

A novel DCT domain transcoder for transcoding video streams with half-pixel motion vectors

Gang Cao^{a,c}, Zhijun Lei^{b,*}, Jintao Li^a, Nicolas D. Georganas^b, Zhenmin Zhu^a

^aDigital Technology Laboratory (DTL), Institute of Computing Technology, Chinese Academy of Sciences, P.O. Box 2704, Beijing, 100080 China

^bDistributed and Collaborative Virtual Environments Research Laboratory (DISCOVER), School of Information Technology and Engineering, University of Ottawa, 800 King Edward Ave, Ottawa, Ont., Canada

^cGraduate School of the Chinese Academy of Sciences, Jia 19 Yuquan Rd., Beijing, 100039 China

Available online 18 September 2004

Abstract

Video transcoding is one of the key techniques to provide heterogeneous multimedia applications with *Universal Multimedia Access* (UMA) service. Recently, compressed domain (DCT, Discrete Cosine Transform) transcoding architectures have been proposed to achieve fast transcoding, in which DCT domain *Inverse Motion Compensation* (IMC) is the most important module to reduce computational complexity. The problem of IMC has been studied for integer-pixel precision motion vectors (MVs), and, subsequently, for half-pixel precision MVs in which extra filtering, hence, extra computation is introduced. Since most current video coding standards, such as H.263, MPEG-2, etc., use half-pixel precision MVs to achieve better quality, reducing computational complexity of IMC for half-pixel MVs is important for real time fast transcoding video streams encoded by these standards. In this paper, we propose a novel half-pixel filter for IMC, which simplifies the extract operations by integrating the interpolation and translation operations into one single step. Compared to other existing half-pixel filtering algorithms, the proposed filter does not introduce any distortion and drift errors. Experimental results demonstrate that the proposed filter achieves faster transcoding than other DCT domain transcoders, and almost the same video quality as that of pixel domain transcoders.

© 2004 Elsevier Ltd. All rights reserved.

Keywords: Video transcoding; DCT domain transcoder; Translation; Interpolation; Half-pixel filter

1. Introduction

Universal multimedia access [1,2] deals with the delivery of media resources under different network conditions, user preferences, and capabilities of terminal devices. The primary motivation of UMA is that any media resources should be available to any users at anytime, anywhere. In transmitting a compressed video bit-stream over heterogeneous networks and to different client devices, it is often needed to dynamically adapt the bit rate of a coded video bit-stream to the available

bandwidth or convert its format to suit the requirement of client devices. Video transcoding is one of the solutions to enable the universal access of encoded bit-streams.

Video Transcoding [3] is the process of converting pre-compressed video signals from one format into another according to the requirements of media resources, network conditions and client devices. Major metrics for evaluating the performance of different transcoders are visual quality and computational complexity. Many transcoding architectures have been proposed in the literature to trade off the visual quality and the computational complexity. The traditional and straightforward implementation of a video transcoder is the drift-free Cascaded Pixel-Domain Transcoder (CPDT) [4], which connects a standard decoder and a

*Corresponding author.

E-mail addresses: caog@ict.ac.cn (G. Cao), leizj@discover.uottawa.ca (Z. Lei), jtli@ict.ac.cn (J. Li), georganas@discover.uottawa.ca (N.D. Georganas).

standard encoder. CPDT is proved to be able to maintain the best visual quality among all transcoding architectures. However, since CPDT does not eliminate the Motion Estimation (ME) and DCT/IDCT operations, which introduce high computational complexity, it is not suitable for real time video transcoding.

By simplifying the structure of CPDT, a DCT Domain Transcoder (DDT) [5] was proposed based on video manipulation operations in the DCT domain. As illustrated in Fig. 1, Motion Compensation (MC) in DDT is implemented completely in DCT domain. The transcoder is composed of three functionality blocks, including IQ1 (Inverse quantization with input quantization parameter), Q2 (quantization with new quantization parameter), and DCT-domain MC (corrects the requantization error introduced by Q2 and IQ2). The details of evolution from CPDT to DDT can be found in [5–7].

