Mixed-radix Algorithm for the Computation of Forward and Inverse MDCT

Jiasong Wu [1 2 3 *], **Huazhong Shu** [2 3 *], **Lotfi Senhadji** [1 2 *], **Limin Luo** [2 3 *]

[1] *LTSI, Laboratoire Traitement du Signal et de l'Image INSERM : U642 , Université de Rennes I , Campus de Beaulieu, 263 Avenue du Général Leclerc - CS 74205 - 35042 Rennes Cedex,FR*

[2] *CRIBS, Centre de Recherche en Information Biomédicale sino-français INSERM : Laboratoire International Associé , Université de Rennes I , SouthEast University , Nankin,CN*

[3] *LIST, Laboratory of Image Science and Technology SouthEast University , Si Pai Lou 2, Nanjing, 210096,CN*

* Correspondence should be adressed to: Jiasong Wu <jwu@seu.edu.cn >
* Correspondence should be adressed to: Huazhong Shu <shu.list@seu.edu.cn >
* Correspondence should be adressed to: Lotfi Senhadji <lotfi.senhadji@univ-rennes1.fr >
* Correspondence should be adressed to: Limin Luo <luo.list@seu.edu.cn >

Abstract

The modified discrete cosine transform (MDCT) and inverse MDCT (IMDCT) are two of the most computational intensive operations in MPEG audio coding standards. A new mixed-radix algorithm for efficient computing the MDCT/IMDCT is presented. The proposed mixed-radix MDCT algorithm is composed of two recursive algorithms. The first algorithm, called the radix-2 decimation in frequency (DIF) algorithm, is obtained by decomposing an N -point MDCT into two MDCTs with the length N /2. The second algorithm, called the radix-3 decimation in time (DIT) algorithm, is obtained by decomposing an N -point MDCT into three MDCTs with the length N /3. Since the proposed MDCT algorithm is also expressed in the form of a simple sparse matrix factorization, the corresponding IMDCT algorithm can be easily derived by simply transposing the matrix factorization. Comparison of the proposed algorithm with some existing ones shows that our proposed algorithm is more suitable for parallel implementation and especially suitable for the layer III of MPEG-1 and MPEG-2 audio encoding and decoding. Moreover, the proposed algorithm can be easily extended to the multidimensional case by using the vector-radix method.

Author Keywords
Index Terms
MDCT ; fast algorithm ; mixed radix ; MPEG ; audio coding

Introduction

The modified discrete cosine transform (MDCT) and inverse MDCT (IMDCT) are extensively used to realize the analysis/synthesis filter banks of time domain aliasing cancellation scheme for subband coding [1], [2]. This filter bank is equivalent to the modulated lapped transform (MLT) introduced by Malvar [3]. The MDCT/IMDCT has been adopted in several international standards and commercial audio coding products such as MPEG-1 [4], MPEG-2 [5], and AC-3 [6] to achieve high quality audio compression. However, the direct computation of the MDCT in MPEG audio encoding and IMDCT in MPEG audio decoding involves an extensive number of arithmetic operations. Therefore, efficient algorithms for their computation are of great importance.

In the past, many fast algorithms have been reported in the literature for computing the MDCT and IMDCT. These algorithms can generally be categorized into two kinds: direct method and indirect method. The term of indirect method means that the MDCT or IMDCT is first converted into other unitary transforms such as discrete Fourier transform (DFT) or discrete cosine transform (DCT), and the latter transforms are then calculated by fast algorithm. These algorithms generally lead to the parallel in and parallel out architecture [7]. This is the most widely used technique for efficient implementation of both MDCT and IMDCT. For example, by decomposing the MDCT kernel and using the symmetry property of cosine function, Iwadare et al. [8] presented an efficient MDCT algorithm that is composed of pre-processing (data shifts, differential calculation and complex pre-multiplication), an N /2-point FFT followed by complex post-multiplications. Fan et al. [9] developed two IMDCT algorithms based respectively on DCT and on the fast Hartley transform for performing the IMDCT quickly. Britanak and Rao [10], [11] proposed an efficient approach for implementing the MDCT and IMDCT based on the N /4-point DCT/DST and corresponding N /4-point IDCT/IDST. Lee [12] then suggested an improvement of this algorithm in the computational speed. Jing and Tai [13] derived a new fast MLT algorithm which first converts an N -point MLT into the N/ 2-point DCT-IV by using Malvar's algorithm [3], and the latter transform is then calculated via N/ 4-point complex-valued FFT with data shuffling. By using a matrix representation, Cheng and Hsu [14] presented a systematic method for realizing the MDCT and IMDCT. A fast algorithm based on the DCT for computing the MDCT and IMDCT was presented by Truong et al. [15]. Shao and Johnson [16] recently derived a new fast algorithm for computing the MDCT and IMDCT based on a modified split-radix FFT algorithm reported in [17]. A comprehensive list of references on this subject can be found in [11], [18]. A notable merit of indirect method is that many mature algorithms and implemented architectures can be used to the fast computation and effective implementation of DFT (for example, [17], [19 –22]) and DCT (e.g., [23 –28]). However, a common drawback of the FFT-based method is the need for complex arithmetic and storage of complex values. The disadvantage of the DCT-based method is generally the introduction of recursive structure, which is not suitable

for parallel implementation [29]. The direct method for efficient calculating the MDCT and IMDCT is mainly based on the use of a regressive formula. Among this kind of methods, Chiang and Liu [30] proposed a regressive algorithm, which can be implemented by parallel VLSI filters. This algorithm was further improved by Chen et al. [31] and Nikolajevic and Fettweis [32]. These regressive algorithms in general emphasize on the merits of serial in and serial out structures [7]. More recently, Shu et al. [33] presented a radix-3 decimation in frequency (DIF) algorithm for the fast computing the MDCT and IMDCT. In their algorithm, an N -point MDCT and IMDCT was realized via the computation of three N /3-point MDCTs and IMDCTs, respectively. Their algorithm also belongs to the direct method, but seems to be more similar to the DCT-based algorithm ([10], [12], [15]).

In this paper, we propose a new mixed-radix algorithm to compute the MDCT/IMDCT, which is composed of two recursive algorithms. The first algorithm, called the radix-2 DIF algorithm, decomposes an N -point MDCT into two N /2-point MDCTs. This algorithm is inspired by a research work presented in [23] where an N -point DCT is decomposed into two DCTs of length N /2. The second algorithm, called the radix-3 decimation in time (DIT) algorithm, is different from Shu's DIF algorithm [33], and decomposes an N -point MDCT into three N /3-point MDCTs. We then combine the radix-3 MDCT algorithm and radix-2 MDCT algorithm to produce the mixed-radix MDCT algorithm, which is similar to Chan's mixed-radix DCT algorithm [24]. The mixed-radix algorithm for computing the IMDCT can be easily derived from that of MDCT.

The paper is organized as follows. In section II, we introduce the definitions and some properties of the MDCT and IMDCT. Section III describes the radix-2 algorithm for the efficient computation of MDCT/IMDCT. Section IV presents the radix-3 algorithm for calculating the MDCT/IMDCT. The analysis of the computational complexity and comparison of the proposed algorithm with some existing ones are given in section V. Section VI concludes the work.

Definitions and some properties of the MDCT and IMDCT

Let {x (n)} be an input data sequence, the MDCT and IMDCT are respectively defined as [2]

$$X(k) = \sum_{n=0}^{N-1} x(n) \cos\left[\frac{\pi}{2N}(2n+1+\frac{N}{2})(2k+1)\right], \quad k = 0, 1, \ \ldots, N/2-1,$$

$$\hat{x}(n) = \sum_{k=0}^{N/2-1} X(k) \cos\left[\frac{\pi}{2N}(2n+1+\frac{N}{2})(2k+1)\right], \quad n = 0, 1, \ \ldots, N-1.$$

where N is the window length. In general, the recovered data sequence {x̂ (n)} does not correspond to the original data sequence {x (n) }. {x̂ (n)} has the following symmetries [14]

$$\hat{x}(3N/4+n) = \hat{x}(3N/4-1-n), \quad n = 0, 1, \ \ldots, N/4-1.$$
$$\hat{x}(N/2-1-n) = -\hat{x}(n),$$

Therefore, only x̂ (n), for n = 0, 1, …, N /4−1 and n = N /2, N /2+1, …, 3N /4−1, need to be calculated.

