産業技術硏究(江原大學校 産業技術硏究所 論文集) 第4輯(1984)
Res. Report, The Inst. of Industrial Technology, Kangweon Nat. Univ., Korea, Vol. 4

DCT와 DST의 高速 알고리즘 開發

朴 鍾 演*

The Development of the Fast Algorithm for the DCT and DST

Chong Yeun Park*

Abstract

A new FDCT algorithm was developed. It needed only half numbers of the discrete output signals and made use of the prime factor FFT algorithm. The FDST algorithm of the second kind of DST was developed by use of the FDCT algorithm.

1. 序 論

FDCT (Fast Discrete Cosine Transform)알고리즘은 영상신호 處理過程[1]과 多重通信分野[2]에서 重要한 硏究과제이며 FDST(Fast Discrete Sine Transform)알고리즘[3]은 영상信號의 코우딩(coding)에서 必要하다. DCT는 N. AHMED[4]에 依하여 定義된 後 $N=2^m$點 DCT에 對하여 FFT 알고리즘을 利用한 高速處理方法이 硏究[5)-7)]되었고 J. Makhaul[8]에 依하여 2次元의 處理過程으로 확장하였다. 그러나 이러한 FDCT 알고리즘은 $N=2^m$點에 대해서는 대단히 有用하지만 $N\neq 2^m$點인 FDCT에 對한 處理에는 N의 값이 크면 클수록 不適合하다고 판단된다.

그러므로 本 硏究에서는 $N\neq 2^m$點 FDCT알고리즘을 얻기 위해서 素因數(Prime factor) FFT(Fast Fourier Transform)알고리즘[9)-12)]을 FDCT의 處理過程에 利用하였다. 또한 FDST는 入力信號의 홀수 번째의 信號에 對하여 부호를 變化시켜 FDCT 알고리즘을 利用하여 처리할 수 있음을 밝혔다. 이러한 FDCT와 FDST 알고리즘은 PRIME 750 컴퓨터의 시뮬레이션 方法으로 確認되었으며 기존의 方法[5)-7)]보다 計算時間의 大部分을 차지하는 곱셈回數가 最少 約 25% 감소함을 밝혔다. 例로서 多重通信의 Group Band에서 使用되는 14點 FDCT 處理結果를 나타내었다.

2. FDCT 알고리즘

M個의 離散信號(Discrete Signal)$x(m)$, m=0 1, ……(M−1)의 DCT는 N. AHMED[4]에 依하여 다음과 같이 定義된다.

$$\left.\begin{aligned} X(o)&=\frac{\sqrt{2}}{M}\sum_{m=0}^{M-1}x(m)\\ X(k)&=\frac{2}{M}\sum_{m=0}^{M-1}x(m)\cdot\cos\left[\frac{(2m+1)}{2M}k\pi\right]\end{aligned}\right\}\quad(1)$$

단 $k=1,2,\cdots\cdots\cdots(M-1)$

式(1)에서 $\frac{\sqrt{2}}{M}$와 $\frac{2}{M}$는 常數이므로 $Z(m)=x(m)\frac{2}{M}$을 離散入力信號로 생각하면 결국 $x(m)$

* 江原大學校 工科大學 電氣工學科 助敎授
* Assistant Professor, Dept. of Electrical Engineering, Kangweon National University

이 $\frac{2}{M}$倍 만큼 증폭된 効果를 갖는다. 따라서 式(I)은 다음과 같이 表現된다.

$$\left.\begin{aligned} Z(o)&=\sum_{m=0}^{M-1} z(m)\cdot\frac{1}{\sqrt{2}} \\ Z(k)&=\sum_{m=0}^{M-1} z(m)\cos\left[\frac{(2m+1)}{2M}k\pi\right] \end{aligned}\right\} \cdots\cdots(2)$$

단 $k=1,2,\cdots\cdots(M-1)$

本 研究에서는 式(2)를 計算하는데 보다 능율적인 方法을 제시하였다. 기존의 研究에서는 式(2)를 計算하는데 2M—點 FFT 알고리즘[4] 혹은 M 個의 入出力 信號를 必要로 하는 M點 FFT 알고리즘[5]을 適用하였다. 그러나 本 研究에서는 M個의 入力信號에서 $\frac{M}{2}$個의 出力信號만을 얻어서 處理하는 方法으로 그 알고리즘의 유도과정은 다음과 같다.

