LOW-COMPLEXITY DCT-DOMAIN VIDEO TRANSCODERS FOR ARBITRARY-SIZE DOWNSCALING

Yuh-Reuy Lee, Chia-Wen Lin*, Sung-Hung Yeh[+], and Yung-Chang Chen*[+]*

[*]Department of Computer Science & Information Engineering.
National Chung Cheng University, Chiayi 621, Taiwan
[+]Department of Electrical Engineering.
National Tsing Hua University, Hsinchu 310, Taiwan

ABSTRACT

In this paper, we propose efficient techniques and architectures for realizing spatial-downscaling transcoders in the DCT domain. We present efficient DCT-domain methods for arbitrary-size downscaling and upscaling. We show that, by integrating the downscaling process into the DCT-domain motion compensation (DCT-MC) operation for B-frames, the computation of DCT-MC and downscaling can be significantly reduced, leading to a simplified cascaded DCT-domain downscaling transcoder (CDDT) without introducing extra quality degradation. We also propose another scheme to further reduce the computation and storage cost which may introduce drifting errors. Experimental results show that the two proposed schemes can achieve significant computation reduction when compared with the original CDDT without any degradation or with introducing acceptable quality degradation, respectively.

1. INTRODUCTION

Networked multimedia services, such as video on demand, video streaming, and distance learning, have been emerging in various network environments. These multimedia services usually use pre-encoded videos for transmission. The heterogeneity of present communication networks and user devices poses difficulties in delivering theses bitstreams to the receivers. The sender may need to convert one preencoded bitstream into a lower bit-rate or lower resolution version to fit the available channel bandwidths, the screen display resolutions, or even the processing powers of diverse clients [1]. Many practical applications such as video resolution conversions from HDTV to digital TV [8], from DVD to VCD (i.e., MPEG-2 -> MPEG-1) and from MPEG-1/2 to MPEG-4 involve such spatial-resolution, format, and bit-rate conversions. Such conversions may involve resolution conversion with an integer or even a non-integer scaling factor (e.g., HDTV->SDTV and CIF-> sub QCIF). Resolution downscaling is also an efficient means of maintaining the perceptual quality of a video with a reduced resolution when transmitting the video over low bit-rate channels [3]. In these applications, resolution resizing with an arbitrary scaling factor is desirable [2,3]. Although scalable coding schemes in current coding standards can achieve dynamic bitrate or resolution conversions to support heterogeneous video communications. They, however, usually just provide a very limited support of heterogeneity of bitrates and resolutions (e.g., MPEG-2 and H.263+), or introduce significantly higher complexity at the client decoder (e.g., MPEG-4 Fine-Grannular Scalability).

Video transcoding [1-6] is a process of converting a previously compressed video bit-stream into another bit-stream with a lower bitrate, a different display format (e.g., downscaling), or a different coding method, etc. It is considered an efficient means of achieving fine and dynamic adaptation of bitrates, resolutions, and formats. In realizing transcoders, the computational complexity and picture quality are usually the two most important concerns. A straightforward realization of video transcoders is to decode a compressed video into pixel values then re-encode it with another bitrate and/or format. This cascaded pixel-domain transcoder is flexible and can be used for bitrate adaptation, spatial and temporal resolution-conversion without drift. It is, however, 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.

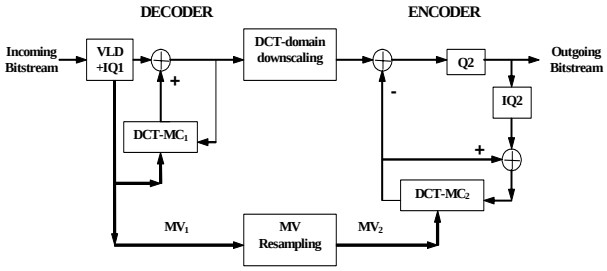

Fig. 1.Cascaded DCT-domain downscaling transcoder (CDDT).

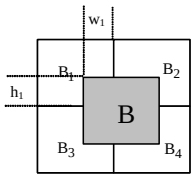

Fig. 2. DCT-domain motion compensation.

