REFERENCE PICTURE SELECTION IN AN ALREADY MPEG ENCODED BITSTREAM

Hoi-Kin Cheung, Yui-Lam Chan, Wan-Chi Siu

Centre for Multimedia Signal Processing
Department of Electronic and Information Engineering
The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

ABSTRACT

Reference picture selection (RPS) is the most common error resilience method for robust transmission over lossy networks. However, RPS has been studied for use in real-time encoding, but has not been examined in transmitting an already encoded MPEG bitstream. In this paper, we propose a compressed-domain approach for the efficient implementation of RPS in the pre-encoded MPEG bitstream with minimum requirement on the server complexity. In the proposed algorithm, a novel macroblock-based algorithm is used to adaptively select the necessary macroblocks, manipulate them in compressed-domain and send the processed macroblocks to the receiver. Experimental results show that, as compared to the original RPS, the new algorithm reduces the required server complexity significantly.

1. INTRODUCTION

Recent advances in compression and networking technologies have resulted in a rapid growth in transmitting high-quality video over various networks such as the Internet and wireless networks. Along with this explosive growth, the need for reliable video transmission is very challeng-ing. However, state-of-the-art video coding standards [1-2] such as MPEG-4 and H.263 are extremely vulnerable to transmission errors. The reason behind is that these stan-dard employ the motion estimation and compensation techniques. They use the previous encoded and reconst-ructed video frame to predict the next frame. Therefore, the loss of information in one frame has considerable impact on the reconstruction of its succeeding frames. As a result, temporal error propagation is a typical trans-mission error effect of the following frames.

In MPEG-4 [1] and H.263 [2], they incorporated two feedback-based techniques: error tracking and reference picture selection. These techniques utilize a feedback channel from the receiver to the server to indicate which parts of the decoded video signal were corrupted and had to be recovered. The error tracking technique makes use of intra-mode coding of macroblocks to stop temporal error propagation but it restricts to severely affected image regions only [3]. Since intra-mode macroblocks are less efficient than inter-mode macroblocks, the error tracking technique deceases the coding efficiency. In order to stop temporal error propagation while maintaining the best coding efficiency, the reference picture selection technique adopted in MPEG-4 and H.263 [4-6] allows the encoder to select one of several previously decoded frames as a reference for motion estimation. In this approach, an acknowledgment sent from the receiver to the server signifies that a given frame has been damaged by transmission errors. Thus, the encoder of the server will not use this frame for future prediction, instead employ a different and uncorrupted reference frame. RPS can ensure to form temporal predictions using a reference frame that is known to be correctly decoded by the receiver. This can limit error propagation without incurring the severe bit-rate overhead of a new intra-mode macroblock. However, RPS has been studied for use in real-time encoding, but has not been examined in transmitting an already encoded bitstream over error-prone networks. Such applications include video on demand.

In this paper, we provide a computationally efficient solution to perform RPS in the pre-encoded video bitstream with minimum requirement on the server complexity. The algorithm developed in this paper is mainly on the DCT-domain in order to alleviate the computational requirement of the server and the quality degradation of the reconstructed video arising from re-encoding. The organization of this paper is as follows. Section 2 of this paper presents a study of impact of adopting RPS in the pre-encoded video on server complexity. The proposed RPS algorithm in the pre-encoded video is then described in Section 3. Simulation results are presented in Section 4. Finally, some concluding remarks are given in Section 5.

2. IMPACTS OF USING RPS IN THE PRE-ENCODED VIDEO ON THE VIDEO SERVER

0-7803-9134-9/05/$20.00 ©2005 IEEE

Now we show the impact of adopting RPS in the pre-encoded video on the server complexity. Consider the case as shown in Figure 1 in which the encoder of the server knows that transmission errors have occurred in frame *n-1* through the feedback channel. Since frame *n-1* is required to act as the reference frame for the reconstruction of the next transmitted frame, frame *n*, in the pre-encoded bitstream, the quantized DCT coefficients of residual signal of frame *n* are no longer valid because they refer to the frame *n-1* which has been corrupted. In order to stop error propagation, in RPS, the last frame available without errors at the decoder should be selected as a reference frame, that is, frame *n-2* for frame *n*. The server then needs to decode frame *0* to frame *n* and perform re-encoding of frame *n* with frame *n-2* as a reference. Thus, straightforward implementation of adopting RPS requires much higher server complexity for these decoding and re-encoding processes. Besides, the video quality suffers from its re-encoding process, which introduces additional degradation. Since frame *n* is used as a reference frame for the following transmitted P-frames, quality degradation propagates to later frames and it means that the reconstructed quality of the video sequence is degraded significantly when RPS is operated in an already encoded video.

