DECODING H.264/AVC USING PRIOR INFORMATION AND SOURCE CONSTRAINTS

Reuben A. Farrugia and Carl J. Debono

Department of Communications and Computer Engineering, University of Malta, Msida, Malta
reuben.farrugia@um.edu.mt, c.debono@ieee.org

ABSTRACT

The H.264/AVC standard employs a number of error-resilient mechanisms to correct transmission errors. These methods assume a packet-loss scenario, where all the macroblocks (MBs) contained within a corrupted slice are dropped and concealed. However, most of the MBs contained within corrupted slices provide minimal (if any) visual distortions and therefore concealing them causes a superfluous drop in the quality of the recovered video content.

This paper presents a novel error control mechanism which employs prior information and residual source-redundancy to recover the most-likelihood feasible H.264/AVC bitstream. Simulation results show that the algorithm recovers a number of corrupted sequences and achieves overall Peak Signal-to-Noise Ratio (PSNR) gains between 1dB and 2dB over the standard. The proposed solution is compatible with the H.264/AVC with no additional bandwidth requirements.

Index Terms—error control, H.264/AVC bitstream, prior information, residual source redundancy.

1. INTRODUCTION

Wireless video transmission offers a number of technical challenges [1], one of the most prominent being the extreme vulnerability of most video coding standards to transmission errors. These errors provide severe degradation in quality of the reconstructed video sequences with the effect of visually impaired macroblocks (MBs) propagating in both the spatial and the temporal domains.

H.264/AVC [2] offers several coding schemes that enhance the error resilience of the compressed video bitstream. However, these methods consider a packet-loss scenario, where the receiver discards all the corrupted packets and conceals all the MBs contained within them, even the uncorrupted ones [3]. This implies that the error resilient methods adopted by the standard operate at a lower bound since they assume a worst-case scenario.

A number of error control mechanisms are proposed in literature. The authors in [4] replaced the variable length code (VLC) tables of the MPEG-4 video compression standard with variable length error correction (VLEC) codes. However, VLEC codes reduce the compression efficiency of the codec since they have longer average codeword lengths. An analysis on the redundancy in compressed video data is found in [5], where it was concluded that significant gain in performance can be achieved when additional video data properties are taken into consideration while decoding. The same authors have later proposed a Viterbi decoding method for H.263 sequences [6]. Sequential decoding with source constraints are adopted in [7], [8] to recover feasible compressed video bitstreams. However, sequential decoding algorithms introduce variable decoding delays which could be a problem in real-time applications [9].

This paper presents a novel error control mechanism which can be used to derive the most-likelihood feasible H.264/AVC sequence. This method provides an iterative solution whereby, at each symbol time, the *M* most-likelihood sequences are stored within a list based on prior information. A set of source constraints is further adopted to eliminate the candidate solutions which provide semantically invalid H.264/AVC sequences, reducing the search space and consequently the computational complexity. This method achieves significant gains in quality when compared to the standard method with Peak Signal-to-Noise Ratio (PSNR) gains approaching 2dB.

This paper is organized as follows; the proposed error control mechanism is described in section 2 with particular emphasis on the source constrains and the prior information. Simulation results are presented in section 3 followed by final comments and conclusion in section 4.

2. PROPOSED ERROR CONTROL METHOD

Wireless video applications encapsulate each slice or frame within RTP/UDP/IP packets using the single Network Abstraction Layer Unit (NALU) mode [10]. The received payload contains a sequence having a length of N bits, whose number of VLC codewords and their original sequence are not known by the receiver and thus have to be estimated.

Once the bitstream is received, the proposed error control method creates an empty state $S_{0,0}$ and stores it in list L_{k-1}, which will contain incomplete feasible sequences. That is, sequences which satisfy all the following conditions: (1) the length of the bitstream is smaller than or equal to N, (2) the number of MBs decoded by the bitstream must be smaller than or equal to the number of MBs in the slice, and (3) the decoded symbols are within the ranges allowed by the H.264/AVC standard [2]. A second list F is initialized as an empty list. This list will contain all the complete feasible VLC sequences of size N such that the number of complete MBs being decoded is equal to the number of MBs in the slice.

