任意长度的离散 W 变换的一种递归算法

凌 琦, 舒华忠, 李松毅, 罗立民
(东南大学影像科学与技术实验室, 江苏南京 210096)

摘 要: 离散 W 变换(DWT)在数字信号和图像处理领域有着广泛的应用. 由于其涉及的计算的复杂性, 众多学者提出了诸多 DWT 的快速算法来降低计算复杂度和硬件复杂度. 本文针对任意长度的序列提出一种新的计算 DWT 的递归方法. 我们利用 Clenshaw 递归关系式推导了一种可以有效计算 II 型, III 型和 IV 型 DWT 系数的递归算法. 结果表明, 该算法不仅结构简单, 而且非常适合采用 VLSI 来并行实现.

关键词: 离散 W 变换; Clenshaw 递归关系式; 任意长度
中图分类号: TP301 文献标识码: A 文章编号: 0372-2112 (2007) 10-1949-05

Computation of Discrete W Transform with Arbitrary Length Using Clenshaw's Recurrence Formula

LING Qi, SHU Hua-zhong, LI Song-yi, LUO Li-min
(*Laboratory of Image Science and Technology, Southeast University, Nanjing, Jiangsu 210096, China*)

Abstract: The Discrete W Transforms (DWT) have been widely used in the field of digital signal and image processing. Due to its high computational complexity, many fast algorithms for computing the DWT have been reported in the literature to improve the computational speed and hardware complexity. In this paper, a recursive algorithm for computing the DWT is proposed. By using Clenshaw's recurrence formula, we derive an efficient method for computing the type-II, -III, and -IV DWT of sequences with general length. The results indicate that the proposed algorithms achieve a simple computational structure which is particularly suitable for parallel VLSI realization.

Key words: discrete W transforms (DWT); Clenshaw's recurrence formula; arbitrary length

1 引言

自从 Wang[1] 引入离散 W 变换(Discrete W Transforms: DWT), 它在离散信号和图像处理领域有广泛的应用. 较之离散傅立叶变换, DWT 的一个显著特点是当输入信号为实序列时不需要进行复数运算. DWT 具有四种类型, 分别称为 I 型、II 型、III 型和 IV 型. Wang 已经证明[2], DWT 与信号处理领域另一重要的变换—DHT (Discrete Hartley Transform) 具有相似的定义.

由于 DWT 涉及大量的运算, 有关其快速算法研究吸引了众多学者的关注[3~10]. Wang[3] 提出了一种素数因子的 DWT-I 型快速算法; 当序列长度 $N = q^m$ 时(其中 q 为奇数), Bi[4] 通过将序列分解为若干短序列的 DWT 和离散余弦变换(DCT), 实现了一种 DWT-II 型的快速算法; Bi[5] 还对 $N = p \times 2^m$ (其中 p 为奇数) 的情形提出了一种基 2 的分解方法; 曾泳鸿[6] 等人提出了一种基于素数因子分解的一维任意长度的 DWT 快速算法, 曾泳鸿[7] 还通过将二维 DWT 转化为一种可分离核的变换来提高计算效率, 并提出了利用基于多项式变换的多维 DWT 的快速算法[8]; 钟广军[9] 等人利用一维 DWT 以及多维多项式变换来计算多维 DFT; Liu 等人[10] 通过将三角函数展开为泰勒级数, 建立了 DWT 与一维信号几何矩之间的关系, 实现了一种仅需进行加法运算的 DWT 计算方法. 此外, 针对不同类型的 DHT, Hu[11] 对于序列长度 $N = 2^m$ 的情况, 提出了基 2 的快速计算 GDHT-II 型、III 型和 IV 型的方法; Bi[12] 提出了序列长度为复合数的快速算法.

