Radix-3x3 algorithm for the 2-D discrete Hartley transform

Jiasong Wu, Huazhong Shu, Lotfi Senhadji, Limin M. Luo

Radix-3×3 algorithm for the 2-D discrete Hartley transform

J.S. Wu, H.Z. Shu, *Senior Member*, *IEEE*, L. Senhadji, *Senior Member*, *IEEE*, L.M. Luo, *Senior Member*, *IEEE*

***Abstract*—In this correspondence, we propose a vector-radix algorithm for the fast computation of two-dimensional (2-D) discrete Hartley transform (DHT). For data sequences whose length is power of three, a radix-3×3 decimation in frequency algorithm is developed. It decomposes a length-*N*×*N* DHT into nine length-(*N*/3)×(*N*/3) DHTs. Comparison of the computational complexity with known algorithms shows that the proposed algorithm, in some cases, reduces significantly the number of arithmetic operations.**

***Index Terms*—2-D discrete Hartley transform, 2-D radix-3×3, vector-radix algorithm**

I. Introduction

The two-dimensional (2-D) discrete Hartley transform (DHT) introduced by Bracewell [1] has become an important tool in image and signal processing [2] and circular convolution [3]. Because the computation of 2-D DHT by the traditional row-column algorithm [4] is still time consuming, many fast algorithms for computing the 2-D DHT have been reported in the literature. These algorithms can be grouped into two categories. The first category uses the method of (split) vector-radix to decompose the whole processing task into many smaller ones [5-10]. Among them, both Bi's decimation in frequency (DIF) algorithm [9] and Bi's decimation in time (DIT) algorithm [10], which support the block size $q*2^m \times q*2^m$, for $m \geq 2$, where q is an odd integer, require the least number of arithmetic operations. These (split) vector-radix algorithms were recently extended to the fast computation of 3-D [11-14] and M-D ([15], [16]) DHT. The second category utilizes other transforms such as discrete Radon transform (DRT) [17-19] or the polynomial transform (PT) [20-22] to speed up the computational efficiency. Among this kind of algorithms, Zeng's method [21], supporting the block size $q^{m_1} \times q^{m_2} \times \cdots \times q^{m_r}$, for $m_i \geq 2$, $i = 1, 2, \ldots, r$, where q is an odd prime integer, and Zeng's approach [22], supporting the block size $2^{m_1} \times 2^{m_2} \times \cdots \times 2^{m_r}$, for $m_i \geq 2$, $i = 1, 2, \ldots, r$, are probably the most efficient algorithms in terms of the arithmetic complexity.

Generally speaking, the (split) vector-radix algorithms have many desirable properties such as regular computational structure, in-place computation, and low implementation cost [9], [10]. On the other hand, the DRT or PT based algorithms require less number of arithmetic operations than that of the (split) vector-radix algorithms, but need a special sequence reordering, thus necessitating extra arithmetic operations, modulo operations, and bit-shift operations and complex computational structures [16].

Most of the (split) vector-radix algorithms developed till now for the fast computation of 2-D DHT dealt with the block size $2^m \times 2^m$, $m \geq 2$. However, in some practical applications, for example, in the computation of 2-D circular convolution where one uses the minimum block sizes compatible to the filter specifications [23], the block size is not limited to $2^m \times 2^m$, $m \geq 2$, even not to $q*2^m \times q*2^m$, $m \geq 2$, where q is an odd integer. Therefore, a new 2-D vector-radix DHT algorithm which supports a more wide range of choices on different block sizes is required. This is the objective of the present paper.

In Ref. [24], Zhao proposed a radix-3 DIF algorithm for efficient computation of 1-D DHT. Inspired by his research work, we present here a new vector-radix-3×3 DIF algorithm for fast computing the 2-D DHT.