The proposed algorithm adopts an iterative method, where each time L_{k-1} is available, a new L_k is initialized as an empty list. For each state in list L_{k-1} the following procedure is followed, assuming that the VLC table to be loaded is known a priori:

1. Load the VLC table. Each VLC instance is made up of a codeword c_i of length λ_i bits with prior probability p_i.
2. For each codeword in the VLC table:
   i. Extract λ_i bits from bitstream x and store this codeword within buffer b.
   ii. Compute the distance between the two codewords using:
   $$m = m' + (1 - p_i)\left|b, c_i\right| \qquad (1)$$
   where $|\ ,\ |$ stands for the hamming distance between the two codewords and m' represents the metric stored in the current state.
   iii. Create a state S, set the bitstream equal to the concatenation of the bitstream stored in the current state and the codeword c_i, and set the new metric equal to m.
   iv. Place the state S within the list L_k.

At the end of each recursion, the list L_k will contain the feasible VLC sequences of k symbols. The states enclosed within the list L_k which contain complete feasible H.264/AVC sequences are stored in F. On the other hand, the incomplete feasible VLC sequences are sorted in ascending order according to the metric, and the first M states are stored in L_{k-1} for the next iteration. The recursion is terminated when list L_k is empty with $k \geq 1$, at which point the list F will hold all the complete feasible bitstreams. The optimal sequence of codewords is then the complete feasible sequence with the smallest metric among all the sequences found in F.

This error control algorithm relies on the contextual module and prior information to function appropriately. The contextual module is used during each recursion to predict the VLC table to be loaded, while the prior information is required to ensure that only the most probable partial H.264/AVC sequences are kept in memory, thus reducing unnecessary storage of data in the lists and aid convergence.

2.1. Contextual Module

Two different entropy coding methods are supported by H.264/AVC: Context-Adaptive VLC (CAVLC) and Context Adaptive Binary Arithmetic Coding (CABAC). The Baseline profile, which is generally adopted for real-time wireless applications, adopts the CAVLC entropy coding method [8], [11] and therefore is the one considered in this work. CAVLC encodes the quantized coefficients using VLC tables which are switched depending on the previous syntax elements and the VLC table adopted in the previous decoding step to achieve higher coding efficiency [11], [12].

The H.264/AVC decoding process of an MB is illustrated in Fig. 1. The decoder keeps track of the processing order of the slice it is decoding and predicts the next VLC table for the following step based on the syntax elements and the current table. The Residual block shown in Fig.1 is adopted by the *Luma Residual*, *I16x16 DC Residual*, *DC Chroma Residual*, and *AC Chroma Residual* to derive the residual coefficients. More information about these coefficients is provided in [13].

To apply the proposed error control method to H.264/AVC bitstreams, it is necessary to each time identify which VLC table must be loaded by each state in list L_{k-1}. The contextual module thus stores all the relevant information for each state and is used to predict the table to be loaded next.

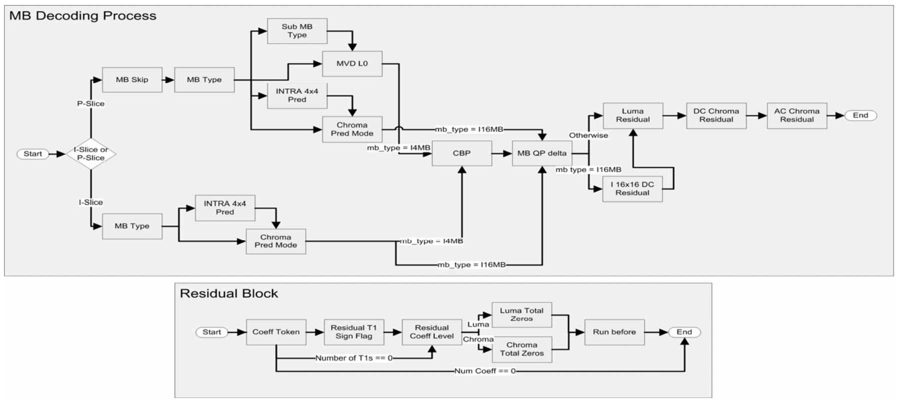

Fig. 1 The MB decoding process used by the H.264/AVC standard

2.2. Prior Information

The performance of the proposed method depends on the distance measure adopted in step 2.ii. The simplest measure that can be used is the hamming distance. However, this is a hard distance measure which does not consider a priori information that is available at the decoder and thus provides limited error control capabilities for VLC sequences. This work assumes that no soft information is available and thus the metric provided in (1) simply multiples the hamming distance with the prior information.