DCT domain motion compensation is the most crucial module for DDT. Many video manipulation functions can be implemented in the compressed domain and significantly reduce the high computational complexity of the corresponding brute force algorithms in the pixel domain. However, it is difficult to process motion-compensated DCT coefficients in the DCT domain. The problem of DCT-domain Inverse Motion Compensation (IMC), i.e., extracting one DCT block that is not aligned to the boundaries of 8×8 blocks in DCT domain, is an essential step of implementing DDT. Some simple translation manipulations for IMC were proposed by Chang et al. in [8] and further studied in [9,10]. However, these schemes involve high cost, when motion vectors are at half-pixel precision. Since half-pixel precision motion estimation has been widely used to improve quality of coded video in many video coding standards, such as H.263 [11], MPEG-2 [12], MPEG-4 [13], etc., implementing an efficient half-pixel IMC is crucial to DDT. In [7], Assunção et al. proposed a horizontal filter and a vertical filter for half-pixel precision IMC. Although the computing complexity is lower than Chang’s method, Assunção’s method introduces some distortion in those blocks located at the right and bottom boundaries of the macroblock (MB).

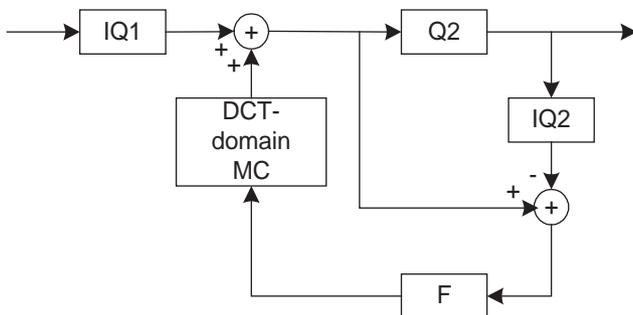


Fig. 1. DCT-domain transcoder.

In this paper, a novel half-pixel filter for DCT domain IMC is proposed and implemented. This novel filter integrates filtering and extraction operations into one step, which can speed up the DCT-domain transcoding process. If half-pixel precision motion vectors exist in both horizontal and vertical directions, only 4 block-extracting calculations for one luminance macroblock are needed by using the proposed filter. The rest of this paper is organized as follows. The problems of IMC, pixel interpolation and previous DCT domain filters, including Chang’s translation method and Assunção’s half-pixel filter, are described in Section 2. The proposed half-pixel filter is introduced in Section 3. Experimental results are presented in Section 4. Concluding remarks are presented in Section 5.

2. IMC and pixel interpolation

2.1. Integer-pixel inverse motion compensation

As illustrated in Fig. 2, the goal of IMC is to obtain the reference DCT block \hat{B} from the four neighboring DCT blocks in the reference frame. In general, \hat{B} pointed by $MV(x,y)$ may not be aligned to the boundaries of the original 8×8 DCT blocks (B_1 – B_4) in the reference frame, and may intersect with J neighboring blocks, $J = 1, 2, 4$.

As we can see from Fig. 2, when $J=1$ \hat{B} can be directly copied from one of the DCT blocks (B_1 – B_4) in the reference frame. The cases of $J=2$ can be derived from the cases of $J=4$. Therefore, in this paper, we only consider the case of $J=4$ as illustrated in Fig. 3, which is the most general and complicated case.

Let $MV(x,y)$ denotes the half-pixel precision motion vector carried by the current DCT block B . Fig. 3 shows that the intersection of the reference block \hat{B} with B_1 form a $m \times n$ rectangle, where $1 \leq m \leq 7$ and $1 \leq n \leq 7$ can be deduced from $MV(x,y)$. This means that the intersections of \hat{B} with $B_2, B_3,$ and B_4 are rectangle of sizes $m \times (8 - n), (8 - m) \times (8 - n),$ and $(8 - m) \times n,$ respectively. The components of $MV(x,y)$ are defined as follows:

$V_{x,w}$ the integer part of $MV(x,y)$ in the horizontal direction,
 $V_{y,w}$ the integer part of $MV(x,y)$ in the vertical direction,

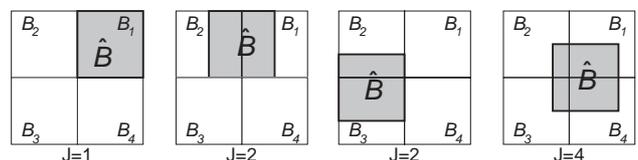


Fig. 2. Different cases of block intersection.

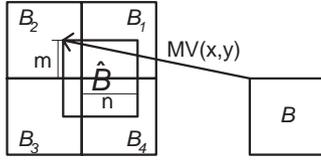


Fig. 3. Inverse Motion Compensation.

