An Effective Error-Resilient Coding for H.264 Video

YUK YING CHUNG, LEO LI FAN CHEN, ZENG ZHENG, XIAOMING CHEN
School of Information Technologies, University of Sydney, NSW 2006, AUSTRALIA

Abstract: - The transmission errors in H.264 may propagate in the temporal direction. Errors in intra-coded picture (I-frame) will propagate into the associated P-B frame if they are in the same Group Of Picture (GOP). Therefore it is important to detect errors in the I-frame rather than in the P-B frame. This paper proposes and demonstrates an effective technique of Error-Resilient Coding based on bit-error detection and Directional Intra-Frame Concealment (DIFC) for H.264 video. The bit error detection is derived from multiblock checksum, chain coverage and remainder coding. DIFC takes advantage of flexible block sizes to deal with detailed movement areas and employs object edge detection to improve the accuracy of spatial interpolation. The results showed that the proposed directional intra-frame concealment has a better performance than the weighted pixel interpolation in H.264 software.

Key-Words: - H.264 video, bit-error detection, error concealment, error-resilient coding, video compression

1 Introduction

Video compression techniques [1] explore the temporal, spatial and coding redundancies within the video sequence and offer solutions to reduce these redundancies. H.264/AVC [1]-[3] is one of the popular techniques and it has been widely used in HDTV broadcasting, streaming multimedia and video conferencing. Due to the inherent problems of all communication systems, video data may be lost or altered due to the noise during transmission over wire or wireless networks. Random Bit Error (RBE) and Erasure Error (EE) [2][3] are two types of errors which always occur during video transmission.

The transmission errors in H.264 may propagate in the temporal direction. Errors in intra-coded picture (I-frame) will propagate into the associated P-B frame if they are in the same Group Of Picture (GOP). Therefore it is important to detect errors in the I-frame rather than in the P-B frame. H.264 video coding has adapted some error control algorithms such as flexible macroblock ordering and multiple reference frames. They are focused on the source encoder which will bring higher complexity and delay to the video coding. Automatic Retransmission Request (ARQ) and Forward Error control (FEC) [4] are the commonly used methods to address channel errors but they both have limitations. ARQ retransmission is not suitable for real time applications such as video conferencing. FEC reduces the channel capacity and it has error recovery ability limitations.

This paper proposes a new Error Detection (ED) algorithm based on multi-block checksum, chain coverage and remainder coding. This ED is based on the information hiding technique. It can be divided into two parts: encoder and decoder. The encoder performs data embedding process. The decoder performs error detection and concealment schemes. Fig. 1 shows the block diagram of the encoder and decoder for the proposed H.264 error resilient coding.

According to the encoder in Fig. 1, the embedded information in the Multi-Block Checksum (MBC) is from the chain blocks. The embedding host is the Least Signification Bit (LSB) in the blocks chosen by a Threshold value T. From the decoder, the bit-stream is extracted from the NAL (network Adaptation Layer). Packets and embedded bit would be read by an error-detector function from the LSB of the arrival block before they can be sent to the inverse quantization and inverse DCT transform. The extracted message will be compared with the information that collected from the related blocks by the same coverage algorithm. If these two messages are the same, all the blocks covered by this check data will be marked as non-error blocks. Otherwise, they will be marked as error blocks and sent to the Directional Intra-Frame Concealment (DIFC) to recover the error blocks. The results showed that the proposed DIFC has a better performance than the weighted pixel interpolation in H.264 software.

In this paper Section 2 explains the details of the new proposed bit error detection algorithm. Section 3 is the description of the proposed directional intra-frame error concealment algorithm. Sections 4 and
provide the testing results and conclusion respectively.

2 Proposed Bit Error Detection Algorithm

The proposed bit error detection algorithm requires modification in H.264 coded video data at both of the encoder and decoder side. The encoder performs the data embedding scheme which will embed the value of checksum. The decoder performs the error detection by comparing the received checksum and the calculated checksum. In this paper we propose three algorithms namely Multi-Block Checksum (MBC), Chain Coverage (CC) and Remainder Coding (RC). They are used to determine what type of information, the location, and the method to be chosen to embed the error checking information.

![Fig. 1 Encoder and decoder for the proposed error-resilient system](image)

In the proposed Error Detection (ED) algorithm, we have developed a new Multi-Block Checksum (MBC) algorithm using the sum of embedding. In this new developed algorithm, a Threshold value \( T \) is employed and this value decides which block to provide information for embedding and in which block to embed the information. This information is called coverage information since it covers more than one block. The parent block embeds the coverage information. The children blocks provide information. One parent block and a number of children blocks together form a Coverage Chain (CC). CC is used to embed the information and detect the message. Parity checking is widely used in error detection. In the proposed ED algorithm, the remainder value instead of parity bit is used to embed information into the MBC. The results show that it can achieve better performance in the error detection.

