An Improved Error Resilience Scheme for Highly Compressed Data

Lorcan Mac Manus and Anil Kokaram

Department of Electronic and Electrical Engineering,
University of Dublin, Trinity College, Dublin 2, Ireland.

Abstract—The coding of highly compressed data streams involves removing as much redundancy from the stream as possible. However, as redundancy is removed, so is the ability of the decoder to recover from error conditions caused by wireless channels, or other lossy communications links. Standard techniques for providing some protection to the stream against channel errors usually involve adding a controlled amount of redundancy back into the stream. Such redundancy might take the form of resynchronization markers, which enable the decoder to restart the decoding process from a known state, in the event of transmission errors.

Transcoding schemes, i.e. algorithms which rearrange the data without altering the bitstream length, can be very successful too; the Error Resilient Entropy Coding (EREC) scheme is particularly applicable to compressed video streams. EREC can be quite easily incorporated into the stack architecture of common communications protocols. This paper presents a modification to EREC which greatly improves its ability to recover uncorrupted data in the event of errors. The new scheme, called Code-Aligned EREC (CA-EREC) achieves this by making minor, imperceptible adjustments to some of the data in the bitstream. This allows the codewords to align with the standard EREC packet boundaries. This paper demonstrates that a subtle change to the standard EREC scheme reduces code loss to an absolute minimum in the event of errors in the channel. Although the scheme presented here has a wider applicability, this paper focuses on Video Coding, and more specifically on the MPEG-4 Video Coding standard.

Keywords—MPEG-4, error resilience, Error Resilient Entropy Code (EREC), Code-Aligned EREC (CA-EREC)

I. INTRODUCTION

WITH the current and recent explosion in wireless, mobile and internet communications, video compression techniques which provide an acceptable quality of service in these error-prone environments have never been more relevant. Consider the traditional analogue methods of transmitting video; transmission here can also be subject to poor channel conditions, however the picture normally exhibits ‘graceful degradation’ so that as channel conditions worsen, the picture quality degrades accordingly. Digitally compressed video suffers from the problem that a small error in the bitstream can have a catastrophic effect on the resulting image. Many techniques for error reconstruction focus on working with the decoded images and take advantage of the temporal and spatial redundancies present in typical images. Techniques which work with the general problem of reconstructing damaged images from the spatial or temporal neighbourhood of the missing pixels are discussed at length in [1]. An error handling technique which combines information from the bitstream with some of these techniques has been presented in [2].

The problem of error handling in personal, mobile, wireless systems is even more severe. Typically such systems involve communications using very low bandwidth media streams. This is because there are typically huge numbers of such de-

vices wishing to access central media repositories (e.g. media databases, access channels) and because mobile devices have limited computing power available for handling media streams. As a result, errors in typical mobile wireless channels cause a more serious effect on the decoded bitstreams. Examples of such devices are the Compaq IPAQ, Handspring Visor, Palm Pilot, etc (also called Personal Digital Assistants, or PDAs). All of these devices can be adapted for use with wireless networking through the use of appropriate radio interfaces.

Because of these more stringent restrictions in the infrastructure supporting mobile communications with PDAs, robustness to errors in video communications with such devices is receiving renewed attention [3], [4], [5]. The interaction between the networking layers and the video application itself is important here [6]. Transcoding schemes are very appropriate for incorporation into this paradigm as they reduce the effort required by error post-processing mechanisms. One particularly applicable scheme for video is the well known Error Resilient Entropy Coding (EREC) scheme described by Kingsbury et al [7], [8]. It is such a scheme that is the subject of this paper, however, to allow the reader to better appreciate the context in which this scheme operates, we will first have a brief discussion on more traditional error handling strategies.

A. Traditional Error Handling Strategies.

The nature of highly compressed digital video is such that traditional error detection and correction strategies such as Forward Error Correction (FEC) do not work well. This has been noted by Shyu et al [9]. A more thorough discussion of these error handling strategies, which work by the addition of controlled redundancy can be found in [10].

Another standard technique called Automatic Repeat Request (ARQ), works by allowing the receiver to request portions of the transmitted data to be resent in the event that they arrive corrupted or do not arrive at all. This technique is not suitable for broadcast video however, as quite often the receiver does not have a backchannel to signal to the transmitter which parts of the video have been lost and even when it does, by the time the retransmitted data arrives, it is often too late, as the images to which the data relate have already been displayed.

