Transactions Letters

Visual Quality Enhancement in DCT-Domain Spatial Downscaling Transcoding Using Generalized DCT Decimation

Yuh-Ruey Lee and Chia-Wen Lin, *Senior Member, IEEE*

***Abstract*—In this paper, we propose a generalized discrete cosine transform (DCT) decimation scheme for DCT-domain spatial downscaling which performs two-fold decimation on subframes of a flexible size larger than the traditional 8×8 block size to improve the visual quality. Efficient sparse-matrix representations are then derived to reduce the computation of the proposed DCT decimation method. We compare the antialiasing filtering performances and computational complexities of the proposed downscaling scheme with the existing DCT-domain and pixel-domain downscaling schemes. Our analysis shows that the proposed scheme can reduce the aliasing artifact compared to the pixel-domain downscaling schemes, whereas the computational complexity may be increased. Experimental results are reported to show the efficacy of the proposed approach.**

***Index Terms*—Compressed-domain processing, spatial downscaling, video adaptation, video coding, video transcoding.**

I. Introduction

VIDEO transcoding [1] is an operation of converting a video bit stream from one format into another format. It is an efficient means of achieving fine and dynamic video adaptation. In realizing a transcoder, the computational cost and the picture quality are usually the two most important concerns. A straightforward realization of a video transcoder is to cascade a decoder followed by an encoder. This cascaded pixel-domain architecture is flexible and can be used for bit rate adaptation and spatio-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.

Recently, discrete cosine transform (DCT)-domain transcoders [2]–[11] have become very attractive because they can avoid the DCT and inverse DCT (IDCT) computations. In addition, several computationally efficient schemes have been developed for implementing the core module of DCT-domain transcoders: DCT-domain motion compensation (DCT-MC) [12]. A cascaded DCT-domain transcoder (CDDT), as depicted in Fig. 1, was first proposed in [3] for spatial downscaling where a DCT-domain bilinear filter was used as the antialiasing filter for the spatial downscaling.

Manuscript received April 22, 2006; revised February 19, 2007. This paper was recommended by Associate Editor J. Cai.

The authors are with the Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi 621, Taiwan, R.O.C. (e-mail: cwlin@cs.ccu.edu.tw; lyj@cs.ccu.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2007.903554

For spatial downscaling in the DCT-domain, a technique called frequency synthesis was first proposed in [4] for converting an HDTV video to an SDTV video at a decoder. The frequency synthesis downscaling method first synthesizes an incoming macroblock consisting of four 8×8 DCT blocks into one 16×16 DCT block, and then obtains the downscaled 8×8 DCT block by extracting the 8×8 low-frequency DCT coefficients of the 16×16 DCT block. The concept of frequency synthesis was further extended in [5] for interlaced video downscaling and three-layer scalable decoding. The method proposed in [6] exploits the multiplication-convolution property of DCT, that is, the multiplication of two signals is equivalent to the symmetric convolution of the corresponding DCT coefficients of the two signals. Using this property, four neighboring 8×8 DCT-block can be merged to a 16×16 DCT block. The 8×8 low-frequency coefficients are then extracted as the downscaled version of the 16×16 block. In [7], a DCT-decimation scheme was proposed for DCT-domain image downscaling and layered image coding. A decomposition method was also proposed in [7] to convert some matrices in matrix operations into sparse forms for computation reduction. The DCT decimation scheme with the 8×8 block size has proven to achieve significantly better visual quality compared to the schemes using pixel-domain filtering followed by down-sampling [7], [8]. Using DCT blocks of a block size larger than 8×8 such as the frequency synthesis method [4] would achieve better antialiasing filtering performance for spatial downscaling, while increasing the computational complexity.

Recently, a few approaches for DCT-domain arbitrary-ratio image downscaling, rather than two-fold downscaling, have been proposed in [8]–[10]. Shu and Chau [8] proposed an arbitrary downsizing algorithm that can use an arbitrary support area from the original compressed image by extending the method in [4] and exploiting the DCT-MC scheme proposed in [12]. In [9], a generalization of the method presented in [7] was proposed to produce a two-step mapping to achieve arbitrary factor resizing. This two-step mapping includes a combined IDCT and resizing operation followed by another combined DCT and resizing operation to generate a downscaled DCT image consisting of 8×8 blocks. The above steps are merged into a series of matrix operations that can be performed in the DCT domain. In [10], the proposed algorithm performs a combination of fast IDCT and DCT of composite lengths on a group of DCT blocks with zeros padding and high frequency coefficients truncation. The proposed scheme has similar PSNR performance but lower computational cost compared to