2.1 Remainder coding

Parity bit (odd/even) coding is the simplest method of error checking but many errors could not be detected by the odd/even rule. Therefore we have proposed the remainder coding which embeds the remainder value instead of the parity bit. Fig. 2 shows an example when the remainder value is equal to three.

![Fig. 2 Example of Remainder Coding](image)

2.2 Multi-Block Checksum

In this paper we propose Multi-Block Checksum (MBC) for the message of coverage. MBC is calculated from the portion-sum of one T-block and the sum of other NT-block or portion-sums of other NT-blocks. The T-block is the block that has more than \( T \) non-zero coefficients. \( T \) is the threshold value to indicate the non-zero number as a filter. The blocks with less than \( T \) non-zero coefficients are called NT-block. The sum of a block is the sum of all coefficients. The portion-sum of the T-block is the sum of all non-zero coefficients. In H.264 coding all blocks have deleted those zero coefficients after the Run-Length coding (RLC). Therefore the sum of a NT-block is the sum of all
non-zero coefficients; and the portion-sum of a T-block is the sum of all RLC coded array coefficients. The parent block embeds coverage information. Other children blocks provide sums or portion-sums information. One parent block with several children blocks form an array block.

The parent block is the T-block. Children blocks can be either T-blocks or NT-blocks. Fig. 3 shows an example of MBC.

![An example of Multi-Block Checksum (MBC)](image)

2.3 Chain Coverage (CC)

In this paper, we propose a new Chain Coverage (CC) algorithm to choose in which host to embed the information. In H.264 coded video, not all the AC blocks are suitable to embed the information. For example, if the block does not have any non-zero coefficients, the video visual quality will become worse if we change any value of the AC coefficient. Therefore we employ a threshold value \( T \) to measure which block has the possibility to embed the information and which block should not use to embed any information. The Threshold value \( T \) is a prefixed integer value. The blocks with the non-zero coefficient number greater than \( T \) will be chosen as the host block to embed MBC. It can be divided into four parts because the Threshold Capability (TC) is the number of T-blocks in an 8x8 subblock. A TC of an 8x8 subblock has the value one which means there is only one T-block and others are NT-blocks. There are four 4x4 blocks in each 8x8 subblock so we have a TC value from 0 to 4. Fig. 4 shows an example when TC equals one.

![An example when TC=1](image)

2.4 Threshold Embedding

Bartolin, Manetti, Piva and Barni [6] have developed a new algorithm for embedding information which uses a threshold value \( T \) in each 8x8 subblock for H.263 video. Only the coefficients that have a number of non-zero coefficients larger than \( T \) would be chosen to embed the information. In their experiments, ten was chosen for the value of \( T \). Results showed that the error detection rate had been improved by 2 to 10 times when compared with other algorithms.

A threshold can be used to detect the Bit-Errors but there are some limitation in this algorithm. The error detection capability of threshold embedding is limited since it does not embed in every block. In H.264 video, blocks are T-blocks as there are usually less than three non-zero coefficients in a block. It will be impossible to embed any message into them if the threshold value for \( T \) is equal to ten. Fig. 5 shows an example of AC coefficient when the threshold value \( T \) equals ten.

In this paper threshold embedding algorithm is compared with the proposed Chain coverage (CC) algorithm. Sections 4.1 to 4.3 show the comparison results of these two algorithms. Section 4.1 compares the PSNR quality of the embedded video. Section 4.2 compares the bit error detection ability. Section 4.3 shows the improvement of using remainder coding.

![AC threshold (T=10)](image)

3 Error Concealment

3.1 Error Concealment in H.264

Weighted pixel interpolation [5] is the Error Concealment algorithm used in H.264 standard video coding. It estimates the value of missing pixel by using pixels from the one pixel wide boundary on the horizontal and vertical neighboring macroblocks as shown in Fig. 6. Missing pixel values are calculated by adding together the multiplication of each pixel that is used to conceal the missing pixels by a weight factor, and then divide the result by the sum of total distance. The value of the weight factor...
depends on how far away it is from the missing pixel.

![Fig. 6 Weighted Pixel Interpolation Error Concealment algorithm in H.264 video coding](image)

### 3.2 Proposed Directional Intra Frame Concealment (DIFC) Algorithm

The Weighted Pixel Interpolation Concealment [5] (WPIC) in H.264 can perform well only if we apply it to the smooth surface, or the patterns in H.264 coded video only move in either vertical or horizontal direction. Reduced performance starts to appear when it applied to an area where the patterns movement is not in a vertical or horizontal direction. Fig. 7 illustrates how a missing macroblock happen in WPIC.