Recently, DCT-domain transcoding schemes [4-6] have become very attractive because they can avoid the DCT and IDCT computations as well as several efficient schemes were developed for implementing the DCT-MC [4,8]. The simplified DCT-domain transcoder proposed in [4], however, cannot be used for spatial/temporal downscaling because it has to use at the encoding stage the same motion vectors decoded from the incoming video. A cascaded DCT-domain downscaling transcoder (CDDT) architecture was first proposed in [5] as depicted in Fig. 1. The CDDT integrates the DCT and IDCT computations and pixel-domain motion compensation into a single DCT-domain motion compensation (DCT-MC) operation as depicted in Fig. 2. The DCT-MC operation is to compute 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 DCT blocks of the associated pixel blocks **b** and **b**$_i$. The relation of B and B_i can be expressed as [7]

$$B = \sum_{i=1}^{4} H_{h_i} B_i H_{w_i} \tag{1}$$

where w_i and $h_i \in \{0,1,\ldots 7\}$. H_{h_i} and H_{w_i} are constant geometric transform matrices defined by the height and width of each sub-block generated by the intersection of **b**$_i$ with **b**.

In [5], a bilinear filtering scheme was used for downscaling the spatial resolution in the DCT domain. A more efficient DCT-domain downscaling scheme, named DCT decimation, was proposed in [6] for image downscaling and later adopted in video transcoding.

In this paper, we propose efficient schemes to arbitrarily resizing video in the DCT domain. We present a DCT-domain video downscaling transcoder architecture. We also propose methods of computation reduction of the transcoder without introducing or with minor performance degradation.

2. DCT-DOMAIN ARBITRARY DOWNSCALING

An arbitrary-size downscaling system with a scaling factor of L/M (can be non-integer) can be viewed as the cascade of systems as shown in Fig. 3. The first stage of the process involves upsampling the input sequence by a factor L. This is done by adding L-1 zeros between each input sample. The second stage of the process is the low-pass filtering which satisfies the filtering parameters for both the interpolation and decimation operations to account for the frequency content added to the original signal via upsampling and to prevent aliasing from occurring in the downsampling stage. Downsampling by a factor of M is the final stage and is realized by keeping one out of every M samples to create the output sequence.

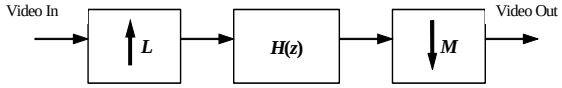

Fig. 3. Arbitrary-size frame resizing.

The typical downscaling scheme in a pixel-domain transcoder involves inverse DCT, filtering, downsampling, and forward DCT. In [6], an efficient DCT decimation scheme was proposed for spatial downscaling in the DCT domain. This scheme extracts the 4x4 low-frequency DCT coefficients from the four 8×8 original blocks **b**$_1$-**b**$_4$, then combines the four 4×4 sub-blocks into an 8×8 block. Note that if the DCT were replaced by a DFT in these operations, this corresponds to a regular filtered, downsampling operation. It is shown in [6] that using a DCT instead of DFT, can also give a good approximation to the downscaling scheme that avoids aliasing. Let B_1, B_2, B_3, and B_4, represent the four original 8×8 DCT blocks; $\hat{B}_1$, $\hat{B}_2$, $\hat{B}_3$ and $\hat{B}_4$ 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 $\hat{B} = \mathrm{DCT}(\hat{b})$ directly from $\hat{B}_1$, $\hat{B}_2$, $\hat{B}_3$, and $\hat{B}_4$, we can use the following expression:

$$\begin{aligned} \hat{B} = T_8 \hat{\mathbf{b}} T_8^t &= \begin{bmatrix} T_L & T_R \end{bmatrix} \begin{bmatrix} T_4^t \hat{B}_1 T_4 & T_4^t \hat{B}_2 T_4 \\ T_4^t \hat{B}_3 T_4 & T_4^t \hat{B}_4 T_4 \end{bmatrix} \begin{bmatrix} T_L^t \\ T_R^t \end{bmatrix} \\ &= (T_L T_4^t)\hat{B}_1 (T_L T_4^t)^t + (T_L T_4^t)\hat{B}_2 (T_R T_4^t)^t + \\ &\quad (T_R T_4^t)\hat{B}_3 (T_L T_4^t)^t + (T_R T_4^t)\hat{B}_4 (T_R T_4^t)^t \end{aligned} \tag{2}$$

where T_4 is a 4-point DCT transform kernel matrix, and T_R and T_L and the four right and four left columns of T_8, the 8-point DCT transform matrix.

