A FAST DCT DOMAIN BASED VIDEO DOWNSCALING SYSTEM

Sudhir Porwal

Image Analysis Center
Defence Electronics Applications Laboratory
Dehradun, INDIA 248001
Email: dealdrdo@del2.vsnl.net.in

Jayanta Mukhopadhyay

Dept. of Computer Sc. & Engg.
Indian Institute of Technology
Kharagpur, INDIA 721302
Email: jay@cse.iitkgp.ernet.in

ABSTRACT

In many multi-media applications DCT based digital video coding standards like MPEG are widely used. In certain applications like video database browsing, picture in picture etc., the video is required to be downscaled before transmitting it over the network. The naive spatial domain approach for video downscaling requires video to be fully decompressed. It requires huge computation time. The computation time greatly be reduced if all the computations are performed in the DCT domain. In this paper, we propose a fast DCT domain based video downscaling system. To reduce computational requirement, we have proposed efficient algorithms for inverse motion compensation. The sparseness of the DCT blocks are also considered for further reduction of computations. We found that our proposed system is 36 times faster than the spatial domain method and PSNR wise produces better quality downscaled video.

Index Terms– Discrete cosine transform (DCT), MPEG, video downscaling, inverse motion compensation, motion-vector refinement.

1. INTRODUCTION

With the advancement of multi-media based applications, video data is available in digital format. The video compression standards such as MPEG are used to efficiently store and transmit the digital video over the network. The MPEG compression standards use the Discrete Cosine Transform (DCT) and Motion Compensation (MC) to achieve the higher degree of compression. The MPEG video can be processed either in spatial domain or in DCT domain. In spatial domain approaches, the video is first fully decompressed, video frames are processed in spatial or pixel domain and then again reencoded as per MPEG video standards. In the DCT domain methods, the video is partially uncompressed (only VLC and Huffman decoded), the DCT blocks of the video are processed and then again the processed DCT blocks are VLC and Huffman encoded. The advantage of DCT domain processing is that it requires less computation and the sparseness of the DCT blocks can also be utilized.

The basic building blocks of a DCT based video downscaling system are intra frame downscaling, inverse motion compensation, and motion estimation and compensation for downscaled frames. There are many algorithms reported in the literature for intra frame downscaling in the DCT domain. Dugad and Ahuja [1] suggested an excellent scheme to perform intra frame downscaling. Their algorithm requires 1.25 multiplication and 1.25 addition per pixel of the original frame.

The inverse motion compensation (IMC) operation is required to convert motion compensated inter frames to intra frames for video manipulation. Several works [2],[3], [4], [5] are reported in the literature to perform IMC in DCT domain. Merhav[3] has proposed an excellent scheme to perform IMC in the DCT domain on 8×8 block basis. He has proposed a computation model based on factorization of the DCT and IDCT matrices. In [5], the shared information in a macroblock is used to speed up the process of IMC. It shows about 19% and 13.5% improvement over the method presented in [3], [4] respectively. We have extended the idea given in [3] to perform IMC and used the shared information present in a macroblock. We would refer this technique as macroblock wise inverse motion compensation (MBIMC). As the details of this algorithm is communicated elsewhere, we briefly discuss the algorithm for the sake of completeness of this paper. It is found that this technique provided 27% improvement over the method presented in [3]. In this work, we have tried to further reduce the computations of MBIMC scheme by utilizing the sparseness of DCT blocks. It results in 70% improvement (in the number of operations required) in MBIMC scheme.

The motion estimation in DCT domain is very expensive. Many researchers have suggested a different scheme to perform motion estimation which uses the motion information present in the original video to compute a good estimate of the motion vectors for downscaled video frames. These schemes are known as motion vector re-estimation techniques. Shen [6] has presented an adaptive motion vector resampling (AMVR) technique. The AMVR technique takes the weighted average of the motion vectors of the four adjacent macroblocks in the original video stream and computes the motion vector for macroblock in the downscaled video. This technique tries to align the new motion vector towards the worst predicted macroblock so that the prediction error is reduced for downscaled macroblock. We have used the MBIMC scheme to perform motion compensation for downscaled video in DCT domain.