In practice, the H.264/AVC codec employs more than 30 different VLC tables whose distributions differ from one another. The probability of occurrence of each codeword within the VLC table was derived using six different video sequences, *Mobile*, *Coastguard*, *Suzie*, *Silent*, *Akiyo* and *Hall*, encoded at QCIF resolution at 30fps. These prior probabilities were appended to each VLC table, thus making them accessible to the error control method at the decoder.

3. SIMULATION RESULTS

The proposed error control mechanism was integrated within the Jointed Model (JM) software [14] which was modified to allow the decoder to decode also damaged bitstreams. The raw video sequences were encoded at QCIF resolution at 30fps, with the format IPPP... and a data rate of 128kbps. The encoder applies a random Intra refresh rate of 5 and has 5 slices per picture. In compliance with the JM software, each slice was encapsulated within RTP/UPD/IP packets and transmitted over a Binary Symmetric Channel (BSC) which yields the highest probability of error in the slice [12], where 8 different error patterns were considered. The sequences used in this section are different from those used to derive the prior probabilities to avoid biasing of results.

The *Foreman* sequence was used to analyse the effect of the size M on the PSNR and the results are summarized in Table I. It can be seen that if M is kept too small (below 100) the proposed error control mechanism does not provide any gain in performance over the Syntax Analysis method described in [15]. This occurs mainly because the feasible list F is most of the time empty and thus provides limited error control capabilities. However, significant gains in terms of PSNR are experienced as M increases. This comes at the expense of an increase in the complexity of the decoder and thus M cannot be too large.

Table I - PSNR for different List Size M

BER	**Syntax Analysis**	**LD M=100**	**LD M=200**	**LD M=300**
1.00E-03	19.825	20.058	20.424	20.288
5.00E-04	22.775	23.266	23.471	23.649
3.00E-04	23.976	24.950	24.889	25.597
1.00E-04	27.941	27.210	28.338	29.366
5.00E-05	28.382	28.847	29.052	28.967
3.00E-05	30.823	30.407	31.600	31.765

The list size M was set to 300 since this was found to provide the best compromise between performance and complexity. The performance of the proposed system was compared to both Syntax Analysis and the packet-loss method adopted by the standard by considering their performance on the *Foreman* and the *Carphone* sequences. These results are illustrated in Fig. 2, where the superiority of the proposed method is evident. The overall PSNR gains are between 1dB and 2dB when compared to the standard decoding method, whereas gains in the region of 1dB were achieved over the Syntax Analysis method.

The gain in quality is even more evident in the subjective results shown Fig. 3, where the Syntax Analysis method does not manage to detect a number of distorted MBs while the standard method conceals unnecessarily a number of uncorrupted MBs. Quantitatively it was found that, for this frame, the proposed decoding method achieves a PSNR gain of 5.63dB and 6.23dB over the standard decoding method and the Syntax Analysis method respectively.

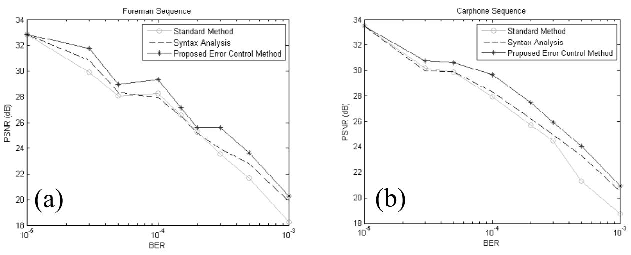

Fig. 2 PSNR for (a) Foreman and (b) Carphone sequences at different BER.

