Integer discrete cosine transform and its fast algorithm

LiZhi Cheng, Hui Xu and Yong Luo

A recursive sparse matrix decomposition for a floating-point discrete cosine transform (DCT) matrix is presented. Based on this matrix decomposition approach, a split-radix DCT algorithm is proposed and a new integer DCT (IntDCT) algorithm that requires only lifting steps and additions is developed.

Introduction: The discrete cosine transform (DCT) of the sequence $x(n)$ is defined as

$$X(k) = \sqrt{\frac{2}{N}}a(k)\sum_{n=0}^{N-1} x(n)\cos\frac{(2n+1)k\pi}{2N}$$

$$\text{or in matrix form } X = C_N x \tag{1}$$

where $a(0) = \sqrt{2}/2$, $a(k) = 1$, $1 \le k \le N-1$, $X = (X(0), X(1), ..., X(N-1))^T$, $x = (x(0), x(1), ..., x(N-1))^T$ and

$$C_N = \sqrt{\frac{2}{N}}\left[a(k)\cos\frac{(2n+1)k\pi}{2N}\right]_{0\le k,n\le N-1}$$

The DCT has been widely used in signal processing. However, floating-point multiplications are inevitable when implementing the DCT since it usually maps integers into floating-point numbers. This property prevents the DCT from being widely used in areas such as mobile device and lossless compression [1 – 3]. The integer DCT (IntDCT), which maps integers into integers, has thus been widely investigated. The IntDCT has become a powerful tool for lossless compression and mobile devices [3, 4] because the IntDCT can be used to express information losslessly [3, 4] and its implementation is greatly simplified compared to the floating-point DCT. However, many problems associated with the IntDCT need to be solved. For instance, until now only 8-point and 16-point fast IntDCT algorithms based on the Walsh-Hadamard transform and lifting scheme have been proposed (see [4, 5]). The purpose of this Letter is to develop a fast IntDCT algorithm for a transform length N, where N is a power of 2.

Outline of algorithm: To derive the IntDCT algorithm, we first develop a floating-point DCT algorithm based on the following recursive matrix decomposition approach.

Proposition (i): Let $N = 2^t$ and Bdiag() indicate the block diagonal matrix, then C_N given by eqn. 1 can be decomposed into the form

$$\begin{aligned} C_N &= P_N^{(1)}B_N P_N^{(2)}\mathrm{Bdiag}\left\{C_{\frac{N}{2}}, C_{\frac{N}{4}}, C_{\frac{N}{4}}\right\} \\ &\times P_N^{(3)}\mathrm{Bdiag}\left\{I_{\frac{N}{2}}, T_{\frac{N}{2}}\right\}G_N \end{aligned} \tag{2}$$

where $I_{N/2}$ and in the following J_M denote the identity and opposite identity matrices, respectively. The permutation-like matrices $P_N^{(m)} = (p_{i,j}^{(m)})_{0\le i,j\le N-1}$ $(m = 1, 2, 3)$ satisfy

$$p_{i,j}^{(1)} = \begin{cases} 1 & i = 2j \quad 0 \le j \le \frac{N}{2}-1 \\ & \text{or } i = 2N-2j-1 \quad \frac{N}{2} \le j \le N-1 \\ 0 & \text{otherwise} \end{cases}$$

$$P_N^{(2)} = \mathrm{Bdiag}\left\{I_{\frac{N}{2}}, P_{\frac{N}{2}}^{(1)}\mathrm{Bdiag}(J_{\frac{N}{4}}, I_{\frac{N}{4}})\right\}$$

$$P_N^{(3)} = \mathrm{Bdiag}\left\{I_{\frac{3N}{4}}, \mathrm{diag}(1,-1,...,-1,1)\cdot P_{\frac{N}{4}}^{(1)}\right\}$$

Moreover,

$$B_N = \mathrm{Bdiag}\left\{1, \frac{\sqrt{2}}{2}\begin{bmatrix} 1 & 1 \\ -1 & 1\end{bmatrix}, \cdots, \frac{\sqrt{2}}{2}\begin{bmatrix} 1 & 1 \\ -1 & 1\end{bmatrix}, 1\right\}$$

$$G_N = \frac{\sqrt{2}}{2}\begin{bmatrix} I_{\frac{N}{2}} & J_{\frac{N}{2}} \\ -I_{\frac{N}{2}} & J_{\frac{N}{2}}\end{bmatrix}$$