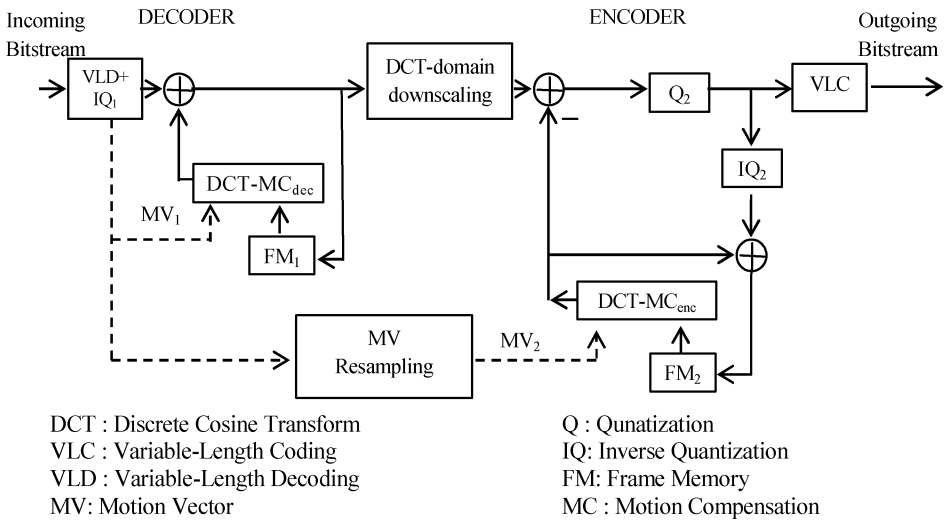

Fig. 1. Cascaded DCT-domain downscaling transcoder (CDDT).

the scheme proposed in [8]. For the case of two-fold downscaling, this method reduces to the frequency synthesis method proposed in [4], in which a downscaled 8×8 DCT block is obtained by extracting the 8×8 lowest coefficients of the 16×16 block synthesized from its corresponding four 8×8 DCT blocks of the input image. Note that, the downscaling method in [10] does not merge the DCT and IDCT operations into a single DCT-domain operation, but instead adopts a composite length DCT/IDCT algorithm for fast computation.

In this paper, we propose a generalized DCT decimation scheme that is based on a combined generalized frequency synthesis and low-frequency coefficients extraction operation to obtain a downscaled DCT image of 8×8 blocks directly in the DCT domain. The proposed scheme can synthesize a DCT block of a vector size larger than 16×16 from a group of neighboring blocks rather than the limited 16×16 size used in [4]. Compared to the zero-padding scheme used in [10] for extending the vector size, our scheme can significantly improve the antialiasing performance of downscaled filter. Besides, we shall show that the computation of the proposed scheme can be significantly reduced by using efficient sparse matrix representations similar to that in [7]. We shall also analyze the antialiasing properties of the proposed scheme to justify its performance. Our method offers the flexibility of applying different block sizes on regions with different spatial characteristics such that the visual quality can be maximized with reasonable computation cost.

II. Generalized DCT Decimation for Spatial Downscaling

A. Formulation of Generalized DCT Decimation

Fig. 2 illustrates the proposed generalized DCT decimation-by-two scheme for an 1-D signal. In our scheme, a group of consecutive 8-sample DCT vectors are first transformed into an N-pixel vector by 8-point IDCT, where N is a multiple of 8. The N-pixel vector is then transformed into its corresponding DCT vector by N-point DCT. DCT decimation is subsequently

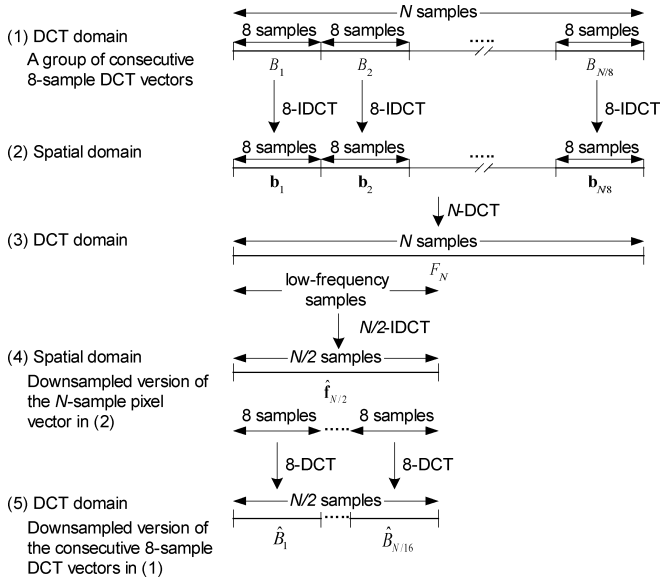

Fig. 2. Conceptual diagram of the proposed two-fold DCT decimation scheme. The input is a group of consecutive 8-sample DCT vectors, whereas the output is the spatially downscaled version of the input.