$V_{x,h}$ the half pixel part of $MV(x,y)$ in the horizontal direction,
 $V_{y,h}$ the half pixel part of $MV(x,y)$ in the vertical direction.

When $MV(x,y)$ only has integer components, i.e., $V_{x,h} = V_{y,h} = 0$, Chang et al. [8] proposed a translation operation to extract the reference DCT block \hat{B} from the 4 original DCT blocks B_1 – B_4 . In this case, m and n can be computed as follows:

$$m = \begin{cases} \text{mod}(\text{abs}(V_{y,w}), 8), & V_{y,w} < 0, \\ 8 - \text{mod}(\text{abs}(V_{y,w}), 8), & V_{y,w} > 0, \end{cases} \quad (1)$$

$$n = \begin{cases} 8 - \text{mod}(\text{abs}(V_{x,w}), 8), & V_{x,w} < 0, \\ \text{mod}(\text{abs}(V_{x,w}), 8), & V_{x,w} > 0. \end{cases} \quad (2)$$

Let I_h be an identity matrix of size $h \times h$, and matrix f_q be a sparse matrix of size 8×8 , $0 \leq q \leq 8$, i.e.,

$$f_q = \begin{bmatrix} 0 & I_{8-q} \\ 0 & 0 \end{bmatrix}. \quad (3)$$

Let F_q denote the DCT values of f_q .

In the pixel domain, the corresponding pixel blocks are denoted by b_1 – b_4 , \hat{b} . The pixel block \hat{b} can be calculated as a superposition of appropriate windowed and shifted versions of b_1 – b_4 , i.e.,

$$\hat{b} = \sum_{i=1}^4 l_{im} b_i r_{in}, \quad (4)$$

where

$$l_{1m} = l_{2m} = f_{8-m},$$

$$l_{3m} = l_{4m} = (f_m)^T,$$

$$r_{1n} = r_{4n} = f_{8-n},$$

$$r_{2n} = r_{3n} = (f_n)^T.$$

By applying the distributive property of matrix multiplication with respect to DCT, DCT block \hat{B} can be calculated as:

$$DCT(\hat{b}) = \sum_{i=1}^4 DCT(l_{im})DCT(b_i)DCT(r_{in}). \quad (5)$$

Let $L_{im} = DCT(l_{im})$, $R_{in} = DCT(r_{in})$, then

$$\hat{B} = \sum_{i=1}^4 L_{im} B_i R_{in}. \quad (6)$$

2.2. Half-pixel inverse motion compensation

When $V_{x,h} \neq 0$ and $V_{y,h} \neq 0$, Eq. (6) is applied four times to extract the predicted block. Sixteen blocks need to be extracted from the reference frame to construct one luminance MB with half-pixel precision motion vector at both horizontal and vertical direction by using the translation manipulation proposed in [8].

Assunção et al. [7] proposed a horizontal filter and a vertical filter to interpolate the corresponding blocks in reference frame for $V_{x,h} \neq 0$ and $V_{y,h} \neq 0$, respectively. Let us consider the DCT blocks B_i of an MC-DCT luminance MB ordered according to their location within the MB as in Fig. 3, $i = 1, 2, 3, 4$. The horizontally filtered blocks B_i^h in the DCT domain are obtained from the Eq. (7), while the vertically filtered blocks B_i^v are obtained from Eq. (8).

$$B_i^h = \begin{cases} B_2 F_1^h + B_1 F_2^h, & i = 2, \\ B_3 F_1^h + B_4 F_2^h, & i = 3, \\ B_i F_3^h, & i = 1, 4, \end{cases} \quad (7)$$

$$B_i^v = \begin{cases} F_1^v B_1 + F_2^v B_4, & i = 1, \\ F_1^v B_2 + F_2^v B_3, & i = 2, \\ F_3^v B_i, & i = 3, 4, \end{cases} \quad (8)$$

where the filter coefficient matrices F_i^h and F_i^v are defined as:

$$F_1^h = DCT(\frac{1}{2}(f_0 + (f_1)^T)),$$

$$F_2^h = DCT(\frac{1}{2}f_7),$$

$$F_1^v = DCT(\frac{1}{2}(f_0 + f_1)),$$

$$F_2^v = DCT(\frac{1}{2}(f_7)^T),$$

$$f_3^h = \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 1 \end{bmatrix},$$

$$F_3^h = DCT(f_3^h),$$

$$F_3^v = DCT((f_3^h)^T).$$

The translation manipulation mentioned introduced in Section 2.1 should also be applied following the interpolation filtering. To extract one luminance macro-block from a reference frame with half-pixel precision MV at both horizontal and vertical directions, 8 filter operations and 4 block extractions are calculated, which involves high computation cost. At the same time, this filter introduces some distortion in those blocks located at the right and bottom boundaries of the MB. The reason will be further investigated in the next Section.

