DCT Computation Based on Variable Complexity Fast Approximations

Krisda Lengwehasatit and Antonio Ortega

Integrated Media Systems Center
Department of Electrical Engineering-Systems
University of Southern California
Los Angeles, CA 90089-2654
Phone: (213) 740-2318, Fax: (213) 740-2318
E-mail: lengweha@sipi.usc.edu and ortega@sipi.usc.edu

Abstract

In this paper we investigate input dependent, variable complexity algorithms to compute the DCT. The basic goal of these algorithms is to not compute those DCT coefficients that will be quantized to zero. These algorithms exploit the fact that for compression applications (i) most of the energy is concentrated in a few DCT coefficients and (ii) as the quantization stepsize increases an increased number of coefficients is set to zero and reduced precision computation of the DCT may be tolerable. Thus we propose two classes of algorithms, the first one selectively prunes the DCT computation while the second uses an approximate computation, without floating point multiplications, that is matched to the quantization level selected.

1 Introduction

The Discrete Cosine Transform (DCT) [1] has been used in many image and video compression standards such as JPEG, MPEG1-2 and H.261/H.263 as signals in the DCT transform domain can usually be represented with fewer bits as the energy tends to be clustered in a few coefficients.

One of the major reasons for the continued popularity of the DCT is the availability of numerous fast algorithms. Whether these algorithms are exact or approximate a common feature in all of them is that they operate with a *fixed* number of operations independent of the input or quantization level. An example of a fast exact algorithm is shown in Figure 1 [2] for a size 8 1-D DCT. This algorithm requires only 13 multiplications and 29 additions. The theoretical bound on the number of nonrational multiplication for size 8 1-D DCT is 11 [3] which is achieved as in [4] and [5] via integer arithmetic and is adopted in many well-known implementation of DCT-based coding standards ([6], [7]).

For situations where even a fast fixed-complexity DCT algorithm is too complex (e.g. computation of large DCTs or complexity-constrained encoding situations) it is possible to resort to approximate computation of the DCT, at the cost of some degradation in the image quality. The following three techniques have been proposed in the literature.

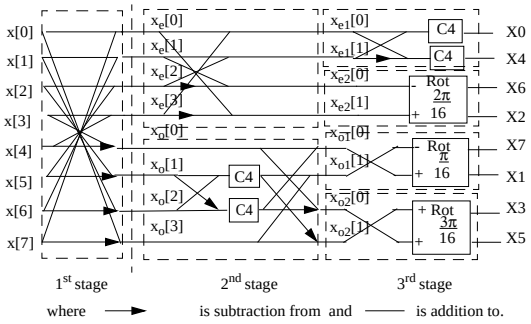


Figure 1: Vetterli and Ligtenberg's fast algorithm where 'Rot' represent rotation operation.

First, some DCT coefficients corresponding to high frequencies can be set to zero directly without being computed since they tend to have less perceptual significance and also have lower energy. Therefore one can prune the DCT operations in a manner similar to the well-known pruned DFT algorithms in [8].

The second method uses distributed arithmetic. DCT coefficients can be represented as a sum of the DCT of each input bitplane, which can be easily computed by a look-up table. The number of bitplanes, starting from the most significant bitplane, is determined by the complexity budget and can also be matched to the quantization level, since the number of bitplanes will determine the precision of the calculation. This method can be easily implemented in hardware but may involve too much overhead for a practical software implementation. In [9] a similar idea is called SNR update.

The third method exploits preprocessing to get rid of empirically unnecessary information before performing DCT on reduced size data. The gain of this method comes from relatively less complex preprocessing and reduced size DCT. In [10] and [11], the simple

Haar subband decomposition is used as the preprocessing. Then DCT coefficients are computed from only low band coefficients. The number of levels of decomposition and the number of bands used to compute DCT can be determined by the complexity budget.

All three techniques are approximate since it cannot be guaranteed that the exact DCT can be computed for all blocks. For example in the third approach one can only achieve an exact computation if all the inputs have all their energy in the lowest frequency band. However if it were easy to determine which blocks have all their energy in the low frequency then the approximate DCT could be used selectively without overall loss. Likewise, tailoring the degree of approximation to the coarseness of quantization would allow us to speed up the computation in cases where coarse quantization is used. This will lead us to variable complexity algorithms (VCAs) which are based on classification of the input data or the quantization coarseness.