performed on the N-sample DCT vector by extracting the $N/2$ low-frequency DCT coefficients followed by $N/2$-point IDCT to obtain a downscaled $N/2$-pixel vector. Consequently, the $N/2$-pixel vector is transformed into a group of consecutive 8-sample DCT vectors by 8-point DCT to form the output DCT array. In the following, we explain the proposed algorithm by using the 1-D example.

1) Step 1. Grouping a Set of 8-Pixel Vectors to a Single N-Pixel Vector: Suppose that $\mathbf{f}_N$ is an 1-D N-pixel vector that is composed of 8-pixel vectors $\mathbf{b}_i, i = 1, \ldots, N/8$. The relation between $\mathbf{b}_i$ and its DCT representation B_i is expressed by

$$\mathbf{b}_i = T_8^t \cdot B_i \tag{1}$$

where T_8 represents the 8-point DCT transform matrix.

The N-point DCT representation of $\mathbf{f}_N$ can be computed by

$$F_N = T_N \cdot \mathbf{f}_N = T_N \begin{bmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_{N/8} \end{bmatrix} = [\, T_{N,1} \quad \cdots \quad T_{N,N/8} \,] \begin{bmatrix} T_8^t B_1 \\ \vdots \\ T_8^t B_{N/8} \end{bmatrix} \tag{2}$$

where T_N represents the N-point DCT transform matrix, that is divided into $N/8$ columns of submatrices $T_{N,i}$ of size $N \times 8$. Equation (2) can be rewritten as

$$F_N = \left(T_{N,1} \cdot T_8^t\right) B_1 + \cdots + \left(T_{N,N/8} \cdot T_8^t\right) B_{N/8} = \sum_{i=1}^{N/8} \left\{ \left(T_{N,i} \cdot T_8^t\right) B_i \right\} \tag{3}$$

where $T_{N,i} \cdot T_8^t$ can be calculated and stored in a look-up table offline for computing F_N.

2) Step 2. Extracting the $N/2$ Low-Frequency Coefficients From an N-Pixel DCT Vector: In order to obtain the downscaled version of F_N, the $N/2$ low-frequency coefficients of F_N are extracted by

$$\widehat{F}_{N/2} = P_{N/2} \cdot F_N = \sum_{i=1}^{N/8} \left\{ \left(P_{N/2} \cdot T_{N,i} \cdot T_8^t\right) B_i \right\} = \sum_{i=1}^{N/8} L_i \cdot B_i \tag{4}$$

where $P_{N/2} = [\, I_{(N/2)} \quad \mathbf{0}_{(N/2)} \,]$ is a matrix used for extracting the $N/2$ low-frequency coefficients. The matrices, $I_{(N/2)}$ and $\mathbf{0}_{(N/2)}$, represent the $(N/2) \times (N/2)$ identity and zero matrices, respectively, and $L_i = P_{N/2} \cdot T_{N,i} \cdot T_8^t$ denotes the ith synthesis matrix of size $(N/2) \times 8$. The synthesis matrices can be calculated and stored offline, thus will not consume extra computation while performing transcoding. The set of matrices, $\{L_i\}_{i=1}^{N/8}$, are an extension of the frequency synthesis matrices presented in [7] to be applied for downscaling an N-pixel vector consisting of two or more 8-sample DCT vectors.

3) Step 3. Converting $\widehat{F}_{N/2}$ to a Downscaled $N/2$-Sample Vector: By performing $N/2$-point IDCT on $\widehat{F}_{N/2}$, we can obtain the corresponding $N/2$-pixel vector $\hat{\mathbf{f}}_{N/2}$ as follows:

$$\hat{\mathbf{f}}_{N/2} = T_{N/2}^t \cdot \widehat{F}_{N/2} = \begin{bmatrix} T_{N/2,1}^t \\ \vdots \\ T_{N/2,N/16+1}^t \end{bmatrix} \widehat{F}_{N/2} = \begin{bmatrix} \hat{\mathbf{b}}_1 \\ \vdots \\ \hat{\mathbf{b}}_{N/16+1} \end{bmatrix}. \tag{5}$$