II. Method

The 2-D $N \times N$ -point DHT of $x(n_1, n_2)$ is defined by [1]

$$X(k_1,k_2) = \frac{1}{N^2}\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} x(n_1,n_2)\operatorname{cas}\left[\frac{2\pi}{N}(n_1k_1+n_2k_2)\right],$$

$$0 \leq n_i, k_i \leq N-1,\ i = 1, 2, \quad (1)$$

where $\operatorname{cas}(x) = \cos(x)+\sin(x)$ and N is assumed to be power of 3. For simplicity, the normalization factor $1/N^2$ is omitted in the following derivation. The following properties can be easily verified:

$$\begin{aligned} X(k_1,N-k_2) &= X(k_1,-k_2),\\ X(N-k_1,k_2) &= X(-k_1,k_2), \qquad 0 \leq k_1, k_2 \leq N-1. \quad (2)\\ X(N-k_1,N-k_2) &= X(-k_1,-k_2), \end{aligned}$$

Based on (2), nine cases need to be calculated instead of computing (1) directly: $X(3k_1, 3k_2)$, $X(3k_1, 3k_2+1)$, $X(3k_1, 3k_2-1)$, $X(3k_1+1, 3k_2)$, $X(3k_1-1, 3k_2)$, $X(3k_1+1, 3k_2+1)$, $X(3k_1-1, 3k_2-1)$, $X(3k_1+1, 3k_2-1)$, and $X(3k_1-1, 3k_2+1)$, for $0 \leq k_1, k_2 \leq N/3-1$. These values can be easily got from the following $A_i(k_1, k_2)$, $i = 0, 1, 2, 3, 4$, and $B_i(k_1, k_2)$, $i = 1, 2, 3, 4$.

This work was supported by National Basic Research Program of China under grant N⁰ 2003CB716102 and Program for Changjiang Scholars and Innovative Research Team in University.

J.S. Wu, H.Z. Shu, and L.M. Luo are with the Laboratory of Image Science and Technology, School of Computer Science and Engineering, Southeast University, 210096, Nanjing, China (e-mail: jswu@seu.edu.cn; shu.list@seu.edu.cn; luo.list@seu.edu.cn).

L. Senhadji is with the INSERM, U642, Rennes, F-35000, France, and with the Université de Rennes 1, LTSI, Rennes, F-35000, France (e-mail: lotfi.senhadji@univ-rennes1.fr).

Authors are all with "Centre de Recherche en Information Biomédicale Sino-Français (CRIBs)".

$$\begin{bmatrix} A_0(k_1,k_2) \\ A_1(k_1,k_2) \\ B_1(k_1,k_2) \\ A_2(k_1,k_2) \\ B_2(k_1,k_2) \\ A_3(k_1,k_2) \\ B_3(k_1,k_2) \\ A_4(k_1,k_2) \\ B_4(k_1,k_2) \end{bmatrix} = \begin{bmatrix} 1 &&&&&&&& \\ & 1 & 1 &&&&&& \\ & 1 & -1 &&&&&& \\ &&& 1 & 1 &&&& \\ &&& 1 & -1 &&&& \\ &&&&& 1 & 1 && \\ &&&&& 1 & -1 && \\ &&&&&&& 1 & 1 \\ &&&&&&& 1 & -1 \end{bmatrix} \times \begin{bmatrix} X(3k_1,3k_2) \\ X(3k_1,3k_2+1) \\ X(3k_1,3k_2-1) \\ X(3k_1+1,3k_2) \\ X(3k_1-1,3k_2) \\ X(3k_1+1,3k_2+1) \\ X(3k_1-1,3k_2-1) \\ X(3k_1+1,3k_2-1) \\ X(3k_1-1,3k_2+1) \end{bmatrix} \quad (3)$$

In order to facilitate the presentation, we introduce some intermediate variables as follows:

$$\begin{bmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \\ S_4 \\ D_1 \\ D_2 \\ D_3 \\ D_4 \end{bmatrix} = \begin{bmatrix} 1 &&&&&&&& \\ & 1 & 1 &&&&&& \\ && 1 &&& 1 &&& \\ &&& 1 &&&& 1 & \\ &&&& 1 && 1 && \\ & 1 & -1 &&&&&& \\ && 1 &&& -1 &&& \\ &&& 1 &&&& -1 & \\ &&&& 1 && -1 && \end{bmatrix} \times \begin{bmatrix} x(n_1,n_2) \\ x(n_1,n_2+N/3) \\ x(n_1,n_2+2N/3) \\ x(n_1+N/3,n_2) \\ x(n_1+N/3,n_2+N/3) \\ x(n_1+N/3,n_2+2N/3) \\ x(n_1+2N/3,n_2) \\ x(n_1+2N/3,n_2+N/3) \\ x(n_1+2N/3,n_2+2N/3) \end{bmatrix} \quad (4)$$