MPEG-4 video provides two key approaches to error control. Special codes, called Resynchronisation Markers may be placed by the encoder in the video stream after a set number of bits. This allows the decoder to detect errors and to restart the decoding process from a known state. Another technique

is that of Data Partitioning where the image data is separated into high priority data and low priority data [11]. The transmission channel may then exploit this segregation by offering a different quality of service to each data stream.

It is, however, the loss of bitstream synchronisation which causes the greatest amount of corruption in the decoded pictures. A single bit error can result in loss of synchronisation. Since the Huffman coding scheme has been chosen to maximise compression efficiency, the codes are usually short and detection of an invalid codeword is very rare [12]. (To allow many invalid codewords would mean increasing the redundancy in the coding scheme.) Thus having many resynchronisation points in the coded stream is useful for two reasons: it allows the decoder to start from a known state and it is frequently the method by which errors are detected in the first place. In MPEG-4, resynchronisation markers are byte-aligned 17 bit codewords. Byte-alignment adds between one and eight bits, or on average 4.5 bits. Each codeword thus adds an average of 21.5 bits to the overall image length. Thus in a CIF¹ format image, compressed at 0.5 bits per pixel, 132 resynchronisation points adds approximately 5.6% (2800 bits) to the bitstream length. This example is deliberately chosen as it may be compared with EREC, which in the example shown below adds 132 resynchronisation points to a similarly compressed image with an overhead of just 0.2%.

The remainder of the paper is laid out as follows: first a discussion on the standard EREC algorithm is presented, which highlights a key deficiency in the scheme. Then the new Code-Aligned EREC scheme is outlined and it is demonstrated how this scheme overcomes the drawbacks of the EREC scheme. This is followed by a more detailed description of how this scheme is applied to MPEG-4 and the paper ends by presenting some results and conclusions.

II. THE ERROR RESILIENT ENTROPY CODE

The new scheme being proposed is best thought of as a modification of the Error Resilient Entropy Coding (EREC) scheme. In the EREC scheme implicit resynchronisation markers are placed in the bitstream with a negligible overhead. The scheme is most suitable for classes of data which can be naturally broken into a number of blocks, each block generating a stream of bits (Figure 1). Video coding is a good example of these types of data.

Fig. 1. Image blocks generate variable length blocks of data.

¹ 352 pixels × 288 pixels

The resulting blocks of data might look something like Figure 2. The traditional way to enable resynchronisation at the start of each of these blocks is to insert a special marker, which cannot be emulated by any valid bitstream and which is itself, resilient to errors. As stated above, such *resynchronisation markers* are supported in MPEG4. They consist of sixteen zeros followed by a single one and the start of the resynchronisation marker is byte aligned. This scheme is simple and very effective, but as the example above illustrates it does add redundancy to the bitstream. The more frequently the encoder employs resynchronisation markers, the more protection it affords to errors, however it also inflates the bitstream, so a trade-off is required, balancing an acceptable quality of service with economic use of the available bandwidth.

Fig. 2. Image blocks generate variable length blocks of data.

The EREC allows for resynchronisation at the start of each block without the additional overhead of inserting redundant resynchronisation markers. It does this by cleverly rearranging the existing blocks so that they fit into a predetermined number, M , of fixed length frames. If the k^{th} block in the video stream has a length of T_k , then the k^{th} EREC frame has a length of S_k and

$$\sum_{n=1}^M T_n = \sum_{k=1}^M S_k$$

So in return for M resynchronisation points in the stream, all that needs to be transmitted to the decoder is the overall bitstream length N . For a typical intra coded picture of 50 Kbits, with FEC allowing for a BER of 10%, this amounts to a coding overhead of just 0.2% [13].

The length of each EREC frame is determined as the integer part of N/M . If the remainder when N is divided by M is R bits, ($R < M$), then the first R EREC frames have their length increased by one bit. Then there are M EREC frames, with bit lengths $S_1, \dots, S_M$. There are M blocks to be transmitted, with bit lengths $T_1, \dots, T_M$.