3. Proposed half-pixel filter

3.1. Half-pixel interpolation

When motion vectors with half-pixel precision are used, either two or four pixels are needed to calculate the actual prediction of one single pixel as shown in Fig. 4. In terms of blocks, this is equivalent to computing the average of either two or four blocks for each pixel.

3.2. Proposed half-pixel filter

The proposed filter for half-pixel motion vectors can be derived directly from the Chang’s algorithm. When the half-pixel components of motion vectors are non-zero, i.e., $V_{x,h} \neq 0$ and $V_{y,h} \neq 0$, m and n can be computed as follows:

$$V_x = \begin{cases} V_{x,w} - 1, & V_{x,h} < 0, \\ V_{x,w}, & V_{x,h} > 0, \end{cases} \quad V_y = \begin{cases} V_{y,w} - 1, & V_{y,h} < 0, \\ V_{y,w}, & V_{y,h} > 0, \end{cases} \quad (9)$$

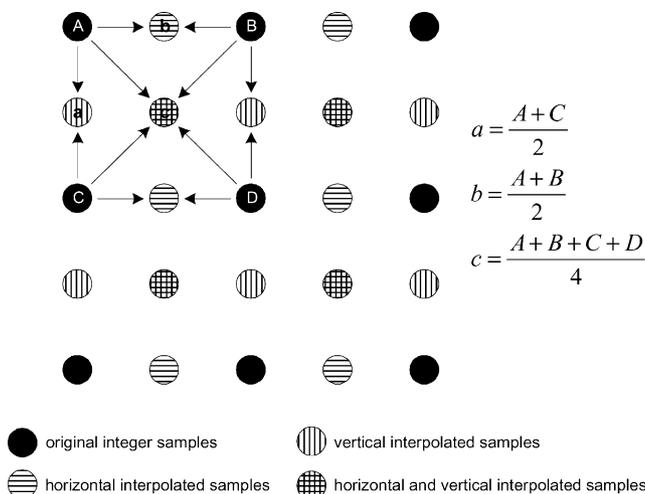


Fig. 4. Half-pixel interpolation.

$$m_y = \begin{cases} \text{mod}(\text{abs}(V_y), 8), & V_y < 0, \\ 8 - \text{mod}(\text{abs}(V_y), 8), & V_y \geq 0, \end{cases} \quad n_x = \begin{cases} 9 - \text{mod}(\text{abs}(V_x), 8), & V_x \leq 0, \\ \text{mod}(\text{abs}(V_x), 8) + 1, & V_x > 0, \end{cases} \quad (10)$$

$$m = \begin{cases} 8, & m_y = 0, \\ m_y, & \text{otherwise,} \end{cases} \quad n = \begin{cases} 1, & n_x = 9, \\ n_x, & \text{otherwise.} \end{cases} \quad (11)$$

Obviously $1 \leq m, n \leq 8$.

As introduced in the previous section, in order to extract one DCT block from the reference frame, if we follow the straightforward translation algorithm proposed by Chang et al. Eq. (6) will be used four times [14,15], i.e.,

$$\hat{B} = \frac{1}{4} \left(\sum_{i=1}^4 L_{im} B_i R_{in} + \sum_{i=1}^4 L_{im} B_i R_{i(n-1)} + \sum_{i=1}^4 L_{i(m-1)} B_i R_{in} + \sum_{i=1}^4 L_{i(m-1)} B_i R_{i(n-1)} \right). \quad (12)$$

Sixteen blocks need to be extracted from the reference frame to construct one luminance MB with half-pixel precision motion vector at both horizontal and vertical directions, which introduces very high computational complexity.