and the ij-elements of the matrix $T_{N/2} = (t_{i,j})_{0\le i,j\le N/2-1}$ have the form

$$t_{i,j} = \begin{cases} \cos\frac{2i+1}{2N}\pi \text{ or} & i = j \quad 0 \le i \le \frac{N}{4}-1 \\ \cos\frac{N-2i-1}{2N}\pi & \qquad\quad \text{or } \frac{N}{4} \le i \le \frac{N}{2}-1 \\ \sin\frac{2i+1}{2N}\pi \text{ or} & i = \frac{N}{2}-j-1 \quad 0 \le i \le \frac{N}{4}-1 \\ -\sin\frac{N-2i-1}{2N}\pi & \qquad\quad \text{or } \frac{N}{4} \le i \le \frac{N}{2}-1 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

We can see that the matrix $T_{N/2}$ contains $N/4$ rotations

$$\Psi\left(\frac{2i+1}{2N}\pi\right) = \begin{bmatrix} t_{ii} & t_{i,\frac{N}{2}-i-1} \\ t_{\frac{N}{2}-i-1,i} & t_{\frac{N}{2}-i-1,\frac{N}{2}-i-1}\end{bmatrix}$$

Proposition (i) can be directly proved by using trigonometric identities and then implementing the matrix-vector product. To save space the proof is omitted here. We now show that Proposition (i) can be used to compute the DCT efficiently. In fact, from eqn. 2 we see that an N-point DCT is decomposed into an $N/2$ point DCT and two $N/4$ point DCTs. By repeating the above decomposition process recursively, a split-radix DCT algorithm is developed. Taking a similar derivation to that used in [5], we find that the DCT algorithm requires $(N/2)\log_2 N$ multiplications and $(3N/2)\log_2 N - N + 1$ additions. Obviously, the complexity of the proposed DCT algorithm is one of the lowest among the complexities of reported DCT algorithms. However, existing DCT algorithms usually use an unscaled DCT transforming matrix

$$\sqrt{\frac{2}{N}}\left[\cos\frac{(2n+1)k\pi}{2N}\right]_{0\le k,n\le N-1}$$
$$\text{or} \quad \left[\cos\frac{(2n+1)k\pi}{2N}\right]_{0\le k,n\le N-1}$$

(no longer a unitary matrix [5]), therefore the IntDCT with a lifting scheme cannot be directly derived from existing DCT algorithms [4, 6]. Since the transforming matrix C_N in the proposed DCT algorithm is unitary, therefore, using the lifting scheme (see [4, 6]) we can derive a fast IntDCT algorithm.

Let $\bar{C}_N$ be the IntDCT matrix that we wish to establish. Using the concept of the IntDCT transform, $\bar{C}_N$ should map integers into integers, and is also an approximation of the DCT matrix C_N. In the following we use Proposition (i) to construct the IntDCT.

The shifting scheme developed in [4, 6] is based on rotation

$$\Psi(\zeta) = \begin{bmatrix} \cos\zeta & \sin\zeta \\ -\sin\zeta & \cos\zeta\end{bmatrix}$$

(rotation angle: ζ). More precisely, the mapping of integers to integers is achieved by decomposing the matrix into the product of a number of rotation matrices $\Psi(\zeta)$ and then using the integer transform $\bar{\Psi}(\zeta)$ to approximate the rotation matrix $\Psi(\zeta)$ (see [4, 6]). Hence the matrix $T_{N/2}$ that contains $N/4$ rotations $\Psi\{[(2i+1)/2N]\pi\}$, $i = 0, 1, ..., N/4 - 1$ can also be easily approximated by the integer transform $\bar{T}_{N/2}$ according to the lifting scheme [4, 6]. Furthermore, B_N and G_N contain $N/2 - 1$ and $N/2$ rotations $\Psi(\pi/4)$ and so the integer transform approximation to B_N and G_N, $\bar{B}_N$ and $\bar{G}_N$, can be easily obtained. Based on the above discussion, the transforming matrix $\bar{C}_N$ for 2^t point IntDCT is defined recursively by $\bar{C}_1 = (1)$, $\bar{C}_2 = \bar{\Psi}(\pi/4)$, and

$$\begin{aligned} \bar{C}_{2^t} &= P_{2^t}^{(1)}\bar{B}_{2^t}P_{2^t}^{(2)}\mathrm{Bdiag}\left(\bar{C}_{2^{t-1}}, \bar{C}_{2^{t-2}}, \bar{C}_{2^{t-2}}\right) \\ &\times P_{2^t}^{(3)}\mathrm{Bdiag}\left(I_{2^{t-1}}, \bar{T}_{2^{t-1}}\right)\bar{G}_N \end{aligned} \tag{4}$$