So the task of the EREC algorithm is to rearrange the blocks $T_1, \dots, T_M$ into the fixed length blocks $S_1, \dots, S_M$. The algo-

rithm that accomplishes this can be quite simple, for example if a block bit length $T_k \leq S_k$ then all of T_k is placed at the start of slot S_k , leaving $R_k = S_k - T_k$ bits free in the k^{th} EREC slot. Then if a block $T_l > S_l$, the first S_l bits of T_l are placed in this slot and the remaining bits $X_l = T_l - S_l$ seek the next available slot as follows. If $R_{l-1} > 0$, R_{l-1} bits of X_l are placed here. This process is continued until all of the X_l bits have been placed. Should X_l attempt to seek a 'home' in the 0^{th} slot, the algorithm instead wraps around to the M^{th} (ie final) slot and proceeds as before. This process is depicted in Figure 3, below.

Fig. 3. EREC encoding process.

Now, as can be seen, the EREC allows decoding to recommence at the start of the next block in the event of an error being encountered. A closer look at the 'chopping' procedure

that occurs in Figure 3(b) is required to understand a shortcoming in this algorithm.

The length of an EREC frame S_k is quite arbitrary with respect to the number of bits in a block T_k . The block T_k is simply truncated to fit into the EREC frame. When it is considered that the block T_k is composed of a sequence of variable length codes, it is apparent that this truncation will often cut one of these variable length codewords in two and the remainder of the codeword will be placed in another EREC frame. With this in mind, consider what will happen should an error be encountered in block 5 in Figure 3(c) above. Some, or all of block 5 will have to be discarded; this is quite acceptable. However, the portion of block 5 which has been placed after block 4 in the fourth EREC frame *cannot be decoded either*, as the start of this block is not guaranteed to be a valid codeword, and there is no way of resynchronising with the bitstream. This means that the remainder of block 6, which has been placed in EREC frames 4 and 3 cannot be decoded either. Thus an error in one EREC frame may have an adverse effect on the decoding of other EREC frames. This is an acknowledged problem and it can be mitigated by arranging the block data so that the higher priority data occurs towards the start of the stream. This is naturally the case with the common video coding schemes. However, the data loss can be quite noticeable. Figure 4 shows the result of losing just half of the AC coefficients in a detail of the 'Caltrain' sequence and Figure 5 shows the same loss in the 'Mom-kid' sequence. The new scheme being presented here provides a much enhanced ability to recover the maximum amount of this AC data. This data would have to be discarded if the standard EREC scheme were used.

Fig. 4. The left image shows a detail from the 'Caltrain' sequence, and the right image shows the same detail with half of the AC coefficients missing. The degradation in picture quality is particularly noticeable in the blurring/ringing on the calendar.

Fig. 5. The left image shows a detail from the ‘Mom-Kid’ sequence, and the right image shows the same detail with half of the AC coefficients missing. The degradation in picture quality is more noticeable around the sharp edges in the image.

III. THE NEW SCHEME: CODE-ALIGNED EREC (CA-EREC)

The new scheme devised exploits the lossy nature of MPEG type video coding algorithms. At some point in most video coding schemes, there is a stage where the source image is divided into blocks, each block is typically an 8 pixel square. A Discrete Cosine Transform (DCT) is then applied to each block in turn. The effect of the DCT is to distort the distribution of information in the block, so that instead of being randomly spread over the entire 64 pixels, the information is compacted into the top left hand corner of the block, a region that corresponds to the low spatial frequencies present in the image. Noting then that the Human Visual System (HVS) is less sensitive to the higher spatial frequencies, an uneven quantisation may be applied to the DCT coefficients, which allows a more accurate reproduction of the grey-level and lower frequency terms than the higher frequency terms. It is this quantisation that achieves the image compression. The reconstructed image is different from the original, but the HVS is not particularly sensitive to the distortions which have been introduced.

The quantised blocks are then ‘zig-zag’ scanned (see Figure 6), and, because many of the coefficients are zero, it is advantageous to run-length code the reordered coefficients. All these coefficients are then transmitted as VLCs.

When the resulting bitstream is fed into the EREC transcoder, the EREC frames usually split VLCs in two, resulting in the problem outlined in the previous section. However, a minor adjustment to the quantised DCT coefficients can result in VLCs which coincide with the end of the EREC frame.