![Fig. 7 Diagonal pattern producing a missing macroblock](image)

If the weighted pixel interpolation method was applied to the missing macroblock in Fig. 7, the upper half of the missing macroblock would be in grayish colour, which will be different from the original pattern. Therefore we should not constrain the use of pixels only from the horizontal and vertical directions. We should consider all the direction of the edge in the pattern. We can take the picture of Fig. 7 as an example. The edge, or the contrast between the black section and the rest of the image is from top right diagonally down to bottom left. We can use this direction to replace the horizontal and vertical direction found in H.264 to conceal the missing macroblock.

In this paper, we propose the Directional Intra-Frame concealment (DIFC). DIFC uses Prewitt Edge Detector to detect the magnitude and the orientation of the edge. The predetermined vertical digital mask:

\[
G_y = \begin{bmatrix}
-1 & 0 & 1 \\
-1 & 0 & 1 \\
-1 & 0 & 1 
\end{bmatrix}
\]

and the horizontal digital mask:

\[
G_x = \begin{bmatrix}
-1 & 1 & 1 \\
0 & 0 & 0 \\
-1 & -1 & -1 
\end{bmatrix}
\]

are applied to each pixel in the neighbouring macroblocks. Its magnitude and direction are given by:

\[
M = \sqrt{G_x^2 + G_y^2}; \quad \theta(x, y) = \arctan\left(\frac{G_y}{G_x}\right)
\]

The circle is classified into 16 directions with 22.5 degree separated in each direction. Each direction comes with a counter to record the total magnitude of this particular direction. When the direction of a particular 3x3 block has been determined, its magnitude will be added into the counter corresponding to that direction. The direction with the largest magnitude value obtained from the direct neighbours of the missing macroblock will be used to conceal the missing macroblock. The weighted interpolation in the selected direction is then taken in each missing pixel as shown in Fig. 8.

In the proposed DIFC algorithm, the new direction obtained from the above method will be taken into account instead of calculating the missing pixel value (black square in Fig. 8) with pixels in the horizontal and vertical directions. The direction showed in Fig. 8 was derived from the image in Fig. 7. Pixels in the diagonal direction will be used for concealment in DIFC. The concealed block in DIFC will be consistent with the edge trend.

![Fig. 8 The proposed DIFC](image)

### 4 Testing Results

Section 4.1 to 4.3 describes the bit error detection tests which compare the Chain Coverage and Threshold Error Detection algorithms. The tests were done using “Trevor” video sequence in QCIF format with 149 frames. The file size for each frame is 176 x 144 pixels.

The concealment test in section 4.4 was done using “Foreman” video sequence. It tested the H.264 Weighted Pixel Interpolation Concealment (WPIC) and the proposed Directional Intra-Frame Concealment (DIFC).

#### 4.1 Comparison of PSNR ratio for Chain Coverage and Threshold Error Detection Algorithms

Embedding noise will affect the performance of PSNR in video files. The PSNR results for Chain Coverage and Threshold embedding in Luma Intra 4x4 blocks are shown in Fig. 9. Fig. 10 provides the illustrated comparison for PSNR in graph format.
As shown in Fig. 10, the PSNR of embedded files are always lower than that of a non-embedded file. The decrease of PSNR could be as high as 15 dB when the Threshold is 4 and remainder is 6. A better PSNR result can be obtained by increasing the threshold value or by reducing the divisor (RE). The embedded PSNR result is acceptable with a threshold value greater than 6 and divisor value less than 5.

<table>
<thead>
<tr>
<th>Threshold Value</th>
<th>PSNR (dB)</th>
<th>PSNR (dB)</th>
<th>PSNR (dB)</th>
<th>PSNR (dB)</th>
<th>PSNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>25.9194</td>
<td>25.2695</td>
<td>22.9492</td>
<td>22.5245</td>
<td>22.3919</td>
</tr>
<tr>
<td>6</td>
<td>25.9194</td>
<td>25.2695</td>
<td>22.9492</td>
<td>22.5245</td>
<td>22.3919</td>
</tr>
<tr>
<td>7</td>
<td>25.9194</td>
<td>25.2695</td>
<td>22.9492</td>
<td>22.5245</td>
<td>22.3919</td>
</tr>
<tr>
<td>9</td>
<td>25.9194</td>
<td>25.2695</td>
<td>22.9492</td>
<td>22.5245</td>
<td>22.3919</td>
</tr>
<tr>
<td>11</td>
<td>25.9194</td>
<td>25.2695</td>
<td>22.9492</td>
<td>22.5245</td>
<td>22.3919</td>
</tr>
</tbody>
</table>

Fig. 9 PSNR results for Chain Coverage and Threshold embedding in Luma Intra 4×4 blocks

4.2 Comparison of Detection Ability for Chain Coverage and Threshold Error Detection Algorithms

The bit error detection test was carried out under a Block Noise environment. The video bit stream was coded in Intra 4×4 predicted blocks and followed by both two embedding algorithms: Chain Coverage and Threshold embedding. Noise was added to the Run Length Coefficients of each 4×4 Luma block.

