WSEAS Transactions on Information Science & Applications, Vol. 2, Issues, Marc 2005, pp. 295-300.

Complexity Reduction Tools for MPEG-2 to H.264 Video Transcoding

HARI KALVA, BRANKO PETLJANSKI, and BORKO FURHT
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 333431
USA
hari@cse.fau.edu, branko@voyager.ee.fau.edu, borko@cse.fau.edu, http://www.cse.fau.edu

Abstract: - The wide deployment of MPEG-2 video for digital video applications today and the expected use of the recently standardized H.264 video create a strong need for video transcoding between these formats. One of the key problems in video transcoding is complexity reduction. In this paper we present tools to enable low complexity transcoding of MPEG-2 video to the H.264 format. The proposed transcoding approach uses the MPEG-2 DCT coefficients to estimate the direction and derives the H.264 prediction modes based on this estimates. The accuracy of this mode estimation can be improved if DCT coefficients of the 16x16 and 4x4 blocks are available. The orthonormal and separable properties of the DCT transform matrix are used to develop the DCT block combination and splitting tools. The proposed tools have low computational complexity and reduce the overall transcoding complexity substantially.

Key-Words: - MPEG-2, H.264, video transcoding, complexity reduction, DCT domain

1 Introduction

Most of the video used in the digital entertainment applications uses the MPEG-2 video coding algorithm. The recently developed H.264 video is very efficient, compressing the video to about one-half of MPEG-2 bitrates at the same quality [4][14]. The H.264 video standard is expected to replace MPEG-2 over the next several years. While mobile phones primarily use MPEG-4 video format today, the H.264 video is expected to appear in mobile devices soon because of the bandwidth and quality advantages over MPEG-4.

The H.264 standard is intended for use in a wide range of applications including high quality and high-bitrate digital video applications such as DVD and digital TV and low bitrate application such as video delivery to mobile devices. However, the computing and communication resources of the end user terminals make it impossible to use the same encoded video content for all applications. For example, the high bitrate video used for a digital TV broadcast cannot be used for streaming video to a mobile terminal. For delivery to mobile terminals, we need video content that is encoded at lower bitrate and lower resolution suitable for low-resource mobile terminals. Pre-encoding video at a few discrete bitrates leads to inefficiencies as the device capabilities vary and pre-encoding video bitstreams for all possible receiver capabilities is impossible. Furthermore, the receiver capabilities such as available CPU, available battery, and available bandwidth may vary during a session and a pre-encoded video stream cannot meet such dynamic needs. To make full use of the receiver capabilities and deliver video suitable for a receiver, video transcoding is necessary. A transcoder for such applications takes a high bitrate video as input and transcodes it to a lower bitrate and/or lower resolution video suitable for a mobile terminal.

The wide use of MPEG-2 video today and the expected use of H.264 video in next generation devices, especially mobile devices, requires transcoding video from MPEG-2 to H.264 format to bridge video communication between such devices. The MPEG-2 and H.264 video coding algorithms are both based on hybrid video coding principles of motion compensated transform coding. However, the algorithms differ substantially in the motion compensation as well as the transform coding resulting in a very high transcoding complexity [7]. These substantial differences also make the earlier MPEG-2 to MPEG-4 transcoding and complexity reduction approaches unsuitable [13][8]. The key contributions of this paper are the transform domain complexity reduction tools for MPEG-2 to H.264 transcoding. The existing work in the H.264 transcoding primarily use transform coefficient conversion and do not estimate MB modes properly resulting inefficiencies [2][15]. We propose a complexity reduction approach that estimates the directional textures of the DCT blocks obtained

from the MPEG-2 video decoding process to estimate the MB prediction modes in H.264. To improve the accuracy of the direction estimates and the MB modes, we use DCT block combination and splitting techniques to derive 16x16 and 4x4 DCT blocks from the 8x8 DCT blocks recovered from the MPEG-2 decoding process.

This rest of the paper is organized as follows: section 2 summarizes the transcoding complexity, section 3 introduces the mode computation using direction estimates, and section 4 discusses the DCT block combination and splitting techniques. Conclusions are presented in section 5.