上述算法除了曾和 Liu 的以外, 大多针对特定的序列长度进行计算. 假如所要计算的 DWT 长度不满足算法的要求, 必须进行补零, 从而降低算法的效率. 此外, 上述算法大部分是基于蝶型结构的快速算法, 不便于采用并行算法实现. 由于递归算法非常适合于硬件设计, 所以在过去的十多年中, Chau 和 Siu[13,14], Wang[15], Aburdene[16] 以及 Nikolajevie 和 Fettweis[17] 分别提出了计算任意长度的 DCT 和 MDCT 系数的递归算法, 章品正[18] 等人利用 Clenshaw 递归关系式得到了二维 Tchebichef 正交矩反变换的快速算法. 在参考上述文献所采用方法的基础上, 我们提出了一种利用 Clenshaw 递归关系式来计算 DWT-II, -III, -IV 的递归算法. 该算法具有公式一致, 结构简单的特点, 并且特别适合于采用

收稿日期: 2006-07-12; 修回日期: 2007-04-28
基金项目: 科学技术部基础研究重大项目(No. 2003CB716102); 教育部新世纪优秀人才支持计划(No. NCET-04-0477); 教育部长江学者和创新团队发展计划

VLSI 实现并行算法.

2 Clenshaw 递归关系式

Clenshaw 递归关系式是一种能够有效计算一系列满足递归关系多项式之和的方法,本文中我们将其用于计算 DWT-II,-III,-IV 型系数.

定义一个由多项式线性组合的函数$f(x)$如下

$$f(x)=\sum_{n=0}^{N-1}c_nF_n(x) \tag{1}$$

其中$F_n(x)$满足如下递归公式

$$F_{n+1}(x)=\alpha(n,x)F_n(x)+\beta(n,x)F_{n-1}(x) \tag{2}$$

这里$\alpha(n,x)$和$\beta(n,x)$是关于 n 和x 的函数.

Clenshaw 递归关系式存在两种形式:升序形式和降序形式,其定义[19]为

2.1 升序形式

定义序列 y_n, $n=0,1,\cdots,N-1$, 它满足如下递归关系:

$$\begin{aligned}&y_{-2}=y_{-1}=0\\&y_n=\frac{1}{\beta(n+1,x)}[y_{n-2}-\alpha(n,x)y_{n-1}-c_n]\end{aligned} \tag{3}$$

则函数$f(x)$可由下式获得

$$f(x)=c_{N-1}F_{N-1}(x)-\beta(N-1,x)F_{N-2}(x)y_{N-2}-F_{N-2}(x)y_{N-3} \tag{4}$$

2.2 降序形式

定义序列 y_n, $n=0,1,\cdots,N-1$, 它满足如下的递归关系:

$$\begin{aligned}&y_{N+1}=y_N=0\\&y_n=\alpha(n,x)y_{n+1}+\beta(n+1,x)y_{n+2}+c_n\end{aligned} \tag{5}$$

则$f(x)$可通过下式计算

$$f(x)=\beta(1,x)F_0(x)y_2+F_1(x)y_1+F_0(x)c_0 \tag{6}$$

3 DWT 的递归算法

本节中利用基于升序的 Clenshaw 递归关系式计算出辅助系数a_n,再利用式(4)得到最后结果,由此给出了 DWT-II 型、DWT-III 型和 DWT-IV 型的递归计算方法.

3.1 DWT-II 型计算方法

长度为 N 的序列$x(n)$的 DWT-II 型的定义如下[1]

$$X^{II}(k)=\sum_{n=0}^{N-1}x(n)\cos\frac{\pi(2n+1)k}{N},\quad k=0,1,\cdots,N-1 \tag{7}$$

其中$\cos\theta=\cos\theta+\sin\theta$.

比较式(7)与式(1),$x(n)$对应式(1)中的 c_k,而$\cos[\pi(2n+1)k]/N$则对应$F_n(x)$,下面我们给出 $F_n(x)$满足的递归关系.令

$$F_n(\theta_k)=\cos\left[\left(n+\frac{1}{2}\right)\frac{2k\pi}{N}\right]=\cos\left[\left(n+\frac{1}{2}\right)\theta_k\right] \tag{8}$$

其中$\theta_k=2k\pi/N$.

