EFFICIENT CHROMINANCE COMPENSATION FOR MPEG2 TO H.264 TRANSCODING

Qiang Tang, Panos Nasiopoulos, Rabab Ward

Electrical and Computer Engineering
University of British Columbia
{qiangt, panosn, rababw}@ece.ubc.ca

ABSTRACT

Although open-loop transcoding is known as the most computational efficient transcoding structure, it is also known to introduce many distortions in the transcoded video. This paper addresses the chrominance distortions resulting from the open-loop MPEG2 to H.264 transcoding structure and proposes algorithms to compensate for the chrominance distortions. The open-loop structure is replaced by a closed-loop transcoding structure, which provides high-quality video by removing the chrominance distortions, resulting in an average of 6dB picture quality improvement.

Index Terms—Transcoding, MPEG2, H.264, Chrominance interpolation, Re-quantization.

1. INTRODUCTION

MPEG2 is presently widely used in many video applications such as digital TV and DVD. H.264, however, is gaining more popularity due to its high data compression efficiency. In the foreseeable future, these two video coding standards will co-exist for a while. Heterogeneous video transcoding is an efficient way to provide universal media access among different video coding standards [1].

Unlike previous standards, H.264 uses an integer transform, which is different from DCT. Since H.264 transform coefficients are different from MPEG2 coefficients, most transcoding schemes implement the MPEG2 to H.264 conversion in the pixel domain [2]-[3]. Nevertheless, an efficient method that transcodes the DCT coefficients to the H.264 integer transform coefficients has been proposed [4]. Based on this method, one can implement MPEG2 to H.264 transcoding in the transform domain by re-using the MPEG2 DCT coefficients. For the case of P frames, this transcoding re-uses the residue data resulting from the MPEG2 motion compensation. This approach though introduces distortions which are due to the substantial differences between the MPEG2 and H.264 motion compensation processes. Some of the errors introduced are the re-quantization error which is addressed in [5], and the luminance half-pixel interpolation error which is addressed in [6]. A transform domain transcoding scheme which is designed to work for Inter macroblocks (MBs) is presented in [7].

However, there is no published work to this date that addresses the chrominance distortions caused by MPEG2 to H.264 transcoding in the transform domain. These distortions are the results of the re-quantization error and the chrominance interpolation errors. This paper analyzes the chrominance distortions arising in MPEG2 to H.264 transcoding and proposes solutions that compensate for these distortions. Experimental results show that our proposed compensations achieve picture quality very close to that obtained by the cascaded pixel-domain transcoding structure, while they significantly reduce the computational complexity.

2. PROBLEM DESCRIPTION

An open-loop P frame transform domain MPEG2 to H.264 transcoding structure is shown in Fig. 1. This is the most efficient transcoding scheme, since it directly re-uses the MPEG2 P frame residue data during the H.264 encoding stage, i.e., it avoids performing motion compensation again. This procedure, however, introduces many video distortions and, for this reason, it is not commonly used in real life for heterogeneous video transcoding applications. Despite that, this structure is always used as the backbone framework which with the addition of other modules may achieve the desired balance between picture quality and computational complexity.

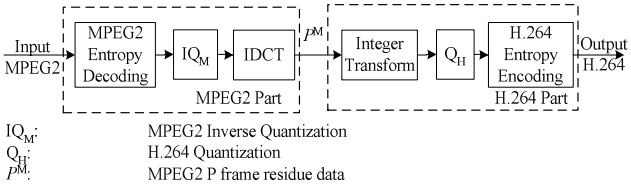

Fig. 1 Open-loop P frame MPEG2 to H.264 transcoding structure

In order to determine how video distortions arise in the open-loop transcoding structure, we have to compare it with the cascaded pixel-domain transcoding scheme which is known to achieve the best video quality. The cascaded pixel-

domain transcoding structure is shown in Fig. 2.

In this structure, a P frame that will be encoded in H.264 is given by:

$$P^{H}(\vec{x}) = F^{Rec}(\vec{x}) - MC_{H}(I^{H}(\vec{x})) = P^{M}(\vec{x}) + MC_{M}(I^{M}(\vec{x})) - MC_{H}(I^{H}(\vec{x})) \quad (1)$$

where MC_M denotes the MPEG2 motion compensation process, MC_H denotes the H.264 motion compensation process, and $\vec{x}$ denotes the 2-D coordinate of one pixel in the video frame.

