A DCT-Domain Video Transcoder for Spatial Resolution Downconversion

Yuh-Reuy Lee[1], Chia-Wen Lin[1], and Cheng-Chien Kao[2]

[1]Department of Computer Science and Information Engineering
National Chung Cheng University
Chiayi 621, Taiwan
cwlin@cs.ccu.edu.tw
http://www.cs.ccu.edu.tw/~cwlin
[2] Computer & Communications Research Lab
Industrial Technology Research Institute
Hsinchu 310, Taiwan
cckao@itri.org.tw

Abstract. Video transcoding is an efficient way for rate adaptation and format conversion in various networked video applications. Several transcoder architectures have been proposed to achieve fast processing. Recently, thanks to its relatively low complexity, the DCT-domain transcoding schemes have become very attractive. In this paper, we investigate efficient architectures for video downscaling in the DCT domain. We propose an efficient method for composing downscaled motion vectors and determining coding modes. We also present a fast algorithm to extract partial DCT coefficients in the DCT-MC operation and a simplified cascaded DCT-domain video transcoder architecture.

1 Introduction

With the rapid advance of multimedia and networking technologies, multimedia services, such as teleconferencing, video-on-demand, and distance learning have become more and more popular in our daily life. In these applications, it is often needed to adapt the bit-rate of a coded video bit-stream to the available bandwidth over heterogeneous network environments [1]. Dynamic bit-rate conversions can be achieved using the scalable coding schemes provided in current video coding standards [2]. However, it can only provide a limited number of levels of scalability (say, up to three levels in the MPEG standards) of video quality, due to the limit on the number of enhancement layers. In many networked multimedia applications, a much finer scaling capability is desirable. Recently, fine-granular scalable (FGS) coding schemes have been proposed in the MPEG-4 standard to support a fine bit-rate adaptation and limited temporal/spatial format conversions. However, the video decoder requires additional functionality to decode the enhancement layers in the FGS encoded bit-streams.

Video transcoding is a process of converting a previously compressed video bit-stream into another bit-stream with a lower bit-rate, a different display format (e.g.,

downscaling), or a different coding method (e.g., the conversion between H.26x and MPEGx, or adding error resilience), etc. To achieve the goal of universal multimedia access (UMA), the video contents need to be adapted to various channel conditions and user equipment capabilities. Spatial resolution reduction [5-9] is one of the key issues for providing UMA in many networked multimedia applications. In realizing transcoders, the computational complexity and picture quality are usually the two most important concerns and need to be traded off to meet various requirements in practical applications. The computational complexity is very critical in real-time applications.

A straightforward realization of video transcoders is to cascade a decoder followed by an encoder as shown in Fig. 1. This cascaded architecture is flexible and can be used for bit-rate adaptation and spatial and temporal resolution-conversion without drift. It is, however, very computationally intensive for real-time applications, even though the motion-vectors and coding-modes of the incoming bit-stream can be reused for fast processing.