Let kind of methods, Chiang

$$X = [X(0), X(1), \ \cdots, X(N/2-1)]^T,$$
$$x = [x(0), x(1), \ \cdots, x(N-1)]^T,$$
$$\hat{x} = [\hat{x}(0), \hat{x}(1), \ \cdots, \hat{x}(N-1)]^T,$$

where T denotes the transposition. Then Eqs. (1) and (2) can be written as

$$X = M_{(N/2)\times N} x,$$
$$\hat{x} = M^T_{(N/2)\times N} X,$$

where $M_{(N/2)\times N}$ is an (N /2) × N MDCT matrix.

From (7) and (8), we know that if a realization of the MDCT is developed, then a realization for the IMDCT can be obtained by transposing the signal flow graph of the MDCT.

Radix-2 algorithm for THE MDCT/IMDCT computation

In this section, we derive a new radix-2 DIF MDCT algorithm, which is obtained by decomposing the N -point MDCT into two MDCTs with the length N /2, and we get its sparse matrix factorization. Then, the corresponding IMDCT algorithm can be easily derived by transposing the MDCT matrix factorization. In the remaining part of this section, the window length N is assumed to be divisible by 4, i.e., N = 4p .

Let us consider the following two sub-sequences

$$A(k) = X(2k) + X(2k+1),\ k = 0,\ 1,\ \ldots,\ N/4 - 1,$$
$$B(k) = X(2k) - X(2k+1),\ k = 0,\ 1,\ \ldots,\ N/4 - 1.$$

Computation of A(k)

$$A(k) = \sum_{n=0}^{N-1} x(n)\left\{\cos\left[\frac{\pi}{2N}(2n+1+\frac{N}{2})(4k+1)\right] + \cos\left[\frac{\pi}{2N}(2n+1+\frac{N}{2})(4k+3)\right]\right\}$$
$$= \sum_{n=0}^{N-1} 2x(n)\cos\left(\frac{\pi}{4}+\theta_n\right) \times \cos\left[\frac{\pi}{N}(2n+1+\frac{N}{4})(2k+1) + \frac{\pi}{4}(2k+1)\right]$$
$$= 2\cos\left(\frac{\pi}{4}+\frac{k\pi}{2}\right)A_1(k) - 2\sin\left(\frac{\pi}{4}+\frac{k\pi}{2}\right)A_2(k),$$

where

$$A_1(k) = \sum_{n=0}^{N-1} x(n)\cos\left(\frac{\pi}{4}+\theta_n\right)\cos\varphi_{n,k},$$
$$A_2(k) = \sum_{n=0}^{N-1} x(n)\cos\left(\frac{\pi}{4}+\theta_n\right)\sin\varphi_{n,k},$$
$$\theta_n = \frac{\pi}{2N}(2n+1),$$
$$\varphi_{n,k} = \frac{\pi}{N}(2n+1+\frac{N}{4})(2k+1).$$

Using the trigonometric identity

$$\cos\left(\frac{\pi}{4}+\frac{k\pi}{2}\right) = (-1)^k \sin\left(\frac{\pi}{4}+\frac{k\pi}{2}\right)$$
$$= (-1)^{\lfloor (k+1)/2 \rfloor}\frac{\sqrt{2}}{2},$$

where $\lfloor x \rfloor$ denotes the lower integer part of x . Eq. (11) becomes

$$A(k) = \sqrt{2}(-1)^{\lfloor (k+1)/2 \rfloor}[A_1(k) - (-1)^k A_2(k)].$$

For the computation of $A_1(k)$, we further decompose Eq. (12) into the form

$$A_1(k) = \sum_{n=0}^{N/2-1} x(n)\cos\left(\frac{\pi}{4}+\theta_n\right)\cos\varphi_{n,k} + \sum_{n=N/2}^{N-1} x(n)\cos\left(\frac{\pi}{4}+\theta_n\right)\cos\varphi_{n,k}$$
$$= \sum_{n=0}^{N/2-1} x(n)\cos\left(\frac{\pi}{4}+\theta_n\right)\cos\varphi_{n,k} - \sum_{n=0}^{N/2-1} x(N/2+n)\cos\left(\frac{3\pi}{4}+\theta_n\right)\cos\varphi_{n,k}$$
$$= \frac{1}{\sqrt{2}}\sum_{n=0}^{N/2-1} a_1(n)\cos\varphi_{n,k},$$

where

$$a_1(n) = [x(n) + x(N/2+n)]\cos\theta_n - [x(n) - x(N/2+n)]\sin\theta_n,\ \text{for } n = 0,\ 1,\ \ldots,\ N/2-1$$

Similarly, Eq. (13) can be rewritten as

$$A_2(k) = \sum_{n=0}^{N/2-1} x(n)\cos\left(\frac{\pi}{4}+\theta_n\right)\sin\varphi_{n,k} + \sum_{n=N/2}^{N-1} x(n)\cos\left(\frac{\pi}{4}+\theta_n\right)\sin\varphi_{n,k}$$

$$-n)\cos\left(\frac{3\pi}{4}-\theta_n\right)\times\sin\left[\frac{3\pi}{2}(2k+1)-\varphi_{n,k}\right] + \sum_{n=0}^{N/2-1} x(N-1-n)\cos\left(\frac{5\pi}{4}-\theta_n\right)\times\sin$$

$$= \frac{(-1)^k}{\sqrt{2}} \sum_{n=0}^{N/2-1} a_2(n)\cos_{n,k},$$

where

$$/2-1-n)-x(N-1-n)]\cos\theta_n - [x(N/2-1-n)+x(N-1-n)]\sin\theta_n, \quad \text{for } n=0,1,$$

From (19) and (21), it can be easily verified that

$$a_2(n) = -a_1(N/2-1-n), \quad \text{for } n=0,1,\ \ldots, N/2-1.$$

Substituting (18) and (20) into (17) and using (22), we get

$$A(k) = (-1)^{\lfloor (k+1)/2 \rfloor} \sum_{n=0}^{N/2-1} a(n)\cos\varphi_{n,k},\ k=0,1,\ \ldots, N/4-1,$$

where

$$a(n) = a_1(n) + a_1(N/2-1-n)$$
$$= a'(n)\cos\theta_n + a'(N/2-1-n)\sin\theta_n,$$

with

$$a'(n) = x(n) - x(N/2-1-n) + x(N/2+n) + x(N-1-n).$$

Eq. (23) shows that A (k) is the MDCT of sequence a (n) whose length is N /2. Moreover, the sequence a (n) possesses the even symmetry property, that is

$$a(n) = a(N/2-1-n), \quad n=0,1,\ \ldots, N/4-1.$$

Computation of B(k)

From (10), we have

$$B(k) = \sum_{n=0}^{N-1} x(n)\left\{\cos\left[\frac{\pi}{2N}(2n+1+\frac{N}{2})(4k+1)\right] - \cos\left[\frac{\pi}{2N}(2n+1+\frac{N}{2})(4k+3)\right]\right\}$$
$$= 2\sin\left(\frac{\pi}{4}+\frac{k\pi}{2}\right)B_1(k) + 2\cos\left(\frac{\pi}{4}+\frac{k\pi}{2}\right)B_2(k)$$
$$= \sqrt{2}(-1)^{\lfloor k/2 \rfloor}[B_1(k) + (-1)^k B_2(k)],$$

where

$$B_1(k) = \sum_{n=0}^{N-1} x(n)\sin\left(\frac{\pi}{4}+\theta_n\right)\cos\varphi_{n,k}$$
$$B_2(k) = \sum_{n=0}^{N-1} x(n)\sin\left(\frac{\pi}{4}+\theta_n\right)\sin\varphi_{n,k}$$

Proceeding with the computation of B (k) in a similar way as for A (k), we have

$$B(k) = (-1)^{\lfloor k/2 \rfloor} \sum_{n=0}^{N/2-1} b(n)\cos\varphi_{n,k},\ k=0,1,\ \ldots, N/4-1,$$

where

$$b(n) = -a'(N/2-1-n)\cos\theta_n + a'(n)\sin\theta_n.$$

Here a ′(n) is given by Eq. (25) .