式(2)의 Z(k)를 얻는데 必要한 計算을 高速으로 처리하기 위한 方法으로 式(2)를 다음과 같이 변형한다.

$$\begin{aligned} Z(k)=&\sum_{m=0}^{\frac{M}{2}-1} z(2m)\cos\left[\frac{\pi(2\cdot 2m+1)k}{2M}\right] \\ &+\sum_{m=0}^{\frac{M}{2}-1} z(2m+1)\cos\left[\frac{\pi\{2\cdot(2m+1)+1\}}{2M}k\right] \end{aligned} \cdots\cdots(3)$$

단 $k=1,2,\cdots\cdots(M-1)$

式(3)에서 入力 離散信號 z(m)을 다음과 같은 y(m)으로 再配置(Rearrangement)하여 생각한다.

$$\left.\begin{aligned} y(m)&=z(2m) \\ y(M-1-m)&=z(2m+1) \end{aligned}\right\} \cdots\cdots(4)$$

단 $m=0,1,\cdots\cdots(M-1)$

式(4)를 利用하여 式(3)를 表現하면 다음과 같다.

$$\begin{aligned} Z(k)=&\sum_{m=0}^{\frac{M}{2}-1}\left[y(m)\cos\left\{\frac{\pi(4m+1)}{2M}k\right\}\right. \\ &\left.+y(M-1-m)\cos\left\{\frac{\pi(4m+3)}{2M}k\right\}\right] \end{aligned} \quad (5)$$

단 $k=1,2,\cdots\cdots(M-1)$

이러한 式(5)의 둘째 項을 G(k)라면 다음과 같다.

$$G(k)=\sum_{m=0}^{\frac{M}{2}-1} y(M-1-m)\cdot\cos\left\{\frac{\pi(4m+3)}{2M}k\right\}$$

$$\cdots\cdots(6)$$

式(6)에서 m′=M−1−m 으로 변수를 치환하여 다음과 같이 表示할 수 있다.

$$G(k)=\sum_{m'=\frac{M}{2}}^{M-1} y(m')\cdot\cos\left[\frac{\pi k(4m'+1)}{2M}\right] \quad (7)$$

式(5)는 式(7)을 이용하면 다음과 같다.

$$Z(k)=\sum_{m=0}^{M-1} y(m)\cos\left\{\frac{\pi k(4m+1)}{2M}\right\} \cdots\cdots(8)$$

式(8)에서 $Z(k)=Z_R(k)+jZ_I(k)$, $y(m)=y_R(m)+jy_I(m)$이라면 다음 관계식이 成立한다.

$$Z_R(k)=\sum_{m=0}^{M-1} y_R(m)\cdot\cos\left\{\frac{\pi k(4m+1)}{2M}\right\} \cdots\cdots(9)$$

$$Z_I(k)=\sum_{m=0}^{M-1} y_I(m)\cdot\cos\left\{\frac{\pi k(4m+1)}{2M}\right\} \cdots(10)$$

式(8)의 Z(k)를 계산하기 위해서는 式(9)와 式(10)을 計算하면 되며 式(10)의 計算方法은 式(9)의 計算方法과 同一하므로 式(9)의 計算方法만을 說明하였다. 式(9)는 指數함수로 表現되며 $R_e[x]$는 x의 실수부분을 意味한다.

$$Z_R(k)=R_e\left[\sum_{m=0}^{M-1} y_R(m)\cdot\exp\left(j\cdot\frac{2\pi}{M}km\right)\cdot\exp\left(j\cdot\frac{\pi k}{2M}\right)\right] \cdots\cdots(11)$$

여기서 $y_R(m)$의 IDFT(Invese Discrete Fourier Transform)을 $Y_R(k)$라 하고 다음 式으로 쓸 수 있다.

$$Y_R(k)=\sum_{m=0}^{M-1} y_R(m)\exp\left(j\frac{2\pi}{M}km\right) \cdots\cdots(12)$$

그러므로 式(11)은 $Y_R(k)$에 依해서 表示할 수 있다.

$$Z_R(k)=R_e\left[Y_R(k)\cdot\exp\left(j\cdot\frac{\pi k}{2M}\right)\right] \cdots(13)$$

또한 式(13)에서 $V_R(k)=Y_R(k)\cdot\exp\left(j\frac{\pi k}{M}\right)$으로 가정하면 $Z_R(k)$는 다음과 같다.