From the above definition, we obtain a fast algorithm for the IntDCT as follows:

Algorithm (i): Computation of the IntDCT

Step (i) Compute $h = \bar{G}_N x$;

Step (ii) Compute $\tilde{h} = P_N^{(1)}\bar{T}_{N/2}h$, where $\tilde{h} = (\tilde{h}(0), \tilde{h}(1), ..., \tilde{h}(N/2-1))^T$, $h = (h(0), h(1), ..., h(N/2-1))^T$;

Step (iii) Compute a length $N/2$ IntDCT of the sequence $h(n)$, $0 \le n \le N/2 - 1$ and two IntDCTs with length $N/4$ of the sequences $h(n)$, $n = N/2, ..., 3N/4 - 1$ and $h(n)$, $n = 3N/4, N - 1$ and let the outputs be $H(k)$, $H_1(k)$ and $H_2(k)$, respectively. Steps (i), (ii) and (iv) can be used recursively for the computations;

Step (iv) Reorder according to $\tilde{X} = P_N^{(2)}[H(0), ..., H(N/2-1), H_1(0), ..., H_1(N/4-1), H_2(0), ..., H_2(N/4-1)]^T$, and then compute $X = P_N^{(3)}\bar{B}_N\tilde{X}$.

Computational complexity analysis of IntDCT algorithm: Note that matrix-vector products $P_N^{(m)}y$ $(m = 1, 2, 3)$ reorder the vector y and involve no arithmetic operations. If rotations $\Psi(\pi/4)$ and $\Psi\{[(2i+1)/2N]\pi\}$ for $1 \le i \le N/2 - 1$ are all implemented according to the lifting scheme developed in [4, 6], then Steps (i), (ii) and (iv) need $(9N - 6)/4$ lifting steps. Let $LC(N)$ and $AC(N)$ represent the number of lifting steps and additions for computing an IntDCT with length N, from Algorithm (i) we obtain $LC(N) = (3N/2)\log_2 N - 3N + 3$, $AC(N) = 2(\log_2 N - 1)N + 2$. When $N = 8$ or 16, if each lifting step is achieved using integer arithmetic as in [4, 6], then the complexity for the proposed IntDCT algorithm will be about the same as that required in [4]. However, in [4] the transforming length N must be 8 or 16 and so the IntDCT algorithm in [4] cannot be generalised to the computation of large scale of IntDCT.

Conclusion: We have developed a fast IntDCT algorithm for the transform length N where N is any power of 2, which overcomes the drawback that existing fast IntDCT algorithms can only be used to compute 8 point and 16 point IntDCTs.

© IEE 2001 *19 October 2000*
Electronics Letters Online No: 20010060
DOI: 10.1049/el:20010060

LiZhi Cheng and Yong Luo (*Department of Mathematics & System Science, National University of Defence Technology, Changsha, 410073, Hunan, People's Republic of China*)

E-mail: lzchen@nudt.edu.cn

Hui Xu (*Electrical Engineering School, National University of Defence Technology, Changsha, 410073, Hunan, People's Republic of China*)

References

1 APOUSLEMAN, G.P., MARCELLIN, M.W., and HUNT, B.H.: 'Compression of hyperspectral imagery using the 3-D DCT and hybrid DPCM/DCT', *IEEE Trans. Geosci. Remote Sens.*, 1995, **33**, pp. 26–34

2 BRITANAK, V., and RAO, K.R.: 'The fast generalized discrete Fourier transforms: A unified approach to the discrete sinusoidal transforms computation', *Signal Process.*, 1999, **79**, pp. 135–150

3 PHILPS, W.: 'Lossless DCT for combined lossy/lossless image coding'. IEEE Int. Conf. Image Process., 1998, Vol. 3, pp. 871–875

4 CHEN, Y.J.: 'Integer discrete cosine transform (IntDCT)'. 2nd Int. Conf. Inform. Commun. Signal Process., Singapore, Dec. 1999, Invited paper

5 RAO K.R., and YIP P.: 'Discrete cosine transform: algorithm, advantages and applications' (Academic Press, New York, 1990)

6 SWELDLENS, W.: 'The lifting scheme: a construction of second generation wavelets', *SIAM J. Math. Anal.*, 1998, **29**, (2), pp. 511–546