Eq. (30) shows that B (k) is the MDCT of the sequence b (n) with the length N /2, and (31) shows that b (n) possesses the even antisymmetry property

$$b(n) = -b(N/2-1-n), \quad n = 0, 1, \ \ldots, N/4-1.$$

Computation of X(k), k = 0, 1, …, N/2− 1

From (23) and (30), and using the relationship $(-1)^{\lfloor (k+1)/2 \rfloor} = (-1)^{k+\lfloor k/2 \rfloor}$, we have

$$X(2k) = \frac{1}{2}[A(k) + B(k)]$$
$$= (-1)^{\lfloor (k+1)/2 \rfloor} \sum_{n=0}^{N/2-1} x_k(n)\cos\varphi_{n,k},$$

where

$$x_k(n) = \frac{1}{2}[a(n) + (-1)^k b(n)].$$

Using (24), (25) and (31), we have

$$x_k(n) = \begin{cases} x'(n) = x^{(1)}(n)\cos\theta_n + x^{(2)}(n)\sin\theta_n, & \text{if } k \text{ is even}, \\ x''(n) = -x^{(1)}(n)\sin\theta_n + x^{(2)}(n)\cos\theta_n, & \text{if } k \text{ is odd}, \end{cases} \quad \text{for } n = 0, 1, \ \ldots, N/2-1,$$

where

$$x^{(1)}(n) = x(n) - x(N/2-1-n) = -x^{(1)}(N/2-1-n),$$
$$x^{(2)}(n) = x(N/2+n) + x(N-1-n) = x^{(2)}(N/2-1-n).$$

It can be easily deduced from (36) that

$$x''(n) = x'(N/2-1-n), \quad \text{for } n = 0, 1, \ \ldots, N/2-1.$$

The computation of X (2k +1) can be realized in a similar way, and we have

$$X(2k+1) = (-1)^{\lfloor (k+1)/2 \rfloor} \sum_{n=0}^{N/2-1} x_k(N/2-1-n)\cos\varphi_{n,k}.$$

Eqs. (33) and (38) show that the computation of N -point MDCT can be realized via the calculation of two N /2-point MDCTs. Note that we can perform the decomposition in (33) and (38) recursively until the required small-length MDCTs, i.e., 4- or 6-point MDCT, is reached. Fig. 1 shows the flowgraph for computing the length-8 MDCT.

To make the proposed radix-2 MDCT algorithm more clear for the readers, we get the MDCT sparse matrix factorization in the following. Based on the proposed radix-2 MDCT algorithm, the matrix $M_{(N/2)\times N}$ in (7) can be decomposed into the following sparse matrix product

$$M_{(N/2)\times N} = D_{N/2} Q_{N/2} \begin{bmatrix} M_{(N/4)\times(N/2)} & \\ & M_{(N/4)\times(N/2)} J_{N/2} \end{bmatrix} \times G_N \begin{bmatrix} I_{N/4} & -J_{N/4} & & \\ -J_{N/4} & I_{N/4} & & \\ & & I_{N/4} & J_{N/4} \\ & & J_{N/4} & I_{N/4} \end{bmatrix},$$

where I_N is the identity matrix and J_N is the reverse identity matrix. $M_{(N/4)\times(N/2)} J_{N/2}$ denotes (N /4)×(N /2) MDCT matrix with reversed order of its columns. G_N is the rotation matrix given by

$$G_N = \begin{bmatrix} \cos\frac{\pi}{2N} & & & & & & \sin\frac{\pi}{2N} \\ & \cos\frac{3\pi}{2N} & & & & \sin\frac{3\pi}{2N} & \\ & & \ddots & & \iddots & & \\ & & & \cos\frac{(N-1)\pi}{2N} & \sin\frac{(N-1)\pi}{2N} & & \\ & & & -\sin\frac{(N-1)\pi}{2N} & \cos\frac{(N-1)\pi}{2N} & & \\ & & \iddots & & \ddots & & \\ & -\sin\frac{3\pi}{2N} & & & & \cos\frac{3\pi}{2N} & \\ -\sin\frac{\pi}{2N} & & & & & & \cos\frac{\pi}{2N} \end{bmatrix}$$

$Q_{N/2}$ is the permutation matrix. For clarity, $Q_{N/2}$ for N = 16 is shown

$$Q_8 = \begin{bmatrix} 1 & & & & & & & \\ & & & 1 & & & & \\ & 1 & & & & & & \\ & & & & 1 & & & \\ & & 1 & & & & & \\ & & & & & 1 & & \\ & & & 1 & & & & \\ & & & & & & & 1 \end{bmatrix},$$

$D_{N/2}$ is the diagonal sign-changing matrix

$$D_{N/2} = \begin{bmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & -1 & & & & & \\ & & & -1 & & & & \\ & & & & -1 & & & \\ & & & & & -1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 & \ddots \end{bmatrix}.$$

Substituting (39) into (8), we can easily get the corresponding IMDCT algorithm, which is in fact the radix-2 DIT IMDCT algorithm. The realization of IMDCT can be easily obtained by reversing the flowgraphs for the MDCT computation.

Radix-3 algorithm for the MDCT/IMDCT computation

In this section, we describe a radix-3 DIT MDCT algorithm which is obtained by decomposing the N -point MDCT into three MDCTs with the length N /3, and get its sparse matrix factorization. Then, the corresponding IMDCT algorithm is easily obtained by transposing the MDCT matrix factorization. The window length N is further assumed to be divisible by 12, i.e., N = 12p .

Eq. (1) can be decomposed as follows

$$\frac{\pi}{2N}(6n+3+\frac{N}{2})(2k+1)\Big]+\sum_{n=0}^{N/3-1}x(3n)\cos\Big[\frac{\pi}{2N}(6n+1+\frac{N}{2})(2k+1)\Big]+\sum_{n=0}^{N/3-1}x(3n+2)\,c$$

$$\phi_{n,k}+\sum_{n=0}^{N/3-1}x(3n)(\cos\alpha_k\cos\phi_{n,k}+\sin\alpha_k\sin\phi_{n,k})+\sum_{n=0}^{N/3-1}x(3n+2)(\cos\alpha_k\cos\phi$$

$$=C(k)+\cos\alpha_k D(k)+\sin\alpha_k E(k),\quad k=0,1,\cdots,N/2-1,$$

where

$$C(k)=\sum_{n=0}^{N/3-1}x(3n+1)\cos\varphi_{n,k},$$

$$D(k)=\sum_{n=0}^{N/3-1}[x(3n)+x(3n+2)]\cos\phi_{n,k},$$

$$E(k)=\sum_{n=0}^{N/3-1}[x(3n)-x(3n+2)]\sin\phi_{n,k},$$

$$\alpha_k=\frac{\pi}{N}(2k+1),$$

$$\phi_{n,k}=\frac{3\pi}{2N}(2n+1+\frac{N}{6})(2k+1).$$

Note that for N = 12 p , Eq. (46) can be rewritten as follows

$$E(\frac{N}{6}-1-k)=\sum_{n=0}^{N/3-1}(-1)^{n+\frac{N}{12}}[x(3n)-x(3n+2)]\cos\phi_{n,k}.$$

Furthermore

$$C(N/3-1-k)=C(N/3+k)=-C(k),$$

$$D(N/3-1-k)=D(N/3+k)=-D(k),$$

$$E(N/3-1-k)=-E(N/3+k)=E(k).$$

Eq. (43) can be computed as follows

$$X(k)=C(k)+\cos\alpha_k D(k)+\sin\alpha_k E(k),$$

$$X(\frac{N}{3}-1-k)=-C(k)+\frac{1}{2}[\cos\alpha_k D(k)+\sin\alpha_k E(k)]-\frac{\sqrt{3}}{2}[\sin\alpha_k D(k)-\cos\alpha_k E(k)],$$

$$X(\frac{N}{3}+k)=-C(k)+\frac{1}{2}[\cos\alpha_k D(k)+\sin\alpha_k E(k)]+\frac{\sqrt{3}}{2}[\sin\alpha_k D(k)-\cos\alpha_k E(k)],$$

where the range of index k in (50)–(53) is from 0 to N /6−1.