$$Z_R(k)=R_e[V_R(k)] \cdots\cdots(14)$$

단 $k=1,2,\cdots\cdots\frac{M}{2}$

그런데 式(13)에서 $Z_R(M-k)$는

$$Z_R(M-k)=R_e\Big[\,Y_R(m-k)\cdot\exp\Big(j\frac{\pi}{2M}(M-k)\Big)\Big]$$

$$=R_e\Big[j\cdot\exp\Big(-j\frac{\pi k}{2M}\Big)\sum_{m=0}^{M-1}y_R(m)\cdot\exp\Big(-j\frac{2\pi k}{M}m\Big)\Big]$$

$$=R_e\Big[j\sum_{m=0}^{M-1}y_R(m)\cdot\exp\Big(-j\cdot\frac{\pi k(4m+1)}{2M}\Big)\Big]$$

$$=\sum_{m=0}^{M-1}y_R(m)\cdot\sin\Big(\frac{\pi k(4m+1)}{2M}\Big)$$

$$=I_m[V_R(k)] \qquad \cdots\cdots(15)$$

단 $k=1,2,\cdots\cdots\frac{M}{2}$

와 같이 되므로 式(8)을 계산하기 위해서는 式(9)의 $Z_R(k)$ 및 式(10)의 $Z_I(k)$를 계산하여야 한다. 그런데 $Z_R(k)$는 式(14)와 式(15)에서 $V_R(k)$를 $k=1,2,\cdots\cdots\frac{M}{2}$個 計算하여 얻을 수 있다. 이와 유사한 方法으로 式(10)의 $Z_I(k)$를 계산하기 위한 式을 쓰면 다음과 같다.

$$Z_I(k)=R_e[V_I(k)]$$

단 $k=1,2,\cdots\cdots\frac{M}{2}$ $\cdots\cdots(16)$

$$Z_I(M-k)=I_m[V_I(M-k)]$$

단 $k=1,2,\cdots\cdots\frac{M}{2}$ $\cdots\cdots(17)$

단 $V_I(k)=\exp\Big(j\frac{\pi k}{2M}\Big)\cdot\sum_{m=0}^{M-1}y_I(m)\exp\Big(j\frac{2\pi}{M}km\Big)$ (18)

이상의 FDCT 알고리즘을 圖示하면 그림 1과 같다.