In this paper, we propose both exact and approximate algorithms based on this VCA approach. In section 2, we show how to classify block inputs before performing various components of a DCT operation so as to achieve a VCA exact DCT algorithm. In section 3, lossy approximate DCT algorithms are proposed. These algorithms are multiplication-free and are such that the degree of approximation can be selected based on the quantization level. For simplicity, all expressions are shown as if 1-D DCT is used. The algorithms can also be applied to 2-D DCT. The conclusions are given in section 4. Our experiments show the potential for some gains in a software implementation, in particular for the approximate algorithms.

2 Variable Complexity Algorithm

2.1 Input classification

A variable complexity algorithm (VCA) as introduced above is an algorithm in which the complexity is input dependent. Our previous work in [12] provides an example of a VCA implementation of the inverse DCT. Given that we operate on quantized data, typical input blocks tend to be sparse. The IDCT VCA detects which input coefficients are zero, so that there are not used in the computation. Thus the key component of the VCA is **classification**. Given that detecting zeros also has a cost, in [12], we also optimize the classification such that the overall complexity including the classification cost itself is minimized. Since different types of blocks require different complexity, the goal of VCA is to minimize the average complexity, where the average is computed with respect to "typical" data.

Consider now the case of the forward DCT. Here our goal should be to avoid the computation of those coefficients that will be later quantized to zero. In addition, if using an approximate DCT algorithm, one would like to generate coefficients with increasingly good approximation as the quantization becomes finer. Thus we would like to incorporate adaptivity to both input and quantization. Figure 2 shows the number of operations required (adds and multiplies) after pruning the DCT computation to avoid calculating those coefficients that will be set to zero for a particular quantization level.

While Fig. 2 indicates the potential for substantial gains, the issue is now how to perform an efficient classification of the input blocks. An example of blockwise classification can be found in [13] where each block is tested to determine whether it is likely to be quantized to zero after DCT. Given the values of the input pixels $x(i,j)$ a proposed test determines whether the sum of absolute values exceeds a quantization-dependent threshold. This work has the limitation of assuming a single quantizer is used for all the DCT coefficients (thus it is better suited for for interframe coding scenarios). We now explore a more general algorithm which can deal with nonuniform quantization and allows a finer classification of the inputs.

The problem of determining whether the output DCTs will be quantized to zero can be viewed geometrically as that of determining whether the input vector is in the deadzone region (Figure 3) or not. Since the DCT is an orthonormal transform if the input is in the deadzone, so is the output. In Figure 3, the deadzone is the solid rectangle in (y_1, y_2) output coordinate where the DCTs are quantized to zero if they fall into this region. The input coordinate is (x_1, x_2). The test region equivalent to [13] is shown as a dashed square.

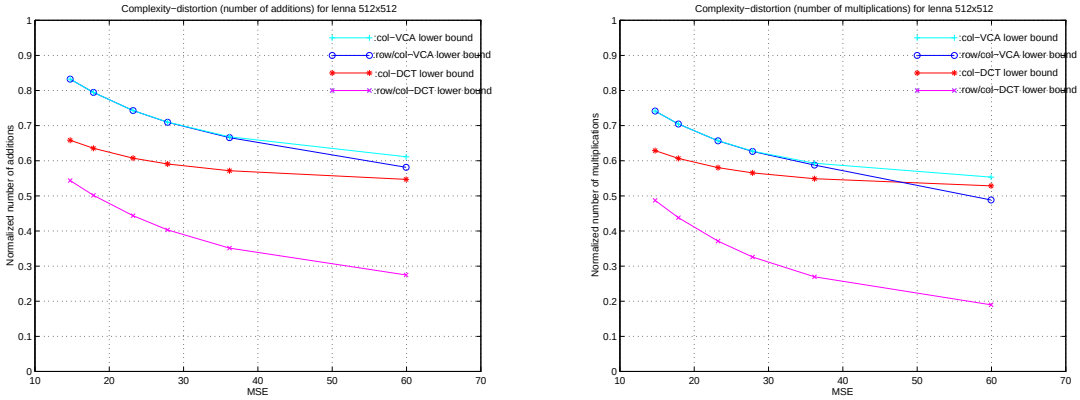
We present a test region based on the following test, **if input** $|x(i)| < q(i)/2$ **,** $\forall\ i$ **then set output** $\bar{X}$ **to a zero vector**. We search for $\bar{q} = [q(0)q(1)...q(N-1)]$ that satisfy the triangle inequality $|D|\bar{q} < |\bar{Q}/2|$ and has the maximal volume, where D is the DCT transform matrix. (Note that the ideal test would be one such that the deadzone corresponding to the test fit as closely as possible within the deadzone corresponding to the actual quantization). This test region is equivalent to another solid square in the deadzone in Figure 3. We use square i.e. $q(i) = q$ for all i, because the maximal square volume is almost the same as in rectangular case and for the sake of simplicity in the test process. In order to perform this test we will need to compute at most N absolute values, N comparisons and $N-1$ logical operations.