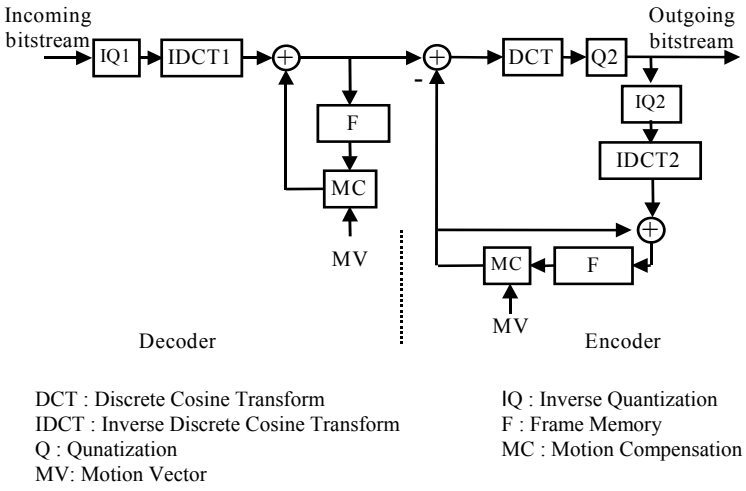


Fig. 1. Cascaded pixel-domain transcoder

For efficient realization of video transcoders, several fast architectures have been proposed in the literature [2-11, 14-15]. In [10], a simplified pixel-domain transcoder (SPDT) was proposed to reduce the computational complexity of the cascade transcoder by reusing motion vectors and merging the decoding and encoding process and eliminating the IDCT and MC (Motion Compensation) operations. [11] proposed a simplified DCT-domain transcoder (SDDT) by performing the motion-compensation in the DCT-domain [12] so that no DCT/IDCT operation is required. This simplification imposes a constraint that this architecture cannot be used for spatial or temporal resolution conversion and GOP structure conversion, that requires new motion vectors. Moreover, it cannot adopt some useful techniques, which may need to change the motion vectors and/or coding modes, for optimizing the performance in transcoding such as motion vector refinement [14]. The cascaded

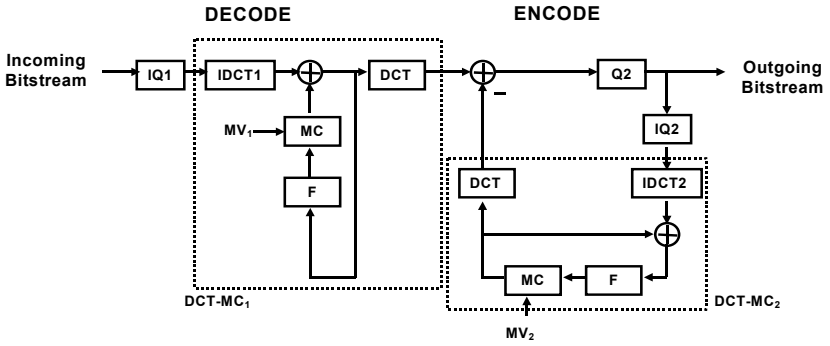
pixel-domain transcoder is drift-free and does not have the aforementioned constraints. However, its computational complexity is still high though the motion estimation doesn't need to be performed.

In this paper, we investigate efficient realizations of video downscaling in the DCT domain. We also propose efficient methods for composing downscaled motion vectors and determining coding modes. We also present a fast algorithm to extract partial DCT coefficients in the DCT-MC operation and a simplified cascaded DCT-domain video transcoder architecture.

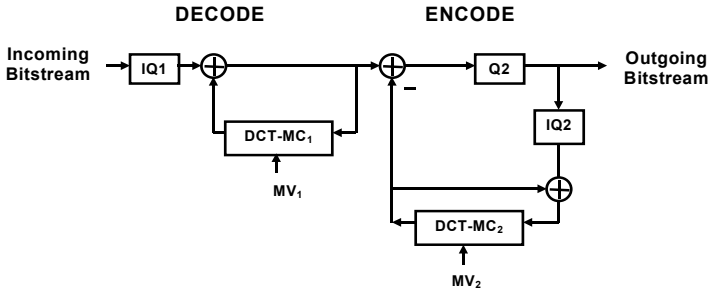
The rest of this paper is organized as follows. In section 2, we discuss existing transcoder architectures, especially the DCT-domain transcoder for spatial downscaling. In section 3, we investigate efficient methods for implementing downsizing and motion compensation in the DCT domain. Finally, the result is summarized in section 4.

2 Cascaded DCT-Domain Transcoder For Spatial Resolution Downscaling

To overcome the constraints of the SDDT, we propose to use the Cascaded DCT-Domain Transcoder (CDDT) architecture which first appeared in [6]. The CDDT can avoid the DCT and IDCT computations required in the pixel-domain architectures as well as preserve the flexibility of changing motion vectors, coding modes as in the CPDT. Referring to Figure 1, by using the linearity property of the DCT transform (i.e., DCT(A+B) = DCT(A) + DCT(B)), the DCT block can be moved out from the encoder loop to form the equivalent architecture in Fig. 2(a). Each combination of IDCT, pixel-domain motion compensation, and DCT as enclosed by the broken lines is equivalent to a DCT-domain MC (DCT-MC) peration. Therefore we can derive the equivalent cascaded DCT-domain transcoder architecture as shown in Fig. 2(b).