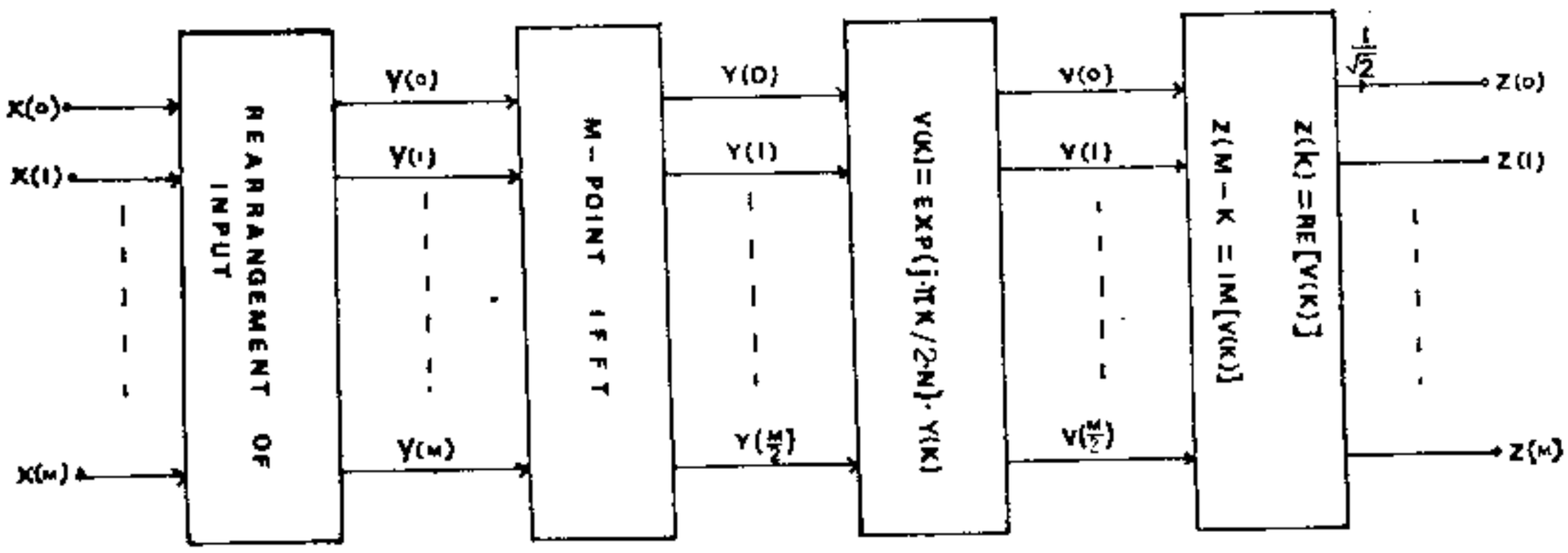


Fig. 1. FDCT System

그림 1에서 $x(k)=x_R(k)+jx_I(k)$, $y(k)=y_R(k)+jy_I(k)$, $Y(k)=Y_R(k)+jY_I(k)$, $V(k)=V_R(k)+jV_I(k)$ 및 $Z(k)=Z_R(k)+jZ_I(k)$를 의미한다. 이러한 FDCT 시스템에서 計算時間의 大部分을 차지하는 곱셈回數가 가장 많이 要求되는 것은 M一點 IFFT이다. 그림 1의 IFFT에 對하여 $M=2^m$인 點의 FFT 알고리즘은 素因數 FFT[9-11] 알고리즘이 곱셈回數를 줄이는데 有力하다. 素因數 FFT 알고리즘을 利用한 例를 컴퓨터 시뮬레이션 過程에서 說明하였다.

3. FDST 알고리즘

M—點 DST(Discrete Sine Transform)는 一種 DST와 二種 DST로 區分된다.[3] 本 硏究에서는 二種 DST의 高速計算方法으로 前項의 FDCT 알고리즘을 利用하는 方法을 다루었다. 入力離散信號 $x(n)$의 二種 DST는 다음과 같이 定義된다.

$$S(o)=\frac{\sqrt{2}}{M}\sum_{n=0}^{M-1}x(n)$$

$$S(m)=\frac{2}{M}\sum_{n=0}^{M-1}x(n)\cdot\sin\Big[\frac{(2n+1)}{2M}m\pi\Big] \qquad \cdots\cdots(19)$$

단 $m=0,1,\cdots\cdots(M-1)$

이러한 式(19)의 DST의 結果는 入力離散信號 $x(n)$의 홀수 번째 값의 부호를 바꾼 信號 $(-1)^n\cdot x(n)$ (단 $n=0,1,2,\cdots\cdots(M-1)$)을

DCT로 처리하여 얻은 結果와 같다. 이것을 證明하기 위하여 式(1)에서 변수 $k=M-m$으로 치환하고 入力離散信號 $x(n)$ 대신에 $(-1)^n x(n)$을 代入하여 다음과 같이 유도할 수 있다. $\bar{X}(\cdot)$은 $(-1)^n \cdot x(n)$의 DCT를 나타낸다.

$$\bar{X}(M-m)=\frac{2}{M}\sum_{n=0}^{M-1}x(n)\cdot(-1)^n \cdot\cos\left[\frac{(2n+1)(M-m)}{2M}\pi\right] \quad (20)$$

단 $m=1,2,\cdots\cdots M$

그런데 式(20)의 $\cos[\cdot]$은 다음과 같은 삼각함수의 項等式이 成立한다.

$$\cos\left[\frac{(2n+1)(M-m)}{2M}\pi\right]$$
$$=\cos\left\{\frac{(2n+1)}{2}\pi\right\}\cdot\cos\left\{\frac{(2n+1)}{2M}m\pi\right\}$$
$$+\sin\left\{\frac{(2n+1)}{2}\pi\right\}\cdot\sin\left\{\frac{(2n+1)}{2M}m\pi\right\}$$
$$=(-1)^n\cdot\sin\left\{\frac{(2n+1)}{2M}m\pi\right\} \quad \cdots\cdots(21)$$

式(21)을 式(20)에 代入하여 整理하고 式(19)와 比較하면 $x(n)$의 DST는 DCT로 처리가능한 다음 式을 얻을 수 있다.

