A Fast Cosine Transform in One and Two Dimensions

JOHN MAKHOUL, SENIOR MEMBER, IEEE

Abstract—The discrete cosine transform (DCT) of an N-point real signal is derived by taking the discrete Fourier transform (DFT) of a $2N$-point even extension of the signal. It is shown that the same result may be obtained using only an N-point DFT of a reordered version of the original signal, with a resulting saving of 1/2. If the fast Fourier transform (FFT) is used to compute the DFT, the result is a fast cosine transform (FCT) that can be computed using on the order of $N \log_2 N$ real multiplications. The method is then extended to two dimensions, with a saving of 1/4 over the traditional method that uses the DFT.

I. Introduction

The discrete cosine transform (DCT) has had a number of applications in image processing (see [1]) and, more recently, in speech processing [2], [3]. Compared to other orthogonal transforms, its performance seems to compare most favorably with the optimal Karhunen-Loève transform of a large number of signal classes [2], [4]. It has been shown that the DCT of an N-point signal can be computed using a $2N$-point discrete Fourier transform (DFT) [4]. Chen *et al.* [1] used matrix factorization to derive a special algorithm to compute the DCT of a signal with N a power 2, resulting in a saving of 1/2 over the previous method when the latter uses the fast Fourier transform (FFT).[1] More recently, Narasimha and Peterson [7] developed a method that employs an N-point DFT of a reordered version of the signal (where N is assumed to be even), resulting in a similar saving of 1/2. When N is a power of 2, use of the FFT results in a saving comparable to that of Chen *et al.* [1]. However, in the algorithm of Narasimha and Peterson, one can use existing software to compute the FFT instead of implementing a special algorithm for the DCT.

The algorithm presented here for the 1-D case is essentially identical to that of Narasimha and Peterson [7].[2] Our algorithm is more general in that N may be odd or even. This generalization and the extension to the 2-D case are facilitated by the view taken that the DCT can be regarded essentially as the DFT of an even extension of the signal. The generalization to the m-D case should then be straightforward, with a resulting saving of $1/2^m$ over the traditional method that employs the DFT. In [7] the authors mention that the inverse DCT can be obtained using a number of computations equal to that of the forward DCT. Here, we show how this can be done. Procedures for the forward and inverse fast cosine transforms are presented for easy implementation on the computer.

Finally, an appendix is included that presents a method and associated flowgraphs for efficient computation of the DFT of a real sequence and the IDFT of a Hermitian symmetric sequence. The flowgraphs are believed to be novel.

II. Discrete Cosine Transform

To motivate the derivation of the DCT presented below, we shall first review some basic discrete-time Fourier theory.

Let $x(n)$ be a discrete-time signal and $X(\omega)$ its Fourier transform. In one definition the cosine transform of $x(n)$ is the real part of $X(\omega)$. The real part of $X(\omega)$ is also equal to the Fourier transform of the even part of $x(n)$, defined by $x_e(n) = [x(n) + x(-n)]/2$ (see [8], for example). Therefore, the cosine transform of $x(n)$ is equal to the Fourier transform of $x_e(n)$. Now, if $x(n)$ is causal, i.e., $x(n) = 0$ for $n < 0$, then $x_e(n)$ and, therefore, the cosine transform uniquely specifies $x(n)$. In that case, $x_e(n)$ can be thought of as an even extension of $x(n)$. Therefore, the cosine transform of a causal $x(n)$ can be obtained as the Fourier transform of an even extension of $x(n)$. This viewpoint forms the basis of the derivation of the DCT below.

As an example, Fig. 1(a) shows a causal signal $x(n)$, and Fig. 1(b) shows an even extension of $x(n)$, $y_1(n) = x(n) + x(-n)$. [$y_1(n)$ is equal to twice the even part of $x(n)$.] Another possible even extension of $x(n)$ is shown in Fig. 1(c), where $y_2(n) = x(n) + x(-n-1)$. $y_1(n)$ is even about the point $n = 0$ while $y_2(n)$ is even about $n = -0.5$. The Fourier transform of $y_1(n)$ is real, while the Fourier transform of $y_2(n)$ contains a linear-phase term corresponding to the half-sample offset. Cosine transforms based on $y_1(n)$ or $y_2(n)$ can be defined and from which $x(n)$ can be determined uniquely.

In the example above we assumed that the Fourier transform is computed at all frequencies. In practice, one usually computes the *discrete* Fourier transform (DFT) at a finite number of equally spaced frequencies. For this case, the signal can be recovered in its aliased periodic form from the DFT [8]. In attempting to form even extensions of a signal where the extended signals are constrained to be periodic, one has addi-

Manuscript received November 27, 1978; revised April 25, 1979 and August 28, 1979. This work was supported by the Advanced Research Projects Agency and monitored by RADC/ETC under Contract F19628-78-C-0136.

The author is with Bolt Beranek and Newman, Inc., Cambridge, MA 02138.

[1] Chen *et al.* [1] claim a larger saving. However, if in the conventional method one takes advantage of the fact that the signal is real, then the saving amounts to only 1/2.

[2] [7] was unknown to the author when this paper was first submitted for publication. The author thanks R. Crochiere and the reviewers for bringing [7] to his attention. The parts of this paper that overlap [7] have been retained to enhance the tutorial aspect of this paper.