The decoder decodes and detects the bit errors. When an error is found in a block, this block will be marked as an error block. The measurement for these error detection method is not the same: (1) the Threshold Embedding method detects the single block, and the detected block number is the measurement for the detection ability; (2) the Chain Coverage embedding detects the chain, the detected error chain could have more than one error block, so the chain error detection rate is lower than the exact block error detection rate.

In this test, the Chain Detection Rate is used to measure the performance of the Chain Coverage detection ability, although it is lower than the real error detection rate.

A comparison of Chain Detection Rate (CDR) of Chain Coverage and the Block Detection Rate (BDR) of Threshold Embedding is shown in Fig. 11 when error step equal to 1. The detection ability is related to the Threshold value T, the equation for T-block CDR, and T-block BDR as defined below:

\[
CDR = \frac{\text{Amount Detected Error chain}}{\text{Amount Error Blocks}} \times 100\%
\]

\[
BDR = \frac{\text{Amount Detected Error blocks}}{\text{Amount Error Blocks}} \times 100\%
\]

Fig. 11 Bit-error detection ability (Error Step=1)

According to Fig. 11, we found that the value of T-CDR is greater than the value of real detected T-Block Rate. It is because some NT-Block errors have been detected by this method and CDR would smaller than the detection ability as some NT-Block are not covered. We found that in the same embedding algorithm, parity embedding could detect less errors. The detection rate for Chain Coverage is around 70% when the threshold values are equal to 6, 7 and 8. The detection improvement for remainder embedding is more than 30%.

4.3 Detection Improvement by Remainder Coding

Fig. 12 shows the improvement percentage of both Chain Coverage and Threshold Embedding by using remainder coding instead of parity bit when Error Steps equal to 1.

According to Fig. 12, we found that the different error steps, different threshold values and different remainder value RE will affect the bit error detection ability. Changing of RE from 2 to 3 (from parity embedding to remainder coding of mode 3) could lead to around 35% improvement in error detection ability. If we change the value of RE from 2 to 4, the improvement could be as high as 55% (while threshold value equal to 9 and error step equal to 4).
4.4 Directional Intra-Frame Concealment (DIFC)

Testing of the H.264 Weighted Pixel Interpolation Concealment (WPIC) and the proposed Directional Intra-Frame Concealment (DIFC) was undertaken using the “Foreman” video sequence. Fig. 13 shows the results of these two concealment algorithms. According to the results of the DIFC algorithm in Fig. 13, we found that the top left part of the image appears to have vertical directions because there is a high contrast between the black line and the neighbouring pixels at the edge of the image. This produces the high magnitude in the vertical direction. The slope near the hat appears to be more smooth than the WPIC in H.264. From the right hand side picture in Fig. 13, we also found that the proposed DIFC has a higher PSNR and better visual quality than the WPIC.

Threshold = 7
Divisor = 3
Error Coefficient Rate: 1%

<table>
<thead>
<tr>
<th>Weighted Pixel Interpolation</th>
<th>Directional Intra-Frame Concealment</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR: 22.144dB</td>
<td>PSNR: 22.220dB</td>
</tr>
<tr>
<td>PSNR: 18.769dB</td>
<td>PSNR: 18.945dB</td>
</tr>
</tbody>
</table>

Fig. 13 Results in comparison of WPIC and DIFC

5 Conclusion

The proposed Chain Coverage Bit Error Detection algorithm can improve the error detection ability by 70% when compared with Threshold embedding. The Remainder coding has a higher error detection rate than odd/even parity embedding. It can have an average rate of 32% and a rate of up to 50% better than parity coding.

In this paper, an effective error resilient coding based on bit-error detection and Directional Intra-Frame Concealment (DIFC) is proposed. The bit error detection based on multi-block sum, chain coverage and remainder coding. The results in section 4 showed that the proposed bit-error detection can have better error detection capacity than the parity coding.

The proposed DIFC takes the advantage of flexible block sizes to deal with detailed movement areas and uses object edge detection to improve the accuracy of spatial interpolation. The testing results showed that it can have better performance than the weighted pixel interpolation in H.264 reference software.

References