Our proposed DCT domain based video downscaling system is integration of above mentioned techniques. The proposed system is compared with spatial and hybrid domain [6] methods. It is shown in the results that our proposed system is 36 times faster than spatial domain method and PSNR wise produces better output than spatial and hybrid system. In next section (section II), we briefly describe the MBIMC scheme followed by further reduction of computations with a proposed modification. Our proposed downscaling system is discussed in section III. Subsequently, results are presented and discussed in section IV.

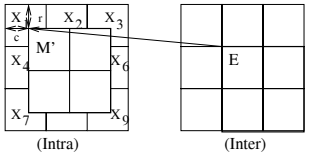

Fig. 1. Macroblockwise inverse motion compensation (MBIMC)

2. MACROBLOCKWISE INVERSE MOTION COMPENSATION (MBIMC)

As stated in the name, this technique performs the IMC on macroblock basis. It is shown in Fig.1, a predicted macroblock may intersect at the most with nine 8×8 DCT blocks. In general, the predicted macroblock M' will not align with any 8×8 DCT block boundaries in the reference frame but it will overlap with nine 8×8 DCT blocks. In Fig. 1, r and c represent the start location of a predicted macroblock. If $x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9$ are the adjacent blocks in spatial domain then a 16×16 block from the 24×24 block can be extracted using the Eq. (1).

$$m' = c_r \begin{bmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{bmatrix} c_c \tag{1}$$

Where m' is the predicted macroblock in spatial domain and c_r is a 16×24 matrix, c_c is 24×16 matrix. These matrices are different for different values of r and c (refer Fig. 1). The structure of c_r matrix is given below.

$$\mathbf{c_r}_{(16,24)} = \begin{bmatrix} 0 & 0 & .. & 0 & 1 & 0 & .. & 0 & 0 & 0 & .. & 0 \\ 0 & 0 & .. & 0 & 0 & 1 & .. & 0 & 0 & 0 & .. & 0 \\ .. & .. & .. & & .. & .. & .. & & .. & .. & .. & \\ .. & .. & .. & & .. & .. & .. & & .. & .. & .. & \\ 0 & 0 & .. & 0 & 0 & 0 & .. & 1 & 0 & 0 & .. & 0 \end{bmatrix}$$
$$(r-1)column \qquad 16column \qquad (8-r+1)column$$

Similarly c_c matrices can also be derived. These matrices work as row and column selector matrices.

Since we have DCT blocks $X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8, X_9$ and we have to extract macroblock M' from these nine DCT blocks. The macroblock M' is a group of four adjacent 8×8 DCT blocks. To achieve this, Eq. (1) is expressed in the DCT domain as shown in Eq.(2).

In Eq.(2), $'0'$ represents a 8×8 matrix of zeros and S_8 represents a 8-point type-II DCT transformation matrix. Transpose of a matrix $\mathbf{A}$ is represented by $\mathbf{A^t}$. The matrix multiplication inside the curly braces results in a 16×16 matrix, which represent the spatial domain block. The premultiplication of $\begin{pmatrix} S_8 & 0 \\ 0 & S_8 \end{pmatrix}$ and post multiplication of $\begin{pmatrix} S_8^t & 0 \\ 0 & S_8^t \end{pmatrix}$ results in a 16×16 macroblock containing four 8×8 DCT blocks. Interestingly, The S_8 can be factorized as shown in Eq. 5.

Table 1. Multiplication complexities of $\mathbf{J_i}$ matrices with 24×24 sparse DCT matrix

Matrix	Computations/column
J_1	2m + 24a
J_2	3m + 25a
J_3	3m + 26a
J_4	3m + 26a
J_5	3m + 28a
J_6	3m + 26a
J_7	3m + 26a
J_8	3m + 25a

$$S_8 = DPB_1B_2MA_1A_2A_3 \tag{5}$$

where D is a diagonal matrix, P is a permutation matrix, B_1, B_2, A_1, A_2, A_3 are sparse matrices of 1, 0 and -1. M is a sparse matrix of real numbers. Please refer the work reported in [3] for further details.