According to (5), each 8-pixel vector $\hat{\mathbf{b}}_j$ in $\hat{\mathbf{f}}_{N/2}$ is computed by

$$\widehat{\mathbf{b}}_j = T_{N/2,j}^t \cdot \widehat{F}_{N/2}, \qquad j = 1, \ldots, \left\lfloor \frac{N}{16} \right\rfloor + 1. \tag{6}$$

As a result, the corresponding 8-sample DCT vector is obtained by $\widehat{B}_j = T_8 \cdot \hat{\mathbf{b}}_j$, where $\hat{\mathbf{b}}_j$ is the jth 8-pixel vector in the downscaled vector $\hat{\mathbf{f}}_{N/2}$. Therefore, $\widehat{B}_j$ can be computed directly from $\widehat{F}_{N/2}$ by

$$\widehat{B}_j = \left(T_8 \cdot T_{N/2,j}^t\right) \hat{F}_{N/2} = S_j \cdot \widehat{F}_{N/2}, \qquad j = 1, \ldots, \left\lfloor \frac{N}{16} \right\rfloor \tag{7}$$

where $S_j = T_8 \cdot T_{N/2,j}^t$ is an $8 \times (N/2)$ matrix. Note that, when N is not a multiple of 16, the last pixel vector $\hat{\mathbf{b}}_{\lfloor N/16 \rfloor + 1}$ is of size 4, but the other pixel vectors are of size 8. Thus, $T_{N/2,\lfloor N/16 \rfloor + 1}^t$ is of size $4 \times (N/2)$. In this case, the corresponding 4-sample DCT vector is obtained by $\widehat{B}_j = T_4 \cdot \hat{\mathbf{b}}_{\lfloor N/16 \rfloor + 1}$. Equation (7) is thus rewritten as

$$\widehat{B}_j = \left(T_4 \cdot T_{N/2,\lfloor N/16 \rfloor + 1}^t\right) \hat{F}_{N/2} = S' \cdot \widehat{F}_{N/2} \tag{8}$$

where $S' = T_4 \cdot T_{N/2,\lfloor N/16 \rfloor + 1}^t$ is a $4 \times (N/2)$ matrix.

The above procedures can be combined together to be performed in the DCT domain as summarized below.

Algorithm: 2:1 Downscaling Using Generalized DCT Decimation

---

1. *Divide one row/column of an input coded frame into N-sample vectors, each vector consisting of 8-sample DCT vectors B_i for $i = 1, \ldots, N/8$.*

2. *Compute the low frequency version $\widehat{F}_{N/2}$ of F_N from the N-sample DCT vector by*

$$\widehat{F}_{N/2} = L_1 \cdot B_1 + \cdots + L_{N/8} \cdot B_{N/8} = \sum_{i=1}^{N/8} L_i \cdot B_i$$

where $L_i = P_{N/2} \cdot T_{N,i} \cdot T_8^t$.

3. *Compute each outgoing downscaled DCT vector, $\widehat{B}_j$, for $j = 1, \ldots, \lfloor N/16 \rfloor + 1$ by $\widehat{B}_j = \left(T_8 \cdot T_{N/2,j}^t\right) \hat{F}_{N/2} = S_j \cdot \widehat{F}_{N/2}$,* for $j = 1, \ldots, \ldots, \lfloor N/16 \rfloor$, and $\widehat{B}_j = \left(T_4 \cdot T_{N/2,\lfloor N/16 \rfloor + 1}^t\right) \hat{F}_{N/2} = S' \cdot \widehat{F}_{N/2}$, for $j = \lfloor N/16 \rfloor + 1$, *where $S_j = T_8 \cdot T_{N/2,j}^t$ and $S' = T_4 \cdot T_{N/2,\lfloor N/16 \rfloor + 1}^t$.*

B. Computation Reduction Using Sparse-Matrix Representations

To reduce the computation for matrix operations in (4) and (7), generalizing the method presented in [4], we decompose L_i into representations of sparse matrices. We observe that there exist the following characteristics among the entries of L_i with dimension of $(N/2) \times 8$.

1) General Case: The rth row of L_i for $r = 0, N/8, 2N/8, 3N/8$, has all its entries being zeros except the rth entry. Hence, we see that about $8/N$ of the entries of these matrices are zeros.

2) Special Case I: $K = N/8$ is even.