$$\bar{X}(M-m)=\frac{2}{M}\cdot\sum_{n=0}^{M-1}x(n)\ (-1)^n\cdot(-1)^n \cdot\sin\left\{\frac{(2n+1)}{2M}m\pi\right\}$$
$$=\frac{2}{M}\cdot\sum_{n=0}^{M-1}x(n)\cdot\sin\left\{\frac{(2n+1)}{2M}m\pi\right\}$$
$$=S(m) \quad \cdots\cdots(22)$$

式(22)를 보면 入力離散信號 $x(n)$의 DST는 $(-1)^n x(n)$을 DCT 시킨 값을 얻은 뒤 出力離散信號의 순서를 逆으로 取하여 얻어진다. 이러한 DST의 計算절차를 圖示하면 그림 2와 같으며 결국 高速 DST 알고리즘은 FDCT 처리과정의 計算時間에 依하여 결정된다. 그런데 式(19)의 S(o)와 式(1)의 X(o)는 일치함을 確認할 수 있다.

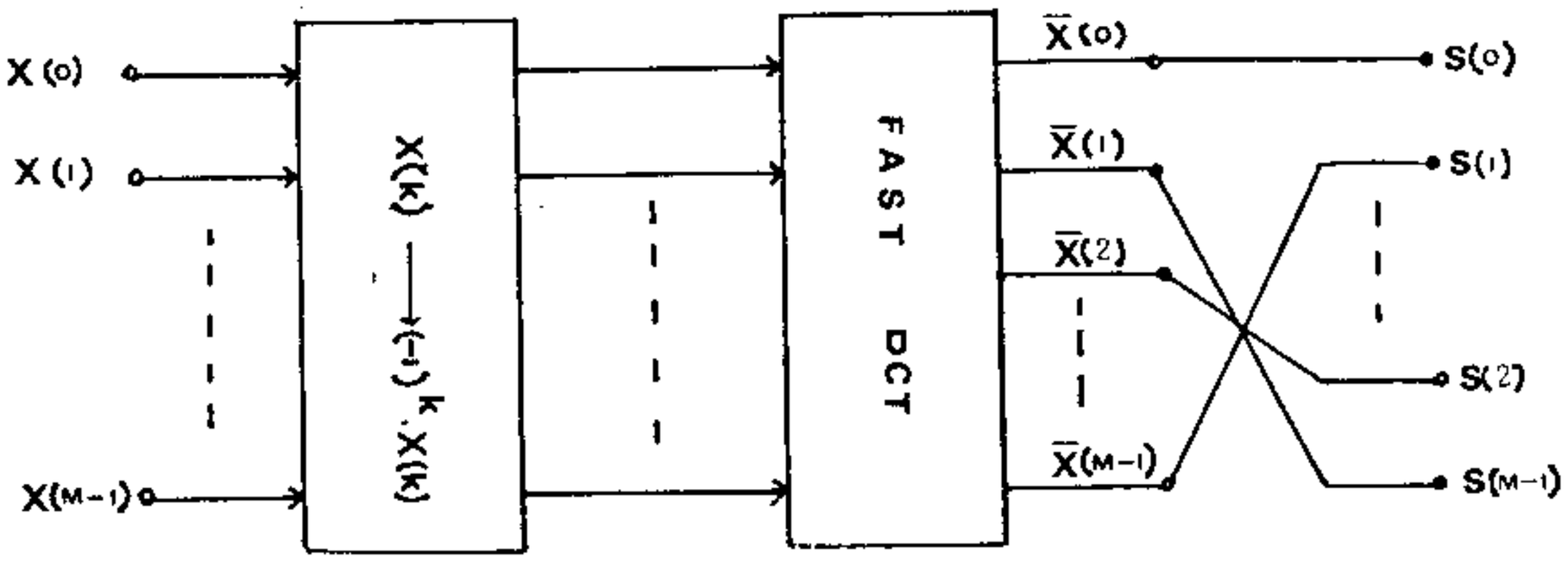


Fig. 2. DST System

4. 컴퓨터에 依한 FDCT와 FDST의 Simulation

FDCT와 FDST의 시뮬레이션 方法은 그림 1과 그림 2에 관한 것으로 앞의 理論에 對한 타당성을 밝히는데 目的이 있으며 FDCT에 對한 시뮬레이션은 多重通信 시스템에서 Group—band (60 KHz — 108 KHz)의 TDM/FDM 變換 (Time Division Multiplexing/Frequency Division Multiplexing Transmultiplexer)過程[2]에서 必要한 14—Point DCT의 경우를 例로 들었다.