$$\begin{bmatrix} E_0 \\ E_1 \\ E_2 \\ E_3 \\ E_4 \\ E_5 \end{bmatrix} = \begin{bmatrix} 1 & 1 &&& \\ 1 && 1 && \\ 1 &&& 1 & \\ 1 &&&& 1 \\ & 1 & 1 && \\ &&& 1 & 1 \end{bmatrix} \times \begin{bmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \\ S_4 \end{bmatrix} \quad (5)$$

$$\phi = \frac{2\pi}{N/3}\left(n_1k_1 + n_2k_2\right), \quad (6)$$

$$\theta_{p,q} = \frac{2\pi}{N}(pn_1 + qn_2), \quad p, q = -1, 0, 1. \quad (7)$$

In the following, we discuss the way for efficiently computing $A_i(k_1, k_2)$, $i = 0, 1, 2, 3, 4$, and $B_i(k_1, k_2)$, $i = 1, 2, 3, 4$.

1. Computation of $A_0(k_1, k_2)$.

$$\begin{aligned} &A_0(k_1,k_2) \\ &= \sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} x(n_1,n_2)\,\mathrm{cas}\left[\frac{2\pi}{N/3}\left(n_1k_1+n_2k_2\right)\right] \quad (8) \\ &= \sum_{n_1=0}^{N/3-1}\sum_{n_2=0}^{N/3-1}\left(E_1+(E_5+S_1)\right)\mathrm{cas}\phi. \end{aligned}$$

2. Computation of $A_1(k_1, k_2)$.

$$\begin{aligned} &A_1(k_1,k_2) \\ &= \sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} x(n_1,n_2)\left\{\mathrm{cas}\left[\frac{2\pi}{N}\left(3n_1k_1+n_2(3k_2+1)\right)\right]\right. \\ &\qquad \left. + \mathrm{cas}\left[\frac{2\pi}{N}\left(3n_1k_1+n_2(3k_2-1)\right)\right]\right\} \\ &= 2\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} x(n_1,n_2)\cos\left(\frac{2\pi n_2}{N}\right) \quad (9) \\ &\qquad \times \mathrm{cas}\left[\frac{2\pi}{N/3}\left(n_1k_1+n_2k_2\right)\right] \\ &= \sum_{n_1=0}^{N/3-1}\sum_{n_2=0}^{N/3-1}\left[\left(2E_1-(E_5+S_1)\right)\cos\theta_{0,1}\right. \\ &\qquad \left. -\sqrt{3}\left(D_1+D_3-D_4\right)\sin\theta_{0,1}\right]\mathrm{cas}\phi. \end{aligned}$$

3. Computation of $B_1(k_1, k_2)$.

$$\begin{aligned} &B_1(k_1,k_2) \\ &= \sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} x(n_1,n_2)\left\{\mathrm{cas}\left[\frac{2\pi}{N}\left(3n_1k_1+n_2(3k_2+1)\right)\right]\right. \\ &\qquad \left. - \mathrm{cas}\left[\frac{2\pi}{N}\left(3n_1k_1+n_2(3k_2-1)\right)\right]\right\} \\ &= 2\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} x(n_1,n_2)\sin\left(\frac{2\pi n_2}{N}\right) \\ &\qquad \times \mathrm{cas}\left[-\frac{2\pi}{N/3}\left(n_1k_1+n_2k_2\right)\right], \end{aligned}$$

we have

$$\begin{aligned} &B_1\left(\frac{N}{3}-k_1, \frac{N}{3}-k_2\right) \\ &= 2\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} x(n_1,n_2)\sin\left(\frac{2\pi n_2}{N}\right)\mathrm{cas}\left[\frac{2\pi}{N/3}\left(n_1k_1+n_2k_2\right)\right] \\ &= \sum_{n_1=0}^{N/3-1}\sum_{n_2=0}^{N/3-1}\left[\left(2E_1-(E_5+S_1)\right)\sin\theta_{0,1}\right. \\ &\qquad \left. +\sqrt{3}\left(D_1+D_3-D_4\right)\cos\theta_{0,1}\right]\mathrm{cas}\phi. \end{aligned} \quad (10)$$

