TRANSCODING OF MPEG-2 VIDEO IN THE FREQUENCY DOMAIN

*Pedro A. A. Assunção** *Mohammed Ghanbari*

Department of Electronic Systems Engineering
University of Essex
Colchester CO4 3SQ - United Kingdom
E-mail: {paaass, ghan}@essex.ac.uk

ABSTRACT

Video transcoding techniques offer the possibility of matching coded video to transmission channels of lower capacity by reducing the bit rate of compressed bit streams. In this paper we propose a new frequency domain video transcoder for bit rate reduction of compressed bit streams. A motion compensation (MC) loop, operating in the frequency domain, is used for drift compensation at reduced computational complexity. We derive approximate matrices for fast computation of the MC blocks in the frequency domain. By using the Lagrangian optimisation in calculating the best quantiser scales for transcoding, we show that transcoded pictures from a high quality bit stream are better than those encoded from original frames at the same reduced bit rates.

1. INTRODUCTION

Video transcoding aims at further bit rate reduction of compressed video for matching bit rates of pre-encoded bit streams to channel constraints. This problem has been addressed in recent work [1, 2, 3], where either open loop techniques or a cascade of decoding-encoding have been used. The former always introduces drift producing increasingly distorted pictures along the length of the group of pictures (GOP). The latter, apart from the complexity and cost, introduces a significant amount of delay which is not suitable for networking applications where fast reaction to network demand is required.

In the previous work we have devised a low delay and drift free architecture for transcoding of compressed MPEG bit streams [4]. More recently, a similar transcoder was presented in [5]. In this paper we propose a simplified and optimised video transcoder, based on our previous scheme, where the complexity and efficiency of transcoding is greatly improved.

We have employed frequency domain motion compensation to eliminate the need for the forward transform (DCT) and its inverse (IDCT) [6]. These were necessary in [4] to accumulate the transcoding error in the pixel domain to compensate picture drift. We derive approximate matrices for computing the motion compensated blocks in the DCT domain (MC-DCT) such that only the basic arithmetic integer operations of *shift* and addition (*add*) are needed. This greatly reduces the computational complexity (number of *shift* and *add* operations) of MC-DCT while maintaining a good drift resilience over long GOPs. The computational complexity of the frequency domain transcoder is reduced by 81% compared with that of its pixel domain equivalent. Compared with the fast MC-DCT algorithm proposed in [7], our method requires 72% less operations.

The Lagrange multiplier method is used to find the optimal quantiser parameters which minimise distortion. Since minimum delay in transcoding is of prime importance, storage of compressed video for optimisation was limited to one video frame. We show that, for constant bit rate (CBR) transcoding, the overall efficiency of the optimised transcoder is 1.5-2.5 dB better than the cascade of decoding-encoding with a TM5 [8] encoder. In the case of transcoding variable bit rate (VBR) bit streams, using fixed quantiser scales, the transcoded pictures are 0-0.5 better than those fully re-encoded.

2. TRANSCODER DESCRIPTION

The basic architecture of the video transcoder proposed in this paper is derived from a cascade of decoder-encoder as we had shown in [4]. A simplified scheme of this architecture, excluding the variable length decoding (VLD) and coding (VLC), is depicted in figure 1. Since MC is primarily defined as a pixel domain operation and its implementation in the DCT domain is not straightforward for the MPEG-2 algorithm, DCT and IDCT were needed. The two buffers *'Ref. P'* and *'Ref. F'* accumulate the transcoding error for compensating drift in the predicted frames. B pictures need both of them, whereas P pictures need only *'Ref. P'* and I pictures do not need any of them at all.

Since motion compensation (MC) can be performed in the frequency domain, there is no need for DCT and IDCT of figure 1, hence the transcoder can be further simplified. In the new scheme shown in figure 2 the transcoding error, given by the difference between the input and output inverse quantised DCT coefficients, is accumulated in the transform domain and added to the DCT blocks of the current picture after MC-DCT. The dashed blocks represent two functions to support the MPEG-2 syntax, (*i*) field/frame DCT conversion (f/F) and (*ii*) motion compensation. Since the transcoding error of all macroblocks is accumulated in frame format, whenever a macroblock of a frame picture is field DCT coded the f/F function converts it into frame

*His work is supported by *Instituto Politécnico de Leiria* and *JNICT/Praxis XXI: Sub-Programa Ciência e Tecnologia do 2º Quadro Comunitário de Apoio*, Portugal.

format. This function is not necessary in field pictures.