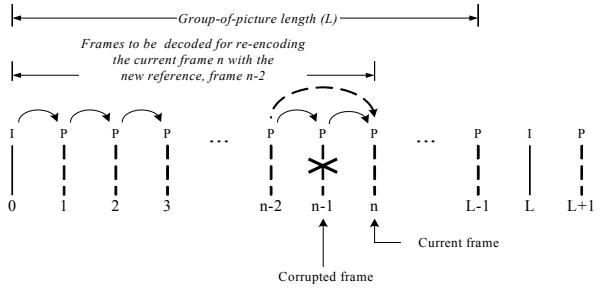

Figure 1. Example of RPS used in a pre-encoded bitstream.

3. THE PROPOSED ALGORITHM

In this section, we propose a macroblock-based algorithm to allow the encoder to select one of several previously and correctly decoded frames of the decoder as a reference frame for re-encoding. It is designed to support RPS in a pre-encoded video. For our new macroblock-based algorithm, two types of macroblock are now defined. For illustration, we use the example in Figure 1 again. Let us assume that the server receives an acknowledgment signifying frame *n-1* has been damaged by transmission errors. The server needs to re-encode frame *n* with a new reference, frame *n-2*. The situation in macroblock level is depicted in Figure 2. We assume that $MB^n_{(k,l)}$ represents the macroblock at the k^{th} row and l^{th} column of frame *n*. $MB^n_{(k,l)}$ is defined as a non-motion compensation (non-MC) macroblock if the macroblock in frame *n* having the same spatial position of $MB^{n-1}_{(k,l)}$, i.e. $MB^n_{(k,l)}$, is coded without motion compensation. Otherwise, it is defined as a motion compensated (MC) macroblock. For example, in Figure 2, since the motion vector of $MB^n_{(0,1)}$, $mv^n_{(0,1)}$, is zero, it means that $MB^n_{(0,1)}$ is a non-MC macroblock. On the other hand, $MB^n_{(1,1)}$ is categorized as a MC macroblock. In this paper, our scheme works at the level of macroblocks. Since the server will process non-MC macroblocks in the compressed-domain, a complete decoding and encoding are not required for those non-MC macroblocks such that the increasing in the computational burden of the server and the re-encoding are greatly reduced.

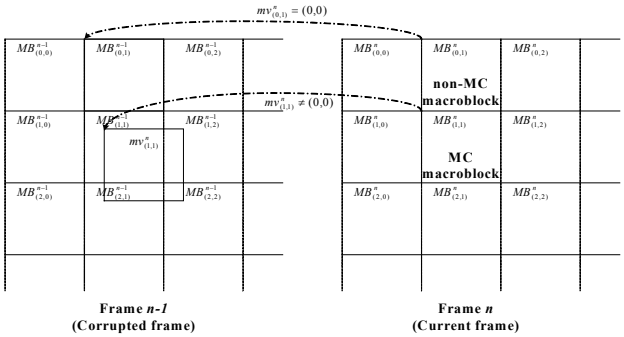

Figure 2. Definition of non-MC and MC macroblocks

In Figure 3, a situation in which frame *n-1* is corrupted is illustrated. We assume that $MB^n_{(k,l)}$ represents the current non-MC macroblock and $MB^{n-1}_{(k,l)}$ represents the best matching macroblock with $MB^n_{(k,l)}$. Since $MB^n_{(k,l)}$ is coded without motion compensation, the spatial position of $MB^{n-1}_{(k,l)}$ is the same as that of $MC\left(MB^n_{(k,l)}, mv^n_{(k,l)}\right)$, and $MC\left(MB^{n-1}_{(k,l)}, mv^{n-1}_{(k,l)}\right)$ represents the best matching macroblock with $MB^{n-1}_{(k,l)}$, where $MC()$ is the motion-compensation operator. Since frame *n-1* is damaged, for $MB^n_{(k,l)}$, we need to compute motion vector $\overline{mv}^n_{(k,l)}$ and prediction errors in the DCT-domain, $Q[DCT(\overline{e}^n_{(k,l)})]$, by using frame *n-2* as a reference. Since $mv^n_{(k,l)} = (0,0)$, then