0096-3518/80/0200-0027$00.75 © 1980 IEEE

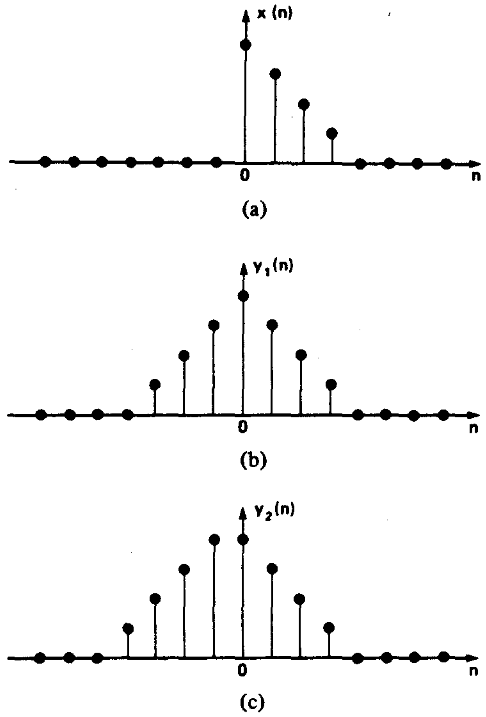

Fig. 1. (a) Causal signal $x(n)$. (b) An even extension of $x(n)$, $y_1(n)$, about $n = 0$. (c) Another even extension of $x(n)$, $y_2(n)$, about $n = -0.5$.

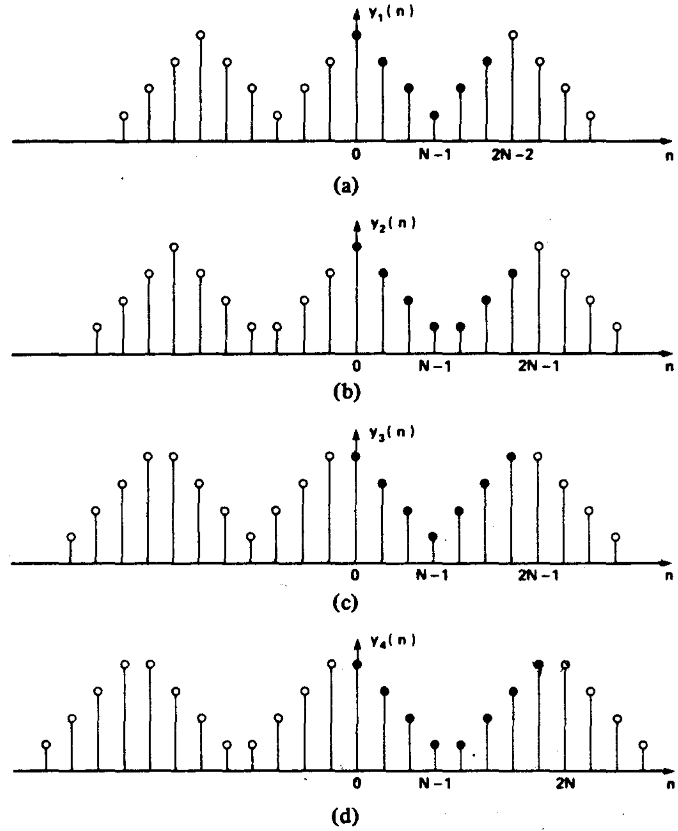

Fig. 2. Four different periodic even extensions of the nonzero $x(n)$ samples in Fig. 1(a). (a) $y_1(n)$ is a $(2N - 2)$-point even extension. (b) and (c) $y_2(n)$ and $y_3(n)$ are two different $(2N - 1)$-point even extensions. (d) $y_4(n)$ is a $2N$-point even extension. The DCT discussed in this paper is derived from the $2N$-point even extension $y_4(n)$.

tional choices in defining those extensions. As an example, let $x(n)$, $0 \leqslant n \leqslant N - 1$, be the sequence given by the four nonzero samples in Fig. 1(a) (i.e., $N = 4$). Fig. 2 shows four different even extensions of $x(n)$. $y_1(n)$ is a $(2N - 2)$-point even extension; $y_2(n)$ and $y_3(n)$ are two different $(2N - 1)$-point even extensions; and $y_4(n)$ is a $2N$-point even extension. Each of the four extension definitions could form the basis for a DCT definition. The most common form of the DCT is the one derived from the $2N$-point even extension $y_4(n)$ and is the one discussed in this paper.

A. Forward DCT

We desire the DCT of an N-point real data sequence $x(n)$, $0 \leqslant n \leqslant N - 1$. The DCT is derived below from the DFT of a $2N$-point even extension of $x(n)$.

Let $y(n)$ be a $2N$-point even extension of $x(n)$ defined by:

$$y(n) = \begin{cases} x(n), & 0 \leqslant n \leqslant N - 1 \\ x(2N - n - 1), & N \leqslant n \leqslant 2N - 1. \end{cases} \tag{1}$$

Then

$$y(2N - n - 1) = y(n). \tag{2}$$

Fig. 3(a) shows an example of a signal $x(n)$, and Fig. 3(b) shows the corresponding even extension of $x(n)$ as defined in (1). Because of the minus 1 on the left-hand side of (2), $y(n)$ is not even about N and, therefore, will not have a real DFT, as we shall see below.