As noticed, if we use F_q to substitute the matrices used in Eq. (12), it can be rewritten as

$$\hat{B} = \frac{1}{4} \left(\begin{aligned} & (F_{8-m} B_1 F_{8-n} + F_{8-m} B_2 (F_n)^T \\ & + (F_m)^T B_3 (F_n)^T + (F_m)^T B_4 F_{8-n}) \\ & + (F_{8-m} B_1 F_{8-(n-1)} + F_{8-m} B_2 (F_{n-1})^T \\ & + (F_m)^T B_3 (F_{n-1})^T + (F_m)^T B_4 F_{8-(n-1)}) \\ & + (F_{8-(m-1)} B_1 F_{8-n} + F_{8-(m-1)} B_2 (F_n)^T \\ & + (F_{m-1})^T B_3 (F_n)^T + (F_{m-1})^T B_4 F_{8-n}) \\ & + (F_{8-(m-1)} B_1 F_{8-(n-1)} + F_{8-(m-1)} B_2 (F_{n-1})^T \\ & + (F_{m-1})^T B_3 (F_{n-1})^T + (F_{m-1})^T B_4 F_{8-(n-1)}) \end{aligned} \right). \quad (13)$$

Through some straight manipulation, we get the following simplified expression as follow:

$$\hat{B} = \frac{1}{4} \left(\begin{aligned} & (F_{8-m} + F_{9-m}) B_1 (F_{8-n} + F_{9-n}) \\ & + (F_{8-m} + F_{9-m}) B_2 (F_{n-1} + F_n)^T \\ & + (F_{m-1} + F_m)^T B_3 (F_{n-1} + F_n)^T \\ & + (F_{m-1} + F_m)^T B_4 (F_{8-n} + F_{9-n}) \end{aligned} \right). \quad (14)$$

If we reorder Eq. (14), then we have:

$$\hat{B} = \left(\frac{1}{2} (F_{(9-m)-1} + F_{9-m}) \right) B_1 \left(\frac{1}{2} (F_{(9-n)-1} + F_{9-n}) \right) \\ + \left(\frac{1}{2} (F_{(9-m)-1} + F_{9-m}) \right) B_2 \left(\frac{1}{2} (F_{n-1} + F_n)^T \right) \\ + \left(\frac{1}{2} (F_{m-1} + F_m)^T \right) B_3 \left(\frac{1}{2} (F_{n-1} + F_n)^T \right) \\ + \left(\frac{1}{2} (F_{m-1} + F_m)^T \right) B_4 \left(\frac{1}{2} (F_{(9-n)-1} + F_{9-n}) \right) \quad (15)$$

Furthermore, if we define a set of 8×8 sparse matrices X_u , $1 \leq u \leq 8$, as

$$X_u = \frac{1}{2}(F_{u-1} + F_u) \quad (16)$$

then, the interpolation and extraction procedure in DCT domain can be integrated into one step as given by

$$\hat{B} = \sum_{i=1}^4 P_{im} B_i Q_{in}, \quad (17)$$

where P_{im} and Q_{in} perform interpolation, windowing, and shifting operations, defined as follows:

$$P_{1m} = P_{2m} = X_{9-m}$$

$$P_{3m} = P_{4m} = (X_m)^T$$

$$Q_{1n} = Q_{4n} = X_{9-n}$$

$$Q_{2n} = Q_{3n} = (X_n)^T$$

This half-pixel filter is derived from Eq. (6). For half-pixel block extraction, no translation operations are needed after the filtering. However, the computing procedure for m and n with half-pixel precision MVs is different from that without half-pixel precision. The maximum value of m or n is 8 and 7 for MV with half-pixel precision and integer-pixel precision, respectively.

By using this set of filters, the half-pixel interpolation operation proposed by Assunção's et al. can be rewritten as

$$F_1^h = (X_1)^T, \quad F_2^h = X_8 \quad (18)$$

$$B_i^h = \begin{cases} B_2(X_1)^T + B_1 X_8, & i = 2 \\ B_3(X_1)^T + B_4 X_8, & i = 3 \\ B_i F_3^h, & i = 1, 4 \end{cases} \quad (19)$$

$$B_i^v = \begin{cases} X_1 B_1 + (X_8)^T B_4, & i = 1 \\ X_1 B_1 + (X_8)^T B_3, & i = 2 \\ F_3^v B_i, & i = 3, 4 \end{cases} \quad (20)$$

Define $x_1 = IDCT(X_1) = IDCT(\frac{1}{2}(f_0 + f_1))$. It should be noticed that the definition of $(f_3^h)^T$ is the same as x_1 except that the most bottom right element is 1 instead of 0.5, which results in distortion on those blocks located at the right and bottom boundaries of the MB.