2 MPEG-2 to H.264 Video Transcoding

The sophisticated tools used in the H.264 coding makes transcoding more complex. For example, if multiple reference frames are used for motion compensation in H.264 video encoding, the transcoder cannot as easily take advantage of the motion vectors. Also, transcoding a video at reduced temporal resolution reduction becomes more complex. Thus complexity reduction is essential to develop realtime transcoding systems.

The key to reducing the transcoding complexity is reusing the information gathered during MPEG-2 decoding stage. In our past work on MPEG-2 to MPEG-4 transcoding we used DCT-domain down sampling and motion vector scaling to reduce the complexity [13]. We also showed that realtime transcoding is possible in software when algorithmic and software optimisations are used.

The MPEG-2 and H.264 video coding algorithms differ substantially and the results of MPEG-2 to MPEG-4 transcoding work cannot be applied. There has been some work in converting DCT coefficients into H.264 transform coefficients [15]. However, our experience with MPEG-2 to MPEG-4 transcoding has shown that converting DCT coefficients to the coefficients in the transcoded domain involves matrix transformations and is computationally expensive. Furthermore, the transform coding in H.264 has much lower complexity compared to DCT and optimal mode selection may not be possible with direct transformations. These conclusions led us to focus on reducing the complexity of prediction – both for inter and intra frames. The existing work related to H.264 transcoding [2][15] does not address coding mode prediction which is most compute intensive portion of H.264 coding. We propose complexity reduction techniques that reduce mode estimation complexity by taking advantage of the DCT coefficients from the MPEG-2 decoding stage.

2.1 Complexity of MB Mode prediction

The macro block (MB) mode prediction process in H.264 encoding is compute intensive and takes up over 50% of the encoding time. For intra-coded frames, all MBs are intra coded. For inter-coded frames, an MB could be inter or intra coded making mode decision for inter-frames more expensive. For intra mode prediction, the decision making process has to evaluate the prediction and compute the residual. There are four prediction modes for 16x16 blocks and 9 prediction modes for each of the 16 4x4 blocks (or four 8x8 blocks). Optimal mode decision requires exhaustive evaluation of all the possible modes and is computationally expensive. Figure 1 shows the prediction direction for the 9 intra prediction modes. The candidate pixels from the neighbouring blocks used to form predictions are shown in upper case in the figure.

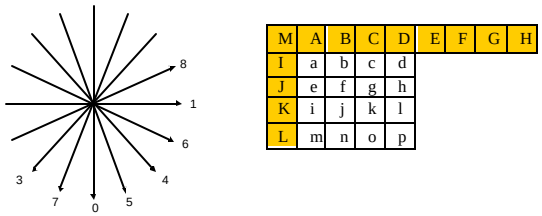

Figure 1. Intra Mode Prediction in H.264

In inter-coded frames, an MB has to be evaluated for intra-coding and inter coding. The final coding mode is determined by evaluating rate-distortion tradeoffs. For inter coding the number of candidate predictors and prediction modes that have to be evaluated are significantly higher and the options and complexity increase proportionally with the number of reference frames used. The prediction complexity can be significantly reduced if can take advantage of the DCT coefficients and motion vectors from the MPEG-2 decoding stage.

3 Mode Computation Using Direction Estimates

The DCT coefficients are readily available after decoding MPEG-2 video bitstreams. For intra coded MPEG-2 blocks, the full DCT coefficients are available. For inter coded DCT, only the DCT coefficients of the residual are available. Full DCT coefficients can be reconstructed using the motion

compensated DCT manipulation proposed by Chang [1]. Full DCT coefficients can also be computed by computing a DCT on the reconstructed pixels. The DCT coefficients contain significant information about the nature of the coded video. The DCT domain information has been applied for transform domain edge detection [11] and scene change detection [9].