The DFT of $y(n)$ is given by

$$Y(k) = \sum_{n=0}^{2N-1} y(n)\, W_{2N}^{nk} \tag{3}$$

where

$$W_M = e^{-j2\pi/M}. \tag{4}$$

Substituting (1) in (3), we have:

$$Y(k) = \sum_{n=0}^{N-1} x(n)\, W_{2N}^{nk} + \sum_{n=N}^{2N-1} x(2N - n - 1)\, W_{2N}^{nk}.$$

By changing the summation variable in the right-hand term, noting that $W_{2N}^{2mN} = 1$ for integer m, and factoring out $W_{2N}^{-k/2}$, we have

$$Y(k) = W_{2N}^{-k/2} \sum_{n=0}^{N-1} x(n)\, [W_{2N}^{nk} W_{2N}^{k/2} + W_{2N}^{-nk} W_{2N}^{-k/2}]. \tag{5}$$

The expression in (5) may be written in two ways:

$$Y(k) = W_{2N}^{-k/2}\, 2 \sum_{n=0}^{N-1} x(n) \cos \frac{\pi(2n + 1)k}{2N}, \quad 0 \leqslant k \leqslant 2N - 1 \tag{6}$$

or

$$Y(k) = W_{2N}^{-k/2}\, 2\, \mathrm{Re}\left[W_{2N}^{k/2} \sum_{n=0}^{N-1} x(n)\, W_{2N}^{nk}\right], \quad 0 \leqslant k \leqslant 2N - 1. \tag{7}$$

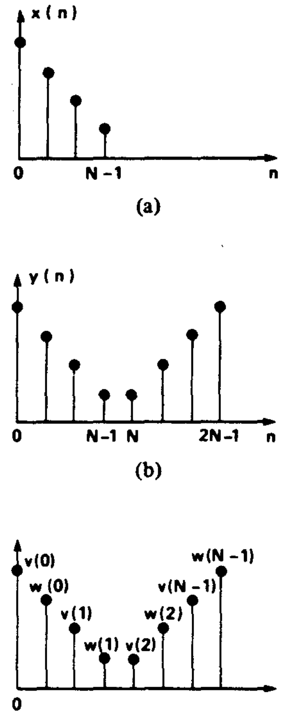

Fig. 3. (a) Original signal $x(n)$, $0 \leq n \leq N-1$. (b) A $2N$-point even extension of $x(n)$, $y(n)$. (c) Division of $y(n)$ into its even and odd parts $v(n)$ and $w(n)$.

By defining the DCT of $x(n)$ as [3]

$$C(k) = 2 \sum_{n=0}^{N-1} x(n) \cos \frac{\pi(2n+1)k}{2N}, \qquad 0 \leq k \leq N-1, \tag{8}$$

we have, from (6) and (8),

$$Y(k) = W_{2N}^{-k/2}\, C(k) \tag{9a}$$

or

$$C(k) = W_{2N}^{k/2}\, Y(k) \tag{9b}$$

and, from (7) and (9a),

$$C(k) = 2 \operatorname{Re}\left[W_{2N}^{k/2} \sum_{n=0}^{N-1} x(n)\, W_{2N}^{nk} \right]. \tag{10}$$

Equation (9) specifies the relationship between the DCT of a sequence and the DFT of the $2N$-point extension of that sequence. Note that $C(k)$ is real and $Y(k)$ is complex. $Y(k)$ would have been equal to $C(k)$ had the sequence $y(n)$ been delayed by half a sample, in which case $y(n)$ would have been truly even.

Therefore, the DCT of $x(n)$ may be computed by taking the $2N$-point DFT of $y(n)$, as in (3), and multiplying the result by $W_{2N}^{k/2}$, as in (9b). From (10) we see that the DCT may also be obtained by taking the $2N$-point DFT of the original sequence $x(n)$ with N zeros appended to it, multiplying the result by $W_{2N}^{k/2}$, then taking twice the real part. The latter method has been the one in common usage [4].

[3]The DCT definition here is slightly different from other definitions [4], mainly in the relative amplitude of $C(0)$ to that of other terms; also, we do not use an orthonormalizing factor. The range on k is the same here as in the literature; however, in this paper we shall also make use of $C(N)$, which, from (8), is equal to zero always since the cosine term is zero for $k = N$.

B. Inverse DCT

Again, we shall derive the inverse DCT (IDCT) from the inverse DFT (IDFT). The IDFT of $Y(k)$ is given by

$$y(n) = \frac{1}{2N} \sum_{k=0}^{2N-1} Y(k)\, W_{2N}^{-nk}. \tag{11}$$

Since $y(n)$ is real, $Y(k)$ is Hermitian symmetric:

$$Y(2N-k) = Y^*(k). \tag{12}$$

Furthermore, from (7), it is simple to show that

$$Y(N) = 0. \tag{13}$$

Using (12) and (13) in (11), one can show that

$$y(n) = \frac{1}{N} \operatorname{Re}\left[\frac{Y(0)}{2} + \sum_{k=1}^{N-1} Y(k)\, W_{2N}^{-nk} \right], \qquad 0 \leq n \leq 2N-1. \tag{14}$$

Substituting (9a) in (14) and using (1), we have the desired IDCT

$$x(n) = \frac{1}{N} \left[\frac{C(0)}{2} + \sum_{k=1}^{N-1} C(k) \cos \frac{\pi(2n+1)k}{2N} \right], \qquad 0 \leq n \leq N-1. \tag{15}$$

Equations (8) and (15) form a DCT pair. Given $C(k)$, $x(n)$ is retrieved by first computing $Y(k)$ using (9a), then taking the $2N$-point *complex* IDFT implied by (14), which results in $y(n)$ and, hence, $x(n)$.

III. Fast Cosine Transform (FCT)

A. Forward FCT

We now show that the DCT may be obtained from the N-point DFT of a real sequence instead of a $2N$-point DFT, resulting in a saving of 1/2.