Let us define $\mathbf{S}$ and $\mathbf{S^t}$ as shown below.

$\mathbf{S} = \begin{bmatrix} S_8 & 0 & 0 \\ 0 & S_8 & 0 \\ 0 & 0 & S_8 \end{bmatrix}$ and $\mathbf{S^t} = \begin{bmatrix} S_8^t & 0 & 0 \\ 0 & S_8^t & 0 \\ 0 & 0 & S_8^t \end{bmatrix}$

Then $\mathbf{S^t}$ can be written as shown in Eq.(3) using the factorization shown in Eq. (5).

In Eq. (4), the matrix multiplication by $\mathbf{P}(\mathbf{P^t})$ and $\mathbf{D}(\mathbf{D^t})$ matrices can be ignored by obvious reasons given in [3]. If we represent $J_r = \mathbf{c_r Q^t}$ and $K_c = \mathbf{Q c_c}$. In Eq. 4, inside the curly braces, there are two matrix multiplication of $\mathbf{B_1}/\mathbf{B_1^t}$, two matrix multiplication of $\mathbf{B_2}/\mathbf{B_2^t}$ and single multiplication of J_r and K_c matrices. Finally four 2D DCT operations are performed over the resulting matrix. it requires 3.43 multiplications and 20.5 addition operations per pixel of the input frame to perform IMC in the DCT domain. These required computations can further be reduced by exploiting the sparseness of the DCT blocks as described in next section.

2.1. Modification over MBIMC

In the proposed modification, we will consider that the 8×8 DCT blocks are sparse in nature. Since most of the energy in a video is concentrated in the low frequency components, the high frequency components are expected to occur with smaller magnitudes. The quantization step during MPEG encoding will also turn most of the high frequency components to zero in the DCT blocks. With the above mentioned observations, we consider only first 16 non-zero values in the zig-zag order of a 8×8 DCT block. The Eq. (4) computes the predicted macroblock in MBIMC scheme. The DCT blocks $X_1, X_2, \ldots, X_9$ are now considered as sparse matrices.

$$\mathbf{M'} = \begin{pmatrix} S_8 & 0 \\ 0 & S_8 \end{pmatrix} \left\{ c_r \begin{bmatrix} S_8^t & 0 & 0 \\ 0 & S_8^t & 0 \\ 0 & 0 & S_8^t \end{bmatrix} \begin{bmatrix} X_1 & X_2 & X_3 \\ X_4 & X_5 & X_6 \\ X_7 & X_8 & X_9 \end{bmatrix} \begin{bmatrix} S_8 & 0 & 0 \\ 0 & S_8 & 0 \\ 0 & 0 & S_8 \end{bmatrix} c_c \right\} \begin{pmatrix} S_8^t & 0 \\ 0 & S_8^t \end{pmatrix} \tag{2}$$

$$\mathbf{S^t} = \underbrace{\begin{bmatrix} (MA_1A_2A_3)^t & 0 & 0 \\ 0 & (MA_1A_2A_3)^t & 0 \\ 0 & 0 & (MA_1A_2A_3)^t \end{bmatrix}}_{\mathbf{Q^t}} \underbrace{\begin{bmatrix} B_2^t & 0 & 0 \\ 0 & B_2^t & 0 \\ 0 & 0 & B_2^t \end{bmatrix}}_{\mathbf{B2^t}} \underbrace{\begin{bmatrix} B_1^t & 0 & 0 \\ 0 & B_1^t & 0 \\ 0 & 0 & B_1^t \end{bmatrix}}_{\mathbf{B1^t}} \underbrace{\begin{bmatrix} P^t & 0 & 0 \\ 0 & P^t & 0 \\ 0 & 0 & P^t \end{bmatrix}}_{\mathbf{P^t}} \underbrace{\begin{bmatrix} D^t & 0 & 0 \\ 0 & D^t & 0 \\ 0 & 0 & D^t \end{bmatrix}}_{\mathbf{D^t}} \tag{3}$$

Similarly, equation for $\mathbf{S}$ is factorized also.