Similarly, we have

$$\begin{aligned} &A_2(k_1,k_2) \\ &= \sum_{n_1=0}^{N/3-1}\sum_{n_2=0}^{N/3-1}\left[\left(2E_0-E_5-S_2\right)\cos\theta_{1,0}\right. \quad (11) \\ &\qquad \left. -\sqrt{3}\left(D_2+D_3+D_4\right)\sin\theta_{1,0}\right]\mathrm{cas}\phi. \end{aligned}$$

$$B_2\left(\frac{N}{3}-k_1,\frac{N}{3}-k_2\right)$$
$$=\sum_{n_1=0}^{N/3-1}\sum_{n_2=0}^{N/3-1}\Big[(2E_0-E_5-S_2)\sin\theta_{1,0} \tag{12}$$
$$+\sqrt{3}(D_2+D_3+D_4)\cos\theta_{1,0}\Big]\text{cas}\phi.$$

$$A_3(k_1,k_2)$$
$$=\sum_{n_1=0}^{N/3-1}\sum_{n_2=0}^{N/3-1}\Big[(2E_3-E_4-S_3)\cos\theta_{1,1} \tag{13}$$
$$-\sqrt{3}(D_1+D_2-D_3)\sin\theta_{1,1}\Big]\text{cas}\phi.$$

$$B_3\left(\frac{N}{3}-k_1,\frac{N}{3}-k_2\right)$$
$$=\sum_{n_1=0}^{N/3-1}\sum_{n_2=0}^{N/3-1}\Big[(2E_3-E_4-S_3)\sin\theta_{1,1} \tag{14}$$
$$+\sqrt{3}(D_1+D_2-D_3)\cos\theta_{1,1}\Big]\text{cas}\phi.$$

$$A_4(k_1,k_2)$$
$$=\sum_{n_1=0}^{N/3-1}\sum_{n_2=0}^{N/3-1}\Big[(2E_2-E_4-S_4)\cos\theta_{1,-1} \tag{15}$$
$$+\sqrt{3}(D_1-D_2+D_4)\sin\theta_{1,-1}\Big]\text{cas}\phi.$$

$$B_4\left(\frac{N}{3}-k_1,\frac{N}{3}-k_2\right)$$
$$=\sum_{n_1=0}^{N/3-1}\sum_{n_2=0}^{N/3-1}\Big[(2E_2-E_4-S_4)\sin\theta_{1,-1} \tag{16}$$
$$-\sqrt{3}(D_1-D_2+D_4)\cos\theta_{1,-1}\Big]\text{cas}\phi.$$

The indices k_1 and k_2 from (8) to (16) are ranged from 0 to $N/3-1$.

So far, we have decomposed a length-$N\times N$ DHT (defined in (1)) into nine length-$(N/3)\times(N/3)$ DHTs (defined in (8) through (16)). Fig. 1 shows the flowgraph of the realization of the proposed algorithm.

III. Computational Complexity and Comparison Analysis

In this section, we consider the computational complexity of the proposed algorithm and compare it with some known algorithms. The detailed analysis is given below.

i) The computation of (4) and (5) requires $(N/3)\times(N/3)\times14$ additions.

ii) The implementation of the butterfly in the input data sequence of $A_i(k_1, k_2)$ and $B_i(N/3-k_1, N/3-k_2)$, $i = 1, 2, 3, 4$, needs 4 multiplications and 2 additions, thus, the computation from (8) to (16) requires $(N/3)\times(N/3)\times16$ multiplications and $(N/3)\times(N/3)\times25$ additions.

iii) The computation of $X(k_1, k_2)$ from $A_i(k_1, k_2)$, $i = 0, 1, 2, 3, 4$, and $B_i(k_1, k_2)$, $i = 1, 2, 3, 4$, requires $(N/3)\times(N/3)\times8$ additions.

