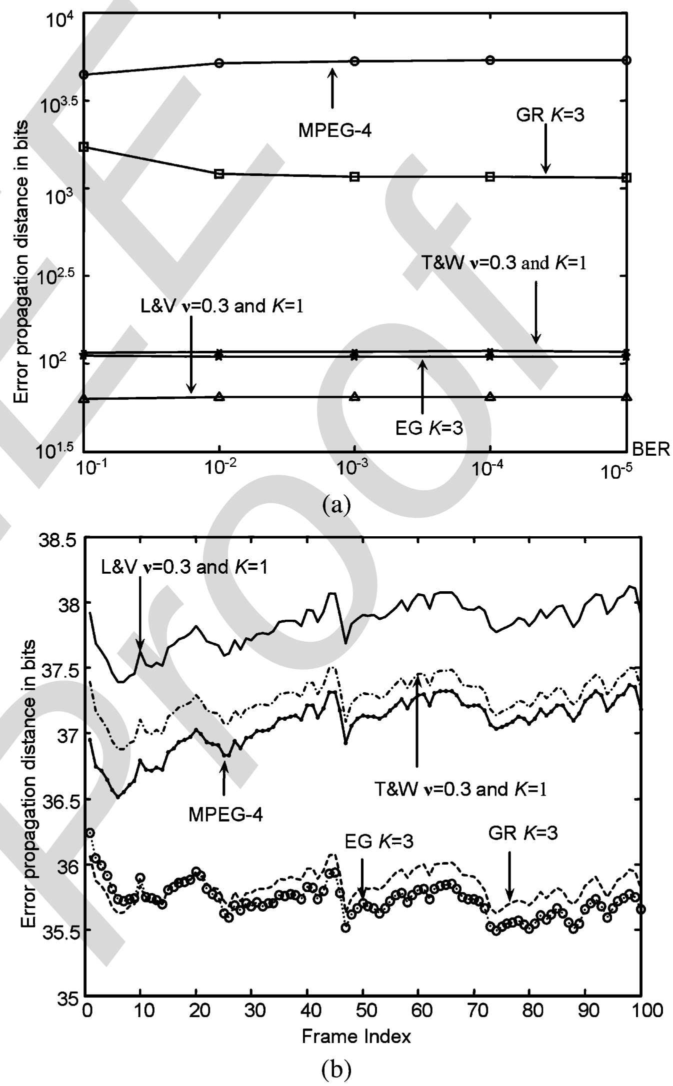

Fig. 3. (a) Error propagation distance of five RVLCs for coding MPEG-4 DCT coefficients in a video packet. (b) PSNR performance (Y-axis) of five RVLCs for coding "akiyo" sequence at BER $= 5\times 10^{-5}$.

[1]CIF sequence "akiyo" is encoded, with a fixed quantizer of 6. Only DCT coefficients are substituted with the test RVLC schemes.

is about 7–9 dB lower than that at BER $= 5 \times 10^{-5}$. This may be due to the fact that error concealment schemes are not so effective at BER $= 5 \times 10^{-4}$ due to the loss of many consecutive blocks in the corrupted region of a video frame. We have concealed the corrupted blocks by using their corresponding dc values. Here, we assumed that the dc value of these blocks is not corrupted due to data partitioning. Fig. 3(b) shows that L&V (with $\nu = 0.3$ and $K = 1$) has the best error resiliency performance (in terms of average PSNR) at random BER of 5×10^{-5} for "akiyo" test sequence, due to its larger Hamming distance $d_f \geq 2$. Furthermore, L&V ($\nu = 0.3$, $K = 1$) and T&W ($\nu = 0.3$, $K = 1$) RVLC schemes result in better performance than the RVLC scheme used in MPEG-4. We believe that it is due to their shorter error propagation distance. However, the PSNR performance of the EG ($K = 3$) and GR ($K = 3$) RVLCs is lower than that of the MPEG-4 RVLC, despite their shorter error propagation distance and higher proportion of nonpropagating errors. We discuss the reasons for this behavior in the following paragraph. Similar performance was also observed for the foreman, mother and daughter, and coast guard MPEG-4 video sequences. However, we have not included the results here due to space limitations.