Figure 2 shows the edge model used for computing the direction. It has been shown that the ratio of vertical energy to the horizontal energy gives the tangent of the ideal edge passing through the centre of the block [10][11]. Consider the edge model shown in the Figure 2 and the properties of the DCT. Let $F(u,v)$ represent the DCT coefficients of an 8x8 block $f(x,y)$ obtained during the MPEG-2 decoding stage.

$$\tan\theta = \frac{\sum_{u=0}^{7} |F(u,0)|}{\sum_{v=0}^{7} |F(0,v)|} \qquad (1)$$

$$m = \frac{1}{n^2}\sum_{0}^{n-1}\sum_{0}^{n-1} f(x,y) = \frac{F(0,0)}{n} + 128 \qquad (2)$$

$$s^2 = \frac{1}{64}\sum_{u=0}^{7}\sum_{u=0}^{7} F(u,v)^2 \qquad (u,v) \neq (0,0) \qquad (3)$$

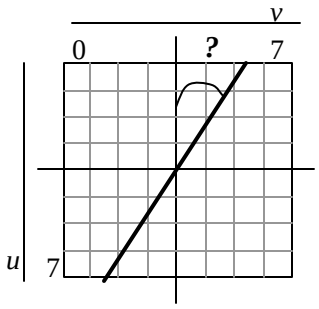

Figure 2. Edge Orientation in an 8x8 block

The edge angle computed can be used to predict the prediction modes shown in Figure 1. The full DCT coefficients can be used to estimate the edge angle, the average energy of the block, the variance, and the standard deviation. The equations (1), (2), and (3) can be used to compute the angle, the average energy (mean), and a measure of the variance (standard deviation), respectively. These metrics can be used to estimate the H.264 prediction modes. The edge direction gives an indication of the best directional predictor. Since intra prediction is based on pixels from the neighbouring blocks, by evaluating the standard deviation of the 16 candidate pixels [4], an estimate of the dc mode is obtained. The DC coefficient, *F(0,0),* gives an estimate of the average energy of a block. The edge angle estimate works relatively well for compressed domain edge detection [11] but has to be refined for mode estimation. The goal in intra prediction is to minimize the residual; typically a measure such as the sum of absolute differences (SAD) between the original and predicted pixels is used. Our initial results have shown that the direction computed using the edge angle does not always give the optimal mode (minimum SAD). The modes 0, 1, and 2 were predicted relatively accurately. Mode 0 and mode 1 are predicted using the edge angle. Mode 2 is predicted using the standard deviation or by comparing the DC coefficient to the scaled average of the prediction pixels with equally good results from either method. The results also indicate that the general prediction direction – horizontal or vertical orientation – can be evaluated accurately. Even using these initial results, the intra mode prediction complexity is reduced by more than 50% as we have to evaluate only the vertical or horizontal modes. We expect the reduction in video quality to be to be less than 0.5 dB compared to the exhaustive search.

The DCT energy can also be used to estimate the activity in the blocks and determine whether a 16x16 or 4x4 intra prediction mode should be used. Alternatively, the SAD can be calculated using the best 16x16 and the best 4x4 modes estimated in order to make the final decision. To compute prediction modes for the 16x16 and 4x4 blocks, the DCT of the 16x16 and 4x4 blocks can be computed from the 8x8 DCT blocks using the DCT combination and segmentation approach discussed in section 4.

Inter frame prediction in H.264 is highly complex. The DCT coefficients decoded from inter-MBs of MPEG-2 video are computed from the motion compensated residual. The actual DCT coefficients can be computed using the motion compensated DCT (MC DCT) formulation proposed in [1]. The decoded motion vectors together with MC DCT coefficients can be used to estimate prediction modes and prediction block sizes for H.264 video.

Similar to the intra-mode estimation, the DCT coefficients can be used to reduce the coding mode options for inter prediction. The DCT energy can be used to determine the variable block size for motion estimation. Higher activity indicates a higher level of detail and possibly warrants smaller block size. The MPEG-2 motion vectors can be used to reduce the complexity of reference picture selection in

H.264. The object motion indicated by the motion vectors of a block in successive frames can be used to select candidate reference frames and motion vector estimates.