2.1 DCT-domain spatial resolution downscaling

The method in [6], however, only deals with the downscaling with scaling factors of powers of 2 (say, 2, 4, and 8). In the following, we generalize the method to achieve arbitrary-size downscaling with factors other than powers of 2 (say, 3, 5, and 7). For the downscaling by m×n ($2 \le m,n \le 8$), our downscaling method is summarized as follows:

(1) Retain a proper number of low-frequency coefficients of DCT block $B_{k,l}$ in each dimension, and drop the remaining high-frequency coefficients. Each block will be reduced to a smaller sub-block $\hat{B}_{k,l}$ with a size of $m'\times n'$, where $m'=g(m)$ and $n'=g(n)$, where $g(r) = \lceil 8/r \rceil$ is the smallest integer not less than $8/r$, and r is the scaling factor. Table 1 shows the suggested number of preserved coefficients for each downscaling/upscaling ratio ranging from 2 to 8.

Table 1. Number of preserved coefficients in the downscaling/upscaling process

Downscaling Ratio: r	2	3	4	5	6	7	8
Preserved number of Coefficients: $g(r)$	4	3	2	2	2	2	1
$r\times g(r)$	8	9	8	10	12	14	8

For example if the downscaling ratio is 3×5, from Table 1, the numbers of preserved low-frequency coefficients at the horizontal and vertical dimensions are $m'=g(3)=3$ and $n'=g(5)=2$, respectively. This will result in a sub-block with a size of 3×2.

(2) Transform each sub-block $\hat{B}_{k,l}$ to the spatial domain using $\hat{\mathbf{b}}_{k,l} = T^t_{m'\times n'} \hat{B}_{k,l} T_{m'\times n'}$.

(3) Concatenate $m\times n$ subblocks to form an $M\times N$ block $\tilde{\mathbf{b}}$, where M=m×m' and N=n×n'.

$$\tilde{\mathbf{b}} = \begin{bmatrix} \hat{\mathbf{b}}_{1,1} & \cdots & \hat{\mathbf{b}}_{m,1} \\ \vdots & \ddots & \vdots \\ \hat{\mathbf{b}}_{1,n} & \cdots & \hat{\mathbf{b}}_{m,n} \end{bmatrix} = \begin{bmatrix} T^t_{m'} \hat{B}_{1,1} T_{n'} & \cdots & T^t_{m'} \hat{B}_{m,1} T_{n'} \\ \vdots & \ddots & \vdots \\ T^t_{m'} \hat{B}_{1,n} T_{n'} & \cdots & T^t_{m'} \hat{B}_{m,n} T_{n'} \end{bmatrix} \tag{3}$$

(4) Compute $\tilde{B} = \mathrm{DCT}_{M\times N}(\tilde{\mathbf{b}}) = T_M \tilde{\mathbf{b}} T_N^t$

(5) Extract the 8×8 low-frequency coefficients of $\tilde{B}$ to form an 8×8 DCT block $\hat{B}$.

$$\hat{B}[k,l] = \frac{8}{\sqrt{MN}} \tilde{B}[k,l] \quad \text{for } k = 0,1, \ldots, 7 \tag{4}$$

Note that the scaling factor $8/\sqrt{MN}$ can be absorbed into the quantization step for computation saving.

Note that, as shown in Eq. (5), steps (2)-(4) in the above procedure can be combined into a single DCT-domain operation

without the need of transforming the DCT blocks into pixel blocks.

$$\widetilde{B} = T_M \begin{bmatrix} T_{m'}^t \hat{B}_{1,1} T_{n'} & \cdots & T_{m'}^t \hat{B}_{m,1} T_{n'} \\ \vdots & \ddots & \vdots \\ T_{m'}^t \hat{B}_{1,n} T_{n'} & \cdots & T_{m'}^t \hat{B}_{m,n} T_{n'} \end{bmatrix} T_N^t \qquad (5)$$

2.2 DCT-domain spatial resolution upscaling

As discussed above, the proposed DCT-domain downscaling operation first converts a set of 8×8 DCT blocks into a temporary DCT block $\widetilde{B}$, and then extracts the final downscaled version $\hat{B}$ from $\widetilde{B}$. Since the upscaling is basically the reverse operation of the downscaling, it involves the procedures that convert $\hat{B}$ into the low-frequency portion of a same number of 8×8 DCT blocks as in the downscaling. The procedures to upscale $\hat{B}$ by the ratio m×n are summarized below.

(1) Expend each 8×8 downscaled block $\hat{B}$ back to its first down-sampled block $\widetilde{B}$ with size M×N, where M=g(m)×m and N=$g(n)$×n by padding M-8 columns and N-8 rows of zero coefficients. The function $g(\cdot)$ is shown in Table 1.