Eqs. (44) , (45) and (49) show that the computation of N -point MDCT can be realized via the calculation of three N /3-point MDCTs. It should be noted that the above radix-3 DIT MDCT algorithm is also recursive; furthermore, it is more efficient than the radix-3 DIF algorithm proposed in [33]. Fig. 2 shows the flowgraph for computing the length-12 MDCT by using the proposed radix-3 DIT algorithm.

Based on the proposed radix-3 MDCT algorithm, the matrix $M_{(N/2)\times N}$ in (7) can be decomposed into the following sparse matrix product

$$M_{(N/2)\times N}=E_{N/2}F_{N/2}\begin{bmatrix}M_{(N/6)\times(N/3)} & & \\ & M_{(N/6)\times(N/3)} & \\ & & J_{N/6}M_{(N/6)\times(N/3)}\end{bmatrix}P_N,$$

where $J_{N/6}M_{(N/6)\times(N/3)}$ denotes (N /6)×(N /3) MDCT matrix with reversed order of its rows. P_N corresponds to the input data permutation of (45) and (49). $E_{N/2}F_{N/2}$ corresponds to the permutation of (51), (52) and (53) by obtaining X (k), k = 0, 1, …, N /2−1, from C (k), D (k) and E (k), k = 0, 1, …, N /6−1. Let N = 12 for example,

$$P_{12}=\begin{bmatrix} 1 & & & & & & & & & & & \\ & 1 & & & & & & & & & & \\ & & 1 & & & & & & & & & \\ & & & 1 & & & & & & & & \\ & & & & 1 & & & & & & & \\ & & & & & 1 & & & & & & \\ 1 & & 1 & & & & & & & & & \\ & 1 & & 1 & & & & & & & & \\ & & 1 & & 1 & & & & & & & \\ & & & 1 & & 1 & & & & & & \\ & & & & 1 & & 1 & & & & & \\ & & & & & 1 & & 1 & & & & \\ -1 & & 1 & & & & & & & & & \\ & 1 & & -1 & & & & & & & & \\ & & 1 & & -1 & & & & & & & \\ & & & 1 & & -1 & & & & & & \\ & & & & 1 & & -1 & & & & & \\ & & & & & 1 & & -1 & & & & \\ & & & & & & 1 & & -1 & & & \\ & & & & & & & 1 & & -1 & & \\ & & & & & & & & 1 & & -1 & \\ & & & & & & & & & 1 & & -1 \end{bmatrix},$$

$$F_6=\begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & \cos\frac{\pi}{N} & & & \sin\frac{\pi}{N} \\ & & & \cos\frac{3\pi}{N} & \sin\frac{3\pi}{N} & \\ & & & \frac{\sqrt{3}}{2}\sin\frac{3\pi}{N} & -\frac{\sqrt{3}}{2}\cos\frac{3\pi}{N} & \\ & & \frac{\sqrt{3}}{2}\sin\frac{\pi}{N} & & & -\frac{\sqrt{3}}{2}\cos\frac{\pi}{N} \end{bmatrix},$$

$$E_6=\begin{bmatrix} 1 & & 1 & & & \\ & 1 & & 1 & & \\ & -1 & & 1/2 & -1 & \\ -1 & & 1/2 & & & -1 \\ -1 & & 1/2 & & & 1 \\ & -1 & & 1/2 & 1 & \end{bmatrix}.$$

Substituting (54) into (8), we can easily obtain the corresponding IMDCT algorithm, which is in fact the radix-3 DIF IMDCT algorithm. The realization of IMDCT can be easily obtained by reversing the flowgraphs for the MDCT computation.

Complexity analysis and comparison results

In this section, we consider the computational complexity of the proposed MDCT and IMDCT algorithms and compare them with some known algorithms.

Computational complexity for the radix-2 MDCT/IMDCT algorithm

- The computation of $x^{(1)}(n)$ and $x^{(2)}(n)$ defined by Eq. (36) , for n = 0, 1, …, N /4−1, requires N /2 additions;

- The symmetry property given by (37) shows that in order to obtain the values of $x_k(n)$ defined by (35), we only need to calculate $x'(n)$ and $x''(n)$ for n = 0, 1, …, N /4−1. Therefore, Eq. (35) can be expressed as follows:

$$\begin{bmatrix} x'(n) \\ x''(n) \end{bmatrix}=\begin{bmatrix} \cos\theta_n & \sin\theta_n \\ -\sin\theta_n & \cos\theta_n \end{bmatrix}\begin{bmatrix} x^{(1)}(n) \\ x^{(2)}(n) \end{bmatrix},\ n=0,1,\ \ldots,N/4-1$$

which generally needs 4 multiplications and 2 additions for each n . However, (58) can be rearranged as follows [34]:

$$\begin{bmatrix} x'(n) \\ x''(n) \end{bmatrix} = \begin{bmatrix} 1 & \tan(\theta_n/2) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\sin\theta_n & 1 \end{bmatrix} \times \begin{bmatrix} 1 & \tan(\theta_n/2) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x^{(1)}(n) \\ x^{(2)}(n) \end{bmatrix} \text{ for } n = 0, 1, \ldots, N/4 - 1,$$

which needs 3 multiplications and 3 additions for each n . Therefore, the computation of (35) needs 3N /4 multiplications and 3N /4 additions.

The computational complexity of the radix-2 MDCT algorithm is therefore given by

$$M_N^{\mathrm{MDCT}} = 2M_{N/2}^{\mathrm{MDCT}} + 3N/4,$$
$$A_N^{\mathrm{MDCT}} = 2A_{N/2}^{\mathrm{MDCT}} + 5N/4.$$

One may use (60) recursively until the 4-point or 6-point MDCT is reached. For the special case of N = 2^n, n ≥ 3, radix-2 MDCT algorithm requires 3(n −1)N /4 multiplications and 5(n −1)N /4 additions with $M_4^{\mathrm{MDCT}} = 3$ and $A_4^{\mathrm{MDCT}} = 5$. For N =3×2^n, n ≥ 3, the algorithm requires 3(n −7/9)N /4 multiplications and 5(n −1/5)N /4 additions with $M_6^{\mathrm{MDCT}} = 1$ and $A_6^{\mathrm{MDCT}} = 6$ (See appendix A).

Similarly, the computational complexity of the radix-2 IMDCT algorithm is given by

$$M_N^{\mathrm{IMDCT}} = 2M_{N/2}^{\mathrm{IMDCT}} + 3N/4,$$
$$A_N^{\mathrm{IMDCT}} = 2A_{N/2}^{\mathrm{IMDCT}} + 5N/4.$$

For the special case of N = 2^n, n ≥ 3, radix-2 IMDCT algorithm requires 3(n −1)N /4 multiplications and (5n −7)N /4 additions with $M_4^{\mathrm{IMDCT}} = 3$ and $A_4^{\mathrm{IMDCT}} = 3$. For N = 3×2^n, n ≥ 3, the algorithm requires 3(n −7/9)N /4 multiplications and 5(n −7/15)N /4 additions with $M_6^{\mathrm{IMDCT}} = 1$ and $A_6^{\mathrm{IMDCT}} = 4$ (See appendix B).