4 Transform Domain DCT Block Combination and Splitting

The MPEG-2 video coding algorithm uses a fixed 8x8 block for DCT and 16x16 block for motion estimation. The H.264 video coding algorithm offers more coding options by supporting variable block size prediction for inter as well as intra block coding. The intra prediction modes can use 16x16 or 4x4 block sizes (8x8 block size can also be used optionally). The inter-coding mode supports 9 different block sizes for motion compensation. The best intra prediction mode can be estimated using the direction estimates of the blocks. The 8x8 DCT blocks recovered from the MPEG-2 decoding stage can be used to estimate the prediction modes of 8x8 blocks in H.264 as well as to reduce the complexity of motion estimation in inter coding. While the 16x16 and 4x4 prediction modes can be estimated based on the 8x8 MPEG-2 DCT coefficients, the accuracy can be improved if 4x4 or 16x16 DCT coefficients are available. Computing the 16x16 and 4x4 DCT coefficients after applying IDCT to the 8x8 DCT blocks is computationally expensive. We developed an approach based on DCT matrix transformations for DCT block combination and splitting. The resulting 4x4 and 16x16 DCT blocks can be use to estimate the intra modes using the techniques discussed in Section 3.

Jiang and Feng discuss a generalized approach to prove the linear relationship between the DCT block and its sub-blocks [5]. This approach is complex but also yields block combination and splitting techniques. Our approach is simple and is based on the fact that the DCT transform matrix is orthonormal and separable; i.e., the transform can be represented as $\tilde{X} = CXC^T$, where C is the orthonormal transform matrix ($C^{-1} = C^T$).

4.1 The 2D DCT

The general form of a 2D DCT is given by equation (4) and the inverse transform, 2D IDCT, is given by equation (5) [10].

$$X[k_1,k_2] = a[k_1]\,a[k_2]\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} x[n_1,n_2]\cos\left(\frac{p(2n_1+1)k_1}{2N_1}\right)\cos\left(\frac{p(2n_2+1)k_2}{2N_2}\right) \quad (4)$$

$$x[n_1,n_2] = \sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} a[k_1]\,a[k_2]X[k_1,k_2]\cos\left(\frac{p(2n_1+1)k_1}{2N_1}\right)\cos\left(\frac{p(2n_2+1)k_2}{2N_2}\right) \quad (5)$$

This may be compactly written in the matrix form as, $\tilde{X} = CXC^T$ and the inverse transform as, $X = C^T\tilde{X}C$, where X is an image of size $N_1 \times N_2$, $\tilde{X}$ is DCT transform of X, and C is cosine matrix as shown in equation (6). For MPEG-2 video, $N_1 = N_2 = 8$ and an 8x8 DCT is used.

$$C_{8x8} = \begin{bmatrix} 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 \\ 0.4904 & 0.4157 & 0.2778 & 0.0975 & -0.0975 & -0.2778 & -0.4157 & -0.4904 \\ 0.4619 & 0.1913 & -0.1913 & -0.4619 & -0.4619 & -0.1913 & 0.1913 & 0.4619 \\ 0.4157 & -0.0975 & -0.4904 & -0.2778 & 0.2778 & 0.4904 & 0.0975 & -0.4157 \\ 0.3536 & -0.3536 & -0.3536 & 0.3536 & 0.3536 & -0.3536 & -0.3536 & 0.3536 \\ 0.2778 & -0.4904 & 0.0975 & 0.4157 & -0.4157 & -0.0975 & 0.4904 & -0.2778 \\ 0.1913 & -0.4619 & 0.4619 & -0.1913 & -0.1913 & 0.4619 & -0.4619 & 0.1913 \\ 0.0975 & -0.2778 & 0.4157 & -0.4904 & 0.4904 & -0.4157 & 0.2778 & -0.0975 \end{bmatrix} \quad (6)$$

4.2 Resizing DCT Blocks

The construction of DCT coefficients of a 16 x 16 block from given four 8 x 8 blocks of horizontal and vertical frequency coefficients would require the computation of the IDCT of the given blocks, spatially merge resulting blocks to create a 16 x 16 block and then apply 16 x 16 DCT on it.