Using the above mentioned notations, the Eq. (2) can be written as given below.

$$\mathbf{M'} = \begin{pmatrix} S_8 & 0 \\ 0 & S_8 \end{pmatrix} \left\{ \mathbf{c_r Q^t B2^t B1^t P^t D^t} \begin{bmatrix} X_1 & X_2 & X_3 \\ X_4 & X_5 & X_6 \\ X_7 & X_8 & X_9 \end{bmatrix} \mathbf{DPB1B2Qc_c} \right\} \begin{pmatrix} S_8^t & 0 \\ 0 & S_8^t \end{pmatrix} \tag{4}$$

Table 2. Computational requirement of Modified MBIMC

Operations	Mults.	Add
Two multiplication of B_1 matrix (requires zero operations per 24×24 matrix, assuming 16 non-zero coeff.)	0	0
Two multiplications of B_2 matrix (requires 72a per 24×24 matrix, assuming 16 non-zero coeff.)		144
One multiplication of J_5 matrix (requires 36m, 336a per 24×24 matrix, assuming 16 non-zero coeff.)	72	672
One multiplication of K_5 matrix (requires 36m, 336a per 24×24 matrix, assuming 16 non-zero coeff.)	72	672
Four 2D DCT operation (requires 40m, 232a per 8×8 block)	160	928
Total computations	232	1744

The matrix $\begin{bmatrix} X_1 & X_2 & X_3 \\ X_4 & X_5 & X_6 \\ X_7 & X_8 & X_9 \end{bmatrix}$ of size 24×24 in Eq. (4) representing the nine adjacent DCT blocks will now contains many zeros (assuming only 16 non-zero coefficients in each block).

The matrix multiplication of $\mathbf{B_1}/\mathbf{B_1^t}$, $\mathbf{B_2}/\mathbf{B_2^t}$, J_r and K_c matrices with the above mentioned 24×24 sparse matrix can further be optimized to reduce the computational complexity of the IMC operation. The multiplication of $\mathbf{B_1}/\mathbf{B_1^t}$ requires no operations. The $\mathbf{B_2}/\mathbf{B_2^t}$ matrix multiplication requires 72 additions only. It may be noted that the J_r and K_c matrices are sparse matrices and both have similar kind of structure. An efficient matrix multiplication technique for J_r can also be implied for K_c. We have optimized the matrix multiplication of J_r and K_c with 24×24 sparse DCT matrix following the similar strategy as discussed in [3]. The computations required to perform matrix multiplication of J_r matrices with 24×24 sparse DCT matrix are calculated and given in Table 1.

Let us consider a case when $r = c = 5$ in Eq. (4). This requires maximum number of computations to perform IMC (refer Table 1). That is why it is considered for finding the computational requirements in the worst case. Total operations required to perform the IMC (when $r = c = 5$) in our proposed modification using Eq. (4) for each 16×16 block is 232m + 1744a (see Table 2). This requires 0.9m + 6.8a operations per pixel which is significantly less from the MBIMC scheme. Here **'m'** and **'a'** represents a multiplication and an addition operation respectively. If we assume that one multiplication is equivalent to three machine instructions and one addition is one machine instruction (refer [6]). We get approximately 70% improvement in speed over MBIMC scheme with an average reduction of 0.4 dB in psnr.