利用关系

$$\begin{aligned}F_{n+1}(\theta_k)+F_{n-1}(\theta_k)&=\cos\left[\left(n+\frac{3}{2}\right)\theta_k\right]+\cos\left[\left(n-\frac{1}{2}\right)\theta_k\right]\\&=2\cos(\theta_k)\cdot\cos\left[\left(n+\frac{1}{2}\right)\theta_k\right]\\&=2\cos(\theta_k)\cdot F_n(\theta_k)\end{aligned}$$

可得

$$F_{n+1}(\theta_k)=2\cos(\theta_k)\cdot F_n(\theta_k)-F_{n-1}(\theta_k) \tag{9}$$

比较式(9)和式(2),容易看出

$$\alpha(n,k)=\alpha(k)=2\cos(\theta_k),\ \beta(n,x)=\beta=-1$$

定义如下的递归关系式:

$$\begin{aligned}&a_{-2}=a_{-1}=0\\&a_n=x(n)+2\cos(\theta_k)a_{n-1}-a_{n-2}\\&\qquad n=0,1,\cdots,N-1\end{aligned} \tag{10}$$

利用式(4),(7)中的$X^{II}(k)$可由下式获得

$$X^{II}(k)=x(N-1)F_{N-2}(\theta_k)+F_{N-2}(\theta_k)a_{N-2}-F_{N-1}(\theta_k)a_{N-3} \tag{11}$$

由式(8),上式右端的函数表达为

$$\begin{aligned}F_{N-1}(\theta_k)&=\cos\left[\left(N-1+\frac{1}{2}\right)\theta_k\right]=\cos\left[\left(N-\frac{1}{2}\right)\theta_k\right]\\&=\cos\left(-\frac{\theta_k}{2}\right)\\F_{N-2}(\theta_k)&=\cos\left[\left(N-2+\frac{1}{2}\right)\theta_k\right]=\cos\left[\left(N-\frac{3}{2}\right)\theta_k\right]\\&=\cos\left(-\frac{3}{2}\theta_k\right)\end{aligned}$$

将上述关系式代入式(11),可得

$$X^{II}(k)=\cos\left(-\frac{\theta_k}{2}\right)\cdot[x(N-1)-a_{N-3}]+\cos\left(-\frac{3}{2}\theta_k\right)a_{N-2} \tag{12}$$

令式(10)中的 $n=N-1$,有

$$x(N-1)-a_{N-3}=a_{N-1}-2\cos(\theta_k)\cdot a_{N-2}$$

则式(12)可写为

$$\begin{aligned}X^{II}(k)&=\cos\left(-\frac{\theta_k}{2}\right)\cdot(a_{N-1}-2\cos\theta_k\cdot a_{N-2})+\cos\left(-\frac{3}{2}\theta_k\right)\cdot a_{N-2}\\&=\cos\left(-\frac{\theta_k}{2}\right)\cdot a_{N-1}-a_{N-2}\\&\quad\cdot\left[2\cos\theta_k\cdot\cos\left(-\frac{\theta_k}{2}\right)-\cos\left(-\frac{3}{2}\theta_k\right)\right]\\&=\cos\left(-\frac{\theta_k}{2}\right)a_{N-1}-\cos\left(\frac{\theta_k}{2}\right)a_{N-2}\end{aligned} \tag{13}$$

根据式(10),我们能够从输入序列 $x(n)$ 得到 a_n.

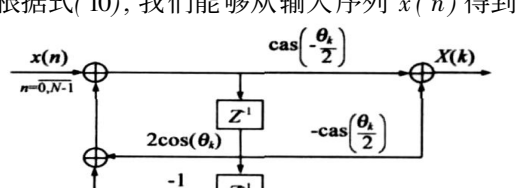


图 1 采用 Clenshaw 递归关系式升序形式得到的计算 DWT-II 信号流图

在第 N 步的时候, 我们可以用式(13)得到第 k 个 DWT-II 型系数 $X^{II}(k)$. 图 1 画出了以这种方法计算第 k 个 DWT-II 型系数的信号流图. 该流图结构可以采用并行 VLSI 实现, 从而提高计算效率.

上面我们采用了 Clenshaw 递归关系式的升序形式计算 $X^{II}(k)$. 我们也尝试了基于降序的 Clenshaw 递归关系式的计算 $X^{II}(k)$的方法, 我们发现, 它需要和升序方式同 样数目的乘法器、加法器以及延时器, 并且输入序列还需要进行倒序运算, 从而更为复杂. 因此本文只选择了基于升序形式的计算方法.

3.2 DWT-III 型计算方法

长度为 N 的序列$x(n)$的 DWT-III 型的定义如下[1]

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

$$k=0,1,\cdots,N-1 \quad (14)$$

令 $$F_n(\varphi_k)=\cos\left(n\frac{(2k+1)\pi}{N}\right)=\cos(n\varphi_k) \quad (15)$$

其中 $\varphi_k=(2k+1)\pi/N$.