iv) Taking the special cases $n_1 = 0$, $n_2 = 0$, $n_1 + n_2 = 0$ or $N/3$, and $n_1 = n_2$ into consideration, $4N$ multiplications and $2N+2$ additions can be saved in the computation through (8) to (16). In fact, let us consider, for example, the number of arithmetic operations that can be saved in the computation of (13) and (14) for the cases where $n_1 + n_2 = 0$ or $N/3$. Letting $a_{11} = 2E_3-E_4-S_3$ and $b_{11} = D_1+D_2-D_3$, when $n_1 + n_2 = 0$, the input data sequences in (13) and (14) become a_{11} and $\sqrt{3}b_{11}$, in such case, 3 multiplications and 2 additions can be saved. When $n_1 + n_2 = N/3$, the input data sequences become $-\frac{1}{2}(a_{11}-b_{11})-2b_{11}$ and $\frac{\sqrt{3}}{2}(a_{11}-b_{11})$, $N-3$ multiplications can be saved. Therefore, for the two cases $n_1 + n_2 = 0$ and $n_1 + n_2 = N/3$, N multiplications and 2 additions can be saved in the computation of (13) and (14). A similar analysis can be done for other cases.

The computational complexity of the proposed algorithm is therefore given by

$$\begin{cases} M_{N\times N} = 16N^2/9 - 4N + 9M_{(N/3)\times(N/3)}, \\ A_{N\times N} = 47N^2/9 - 2N - 2 + 9A_{(N/3)\times(N/3)}, \end{cases} \tag{17}$$

with initial values $M_{3\times3} = 4$ and $A_{3\times3} = 47$.

Table I shows the computational complexity required by the proposed algorithm and Zeng's method [21] for the block size $3^m\times3^m$, $m\geq1$. Tables II and III present the computational complexities of Zeng's approach [22] and Bi's algorithm ($q = 1$) [9], [10] for the block size $2^m\times2^m$, $m \geq 2$ and Bi's algorithm ($q = 3$) for the block size $3*2^m\times3*2^m$, $m \geq 0$, respectively. Note that in Bi's algorithm for $q = 3$, the same initial values $M_{3\times3} = 4$ and $A_{3\times3} = 47$ are used. To make the comparison more clear, Fig. 2 shows the number of multiplications plus additions, involved in the computation of the length-$N\times N$ DHT, using the proposed method and the algorithms presented in [9], [10], [21] and [22].

It can be seen from Table I that the proposed algorithm is more efficient than Zeng's method [21] in terms of the total number of arithmetic operations. Tables II and III, as well as Fig. 2, show that the proposed algorithm is more efficient, in some cases, than the algorithms presented in [9], [10], and [22] in terms of the number of arithmetic operations, especially for the cases where there is many zero padding, such as block sizes 9×9 and 81×81. Thus, user may favor a given technique depending on the selected block size. For more detailed discussion about the vector-radix algorithm and the polynomial algorithm, we refer the readers to Ref. [16].

IV. Conclusions

We have proposed in this paper a radix-3×3 DIF algorithm for the fast computation of 2-D DHT. Compared with some known algorithms, the proposed one achieves substantial saving on the number of arithmetic operations. Moreover, the proposed algorithm possesses properties such as the butterfly-style and in-place computations that are highly desirable for software as well as hardware implementation. It can be used for fast convolution where one uses the minimum block sizes compatible to the filter specifications. Note that our radix-3×3 algorithm can be easily extended to the arbitrary radix-$q \times q$ case, where q is an odd integer, which provides a wider choice of block sizes. Thus, user can favor an approach for a desired block. Furthermore, our proposed algorithm, combing with other split vector-radix algorithms [5-10], can realize the 2-D DHT with block size $3^m 2^n \times 3^m 2^n$, m, $n \geq 2$. Since the DHT is an efficient alternative to the DFT for real data, the proposed algorithm may also find its application in array signal processing [25].

REFERENCES

[1] R.N. Bracewell, *The Hartley Transform*, New York, NY: Oxford Univ. Press, 1986.

[2] I. Duleba, "Hartley transform in compression of medical ultrasonic images," *Proc. IEEE ICIAP*, Sept. 1999, pp. 722-727.