Splitting a DCT block into its sub-blocks, for example, splitting an 8x8 DCT into four 4x4 DCTs, requires the IDCT of the 8x8 block, partitioning of the 8 x 8 block of the spatial values into four 4 x 4 blocks and then performing a 4 x4 DCT on them to generate 4 x 4 blocks of horizontal and vertical frequency coefficients.

Both the aforementioned methods require significant computational resources; first we need to transform the frequency domain coefficients to the spatial domain, merge resulting blocks (or partition a block) and then apply DCT to generate desired block size of DCT coefficient.

4.2.1 Combining 8x8 Blocks to Create 16x16 DCT

Let $\tilde{X}_{8x8}^{mn}$ denote an 8 x 8 block of horizontal and vertical frequency coefficients, where *m* and *n* represent the spatial position of a 8x8 block within a macro block (16x16 block). If C_{8x8} is the cosine transformation matrix for an 8 x 8 block size and C_{16x16} is the cosine transformation matrix for 16 x 16 block sizes, then resulting DCT coefficients $\tilde{X}_{16x16}$ can be calculated with equation (7).

$$\tilde{X}_{16x16} = C_{16x16}\begin{bmatrix} C_{8x8} & 0 \\ 0 & C_{8x8} \end{bmatrix}^T \begin{bmatrix} \tilde{X}_{8x8}^{00} & \tilde{X}_{8x8}^{01} \\ \tilde{X}_{8x8}^{10} & \tilde{X}_{8x8}^{11} \end{bmatrix} \begin{bmatrix} C_{8x8} & 0 \\ 0 & C_{8x8} \end{bmatrix} C_{16x16}^T \quad (7)$$

Observing equation (7) closely, the inner matrix multiplications are essentially performing an IDCT, converting the four 8x8 DCT blocks to the spatial domain. Subsequently, a 16x16 DCT is applied to the 16x16 image block giving a 16x16 DCT block. Since C_{8x8} and C_{16x16} are constant cosine matrices, they can be replaced with a new matrix T_{16x16}.

$$T_{16x16} = C_{16x16}\begin{bmatrix} C_{8x8} & 0 \\ 0 & C_{8x8} \end{bmatrix}^T \quad (8)$$

As a result, transformation in equation (7) can be rewritten as shown in equation (9).

$$\tilde{X}_{16x16} = T_{16x16}\begin{bmatrix} \tilde{X}_{8x8}^{00} & \tilde{X}_{8x8}^{01} \\ \tilde{X}_{8x8}^{10} & \tilde{X}_{8x8}^{11} \end{bmatrix} T_{16x16}^T \quad (9)$$

The preceding discussion is valid for all blocks with a block size that is a multiple of the sub-block size. A similar approach can be used to derive a 32x32 DCT from four 16x16 DCT blocks.

4.2.1 Splitting 8x8 DCT to Create 4x4 DCT Blocks

Let $\tilde{X}_{8x8}$ denote an 8 x 8 block of horizontal and vertical frequency coefficients, C_{8x8} is cosine transformation matrix for block of size 8 x 8 and C_{4x4} is cosine transformation matrix for blocks 4 x 4. The resulting DCT coefficients, $\tilde{X}_{4x4}^{mn}$, are obtained using equation (10).

$$\begin{bmatrix} \tilde{X}_{4x4}^{00} & \tilde{X}_{4x4}^{01} \\ \tilde{X}_{4x4}^{10} & \tilde{X}_{4x4}^{11} \end{bmatrix}_{8x8} = \begin{bmatrix} C_{4x4} & 0 \\ 0 & C_{4x4} \end{bmatrix} C_{8x8}{}^T \tilde{X}_{8x8} C_{8x8} \begin{bmatrix} C_{4x4} & 0 \\ 0 & C_{4x4} \end{bmatrix}^T \quad (10)$$