For example, if the upscaling ratio is 3×2, then N=g(3)×3=3×3=9 , and M=g(2)×2=4×2=8. Then we expend $\hat{B}$ to $\widetilde{B}$ by zero padding and coefficient scaling as shown in Eq. (6). Again, the scaling factor can be absorbed into the quantization operation in the transcoder.

$$\widetilde{B}[k,l] = \begin{cases} \dfrac{\sqrt{MN}}{8} \hat{B}[k,l] & \text{for } k,l = 0,1,\ldots,7 \\ 0 & k,l \geq 8 \end{cases} \qquad (6)$$

(2) Perform M×N-point IDCT to obtain the pixel block $\tilde{\mathbf{b}} = \mathrm{IDCT}_{M\times N}(\widetilde{B}) = T_M^t \widetilde{B} T_N$.

(3) Divide $\tilde{\mathbf{b}}$ into sub-blocks with a size of $g(m)$×$g(n)$ as follows:

$$\tilde{\mathbf{b}} = \begin{bmatrix} \hat{\mathbf{b}}_{1,1} & \cdots & \hat{\mathbf{b}}_{m,1} \\ \vdots & \ddots & \vdots \\ \hat{\mathbf{b}}_{1,n} & \cdots & \hat{\mathbf{b}}_{m,n} \end{bmatrix}, \text{ where } \hat{\mathbf{b}}_{i,j} \text{ are } g(m)\times g(n) \text{ sub-blocks.}$$

(4) Transform each sub-block $\hat{\mathbf{b}}_{i,j}$ back to DCT-domain using $g(m)$×$g(n)$-point DCT.

$$\hat{B}_{i,j} = \mathrm{DCT}_{g(m)\times g(n)}(\hat{\mathbf{b}}_{i,j}) = T_{g(m)} \hat{\mathbf{b}}_{i,j} T_{g(n)}^t \qquad (7)$$

where $\hat{B}_{i,j}$ has the low-frequency DCT coefficients of the corresponding 8×8 block in the original image. Hence, we can recover the original block with its low-frequency as the $\hat{B}_{i,j}$ and other entries are all set to be zero, denoted as

$$\overline{B}_{i,j} = \begin{bmatrix} \hat{B}_{i,j} & \mathbf{0}_{8-g(m)\times g(n)} \\ \mathbf{0}_{g(m)\times 8-g(n)} & \mathbf{0}_{8-g(m)\times 8-g(n)} \end{bmatrix} \qquad (8)$$

where $\mathbf{0}_{k\times l}$ denotes the k×l zero matrix.

Note that, all these transformations can be combined into a simple form of matrix operations as similar to (2) without the need of converting DCT blocks into pixel values. Therefore the computation is very efficient.

3. COMPUTATION REDUCTION USING PARTIAL DECODING

We can observe from Fig. 1 that, the decoder-loop of CDDT is operated at the full picture resolution, while the encoding is performed at a reduced resolution. As described in Sec. 2, instead of using the whole DCT coefficients decoded from the decoder loop, the DCT decimation scheme only exploits the $g(m)$×$g(n)$ low-frequency DCT coefficients of each decoded block for downscaling by m×n. Furthermore, as compared to the CDDT in Fig. 1, decoding a B-frame with such a downsized resolution will not result in extra drifting error in the downscaling transcoder, since B-frames are not used for the predictions of other frames. A feasible approach for reducing the complexity of CDDT is to perform the full-resolution decoding for I- and P-frames, and reduced-resolution decoding for B-frames as depicted in Fig. 4 (scheme **A**). In this way, for B-frames, only the reduced-resolution DCT-MC is required in the decoder-loop, and the DCT-domain downscaling process of B-frames can also be saved. Since B-frames usually occupy a large portion of an I-B-P structured MPEG video, the computation saving can be very significant.

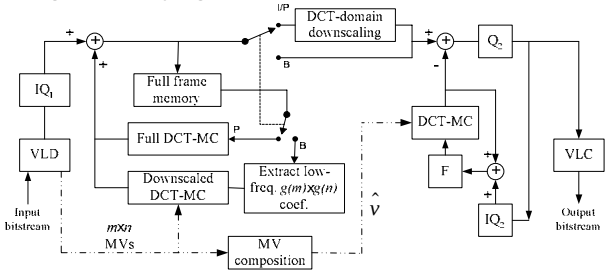

Fig. 4. Proposed architectures for computation reduction.