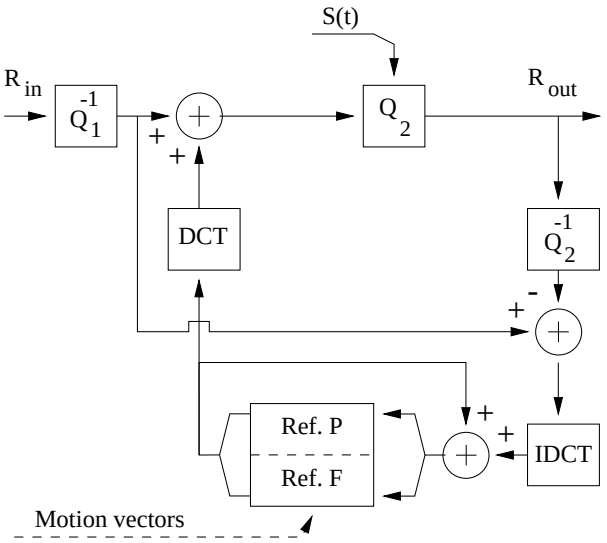

Figure 1. Video transcoder - pixel domain

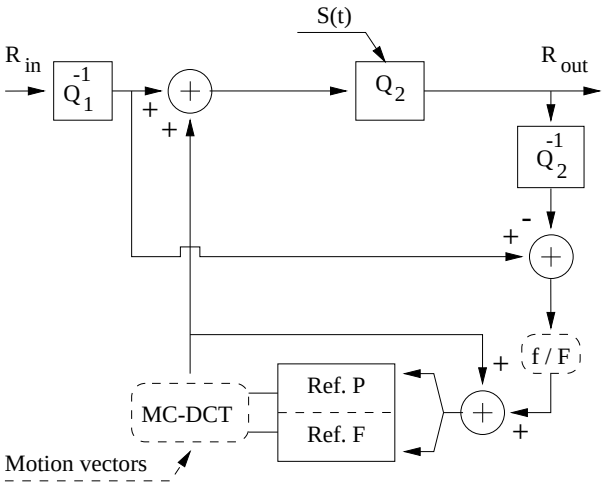

Figure 2. Video transcoder - DCT domain

3. FAST COMPUTATION OF MC-DCT

In general, the extraction of one MC pixel block $\hat{\mathbf{b}}$ from the reference frame involves 4 neighbouring blocks $\mathbf{b_i}, i = 1, \cdots 4$ that are intersected by $\hat{\mathbf{b}}$. Hence, $\hat{\mathbf{b}}$ is formed by 4 subblocks, which can be extracted by multiplying each block $\mathbf{b_i}$ with the appropriate matrices $\mathbf{h_{hi}}, \mathbf{h_{wi}}$, as shown in [6]. These matrices perform window and shift operations on each block $\mathbf{b_i}$ such that the overlapped subblocks are moved into the correct place in $\hat{\mathbf{b}}$. The number of rows (h) and columns (w), that each block $\mathbf{b_i}$ is intersected by the MC block $\hat{\mathbf{b}}$ *i.e.* the size of each subblock, defines which matrices apply for each $\mathbf{b_i}$. The structure of $\mathbf{h_{hi}}, \mathbf{h_{wi}}$ is that of either $\mathbf{u_n}$ or $\mathbf{l_m}$, where the size n, m of the identity sub-matrices is given by the size of each subblock in $\mathbf{b_i}$.

$$\mathbf{u_n} \equiv \begin{bmatrix} 0 & I_n \\ 0 & 0 \end{bmatrix}, \quad \mathbf{l_m} \equiv \begin{bmatrix} 0 & 0 \\ I_m & 0 \end{bmatrix}, \quad m, n = 1 \cdots 7$$

The matrices $H_{hi} = DCT(h_{hi}), H_{wi} = DCT(h_{wi})$ can be pre-computed and used as constants to extract the MC-DCT block $\hat{B} = DCT(\hat{\mathbf{b}})$ directly from the DCT blocks $B_i = DCT(\mathbf{b_i})$. By applying the distributive property of matrix multiplication with respect to DCT this leads to equation 1. Taking into account that $\mathbf{u_n}$ is the transpose of $\mathbf{l_m}$ for $m = n$ and thus their transforms are also transposed, only 7 (instead of 14) different matrices need to be stored.

$$\hat{B} = \sum_{i=1}^{4} H_{hi}.B_i.H_{wi} \quad w, h \in \{1, 2 \cdots 7\} \tag{1}$$