In the open-loop structure (Fig. 1), the residue data to be encoded in H.264 is still $P^M(\vec{x})$, since H.264 re-uses the MPEG2 residue data. Thus, the difference between $P^H(\vec{x})$ and $P^M(\vec{x})$ will introduce distortions for the transcoded frame (see (1)). These distortions become worse frame by frame until the next I frame arrives.

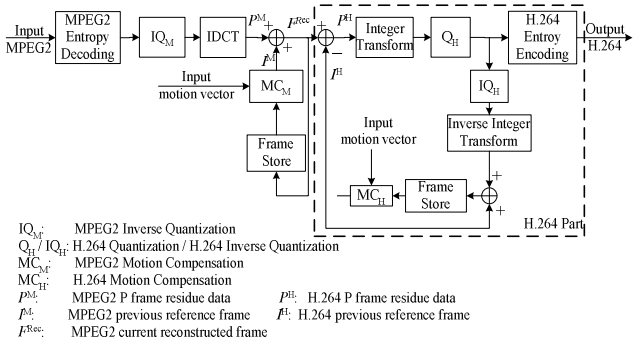

Fig. 2 Cascaded pixel-domain transcoding structure

3. PROPOSED ALGORITHM

As discussed above, the distortions in the open-loop transcoding structure are due to the differences between $P^H(\vec{x})$ and $P^M(\vec{x})$. We observe from equation (1) that the these distortions are equal to the difference between $MC_M(I^M(\vec{x}))$ and $MC_H(I^H(\vec{x}))$. Our objective is to derive an efficient algorithm to compensate for this difference.

The chrominance distortions are results of two errors. One is the re-quantization error [5] and the other is the chrominance interpolation error. The chrominance component is sub-sampled in the vertical and horizontal directions by a ratio of 1:2 as compared to luminance. When the luminance motion vector points to a half pixel, the corresponding chrominance motion vector points to a quarter or three-quarter sub-pixel. The interpolation errors are related to the different interpolation filters used by MPEG2 and H.264 to estimate these sub-pixels.

Since the values of the sub-pixels are not known, the encoder estimates them using interpolation filters. In Fig. 3, $I^M(\vec{x}_A)$, $I^M(\vec{x}_B)$ and $I^H(\vec{x}_A)$, $I^H(\vec{x}_B)$ denote integer position pixels in the MPEG2 reference frame I^M and the H.264 reference frame I^H, respectively. $F^{Rec}(\vec{x})$ denotes the pixel in the current frame which will go through the motion compensation process. $I^M(\vec{x}_{0.25})$, $I^H(\vec{x}_{0.25})$ are the quarter sub-pixels. $I^M(\vec{x}_{0.5})$, $I^H(\vec{x}_{0.5})$ are the half sub-pixels, and $I^M(\vec{x}_{0.75})$, $I^H(\vec{x}_{0.75})$ are the three-quarter sub-pixels.

Based on the values of the chrominance motion vectors, four categories of chrominance distortions may be defined: integer pixel; quarter pixel, half pixel and three-quarter pixel (Table 1). The following subsections describe these four types of chrominance distortions.

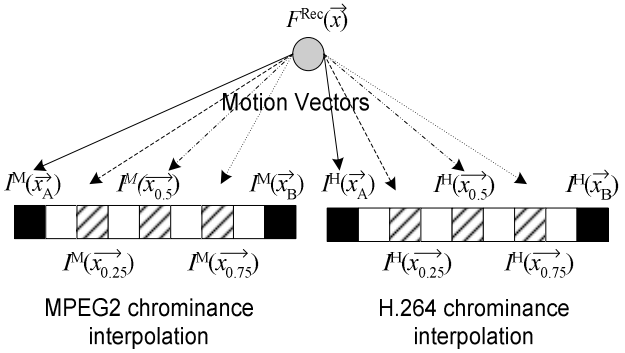

Fig. 3 The MPEG2 and H.264 chrominance interpolation

Table 1 Four categories of chrominance distortions based on the motion vector

Luminance motion vector	0	0.5	1	1.5	
Chrominance motion vector	0	0.25	0.5	0.75	……
Distortion types	Integer pixel distortion	Quarter pixel distortion	Half pixel distortion	Three-quarter pixel distortion	

3.1. Chrominance integer pixel distortion compensation

If the chrominance motion vector points to a pixel at an integer position in the reference frame, then the vector has an integer value. For example, in Fig. 3, the motion vector of $F^{Rec}(\vec{x})$ points to $I^M(\vec{x}_A)$ in the MPEG2 reference frame and points to $I^H(\vec{x}_A)$ in the H.264 reference frame.