(a)



(b)

Fig. 2. (a) An equivalent transform of the cascaded pixel domain transcoder; (b) cascaded DCT-domain transcoder

The MC-DCT operation shown in Fig. 3 can be interpreted as computing the coefficients of the target DCT block B from the coefficients of its four neighboring DCT blocks, B_i, i = 1 to 4, where B = DCT(**b**) and B_i = DCT(**b**$_i$) are the 8×8 blocks of the DCT coefficients of the associated pixel-domain blocks **b** and **b**$_i$ of the image data. A close-form solution to computing the DCT coefficients in the DCT-MC operation was firstly proposed in [12] as follows.

$$B = \sum_{i=1}^{4} H_{h_i} B_i H_{w_i} \qquad (\mathbf{1})$$

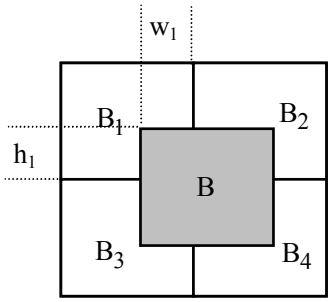
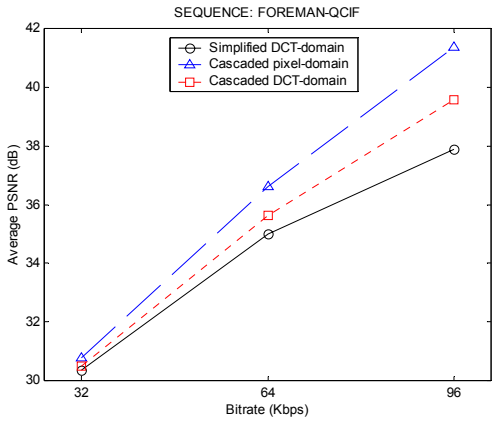


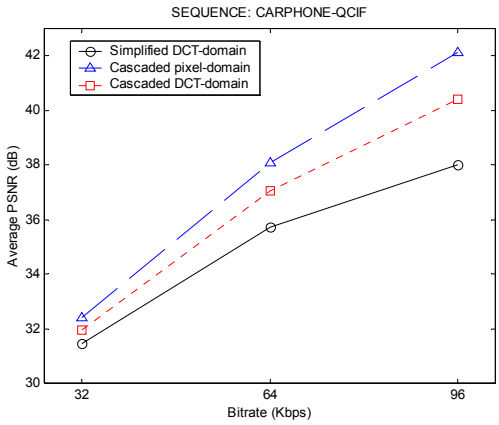
Fig. 3. DCT-domain motion compensation

where w_i and $h_i \in \{1,2,\ldots 7\}$. H_{h_i} and H_{w_i} are constant geometric transform matrices defined by the height and width of each subblock generated by the intersection of **b**$_i$ with **b**. Direct computation of Eq. (1) requires 8 matrix multiplications and 3 matrix additions. Note that, the following equalities holds for the geometric transform matrices: $H_{h_1} = H_{h_2}$, $H_{h_3} = H_{h_4}$, $H_{w_1} = H_{w_3}$, and $H_{w_2} = H_{w_4}$. Using these equalities, the number of operations in Eq. (1) can be reduced to 6 matrix multiplications and 3 matrix additions. Moreover, since H_{h_i} and

H_{w_i} are deterministic, they can be pre-computed and then pre-stored in memory. Therefore, no additional DCT computation is required for the computation of Eq. (1).