As an example, suppose the last bit of a particular VLC does not fit into the current EREC frame. A search is made of all of the AC coefficients that have been placed in this EREC frame.

Fig. 6. Simplified image of DCT compression being applied to a block and the zig-zag scanning that follows.

Suppose one is found that represents the first AC coefficient of a luminance block, and that this VLC has a value of 10. We know that the default quantisation matrix for MPEG-4 specifies that the first AC coefficient in a block be divided by 17. If we assume that the encoder rounded the quantised coefficient to the nearest integer, we know that the original DCT value for this coefficient lay somewhere between $9.5 \times 17 = 161.5$ and $10.5 \times 17 = 178.5$. The standard MPEG-4 specified VLC for a quantised AC value of 10 is ‘000111110’, a nine bit code word. For 9 is ‘001011110’, an eight bit code word. If the CA-EREC algorithm were to adjust this AC value from 10 to 9, then, it would supply the extra bit that would allow the last VLC to fit neatly within the EREC frame. This would ensure that the leftover bits would begin with a valid code word in the partially filled EREC frames (Blocks 1, 3 and 4 of Figure 3(c) above). The reconstructed DCT coefficient would then be $9 \times 17 = 153$, which obviously lies outside of the original range of possible values for this coefficient. In our experiments however, this distortion has not been noticeable.

The CA-EREC algorithm starts by setting up the standard EREC frames. It then writes the DC coefficients and block pattern codes at the start of each of the EREC frames, and follows this by writing all the AC coefficients in turn. When it reaches the end of an EREC frame it checks to see what adjustment (if any) is required to allow the last AC code word to fit exactly into the EREC packet. This is done in the following manner.

Supposing this code word has a length of C and that there are r remaining bits in the EREC frame (such that $r < C$). Now if $C - r < r$ fitting the code word into the EREC frame requires the least adjustment to the existing code words, and CA-EREC attempts to reduce the length of the code words in the EREC frame by $C - r$ to allow the last code word to fit (i.e. make $r = C$). If, on the other hand, it is less ‘expensive’ to push the last code word into a different EREC frame, CA-EREC attempts to fill up the EREC frame by increasing the length of the code words currently in the EREC frame by r (i.e. make $r = 0$).

If CA-EREC needs to increase the lengths of the code words it does this by looking at the effect of incrementing each one by 1 and changing the coefficients with a value of 0 to have

a value of 1. If the code words need to have their overall length reduced, CA-EREC looks at the effect of reducing every AC coefficient by 1. In both cases the most efficient set of adjustments is chosen, that is the set of adjustments which provides the required length change while introducing the least amount of distortion. CA-EREC only attempts to adjust the AC coefficients, as these coefficients are not differentially coded, hence the distortion introduced does not propagate throughout the image. This process is depicted in Figure 7.

Fig. 7. Steps taken by the Code Aligned EREC transcoding scheme. In above example, the code '11' in (b) might represent a value of 9, say, and '011' in (c) might represent a value of 10. Substitution of '011' for '11' introduces some distortion into the image, but allows the codes to align with the EREC packet boundary.

Returning once again to Figure 3(c), should an error occur in EREC frame number 5, some or all of the frame will be discarded, as before. However, because the codes which make up Block 5 have been adjusted to ensure that the end of EREC frame 5 coincides with the end of a valid VLC, when Block 4 has been read out fully from EREC frame 4, the end of Block 5 will be recovered as we now know that it starts with a valid code word. This enables the remains of Block 6 to be read out from EREC frame 4 and 3 and used in the subsequent decoding process. Thus this new scheme *allows uncorrupted information to be recovered from all of the valid EREC frames.*

IV. RESULTS

It is, of course, vital that the distortions introduced by this method are not unacceptable. Figures 9, 10, 11 show an original frame from the 'Foreman' sequence, the result of decoding this frame with normal MPEG-4 quantisation and the result of decoding the frame when normal MPEG-4 quantisation is further modified by the scheme outlined above. In this instance the 396 macroblocks² that compose this CIF image are fitted into 132 EREC frames. As can be seen the distortions are imperceptible. Figure 12 shows the distortions introduced by

²A macroblock is a 16 pixel $\times$ 16 pixel section of the image.