$$T_{8x8} = \begin{bmatrix} 0.7071 & 0.6407 & 0 & -0.2250 & 0 & 0.1503 & 0 & -0.1274 \\ 0 & 0.2940 & 0.7071 & 0.5594 & 0 & -0.2492 & 0 & 0.1964 \\ 0 & -0.0528 & 0 & 0.3629 & 0.7071 & 0.5432 & 0 & -0.2654 \\ 0 & 0.0162 & 0 & -0.0690 & 0 & 0.3468 & 0.7071 & 0.6122 \\ 0.7071 & -0.6407 & 0 & 0.2250 & 0 & -0.1503 & 0 & 0.1274 \\ 0 & 0.2940 & -0.7071 & 0.5594 & 0 & -0.2492 & 0 & 0.1964 \\ 0 & 0.0528 & 0 & -0.3629 & 0.7071 & -0.5432 & 0 & 0.2654 \\ 0 & 0.0162 & 0 & -0.0690 & 0 & 0.3468 & -0.7071 & 0.6122 \end{bmatrix} \quad (12)$$

$$T_{8x8} = \begin{bmatrix} C_{4x4} & 0 \\ 0 & C_{4x4} \end{bmatrix} C_{8x8}{}^T \quad (11)$$

Since C_{8x8} and C_{4x4} are constant cosine matrices, they can be replaced with a new constant matrix T_{8x8} shown in equation (12).

$$\begin{bmatrix} \tilde{X}_{4x4}^{00} & \tilde{X}_{4x4}^{01} \\ \tilde{X}_{4x4}^{10} & \tilde{X}_{4x4}^{11} \end{bmatrix} = T_{8x8}\tilde{X}_{8x8}T_{8x8}^T \quad (13)$$

As a result, transformation in equation (10) can be rewritten as shown in equation (13).

4.3 Computational Complexity of Block Splitting

The alternative to block splitting, one 8x8 IDCT followed by four 4x4 DCTs on the resulting pixels, is computationally expensive. The symmetry and sparseness of the transformation matrix T_{8x8} can be exploited to reduce the computations. Computational complexity of block splitting as shown in equation (13) is discussed is this section.

The transformation matrix T_{8x8} shown in equation (12) consists of five non-zero elements per row. For each element of $\tilde{X}_{8x8}T_{8x8}^T$ matrix, we need five multiplications and four additions. There are total of 64 elements in the matrix, which results in 320 multiplications and 256 additions. Mode estimation algorithm described in Section 3 does not require calculating all DCT coefficients in sub-matrix; only the coefficients from first row and first column in resulting $\tilde{X}_{4x4}^{mn}$ matrices. That results in total of 440 multiplications and 352 additions. Taking advantage of the symmetry of the transform matrix reduces the computations to 100 multiplications and 288 additions. The DCT operation results in energy compaction in frequency domain and together with the quantization applied in video coding results in very few non-zero coefficients. The number of non-zero coefficients is a function of the content and the bitrate (quantization) used. If only the DC value of the DCT block is non-zero, the computations reduce to 1 multiplications and 1 additions. If the first row and column of the DCT matrix are non-zero, the mode estimation with block splitting would require 40 multiplications and 36 additions. The complexity thus is much lower than direct transformations.

The complexity of intra mode estimation for the four 4x4 blocks of an 8x8 block requires 2096 additions, 596 shift operations, and 12 multiplications. Assuming the cost of addition and shift operations is 1 cycle and the multiplication is 10 cycles, the exhaustive mode estimation takes 2704 cycles. The block splitting approach requires 1288 cycles for block splitting and another 100 cycles of computation for mode estimation which represents about 50% savings compared to the exhaustive approach. Since the estimation may not match the optimal mode computed using the H.264 exhaustive search, the video coding efficiency drops slightly in terms of PSNR. The quality of the mode estimation can be improved by using the proposed technique to estimate the general direction

(horizontal or vertically dominant) and then use exhaustive search to identify the best match.

5 Conclusion

In this paper we presented the problem of video transcoding and its complexity. The MPEG-2 and H.264 video coding algorithms differ substantially and transcoding from MPEG-2 to H.264 cannot use tools and techniques used in well studied approaches such as MPEG-2 to MPEG-4 transcoding. Complexity reduction is critical to enable realtime services and high capacity transcoders.