(a)



(b)

Fig. 4. Performance comparison of average PSNR with three different transcoders. the incoming sequence was encoded at 128 kb/s, and transcoded to 96 kb/s, 64 kb/s, and 32 kb/s, respectively for: (a) "foreman" sequence; (b) "carphone" sequence

We compare the PSNR performance of CPDT, SDDT, and CDDT in Fig. 4. Two test sequences: "foreman" and "carphone" were used for simulation. Each incoming sequence was encoded at 128 Kbps and transcoded into 96, 64,and 32 Kbps, respectively. It is interesting to observe that, though all the three transcoding architectures are mathematically equivalent by assuming that motion compensation is a linear operation, DCT and IDCT can cancel out each other, and DCT/IDCT has distributive property, the performance are quite different. The CPDT architecture outperforms the other two.

Though the performance of the DCT-domain transcoders is not as ggod as the SPDT, the main advantage of the DCT-domain transcoders lies on the existing efficient algorithms for fast DCT-domain transcoding [10,11,18,19], which make them very attractive. For spatial resolution downscaling, we propose to use the cascaded DCT-domain transcoder shown in Fig. 5. This transcoder can be divided into four main functional blocks: decoder, downscaler, encoder, and MV composer, where all the operations are done in the DCT domain. In the following, we will investigate efficient schemes for DCT-domain downscaling.

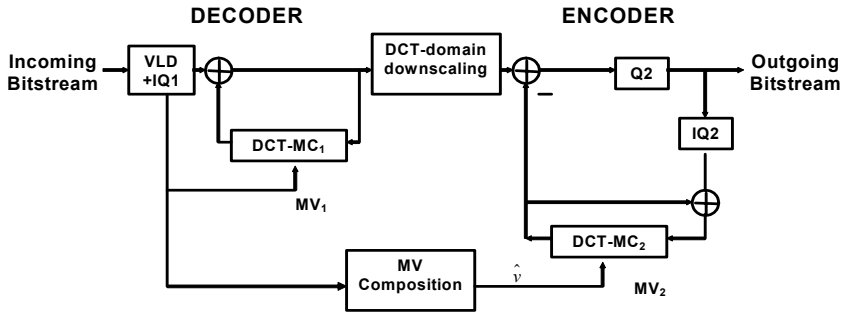


Fig. 5. Proposed DCT-domain spatial resolution down-conversion transcoder

3 Algorithms for DCT-Domain Spatial Resolution Downscaling

3.1 DCT-Domain Motion Compensation with Spatial Downscaling

Consider the spatial downscaling problem illustrated in Fig. 6, where $\mathbf{b}_1$, $\mathbf{b}_2$, $\mathbf{b}_3$, $\mathbf{b}_4$ are the four original 8×8 blocks, and $\mathbf{b}$ is the 8×8 downsized block. In the pixel domain, the downscaling operation is to extract one representative pixel (e.g., the average) out of each 2x2 pixels. In the following, we will discuss two schemes for spatial downscaling in the DCT domain which may be adopted in our DCT-domain downscaling transcoder.

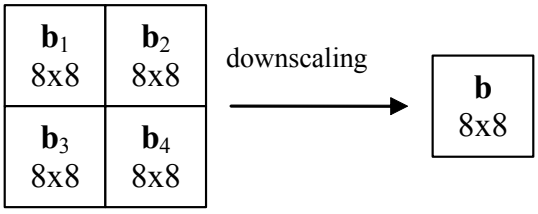


Fig. 6. Spatial resolution down-conversion

A. Filtering + Subsampling

Pixel averaging is the simplest way to achieving the downscaling, which can be implemented using the bilinear interpolation expressed below [6,14].

$$\mathbf{b} = \sum_{i=1}^{4} \mathbf{h}_i \mathbf{b}_i \mathbf{g}_i \tag{2}$$

The filter matrices, $\mathbf{h}_i$ and $\mathbf{g}_i$, are

$$\begin{cases} \mathbf{h}_1 = \mathbf{h}_2 = \mathbf{g}_1^t = \mathbf{g}_3^t = \begin{bmatrix} \mathbf{q}_{4\times 8} \\ \mathbf{0}_{4\times 8} \end{bmatrix} \\ \mathbf{h}_3 = \mathbf{h}_4 = \mathbf{g}_2^t = \mathbf{g}_4^t = \begin{bmatrix} \mathbf{0}_{4\times 8} \\ \mathbf{q}_{4\times 8} \end{bmatrix} \end{cases} \tag{3}$$

where

$$\mathbf{q}_{4\times 8} = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 \end{bmatrix}$$

, and $\mathbf{0}_{4\times 8}$ is a 4×8 zero matrix.