그림 1의 시스템에서 M=14로 취하여 實現할 경우에 14—Point IFFT가 必要하며 $14\neq 2^m$이기 때문에 보통의 $M=2^m$ Point FFT 알고리즘은 計算時間의 側面에서 부적합하다. 따라서 本 硏究에서는 C. Burrus[9]에 依한 素因數 FFT 알고리즘을 利用하였다. C. Burrus[9]에 依한 理論을 인용하면 다음의 DFT C(k)는,

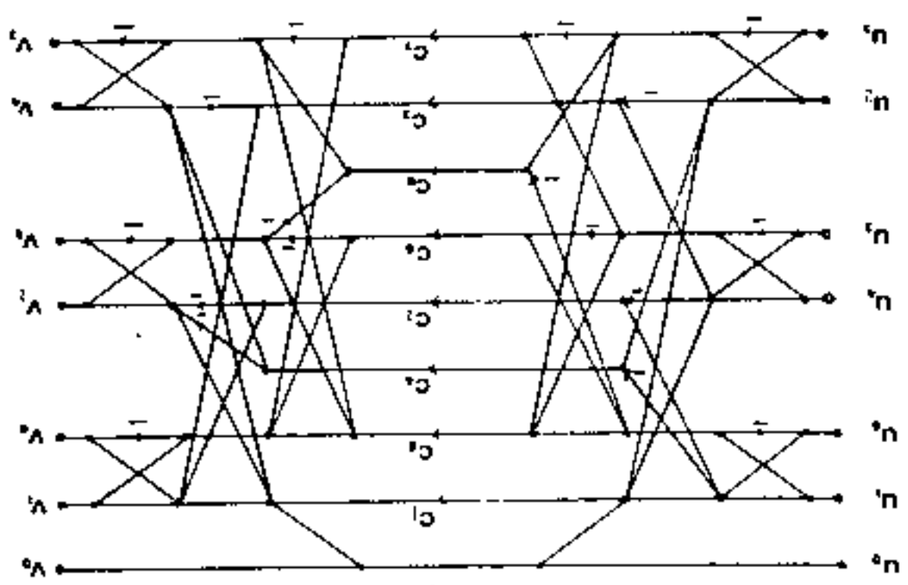
$$C(k)=\sum_{n=0}^{N-1}x(n)\ \ W_N^{nk} : W_r=e^{-j\frac{2\pi}{N}} \quad \cdots\cdots(23)$$

이며 이러한 1次元 DFT는 다음 式과 같은 2次元 DFT로 처리 가능함을 밝히고 있다.

$$\hat{C}(k_1, k_2) = \sum_{n_2=0}^{N_2-1} \sum_{n_1=0}^{N_1-1} \hat{x}(n_1, n_2) W_{N1}^{n1k1} W_{N2}^{n2k2} \quad \cdots\cdots(24)$$

式(24)에서 $\hat{C}(k_1, k_2)$는 C(k)의 二次元으로 表示한 것이며 $\hat{x}(n_1, n_2)$는 $x(n)$의 일부로서 素因數 FFT로 처리하기 위한 것이며 $N=N_1, N_2$로 분해된다. 이때 N_1과 N_2는 素因數이다.

本 硏究에서 例로서 取한 N=14의 境遇에 $N_1=7$, $N_2=2$로 素因數 분해하여 2—Point FFT와 7—Point FFT 알고리즘이 必要하다. 2—Point DFT는 實數 離散 入力 信號에 관하여 2回의 곱셈으로 처리되며 7—Point DFT는 Wingograd FFT 方法[10]을 적용하였다. 7—Point IFFT 알고리즘을 圖示하면 그림 3과 같이 實數 離散 入力信號에 對해서 8回의 곱셈이 必要하다. 그림 3은 Wingograd FFT 알고리즘[10]을 IFFT로 변형한 것이다.