Computational complexity for the radix-3 MDCT/IMDCT algorithm

- The computation of the input data of D (k) and E (N/ 6−1 − k) in Eqs. (45) and (49) requires 2N /3 additions;
- The computation of X (k), X (N /3−1 − k) and X (N /3+k) in (51)–(53) requires 2N /3 multiplications and N additions; However, when k = (N /4−1)/2, we have $\theta_k = \pi/4$, $\cos\theta_k = \sin\theta_k = \sqrt{2}/2$. In such case, 2 multiplications can be saved.

The computational complexity of the radix-3 MDCT algorithm is given in both recursive and non-recursive forms as follows

$$M_N^{\mathrm{MDCT}} = 3M_{N/3}^{\mathrm{MDCT}} + 2N/3 - 2 = 2mN/3 + N/2 + 1,$$
$$A_N^{\mathrm{MDCT}} = 3A_{N/3}^{\mathrm{MDCT}} + 5N/3 = 5mN/3 + 5N/4,\ N = 4\times 3^m,\ m \geq 1,$$

with $M_4^{\mathrm{MDCT}} = 3$ and $A_4^{\mathrm{MDCT}} = 5$.

For the computation of IMDCT, by using the symmetries presented in (3), we can further save N /3 additions in the post-processing of IMDCT algorithm, corresponding to the pre-processing of MDCT algorithm (see Fig. 2). The computational complexity of the radix-3 IMDCT algorithm is therefore given by

$$M_N^{\mathrm{IMDCT}} = 3M_{N/3}^{\mathrm{IMDCT}} + 2N/3 - 2 = 2mN/3 + N/2 + 1,$$
$$A_N^{\mathrm{IMDCT}} = 3A_{N/3}^{\mathrm{IMDCT}} + 4N/3 = 4mN/3 + 3N/4,\ N = 4\times 3^m,\ m \geq 1.$$

with $M_4^{\mathrm{IMDCT}} = 3$ and $A_4^{\mathrm{IMDCT}} = 3$.

Mixed-radix MDCT/IMDCT algorithm and comparison with some existing fast algorithms

Since the radix-3 MDCT/IMDCT algorithm is relatively more efficient than the radix-2 MDCT/IMDCT algorithm, therefore, for N = $3^m \times 2^n$, m ≥ 2, n ≥ 2, we first use the radix-3 algorithm until N_1 = 3×2^n, n ≥ 2, which is then computed by the radix-2 algorithm. To make the proposed mixed-radix algorithm more clear, we give the 12-point and 36-point MDCT flowgraphs in Fig. 3 and Fig. 4 , respectively.

We first consider the case where the length of the sequences is $N = 2^n$, $n \geq 2$. Table I lists the computational complexity of the radix-2 MDCT/IMDCT algorithm and that of the algorithms presented in [9], [11 –13] and [15]. It can be observed from this Table that the proposed algorithm for computing the MDCT and IMDCT require more number of arithmetic operations than the algorithms presented in [11 –13] and [15]. However, our proposed algorithm uses real arithmetic only compared to Jing's algorithm [13]. The radix-2 IMDCT algorithm is more efficient than the second algorithm but less efficient than the first one presented in [9]. In [9], the authors showed the flowgraphs of two algorithms for $N = 16$. But for higher value of N , the generalized flowgraph is difficult to obtain. Furthermore, compared to the algorithms presented in [12] and [15], our algorithm as well as those reported in [9], [11] and [13] do not introduce the recursive structure as mentioned in [29], which will be discussed in detail in the following.

To test the performance of the proposed mixed-radix MDCT/IMDCT algorithm for $N = 3^m \times 2^n$, $m \geq 1$, $n \geq 2$, we compare it with Jing 's algorithm [13]/Fan's algorithm [9] for which the zero-padding is included, and Lee's algorithm [12] for which Bi's algorithm presented in [25] is used to compute the scaled DCT (SDCT). Table II lists the number of arithmetic operations needed by these algorithms for computing the MDCT and IMDCT of length $N = 3^m \times 2^n$, $m \geq 1$, $n \geq 2$ (N is less than 500). It can be seen from the Table that, in most cases, the mixed-radix MDCT/IMDCT algorithm is more efficient than Jing's algorithm [13] for MDCT and Fan's algorithm [9] for IMDCT in terms of overall computational complexity, but less efficient than Lee's algorithm [12].

It should be pointed out that there are other important issues for designing a good algorithm besides the computational complexity. As indicated by Yun [35], considerations such as data access scheme, modularity, and regularity are also of great importance for a good algorithm. The above design criteria will affect the effectiveness of the algorithm when implementation is concerned [23]. In the following, we will compare these criteria comprehensively of our algorithm with the algorithms reported in [10], [12] and [15], which are more similar to our algorithm.

Data access scheme

Both Lee's algorithm [12] and Truong's algorithm [15] introduce the recursive structure in the course of post-processing. As noted in [29], the potential drawback of the recursive structure is that it does not support parallel processing. Britanak's algorithm [10] does not introduce this recursive structure directly. However, the main process of the algorithms presented in [10], [12] and [15] is converting the MDCT/IMDCT computation into DCT computation. To the authors' knowledge, all the radix-type DCT algorithms [23 –25] seem to introduce the recursive structure. Figs. 3 and 4 show that our algorithm does not introduce recursive structure and mainly use butterfly-style structure, which seems to be more suitable for parallel-in and parallel-out implementation and in-place computation. Just like [29], we take I/O form into consideration for comparison purpose, the result is shown in Table III .

Modularity and Regularity

The proposed algorithm is completely recursive in nature, as noted in [23], which makes it very regular and modular structure suitable for VLSI implementation. However, the algorithms presented in [10], [12] and [15] need to transform the MDCT/IMDCT to DCT (or IDCT) first, and then can be computed by recursive DCT algorithm ([23 –25]), which we call it “incompletely recursive” in Table III . Furthermore, our algorithm can easily be extended to higher dimensional MDCT/IMDCT by using the vector-radix method, e.g., 2-D MDCT/IMDCT [36], which could find its applications in image coding [37] and digital image watermarking [38].

Suitability for the layer III of MPEG-1 and MPEG-2 audio encoding and decoding

The layer III of MPEG-1 and MPEG-2 specifies two different MDCT/IMDCT block sizes: $N = 12$ (short block) and $N = 36$ (long block). Since the algorithms presented in [10], [12] and [15] are mainly proposed for MPEG audio encoding and decoding, so we study the performance of our algorithm with those algorithms for these special cases.

Arithmetic complexity

In Table III , we give the number of arithmetic operations required by the proposed algorithm and that of the algorithms presented in [10], [12] and [15] for computing the 12- and 36-point MDCTs/IMDCTs. It can be seen from this Table that for 12-point MDCT/IMDCT, the proposed algorithm has the best performance in terms of the arithmetic complexity. However, for the computation of 36-point MDCT/IMDCT, the proposed algorithm requires more number of arithmetic operations than the algorithms reported in [12] and [15], but less than [10].

Basic module

In Table III , we give the basic module of our algorithm and the algorithms presented in [10], [12], and [15]. Fig. 5 shows that the structure for implementing 6-point IMDCT module is almost identical to that of 3-point DCT module; however, the implementation of 6-point MDCT module is easier than that of 9-point DCT module. Of course, 9-point DCT module could also be implemented by 3-point DCT module by using the radix-3 DCT algorithm [24], however, this course introduces recursive structure again. Therefore, the implementation of basic module for the proposed algorithm seems to be simpler than that of the algorithms reported in [10], [12] and [15].