This test classifies the input into 2 classes to which we assign either full operation DCT or no operation DCT. Consider now the baseline fast algorithm in Figure 1. It can be seen that the computation is divided into three stages. From Figure 1, let

$$D = \begin{bmatrix} S_3^{e1} & 0 & 0 & 0 \\ 0 & S_3^{e2} & 0 & 0 \\ 0 & 0 & S_3^{o1} & 0 \\ 0 & 0 & 0 & S_3^{o2} \end{bmatrix} \begin{bmatrix} S_2^e & 0 \\ 0 & S_2^o \end{bmatrix} S_1$$

where S_1, S_2, S_3 correspond, respectively, to 8×8, 4×4 and 2×2 matrices. Therefore, we can write the output of each stage as follows.

$$S_1 \bar{x} = \begin{bmatrix} \bar{x_e} & 0 \\ 0 & \bar{x_o} \end{bmatrix} \qquad (1)$$



(a) (b)

Figure 2: Comparisons of original and pruned algorithms for different distortions (a) number of additions, (b) number of multiplications. The DCT lower bound corresponds to computing only the subset of coefficients that will be non-zero after quantization. The VCA lower bound corresponds to pruning subject to the classification mechanisms of Section 2.2, i.e. we can only prune subsets of coefficients which are tested jointly in the algorithm.

$$\begin{bmatrix} S_2^e & 0 \\ 0 & S_2^o \end{bmatrix} S_1 \bar{x} \quad = \quad \begin{bmatrix} \bar{x}_{e1} & 0 & 0 & 0 \\ 0 & \bar{x}_{e2} & 0 & 0 \\ 0 & 0 & \bar{x}_{o1} & 0 \\ 0 & 0 & 0 & \bar{x}_{o2} \end{bmatrix} \quad (2)$$

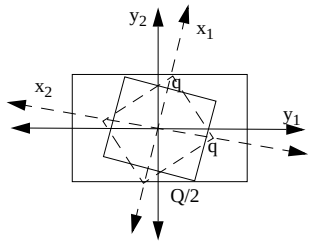


Figure 3: Geometric representation of deadzone after rotation.

From the above, we can apply the proposed test method at the beginning of each stage, i.e. testing $\bar{x}$ before S_1, $\bar{x}_e$ and $\bar{x}_o$ before S_2^e and S_2^o respectively, and $(\bar{x}_{e1}, \bar{x}_{e2}, \bar{x}_{o1}, \bar{x}_{o2})$ before $(S_3^{e1}, S_3^{e2}, S_3^{o1}, S_3^{o2})$ respectively. To clarify the test pattern, we test if $|\bar{x}| < \bar{q}$. If the test is not satisfied, we perform the first stage operation (S_1) to get $\bar{x}_e$ and $\bar{x}_o$. Then we test if $|\bar{x}_e| < \bar{q_e}$, the operation S_2^e is done on $\bar{x}_e$ and test if $|\bar{x}_o| < \bar{q_o}$, the operation S_2^o is done on $\bar{x}_o$ where $\bar{q}_e$ and $\bar{q}_o$ are obtained independently from triangle inequality condition on S_2^e and S_2^o, respectively. The test and operation proceed in a similar manner for the next stage.

Note that this classification is not restricted to detecting all-zero blocks as in [13] and can thus be used to determine whether subsets of the output coefficients will be zero. This method can also be extended to a separable 2-D DCT (row-column 1-D DCT) with the use of Kronecker (or tensor) product. Furthermore, "on the fly" classification for a nonseparable 2-D DCT is possible by simply postmultiplying (1) and (2) with S_1^t and $S_1^t \begin{bmatrix} S_2^e & 0 \\ 0 & S_2^o \end{bmatrix}^t$, respectively.