$L_i(r, c) = (-1)^{r+c} L_{K-i+1}(r, c)$ for $i = 1, \ldots, N/16$, $r = 0, \ldots, N/2$, and $c = 0, \ldots, 7$, where $L_i(r, c)$ denotes the entry on the rth row and the cth column.

3) Special Case II: $K = N/8$ is odd.

$L_i(r,c) = (-1)^{r+c} L_{K-i+1}(r,c)$ for $i = 1,\ldots,N/16$, $r = 0,\ldots,N/2$, and $c = 0,\ldots,7$. For the matrix with $i = \lfloor N/16 \rfloor + 1$, the entries with odd value of $r+c$ are zero for $r \neq 0, N/8, 2N/8, 3N/8$. Therefore, at most, half entries of this matrix are not zero.

In [5], *special case I* was reported for $N = 16$. In our proposed scheme, the additional properties of the *general case* and *special case II* are further used for further reducing the computation of matrix operations.

Based on these facts, we define two new matrices, C_i^N and D_i^N, to reduce the computations in (4). We derive the sparse matrix representation for an even K in the following. The representation for an odd K can also be obtained similarly. For $i = 1,\ldots,K/2$, we define

$$C_i^N(r,c) = \begin{cases} L_i(r,c), & r+c \text{ is even} \\ 0, & \text{otherwise} \end{cases}$$
$$D_i^N(r,c) = \begin{cases} L_i(r,c), & r+c \text{ is odd} \\ 0, & \text{otherwise.} \end{cases}$$

Therefore we have $C_i^N + D_i^N = L_i$ and $C_i^N - D_i^N = L_{K-i+1}$, respectively. The matrices C_i^N and D_i^N have at least half zero entries. Substituting L_i with C_i^N and D_i^N, (4) becomes

$$\begin{aligned}\widehat{F}_{N/2} =& (C_1^N + D_1^N)B_1 + \cdots + (C_{K/2}^N + D_{K/2}^N)B_{K/2} \\ &+ (C_1^N - D_1^N)B_K + \cdots + (C_{K/2}^N - D_{K/2}^N) \\ &\cdot B_{(K/2)+1}. \end{aligned} \tag{9}$$

Grouping the terms with the same factors C_i^N and D_i^N, respectively, we obtain

$$\widehat{F}_{N/2} = \sum_{i=1}^{K/2} \{C_i^N(B_i + B_{K-i+1}) + D_i^N(B_i - B_{K-i+1})\}. \tag{10}$$

Because that both C_i^N and D_i^N are sparse matrices, the computation required for (10) is significantly less than that for (4). Equation (10), therefore, substitutes Step 2 of the aforementioned downscaling algorithm for computation reduction. In addition, because the transpose of S_j in (7) is identical to L_j, the sparse matrix representations can be applied to (7) to further reduce the computation.

III. Analysis of the Proposed DCT Decimation Filters

The operation of retaining the low-frequency coefficients of a DCT vector and taking the half-size IDCT is, in effect, to perform antialiasing filtering followed by down-sampling on the vector in the pixel domain. This results in a downscaled version of the vector. In the following, we shall analyze the antialiasing filtering performances and computational complexities of DCT-decimation filters for the 1-D case. The analysis of the antialiasing property is similar to that in [7].

When an 1-D signal $\mathbf{x}$ of N samples is downscaled by two, the resulting $N/2$-sample signal $\mathbf{y}$ can be represented by

$$\mathbf{y} = T_{N/2}^t \cdot P_{N/2} \cdot T_N \cdot \mathbf{x} = G_{(N/2)\times N} \cdot \mathbf{x}. \tag{11}$$

where $G_{(N/2)\times N}$ denotes the decimation filter.

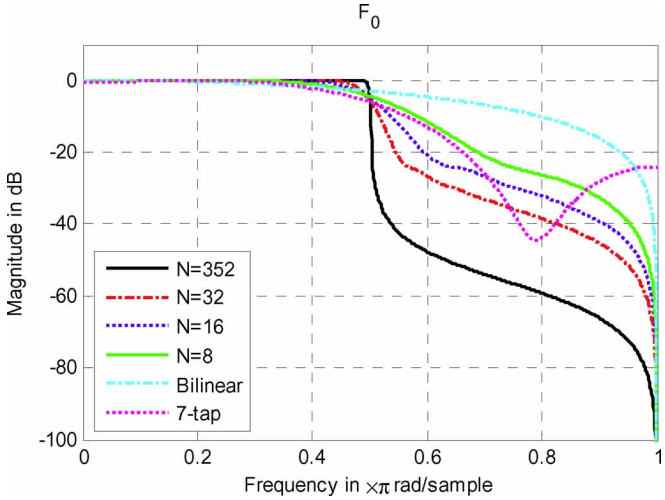

Fig. 3. Comparison of magnitude responses of DCT-domain and pixel-domain decimation filters.