定义如下的递归关系式:

$$b_{-2}=b_{-1}=0$$

$$b_n=x(n)+2\cos(\varphi_k)b_{n-1}-b_{n-2},$$

$$n=0,1,\cdots,N-1 \quad (16)$$

采用与 3.1 节中类似的方法, 可得

$$X^{III}(k)=b_{N-2}-\cos(-\varphi_k)b_{N-1} \quad (17)$$

由式(16), 我们可以得到 b_n. 在第 N 步的时候, 可以用式(17)计算出第 k 个 DWT-III 型系数 $X^{III}(k)$. 图 2 给出了信号流图.

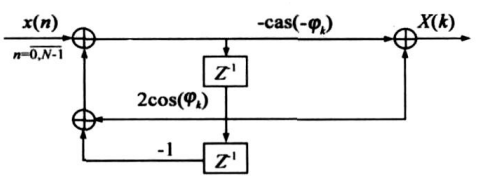


图 2　采用 Clenshaw 递归关系式升序形式得到的计算 DWT-III 信号流图

3.3 DWT-IV 型计算方法

长度为 N 的序列$x(n)$的 DWT-IV 型的定义如下[1]

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

$$k=0,1,\cdots,N-1 \quad (18)$$

令 $$F_n(\phi_k)=\cos\left[\left(n+\frac{1}{2}\right)\frac{(2k+1)\pi}{N}\right]=\cos\left[\left(n+\frac{1}{2}\right)\phi_k\right] \quad (19)$$

其中 $\phi_k=(2k+1)\pi/N$.

定义如下的递归关系式:

$$c_{-2}=c_{-1}=0$$

$$c_n=x(n)+2\cos(\phi_k)c_{n-1}-c_{n-2},$$

$$n=0,1,\cdots,N-1 \quad (20)$$

采用与 3.1 节中类似的方法, 可得

$$X^{IV}(k)=-\cos\left(-\frac{\phi_k}{2}\right)c_{N-1}+\cos\left(\frac{\phi_k}{2}\right)c_{n-2} \quad (21)$$

根据式(20), 我们得到 c_n. 在第 N 步时, 可通过式(21)计算出第 k 个 DWT-IV 型系数. 图 3 给出了信号流图.

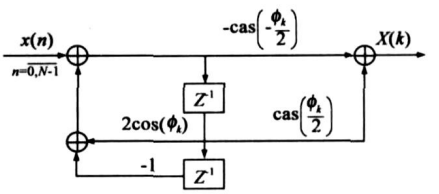


图 3　采用 Clenshaw 递归关系式升序形式得到的计算 DWT-IV 信号流图

3.4 DWT 的并行结构

我们提出 DWT 递归算法的主要目的就是寻找一种适合硬件设计的结构, 提高计算效率, 并简化结构使得其易于硬件实现. 图 4 为相关算法的并行结构图, $x(n)$ 可以并行地输入到 N 个处理单元(Processing Unit)中, 每个处理单元只负责计算一个 $X(k)$, 且每个处理单元递归运算时所需要用到的乘法器 $2\cos\theta_k$ 只和 k 有关, 与 n 无关, 是一个常数. 最后得出的结果可以由多路选择器选择按顺序输出.

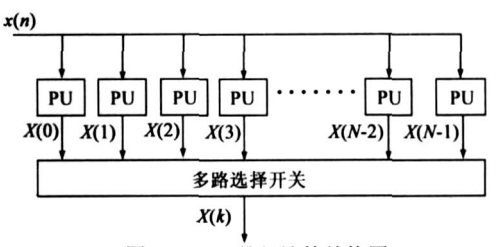


图 4　DWT 并行计算结构图

3.5 二维 DWT 计算方法

因为二维 DWT 变换的核是不可分离的[7], 无法使用传统的行-列算法来计算, 利用文献[20]中提出的方法, 我们可以将二维 DWT 转化为一种可以具有可分离核形式的类 DWT 变换.

首先给出二维 DWT 定义:

$$X(k_1,k_1)=\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1}x(n_1,n_2)\cos(\alpha+\beta) \quad (22)$$

其中 $\alpha=2\pi(n_1+a)(k_1+b)/N_1$, $\beta=2\pi(n_2+a)(k_2+b)/N_2$, $a,b\in\{0,1/2\}$.