The above bilinear interpolation procedure can be performed in the DCT domain directly to obtain the DCT coefficients of the downsized block (i.e., $B = \mathrm{DCT}(\mathbf{b})$) as follows:

$$B = \sum_{i=1}^{4} \mathrm{DCT}(\mathbf{h}_i)\,\mathrm{DCT}(\mathbf{b}_i)\,\mathrm{DCT}(\mathbf{g}_i) = \sum_{i=1}^{4} H_i B_i G_i \tag{4}$$

Other filtering methods with a larger number of filter taps in $\mathbf{h}_i$ and $\mathbf{g}_i$ may achieve better performance than the bilinear interpolation. However, the complexity may increase in pixel-domain implementations due to the increase in the filter length. Nevertheless, the DCT-domain implementation cost will be close to the bilinear interpolation, since in Eq. (4) H_i and G_i can be precomputed and stored, thus no extra cost will be incurred.

B. DCT Decimation

It was proposed in [13,14] a DCT decimation scheme that extracts the 4x4 low-frequency DCT coefficients from the four original blocks $\mathbf{b}_1$-$\mathbf{b}_4$, then performs 4x4 IDCT to obtain four 4x4 subblocks, and finally combine the four subblocks into an 8x8 blocks. This approach was shown to achieve significant performance improvement over the filtering schemes [14]. [8] interpreted the DCT decimation as basis vectors resampling, and presented a compressed-domain approach for the DCT decimation as described below.

Let B_1, B_2, B_3, and B_4, represent the four original 8×8 blocks; $\hat{B}_1$, $\hat{B}_2$, $\hat{B}_3$ and $\hat{B}_4$ be the four 4×4 low-frequency sub-blocks of B_1, B_2, B_3, and B_4, respectively; $\hat{\mathrm{b}}_i = \mathrm{IDCT}(\hat{B}_i)$, i = 1, …, 4. Then $\hat{\mathbf{b}} = \begin{bmatrix} \hat{\mathbf{b}}_1 & \hat{\mathbf{b}}_2 \\ \hat{\mathbf{b}}_3 & \hat{\mathbf{b}}_4 \end{bmatrix}_{8\times 8}$ is the downscaled version of $\mathbf{b} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 \\ \mathbf{b}_3 & \mathbf{b}_4 \end{bmatrix}_{16\times 16}$. To compute $\hat{B} \overset{def}{=} \mathrm{DCT}(\hat{b})$ from $\hat{B}_1$, $\hat{B}_2$, $\hat{B}_3$ and $\hat{B}_4$, we can use the following expression:

$$
\begin{aligned}
\hat{B} &= T\hat{b}T^t \\
&= \begin{bmatrix} T_L & T_R \end{bmatrix} \begin{bmatrix} \hat{\mathbf{b}}_1 & \hat{\mathbf{b}}_2 \\ \hat{\mathbf{b}}_3 & \hat{\mathbf{b}}_4 \end{bmatrix} \begin{bmatrix} T_L^t \\ T_R^t \end{bmatrix} \\
&= \begin{bmatrix} T_L & T_R \end{bmatrix} \begin{bmatrix} T_4^t \hat{B}_1 T_4 & T_4^t \hat{B}_2 T_4 \\ T_4^t \hat{B}_3 T_4 & T_4^t \hat{B}_4 T_4 \end{bmatrix} \begin{bmatrix} T_L^t \\ T_R^t \end{bmatrix} \\
&= (T_L T_4^t)\hat{B}_1 (T_L T_4^t)^t + (T_L T_4^t)\hat{B}_2 (T_R T_4^t)^t + (T_R T_4^t)\hat{B}_3 (T_L T_4^t)^t \\
&\quad + (T_R T_4^t)\hat{B}_4 (T_R T_4^t)^t
\end{aligned}
\tag{5}
$$

In addition to the above formulation, [8] also proposed a decomposition method to convert Eq. (5) into a new form so that matrices in the matrix multiplications become more sparse to reduce the computation.