Module sharing

As indicated by [12], sharing module leads to reduced hardware in implementation. Lee's algorithm [12] employs the same module to realize the MDCT and IMDCT, that is, 12-point (36-point) MDCT and 12-point (36-point) IMDCT can be computed by the same 3-point (9-point) SDCT module. However, in his algorithm, 12-point MDCT/IMDCT and 36-point MDCT/IMDCT can not be shared module. As noted in [39], in the short block mode, three short blocks (N = 12) replace a long block (N = 36) so that the number of MDCT/IMDCT samples for a frame of audio samples remains unchanged regardless of the block size selection. The process of the work mode is similar to that of our algorithm which uses three 12-point MDCTs/IMDCTs to compute a 36-point MDCT/IMDCT. That is, 12-point MDCT/IMDCT and 36-point MDCT/IMDCT share the same 12-point MDCT/IMDCT module. Furthermore, one of the most important applications for the layer III of MPEG-1 and MPEG-2 is MP3 player. Generally speaking, the process of encoding, where MDCT is used, is more complicated than the process of decoding, where IMDCT is used. So, the music is often encoded previously and uploaded to the internet. And then, we download it from the internet to our MP3 player, which performs the process of decoding, using the 12- and 36-point IMDCTs only. For this important application, the utilization of MDCT and IMDCT is divided. It seems to suggest that the sharing module of 12-point MDCT/IMDCT and 36-point MDCT/IMDCT is more important than that of 12-point (36-point) MDCT and 12-point (36-point) IMDCT.

Conclusion

A mixed-radix algorithm is presented for computing the MDCT and IMDCT of a sequence with length $N = 3^m \times 2^n$, $m \geq 0$, $n \geq 2$. Compared to other existing algorithms, the main improvement achieved is to derive an efficient decomposition method which is recursive in nature and is very regular and modular. Because the recursive structure in the course of post-processing is not included in our algorithm, the in-place computation can also be implemented, which is more suitable for parallel implementation and especially suitable for the layer III of MPEG-1 and MPEG-2 audio encoding and decoding. Note also that for N = 12, the proposed radix-2 approach for computing the MDCT and IMDCT requires fewer or the same number of arithmetic operations than those of the known algorithms. Moreover, because the proposed algorithm is expressed in a simple sparse matrix form, it allows for an extension to the multidimensional case.

Ackowledgements:

The authors are grateful to the anonymous reviewers for their constructive comments and suggestions to greatly improve the quality of this work and the clarity of the presentation.

This work was supported by National Basic Research Program of China under grant N^0 2003CB716102, Program for New Century Excellent Talents in University under grant N^0 NCET-04-0477, and Program for Changjiang Scholars and Innovative Research Team in University.

Appendix A
Computation of Length-6 MDCT: 1 multiplication and 6 additions

The input data sequence is$\{x_0, x_1, x_2, x_3, x_4, x_5\}$ and the output data sequence is$\{y_0, y_1, y_2\}$. First consider the length-6 MDCT matrix with the input data sequence$\{x_0, x_1, x_2, x_3, x_4, x_5\}$, the length-6 MDCT is given by

$$\begin{bmatrix} d_0 & 0 & -d_0 & -d_1 & -1 & -d_1 \\ -1 & 0 & 1 & 0 & -1 & 0 \\ d_0 & 0 & -d_0 & d_1 & -1 & d_1 \end{bmatrix}$$

where $d_0 = 1/2$ and $d_1 = \sqrt{3}/2$.

The output data sequence $\{y_0, y_1, y_2\}$ of MDCT is given by

$$a_1 = x_0 - x_2;\ a_2 = d_0 \cdot a_1 - x_4;\ a_3 = x_3 + x_5$$
$$m_1 = d_1 \cdot a_3;$$
$$y_0 = a_2 - m_1; y_1 = -a_1 - x_4; y_2 = a_2 + m_1.$$

Since the multiplication by (1/2) is simply a right-shift operation, hence, the computation of the length-6 MDCT only requires 1 multiplication and 6 additions.

Appendix B
Computation of Length-6 IMDCT: 1 multiplication and 4 additions

The input data sequence is$\{x_0, x_1, x_2\}$ and the output data sequence is $\{y_0, y_1, y_2, y_3, y_4, y_5\}$. The length-6 IMDCT is given by

$$\begin{bmatrix} d_0 & 0 & -d_0 & -d_1 & -1 & -d_1 \\ -1 & 0 & 1 & 0 & -1 & 0 \\ d_0 & 0 & -d_0 & d_1 & -1 & d_1 \end{bmatrix}^T$$

The output data sequence $\{y_0, y_1, y_2, y_3, y_4, y_5\}$ of IMDCT is given by

$$a_1 = x_0 + x_2; a_2 = x_2 - x_0$$
$$m_1 = d_0 \cdot a_1; m_2 = d_1 \cdot a_2$$
$$y_0 = m_1 - x_1; y_1 = 0; y_2 = -y_0; y_3 = m_2;$$
$$y_4 = -a_1 - x_1; y_5 = y_3.$$

Notably, the length-6 IMDCT requires 1 multiplication and 4 additions.

References:

- 1 . Princen JP , Bradley AB . Analysis/synthesis filter bank design based on time domain aliasing cancellation . IEEE Trans Acoust, Speech, Signal Process . ASSP-34 : (5) Oct 1986 ; 1153 - 1161
- 2 . Princen JP , Johnson AW , Bradley AB . Subband/transform coding using filter bank designs based on time domain aliasing cancellation . Proc. IEEE Int. Conf. Acoust., Speech, Signal Process Dallas, TX Apr. 1987 2161 - 2164
- 3 . Malvar HS . Signal Processing with lapped transforms . Artech House ; Norwood, MA 1992 ;
- 4 . “Information technology—Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s—Part 3: Audio,” ISO/IEC, IS 11172-3, (MPEG-1), 1992.
- 5 . “Information technology—Generic coding of moving pictures and associated audio—Part 3: Audio,” ISO/IEC, IS 13818-3, (MPEG-2), 1994.
- 6 . Digital audio compression (AC-3) standard, Audio Specialist Group T3/S7 . Dec 1995 ;
- 7 . Hwang YT , Lai SC . A novel MDCT/IMDCT computing kernel design . IEEE SIPS . Nov 2005 ; 526 - 531
- 8 . Iwadare M , Sugiyama A , Hazu F , Hirano A , Nishitani T . A 128 kb/s Hi-Fi audio CODEC based on adaptive transform coding with adaptive block size MDCT . IEEE J Select Areas Comm . 10 : (1) 138 - 144 Jan 1992 ;
- 9 . Fan YH , Madisetti VK , Mersereau RM . On fast algorithms for computing the inverse modified discrete cosine transform . IEEE Signal Process Lett . 6 : (3) 61 - 64 Mar 1999 ;
- 10 . Britanak V , Rao KR . An efficient implementation of the forward and inverse MDCT in MPEG audio coding . IEEE Signal Processing Lett . 8 : 48 - 51 Feb 2001 ;
- 11 . Britanak V , Rao KR . A new fast algorithm for the unified forward and inverse MDCT/MDST computation . Signal Process . 82 : (3) 433 - 459 Mar 2002 ;
- 12 . Lee SW . Improved algorithm for efficient computation of the forward and backward MDCT in MPEG audio coder . IEEE Trans Circuits Syst-II: Analog Digital Signal Processing . 48 : (10) 990 - 994 Oct 2001 ;
- 13 . Jing CY , Tai HM . Fast algorithm for computing modulated lapped transform . Electron Lett . 37 : (12) 796 - 797 Jun 2001 ;
- 14 . Cheng MH , Hsu YH . Fast IMDCT and MDCT algorithms—a matrix approach . IEEE Trans Signal Process . 51 : (1) 221 - 229 Jan 2003 ;
- 15 . Truong TK , Chen PD , Cheng TC . Fast algorithm for computing the forward and inverse MDCT in MPEG audio coding . Signal Process . 86 : (5) 1055 - 1060 May 2006 ;
- 16 . Shao X , Johnson SG . Type-IV DCT, DST, and MDCT algorithms with reduced numbers of arithmetic operations . Signal Process . 88 : (6) 1313 - 1326 Jun 2008 ;
- 17 . Johnson SG , Frigo M . A modified split-radix FFT with fewer arithmetic operations . IEEE Trans Signal Process . 55 : (1) 111 - 119 Jan 2007 ;
- 18 . Britanak V . An efficient computing of oddly stacked MDCT/MDST via evenly stacked MDCT/MDST and vice versa . Signal Process . 85 : (7) 1353 - 1374 Jul 2005 ;
- 19 . Bouguezel S , Ahmad MO , Swamy MNS . A new radix-2/8 FFT algorithm for length- $q \times 2^m$ DFTs . IEEE Trans Circuits Syst -I: Regular Papers . 51 : (9) 1723 - 1732 Sept 2004 ;
- 20 . Duhamel P , Vetterli M . Fast Fourier transforms: A tutorial review and a state of the art . Signal Process . 19 : (4) 259 - 299 Apr 1990 ;
- 21 . Meher PK . Efficient systolic implementation of DFT using a low-complexity convolution-like formulation . IEEE Trans Circuits Syst -II: Express Briefs . 53 : (8) 702 - 706 Aug 2006 ;
- 22 . Cheng C , Parhi KK . Low-cost fast VLSI algorithm for discrete Fourier transform . IEEE Trans Circuits Syst -I: Regular Papers . 54 : (4) 791 - 806 Apr 2007 ;
- 23 . Kok CW . Fast algorithm for computing discrete cosine transform . IEEE Trans Signal Process . 45 : (3) 757 - 760 Mar 1997 ;
- 24 . Chan YH , Siu WC . Mixed-radix discrete cosine transform . IEEE Trans Signal Process . 41 : (11) 3157 - 3161 Nov 1993 ;
- 25 . Bi G , Yu LW . DCT algorithms for composite sequence lengths . IEEE Trans Signal Process . 46 : (3) 554 - 562 Mar 1998 ;
- 26 . Britanak V , Yip P , Rao KR . Discrete cosine and sine transforms: general properties, fast algorithms and integer approximations . Academic Press Inc. Elsevier Science ; Amsterdam 2007 ;
- 27 . Meher PK . Unified systolic-like architecture for DCT and DST using distributed arithmetic . IEEE Trans Circuits Syst -I: Regular Papers . 53 : (12) 2656 - 2663 Dec 2006 ;
- 28 . Wahid KA , Dimitrov VS , Jullien GA . On the error-free realization of a scaled DCT algorithm and its VLSI implementation . IEEE Trans Circuits Syst -II: Express Briefs . 54 : (8) 700 - 704 Aug 2007 ;
- 29 . Tan TC , Bi G , Zeng Y , Tan HN . DCT hardware structure for sequentially presented data . Signal Process . 81 : (11) 2333 - 2342 Nov 2001 ;
- 30 . Chiang HC , Liu JC . Regressive implementations for the forward and inverse MDCT in MPEG audio coding . IEEE Signal Process Lett . 3 : 116 - 118 Apr 1996 ;
- 31 . Chen CH , Liu BD , Yang JF . Recursive architectures for realizing modified discrete cosine transform and its inverse . IEEE Trans Circuits Syst-II: Analog and Digital Signal Process . 50 : (1) 38 - 44 Jan 2003 ;
- 32 . Nikolajevic V , Fettweis G . Computation of forward and inverse MDCT using Clenshaw's recurrence formula . IEEE Trans Signal Process . 51 : 1439 - 1444 May 2003 ;
- 33 . Shu H , Bao X , Toumoulin C , Luo L . Radix-3 algorithm for the fast computation of forward and inverse MDCT . IEEE Signal Process Lett . 14 : (2) 93 - 96 Feb 2007 ;
- 34 . Krishnan T , Oraintara S . Fast and lossless implementation of the forward and inverse MDCT computation in MPEG audio coding . IEEE ISCAS . 2 : May 2002 ; II-181 - II-184
- 35 . Yun HD , Lee SU . On the fixed-point-error analysis of several fast DCT algorithms . IEEE Trans Circuits Syst Video Technol . 3 : 27 - 41 Feb 1993 ;
- 36 . Wu JS , Shu HZ , Senhadji L , Luo LM . A fast algorithm for the computation of 2-D forward and inverse MDCT . Signal Process . 88 : (6) 1436 - 1446 Jun 2008 ;
- 37 . Lashko O . Modulated lapped transform: application in image coding and effective algorithm of its realization . Proc. TCSET Slavsko, Ukraine Feb. 2002 243 - 244
- 38 . Tungala NC , Noore A . Elimination of visual artifacts in digital image watermarking . Proc 35th SSST . Mar 2003 ; 64 - 68
- 39 . Pan D . A tutorial on MPEG/audio compression . IEEE Multimedia . 2 : (2) 60 - 74 Summer 1995 ;

Fig. 1
Flowgraph of a length-8 MDCT (radix-2 DIF)

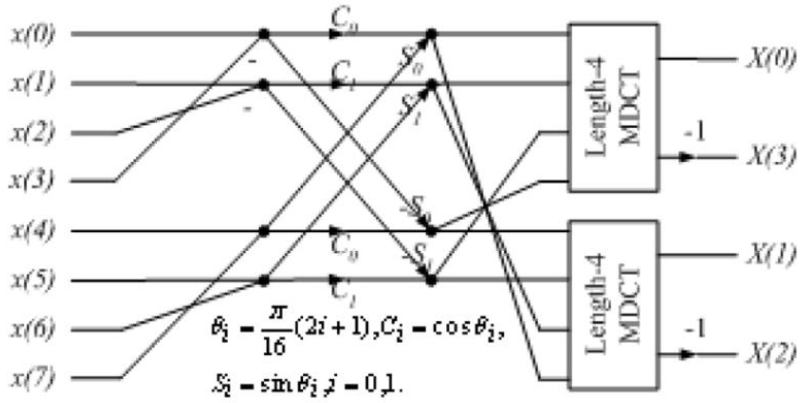

Fig. 2
Flowgraph of a length-12 MDCT (radix-3 DIT)

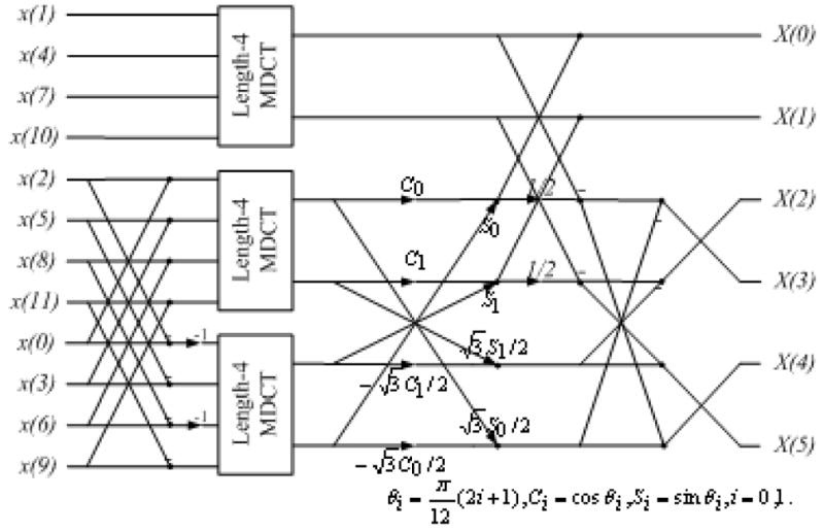

Fig. 3
Flowgraph of a length-12 MDCT (mixed-radix)

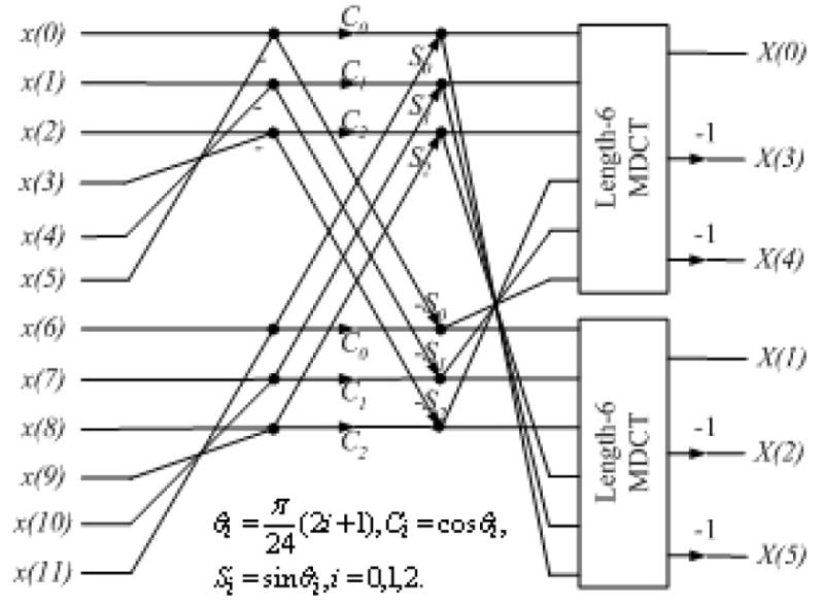

Fig. 4

Flowgraph of a length-36 MDCT (mixed-radix)