The result of the classification is shown in terms of number of operations as normalized by the fast algorithm in Figure 1. We encode the "lenna" image of size 512x512 using different quantization parameters to obtain the complexity at different quality for decoded images. The DCT computation is exact and the only distortion is that introduced by quantization. The results of pruning for each quantization level were shown in Figure 2 (a), (b), where separable row-column 1-D DCT was used. Classification costs were not included and thus these results can serve bounds. DCT bounds indicate the number of operations needed when only non-zero coefficients are computed, which VCA bounds correspond to pruning operations following the classification tree.

2.2 VCA with Optimal Classification

When the complexity of the tests is taken into account the total complexity can be higher than that of the original fixed complexity algorithm, as seen in Figure 4. As in [12], we optimize the classification such that only tests that provide reductions in average complexity are kept (i.e. the savings achieved when operations are skipped outweigh the overhead of testing for all inputs). This optimization is based on training on a set of images and is performed through exhaustive search (since the number of tests involved is small.)

We use "baboon", "boat", "creek" and "lake" as training data to design the VCA at each quantization parameter for "lenna" image. The result of both estimated complexity and CPU clock[1] savings are shown in Figure 4. In order to design the optimal test structure, we weight addition, multiplication, logical operation, absolute value, comparison, and 'if' as 1, 2, 1, 1, 1, 3, respectively. It can be seen that when the quantization parameter is small i.e. in small MSE region,

[1] The implementation is run on a Sun Ultra-1 running Solaris 2.5.

the complexity is the same as the original fixed complexity algorithm. This means that there is no test in the algorithm because in the optimization process it is determined that the tests would not result in any savings. On the other hand, in the high MSE region given that there will be more zero outputs there is something to be gained from the VCA approach.

From our results we can see that the gains are modest due in part to the overhead involved in testing. However a major reason for the lack of more significant gains is the fact that we are still computing an exact DCT, when in fact at high MSEs an approximate DCT would be acceptable given that data is going to be coarsely quantized.

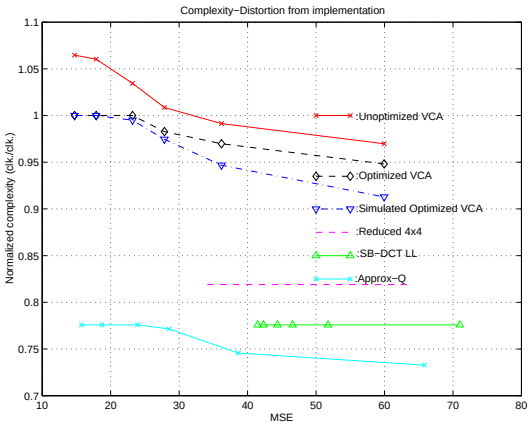


Figure 4: Complexity(clock cycle)-distortion comparison.

3 DCT Approximation

The DCT can be approximated using a subband decomposition as in [11] and [10]. This approach exploits the fact that, with appropriate post-processing, the DCT coefficients can be obtained after the subband decomposition, and in typical natural images one can disregard high frequency subband components without greatly affecting the accuracy of the calculated DCT. Therefore, the DCT coefficients can be approximated from post-processing only low frequency subband coefficients. Because a simple subband decomposition can be used (Haar filters for example) the overhead for pre-processing is small. Subband decomposition hence can be viewed as a pre-processing that reduces the interdependencies among the inputs and gives some clues of which information can be disregarded.

Instead we select to directly approximate the DCT operation with parameters that can be easily computed, by for example replacing multiplications with binary shifts and additions, so that floating point operation can be avoided. Our proposed approximate DCT is shown in figure 5. It can be verified that as p_o ranging from identity matrix, $p_{o1} = \begin{bmatrix} q_1 & 0 \\ 0 & I \end{bmatrix}$, to

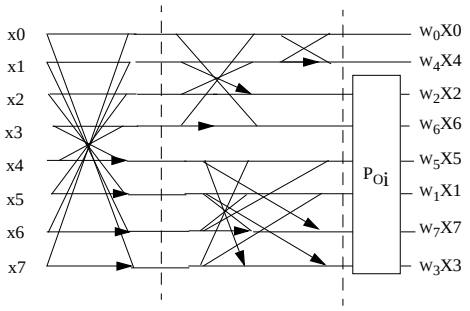


Figure 5: The approximate DCT algorithm.