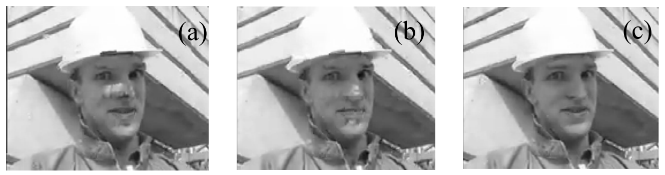

Fig. 3 Frame 202 for the Foreman sequence (a) Syntax Analysis method (27.00dB), (b) Standard Decoding (27.60dB), and (c) proposed solution (33.23dB).

4. CONCLUSION

This paper has presented a novel error control method which can be used to decode damaged H.264/AVC video sequences. This technique ensures that only feasible slices are decoded by exploiting the residual source constraints. It further adopts prior information to assign priorities within the lists of sequences.

The proposed solution manages to completely recover a number of corrupted sequences, thus instituting additional resilience to transmission errors. However, the recovered sequences may still contain errors which may provide visual artifacts whose effect can be alleviated using advanced error detection and concealment methods [16].

5. REFERENCES

[1] T. Stockhammer, and M.M. Hannuksela, "H.264/AVC Video for Wireless Transmission," *IEEE Wireless Commun.*, vol. 12, no. 4, pp. 6-13, Aug 2005.

[2] *Advanced Video Coding for Generic Audiovisual Services*, ITU-T Rec. H.264, 2005.

[3] M. Etoh, and T. Yoshimura, "Wireless Video Applications in 3G and Beyond," *IEEE Wireless Commun.*, vol. 12, no. 4, pp. 66-72, Aug. 2005.

[4] V. Buttigieg, and R. Deguara, "Using Variable-Length Error-Correcting Codes in MPEG-4 Video," in IEEE *Proc. ISIT'05*, 2005.

[5] H. Nguyen, and P. Duhamel, "Estimation of Redundancy in Compressed Image and Video Data for Joint Source-Channel Decoding," in *IEEE Proc. GLOBECOM'03*, 2003.

[6] H. Nguyen, P. Duhamel, J. Brouet, and. Rouffet, "Optimal VLC Sequence Decoding Exploiting Additional Video Stream Properties," in *IEEE Proc. ICASP'04*, 2004.

[7] C. Weidmann, P. Kadlec, O. Nemethova, and A. Al Moghrabi, "Combined Sequential Decoding and Error Concealment of H.264 Video," in *IEEE Proc. MMSP'04*, 2004.

[8] C. Bergeron, and C. Lamey-Bergot, "Soft-Input Decoding of Variable Length Codes applied to the H.264 Standard," in *IEEE Proc. MMSP'04*, 2004.

[9] S. Lin, D.J. Costello, *Error Control Coding: Fundamentals and Application*, Englewood Cliffs, Prentice-Hall, 1983.

[10] S. Wenger, M.M. Hannuksela, T. Stockhammer, M. Westerlund, and D. Singer, "RTP Payload Format for H.264 Video," *IETF RFC 3984*, Feb. 2005.

[11] G.J. Sullivan, and T. Wiegand, "Video Compression – From Concepts to the H.264/AVC Standard," *Proc. IEEE*, vol. 93, no. 1, pp. 18-31, 2005.

[12] T. Wiegand, G.J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560-576, Jul. 2003.

[13] G. Bjøntegaard, and A. Luthra, "Context-adaptive VLC (CAVLC) coding of coefficients", *document in JVT-C028rl*, Fairfax, VA, May 2002.

[14] H.264/AVC Software Coordination, "JM Software," ver. 12.2

[15] L. Superiori, O. Nemethova, and M. Rupp, "Performance of a H.264/AVC Error Detection Algorithm Based on Syntax Analysis," *J. of Mobile Multimedia*, vol. 3, no. 4, pp. 314-330, 2007.

[16] R.A. Farrugia and C.J. Debono, "A Robust Error Detection Mechanism for H.264/AVC Coded Video Sequences based on Support Vector Machines," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 12, pp. 1766-1770, Dec. 2008.