3. PROPOSED VIDEO DOWNSCALING SYSTEM

Our proposed video downscaling system is shown in Fig. 2. The input video is VLC decoded and inverse quantized to get the DCT blocks. The optimized MBIMC scheme is used to convert inter blocks in to intra blocks. All the intra blocks are downscaled by two using DCT domain based algorithm [1]. In the lower half portion of the Fig. 2, the downscaled DCT frames are re-encoded as per MPEG standards. AMVR system [6] is used to re-estimate the motion vector which is fed to DCT based motion compensation block. The MTSS [7] is used to determine the type of the macroblock. As per MTSS, a downscaled macroblock is coded as intra macroblock if and only if there are more than two intra coded macroblocks (out of four adjacent macroblocks) in the original video stream. Since motion vectors are computed using AMVR technique[6], the modified MBIMC scheme is used to compute the predicted macroblock for downscaled video. One should note here that all the operations are being performed in the DCT do-

Table 3. Computational complexities of different video downscaling systems for downscaling a P frame from CIF resolution to QCIF resolution

Function	Complexity			
	Mults.	Adds	Shifts	Total Cost
Spatial Domain based downscaling				
Input CIF frame processing				
Inverse Quant. + IDCT (144m, 464a per 8×8 block)	228096	734976		
Inverse Motion Compensation (256a per 16×16 block)		101376		
Output QCIF frame processing				
Downscale by 2 (3a, 1s per pixel)		76032	25344	
Full search ME (± 15 pels, 738048a per 16×16 block)		73066752		
Motion Compensation (256a per 16×16 block)		25344		
DCT + Quant. (144m, 464a per 8×8 block)	57024	183744		
Total	285120	74188224	25344	
Total Operation count (Add = 1 op, shift = 1 op, Mult. = 3ops)				**75068928**
Hybrid (Spatial + DCT domain) downscaling				
Input CIF frame processing				
Inverse Quant. + IDCT (144m, 464a per 8×8 block)	228096	734976		
Inverse Motion Compensation (256a per 16×16 block)		101376		
Output QCIF frame processing				
Downscale by 2 (3a, 1s per pixel)		76032	25344	
AMVR (9m, 30a, 1shift per 16×16 block)	891	2970	99	
Motion Compensation (256a per 16×16 block)		25344		
DCT + Quant. (144m, 464a per 8×8 block)	57024	183744		
Total	286011	1124442	25443	
Total Operation count (Add = 1 op, shift = 1 op, Mult. = 3ops)				**2007918**
Proposed Pure DCT domain based downscaling				
Input CIF frame processing				
Inverse Quant. (64m per 8×8 block)	101376			
IMC using modified MBIMC ((0.9m, 6.8a per pixel)	91238	689357		
(assuming only 16 non-zero coeff.)				
Output QCIF frame processing				
DCT Downscale by 2 (1.25m, 1,25a per pixel)	126720	126720		
AMVR (9m, 30a, 1shift per 16×16 block)	891	2970	99	
DCT domain MC (0.9m, 6.8a per pixel)	22810	172339		
(assuming only 16 non-zero coeff.)				
Quant. (64m per 8×8 block)	25344			
Total	368379	991386	99	
Total Operation count (Add = 1 op, shift = 1 op, Mult. = 3ops)				**2096622**

Table 4. Comparison of video downscaling methods

	PSNR (dB)								
	Spatial Domain			Hybrid Domain			DCT Domain		
video	Y	U	V	Y	U	V	Y	U	V
Coastguard	25.17	32.54	32.55	24.96	32.45	32.45	25.13	42.22	43.16
Foreman	28.61	32.20	32.08	28.29	32.00	31.92	29.42	40.09	41.09
Susi	33.90	32.94	32.44	33.86	32.93	32.43	36.83	49.92	49.23
Tennis	24.98	32.36	31.58	24.97	32.36	31.57	26.49	41.60	41.95

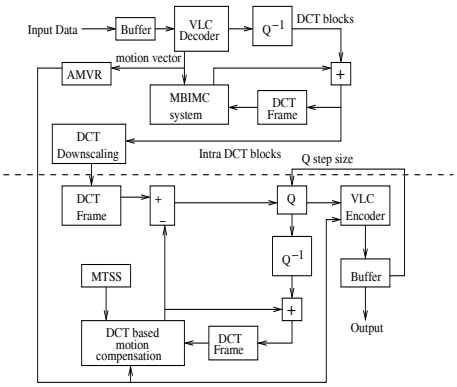

Fig. 2. Proposed DCT domain based Video Downscaling System

main. The DCT being linear orthonormal transform, the predicted DCT blocks can directly be subtracted from intra DCT blocks, to compute the error DCT blocks. Finally the error bits are computed and sent to rate control module which sets a new quantization step size to control the bit rate of the downscaled video.