Divide the sequence $y(n)$ into two N-point sequences

$$\left. \begin{aligned} v(n) &= y(2n) \\ w(n) &= y(2n+1) \end{aligned} \right\} \quad 0 \leq n \leq N-1 \tag{16}$$

where $v(n)$ and $w(n)$ are the sets of even and odd points in $y(n)$, respectively. Fig. 3(c) shows how this division takes place in the given example. Note that each of $v(n)$ and $w(n)$ contain *all* the original samples of $x(n)$, and that $w(n)$ is simply the reverse sequence of $v(n)$. In fact, from (2) and (16), one can show that

$$w(n) = v(N-n-1), \qquad 0 \leq n \leq N-1. \tag{17}$$

Substituting (16) in (3), we have

$$Y(k) = \sum_{n=0}^{N-1} v(n)\, W_{2N}^{2nk} + \sum_{n=0}^{N-1} w(n)\, W_{2N}^{(2n+1)k}. \tag{18}$$

Substituting (17) in (18), noting that $W_{2N}^{2nk} = W_N^{nk}$, rearranging terms, and using (9), one can show that

$$C(k) = 2\,\mathrm{Re}\left[W_{4N}^{k} \sum_{n=0}^{N-1} v(n)\, W_N^{nk}\right], \quad 0 \leqslant k \leqslant N-1. \tag{19a}$$

The difference between (19a) and (10) is the use of W_N instead of W_{2N} in the summation, and $v(n)$ instead of $x(n)$. The result is that one can now compute the DCT from an N-point DFT instead of a $2N$-point DFT. Equation (19a) may be rewritten as

$$C(k) = 2 \sum_{n=0}^{N-1} v(n) \cos \frac{\pi(4n+1)k}{2N}, \quad 0 \leqslant k \leqslant N-1, \tag{19b}$$

which gives an alternate defintion of the DCT in terms of the reordered sequence $v(n)$ [compare (19b) with (8)].

The sequence $v(n)$ can be written directly in terms of $x(n)$:

$$v(n) = \begin{cases} x(2n), & 0 \leqslant n \leqslant \left[\frac{N-1}{2}\right] \\ x(2N-2n-1), & \left[\frac{N+1}{2}\right] \leqslant n \leqslant N-1 \end{cases} \tag{20}$$

where $[a]$ denotes "integer part of a." Therefore, $v(n)$ is obtained by taking the even points in $x(n)$ in order, followed by the odd points in their reverse order. Note that (20) applies for any value of N, odd or even.

The specification of the FCT is now complete; the computational procedure is given in Section III-B. We simply note here that since $v(n)$ is real, its N-point DFT can be computed from the $(N/2)$-point DFT of a complex sequence (see the Appendix).

B. Inverse FCT

The idea here is to compute $v(n)$ from the DCT first, then use (20) to obtain $x(n)$. Substituting $v(n) = y(2n)$ in (14), we have

$$v(n) = \frac{1}{N}\,\mathrm{Re}\left[\frac{Y(0)}{2} + \sum_{k=1}^{N-1} Y(k)\, W_N^{-nk}\right], \quad 0 \leqslant n \leqslant N-1. \tag{21}$$

Equation (21) indicates that $v(n)$ can be computed using an N-point complex IDFT instead of the $2N$-point complex IDFT implied by (14). However, the number of computations is still about twice that used in the forward FCT. We now show that the inverse FCT can, in fact, be computed with the same number of computations as the forward FCT. The method is to compute $V(k)$ from $C(k)$, then compute the IDFT of $V(k)$ to obtain $v(n)$.

Equation (19a) can be rewritten as

$$C(k) = \mathrm{Re}\,[2W_{4N}^{k} V(k)] \tag{22}$$

where $V(k)$ is the DFT of $v(n)$. To compute $V(k)$ from $C(k)$ in (22) we need also a knowledge of the imaginary part of the term in brackets. Denote the imaginary part by $C_I(k)$ and the whole complex number by $C_c(k)$, where

$$C_c(k) = C(k) + jC_I(k) = 2W_{4N}^{k} V(k); \tag{23}$$

then

$$V(k) = \tfrac{1}{2} W_{4N}^{-k} C_c(k). \tag{24}$$

We first need to determine $C_I(k)$. Using the fact that $V(k)$ is Hermitian symmetric

$$V(N-k) = V^*(k), \tag{25}$$

one can show, using (23), that

$$C_c(N-k) = -jC_c^*(k) = -[C_I(k) + jC(k)]. \tag{26}$$

From (23) and (26), we conclude that

$$C_I(k) = -C(N-k)$$

and

$$C_c(k) = C(k) - jC(N-k) = 2W_{4N}^{k} V(k). \tag{27}$$

(Note that one can take advantage of (27) in (23) for computing $C(k)$ since one can compute $C(k)$ and $C(N-k)$ simultaneously.) From (27), we have

$$V(k) = \tfrac{1}{2} W_{4N}^{-k} [C(k) - jC(N-k)], \quad 0 \leqslant k \leqslant N-1. \tag{28}$$

$V(k)$ is computed from (28) for $0 \leqslant k \leqslant N/2$, then use (25) for $k > N/2$. In computing $V(0)$, one needs the value of $C(N)$, which, from (9b) and (13), is seen easily to be equal to zero

$$C(N) = 0. \tag{29}$$

After computing $V(k)$, $v(n)$ is obtained as the IDFT

$$v(n) = \frac{1}{N} \sum_{k=0}^{N-1} V(k)\, W_N^{-nk}. \tag{30}$$

It would seem that (30) again requires an N-point complex IDFT. However, in the Appendix we show how the IDFT of a Hermitian symmetric sequence can be computed using an $(N/2)$-point complex IDFT, the same as in the forward FCT.

We are now ready to specify the complete procedure for computing the FCT and the inverse FCT.

FCT Procedure: Given a real sequence $x(n)$, $0 \leqslant n \leqslant N-1$.