In this case, $MC_M(I^M(\vec{x}))$ and $MC_H(I^H(\vec{x}))$ in (1) become:

$$\text{MPEG2: } MC_M(I^M(\vec{x})) = I^M(\vec{x}_A) \quad (2)$$

$$\text{H.264: } MC_H(I^H(\vec{x})) = I^H(\vec{x}_A) \quad (3)$$

From (2) and (3), (1) becomes:

$$P^H(\vec{x}) = P^M(\vec{x}) + I^M(\vec{x}_A) - I^H(\vec{x}_A) \quad (4)$$

Since the open-loop structure uses $P^M(\vec{x})$ and the cascaded structure uses $P^H(\vec{x})$ as the residue of $F^{Rec}(\vec{x})$, distortion is introduced in the open-loop structure. Therefore, in the open-loop structure, the value of $P^M(\vec{x})$ should be changed

to the value of $P^H(\vec{x})$ in H.264 encoding. From (4), the difference between $P^H(\vec{x})$ and $P^M(\vec{x})$ is equal to:

$$\Delta\vec{x}_A = I^M(\vec{x}_A) - I^H(\vec{x}_A) \quad (5)$$

The difference $\Delta\vec{x}_A$ is addressed as the re-quantization error in [5]. The efficient algorithm in [5] can be used to compensate for the distortion in the chrominance case as well. $P^H(\vec{x})$ will be given by:

$$P^H(\vec{x}) = P^M(\vec{x}) + \Delta\vec{x}_A \quad (6)$$

From (6), one can see that neither MPEG2 motion compensation nor H.264 motion compensation is needed to compensate for chrominance integer pixel distortion in MPEG2 to H.264 transcoding.

3.2. Chrominance quarter pixel distortion compensation

If the chrominance motion vector has a quarter-pixel value, i.e., 0.25, 1.25, ·····, the encoder has to interpolate these quarter sub-pixels in the reference frame ($I^M(\vec{x}_{0.25})$ and $I^H(\vec{x}_{0.25})$ in Fig. 3). Unfortunately, H.264 uses a different interpolation filter from that of MPEG2 [8]. In this case, $MC_M(I^M(\vec{x}))$ and $MC_H(I^H(\vec{x}))$ in (1) become:

$$MC_M(I^M(\vec{x})) = I^M(\vec{x}_{0.25}) = I^M(\vec{x}_A) \quad (7)$$

$$MC_H(I^H(\vec{x})) = I^H(\vec{x}_{0.25}) = (48I^H(\vec{x}_A)+16I^H(\vec{x}_B)+32)/64 \quad (8)$$

In this case, (1) can be expressed as:

$$P^H(\vec{x}) = P^M(\vec{x}) + I^M(\vec{x}_A) - (48I^H(\vec{x}_A)+16I^H(\vec{x}_B)+32)/64 \quad (9)$$

As discussed above, the value of $P^M(\vec{x})$ should be changed to the value of $P^H(\vec{x})$. By using (5) and (9), we derive the equation to adjust the value of $P^M(\vec{x})$ to the value of $P^H(\vec{x})$:

$$P^H(\vec{x}) = P^M(\vec{x}) + \Delta\vec{x}_A + (I^H(\vec{x}_A) - I^H(\vec{x}_B) - 2)/4 \quad (10)$$

From (10), one can see that the adjustment is composed of two parts: $\Delta\vec{x}_A$, which is related to the re-quantization error, and $(I^H(\vec{x}_A) - I^H(\vec{x}_B) - 2)/4$, which is related to the chrominance quarter pixel interpolation.

3.3. Chrominance half pixel distortion compensation

To interpolate half-pixel samples in the reference frame, H.264 uses the same linear filter as MPEG2. Here $MC_M(I^M(\vec{x}))$ and $MC_H(I^H(\vec{x}))$ in (1) become:

$$MC_M(I^M(\vec{x})) = I^M(\vec{x}_{0.5}) = (I^M(\vec{x}_A) + I^M(\vec{x}_B) + 1)/2 \quad (11)$$

$$MC_H(I^H(\vec{x})) = I^H(\vec{x}_{0.5}) = (I^H(\vec{x}_A) + I^H(\vec{x}_B) + 1)/2 \quad (12)$$

Then (1) is derived as:

$$P^H(\vec{x}) = P^M(\vec{x}) + (I^M(\vec{x}_A) + I^M(\vec{x}_B) + 1)/2 - (I^H(\vec{x}_A) + I^H(\vec{x}_B) + 1)/2 \quad (13)$$

Again, we derive the equation to adjust the value of $P^M(\vec{x})$ to the value of $P^H(\vec{x})$:

$$P^H(\vec{x}) = P^M(\vec{x}) + (\Delta\vec{x}_A + \Delta\vec{x}_B)/2 \quad (14)$$

From (14), we observe that only the re-quantization errors $(\Delta\vec{x}_A, \Delta\vec{x}_B)$ are needed to compensate for the chrominance half-pixel distortion.

