DCT-BASED SUBPIXEL MOTION COMPENSATION AND FULLY DCT-BASED VIDEO CODER*

Ut-Va Koc
koc@bell-labs.com
Lucent Technologies Bell Labs
600 Mountain Ave.
Murray Hill, NJ 07974

and

K. J. Ray Liu
kjrliu@eng.umd.edu
Electrical Engineering Department
and Institute for Systems Research
University of Maryland at College Park
College Park, MD 20742

ABSTRACT

The feedback loop for temporal prediction of traditional implementations of an MPEG-compliant video coder requires conversion of images from the spatial domain to the transform domain and then back to the spatial domain due to the restriction of motion estimation schemes performed only in the spatial domain, severely limiting the throughput of the coder. However, it was shown that the low-complexity DCT-based motion estimation scheme (DXT-ME) can remove this bottleneck and thus reduce the overall coder complexity. In this paper, we develop efficient DCT-based motion compensation algorithms to complete the fully DCT-based motion compensated video coder. We demonstrate that the performance of this new coder structure is comparable to the conventional one while the overall coder complexity is lower.

1. INTRODUCTION

Many video coding standards, such as CCITT H.261, MPEG1, MPEG2, and HDTV, are based on the hybrid DCT motion-compensated scheme to remove spatial redundancy through Discrete Cosine Transform (DCT) and temporal redundancy through block-based motion estimation. The conventional implementations of a standard-compliant coder adopt the spatial-domain DCT motion-compensated video coder structure as shown in Fig. 1(a). The feedback loop of this architecture consists of a DCT, an Inverse DCT (IDCT) and a spatial-domain motion estimator (SD-ME) which is usually the full search block matching approach (BKM). In addition to the additional complexity added to the overall architecture, this feedback loop limits the throughput of the coder and becomes the bottleneck of a real-time high-end video codec.

2. FULLY DCT-BASED MOTION-COMPENSATED VIDEO CODER STRUCTURE

This bottleneck can be overcome by simplifying the feedback loop. The resulting architecture, called fully DCT-based motion-compensated video coder structure depicted in Fig. 1(b), has one simple element in the loop: DCT-domain motion predictor. The integer-pel and subpixel DCT-based motion estimation algorithms have been developed and reported in [5, 6, 7]. In this summary, we describe subpixel DCT-based motion compensation methods to complete the fully DCT-based motion-compensated video coder structure [4] based on these DCT-based motion estimation and compensation schemes.

3. SUBPIXEL DCT-BASED MOTION COMPENSATION

Without converting back to the spatial domain before motion compensation in order to exploit the benefits from the DCT-based motion estimation algorithms, DCT-based motion compensation schemes are essential to the realization of the fully DCT-based structure, In [1, 8], direct full-pixel motion compensation methods in DCT domain, as in Fig. 3(c), are developed for the use in decoders mainly. However, these schemes can be applied to the implementation of the fully DCT-based structure at a full pixel motion accuracy level.

For the case of subpixel motion, interpolation is used to predict interpixel values. According to the MPEG standards, bilinear interpolation is recommended for its simplicity in implementation and effectiveness in prediction, though it is well known that a range of other interpolation functions, such as cubic, spline, Gaussian, and Lagrange interpolations, can provide better approximation accuracy and more pleasant visual quality [9, 10, 3, 2]. The complexity argument is true if the interpolation operation is performed in the spatial domain, but in the DCT domain, it is possible to employ better interpolation functions than the bilinear interpolation without any additional computational load increase.

4. INTERPOLATION FILTER

*This work is supported in part by the ONR grant N00014-93-1-0566, NSF grant MIP9309506, and MIPS/MicroStar.

0-8186-8183-7/97 $10.00

For simplicity of derivations, we start with the one dimensional half-pel bilinear interpolation and then proceed to the two dimensional case of quarter-pel accuracy with other interpolation functions. Consider two one dimensional adjacent blocks, $x_{1a}(n)$ and $x_{1b}(n)$ for $n = 0, \ldots, N-1$ as shown in Fig. 2. We want to extract a block $\{x_2(n)\}_{n=0}^{N-1}$ displaced u pixels to the right of $x_{1a}(0)$ where u is supposed to be an odd multiple of 0.5 (i.e. half-pel motion). Therefore, we can show that