Our proposed DCT domain based system is compared with spatial and hybrid domain [6] video downscaling systems. The computational complexities of each system is given in Table 3. The complexity of each operation is shown in number of multiplication, addition and shift operations. In Table 3, **'m'** represents multiplication, **'a'** represents addition operations. It is assumed that one multiplication is equivalent to three machine operation and an addition and a shift is equivalent to single machine operation respectively [6]. It is shown in the Table 3 that the proposed downscaling system requires few more number of operations than hybrid system but it is approximately 36 times faster than spatial domain system.

4. EXPERIMENTAL RESULTS

We have implemented the spatial domain, hybrid domain and our proposed DCT domain based video downscaling system. The four different videos with I and P frames are used to record the experimental results, i.e. "Coastguard", "Foreman", "Susi" and "Tennis". All sequences are CIF resolution with 352 pixels and 288 lines at 1.5 Mpbs. The downscaled video is stored at 500 Kbps bit rate. The downscaled frames are upsampled to original resolution to compute the PSNR with original video frames. The spatial domain upsampling is used for spatial and hybrid domain technique and DCT domain based upsampling technique[1] is used to perform upsampling of video frames, downscaled by our proposed system. The average PSNR is shown in Table 4.

In Table 4, the average PSNR of 150 frames of each video for luminance (Y) and chrominance (U and V) is shown. We can see that in most cases the average PSNR of our proposed system is improved by 1 dB or above compared to those obtained by hybrid or spatial domain approaches.

5. CONCLUSION

In this paper, we proposed a DCT domain based video downscaling system. It is shown that our proposed downscaling system performs all the operations solely in DCT domain. It is computationally as efficient as a hybrid domain system, but It produces better quality of video output than a hybrid system. The quality of downscaled video of our proposed system is also better than the spatial domain approach which is 36 times more computationally expensive. The MBIMC scheme and the modification discussed in this paper can also be combined with method proposed in [8] for arbitrary downsizing of the video.

6. REFERENCES

[1] R Dugad and N. Ahuja, A fast scheme for image size change in the compressed domain, IEEE Trans. Circuits Syst. Video Technology, vol. 11, pp. 461-474, Apr. 2001.

[2] S. F. Chang and D. G. Messerschmitt, "Manipulation and compositing of MC-DCT compressed video," IEEE J. Select. Areas Commun., vol. 13, pp. 1-11, jan. 1995.

[3] N. Merhav and V. Bhaskaran, Fast algorithms for DCT-domain image downsampling and for inverse motion compensation, IEEE Trans. Circuits Syst. Video Technology, vol. 7, pp. 468-476, June 1997.

[4] P. A. A. Assuncao and M. Ghanbari, "Transcoding of MPEG-2 video in the frequency domain," in ICASSP 1997, 1997, pp. 2633-2636.

[5] Junehwa Song, Boon-Lock Yeo, "A Fast Algorithm for DCT-Domain Inverse Motion Compensation Based on Shared Information in a Macroblock," IEEE Trans. on Circuits and Systems for Video Technology, vol. 10, No. 5, Aug. 2000.

[6] Bo Shen, Ishwar K. Sethi, Bhaskaran Vasudev, Adaptive Motion Vector Resampling for Compressed Video Downscaling, IEEE Trans. Circuits Syst. Video Technology, Vol. 9, No. 6, Sept., 1999.

[7] M. R. Hashemi, L. Winger, and S. Panchanathan, Macroblock type selection for compressed domain downscaling of MPEG video, IEEE Canadian Conf. on Elect. and Comp. Engg.,Canada, May9-12, 1999.

[8] Haiyan Shu, Lap-Pui Chau, An Efficient Arbitrary Downsizing Algorithm for Video Transcoding, IEEE Trans. Circuits Syst. Video Technology, Vol. 14, No. 6, June 2004.