[3] N.C. Hu and F.F. Lu, "Fast computation of the two-dimensional generalised Hartley transforms," *IEE Proc. Vis. Image Signal Process.*, vol. 142, no.1, pp. 35-39, Feb. 1995.

[4] R.N. Bracewell, O. Buneman, H. Hao, and J. Villasenor, "Fast two-dimensional Hartley transform," *Proc. IEEE*, vol. 74, no. 9, pp. 1282-1283, Sept. 1986.

[5] R. Kumaresan and P.K. Gupta, "Vector-radix algorithm for a 2-D discrete Hartley transform," *Proc. IEEE*, vol. 74, no. 5, pp. 755-757, May 1986.

[6] E.A. Jonckheere and C. Ma, "Split-radix fast Hartley transform in one and two dimensions," *IEEE Trans. Signal Process.*, vol. 39, no.2, pp. 499-503, Feb. 1991.

[7] S. Huang, J. Wang, and H. Qiu, "Split vector radix algorithm for two-dimensional Hartley transform," *IEEE Trans. Aerosp. Electron. Syst.* vol. 27, no.6, pp. 865-868, Nov. 1991.

[8] J.L. Wu and S.C. Pei, "The vector split-radix algorithm for 2D DHT," *IEEE Trans. Signal Process.*, vol. 41, no.2, pp. 960-965, Feb. 1993.

[9] G. Bi, "Split-radix algorithm for 2-D discrete Hartley transform," *Signal Process.*, vol. 63, no. 1, pp. 45-53, Nov. 1997.

[10] G. Bi, A.C. Kot, and Z. Meng, "Computation of 2D discrete Hartley transform," *Electron. Lett.,* vol. 34, no.11, pp. 1058-1059, May 1998.

[11] S. Boussakta, O.H. Alshibami, and M.Y. Aziz, "Radix-2×2×2 algorithm for the 3-D discrete Hartley transform," *IEEE Trans. Signal Process.*, vol. 49, no. 12, pp. 3145-3156, Dec. 2001.

[12] O. Alshibami and S. Boussakta, "Fast 3-D decimation-in-frequency algorithm for 3-D Hartley transform," *Signal Process.,* vol. 82, no. 1, pp. 121-126, Jan. 2002.

[13] S. Bouguezel, M.O. Ahmad, and M.N.S. Swamy, "An efficient three-dimensional decimation-in-time FHT algorithm based on the radix-2/4 approach," *Proc. IEEE ISSPIT,* Dec. 2004, pp. 52-55.

[14] S. Bouguezel, M.O. Ahmad, and M.N.S. Swamy, "A split vector-radix algorithm for the 3-D discrete Hartley transform," *IEEE Trans. Circuits Syst.-I: Regular paper*, vol. 53, no. 9, pp. 1966-1976, Sept. 2006.

[15] S. Bouguezel, M.O. Ahmad, and M.N.S. Swamy, "An efficient multidimensional decimation-in-frequency FHT algorithm based on the radix-2/4 approach," *IEEE ISCAS,* May 2005, pp. 2405-2408.

[16] S. Bouguezel, M.N.S. Swamy, and M.O. Ahmad, "Multidimensional vector radix FHT algorithms," *IEEE Trans. Circuits Syst.-I: Regular paper*, vol. 53, no. 4, pp. 905-917, Apr. 2006.

[17] D. Yang, "New fast algorithm to compute two-dimensional discrete Hartley transform," *Electron. Lett.,* vol. 25, no. 25, pp. 1705-1706, Dec. 1989.

[18] W. Ma, "Algorithm for computing two-dimensional discrete Hartley transform of size $p^n \times p^n$," *Electron. Lett.,* vol. 26, no. 21, pp. 1795-1797, Oct. 1990.

[19] W. Ma, "Number of multiplications necessary to compute length-2^n two-dimensional discrete Hartley transform DHT (2^n; 2)," *Electron. Lett.,* vol. 28, no. 5, pp. 480-482, Feb. 1992.

[20] S.C. Chan and K.L. Ho, "Polynomial transform fast Hartley transform," *IEEE ISCAS*, vol. 1, Jun. 1991, pp. 642-645.