The input signal is linearly transformed with the matrix $G_{(N/2)\times N}$ to obtain a corresponding down-sampled output signal. The linear transformation can be represented as an N-band filter bank [7], where the impulse response of the ith subband filter h_i is defined as the reverse of the ith row of the matrix $G_{(N/2)\times N}$. Hence, the z-transform of the output $\mathbf{y}$ can be obtained by

$$Y(z) = \frac{1}{N} \sum_{k=0}^{N-1} X(W^{-k}\sqrt{z}) F_k(W^{-k}\sqrt{z}) \tag{12}$$

where $W = \exp(j2\pi/N)$, $j = \sqrt{-1}$ and

$$F_k(z) = \sum_{i=0}^{N/2-1} z^{-2i} H_i(z) W^{-2ki} \tag{13}$$

where $H_i(z)$ is the z-transform of h_i.

Fig. 3 shows the magnitude responses of DCT-decimation with $N = 8$, 16, 32, and 352, respectively, and the two pixel-domain antialiasing filters: the bilinear filter and 7-tap Gaussian filter with the coefficients $(-1,0,9,16,9,0,-1)/32$. In the proposed scheme, as N increases, the gain of DCT decimation filter becomes much flatter in the low frequency part $(0 \sim \pi/2)$, while the gain decreases rapidly in the high-frequency part $(\pi/2 \sim \pi)$. Lower gains at the frequencies higher than the stop-band imply less visible aliasing artifacts in the downscaled image. Increasing N of DCT-decimation would obviously improve the antialiasing performance of the downscaling filter, thereby enhancing the visual quality of the downscaled image.

The computational complexity of the proposed DCT decimation scheme, however, increases as the vector size N increases. The total numbers of multiplications and additions required for computing each outgoing 8-sample DCT vector $\hat{B}_j$ in (8) with an N-sample DCT vector are $12N$ and $12N-16$, respectively. Using the sparse-matrix representation in (10), the total numbers of multiplications and additions are reduced to $8N-28$ and $8N-24$, respectively. In Table I, the computational complexities of the proposed DCT-decimation scheme with various vector sizes are compared for the 1-D case, which will be doubled in the 2-D case when two separable 1-D filters are used to

TABLE I
COMPUTATIONAL COMPLEXITIES OF THE GENERALIZED DCT-DECIMATION SCHEME FOR AN 1-D 8-SAMPLE VECTOR WITH AND WITHOUT SPARSE-MATRIX DECOMPOSITION FOR DIFFERENT VECTOR SIZES

N	Avg. multiplications		Avg. additions	
	Gen. DCT decimation	Sparse-matrix decomposition	Gen. DCT decimation	Sparse-matrix decomposition
352	4224	2788	4208	2792
32	384	228	368	232
16	192	100	176	104
8 [7]	64	20	64	20

TABLE II
COMPUTATIONAL COMPLEXITIES OF PIXEL-DOMAIN DOWNSCALING FOR AN 1-D 8-PIXEL VECTOR USING THE BILINEAR AND 7-TAP GAUSSIAN FILTERS

Tap Length	Avg. multiplications	Avg. additions
Bilinear	8	8
7-tap Gaussian	56	48

realize a 2-D filter. The numbers of operations for $N = 8$ are calculated based on the method presented in [7]. Table II shows the average computational complexities of pixel-domain downscaling using the bilinear and 7-tap Gaussian filters.

IV. EXPERIMENTAL RESULTS

In our experiments, one CIF (352×288) sequence, *Container*, and two ITUR-601 (704×576) sequences, *City* and *Harbour*, all with 150 frames, are encoded by a front-end MPEG-2 encoder. Each coded video is then transcoded by using the CDDT shown in Fig. 1 that implements the proposed DCT-decimation scheme, resulting in a spatially downscaled video of quarter size. We also implement the bilinear filter and the 7-tap Gaussian filter on a cascaded pixel-domain downscaling transcoder for performance comparison.