1) Form the sequence $v(n)$ from (20).
2) Compute $V(k)$, $0 \leqslant k \leqslant N-1$, the DFT of $v(n)$.
3) Multiply $V(k)$ by $2\exp(-j\pi k/2N)$. From (27), we see that the real part will be $C(k)$ and the negative of the imaginary part will be $C(N-k)$. Therefore, the value of k is varied in the range $0 \leqslant k \leqslant [N/2]$.

IFCT Procedure: Given the DCT $C(k)$, $0 \leqslant k \leqslant N-1$, and $C(N) = 0$.

1) Compute $V(k)$ from (28).
2) Compute the IDFT of $V(k)$, $v(n)$.
3) Retrieve $x(n)$ from $v(n)$ using (20).

C. Computational Considerations

Since most researchers have some form of the DFT available on their computers, the FCT and IFCT procedures given in the previous section can be implemented easily. Furthermore, the procedure is quite general and may be used for any value of N. We have also seen that the N-point DFT of $v(n)$ and the N-point IDFT of $V(k)$ can be computed from the $(N/2)$-point DFT and $(N/2)$-point IDFT, respectively, of some complex sequence. For a highly composite value of N, one can, of course, use the FFT to great advantage. Maximum savings accrue when N is a power of 2. In the latter case, if one makes use of the fact that the sequence is real, the total number of computations for the FCT or the IFCT is on the order of $N \log_2 N$, the same as in [1]. The major difference here is that we do not require a specialized algorithm.

In computing $C(k)$ from (27) one first takes the N-point DFT of $v(n)$, and therefore one needs a table of sines or cosines where the unit circle is divided into N equal segments. However, multiplying afterwards by W_{4N}^{k} requires a table where the unit circle is divided into $4N$ segments. Since k is in the range $0 \leqslant k \leqslant N-1$, the N values of sines and cosines are all in the first quadrant. This point is made to emphasize the fact that the DCT of a sequence of length N requires an exponential table four times as large as that required for an N-point DFT.

IV. Two-Dimensional Fast Cosine Transform

In this section we present results analogous to the 1-D case given in Sections II and III. We show how the DCT of a 2-D real sequence $\{x(n_1, n_2), 0 \leqslant n_1 \leqslant N_1 - 1, 0 \leqslant n_2 \leqslant N_2 - 1\}$ can be computed using an $(N_1 \times N_2)$-point real DFT instead of the $(2N_1 \times 2N_2)$-point real DFT required in the traditional method, resulting in a saving of 1/4. Since the methods used here are similar to the 1-D case, no detailed derivations will be given.

A. Two-Dimensional DCT and IDCT

In a manner analogous to the 1-D case in (1), define a $(2N_1 \times 2N_2)$-point even extension of $x(n_1, n_2)$ in the n_1 and n_2 directions:

$$y(n_1,n_2) = \begin{cases} x(n_1,n_2); & 0 \leqslant n_1 \leqslant N_1 - 1, \quad 0 \leqslant n_2 \leqslant N_2 - 1 \\ y(2N_1 - n_1 - 1, n_2); & N_1 \leqslant n_1 \leqslant 2N_1 - 1, \quad 0 \leqslant n_2 \leqslant N_2 - 1 \\ y(n_1, 2N_2 - n_2 - 1); & 0 \leqslant n_1 \leqslant N_1 - 1, \quad N_2 \leqslant n_2 \leqslant 2N_2 - 1 \\ y(2N_1 - n_1 - 1, 2N_2 - n_2 - 1); & N_1 \leqslant n_1 \leqslant 2N_1 - 1, \quad N_2 \leqslant n_2 \leqslant 2N_2 - 1. \end{cases} \tag{31}$$

Fig. 4 shows an example where $N_1 = 3$ and $N_2 = 4$; the numbers in the figure are the sample values. Note that the number of samples in $y(n_1, n_2)$ is $4N_1N_2$, i.e., four times that in $x(n_1, n_2)$.[4] The 2-D DFT of $y(n_1, n_2)$ is defined by

$$Y(k_1,k_2) = \sum_{n_1=0}^{2N_1-1} \sum_{n_2=0}^{2N_2-1} y(n_1,n_2)\, W_{N_1}^{n_1 k_1} W_{N_2}^{n_2 k_2}. \tag{32}$$

From (31) and (32), one can show that

$$Y(k_1,k_2) = W_{2N_1}^{-k_1/2} W_{2N_2}^{-k_2/2} C(k_1,k_2) \tag{33}$$

[4] In the m-D case, the number of samples in the extended sequence $y(n_1, \cdots, n_m)$ is 2^m times the number of samples in $x(n_1, \cdots, n_m)$.

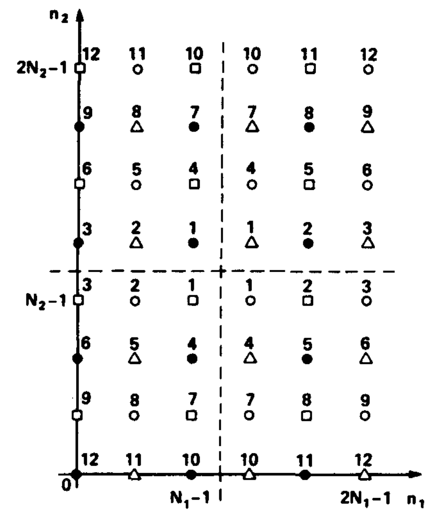

Fig. 4. An example of a 2-D $(N_1 \times N_2)$-point sequence and its $(2N_1 \times 2N_2)$-point even extension. Here, $N_1 = 3$ and $N_2 = 4$. The sample values are given numerically in the figure. The four sequences defined in (39) in the text are indicated in the figure as follows: filled circles, $v(n_1, n_2)$; triangles, $w_1(n_1, n_2)$; squares, $w_2(n_1, n_2)$; open circles, $w_3(n_1, n_2)$.

where

$$C(k_1,k_2) = 4 \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x(n_1,n_2) \cos \frac{\pi(2n_1+1)k_1}{2N_1} \cdot \cos \frac{\pi(2n_2+1)k_2}{2N_2},$$

$$0 \leqslant k_1 \leqslant N_1 - 1, \quad 0 \leqslant k_2 \leqslant N_2 - 1 \tag{34}$$

is the 2-D DCT of the sequence $x(n_1, n_2)$. The computation of $C(k_1, k_2)$ from either (32) and (33) or from (34) requires one to take the DFT of a $(2N_1 \times 2N_2)$-point real sequence.