$$x_2(n) = \begin{cases} \frac{1}{2}[x_{1a}(N+n-i) \\ \quad +x_{1a}(N+n-i+1)], & 0 \le n \le i-2, \\ \frac{1}{2}[x_{1a}(N-1)+x_{1b}(0)], & n = i-1, \\ \frac{1}{2}[x_{1b}(n-i) \\ \quad +x_{1b}(n-i+1)], & N-1 \ge n \ge i, \end{cases} \quad (1)$$

where $i = \lceil u \rceil$. In the matrix form,

$$\vec{\mathbf{x}}_2 = \mathbf{G}_{BL}(i)\vec{\mathbf{x}}_{1a} + \mathbf{G}_{BR}(i)\vec{\mathbf{x}}_{1b}, \quad (2)$$

where $\vec{\mathbf{x}}_2$, $\vec{\mathbf{x}}_{1a}$, and $\vec{\mathbf{x}}_{1b}$ are the column vectors of $x_2(n)$, $x_{1a}(n)$ and $x_{1b}(n)$ respectively, and $\mathbf{G}_{BL}(i)$ and $\mathbf{G}_{BR}(i)$ are defined as follows:

$$\mathbf{G}_{BL}(i) = \frac{1}{2}\left\{\begin{bmatrix} \mathbf{0} & \mathbf{I}_i \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{I}_{i-1} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}\right\}, \quad (3)$$

$$\mathbf{G}_{BR}(i) = \frac{1}{2}\left\{\begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{I}_{N-i} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{I}_{N-i+1} & \mathbf{0} \end{bmatrix}\right\} \quad (4)$$

In the DCT domain,

$$\begin{aligned} \mathrm{DCT}\{\vec{\mathbf{x}}_2\} = {} & \mathrm{DCT}\{\mathbf{G}_{BL}(i)\}\mathrm{DCT}\{\vec{\mathbf{x}}_{1a}\} \\ & +\mathrm{DCT}\{\mathbf{G}_{BR}(i)\}\mathrm{DCT}\{\vec{\mathbf{x}}_{1b}\}. \end{aligned} \quad (5)$$

Here $\mathbf{G}_{BL}(i)$ and $\mathbf{G}_{BR}(i)$ can be regarded as bilinear interpolation filter matrices which act as a linear filter or transform. Therefore, $\mathbf{G}_{BL}(i)$ and $\mathbf{G}_{BR}(i)$ can be replaced by any FIR filter or interpolation function of finite duration (preferably with the length much smaller than the block size N).

5. BILINEAR INTERPOLATED SUBPIXEL MOTION COMPENSATION

For the 2-D case, if (u, v) is the displacement of the reconstructed block $\hat{\mathbf{B}}_{ref}$ measured from the upper left corner of the block $\mathbf{B}_1$, then for $h_U = \lceil u \rceil$ and $v_L = \lceil v \rceil$,

$$\mathrm{DCT}\{\hat{\mathbf{B}}_{ref}\} = \sum_{k=1}^{4} \mathrm{DCT}\{\mathbf{H}_k\}\mathrm{DCT}\{\mathbf{B}_k\}\mathrm{DCT}\{\mathbf{V}_k\}, \quad (6)$$

where

$$\mathbf{H}_1 = \mathbf{H}_3 = \mathbf{H}_U = \mathbf{G}_{BL}(h_U),\ \mathbf{H}_2 = \mathbf{H}_4 = \mathbf{H}_L = \mathbf{G}_{BR}(h_U),$$
$$\mathbf{V}_1 = \mathbf{V}_2 = \mathbf{V}_L = \mathbf{G}_{BL}^T(v_L),\ \mathbf{V}_3 = \mathbf{V}_4 = \mathbf{V}_R = \mathbf{G}_{BR}^T(v_L).$$

Here

$$[\mathbf{G}_{BL}(h_U) \quad \mathbf{G}_{BR}(h_U)] = \begin{bmatrix} 0 & \cdots & 0 & 0.5 & 0.5 & 0 & \cdots \\ \vdots & \ddots & \vdots & 0 & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \cdots & 0 & 0.5 & 0.5 \end{bmatrix} \quad (7)$$

Once again, $\mathbf{G}_{BL}(\cdot)$ and $\mathbf{G}_{BR}(\cdot)$ can be precomputed and stored in the memory as in the case of integer-pel motion compensation and thus the extra computational load for doing bilinear interpolation is eliminated.