We proposed an innovative approach to complexity reduction by using the direction estimates in DCT blocks. To improve the accuracy of the direction estimates, we propose to use DCT block combination and splitting techniques to estimate the texture directions in 4x4 and 16x16 blocks. The transform matrices have a large number of zeros and result in reduced number of multiplications. The small number of non-zero coefficients in quantized DCT blocks reduces the multiplications further. The proposed approach requires less computation than explicit IDCT followed by DCT. The directional estimates and DCT block transformations together reduce the complexity of MPEG-2 to H.264 video transcoding substantially. For intra mode estimation the mode computation cost is reduced by about 50%. The proposed methods are applicable in transcoding any DCT-based video compression to the H.264 format.

References:

[1] S.F. Chang and D.G. Messerschmitt, Manipulation and Composting of MC-DCT Compressed Video, *IEEE Journal of Selected Areas in Communications*, vol. 13, pp.1-11, January 1995.

[2] C. Chen, P-H.Wu, H. Chen, MPEG-2 To H.264 Transcoding, *Picture Coding Symposium*, 15-17 Dec, 2004.

[3] Implementation Studies Group, Main Results of the AVC Complexity Analysis, *MPEG Document N4964, ISO/IEC JTC11/SC29/WG11*, July 2002.

[4] ISO/IEC JTC1/SC29/WG11, Text of ISO/IEC FDIS 14496-10: Information Technology – Coding of audio-visual objects – Part 10: Advanced Video Coding, *MPEG Document N5555, ISO/IEC JTC11/SC29/WG11*, March 2003.

[5] J. Jiang and G. Feng, The Spatial Relationship of DCT Coefficients Between a Block and Its Sub-blocks, *IEEE Transactions On Signal Processing*, Vol. 50, No. 5, May 2002, pp 1160-1169.

[6] H. Kalva, Error Resilient Transmission of H.264 Video, *Seventh INFORMS Telecommunications Conference*, March 2004.

[7] H. Kalva, Issues in H.264/MPEG–2 Video Transcoding, *Proceedings of the IEEE Consumer Communications and Networking Conference*, Jan. 2004, pp. 657–659.

[8] H. Kalva, A. Vetro, and H. Sun, Performance Optimization of the MPEG–2 to MPEG–4 Video Transcoder, *SPIE Conference on Microtechnologies for the New Millennium*, VLSI Circuits and Systems, May 2003 (invited paper).

[9] S.-W. Lee, Y.-M. Kim, and S.W. Choi, Fast Scene Change Detection using Direct Feature Extraction from MPEG Compressed Videos, *IEEE Transactions on Multimedia*, Vol 2, No. 4, December 2000, pp. 240-254.

[10] K.R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*, Academic Press, 1990, ISBN0-12-580203-X.

[11] B. Shen and I.K. Sethi, Direct feature extraction from compressed images, *Proc. SPIE Storage & Retrieval for Image and Video Databases IV*, Vol.2670, 1996.

[12] A. Vetro, C. Christopoulos, and H. Sun, Video transcoding architectures and techniques: an overview, *IEEE Signal Processing Magazine*, Vol. 20 , No. 2, March 2003, pp. 18 – 29.

[13] A. Vetro, T.Hata, N. Kuwahara, H. Kalva, and S. Sekiguchi, Complexity–quality analysis of transcoding architectures for reduced spatial resolution, *IEEE Transactions on Consumer Electronics*, Volume: 48 Issue: 3 , August 2002, Page(s): 515 –521.

[14] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the H.264/AVC Video Coding Standard, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 13, No. 7, July 2003.

[15] J. Xin, A. Vetro, and H. Sun, Converting DCT Coefficients to H.264/AVC Transform Coefficients, *Proceedings of the IEEE Pacific-Rim Conference on Multimedia (PCM)*, IEEE PCM 2004, November 2004.

[16] J. Xin, A. Vetro, and H. Sun, Efficient Macroblock Coding-mode Decision for H.264/AVC Video Coding, *Picture Coding Symposium*, 15-17 Dec, 2004.