3.3. Complexity analysis

The numbers of matrix operations (multiplication and addition) for calculating 4 MC-DCT blocks in a luminance MB with half-pixel MV are listed in Table 1. In order to extract a luminance MB with integer-pixel MV, the Chang's method needs 24 multiplications and 12 additions, which is the same as the proposed filter for a luminance MB with half-pixel MV in both directions.

Table 1
Number of matrix operations with different filters

Method	Multiplication	Addition
Chang	96	60
Assunção	36	16
Proposed	24	12

In order to extract a luminance MB with half-pixel MV in both directions, Chang's method needs to extract 4 luminance MBs, and Assunção's methods needs additional 8 filtering operations besides a luminance MB extraction operation. Compared with Assunção's filter, the proposed filter can reduce 33.33% and 25.00% of the matrix multiplications and addition operations.

As in Eq. (16), X_u are the new half-pixel filters, $1 \leq u \leq 8$. To use these filters in a DCT domain transcoder, eight matrices need to be calculated and stored. This is the overhead of using the proposed filters. However, considering the improvement on transcoding speed as illustrated in Table 1, this overhead is worthy. In addition, since most transcoders are currently deployed at video servers or proxies, which usually have large amount of memory.

4. Experimental results

In this work, we implemented a DDT transcoder based on H.263 codec [16]. Chang's translation algorithm was implemented for integer-pixel MVs. Assunção's half-pixel filter and the proposed half-pixel filter were implemented for half-pixel MVs, separately. A CPDT transcoder was also implemented as a reference transcoder because CPDT introduces no drift errors. Both CPDT and DDT reuse the incoming MVs from the input bit stream. To verify the performances of the proposed half-pixel filter, extensive simulations were carried out and results of the simulations were compared with the performance of CPDT. In our experiments, the transcoder using Assunção's half-pixel filter and the proposed half-pixel filter are denoted as DDT-Assunção and DDT-Novel, respectively.

We encoded 300 frames of Quarter Common Intermediate File (QCIF) format "Carphone", "Foreman", "Table Tennis" test sequences using fixed quantization parameter ($Q1=3$) with half-pixel MV precision. Only the first frame was encoded as Intra frame, and all other frames were encoded as Inter (P) frames. The default intra refresh frequency 132 in H.263 is adopted. The percentage of half-pixel MVs is shown in Table 2. As we can see from Table 2, motion vectors with half-pixel precision is a significant part of all motion vectors, therefore, improving the speed of half-pixel filters will improve the efficiency of the transcoder significantly.

Table 2
Percentage of half-pixel MV

Sequence	Y (%)	UV (%)
Carphone	39.27	46.35
Foreman	47.35	58.58
Table Tennis	24.47	29.56

Table 3
Comparison between CPDT and DDT using different filters with Q2=5 (dB)

		Carphone	Foreman	Table tennis
PSNR(Y)	CPDT	36.25	35.45	35.58
	DDT-Assunção	35.72	35.05	35.29
	DDT-Proposed	36.17	35.45	35.53
PSNR(Cb)	CPDT	40.08	40.11	40.26
	DDT-Assunção	39.26	39.29	39.76
	DDT-Proposed	39.60	39.82	40.21
PSNR(Cr)	CPDT	40.65	40.62	39.05
	DDT-Assunção	40.27	40.08	38.69
	DDT-Proposed	40.59	40.56	38.96

Table 4
Comparison between CPDT and DDT using different filters with Q2=10 (dB)

		Carphone	Foreman	Table tennis
PSNR(Y)	CPDT	32.42	31.64	31.94
	DDT-Assunção	31.79	31.22	31.60
	DDT-Proposed	32.41	31.67	31.92
PSNR(Cb)	CPDT	37.25	37.42	37.54
	DDT-Assunção	36.70	37.01	37.40
	DDT-Proposed	37.13	37.40	37.51
PSNR(Cr)	CPDT	37.69	37.61	36.00
	DDT-Assunção	37.58	37.10	35.70
	DDT-Proposed	37.91	37.73	36.02