[21] Y. H. Zeng, G. Bi, and A. C. Kot, "Fast algorithm for multi-dimensional discrete Hartley transform with size $q^{l_1} \times q^{l_2} \times \cdots \times q^{l_r}$," *Signal Process.*, vol. 82, no. 3, pp. 497-502, Mar. 2002.

[22] Y. H. Zeng, G. Bi, and A. R. Leyman, "New algorithms for multidimensional discrete Hartley transform," *Signal Process.*, vol. 82, no. 8, pp. 1086-1095, Aug. 2002.

[23] E. Dubois and A.N. Venetsanopoulos, "A new algorithm for the radix-3 FFT," *IEEE Trans. Acoust., Speech, Signal Process.,* vol. 26, no. 3, pp. 222-225, Jun. 1978.

[24] Z.J. Zhao, "In-place radix-3 fast Hartley transform algorithm," *Electron. Lett.,* vol. 28, no.3, pp. 319-321, Jan. 1992.

[25] C.J. Ju, "Algorithm of defining 1-D indexing for M-D mixed radix FFT implementation," *IEEE PRCCCSP*, vol.2, May 1993, pp. 484-488.

Table I The computational complexity required by the proposed algorithm and Zeng's method for the block size $3^m \times 3^m$, $m \geq 1$

$N\times N$ (N)	Proposed algorithm			Zeng's method [21]		
	Mults.	Adds.	M.+A.	Mults.	Adds.	M.+A.
3	4	47	51	4	47	51
9	144	826	970	136	892	1028
27	2484	11185	13669	2188	12394	14582
81	33696	134764	168460	28432	150904	179336
243	407268	1520755	1928023	334612	1712422	2047034
729	4607280	16460638	21067918	3720088	18600436	22320524

Table II The computational complexity needed by Zeng's approach and Bi's algorithm (q=1) for the block size $2^m \times 2^m$, $m \geq 2$

$N\times N$ (N)	Zeng's approach [22]			Bi's algorithm (q=1) [9][10]		
	Mults.	Adds.	M.+A.	Mults.	Adds.	M.+A.
4	2	58	60	2	58	60
8	26	354	380	24	408	432
16	218	2018	2236	264	2216	2480
32	1370	10594	11964	1800	11368	13168
64	7514	52578	60092	10536	55176	65712
128	38234	251234	289468	55560	260840	316400
256	185690	1168738	1354428	277992	1200712	1478704
512	873818	5330274	6204092	1333320	5443368	6776688
1024	4019546	23942498	27962044	6232872	24305288	30538160

Table III The computational complexity needed by Bi's algorithm (q=3) for the block size $3*2^m \times 3*2^m$, $m \geq 0$

$N\times N$ (N)	Bi's algorithm (q=3) [9][10]		
	Mults.	Adds.	M.+A.
3	4	47	51
6	16	260	276
12	64	1328	1392
24	472	6680	7152
48	3400	31976	35376
96	20296	150440	170736
192	111208	689096	800304
384	565576	3117608	3683184
768	2764072	13886600	16650672