3.2 Motiov Vector Composition and Mode Decision

After downscaling, the motion vectors need to be re-estimated and scaled to obtain a correct value. Full-rang motion re-estimation is computationally too expensive, thus not suited to practical applications. Several methods were proposed for fast composing the downscaled MVs based on the motion information of the original frame [7,14,17].

In [14], three methods for composing new motion vectors for the downsized video were compared: median filtering, averaging, and majority voting. It was shown in [14] that the median filtering scheme outperforms the other two. We propose to generalize the media filtering scheme to find the activity-weighted median of the four original vectors: v_1, v_2, v_3, v_4. In our method the distance between each vector and the rest is calculated as the sum of the activity-weighted distances as follows:

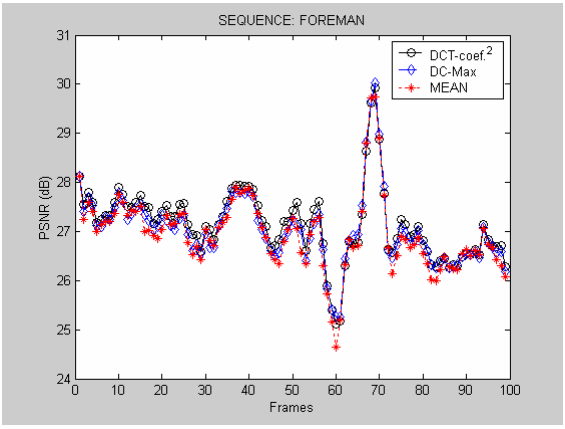
$$
d_i = \frac{1}{\mathrm{ACT}_i} \sum_{\substack{j=1 \\ j\neq i}}^{4} \left\| v_i - v_j \right\| \tag{6}
$$

where the MB activity can be the squared or absolute sum of DCT coefficients, the number of nonzero DCT coefficients, or simply the DC value. In our method, we

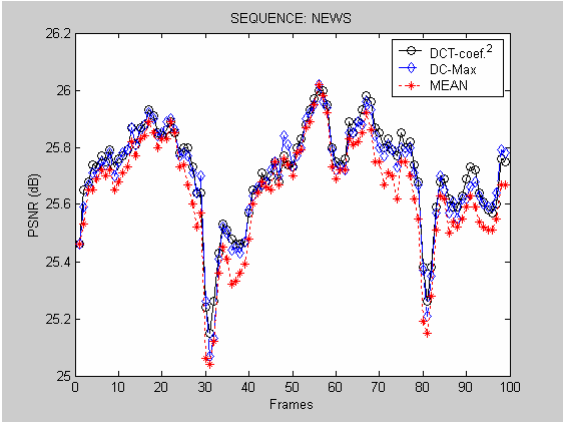
adopted the squared sum of DCT coefficients of MB as the activity measure. The activity-weighted median is obtained by finding the vector with the least distance from all. That is

$$v = \frac{1}{2}\arg \min_{v_i \in \{v_1, v_2, v_3, v_4\}} d_i \tag{7}$$

Fig. 7 shows the PSNR comparison of three motion vector composition scheme: activity-weighted median (denoted by DCT-coef^2), the maximum DC method in [17] (denoted by DC-Max), and the average vector scheme (denoted by MEAN). The simulation result that the activity-weighted media outperforms the other two.



(a)



(b)

Fig. 7. PSNR performance comparison of three motion vector composition schemes. The input sequences: (a) "foreman" sequence; (b) "news" sequence, are transcoder form 256 Kbps, 10fps into 64 Kbps, 10fps

After the down-conversion, the MB coding modes also need to be re-determined. In our method, the rules for determining the code modes are as follows:

(1) If at least one of the four original MBs is intra-coded, then the mode for the downscaled MB is set as Intra.