For simplicity, in the following, we show the simplified DCT-MC for decoding B-frames from only one reference frame. It can be easily extended to the case with bidirectional prediction. By incorporating the DCT decimation into the DCT-MC of the decoder-loop for B-frames, we can extract the $g(m)$×$g(n)$ low-frequency coefficients by

$$\hat{B} = P_{g(m)} \left(\sum_{i=1}^{4} H_{h_i} B_i H_{w_i} \right) P_{g(n)}^t = \sum_{i=1}^{4} P_{g(m)} H_{h_i} B_i H_{w_i} P_{g(n)}^t \qquad (9)$$

where $P_k = \begin{bmatrix} I_{k\times k} & \mathbf{0}_{8-k\times k} \\ \mathbf{0}_{k\times 8-k} & \mathbf{0}_{8-k\times 8-k} \end{bmatrix}$, $I_{k\times k}$ is a k×k identity matrix, and $\mathbf{0}$'s are zero matrices. Then each term in (11) becomes

$$P_{g(m)} H_{h_i} B_i H_{w_i} P_{g(n)}^t = \begin{bmatrix} H_{h_i}^{11} & H_{h_i}^{12} \end{bmatrix} \begin{bmatrix} B_i^{11} & B_i^{12} \\ B_i^{21} & B_i^{22} \end{bmatrix} \begin{bmatrix} H_{w_i}^{11} \\ H_{w_i}^{21} \end{bmatrix} \qquad (10)$$

$$= \left(H_{h_i}^{11} B_i^{11} + H_{h_i}^{12} B_i^{21} \right) H_{w_i}^{11} + \left(H_{h_i}^{11} B_i^{12} + H_{h_i}^{12} B_i^{22} \right) H_{w_i}^{21}$$

where the sub-matrices $H_{h_i}^{11}$ and $H_{h_i}^{12}$ are of sizes g(n)×g(m) and (8-g(n)×g(m), respectively; B_i^{11}, B_i^{12}, B_i^{21}, and B_i^{22} are of sizes g(m)×g(n), (8-g(m))×g(n), g(m)×(8-g(n)), and (8-g(m))×(8-g(n)), respectively; $H_{w_i}^{12}$ and $H_{w_i}^{21}$ are of sizes, g(n)×g(m) and g(n)×(8-g(m)), respectively. Using this approach, only the left-top entries need to be computed, thus the computational complexity of the DCT-MC$_1$ can be reduced significantly. In addition, the computation for DCT-domain downscaling is also saved.

The computation with Eq. (10) can be further reduced by applying the reduced-resolution decoding for all I-, P- and B-frames (scheme **B**). With scheme **B**, only g(m)×g(n) low-frequency coefficients (i.e., the subblock B^{11}) of each block in the reference frame (I- or P-frame) for a B-frame are extracted and stored, B^{12}, B^{21}, and B^{22} in Eq. (10) are thus all zero matrices. Eq. (10) can thus reduce to

$$\begin{aligned}\hat{B} &= \sum_{i=1}^{4} H_{h_i}^{11} B_i^{11} H_{w_i}^{11} \\ &= H_{h_1}^{11}\left(B_1^{11}H_{w_1}^{11} + B_2^{11}H_{w_2}^{11}\right) + H_{h_3}^{11}\left(B_3^{11}H_{w_3}^{11} + B_4^{11}H_{w_4}^{11}\right)\end{aligned} \tag{11}$$

The simplifcation in Eq. (11) reduces not only the computational cost but also the frame storage cost significantly. Such simplification will, however, lead to the mismatch between the frame stores of the front-end encoder and the reduced-resolution decoder-loop of the transcoder, thereby resulting in drifting errors, which will be investigated in the following section.

3. EXPERIMENTAL RESULTS

We compare the performance of the proposed compressed-domain downscaling transcoder (CDDT), and the proposed computation-reduction schemes **A** and **B**. Three test sequences "Mobile," "Flower Garden," and "Football" with a frame size of 704×576 (or 720×240) and frame rate of 25 fps, respectively are used for comparison. These teat sequences were pre-encoded at 14 Mbps with an MPEG-2 encoder using the TM5 rate control. The GOP structure used was (15,3). The pre-encoded bit-streams are then transcoded into 512 Kbps, with a downscaled frame-size of 224×192 (or 240×80; i.e., downscaled by 3×3) using the proposed CDDT and schemes **A** and **B**. The experiments were performed on a Pentium-IV 1.8 GHz PC. In order to compare the quality of each scheme, a 13-tap almost-ideal sinc filter with a cut-off frequency of $\pi/3$ was used to generate the downscaled version of each test sequence as the ground truth for performance evaluation. In addition to the proposed methods, we also implemented the shared information method proposed in [8] to achieve further speed-up.