The encoded bit streams were transcoded using Q2=5 and Q2=10, and the average PSNR of CPDT, DDT-Assunção and DDT-Novel are shown in Tables 3 and 4, respectively. As we can see, the proposed DDT outperforms DDT-Assunção in every sequence. In the DDT transcoder, drift errors introduced by Assunção’s filter are significant in most frames of the test sequences. The reason is that, in Assunção’s filter, the bottom right element of $(f_3^h)^T$ is 1 instead of 0.5, which results in distortion on those blocks located at the right and bottom boundaries of the MB. As we derived, the bottom right element of $(f_3^h)^T$ should be 0.5. The motion-compensated predictive coding algorithm is

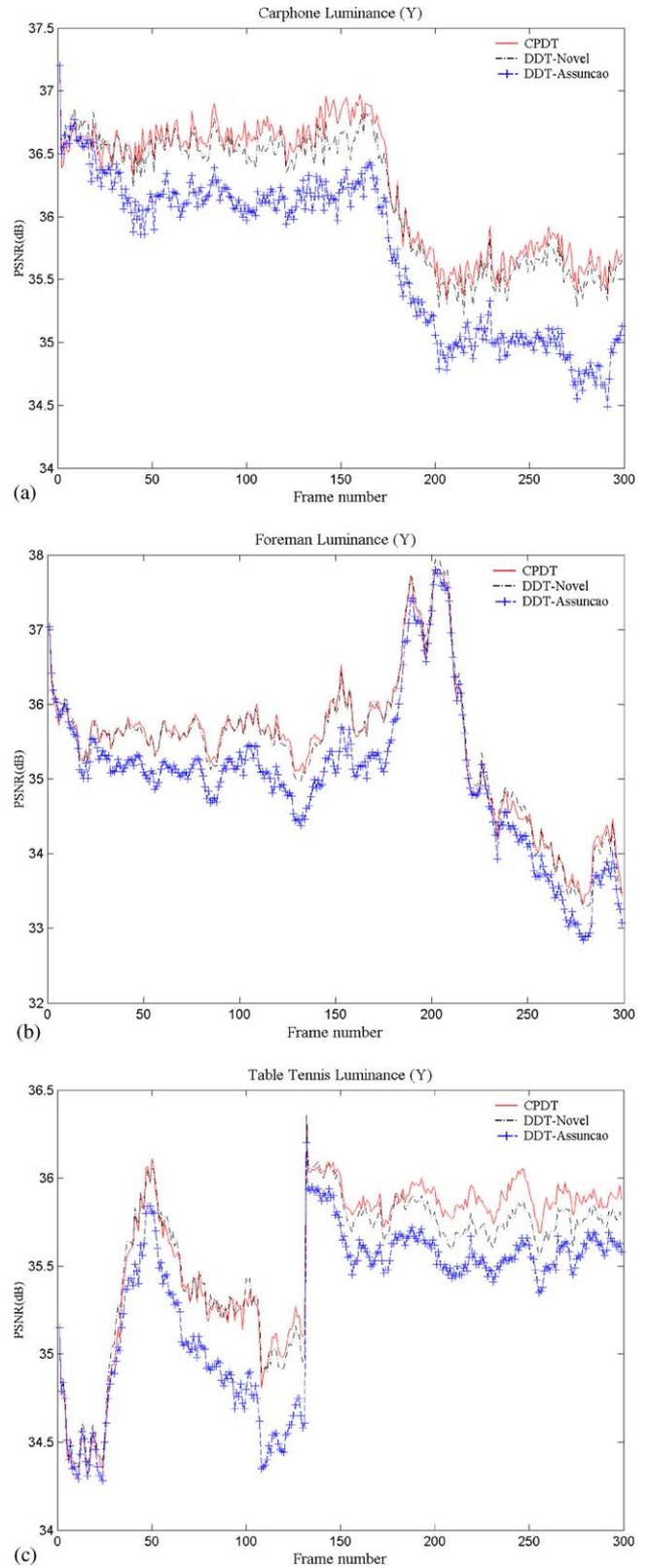


Fig. 5. Performance of CPDT and DDT using different half-pixel filters when Q2=5: (a) Carphone Luminance (Y). (b) Foreman Luminance (Y) and (c) Table Tennis Luminance (Y).

used to reduce the temporal redundancy between the consecutive frames in the video encoder. The following frame depends on the current frame, which the minor distortion errors in the current frame can propagate to the following continuous frames until the intra blocks are presented.