Generally, an RVLC scheme that has shorter error propagation distance and higher proportion of nonpropagating errors performs better in terms of PSNR, because *1*) it enables recovery of more data from a corrupted packet during backward decoding, and *2*) it reduces the probability that a corrupted block is decoded and displayed as an error-free block. However, an RVLC codeword in MPEG-4 represents a combination of Level (magnitude of quantized DCT coefficients), Run (number of "0" magnitude coefficients preceding the current nonzero coefficient), and LAST (flag that indicates if the current coefficient is the last nonzero coefficient of the block) [1]. Even a nonpropagating error in a codeword can change the value of one or more of these attributes. Please note that the "Level" and "Run" changes cause only minor PSNR degradation since the error(s) are mostly limited to an 8×8 block [7]. However, change in the "LAST" mark breaks the data synchronization, which introduces serious distortion in the decoded bit stream. In such a case, the nonpropagating error introduces a syntax error, such as more than 64 decoded coefficients in a block or no data left for decoding the remaining blocks in the data packet, as discussed in [7]. Moreover, our observations reveal that these syntax error(s) can usually be detected only after they have propagated in the bit stream for long distance. Therefore, the presence of several nonpropagating errors in an MPEG-4 video packet is likely to significantly degrade the PSNR of a reconstructed frame. As a result, a lower proportion of propagating errors (i.e., when the fraction of nonpropagating errors is large) in the MPEG-4 bit stream may not always result in better error resiliency.

IV. Conclusion

We discussed the mathematical framework to evaluate error resiliency (i.e., error propagation distance) of an RVLC scheme. The error propagation distance depends on the code tree structure as well as source distribution. It may be noted here that the code tree structure itself depends on the properties of an RVLC scheme. The error resiliency performance of an RVLC scheme also depends on the syntax and semantics definitions of the bit stream. Generally, an RVLC scheme that has shorter error propagation distance performs better in terms of PSNR, because it enables recovery of more data from a corrupted packet during backward decoding. It also reduces the probability that a corrupted block is decoded and displayed as a noncorrupted block. Since the nonpropagating errors do not cause loss of synchronization in the bit stream, they introduce much less degradation in the reconstructed frame quality as compared to the propagating errors. However, change in the "LAST" mark in the MPEG-4 bit stream due to nonpropagating error does introduce the loss of synchronization in the bit stream. Therefore, it causes significant degradations in the reconstructed frame quality. Due to the shorter error propagation distance, L&V ($\nu = 0.3$, $K = 1$) and T&W ($\nu = 0.3$, $K = 1$) RVLC schemes provide better error resiliency performance than the RVLC scheme used in MPEG-4 with small coding overhead, for test video sequences.

TABLE I
RELATIVE CODING EFFICIENCY AND AVERAGE PSNR PERFORMANCE OF RVLCS FOR MPEG-4 VIDEO SEQUENCE 'AKIYO'

Codes	Coding efficiency	Avg. PSNR (dB) Sequence: akiyo	
		BER=5x10^{-5}	BER=5x10^{-4}
MPEG-4	100.00%	37.07	28.98
T&W (ν=0:3 and *K*=1)	100.96%	37.26	29.12
L&V (ν=0:3 and *K*=1)	102.60%	37.84	29.27
EG (*K*=3)	103.37%	35.73	28.54
GR (*K*=3)	104.83%	35.83	28.59

References

[1] ISO/IEC JTC1/SC29/WG11/N3908, MPEG-4 Video Verification Model Vers. 18.0, Jan. 2001.

[2] ITU-T Recommendation H.263, Video Coding for Low Bit Rate Communication, Annex D, Feb. 1998.

[3] J. Wen and J. Villasenor, "Reversible variable length codes for efficient and robust image and video coding," in *Proc. IEEE Data Compression Conf.*, Snowbird, UT, Mar. 1998, pp. 471–480.

[4] T. Chujoh and T. Watanabe, "Reversible variable length codes and their error detecting capacity," in *Proc. Picture Coding Symp.*, Portland, OR, Apr. 1999, pp. 341–344.

[5] W. Tsai and J. L. Wu, "On constructing the Huffman-code based reversible variable length codes," *IEEE Trans. Commun.*, vol. 49, no. 9, pp. 1506–1509, Sep. 2001.

[6] K. Lakovic and J. Villasenor, "On design of error-correcting reversible variable length codes," *IEEE Commun. Lett.*, vol. 6, no. 8, pp. 337–339, Aug. 2002.

[7] S. Kumar and L. Xu, "Improved RVLC-based data recovery in MPEG-4 video coding standard," *Real Time Imaging (Special Issue on Low Bit-Rate Multimedia Communication)*, vol. 10, pp. 315–323, Oct. 2004.

[8] J. Wen and J. Villasenor, "Structured prefix codes for quantized low-shape-parameters generalized Gaussian sources," *IEEE Trans. Inf. Theory*, vol. 45, no. 4, pp. 1307–1314, Apr. 1999.

[9] J. C. Maxted and J. P. Robinson, "Error recovery for variable length codes," *IEEE Trans. Inf. Theory*, vol. 31, no. 6, pp. 794–801, Jun. 1985.

[10] ISO/IEC JTC 1/SC29/WG11/N4350, ISO/IEC 14 496-2: MPEG-4 Video Draft International Standard, Annex E, Jul. 2001.