$$\overline{mv}^n_{(k,l)} = mv^{n-1}_{(k,l)} \quad (1)$$

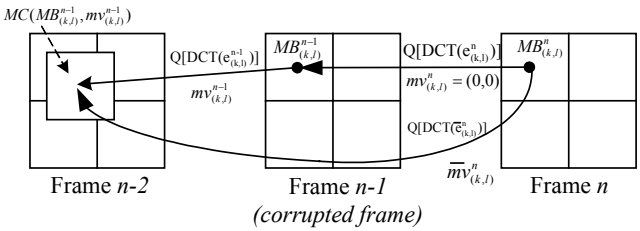

Figure 3. RPS in a non-MC macroblock.

The reconstructed macroblocks $MB^n_{(k,l)}$ and $MB^{n-1}_{(k,l)}$ are given by

$$MB^n_{(k,l)} = MB^{n-1}_{(k,l)} + e^n_{(k,l)} \quad (2)$$

and

$$MB_{(k,l)}^{n-1} = MC(MB_{(k,l)}^{n-1}, mv_{(k,l)}^{n-1}) + e_{(k,l)}^{n-1} \quad (3)$$

where $e_{(k,l)}^{n}$ and $e_{(k,l)}^{n-1}$ represent the prediction errors of $MB_{(k,l)}^{n}$ and $MB_{(k,l)}^{n-1}$ respectively.

Using (2) and (3), the new prediction errors between $MB_{(k,l)}^{n}$ and $MC(MB_{(k,l)}^{n-1}, mv_{(k,l)}^{n-1})$, $\bar{e}_{(k,l)}^{n}$, can be written as

$$\bar{e}_{(k,l)}^{n} = e_{(k,l)}^{n} + e_{(k,l)}^{n-1} \quad (4)$$

To generate MPEG bitstream, $\bar{e}_{(k,l)}^{n}$ is transformed and then quantized. Hence, we will discuss how to compute $Q(DCT(\bar{e}_{(k,l)}^{n}))$ in compressed-domain.

In the pre-encoded MPEG bitstream, the quantized DCT coefficient of $e_{(k,l)}^{n}$ is available. Let it be $E_{(k,l)}^{n}$, that is, $e_{(k,l)}^{n} = DCT^{-1}\left(Q^{-1}\left(E_{(k,l)}^{n}\right)\right)$. (4) becomes

$$\bar{e}_{(k,l)}^{n} = DCT^{-1}(Q^{-1}(E_{(k,l)}^{n})) + DCT^{-1}(Q^{-1}(E_{(k,l)}^{n-1})) \quad (5)$$

By applying DCT to (5) and taking into account the linearity of DCT, (5) can be written as

$$DCT(\bar{e}_{(k,l)}^{n}) = Q^{-1}(E_{(k,l)}^{n}) + Q^{-1}(E_{(k,l)}^{n-1}) \quad (6)$$

Note that quantization is not a linear operation. In the MPEG-4 standard, dequantization is given by

$$Q^{-1}(x) = \begin{cases} 0 & if \quad x = 0 \\ ((2 \times x + k) \times Qm \times Qf)/16 & otherwise \end{cases} \quad (7)$$

where $k = sign(x)$ (dead zone is enabled) if x≠0, otherwise k is equal to *0*, Qm is the element of the quantization matrix and Qf is the quantization factor.