3.4. Chrominance three-quarter pixel distortion compensation

If the chrominance motion vector has a three-quarter pixel value, i.e., 0.75, 1.75, ·····, the encoder has to interpolate these sub-pixels in the reference frame. H.264 and MPEG2 use different filters to interpolate these samples [8]. In this case, $MC_M(I^M(\vec{x}))$ and $MC_H(I^H(\vec{x}))$ in (1) become:

$$MC_M(I^M(\vec{x})) = I^M(\vec{x}_{0.75}) = (I^M(\vec{x}_A) + I^M(\vec{x}_B) + 1)/2 \quad (15)$$

$$MC_H(I^H(\vec{x})) = I^H(\vec{x}_{0.75}) = (16I^H(\vec{x}_A) + 48I^H(\vec{x}_B) + 32)/64 \quad (16)$$

Then, (1) is expressed as:

$$P^H(x) = P^H(\vec{x}) = P^M(\vec{x}) + (I^M(\vec{x}_A) + I^M(\vec{x}_B) + 1)/2 - (16I^H(\vec{x}_A) + 48I^H(\vec{x}_B) + 32)/64 \quad (17)$$

Once more, we derive the equation to adjust the value of $P^M(\vec{x})$ to the value of $P^H(\vec{x})$:

$$P^H(\vec{x}) = P^M(\vec{x}) + (\Delta\vec{x}_A + \Delta\vec{x}_B)/2 + (I^H(\vec{x}_A) - I^H(\vec{x}_B))/4 \quad (18)$$

Similar to the case of the quarter pixel distortion, the adjustment is composed of two parts: one is related to the re-quantization errors ($(\Delta\vec{x}_A + \Delta\vec{x}_B)/2$) and the other is related to chrominance three-quarter pixel interpolation ($(I^H(\vec{x}_A) - I^H(\vec{x}_B))/4$).

4. PROPOSED TRANSCODING STRUCTURE

From the above it is clear that, in order to compensate for the chrominance distortion, we need to calculate re-quantization errors and reconstruct integer samples in the H.264 reference frame. Unlike the cascaded structure, which implements both MPEG2 and H.264 motion compensation processes, the proposed structure does not need to implement MPEG2 motion compensation. In addition, the equations ((6), (10), (14), (18)), which are used to adjust the residue, are much simpler than the interpolation equations. To reconstruct the integer samples of previous H.264 reference frames, a closed-loop transcoding structure is used. These samples will be used to remove the distortions related to mismatches between the MPEG2 and H.264 chrominance interpolation. The proposed structure is shown in Fig. 4. The shaded blocks in Fig. 4 can be replaced by the DCT-to-HT

module introduced in [4]. However, in order to avoid the errors generated by our own implementation of the DCT-to-HT module, we use inverse DCT followed by H.264 integer transform. This implementation does not affect our proposed scheme since our algorithm is applied on the resulting integer transform coefficients and it does not matter how these coefficients were calculated.

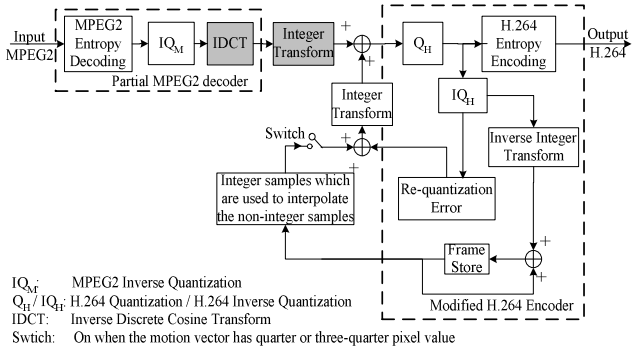

Fig. 4 The proposed closed-loop transcoding structure

5. EXPERIMENTAL RESULTS

Four video sequences (with motion details from subtle to large) are used: Akiyo, Foreman, Football and Mobile & Calendar. The Test Model 5 MPEG2 reference software and the Nokia H.264 reference software are used in our implementation.