The brute-force computation of equation 1 in the case where the MC block is not aligned in any direction with the block structure, requires eight matrix multiplications and four matrix additions, both using floating point precision. Our aim is to reduce the number and complexity of the operations involved in solving equation 1. We achieve this goal by approximating the elements of H_{wi}, H_{hi} to binary numbers with a maximum distortion of 1/32. As an example we show a number of elements of matrix $H_{w1}, w = 5$, which is used to extract the DCT components of the pixel subblock located at the up left corner of $\hat{\mathbf{b}}$.

$$\begin{bmatrix} 0.625000 & 0.418576 & -0.163320 & \cdots & 0.083260 \\ -0.418576 & -0.062600 & 0.498124 & \cdots & -0.115485 \\ -0.163320 & -0.498124 & -0.566942 & \cdots & 0.136412 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -0.083260 & -0.115485 & -0.136412 & \cdots & -0.540954 \end{bmatrix}$$

Each element of the rounded matrix, in terms of powers of two is:

$$\begin{bmatrix} \frac{1}{2}+\frac{1}{8} & \frac{1}{2}-\frac{1}{16} & -\frac{1}{8}-\frac{1}{16} & \cdots & \frac{1}{16} \\ \frac{1}{16}-\frac{1}{2} & -\frac{1}{16} & \frac{1}{2} & \cdots & -\frac{1}{8} \\ -\frac{1}{8}-\frac{1}{16} & -\frac{1}{2} & -\frac{1}{2}-\frac{1}{16} & \cdots & \frac{1}{8} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\frac{1}{16} & -\frac{1}{8} & -\frac{1}{8} & \cdots & -\frac{1}{2}-\frac{1}{16} \end{bmatrix}$$

By approximating all the constant matrices in a similar way, only basic integer operations, such as *shift-right* and additions (*add*), are needed to solve equation 1. Since the elements of DCT blocks are in the range [-2048, 2047], shifting the actual values would result in zero for most elements. In order to maintain precision in intermediate operations, the transcoding error of each DCT coefficient is multiplied by 2^8 and stored in this format. This is in fact a scaling factor which can be included in the quantisation and inverse quantisation functions without additional complexity

Furthermore, by employing simple data manipulation the multiplication of the constant matrices by the DCT blocks can be implemented with a reduced number of operations. If all DCT coefficients of the same column that are *right-shifted* by k (multiplied by $\frac{1}{2^k}$) are added together before

shifting, then the number of *shift* operations will be reduced. This is explained in the following example using the third line of the matrix above, where both *eqn* 2 and *eqn* 3 need 5 additions but the number of *shifts* is quite different; *eqn* 2 requires 6 *shifts* whilst *eqn* 3 is implemented with 3 *shifts* only.

$$y = -\left(\tfrac{1}{8}+\tfrac{1}{16}\right)x_1 - \tfrac{1}{2}x_2 - \left(\tfrac{1}{2}+\tfrac{1}{16}\right)x_3 + \tfrac{1}{8}x_8 \quad (2)$$

$$= \tfrac{1}{2}(-x_2 - x_3) + \tfrac{1}{8}(x_8 - x_1) + \tfrac{1}{16}(-x_1 - x_3) \quad (3)$$

The number of operations required for pre-multiplication with any of the H_{wi}, H_{hi} matrices is the same as for post-multiplication. Since there are 14 different matrices, but 7 of them are the transposes of the other 7 we only need to account for 7. The number of operations (*shift, add*) required for each matrix is depicted in table 1

Matrix (n)	No of shift	No of add
1	17	65
2	24	67
3	26	78
4	28	66
5	29	73
6	30	67
7	29	87

Table 1. Number of operations for each matrix

Since in equation 1 $H_{h1} = H_{h2}$, then only 6 matrix multiplications (instead of 8) and 3 additions are required. The worst case in terms of number of operations is for $H_{hi}, H_{wi} = \{U_3, L_5, U_5, L_3\}$ corresponding to the case where the MC block intersects the blocks in the reference picture in 3 rows and 5 columns or vice versa. Considering that all block coefficients are non-zero, the total number of operations for extracting one block is 810. This corresponds to a reduction of 72% compared with 2928 operations required by the fast MC-DCT algorithm proposed in [7]. Comparing with the DCT/IDCT approach used in the pixel domain transcoder, the reduction in computational complexity of the algorithm proposed in [7] is of 32% whereas in the case of our method is of 81%. This is a worst case comparison since the reference for accounting complexity of DCT/IDCT is the fastest existing algorithm for 8-point DCT [9].