(2) If all the four original MBs are inter-coded, the resulting downscaled MB will also be inter-coded.

(3) If at least one original MB is skipped, and the reset are inter-coded, the resulting downscaled MB will be inter-coded.

(4) If all the four original MBs are skipped, the resulting downscaled MB will also be skipped. Note, the motion vectors of skipped MBs are set to zero.

3.3 Computation Reduction in Proposed Cascaded DCT-Domain Downscaling Transcoder

In Fig. 4, the two DCT-MCs are the most expensive operation. In our previous work [18], we showed that for each 8×8 DCT block, usually only a small number of low-frequency coefficients are significant. Therefore we can use the fast significant coefficients extraction scheme proposed in [18] to reduce the computation for DCT-MC. The concept of significant coefficients extraction is illustrated in Fig. 8, where only partial coefficients (i.e., $n \leq 8$) of the target block need to be computed.

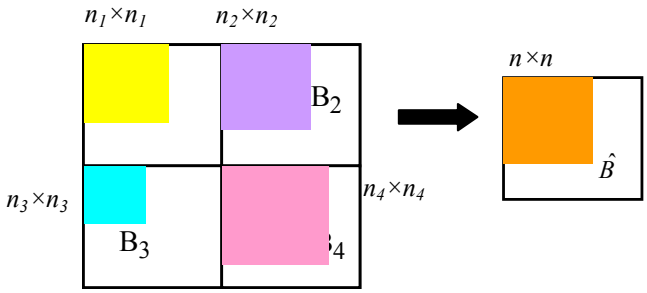


Fig. 8. Computation reduction for DCT-MC using significant coefficients extraction

The DCT-domain down-conversion transcoder can be further simplified by moving the downscaling operation into the decoder loop so that the decoder only needs to decode one quarter of the original picture size. Fig. 9 depicts the proposed simplified architecture. With this architecture both the computation and memory cost will be reduced significantly. However, similar to the down-conversion architectures in [20,21], this simplified transcoder will result in drift errors due to the mismatch in the frame stores between the front-end encoder and the reduced-resolution decoder loop of the transcoder. Several approaches have been presented to mitigate the drift problem [20,21], which may introduce some extra complexity. In MPEG video, since the drift in B frames will not result in error propagation, a feasible approach is to

perform full-resolution decoding for I and P frames, and quarter-resolution decoding for B frames.

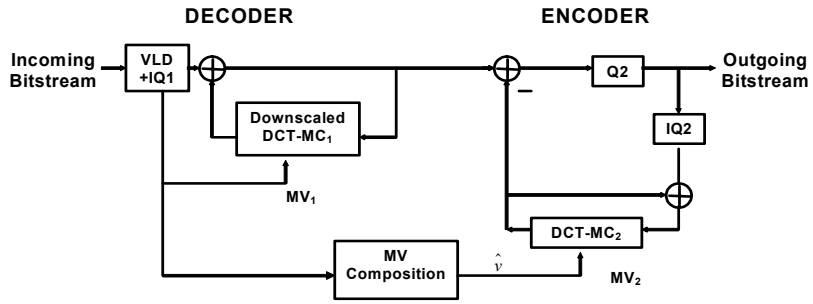


Fig. 9. Simplified DCT-domain spatial resolution down-conversion transcoder

4 Summary

In this paper, we presented architectures for implementing spatial downscaling video transcoders in the DCT domain and efficient methods for implementing DCT-domain motion compensation with downscaling. We proposed an activity-weighted median filtering scheme for composing the downscaled motion vectors, and also a method for determining the decision mode. We have also presented efficient schemes for reducing the computational cost of the downscaling trancoder.