Following the test methodology presented in [6]–[11], each downscaled image is decoded and up-scaled to its original size for performance evaluation. It is shown in [5] that, given a decimation filter matrix G, the optimal least-squares upscaling filter matrix that minimizes the error between the original-sized image and its reconstructed (downscaled and then upscaled) one is the Moore–Penrose pseudo-inverse of G as follows:

$$G^{+} = G^{t} \cdot \left(G \cdot G^{t}\right)^{-1}. \tag{14}$$

Substituting the decimation filter matrix in (11) into (14), the least-squares interpolation filter matrix of the generalized DCT-decimation scheme is obtained by

$$\begin{aligned} &G^{+} \\ &= \left(T^{t}_{N/2} \cdot P_{N/2} \cdot T_{N}\right)^{t} \left(T^{t}_{N/2} \cdot P_{N/2} \cdot T_{N} \cdot T^{t}_{N} \cdot P^{t}_{N/2} \cdot T_{N/2}\right)^{-1} \\ &= T^{t}_{N} \cdot P^{t}_{N/2} \cdot T_{N/2}. \end{aligned} \tag{15}$$

The least-squares upscaling filter in (15) can be implemented using the following generalized DCT-interpolation process.

1) Divide each downscaled pixel vector into $N/2$-sample pixel vector. Then transform each pixel vector to an $N/2$-sample DCT vector by $N/2$-point DCT.
2) Expand the size of each $N/2$-sample DCT vector to N-sample by padding zero coefficients in the high-frequency bands.
3) Apply N-point IDCT to transform each expanded DCT vector into its corresponding N-sample pixel vector

In our experiments, the optimal least-squares upscaling filters of the pixel-domain bilinear and 7-tap downscaling filters are also implemented. The average peak signal-to-noise ratio (PSNR) between the up-scaled version of each downscaled bitstream and its original uncompressed one is evaluated. Fig. 4 compares the rate-PSNR performances of DCT-decimation using four vector sizes (352×288, 32×32, 16×16, and

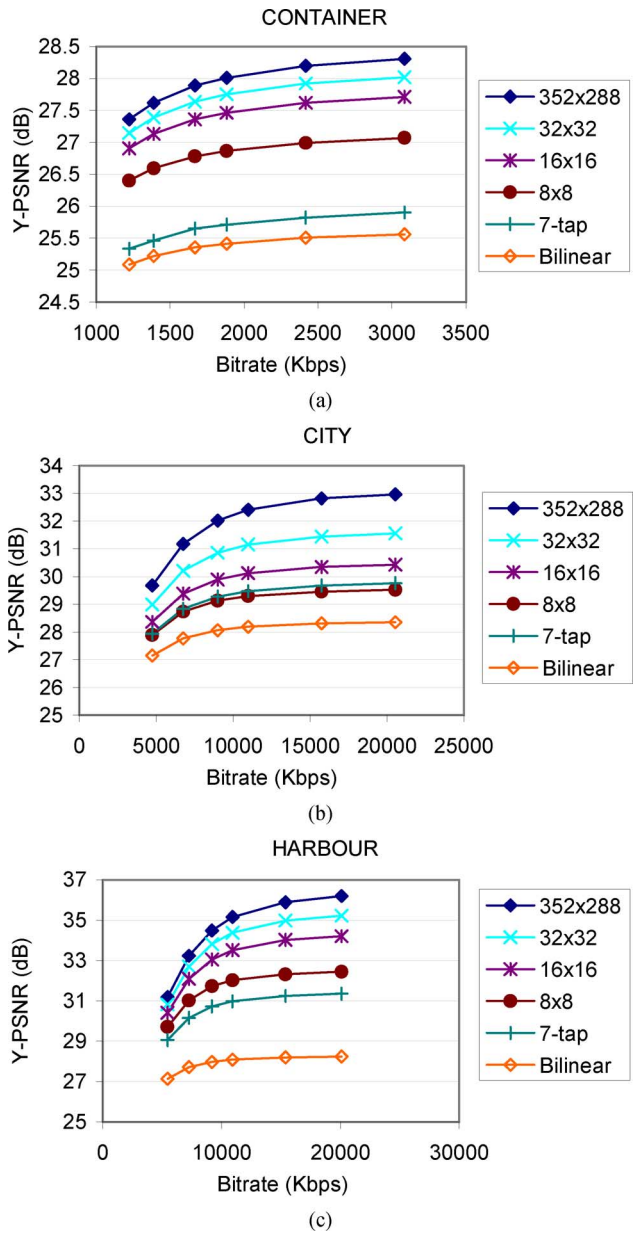

Fig. 4. Average luminance PSNR performance comparison of various downscaling filters followed by their corresponding least-squares upscaling filters for (a) *Container*, (b) *City*, and (c) *Harbour*. The test sequences are first intra-coded with $\mathrm{QP} = (5, 7, 10, 12, 16, 20)$ and then downscaled to their quarter size using the various downscaling filters. Each downscaled image is then upscaled to its original size using the corresponding least-squares upscaling filters of the downscaling filters.