From (32)-(34), one can show that $Y(k_1, k_2)$ has the following properties:

$$\begin{aligned} Y(2N_1 - k_1, 2N_2 - k_2) &= Y^*(k_1,k_2) \\ Y(2N_1 - k_1, 0) &= Y^*(k_1, 0) \\ Y(0, 2N_2 - k_2) &= Y^*(0, k_2) \\ Y(N_1, k_2) = Y(k_1, N_2) &= 0. \end{aligned} \tag{35}$$

The first three equations constitute the Hermitian symmetric properties in 2-D, and they are derivable from (32) for real $y(n_1, n_2)$. The last equation in (35) is analogous to (13) in 1-D, and is a consequence of the particular type of even symmetry of $y(n_1, n_2)$. Substituting (35) in the equation for the 2-D IDFT

$$y(n_1,n_2)=\frac{1}{4N_1N_2}\sum_{k_1=0}^{2N_1-1}\sum_{k_2=0}^{2N_2-1}Y(k_1,k_2)\,W_{2N_1}^{-n_1k_1}W_{2N_2}^{-n_2k_2}, \tag{36}$$

one can show that the 2-D IDCT of $C(k_1,k_2)$ is given by

$$x(n_1,n_2)=\frac{1}{N_1N_2}\sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1}C'(k_1,k_2)\cdot\cos\frac{\pi(2n_1+1)k_1}{2N_1}\cos\frac{\pi(2n_2+1)k_2}{2N_2} \tag{37}$$

where

$$C'(k_1,k_2)=\begin{cases}C(0,0)/4, & k_1=0,\ k_2=0\\ C(k_1,0)/2, & k_1\neq 0,\ k_2=0\\ C(0,k_2)/2, & k_1=0,\ k_2\neq 0\\ C(k_1,k_2), & k_1\neq 0,\ k_2\neq 0.\end{cases} \tag{38}$$

$x(n_1,n_2)$ may be computed using a 2-D $(2N_1\times 2N_2)$-point IDFT.

B. Two-Dimensional FCT and IFCT

The corresponding fast algorithms in the 2-D case are obtained in a manner analogous to the 1-D case. We divide $y(n_1,n_2)$ into four $(N_1\times N_2)$-point sequences, starting with the four points at (0, 0), (1, 0), (0, 1), and (1, 1), respectively, and taking every second point after that.[5] The four sequences are then given by

$$\begin{aligned}v(n_1,n_2)&=y(2n_1,2n_2)\\ w_1(n_1,n_2)&=y(2n_1+1,2n_2)\\ w_2(n_1,n_2)&=y(2n_1,2n_2+1)\\ w_3(n_1,n_2)&=y(2n_1+1,2n_2+1)\end{aligned} \tag{39}$$

$$v(n_1,n_2)=\begin{cases}x(2n_1,2n_2); & 0\leqslant n_1\leqslant\left[\frac{N_1-1}{2}\right],\quad 0\leqslant n_2\leqslant\left[\frac{N_2-1}{2}\right]\\ x(2N_1-2n_1-1,2n_2); & \left[\frac{N_1+1}{2}\right]\leqslant n_1\leqslant N_1-1,\quad 0\leqslant n_2\leqslant\left[\frac{N_2-1}{2}\right]\\ x(2n_1,2N_2-2n_2-1); & 0\leqslant n_1\leqslant\left[\frac{N_1-1}{2}\right],\quad \left[\frac{N_2+1}{2}\right]\leqslant n_2\leqslant N_2-1\\ x(2N_1-2n_1-1,2N_2-2n_2-1); & \left[\frac{N_1+1}{2}\right]\leqslant n_1\leqslant N_1-1,\quad \left[\frac{N_2+1}{2}\right]\leqslant n_2\leqslant N_2-1.\end{cases} \tag{44}$$

where $0\leqslant n_1\leqslant N_1-1$ and $0\leqslant n_2\leqslant N_2-1$. In the example in Fig. 4 note how each of the sequences in (39) contains all of the samples in $x(n_1,n_2)$, but in a reordered fashion. From (39) and (31), one can show that

$$\begin{aligned}w_1(n_1,n_2)&=v(N_1-n_1-1,n_2)\\ w_2(n_1,n_2)&=v(n_1,N_2-n_2-1)\\ w_3(n_1,n_2)&=v(N_1-n_1-1,N_2-n_2-1).\end{aligned} \tag{40}$$

Let $V(k_1,k_2)$ be the 2-D $(N_1\times N_2)$-point DFT of $v(n_1,n_2)$:

$$V(k_1,k_2)=\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1}v(n_1,n_2)\,W_{N_1}^{n_1k_1}W_{N_2}^{n_2k_2}. \tag{41}$$