References

1. Moura, J., Jasinschi, R., Shiojiri-H, H., Lin, C.: Scalable Video Coding over Heterogeneous Networks. *Proc. SPIE* 2602 (1996) 294-306
2. Ghanbari, M.: Two-Layer Coding of Video Signals for VBR Networks. *IEEE J. Select. Areas Commun.* 7 (1989) 771-781
3. Sun, H., Kwok, W., Zdepski, J. W.: Architecture for MPEG Compressed Bitstream Scaling. *IEEE Trans. Circuits Syst. Video Technol*. 6 (1996) 191-199
4. Eleftheriadis, A. Anastassiou, D.: Constrained and General Dynamic Rate Shaping of Compressed Digital Video. *Proc. IEEE Int. Conf. Image Processing* (1995)
5. Hu, Q., Panchanathan, s.: Image/Video Spatial Scalability in Compressed Domain. *IEEE Trans. Ind. Electron.* 45 (1998) 23–31
6. Zhu, W., Yang, K., Beacken, M.: CIF-to-QCIF Video Bitstream Down-Conversion in the DCT Domain. *Bell Labs technical journal* 3 (1998) 21-29
7. Yin, P., Wu, M., Liu, B.: Video Transcoding by Reducing Spatial Resolution. *Proc. IEEE Int. Conf. Image Processin* (2000)
8. R. Dugad and N. Ahuja, "A Fast Scheme for Image Size Change in the Compressed Domain. *IEEE Trans. Circuit Syst. Video Technol*. 11 (2001) 461-474

9. N. Merhav and V. Bhaskaran, “Fast Algorithms for DCT-Domain Image Down-Sampling and for Inverse Motion Compensation. *IEEE Trans. Circuits Syst. Video Technol.* 7 (1997) 468–476
10. Keesman, g. *et al*.: Transcoding of MPEG Bitstreams. *Signal Processing: Image Commun*. 8 (1996) 481-500
11. Assuncao, P. A. A., Ghanbari, M.: A Frequency-Domain Video Transcoder for Dynamic Bit-rate Reduction of MPEG-2 Bit Streams. *IEEE Trans. Circuits Syst. Video Technol*. 8 (1998) 953-967
12. Chang, S. F., Messerschmitt, D. G.: Manipulation and Compositing of MC-DCT Compressed Video. *IEEE J. Select. Areas Commun.* (1995) 1-11
13. Tan, K. H., Ghanbari, M.: Layered Image Coding Using the DCT Pyramid. *IEEE Trans. Image Processing* 4 (1995) 512-516
14. Shanableh T., Ghanbari, M.: Heterogeneous Video Transcoding to Llower Spatio-temporal Resolutions and Different Encoding Formats. *IEEE Trans. on Multimedia* 2 (2000) 101-110
15. Shanableh T., Ghanbari, M.: Transcoding Architectures for DCT-Domain Heterogeneous Video Transcoding. *Proc. IEEE Int. Conf. Image Processing* (2001)
16. Seo, K., Kim J.: Fast Motion Vector Refinement for MPEG-1 to MPEG-4 Transcoding with Spatial Down-sampling in DCT Domain. *Proc. IEEE Int. Conf. Image Processing* (2001) 469-472
17 Chen, M.-J., M.-C. Chu, M.-C., Lo, S.-Y.: Motion Vector Composition Algorithm for Spatial Scalability in Compressed Video. *IEEE Trans. Consumer Electronics* 47 (2001) 319-325
18 Lin, C.-W., Lee, Y.-R.: Fast Algorithms for DCT Domain Video Transcoding. *Proc. IEEE Int. Conf. Image Processing* (2001) 421-424
19 Song, J., Yeo, B.-L.: A Fast Algorithm for DCT-Domain Inverse Motion Compensation based on Shared Information in a Macroblock. *IEEE Trans. Circuits Syst. Video Technol*. 10 (2000) 767-775
20 Vetro, A., Sun, H., DaGraca, P., Poon, T.: Minimum Drift Architectures for Three-layer Scalable DTV Decoding. *IEEE Trans. Consumer Electronics* 44 (1998)
21 Vetro, A., Sun, H.: Frequency Domain Down-Conversion Using an Optimal Motion Compensation Scheme. *Int'l Journal of Imaging Systems & Technology* 9 (1998)