$C_1=-1.16666667$, $C_2=0.79015647$, $C_3=0.055854267$, $C_4=0.7343022$, $C_5=-0.44095855$, $C_6=-0.34087293$, $C_7=-0.53396936$, $C_8=-0.87484229$

Fig.3. Seven-point DFT using Wingograd's FFT algorithm

그림 1에서 나타낸 FDCT 알고리즘은 M의 값이 크면 클수록 素因數 要素가 많아져서 일반적인 FFT 알고리즘을 適用한 方法보다 有利하며 그림 1과 같은 FDCT 알고리즘의 컴퓨터에 依한 시뮬레이션 方法은 부록에 나타내었다. 例로서 入力信號를 $x(n)=n+j\frac{M}{10}$ (단 n=1, 2,……14)으로 取하여 式(1)에 依한 計算結果와 그림 1의 시스템에 對한 FDCT의 시뮬레이션 結果는 표 1에 나타낸 것처럼 거의 일치하였다.

또한 그림 1의 FDCT 알고리즘은 부록의 NN=M으로 임의의 M點에 對해서 成立함이 確認되었으며 그림 2에 對한 FDST 시스템의 시뮬레이션도 그림 1의 시스템과 유사한 方法으로 確認되었다.

Table 1. Simulation results of DCT and FDCT

	DCT BY DEF	**DCT BY IFFT**
1	103.0000000000	103.0000000000
2	−41.5824127197	−41.5823669434
3	−1.6883797646	−1.6883680820
4	−5.5412521362	−5.5412721634
5	−0.6400400400	−0.6400339603
6	−1.4333362579	−1.4333815575
7	0.6876564026	0.6875635386
8	1.8887197971	1.8887019157
9	1.4892206192	1.4892072678
10	1.9053502083	1.9053459167
11	0.9690750837	0.9690437317
12	1.0906260014	1.0906097889
13	−3.5483131409	−3.5483303070
14	10.2999954224	10.2999954224

5. 結 論

DCT의 離散入力信號의 再配置와 M點 FFT의 離散出力信號 가운데 $\frac{M}{2}$點을 計算하여 利用하는 DCT의 高速 알고리즘을 開發하였다. 이 方法은 기존의 M點 FFT 알고리즘에 必要한 計算量을 約 25% 감소 시킬 수 있다.

또한 DST의 高速 알고리즘은 삼각함수의 항등관계式을 適用하여 FDCT 알고리즘을 適用할 수 있음을 밝혔다. 여기서 밝힌 M點 FDCT와 FDST 알고리즘은 M이 2의 제곱 승이 아닌 境遇에 適合하므로 여기서 使用되는 FFT 알고리즘은 素因數 FFT 알고리즘이 가장 適合한 것

으로 나타났다. 이러한 FDCT와 FDST 알고리즘은 PRIME 750 컴퓨터에 依한 시뮬레이션 方法으로 理論의 타당성을 확인하였다.

參 考 文 獻

1. W. Chen, "A fast computational algorithm for the discrete cosine transform", IEEE Trans. Commun., Vol. COM—25, pp.1004—1009, Sep. 1978
2. M.J. Narasimha and A.M. Peterson, "Design of a 24-channel transmultiplexer", IEEE Trans, Acous., Speech, and Sig. Proc., Vol. ASSP—27, pp.752—762, Dec. 1979
3. Z. Wang, "A fast algorithm for the discrete sine transform implemented by the fast cosine transform", IEEE Trans. Acous., Speech, and Sig. Proc., Vol. ASSP—30, pp.814—815, Oct. 1982
4. N. Ahmed, T. Natarajan, and K.R. Rao, "Discrete cosine transform", IEEE Trans. Computers, pp.90—93, Jan. 1974
5. Z. Wang, "Reconsideration of a fast computational algorithm for the discrete cosine transform", IEEE Trans. Commun., Vol. COM—31, pp.121—123, Jan. 1983
6. 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
7. G. Bertocci, B.W. Scheonherr, and D.G. Messerschmitt, "An approach to the implementation of a discrete cosine transform", IEEE Trans. Commun., Vol. COM—30, pp.635—637, Apr. 1982
8. J. Makhoul, "A fast cosine transform in one and two dimension", IEEE Trans. Acous., Speech, and Sig. Proc., Vol. ASSP—28, pp. 27—34, Feb. 1980
9. C.S. Burrus, and P.W. Eschenbacher, "An in-place, in-order prme factor FFT algorithm", IEEE Trans. Acous., Speech, and Sig. Pooc., Vol. ASSP—29, pp.806—817, Aug. 1981
10. J.H. Rothweiler, "Implementation of the in-order prime factor transform for variable sizes", IEEE Trans. Acous., Speech, and Sig. Proc., Vol. ASSP—30, pp.105—109, Feb. 1982
11. D.P. Kolba, and T.W. Parhs, "A prime factor FFT algorithm using high Speed convolution", IEEE Trans. Acous., Speech, and Sig. Pooc., Vol. ASSP—25, pp.281—294, Aug. 1977
12. S. Chu, and C.S. Bruss, "A prime factor FFT algorithm using distributed arithmetic", IEEE Trans. Acous., Speech, and Sig. Proc., Vol. ASSP—30, pp.217—226, Apr. 1982
13. Programs for digital signal processing. New York : IEEE Press, 1979