Then, by substituting (40) and (39) in (32), and using (33), one can show that

$$C(k_1,k_2)=2\,\mathrm{Re}\,\{W_{4N_1}^{k_1}[W_{4N_2}^{k_2}V(k_1,k_2)+W_{4N_2}^{-k_2}V(k_1,N_2-k_2)]\} \tag{42a}$$

or equivalently,

$$C(k_1,k_2)=2\,\mathrm{Re}\,\{W_{4N_2}^{k_2}[W_{4N_1}^{k_1}V(k_1,k_2)+W_{4N_1}^{-k_1}V(N_1-k_1,k_2)]\}. \tag{42b}$$

From (42), it is simple to show that

$$C(k_1,k_2)=4\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1}v(n_1,n_2)\cos\frac{\pi(4n_1+1)k_1}{2N_1}\cdot\cos\frac{\pi(4n_2+1)k_2}{2N_2}. \tag{43}$$

It is clear from (42) and (41) that $C(k_1,k_2)$ can be computed using an $(N_1\times N_2)$-point DFT instead of a $(2N_1\times 2N_2)$-point DFT, resulting in a saving of 1/4.

The only remaining step is how to obtain $v(n_1,n_2)$ directly from $x(n_1,n_2)$ instead of using (39). One can show that (44) holds.

The 2-D IFCT of $C(k_1,k_2)$ is obtained by first computing $V(k_1,k_2)$ from $C(k_1,k_2)$. In a manner analogous to the 1-D case, we define a complex quantity $C_c(k_1,k_2)$ by not taking the real part in (42a), and then show that

$$C_c(k_1,k_2)=C(k_1,k_2)-jC(N_1-k_1,k_2). \tag{45}$$

Find $C_c^*(N_1-k_1,N_2-k_2)$, then add and subtract the result from $C_c(k_1,k_2)$. The answer can be shown to reduce to

$$V(k_1,k_2)=\tfrac{1}{4}\,W_{4N_1}^{-k_1}W_{4N_2}^{-k_2}\,\{[C(k_1,k_2)$$

[5] In the m-D case, $y(n_1,\cdots,n_m)$ is divided into 2^m sequences, starting with each of the 2^m corners of the unit m-D cube, from $(0,0,\cdots,0)$ to $(1,1,\cdots,1)$, and taking every second point after that. The sequence that begins at $(i_1,i_2,\cdots,i_m)$, where each $i_k=0$ or 1, is defined by $y(2n_1+i_1,2n_2+i_2,\cdots,2n_m+i_m)$.

$$- C(N_1 - k_1, N_2 - k_2)] - j[C(N_1 - k_1, k_2) + C(k_1, N_2 - k_2)]\} \tag{46}$$

where $0 \leq k_1 \leq N_1 - 1$ and $0 \leq k_2 \leq N_2 - 1$. However, since $V(k_1, k_2)$ is Hermitian symmetric, one need compute only half the values in (46). In performing the computations, one needs the fact that

$$C(N_1, k_2) = C(k_1, N_2) = 0, \quad \text{all } k_1 \text{ and } k_2, \tag{47}$$

which can be shown to be true from (33) and (35), or from (43).

After $V(k_1, k_2)$ is computed from (46) and (47), $v(n_1, n_2)$ is evaluated from the IDFT

$$v(n_1, n_2) = \frac{1}{N_1 N_2} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} V(k_1, k_2)\, W_{N_1}^{-n_1 k_1} W_{N_2}^{-n_2 k_2}. \tag{48}$$

The sequence $x(n_1, n_2)$ is then retrieved from $v(n_1, n_2)$ by using (44).

V. Conclusion

We showed how the DCT of an N-point sequence may be derived from the DFT of a $2N$-point even extension of the given sequence. Then, we presented a fast algorithm (FCT), first developed by Narasimha and Peterson [7], which allows for the computation of the DCT of a sequence from just an N-point DFT of a reordered version of the same sequence, with a resulting saving of 1/2. Therefore, one can use existing FFT software to compute the DCT. For N a power of 2, the number of computations is comparable to that reported by Chen *et al.* [1], who used a specialized algorithm.

The FCT algorithm in this paper is more general than that of [7] in that N may be odd or even. Also, an algorithm was developed here for computing the inverse FCT using the same number of computations as in the forward method.

The method was then extended to the 2-D case, where a saving of 1/4 was achieved. The method can be generalized to compute an m-D DCT using an m-D DFT, with a saving of $1/2^m$ over traditional methods that use the DFT.

Appendix
DFT and IDFT for a Real Sequence

Let $v(n)$, $0 \leq n \leq N - 1$ be a *real* sequence, where N is divisible by 2. We wish to compute the DFT of $v(n)$, $V(k)$, $0 \leq k \leq N - 1$, using an $N/2$-point DFT. Except for the flowgraphs in Fig. 5, the procedure given below is well known (see, for example, [6]).

DFT Procedure

1) Place the even and odd points of $v(n)$ in the real and imaginary parts, respectively, of a complex vector $t(n) = t_R(n) + j t_I(n)$, where

$$\left.\begin{aligned} t_R(n) &= v(2n) \\ t_I(n) &= v(2n+1) \end{aligned}\right\} 0 \leq n \leq \frac{N}{2} - 1. \tag{A-1}$$

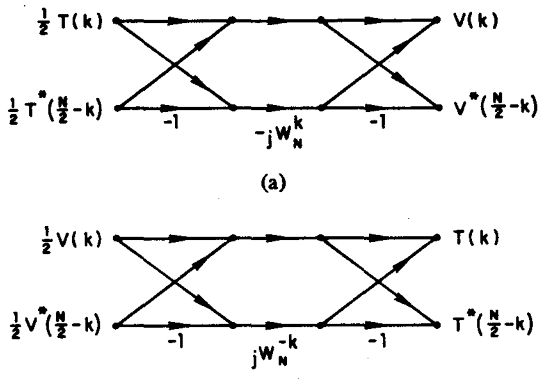

Fig. 5. (a) Supplementary flowgraph for computing the FFT of a real sequence. The computation in the figure is performed for $0 \leq k \leq [N/4]$. (b) Supplementary flowgraph for computing the IFFT of a Hermitian symmetric sequence. The computation is performed for $0 \leq k \leq [N/4]$.