Table 2. Performance comparison of average PSNR and processing speed of CPDT, CDDT and proposed schemes with (a) "Mobil", and (b) "Flower Garden" sequences.

(a)

Schemes		Speed (fps)		Average PSNR (dB)		
			shared	Y	C_r	C_b
Arbitrary CDDT		3.5	4.3	24.45	29.23	29.61
Comput. Reduction	**A**	8.7	9.9	24.45	29.23	29.61
	B	13.4	14.9	24.26	29.51	29.90

(b)

Schemes		Speed (fps)		Average PSNR (dB)		
			shared	Y	C_r	C_b
Proposed CDDT		4.15	4.89	24.64	30.69	34.68
Comput. Reduction	**A**	10.6	11.7	24.64	30.69	34.68
	B	14.5	15.9	24.44	30.82	34.83

Table 2 compares the average PSNR performance and processing speed of various transcoders. The processing speed using the proposed methods plus the shared information method in [8] is indicated as "Shared Info." The experimental results show that, as compared to the proposed CDDT, the computation-reduction scheme **A** can increase the processing speed by about 2.5 times without introducing any quality degradation for videos with the (15,3) GOP structure. The proposed scheme **B** can further increase the speed (up to about 3~3.5 times), while introducing about 0.2 dB quality degradation in the luminance component due to the extra drifting error caused by the reduced-size decoding. The frame-memory size of scheme **B** is also significantly less than that of scheme **A** due to the reduced-size decoding used in scheme **B**. The speed-up gain of scheme **A** is dependent on the GOP structure and size used. The larger the number of B-frames in a GOP, the higher the performance gain of proposed scheme **A**. The speed-up gain of method **B** depends less on the GOP structure since it reduces the decoding computational cost for all the I-, P-, and B-frames.

5. CONCLUSIONS

In this paper, we proposed efficient architectures for DCT-domain spatial-downscaling video transcoders. We have proposed a DCT-domain arbitrary resizing schemes and the associated DCT-transcoder architecture. We have also proposed three schemes to integrating the DCT-domain decoding and downscaling operations in the downscaling CDDT into a reduced-resolution DTC-MC so as to achieve significant computation reduction. The proposed scheme **A** can speed up the decoding and downscaling of B-frames without sacrificing the visual quality, while schemes **B** can speed up the decoding and downscaling of I-, P- and B-frames as well as reduce the storage cost with acceptable quality degradation. The proposed d computation reduction schemes can achieve up to 3~3.5 times speed-up with the (15,3) GOP structure when compared to the original method.

REFERENCES

[1] T. Shanableh and M. Ghanbari, "Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats," *IEEE Trans. on Multimedia*, vol. 2, no. 2, pp. 101-110, Jun. 2000.

[2] G. Shen, B. Zeng, Y.-Q. Zhang, and M.-L. Liou, "Transcoder with arbitrary resizing capability," in *Proc. IEEE ISCAS*, pp. V-25-28, May 2001, Sydney, Australia.

[3] Y.-P. Tang, H. Sun, and Y. Q. Liang, "On the methods and applications of arbitrary downsizing video transcoder," in *Proc. IEEE ICME*, pp. 301-304, Aug. 2002, Lausanne.

[4] P. A. A. Assuncao and M. Ghanbari, "A frequency-domain video transcoder for dynamic bit-rate reduction of MPEG-2 bit streams," *IEEE Trans. Circuits Syst. Video Technol*., vol. 8, no. 8, pp. 953-967, Dec. 1998.

[5] W. Zhu, K. Yang, and M. Beacken, "CIF-to-QCIF video bitstream down-conversion in the DCT domain," *Bell Labs technical journal* vol. 3, no. 3, pp. 21-29, Jul.-Sep. 1998.

[6] R. Dugad and N. Ahuja, "A fast scheme for image size change in the compressed domain," *IEEE Trans. Circuit Syst. Video Technol*., vol. 11, no. 4, pp. 461-474, Apr. 2001.

[7] S. F. Chang and D. G. Messerschmitt, "Manipulation and compositing of MC-DCT compressed video," *IEEE J. Select. Areas Commun*., vol. 13, no. 1, pp. 1-11, Jan. 1995.

[8] J. Song and B.-L. Yeo, "A fast algorithm for DCT-domain inverse motion compensation based on shared information in a macroblock," *IEEE Trans. Circuits Syst. Video Technol*., vol. 10, no. 5, pp. 767-775, Aug. 2000.