利用三角函数性质

$$2\cos(\alpha+\beta)=\cos(\alpha)\cos(\beta)+\cos(\alpha)\cos(-\beta)+\cos(-\alpha)\cos(\beta)-\cos(-\alpha)\cos(-\beta) \quad (23)$$

可以将二维 DWT 变换 $X(k_1,k_2)$变化如下

$$2X(k_1,k_2)=T(k_1,k_2)+T(k_1,N_2-k_2)+T(N_1-k_1,k_2)-T(N_1-k_1,N_2-k_2) \quad (24)$$

其中

$$T(k_1, k_2)=\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1}x(n_1, n_2)\cos(\alpha)\cos(\beta) \quad (25)$$

式(24)表明二维 DWT 对应于 4 个可以使用行-列算法的类 DWT 变换形式. 将 $X(k_1, k_2)$ 分为 4 块矩阵计算, 可以得到如下的形式:

$$2X(k_1, N_2-k_2)=T(k_1, N_2-k_2)+T(k_1, k_2)+T(N_1-k_1, N_2-k_2)-T(N_1-k_1, k_2) \quad (26)$$

$$2X(N_1-k_1, k_2)=T(N_1-k_1, k_2)+T(N_1-k_1, N_2-k_2)+T(k_1, k_2)-T(k_1, N_2-k_2) \quad (27)$$

$$2X(N_1-k_1, N_2-k_2)=T(N_1-k_1, N_2-k_2)+T(N_1-k_1, k_2)+T(k_1, N_2-k_2)-T(k_1, k_2) \quad (28)$$

可以看出只要计算式(24)中 $k_1=0, 1, \cdots, N_1/2-1$, $k_2=0, 1, \cdots, N_2/2-1$ 的序列就可以通过式(26)~(28)得到整个矩阵的其他部分, 由此可以看出将一个长度为 $N_1\times N_2$ 的序列转换为 4 个长度为 $N_1/2\times N_2/2$ 的类 DWT 变换. 从而可以利用上面提出的一维 DWT 递归结构借助行-列算法计算二维 DWT.

4 计算复杂度分析和讨论

本文提出的 DWT 递归算法具有公式一致、结构简单的特点, 算法的流程也很简单(图 5). 每个 $x(k)$ 的计算相对独立, 可以如图 4 直接并行地设计 N 个递归结构, 适合利用 VLSI 实现并行计算. 为了比较, 我们给出文献[6]中提出的任意长度 DWT 算法的流程图(图 6). 可以看出当 N 较大时, 程序的深度可能将很大, 且包含很多子程序, 当遇到较大的素因子时, 程序运行较慢, 如果利用并行电路实现时则需要等待底层计算出结果以后再对得到的结果进行附加运算, 然后高层才可以进行下一步的计算.

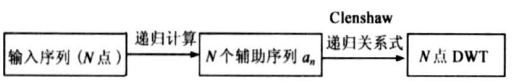


图 5 DWT 递归算法流程示意图

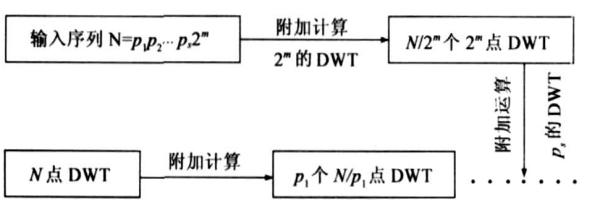


图 6 文献[6]中 DWT 算法流程示意图

表 1 用递归算法得出的计算复杂度

	DWT-II		DWT-III		DWT-IV	
	普通	并行	普通	并行	普通	并行
延时器	2	2N	2	2N	2	2N
乘法器	3	3N	2	2N	3	3N
加法器	3	3N	3	3N	3	3N
乘法	$N+2$	$(N+2)/N$	$N+1$	$(N+1)/N$	$N+2$	$(N+2)/N$
加法	$2N+1$	$(2N+1)/N$	$2N+1$	$(2N+1)/N$	$2N+1$	$(2N+1)/N$
总循环次数	N^2	N	N^2	N	N^2	N