2) Compute the $N/2$-point DFT of $t(n)$, $T(k)$, $0 \leq k \leq N/2 - 1$.

3) Compute $V(k)$ from $T(k)$ using the formula [5]

$$V(k) = \frac{1}{2}\left[T(k) + T^*\left(\frac{N}{2} - k\right)\right] - jW_N^k \frac{1}{2}\left[T(k) - T^*\left(\frac{N}{2} - k\right)\right]. \tag{A-2}$$

The computations in the last step can be made more efficient. From (A-2), one can write

$$V^*\left(\frac{N}{2} - k\right) = \frac{1}{2}\left[T(k) + T^*\left(\frac{N}{2} - k\right)\right] + jW_N^k \frac{1}{2}\left[T(k) - T^*\left(\frac{N}{2} - k\right)\right]. \tag{A-3}$$

Given $T(k)$, Fig. 5(a) shows the flowgraph [5] that implements (A-2) and (A-3) to compute $V(k)$. Note that the values of $V(k)$ are computed two at a time. Therefore, the value of k in Fig. 5(a) should range between $0 \leq k \leq [N/4]$. The other values of $V(k)$, $k > N/2$, may be obtained by noting that $V(k)$ is Hermitian symmetric. There are two points in $V(k)$ that are real and require no multiplication. They are

$$\begin{aligned} V(0) &= \mathrm{Re}\,[T(0)] + \mathrm{Im}\,[T(0)] \\ V\left(\frac{N}{2}\right) &= \mathrm{Re}\,[T(0)] - \mathrm{Im}\,[T(0)]. \end{aligned} \tag{A-4}$$

Also, if N is divisible by 4, one can show that

$$V\left(\frac{N}{4}\right) = T^*\left(\frac{N}{4}\right).$$

Given a Hermitian symmetric $V(k)$, $0 \leq k \leq N - 1$, we wish to compute the IDFT, $v(n)$, $0 \leq n \leq N - 1$. From (A-2) and (A-3), one can easily solve for $T(k)$ and $T^*(N/2 - k)$. The resulting equations can be implemented using the flowgraph in Fig. 5(b). The IDFT procedure is, then, as shown in the following.

IDFT Procedure

1) Compute $T(k)$ from $V(k)$ using the flowgraph in Fig. 5(b), where the range of k in the figure is $0 \leq k \leq [N/4]$.

2) Compute the $(N/2)$-point IDFT of $T(k)$, $t(n)$, $0 \leq n \leq (N/2) - 1$.

3) $v(n)$ is obtained from $t(n)$ by using (A-1).

Finally, if N is not divisible by 2, one can compute the DFT of two separate sequences using the same method given above [6].

Acknowledgment

The author wishes to thank M. Berouti and the reviewers for their comments.

References

[1] W. Chen, C. H. Smith, and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Trans. Commun.*, vol. COM-25, pp. 1004-1009, Sept. 1977.

[2] R. Zelinski and P. Noll, "Adaptive transform coding of speech signals," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, pp. 299-309, Aug. 1977.

[3] J. M. Tribolet and R. E. Crochiere, "A vocoder-driven adaptation strategy for low-bit rate adaptive transform coding of speech," in *Proc. 1978 Int. Conf. Digital Signal Processing*, Florence, Italy, Aug. 30-Sept. 2, 1978.

[4] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-25, pp. 90-93, Jan. 1974.

[5] J. Makhoul, "Speaker-machine interaction in automatic speech recognition," Res. Lab. Electron., Massachusetts Inst. Tech., Cambridge, MA, Tech. Rep. 480, Dec. 15, 1970; also, Ph.D. dissertation, Dept. Elec. Eng., Massachusetts Inst. Tech., Cambridge, MA, 1970.

[6] E. O. Brigham, *The Fast Fourier Transform.* Englewood Cliffs, NJ: Prentice-Hall, 1974, pp. 166-169.

[7] M. J. Narasimha and A. M. Peterson, "On the computation of the discrete cosine transform," *IEEE Trans. Commun.*, vol. COM-26, pp. 934-936, June 1978.

[8] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1975.

John Makhoul (S'64-M'70-SM'78) was born in Deirmimas, Lebanon, on September 19, 1942. He received the B.E. degree from the American University of Beirut, Beirut, Lebanon, in 1964, the M.Sc. degree from Ohio State University, Columbus, in 1965, and the Ph.D. degree from the Massachusetts Institute of Technology, Cambridge, in 1970, all in electrical engineering.

From 1964-1966 he did research in electromagnetic diffraction at the Antenna Laboratory, Ohio State University. From 1966-1970 he was a Research Assistant at the Research Laboratory of Electronics, Massachusetts Institute of Technology, working on speech synthesis, automatic speech recognition, and analog and digital systems design. Since 1970 he has been with Bolt Beranek and Newman, Inc., Cambridge, MA, where he is a Supervisory Scientist working primarily on speech analysis, synthesis and compression, speech enhancement, automatic speech recognition, and other aspects of digital signal processing, including lattice structures and adaptive digital filtering.

Dr. Makhoul is a Fellow of the Acoustical Society of America.