6. CUBIC INTERPOLATED SUBPIXEL MOTION COMPENSATION

Three different interpolation functions, namely cubic, cubic spline and bilinear interpolations, are plotted in Fig. 4(a). As can be seen, the bilinear interpolation has the shortest filter length and the cubic spline has a longest ripple but the cubic spline has the smallest approximation error among these three [2]. To compromise between filter length and approximation accuracy, we choose the cubic interpolation in the simulation. By choosing the resolution of the filter as half a pixel length, the bilinear interpolation

$$f_{hb}(n) = [0.5, 1, 0.5]$$

and the cubic interpolation

$$f_{hc}(n) = [-0.0625, 0, 0.5625, 1.0000, 0.5625, 0, -0.0625].$$

From Fig. 4(b), it is clear that the contributions at the half-pel position from all the pixel values are summed up and give rise to the bilinear filter matrices $\mathbf{G}_{BL}(\cdot)$ and $\mathbf{G}_{BR}(\cdot)$. In a similar way, as in Fig. 4(c), the cubic filter matrices $\mathbf{G}_{CL}(\cdot)$ and $\mathbf{G}_{CR}(\cdot)$ can be defined as

$$\mathbf{G}_{CL}(i) = \begin{bmatrix} \mathbf{0} & -0.0625\mathbf{I}_{i+1} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & 0.5625\mathbf{I}_i \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & 0.5625\mathbf{I}_{i-1} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & -0.0625\mathbf{I}_{i-2} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad (8)$$

$$\mathbf{G}_{CR}(i) = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ -0.0625\mathbf{I}_{N-i-1} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ 0.5625\mathbf{I}_{N-i} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ 0.5625\mathbf{I}_{N-i+1} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ -0.0625\mathbf{I}_{N-i+2} & \mathbf{0} \end{bmatrix}. \quad (9)$$

Here $\mathbf{G}_{CL}(\cdot)$ and $\mathbf{G}_{CR}(\cdot)$ can be precomputed and stored. Therefore, its computational complexity remains the same as both integer-pel and half-pel bilinear interpolated DCT-based motion compensation methods. The

reconstructed DCT block and the corresponding motion-compensated residual can be obtained in a similar fashion:

$$\mathrm{DCT}\{\hat{\mathbf{B}}_{ref}\} = \sum_{k=1}^{4} \mathrm{DCT}\{\mathbf{H}_k\}\mathrm{DCT}\{\mathbf{B}_k\}\mathrm{DCT}\{\mathbf{V}_k\} \quad (10)$$

$$\mathrm{DCT}\{DFD\} = \mathrm{DCT}\{\hat{\mathbf{B}}_{ref}\} - \mathrm{DCT}\{\mathbf{C}\}, \quad (11)$$

where

$$\mathbf{H}_1 = \mathbf{H}_3 = \mathbf{H}_U = \mathbf{G}_{CL}(h_U),\ \mathbf{H}_2 = \mathbf{H}_4 = \mathbf{H}_L = \mathbf{G}_{CR}(h_U),$$
$$\mathbf{V}_1 = \mathbf{V}_2 = \mathbf{V}_L = \mathbf{G}^T_{CL}(v_L),\ \mathbf{V}_3 = \mathbf{V}_4 = \mathbf{V}_R = \mathbf{G}^T_{CR}(v_L).$$

This idea can be extended to other interpolation functions such as sharped Gaussian [10] and quarter-pel accuracy.

7. SIMULATION

Simulation is performed on the Infrared Car and Miss America sequences to demonstrate the effectiveness of our bilinear and cubic motion compensation methods and the performance of the fully DCT-based video coder structure. As shown in Fig. 5, the DCT-based motion compensation algorithms generate subpixel motion estimates and the subpixel DCT-based motion estimation algorithms produce the corresponding motion compensated residuals of the sequences "Infrared Car" and "Miss America". It can be seen that the cubic interpolation approach achieves lower MSE and BPS values than the bilinear interpolation.

8. CONCLUSION

Never before has it been attempted to implement a fully DCT-based video coder architecture which has a much simpler feedback loop composed of only one major component instead of three and thus achieve higher throughput and lower complexity than the conventional hybrid structure. The subpixel DCT-based motion compensation methods are presented to realize this fully DCT-based structure along with our previous efforts in devising DCT-based motion estimation. It is currently being undertaken to compare the performance of the fully DCT-based structure with that of the conventional video coder in terms of bits per samples of residuals and the overall compression rate after including the motion vectors. Detailed studies of the implementation complexity will also be considered.