表 2 DWT-II 计算时间比较

	文献[5]中的算法		文献[6]中的算法		递归并行算法	
	TM/N	TA/N	TM/N	TA/N	TM/N	TA/N
12	1.167	5.833	2.667	6	1.1667	2.083
36	1.611	6.944	4.444	9.111	1.056	2.028
64	2.969	7.156	1.531	6.594	1.031	2.016
72	2.361	8.194	4.694	9.861	1.028	2.014
192	3.240	9.927	4.198	10.594	1.010	2.005
240	2.925	10.525	4.290	13.875	1.008	2.004
576	3.955	12.476	5.975	10.041	1.003	2.002
1152	4.491	13.960	6.460	15.158	1.002	2.001

本算法的计算量可以从各种类型的递归算法的信号流图中直接得出, 我们将其列在表 1 中, 表中的数字是计算每一点的 DWT 所需要的运算量, 虽然普通的结构与直接计算复杂度大致相同, 但是可以看出并行结构只需要 N 次递归, 对于每一个 $X(k)$ 计算的结构均相同, 只是在乘法器的参数上有所不同, 且乘法器上的参数 θ_k 与 n 无关, 对于一个固定的 k, θ_k 为常数, 这样就使得计算循环次数降到了 N, 其代价是硬件设计时需要 N 个递归处理单元. 由于迄今为止尚未见有关 DWT 递归算法的文献报道, 我们列出了文献[5]和[6]中蝶型算法平均计算一点 DWT 所需要的时间, 与递归的并行算法进行比较, 结果见表 2(其中 TM 和 TA 分别为计算一个乘法和加法所需要的时间, 这里都设为 1). 从表中可以看出, 在利用了并行算法以后, 平均计算一点 DWT 所需要的复杂度是递减的, 这是因为并行结构中设置了更多的 PU, N 点的 $X(k)$ 可以直接并行计算, 而不像上述算法需要等待其他计算的结果才可以开始计算, 所以节省了时间. 文献[5]和[6]中的算法也可以用类似并行 FFT 电路的方法通过设计并行电路来提高计算速度, 但是对于每个不同的 N 都需要按照其因子的组合来重新设计, 不像递归算法只需要添加或减少 PU 来实现其他的长度那么简单. 假如有很多不同长度的 N 需要计算时, 利用上述算法针对不同的 N 分别设计出相应的并行电路结构的工作量将会很大. 综合以上分析, 本文提出的计算任意长度的 DWT 方法更有效, 且硬件实现更为简单.

5 结论

本文首先利用 Clenshaw 递归关系式得到了一种可以用硬件实现并行计算的结构, 利用 N 个递归结构进行递归计算, 可以使计算效率提高 N 倍. 本文的算法对于任意长度的 DWT 都可以用相同的结构来实现, 具有公式一致、结构简单的特点, 而且在计算的时候不需要补零来达到特定的序列长度而降低计算效率. 对于输入输出序列都不需要进行数据重排, 因而更加适合于实时计算. 由于递归算法自身易于并行计算的特点, 此

算法非常适合利用并行 VLSI 来实现, 并且可以利用文中提出的方法将二维 DWT 转化成一维 DWT 来计算.

参考文献:

[1] Wang Z, Hunt B R. The discrete W transform[J]. Appl Math Comput, 1985, 16(1): 19- 48.

[2] Wang Z. Comments on 'Generalized discrete Hartley transform'[J]. IEEE Trans on Signal Process, 1995, 43(7): 1711- 1712.

[3] Wang Z. A prime factor fast W transform algorithm[J]. IEEE Trans on Signal Process, 1992, 40(9): 2361- 2368.

[4] Bi G, Liu J. Fast odd-factor algorithm for W transform[J]. Electron. Lett, 1998, 34(5): 431- 433.

[5] Bi G. On computation of the discrete W transform[J]. IEEE Trans on Signal Process, 1999, 47(5): 1450- 1453.