more complex matrix (p_{o2},p_{o3},p_{o4},and p_{o5}[2]) the corresponding post-processing required for exact DCT, which is the DCT of the inverse of pre-processing matrix ($D \cdot D^{-1}_{approx}$), gets closer to the identity matrix (up to scaling factors). The scaling factors at the output which is $\mathtt{diag}(D \cdot D^{-1}_{approx})$ can be absorbed in quantization. The number of operations required for these approximate DCTs (referred to as #1,2,3,4 and 5) are 24 ADDs+2 SHIFTs, 33 ADDs+7 SHIFTs, 38 ADDs+8 SHIFTs, 38 ADDs+12 SHIFTs and 42 ADDs+12 SHIFTs, respectively as compared to 13 MULs+ 29 ADDs in [2]. Since this algorithm is lossy, the approximation introduces more distortion to reconstructed images. The rate-distortion curves

[2]

$$p_{o2} = \begin{bmatrix} q_1 & \bar{0}^t & \bar{0}^t & \bar{0}^t & \bar{0}^t \\ \bar{0} & 1 & 0 & 1/8 & -1/8 \\ \bar{0} & -1/8 & 1 & 1/8 & 0 \\ \bar{0} & 0 & -1/8 & 1 & 1/8 \\ \bar{0} & 1/8 & 1/8 & 0 & 1 \end{bmatrix}$$

$$p_{o3} = \begin{bmatrix} q_2 & \bar{0}^t & \bar{0}^t & \bar{0}^t & \bar{0}^t \\ \bar{0} & 1 & 1/8 & 1/8 & -1/8 \\ \bar{0} & -1/8 & 1 & 1/8 & -1/8 \\ \bar{0} & -1/8 & -1/8 & 1 & 1/8 \\ \bar{0} & 1/8 & 1/8 & -1/8 & 1 \end{bmatrix}$$

$$p_{o4} = \begin{bmatrix} q_3 & \bar{0}^t & \bar{0}^t & \bar{0}^t & \bar{0}^t \\ \bar{0} & 1 & 1/8 & 1/8 & -1/4 \\ \bar{0} & -1/8 & 1 & 1/4 & -1/8 \\ \bar{0} & -1/8 & -1/4 & 1 & 1/8 \\ \bar{0} & 1/4 & 1/8 & -1/8 & 1 \end{bmatrix}$$

$$p_{o5} = \begin{bmatrix} q_3 & \bar{0}^t & \bar{0}^t & \bar{0}^t & \bar{0}^t \\ \bar{0} & 1 & 1/8 & 1/8 & -3/16 \\ \bar{0} & -1/8 & 1 & 3/16 & -1/8 \\ \bar{0} & -1/8 & -3/16 & 1 & 1/8 \\ \bar{0} & 3/16 & 1/8 & -1/8 & 1 \end{bmatrix}$$

where

$$q_1 = \begin{bmatrix} 1 & -1/2 \\ 1/2 & 1 \end{bmatrix}$$

$$q_2 = \begin{bmatrix} 1 & -3/8 \\ 1/2 & 1 \end{bmatrix}$$

$$q_3 = \begin{bmatrix} 1 & -3/8 \\ 3/8 & 1 \end{bmatrix}$$

$$\bar{0} = [\ 0 \quad 0\ ]$$

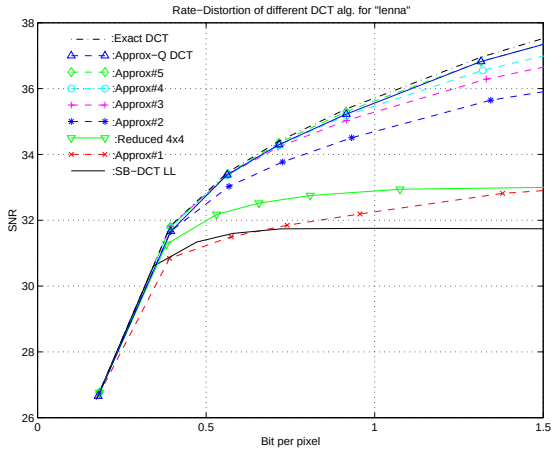


Figure 6: Rate-Distortion curve of 512x512 lenna image JPEG coding using various DCT algorithms.

of these approximate DCTs using JPEG coding are shown in Figure 6.