We have also measured the possible drift due to matrix truncation in the transcoder. A GOP of 45 predicted frames was transcoded using (*i*) the proposed fast MC-DCT algorithm and (*ii*) the floating point computation of MC-DCT with full precision. The accumulated error due to integer computation of MC-DCT after 45 frames is only 0.2 dB compared with the full precision of floating point computation.

3.1. MPEG-2 special MC cases

In MPEG-2, motion compensation can be half-pixel precision and several possible combinations of current/previous picture type (frame or field) and prediction modes (frame, field,16x8, dual prime) may occur in a compressed bit stream [10]. The half-pixel accuracy involves either two or four pixels for calculating the actual prediction of each pixel. In terms of blocks this is equivalent to compute the average, for each pixel, of either two or four blocks. In the DCT domain, this would need the extraction of two or four blocks, which would be equivalent to solve equation 1 either twice or four times. In order to avoid solving equation 1 more than once per block, we have applied a linear filtering to the MC-DCT block. Vertical filtering is implemented by pre-multiplying the DCT block with the matrix of DCT filter coefficients V, whilst horizontal filtering uses post-multiplication by another DCT matrix H. These can also be pre-multiplied with U_n, L_m and simplified through a similar process as described above, such that the extraction and filtering of a DCT block is done in one step only. This implies that 14 more constant matrices need to be stored.

In order to support field prediction the transcoder should be capable of extracting field DCT blocks from frame pictures and also generating frame DCT blocks from field DCT blocks. The former is obtained by extracting the two frame DCT blocks and pre-multiplying them by constant matrices F_1, F_2. The latter is obtained in the same way but using different matrices G_1, G_2. These, are also used by the f/F function of figure 2. The 16x8 MC mode does not need any further operations since the only difference from the normal case is that two MVs, instead of one, are used for each macroblock. The dual-prime MC mode can be used in MPEG-2 when P pictures are encoded without any B picture between them. Either two or four predictions are used for each block of a field or frame picture respectively. This case is implemented by using the $F_i, G_i, i = 1, 2$ matrices without any further processing. The $F_i, G_i, i = 1, 2$ matrices are the DCT transforms of the $f_i, g_i, i = 1, 2$ which perform the desired operation in the pixel domain.

4. SIMULATION

The efficiency of the proposed transcoder has been made optimal in a rate-distortion sense by minimising the transcoding distortion for a given external bandwidth constraint. This bandwidth constraint is actually a reduction factor $0 < S(t) \leq 1$ for the number of bits used in the output stream. Defining $D_T(t) = \|R_{in}(t) - R_{out}(t)\|$ as the transcoding distortion of converting the input bit rate $R_{in}(t)$ to the output bit rate $R_{out}(t)$, the problem of optimal transcoding can be formulated as follows,

$$min\,\{D_T(t)\} \text{ subject to } R_{out}(t) \leq S(t).R_{in}(t) \quad (4)$$

The solution of this problem provides the optimal quantisers to be used for transcoding. These can be found through a search algorithm using the Lagrange multiplier method [11]. Since the optimisation algorithm needs a certain amount of stored DCT macroblocks and we aim for a low delay transcoder, either one frame or one slice can be used. In this experiment we have used one frame storage.

In order to show the efficiency of this frequency domain transcoder two high quality bit streams were generated (N=15, M=3), one CBR and one VBR, by encoding the MOBILE sequence using a standard TM5 MPEG-2 encoder [8]. These bit streams were then transcoded into a lower rate and the resultant pictures compared with those fully re-encoded (standard decoder-encoder) from the same bit

stream, at the same lower rates. Also, the original image sequence was directly encoded at the lower rates. Figures 3 and 4 show the peak signal to noise ratio (PSNR) for the CBR and VBR cases respectively.

The CBR experiment was to transcode and re-encode a 4 Mbit/s bit stream into 2 Mbit/s and compare the resulting picture quality with that of another bit stream encoded directly from original pictures at 2 Mbit/s. As figure 3 shows, transcoding from a high quality bit stream produces even better results than encoding the original video with a standard encoder. This is because the best quantiser step sizes for transcoding are found through optimisation, hence the bit allocation in the output bit stream is optimal in rate-distortion sense.