8×8) and the two pixel-domain downscaling filters. Each test video is first intra-encoded by a front-end coder with $\mathrm{QP} = (5, 7, 10, 12, 16, 20)$. The compressed video is then downscaled into its quarter size and re-encoded. We compare the PSNR values of the output images of the downscaling filters (followed by their corresponding least-squares upscaling filters) rather than the final outputs of the transcoders. One can find the comparisons of overall performances (including the effects of DCT-MCs, downscaling, and re-quantization) of transcoders with different downscaling schemes in [11]. Compared to the 7-tap Gaussian filter, the proposed scheme (followed by its optimal upscaling filter) achieves significant PSNR improvements by up to 2.0–2.5, 1.7–3.4, and 2.1–5.2 dB, respectively. Fig. 4 also shows that downscaling with a vector size larger than 8×8 achieves significant PSNR performance improvement compared to the 8×8-sized scheme.

The performance improvement, however, comes with increased complexity. According to our profiling results, DCT-decimation with N =8, 16, and 32 respectively consumes about 2%, 7%, and 12% computation of a CDDT without specific code optimization. For sequence with fine spatial details (e.g., *Harbour* and *City*), using vector size of $N = 32$ achieves significant PSNR improvement (up to 2 dB) over $N = 16$, and very close performance to those of the largest vector sizes, while consuming a reasonable amount of computation. DCT-decimation with the three vector sizes, N =8, 16, and 32, can usually provide sufficient flexibility for achieving a good tradeoff between computational complexity and visual quality.

V. Conclusion

We proposed a generalized DCT decimation scheme which can adopt a vector size larger than 8×8. We also showed how to reduce the computation of the proposed scheme using sparse-matrix representations. We have compared the antialiasing properties and computational complexities of various downscaling filters. Experimental results show that the proposed scheme with vector sizes of N =16 and 32 usually achieves significantly better visual quality over that with $N = 8$, while leading to increased computational complexity.

Note that, the performance gain using a larger vector size is content dependent. Typically, applying a small vector size (e.g., $N = 8$) can do a good job for low-activity regions, whereas high-activity regions usually require a larger vector size (e.g., N =16 or 32) to achieve good visual quality. DCT-decimation with the three vector sizes, N =8, 16, and 32, can usually provide sufficient flexibility in achieving a good tradeoff between computational complexity and visual quality. The proposed framework also offers the flexibility of adaptively applying variable block sizes on regions with different spatial activities such that the visual quality can be maximized without introducing heavy computation.

References

[1] J. Xin, C.-W. Lin, and M.-T. Sun, "Digital video transcoding," *Proc. IEEE*, vol. 93, no. 1, pp. 84–97, Jan. 2005.

[2] P. 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.

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

[4] J. Bao, H. Sun, and T. C. Poon, "HDTV down-conversion decoder," *IEEE Trans. Consum. Electron.*, vol. 42, no. 3, pp. 402–410, Aug. 1996.

[5] A. Vetro, H. Sun, P. DaGraca, and T. Poon, "Minimum drift architectures for three-layer scalable DTV decoding," *IEEE Trans. Consum. Electron.*, vol. 44, no. 3, pp. 527–536, Aug. 1998.

[6] Y. S. Park and H. W. Park, "Design and analysis of image resizing filter in the block-DCT domain," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 2, pp. 274–279, Feb. 2004.

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

[8] H. Shu and L.-P. Chau, "An efficient arbitrary downsizing algorithm for video transcoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 6, pp. 887–891, Jun. 2004.

[9] C. L. Salazar and T. D. Tran, "On resizing images in the DCT domain," in *Proc. IEEE Int. Conf. Image Process.*, Singapore, Oct. 2004, vol. 4, pp. 2797–2800.

[10] Y. S. Park and H. W. Park, "Arbitrary-ratio image resizing using fast DCT of composite length for DCT-based transcoder," *IEEE Trans. Image Process.*, vol. 15, no. 2, pp. 494–500, Feb. 2006.

[11] Y.-R. Lee and C.-W. Lin, "DCT-domain spatial transcoding using generalized DCT decimation," in *Proc. IEEE Int. Conf. Image Process.*, Genova, Italy, Sep. 2005, vol. 1, pp. 821–824.

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