부 록

```
      SUBROUTINE IFFT(X, Y, NN, M,
     NI, A, B)
      INTEGER NI(14), I(16)
      REAL X(1), Y(1), A(1), B(1)
      UNSC=FLOAT(9.)
      C71=-1.16666667
      C72=0.79015647
      C73=0.055854267
      C74=0.7343022
      C75=-0.44095855
      C76=0.34087293
      C77=-0.53396936
      C78=-0.87484229
   77 DO 10 K=1, M
      N1=NI(K)
      N2=NN/N1
      DO 20 J=1, NN, N1
      I(1)=J
      IT=J
      DO 30 L=1, N1
      IT=IT+N2
      IF(IT. GT. NN) IT=IT-NN
      I(L)=IT
   30 CONTINUE
      IF(N1. EQ. 2) GO TO 102
      IF(N1. EQ. 7) GO TO 107
   20 CONTINUE
   10 CONTINUE
      L=1
      DO 2 K=1, NN
      A(K)=X(L)
      B(K)=Y(L)
      L=L+UNSC
      IF(L. GT. NN) L=L-NN
    2 CONTINUE
      RETURN
  102 T=X(I(1))
      X(I(1))=T+X(I(2))
      X(I(2))=T-X(I(2))
      T=Y(I(1))
      Y(I(1))=T+Y(I(2))
      Y(I(2))=T-Y(I(2))
      GO TO 20
  107 R1=X(I(2))+X(I(7))
      R2=X(I(2))-X(I(7))
      S1=Y(I(2))+Y(I(7))
      S2=Y(I(2))-Y(I(7))
      R3=X(I(3))+X(I(6))
      R4=X(I(3))-X(I(6))
      S3=Y(I(3))+Y(I(6))
      S4=Y(I(3))-Y(I(6))
      R5=X(I(4))+X(I(5))
      R6=X(I(4))-X(I(5))
      S5=Y(I(4))+Y(I(5))
      S6=Y(I(4))-Y(I(5))
      T1=R1+R3+R5
      U1=S1+S3+S5
      X(I(1))=X(I(1))+T1
      Y(I(1))=Y(I(1))+U1
      T1=X(I(1))+C71*T1
      U1=Y(I(1))+C71*U1
      T2=C72*(R1-R5)
      U2=C73*(S1-S5)
      T3=C73*(R5-R3)
      U3=C73*(S5-S3)
      T4=C74*(R3-R1)
      U4=C74*(S3-S1)
      R1=T1+T2+T3
      R3=T1-T2-T4
      R5=T1-T3+T4
      S1=U1+U2+U3
      S3=U1-U2-U4
      S5=U1-U3+U4
      U1=C75*(S2+S4-S6)
      T1=C75*(R2+R4-R6)
      T2=C76*(R2+R6)
      U2=C76*(S2+S6)
      T3=C77*(R4+R6)
```

```
U3=C77*(S4+S6)
T4=C78*(R4-R2)
U4=C78*(S4-S2)
R2=T1+T2+T3
R4=T1-T2-T4
R6=T1-T3+T4
S2=U1+U2+U3
S4=U1-U2-U4
S6=U1-U3+U4
X(I(2))=R1+S2
X(I(7))=R1-S2
Y(I(2))=S1-R2
Y(I(7))=S1+R2
X(I(3))=R3+S4
X(I(6))=R3-S4
Y(I(3))=S3-R4
Y(I(6))=S3+R4
X(I(4))=R5-S6
X(I(5))=R5+S6
Y(I(5))=S5-R6
GO TO 20
END
```