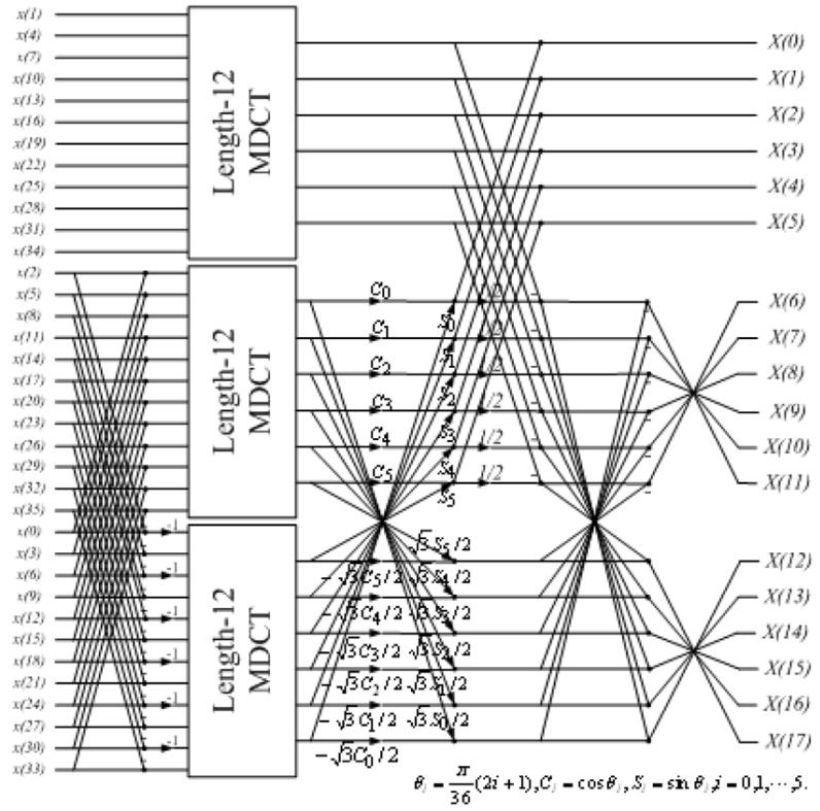

Fig. 5

Flowgraph of a length-6 MDCT (a) and a length-6 IMDCT (b). "∘" signifies that input is set to zero or output equals to zero.

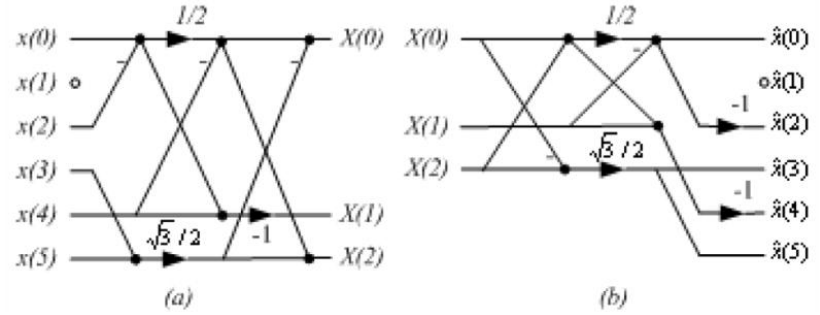

Table I

Comparison of our radix-2 MDCT/IMDCT algorithm with some existing algorithms in terms of arithmetic complexity for $N = 2^n$, $n \geq 2$. M and IM correspond to MDCT and IMDCT, respectively.

Algorithm	The number of multiplications	The number of additions	Recursive structure?
radix-2 algorithm	$3(n-1)N/4$	$5(n-1)N/4$ –M $(5n-7)N/4$ –IM	no
[15]	$(n+1)N/4$	$(3n-1)N/4$ –M $3(n-1)N/4$ –IM	yes*
[12]	$(n+1)N/4$	$(3n-1)N/4$ –M $3(n-1)N/4$ –IM	yes*
[11]	$(n+1)N/4$	$3(n+1)N/4$ –M $(3n+1)N/4$ –IM	no
[13]	$(n+1)N/4$	$(3n-1)N/4$ –M	no
[9]-I	$(n+1)N/4$	$3(n-1)N/4$ –IM	no
[9]-II	$(3n-4)N$	$(3n-2)N$ - IM	no

* denotes that the algorithm introduces recursive structure in the course of post-processing.

Table II

Comparison of our mixed-radix MDCT/IMDCT algorithm with Jing's algorithm/Fan's algorithm and Lee's algorithm in terms of arithmetic complexity for $N = 3^m \times 2^n$, $m \geq 1$, $n \geq 2$ (N is less than 500). M and IM correspond to MDCT and IMDCT, respectively.

	proposed algorithm		Jing's algorithm[13]* /Fan's algorithm[9]-I*		Lee's algorithm [12]	
N	Mul	Add (M/IM)	Mul	Add (M/IM)	Mul	Add (M/IM)
$6=3^1\times2^1$	1	6/4	8	16/12	8*	16/12*
$12=3^1\times2^2$	11	27/23	20	44/36	11	29/23
$24=3^1\times2^3$	40	84/76	48	112/96	26	72/60
$48=3^1\times2^4$	116	228/212	112	272/240	64	180/156
$96=3^1\times2^5$	304	576/544	256	640/576	152	432/384
$192=3^1\times2^6$	752	1392/1328	576	1472/1344	352	1008/912
$384=3^1\times2^7$	1792	3264/3136	1280	3328/3072	800	2304/2112
$36=3^2\times2^2$	55	141/117	112	272/240	43	133/115
$72=3^2\times2^3$	166	372/324	256	640/576	104	312/276
$144=3^2\times2^4$	442	924/828	576	1472/1344	244	732/660
$288=3^2\times2^5$	1102	2208/2016	1280	3328/3072	560	1680/1536
$108=3^3\times2^2$	235	603/495	256	640/576	195	527/473
$216=3^3\times2^3$	640	1476/1260	576	1472/1344	444	1216/1108
$432=3^3\times2^4$	1612	3492/3060	1280	3328/3072	996	2756/2540
$324=3^4\times2^2$	919	2349/1917	1280	3328/3072	771	2045/1883

* denotes that the zero-padding is included.

Table III

Comparison of our algorithm comprehensively with the algorithms presented in [10], [12] and [15]. M, IM, D, and ID correspond to MDCT, IMDCT, DCT, and IDCT, respectively.

Algorithm			**Proposed algorithm**	[15]	[12]	[10]
Data access	**I/O form**		**Parallel-in Parallel-out**	**Serial-in Serial-out**	**Serial-in Serial-out**	**Serial-in Serial-out**
Modularity Regularity	**Recursive**		**completely**	**incompletely**	**incompletely**	**incompletely**
Suitability for the layer III of MPEG-1 and MPEG-2 audio encoding and decoding	Arithmetic complexity	N =12	11/27-M 11/23-IM	11/27-M 11/23-IM	11/29-M 11/23-IM	13/39-M 13/33-IM
		N =36	55/141-M 55/117-M	43/129-M 43/115-IM	43/133-M 43/115-IM	47/165-M 47/151-IM
	Basic module		6-point M and IM	3- and 9-point D	3- and 9-point D	3-point D and ID 9-point D and ID
	Module sharing		M and M IM and IM	no	M and IM	no