Substituting (7) into (6), we obtain

$$DCT(\bar{e}_{(k,l)}^{n}) = 2(E_{(k,l)}^{n} + E_{(k,l)}^{n-1}) \times Qm \times Qf/16 + [sign(E_{(k,l)}^{n}) + sign(E_{(k,l)}^{n-1})] \times Qm \times Qf/16 \quad (8)$$

These are the DCT coefficients we want to reconstruct in the receiver. Equation (8) gives us a hint of obtaining $Q(DCT(\bar{e}_{(k,l)}^{n}))$ by directly adding $E_{(k,l)}^{n}$ and $E_{(k,l)}^{n-1}$. That is,

$$Q(DCT(\bar{e}_{(k,l)}^{n})) = E_{(k,l)}^{n} + E_{(k,l)}^{n-1} \quad (9)$$

If $Q(DCT(\bar{e}_{(k,l)}^{n}))$ undergoes decoding in the receiver, we get

$$DCT(\bar{e}_{(k,l)}^{n}) = 2(E_{(k,l)}^{n} + E_{(k,l)}^{n-1}) \times Qm \times Qf/16 + [sign(E_{(k,l)}^{n} + E_{(k,l)}^{n-1})] \times Qm \times Qf/16 \quad (10)$$

Comparing (8) and (9), they are not equal since $sign(E_{(k,l)}^{n} + E_{(k,l)}^{n-1}) \neq sign(E_{(k,l)}^{n}) + sign(E_{(k,l)}^{n-1})$ when $E_{(k,l)}^{n}$ and $E_{(k,l)}^{n-1}$ are in opposite sign with different magnitude. In this case, $sign(E_{(k,l)}^{n}) + sign(E_{(k,l)}^{n-1})$ is equal to zero whereas $sign(E_{(k,l)}^{n} + E_{(k,l)}^{n-1})$ is either equal to 1 or –1. Thus, the server should compensate this misalignment error by adjusting $\pm Qm \times Qf/16$ which is half of the quantization factor. That means it cannot be represented in the MPEG bitstream. One alternative to deal with this problem is to double the term of $E_{(k,l)}^{n} + E_{(k,l)}^{n-1}$ prior to the compensation of misalignment error. In other words, the server computes the final expression of $Q(DCT(\bar{e}_{(k,l)}^{n}))$,

$$Q(DCT(\bar{e}_{(k,l)}^{n})) = 2(E_{(k,l)}^{n} + E_{(k,l)}^{n-1}) + sign(E_{(k,l)}^{n}) + sign(E_{(k,l)}^{n-1}) \quad (11)$$

In the receiver, the dead zone of the dequantization process is disabled and modified as shown in (12) during decoding the frame with a new reference.

$$Q^{-1}(x) = (x \times Qm \times Qf)/16 \quad (12)$$

For the proposed algorithm, only a small change is needed for the receiver to equip with the dequantization as shown in (12). It is exactly the MPEG dequantization equation for decoding intra-block with halved quantization factor.

For MC macroblocks, the above technique cannot be employed since $MB_{(k,l)}^{n-1}$ is not on a macroblock boundary,. In other words, $E_{(k,l)}^{n-1}$ is not available from the bitstream. To reconstruct MC macroblocks of frame *n* directly, the server will examine the motion vectors in the MPEG bitstream and all the related macroblocks from the previous nearest I-frame to frame *n-1* should be decoding and perform re-encoding for those MC macroblocks with frame *n-2* as the reference.

4. SIMULATION RESULTS

In this section, simulations have been performed to evaluate the overall efficiency of the original RPS and proposed RPS algorithms in an already MPEG encoded bitstream. Three video sequences, "Salesman", "Foreman", "Mother and Daughter" were used for the evaluation. We have simulated the situation of the I-P structure for L=15. For all testing sequences, the frame-rate of the video stream was 30 frames/s.

The detailed comparison of the average number of macroblocks to be decoded by the server is tabulated in Table 1. The average number of macroblocks decoded for decoding is directly proportional to the server complexity. In Table 1, we show that the proposed RPS outperforms the original one in all sequences. The results are more significant for the sequences "Mother and Daughter" and "Salesman". The saving in macroblocks to be decoded by the server is from 40-75% for those sequences. It is due to the reason that those sequences contain more non-MC macroblocks in which the compressed-domain technique

can be employed. For sequence containing high motion activities such as "Foreman", there still has about 16% saving. Figure 4 shows the comparisons of the number of macroblocks to be decoded by the server of the proposed and original algorithms for the "Mother and Daughter" sequence. The x-axis is the corrupted frame number and the y-axis is the required number of to be decoded macroblocks by the server. It is obvious that the proposed method can achieve significant performance improvement in terms of the server complexity. Note that, the non-MC macroblock is not necessary to perform re-encoding which means it has further saving. Since re-encoding for non-MC macroblocks is not needed, there is significant PSNR improvement of the succeeding frames after the corrupted frames for all video sequences, as depicted in Table 2.