Sequence:	SNR Normal Decode:	SNR with CA-EREC decode
Mom-Kid	27.72dB	27.56dB
Caltrain	30.51dB	30.32dB
Coastguard	32.48dB	32.21dB

Fig. 8. SNR for different sequences comparing normal MPEG-4 decode with decoding of a stream transcoded by CA-EREC.

the scheme with respect to the standard decoded image. If the Signal to Noise ratio is defined as:

$$SNR = 20 \log_{10} \frac{\sum I_1}{\sum |I_1 - I_2|}$$

where I_1 is the original image and I_2 is the representation of this image, then as a quantitative comparison, the Signal to Noise ratio (SNR) of the MPEG-4 coded image is 31.47dB and the SNR of the MPEG-4 coded image, transcoded by CA-EREC is 31.20dB, a marginal decrease. Table 8 shows some other SNR comparisons with different sequences.

Fig. 9. Original 'Foreman' image.

To conclude the example of Figure 3(c), where CA-EREC results in recovery of the end of one of the EREC frames, which corresponds to one sixth of the entire image, Figure 13 shows how the loss of half of the AC coefficients affects picture quality in the top third of the decoded 'Foreman' image and compares this with the improvement if this loss can be limited to just one sixth of the image.

V. CONCLUSIONS

As can be seen, this method overcomes one of the important limitations of the EREC scheme without adding any additional redundancy, or introducing significant distortion to the image.

Fig. 10. Decoded 'Foreman' image.

Fig. 12. Difference between the decoded 'Foreman' image and the transcoded version (Scaled by a factor of 20).

Fig. 11. Modified 'Foreman' image.

Fig. 13. The left picture shows the loss of half of the AC coefficients in top third of 'Foreman' image. The blurring this causes is particularly apparent in the 'Siemens' logo. The right picture shows the loss of half of the AC coefficients in just one sixth of the 'Foreman' image.

It is thus a useful addition to the EREC algorithm, which is well recognised as being an important tool for error resilient coding of video data.

REFERENCES

- [1] A. C. Kokaram, *Motion Picture Restoration: Digital Algorithms for Artefact Suppression in Degraded Motion Picture Film and Video*, Springer Verlag ISBN 3-540-76040-7, 1998.
- [2] A. Kokaram and L. Mac Manus, "Unified image/bitstream analysis for the robust decoding of compressed video sequences for wireless networks.," *Eusipco 2000, Tampere, Finland*, September 4-8th 2000.
- [3] Editors: L. Chiariglione, M. Decina, F. G. B. De Natale, and D. D. Giusto, "Special issue on packet video communications.," *Signal Processing, Image Communication*, vol. 16, no. 8, pp. 701-825, May 2001.
- [4] Editors: A. Katsaggelos, "Multimedia communications over networks.," *IEEE Signal Processing magazine*, vol. 17, no. 4, July 2000.
- [5] Editors: C. W. Chen, P. Cosman, N. Kingsbury, J. Liang, and J. W. Modestino, "Special issue on error resilient image and video transmission.," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 809-1144, June 2000.
- [6] L. Doyle, A. Kokaram, and D. O'Mahony, "Error-resilience in multimedia applications over ad-hoc networks.," May 7-11 2001.
- [7] N. G. Kingsbury and D. W. Redmill, "The erc: An error-resilient technique for coding variable length blocks of data.," *IEEE Transactions on Image Processing*, no. 5, pp. 565-574, 1996.
- [8] D. W. Redmill, *Image and Video Coding for Noisy Channels.*, Ph.D. thesis, University of Cambridge, England, 1994.
- [9] H.-C. Shyu and J.-J. Leou, "Detection and concealment of transmission errors in MPEG2 images - a genetic algorithm approach.," *IEEE Transactions on CCts and Sys. for Video Technology*, vol. 9, no. 6, pp. 937-948, September 1999.
- [10] John G. Proakis, *Digital Communications*, McGraw-Hill, ISBN 0-0723-2111-3, 2000.
- [11] ISO/IEC, "ISO/IEC 14496-2," w2502, 1999.
- [12] Steven J. Whitehouse, *Error Resilient Image Compression.*, Ph.D. thesis, University of Cambridge, England, 2000.
- [13] Robert Swann, *MPEG-2 Coding for Noisy Channels.*, Ph.D. thesis, University of Cambridge, England, 1998.