Fig. 5 illustrates the PSNR of Luminance component of the transcoded “Carphone”, “Foreman” and “Table Tennis” test sequences when $Q2 = 5$. Compared with the drift-free CPDT, the transcoded video quality by the proposed filters is almost the same as that by the CPDT. As has been pointed out in [6,10], integer rounding and saturation in DCT/IDCT implementation is one of the reasons that cause the drift error in DCT domain transcoding. The proposed filters merge interpolation and translation into one step, which reduces the number of DCT/IDCT integer rounding and saturation, and hence improves the transcoded video quality. For the Y Components of the sequences, the PSNR of the proposed DDT is just 0.08dB lower than that of CPDT at most. When $Q2 = 10$, the PSNR of some components outperforms the CPDT.

5. Conclusion and future work

DCT-domain transcoding is becoming a more and more important technology for transmitting pre-encoded high bit rate video over heterogeneous networks. In the DCT domain transcoder, DCT domain inverse motion compensation is the most critical module and the key for DCT domain transcoding. In this work, we proposed a novel DCT domain half-pixel filter that can be used in the DCT domain transcoder to implement the inverse motion compensation. The computation complexity is much lower than that of DCT domain transcoders using other filters. Our experimental results show that the DCT domain transcoder using the proposed filter can achieve almost the same quality as CPDT, and may outperforms the CPDT sometimes.

Acknowledgements

This research was supported in part by hi-tech research and development program of China

(2001AA114060). The authors would like to thank Hai-peng Yang of BroadVision for valuable suggestions and comments.

References

- [1] Bormans J, Gelissen J, Perkis A. MPEG-21: the 21st century multimedia framework. *IEEE Signal Processing Magazine* 2003;20:53–62.
- [2] Pereira F, Burnett I. Universal multimedia experiences for tomorrow. *IEEE Signal Processing Magazine* 2003;20:63–73.
- [3] Vetro A, Christopoulos C, Sun HF. Video transcoding architectures and techniques: an overview. *IEEE Signal Processing Magazine* 2003;20:18–29.
- [4] Keesman G, Hellinghuizen R, Hoekseman F, Heideman G. Transcoding of MPEG bitstreams. *Signal Processing: Image Communications* 1996;8:481–500.
- [5] Assunção PAA, Ghanbari M. Transcoding of MPEG-2 video in the frequency domain. *Proceedings of ICASSP1997*; 4. p. 2633–6.
- [6] Youn J, Sun MT, Xin J. Video transcoder architectures for bit rate scaling of H.263 bit streams. *Proceedings of ACM Multimedia 1999*. p. 243–50.
- [7] Assunção PAA, Ghanbari M. A frequency-domain video transcoder for dynamic bit-rate reduction of MPEG-2 bit streams. *IEEE Transactions on Circuits and Systems for Video Technology* 1998;8:953–67.
- [8] Chang SF, Messerschmitt DG. Manipulation and compositing of MC-DCT compressed video. *IEEE Journal on Selected Areas in Communications* 1995;13:1–11.
- [9] Merhav N, Bhaskaran V. Fast algorithms for DCT-domain image down-sampling and for inverse motion compensation. *IEEE Transactions on Circuits and Systems for Video Technology* 1997;7:468–76.
- [10] Song J, Yeo BL. A fast algorithm for DCT-domain inverse motion compensation based on shared information in a macro-block. *IEEE Transactions on Circuits and Systems for Video Technology* 2000;10:767–75.
- [11] ITU-T Recommendation H.263+. Video coding for low bit-rate communications. 1998.
- [12] ISO/IEC 13818-2. Information Technology—Generic coding of moving pictures and associated audio information Part 2—video. 1995.
- [13] ISO/IEC 14496-2. Coding of audio-visual objects-part 2: visual. 2001.
- [14] Shanableh T, Ghanbari M. Transcoding architectures for DCT-domain heterogeneous video transcoding. *Proceedings of ICIP 2001*. p. 433–6.
- [15] Shanableh T, Ghanbari M. Hybrid DCT/pixel domain architecture for heterogeneous video transcoding. *Signal Processing: Image Communications* 2003;18:601–20.
- [16] Cote G, Erol B, Gallant M, Kossentini F. H.263+: Video coding at low bit rates. *IEEE Transactions on Circuits and Systems for Video Technology* 1998;8:849–66.