Table 1. Saving of macroblocks to be decoded by the server of the proposed RPS as compared with the original RPS.

Sequences	Input bitrate	Savings (%)
Salesman	1.14 Mbps	40.6
	686 kbps	43.8
	256 kbps	51.5
Foreman	1.12 Mbps	16.7
	712 kbps	15.7
	354 kbps	15.8
Mother and daughter	537 kbps	74.7
	249 kbps	74.1
	112 kbps	69.9

Table 2. Average PSNR of the succeeding frames after the corrupted frames.

Sequences	Input bitrate	Original RPS (dB)	Proposed RPS (dB)
Salesman	1.14 Mbps	42.5	42.8
	686 kbps	39.9	40.1
	256 kbps	36.8	37.0
Foreman	1.12 Mbps	41.5	42.2
	712 kbps	38.8	39.4
	354 kbps	35.9	36.3
Mother and daughter	537 kbps	44.2	44.4
	249 kbps	42.4	42.7
	112 kbps	39.9	40.0

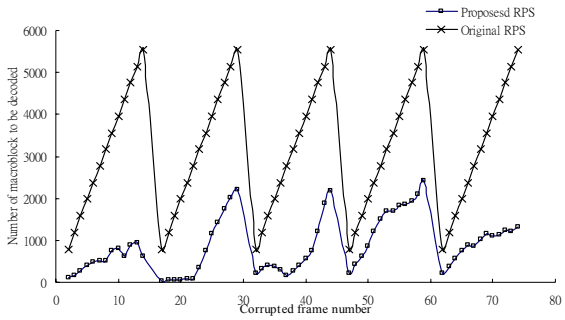

Figure 4. Number of macroblocks to be decoded by the server for "Mother and daughter" sequence.

5. CONCLUSION

In this paper, we have proposed an efficient RPS algorithm for an already MPEG encoded bitstream. The proposed algorithm is motivated by the center-biased motion vector distribution characteristics of real-world video sequences. With the motion information, the video server organizes the macroblocks in the frame with a new reference frame into two categories. Then it selects the necessary macroblocks adaptively, processes them in the compressed-domain and sends the processed macroblocks to the receiver. Simulation results show that, with our proposed algorithm, RPS can work well in the already encoded MPEG bitstream and minimize the server complexity significantly.

6. ACKNOWLEDGMENT

The work described in this paper is partially supported by the Centre for Multimedia Signal Processing, Department of Electronic and Information Engineering, Hong Kong Polytechnic University and a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (PolyU 5216/03E).

7. REFERENCES

[1] ISO/IEC 14496-2, "Information technology – coding of audio-visual objects: visual," 1998.

[2] ITU-T Recommendation H.263, "Video coding for low bit rate communication," Sept. 1997.

[3] H-J Chiou, Y.-R Lee and C-W Lin, "Error-Resilient transcoding using adaptive intra refresh for video streaming," in Proc. International Symposium on Circuits and Systems, Vol. 3, pp. 777-80, May 2004.

[4] B. Girod and N. Farber. "Feedback-based error control for mobile video transmission," In Proc. IEEE, Vol. 87 , pp.1707 – 1723, Oct. 1999.

[5] S. Fukunaga, T. Nakai, and H. Inoue, "Error resilient video coding by dynamic replacing of reference pictures," in Proc. IEEE Global Telecommunications Conf., Vol. 3, pp. 1503–1508, Nov. 1996.

[6] Y. Tomita, T. Kimura, and T. Ichikawa, "Error resilient modified inter-frame coding system for limited reference picture memories," in Proc. Int. Picture Coding Symposium, pp. 743–748, Sept. 1997.