[6] 曾泳鸿, 蒋增荣. 任意长度 W 变换的统一算法及其实现[J]. 计算数学, 1996, 18(3): 321- 327.
Zeng Yong-hong, Jiang Zeng-rong. A Unified fast algorithm for the discrete W transform with arbitrary length[J]. Mathematica Numerica Sinica, 1996, 18(3): 321- 327. (in Chinese)

[7] 曾泳鸿, 张小水. 二维离散 W 变换的快速算法及其应用[J]. 数值计算与计算机应用, 1997, 18(1): 8- 14.
Zeng Yong-hong, Zhang Xiao-shui. A fast algorithm for 2D discrete W transform and its application[J]. Journal on Numerical Methods and Computer Applications, 1997, 18(1): 8- 14. (in Chinese)

[8] 曾泳鸿, 李晓梅. 用多项式变换计算多维离散 W 变换[J]. 计算数学, 1998, 20(3): 291- 298.
Zeng Yong-hong, Li Xiao-mei. Computing multi-dimensional discrete W transforms by polynomial transforms[J]. Mathematica Numerica Sinica, 1998, 20(3): 291- 298. (in Chinese)

[9] 钟广军, 成礼智, 陈火旺. 多维 DFT 的多维多项式变换与离散 W 变换算法[J]. 电子学报, 2001, 29(8): 1053- 1056.
Zhong Guang-jun, Cheng Li-zhi, Chen Huo-wang. Multidimensional polynomial transform and discrete W transform algorithms for multidimensional DFT[J]. Acta Electronica Sinica, 2001, 29(8): 1053- 1056. (in Chinese)

[10] Liu J G, Liu Y Z, Wang G Y. Fast discrete W transforms via computation of moments[J]. IEEE Trans on Signal Process, 2005, 53(2): 654- 659.

[11] Hu N C, Chang H I, Ersoy O K. Generalized discrete Hartley transforms[J]. IEEE Trans on Signal Process, 1992, 40(12): 2361- 2368.

[12] Bi G, Chen Y, Zeng Y. Fast algorithms for generalized discrete Hartley transforms of composite sequence lengths[J]. IEEE Trans on Circuits and Systems Ⅱ: Analog and Digital Signal Process, 2000, 47(9): 893- 901.

[13] L-p Chau, W-C Siu. Recursive algorithm for the discrete cosine transform with general lengths[J]. Electron Lett, 1994, 30(2): 197- 198.

[14] L-p Chau, W-C Siu. Efficient recursive algorithms for the inverse discrete cosine transform[J]. IEEE Signal Process. Lett, 2000, 7(10): 276- 277.

[15] Z Wang, G A Jullien, W C Miller. Recursive algorithm for the forward and inverse discrete cosine transform with arbitrary lengths[J]. IEEE Signal Process Lett, 1994, 1(7): 101- 102.

[16] M F Aburdene, J Zheng, R J Kozick. Computation of discrete cosine transform using Clenshaw's recurrence formula[J]. IEEE Signal Process Lett, 1995, 2(8): 155- 156.

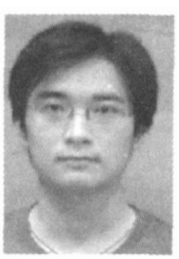
[17] V Nikolajevic, G Fettweis. Computation of Forward and Inverse MDCT Using Chebyshev's Recurrence Formula[J]. IEEE Trans on Signal Process, 2003, 51(5): 1439- 1444.

[18] 章品正, 舒华忠, 杨冠羽, 徐旦华. 二维 Tchebichef 正交矩反变换的快速算法[J]. 计算机学报, 2006, 29(4): 648- 651.
Zhang Pin-zheng, Shu Hua-zhong, Yang Guan-yu, Xu Dan-hua. A method for efficiently computing the two2Dimensional inverse tchebichef orthogonal moments[J]. Chinese Journal of Computers, 2006, 29(4): 648- 651. (in Chinese)

[19] W H Press, S A Teukolsky, W T Vetterling, B P Flannery. Numerical Recipes in C[M]. Cambridge, U.K: Cambridge Univ. Press, 1992. 178- 183.

[20] R N Bracewell, O Buneman, H Hao, J Villasenor. Fast two-dimensional hartlety transform[J]. Proc IEEE, 1986, 74(9): 1282- 1283.

作者简介:



凌　琦　男, 1982 年生于江苏镇江, 硕士研究生. 目前主要的研究方向为 DSP 快速算法. E-mail: pet19821231@163.com

舒华忠　男, 1965 年生于江西省玉山县, 教授、博士生导师. 主要研究领域为医学图像处理、模式识别和 DSP 快速算法等.