REFERENCES

[1] S.-F. Chang and D. G. Messerschmitt, "Manipulation and compositing of MC-DCT compressed video", *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 1, pp. 1, January 1995.

[2] H. Hou and H. C. Andrews, "Cubic splines for image interpolation and digital filtering", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 6, pp. 508–517, 1978.

[3] A. K. Jain, *Fundamental of Digital Image Processing*, Prentice-Hall, 1989.

[4] U. V. Koc, *Low Complexity and High Throughput Fully DCT-Based Motion Compensated Video Coders*, PhD thesis, University of Maryland, College Park, July 1996.

[5] U.-V. Koc and K. J. R. Liu, "Discrete-Cosine/Sine-Transform based motion estimation", in *Proceedings of IEEE International Conference on Image Processing (ICIP)*, Austin, Texas, November 1994, vol. 3, pp. 771–775.

[6] U. V. Koc and K. J. R. Liu, "Adaptive overlapping approach for DCT-based motion estimation", in *Proceedings of IEEE International Conference on Image Processing*, Washington, DC, October 1995, vol. I, pp. 223–226.

[7] U. V. Koc and K. J. R. Liu, "DCT-based subpixel motion estimation", in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Altanta, GA, 1996, pp. IV–1931–1934.

[8] N. Merhav and V. Bhaskaran, "A fast algorithm for DCT-domain inverse motion compensation", in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, 1996, vol. IV, pp. 2309–2312.

[9] R. W. Schafer and L. R. Rabiner, "A digital signal processing approach to interpolation", *Proceedings of the IEEE*, pp. 692–702, June 1973.

[10] W. F. Schreiber, *Fundamentals of Electronic Imaging Systems – Some Aspects of Image Processing*, Springer-Verlag, 3rd edition, 1993.

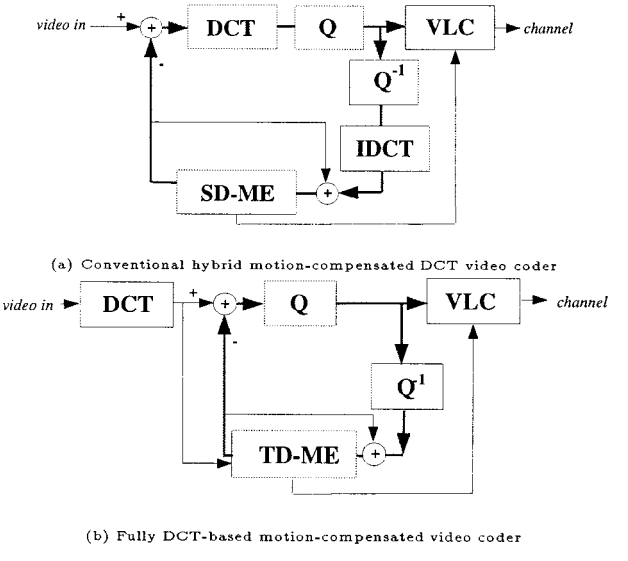

Figure 1: Different motion-compensated DCT video coder structures: (a) motion estimation/compensation are performed in the spatial domain; (b) motion estimation/compensation are completed in the transform (DCT) domain.

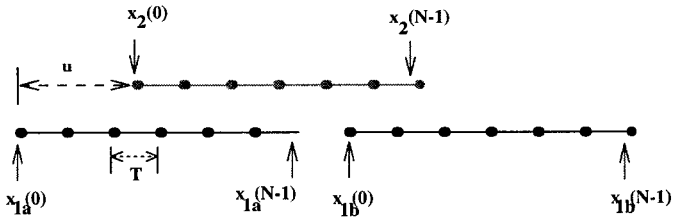

Figure 2: Illustration of extraction of the subpel displaced block $x_2(n)$ from two adjacent 1-D blocks $x_{1a}(n)$ and $x_{1b}(n)$ with bilinear interpolation.