In this experiment, we encode "lenna" image with JPEG compression using the example quantization matrix [14]. It can be seen that, as expected, in high distortion range the increase in distortion introduced by the approximate DCTs is less than that in low distortion range. This can be explained by the fact that the distortion introduced by quantization is larger than the distortion from the DCT approximation and thus masks it out. Therefore, we develop an algorithm which is quantization parameter dependent (shown as a solid line, Approx-Q, in Figure 6) in which the selection of the approximate algorithm is made at the beginning of the encoding process depending on the quantization parameter. It uses a coarser DCT approximation algorithm for low quality coding and finer DCT approximation for high quality coding for small degradation. The degradation of the decoded image introduced by the approximate DCT is 0.12 dB at 0.18 bpp, 0.15 dB at 0.91 bpp and 0.64 dB at 3.17 bpp, respectively. From Figure 6, it is obvious that even with the Approx#2, the R-D curve is better than other reduced complexity approximate algorithms such as pruned DCT (computing only low frequency 4x4 DCT) and subband DCT (using low-low subband [11]). This is because we do not lose high frequency information which is present at high rates.

It is worthwhile pointing out here that since the structure of the approximate DCT algorithm is similar to the fast algorithm in Figure 1 we can apply the concept of classification and the optimization technique to obtain the optimal VCA as in section 2 to the Approx-Q DCT algorithm presented in section 3. The algorithm is now become data-dependent and will be called Approx-VCA DCT. However, since the cost of multiplication is already eliminated from the Approx-Q DCT while the cost involving with tests is still the same, the further complexity reduction is predicted to be not as significant as the optimal VCA in section 2 compared to the baseline algorithm.

4 Summary and Conclusions

We present a novel lossless VCA and a lossy Approx-VCA DCT algorithm. The former computes exact DCT with data-dependent complexity. The latter is a multiplication-free algorithm sacrificing the accuracy of the DCT coefficients giving approximated DCT coefficients but yields small number of operations. In a software implementation only the approximate algorithm shows some significant gain (e.g. 22-27% reductions in compute time) depending on the quantization level. Also the experiments show that even with the approximate DCT, the R-D performance is only slightly degraded depending on the target rate.

References

[1] K. Rao and P. Yip, *Discrete Cosine Transform*. Academic Press, 1990.

[2] A. Ligtenberg and M. Vetterli, "A discrete Fourier/cosine transform chip," *IEEE J. on Sel. Areas in Comm.*, vol. SAC-4, pp. 49–61, Jan 1986.

[3] P. Duhamel and H. H'Mida, "New 2n DCT algorithms suitable for VLSI implementation," in *Proc. of ICASSP'87*, (Dallas), p. 1805, Apr 1987.

[4] C. Loeffler, A. Ligtenberg, and G. Moschytz, "Practical fast 1-D DCT algorithms with 11 multiplications," in *In Proc. of ICASSP'89*, pp. 988–991, 1989.

[5] Z. Wang, "Fast algorithms for the discrete W-transform and for the Discrete Fourier Transform," *IEEE Trans. on Acoustics, Speech and Signal Proc.*, vol. ASSP-32, Aug 1984.

[6] "The independent JPEG's group software JPEG, version 6." `ftp://ftp.uu.net`.

[7] "MPEG video software decoder, v2.2." `http://bmrc.berkeley.edu/projects/mpeg/`.

[8] H. Sorensen and C. Burrus, "Efficient computation of the DFT with only a subset of input or output points," *IEEE Trans. on Signal Proc.*, 1993.

[9] J. Winograd and S. Nawab, "Incremental refinement of DFT and STFT approximations," *IEEE Signal Proc. Letters*, 1995.

[10] A. Hossen and U. Heute, "Fast approximation DCT: Basic-idea, error analysis, application," in *Proc. of ICASSP'97*, pp. 2005–2008, 1997.

[11] S. Jung, S. Mitra, and D. Mukherjee, "Subband DCT: Definition, analysis, and applications," *IEEE Trans. on Circ. and Sys. for Video Tech.*, 1996.

[12] K. Lengwehasatit and A. Ortega, "DCT computation with minimal average number of operations," in *Proc. of VCIP'97*, (San Jose, CA), Feb. 1997.

[13] X. Bo and Z. Xulong, "A new algorithm for the implementation of h.263 video coding," *IEEE Trans. on Circ. and Sys. for Video Tech.*, 1997. Submitted.

[14] W. Pennebaker and J. Mitchell, *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, 1994.