Since the Mobile sequence is the most complicated sequence in terms of color information, only the rate-distortion curves of the Mobile sequence are shown in Fig. 5 (includes both U and V color components). Three types of results are compared. The top one is the result from the cascaded decoder-encoder structure. The middle one refers to the result from the proposed transcoding structure. The bottom one represents the result from the open-loop structure. We observe that our proposed compensation procedure improves the picture quality compared to the open-loop structure by an average of 6dB. It can also be seen that the difference in picture quality between what our proposed structure yields and that of the cascaded structure is negligible. At the same time, the computational complexity is significantly reduced (35% decrease in terms of transcoding time).

6. CONCLUSION

The open-loop transcoding is known as the most computational efficient transcoding structure. In heterogeneous transcoding, the open-loop transcoding usually introduces many distortions in the transcoded video. The reason for that is the different motion compensation processes used by the different video coding standards. This paper addresses the chrominance distortions resulting from the open-loop MPEG2 to H.264 transcoding structure. Then, the equations are derived to compensate for the chrominance distortions. Finally, the open-loop structure is replaced by a closed-loop transcoding structure which provides high-quality video by removing the chrominance distortions, resulting in an average of 6dB picture quality improvement.

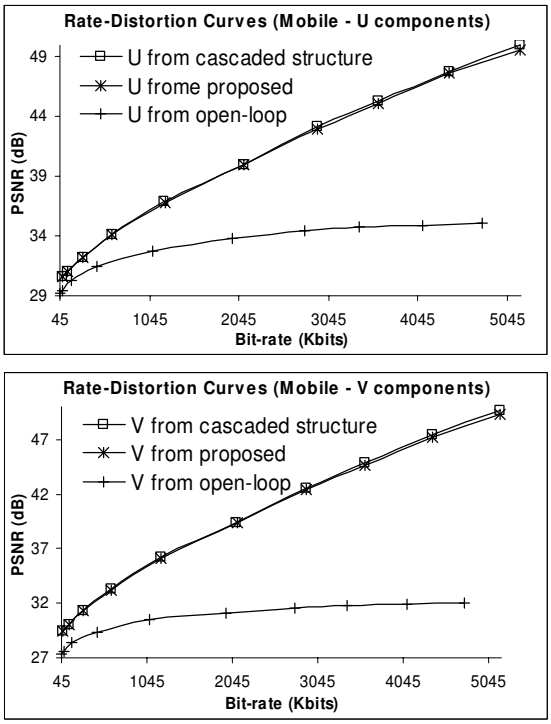

Fig. 5 Rate-distortion curves of chrominance in Mobile & Calendar sequence.

7. REFERENCES

[1] J. Xin, C.W. Lin, and M.T. Sun, "Digital video transcoding," *Proceedings of the IEEE*, pp. 84–97, Jan. 2005.

[2] Z. Zhou, S. Sun, S. Lei, and M.T. Sun, "Motion Information and Coding Mode Reuse for MPEG-2 to H.264 Transcoding", *ISCAS 2005*, pp. 1230–1233, May 2005.

[3] X. Lu, A. Tourapis, P. Yin and J. Boyce, "Fast Mode Decision and Motion Estimation for H.264 with a Focus on MPEG2/H.264 Transcoding," *ISCAS 2005*, pp. 1246–1249, May 2005.

[4] J. Xin, A. Vetro, and H. Sun, "Converting DCT coefficients to H.264/AVC transform coefficients," *Pacificrim Conference on Multimedia (PCM)*, pp. 939–946, 2004.

[5] Q. Tang, P. Nasiopoulos, R. Ward, "An Efficient Re-quantization Error Compensation for MPEG2 to H.264 Transcoding, "*2006 IEEE ISSPIT*, pp. 530–535, Aug. 2006.

[6] Q. Tang, R. Ward, P. Nasiopoulos, ""An Efficient Half-Pixel Motion Compensation MPEG2 to H.264 Transcoding," *IEEE ICIP 2006*, pp.865-868, Oct. 2006.

[7] T. Qian, J. Sun, D. Li, X. Yang, J. Wang, "Transform Domain Transcoding From MPEG2 to H.264 With Interpolation Drift-Error Compensation," *IEEE Trans. on CSVT*, pp. 523–534, Apr. 2006.

[8] Iain E.G. Richardson, *H.264 and MPEG-4 Video Compression - Video Coding for Next-generation Multimedia*, John Wiley & Sons Ltd, Chichester, England, 2003.