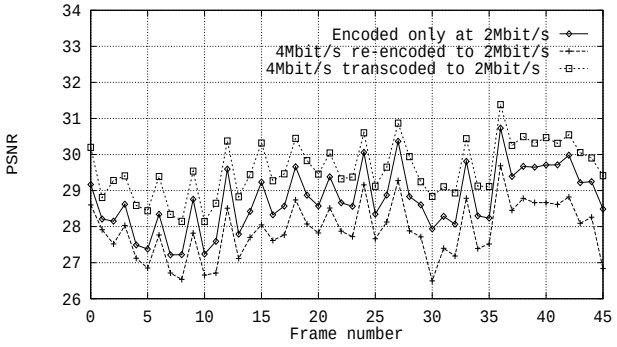

Figure 3. PSNR for CBR transcoding

In the VBR case the high quality bit stream was generated using fixed quantiser step sizes for each picture type ($Q_1^I = 6, Q_1^P = 8, Q_1^B = 8$). Then this bit stream was transcoded and fully re-encoded using $Q_2^I = 10, Q_2^P = 14, Q_2^B = 14$. Note that, in this experiment the quantiser step sizes are kept fixed and the optimisation algorithm is not used. As it is shown in figure 4 the proposed transcoder outperforms a re-encoding system and the difference to a standard encoder is less than 0.5 dB, on average.

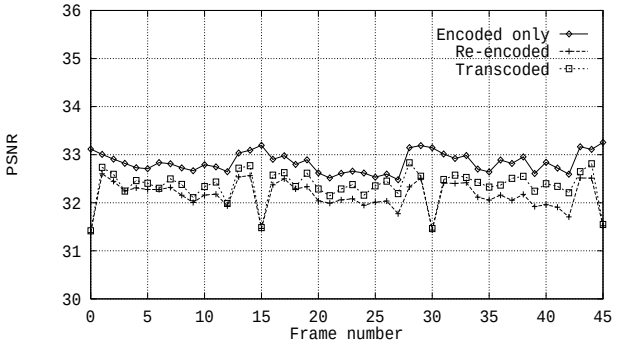

Figure 4. PSNR for VBR transcoding

5. CONCLUSION

We have proposed frequency domain video transcoder for reducing the bit rate of MPEG-2 compressed bit streams. The complexity of the MC-DCT function was reduced by 72% compared to a fast algorithm recently proposed in the literature and by 81% compared with the DCT/IDCT approach. This is achieved by approximating the transformation matrices such that only the integer operations of *shift* and *add* are used. Also the Lagrangian optimatisation was shown to give the transcoder a better efficiency than that of a standard encoder. The simulation results show that optimal transcoding from a high quality bit stream is even better than encoding the original pictures at the same bit rate. This proves that compressed video can be further compressed, at reduced complexity, achieving the same subjective picture quality as if uncompressed video was used.

REFERENCES

[1] H. Sun, W. Kwok, and J. W. Zdepski, "Architectures for MPEG compressed bitstream scaling," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, pp. 191–199, April 1996.

[2] Y. Nakajima, H. Hori, and T. Kanoh, "Rate conversion of MPEG coded video by re-quantization process," in *International Conference on Image Processing*, vol. 3, (Washington DC - USA), pp. 408–411, October 1995.

[3] A. Eleftheriadis and D. Anastassiou, "Meeting arbitrary QoS constraints using dynamic rate shaping of coded digital video," in *Fifth International Workshop on Network and Operating System for Digital Audio and Video*, (Durham, New Hampshire), pp. 95–106, April 1995.

[4] P. Assuncao and M. Ghanbari, "Post-processing of MPEG2 coded video for transmission at lower bit rates," in *IEEE International Conference in Acoustics, Speech, and Signal Processing, ICASSP'96*, vol. 4, (Atlanta - USA), pp. 1999–2002, May 1996.

[5] G. Keesman, R. Hellinghuizen, F. Hoeksema, and G. Heideman, "Transcoding of MPEG-2 bitstreams," *Signal Processing: Image Communication*, vol. 8, pp. 481–500, September 1996.

[6] S.-F. Chang and D. G. Messerschmitt, "Manipulation and compositing of MC-DCT compressed video," *IEEE Journal on Selected Areas in Communications*, vol. 13, pp. 1–11, January 1995.

[7] N. Merhav and V. Bhaskaran, "A fast algorithm for DCT-domain inverse motion compensation," in *International Conference on Acoustics Systems and Signal Processing*, (Atlanta, Georgia - USA), May 1996.

[8] ISO/IEC, "Test model 5, iso/iec jtc1/sc29/wg11/ n0400, mpeg93/457, april 1993.," April 1993.

[9] Y. Arai, T. Agui, and M. Nakajima, "A fast DCT-SQ scheme for images," *The transactions of the IEICE*, vol. E71, pp. 1095–1097, November 1988.

[10] ISO/IEC 13818-2, "Generic coding of moving pictures and associated audio, recommendation h.262," March 1994.

[11] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantisers," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, pp. 1445–1453, September 1988.