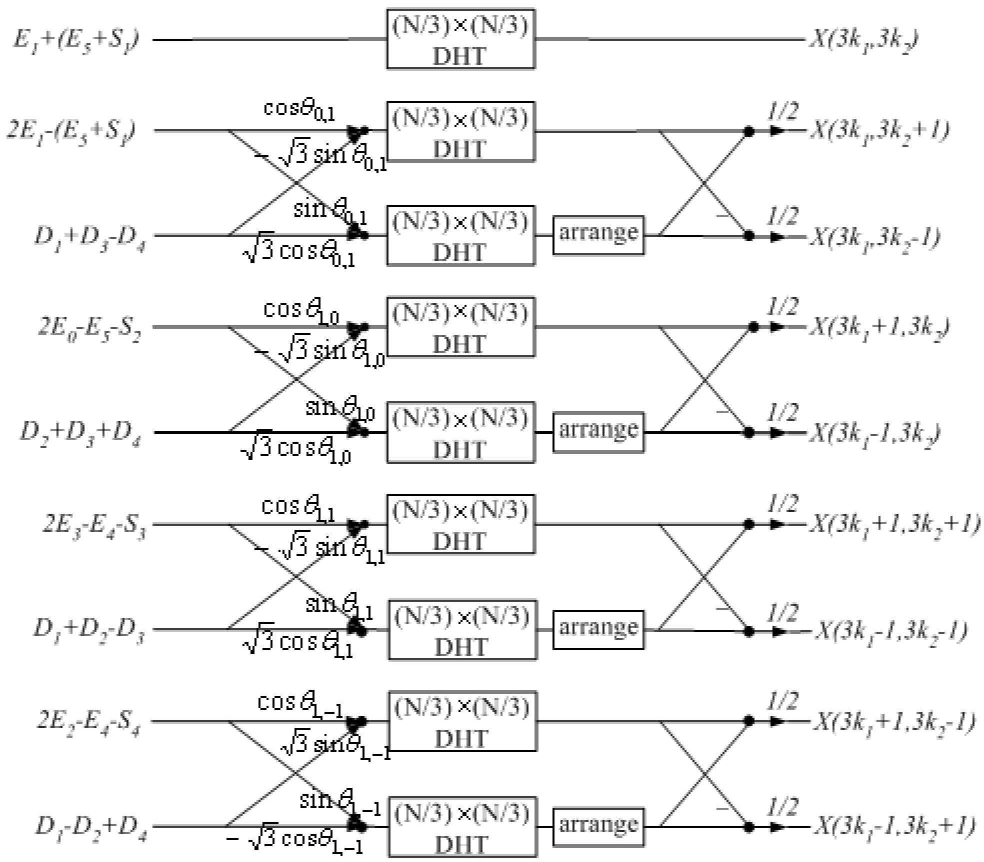

Fig. 1. Radix-3×3 algorithm for the 2-D DHT
“arrange” means the time-reversal operation of the sequence

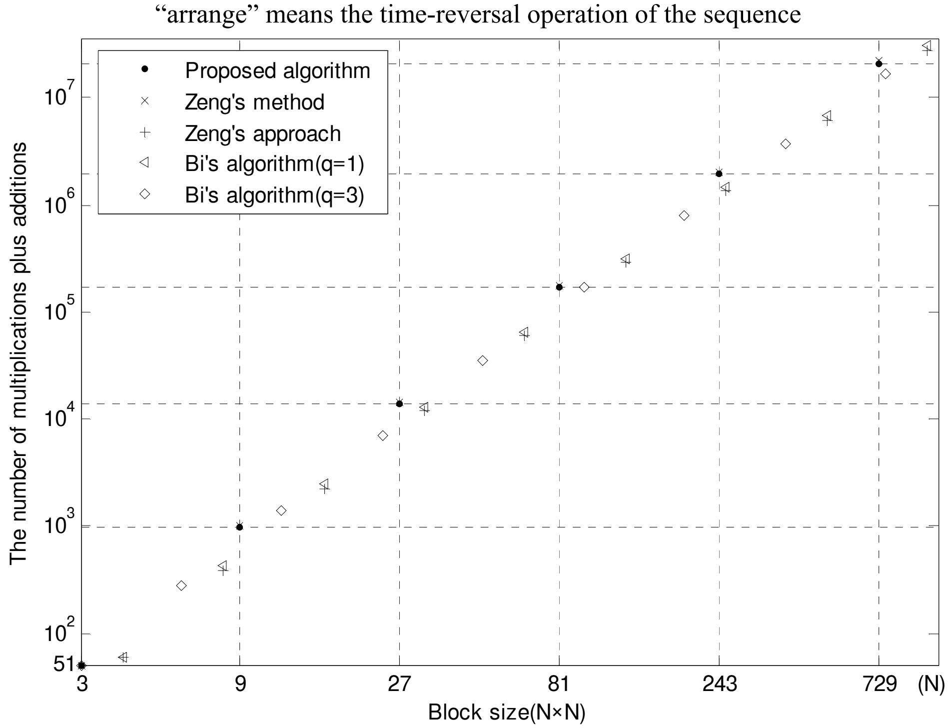

Fig.2. Comparison among the proposed algorithm, Bi’s algorithm and Zeng’s algorithm in terms of multiplications plus additions.