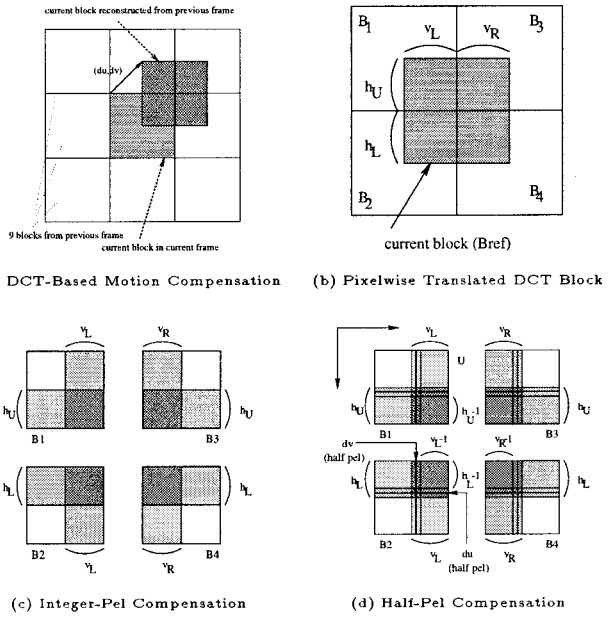

Figure 3: (a) Prediction of current block in current frame from four contiguous DCT blocks selected among nine neighboring blocks in previous frame based upon the estimated displacement vector for current block. (b) Schematic diagram of how a pixelwise translated DCT block is extracted from four contiguous DCT blocks. (c) Decomposition of integer-pel DCT-based translation as four matrix multiplication operations. (d) Decomposition of half-pel DCT-based translation as four matrix multiplication operations.

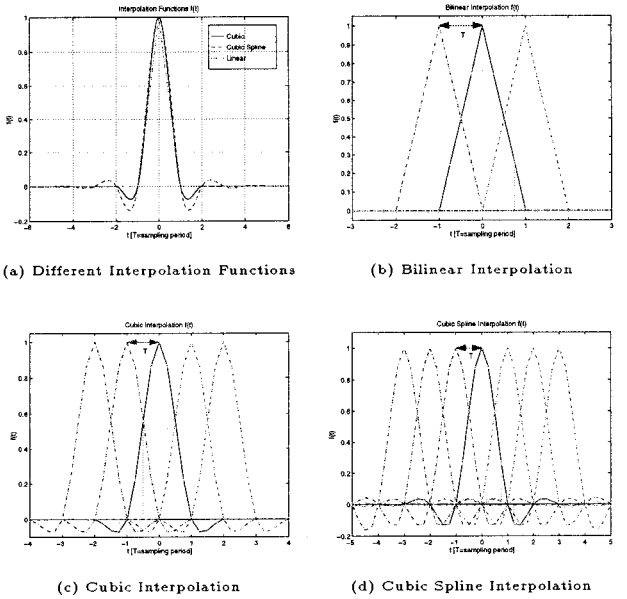

Figure 4: (a) plots different interpolation functions. (b), (c), and (d) depict how to form a pre or post multiplication matrix for half-pel or even quarter-pel DCT-based motion compensation.

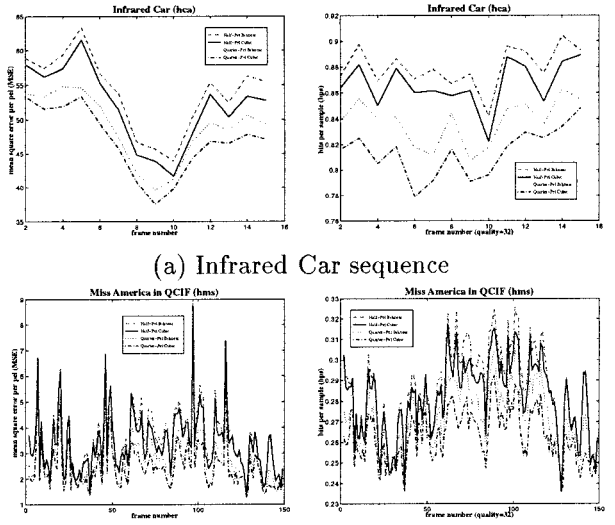

Figure 5: DCT-based subpixel motion compensation methods are used to generate motion compensated residues based on the motion estimates from the subpixel DCT-based motion estimation algorithms. The MSE-per-pixel values indicate how much redundancy in the video data has been removed by the motion estimation and compensation algorithms.