
Article:
Boussakta, S., Alshibami, O.H. and Aziz, M.Y. (2001) Radix-2 x 2 x 2 algorithm for the 3-D discrete hartley transform. IEEE Transactions on Signal Processing, 49 (12). pp. 3145-3156. ISSN: 1053-587X

https://doi.org/10.1109/78.969521

Radix-2 × 2 × 2 Algorithm for the 3-D Discrete Hartley Transform

Said Boussakta, *Member, IEEE*, Osama Hamoud Alshibami, *Student Member, IEEE*, and Mohammed Yunis Aziz, *Student Member, IEEE*

***Abstract*—The discrete Hartley transform (DHT) has proved to be a valuable tool in digital signal/image processing and communications and has also attracted research interests in many multidimensional applications. Although many fast algorithms have been developed for the calculation of one- and two-dimensional (1-D and 2-D) DHT, the development of multidimensional algorithms in three and more dimensions is still unexplored and has not been given similar attention; hence, the multidimensional Hartley transform is usually calculated through the row-column approach. However, proper multidimensional algorithms can be more efficient than the row-column method and need to be developed. Therefore, it is the aim of this paper to introduce the concept and derivation of the three-dimensional (3-D) radix-2 × 2 × 2 algorithm for fast calculation of the 3-D discrete Hartley transform. The proposed algorithm is based on the principles of the divide-and-conquer approach applied directly in 3-D. It has a simple butterfly structure and has been found to offer significant savings in arithmetic operations compared with the row-column approach based on similar algorithms.**

***Index Terms*—3-D filtering, 3-D Hartley transform, 3-D image processing, 3-D radix-2 × 2 × 2, 3-D spectrum analysis.**

I. Introduction

THE discrete Hartley transform (DHT) [1]–[13] has been used in many applications in signal/image processing and communications. These applications include filtering [14], adaptive digital filters [15], spectrum analysis [16], error control coding [17], multicarrier-based modulation [18], geophysical applications [19], fast interpolation [20], and power quality assessment [21]. These applications have been increased to cover multidimensional applications such as motion analysis [22], image processing [23], multidimensional filtering [24], multidimensional spectrum analysis and dose calculation in radionuclide therapy [25], etc.

The multidimensional discrete Hartley transform (m-D DHT) is closely related to the multidimensional discrete Fourier transform (m-D DFT) and has been proposed as an alternative tool suitable for real data [13]. The three-dimensional (3-D) DHT has the main properties of the 3-D DFT. The advantages of the m-D DHT over the m-D DFT are that it is a real-to-real transform and that the forward and inverse transforms are similar. Therefore, the m-D DHT is more suitable for 3-D image and multidimensional signal processing applications with real input data.

Although several algorithms have been developed for the one-dimensional (1-D) [1], [2], [4], [8], [9] and two-dimensional (2-D) [9]–[12] Hartley transforms, these algorithms cannot be applied directly to the calculation of the m-D DHT in a row-column approach because the m-D DHT is not separable [12], [13]. Hao and Bracewell have proposed the computation of the 3-D discrete Hartley transform using 1-D algorithms applied over each dimension to give an intermediate transform; the 3-D Hartley transform is then calculated from the intermediate transform at the expense of more additions and multiplications [13]. Boussakta and Holt proposed the calculation of m-D DHT using an index mapping scheme and the multiplication-free 1-D Fermat number transforms FNTs [26]. Meher *et al.* [28] proposed another algorithm involving the calculation of both 1-D DHT and 1-D DFT combined with prime factor and Winograd algorithms. Bortfield proposed the calculation of the m-D DHT using the 1-D complex Fourier transform [29].

However, all those papers reorder the input and map the 3-D problem into 1-D and then use other 1-D transforms for the calculation of the multidimensional Hartley transform. Thus, the development of fast Hartley transform algorithms in 3-D and more dimensions is still unexplored and has not been given the attention that has been given to the 1-D and 2-D transforms.

It is the aim of this paper to introduce the concept and derivation of the 3-D radix-2 × 2 × 2 algorithm for fast calculation of the 3-D DHT. The proposed algorithm has a simple butterfly structure and can be implemented in place. Compared with the commonly used row-column approach based on similar algorithms, the radix-2 × 2 × 2 is found to offer significant savings in the number of arithmetic operations. An example is given showing the validity of this algorithm.

II. Transform Definition and Properties

The 3-D DHT is a real-to-real transform and has the same inverse as given below.

Manuscript received March 6, 2000; revised August 24, 2001. This work was supported by the EPSRC under Grant GRM/M42060/02. The associate editor coordinating the review of this paper and approving it for publication was Dr. Xiang-Gen Xia.

The authors are with the Institute of Integrated Information Systems, School of Electronic and Electrical Engineering, University of Leeds, Leeds, U.K. (e-mail: s.boussakta@ee.leeds.ac.uk; eenoha@leeds.ac.uk; eenma@ee.leeds.ac.uk).

Publisher Item Identifier S 1053-587X(01)10500-3.

A. Transform Definition

The 3-D DHT of the 3-D input data $x(n_1, n_2, n_3)$ of size $N_1 \times N_2 \times N_3$ is defined as [1]

$$X(k_1,k_2,k_3) = \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1}\sum_{n_3=0}^{N_3-1} x(n_1,n_2,n_3) \times \operatorname{cas}\left(\frac{2\pi}{N_1}n_1k_1 + \frac{2\pi}{N_2}n_2k_2 + \frac{2\pi}{N_3}n_3k_3\right) \quad k_i = 0,1,\ldots,N_i-1; \quad i=1,2,3 \tag{1}$$

where $\operatorname{cas}(\alpha) = \cos(\alpha) + \sin(\alpha)$, and $\alpha = (2\pi/N_1)n_1k_1 + (2\pi/N_2)n_2k_2 + (2\pi/N_3)n_3k_3$.

The inverse transform is

$$x(n_1,n_2,n_3) = \frac{1}{N_1N_2N_3}\sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1}\sum_{k_3=0}^{N_3-1} X(k_1,k_2,k_3) \times \operatorname{cas}\left(\frac{2\pi}{N_1}n_1k_1 + \frac{2\pi}{N_2}n_2k_2 + \frac{2\pi}{N_3}n_3k_3\right) \quad n_i = 0,1,\ldots,N_i-1; \quad i=1,2,3. \tag{2}$$

The factor $1/(N_1N_2N_3)$ can be split between the forward and inverse transforms to make them exactly the same.

B. Some Properties of the 3-D Hartley Transform

The 3-D DHT can be applied in 3-D spectrum analysis and in many applications in image and multidimensional signal processing [22]–[25]. It has the main properties of the 3-D DFT, but it is a real-to-real transform and, hence, is more suitable for 3-D and multidimensional applications when the input data is real.

1) Calculation of 3-D Convolution: The 3-D discrete Hartley transform has the 3-D cyclic convolution property and, hence, can be used to calculate the 3-D convolution/correlation and related functions for 3-D applications. The convolution property defined in [1] and [2] can be extended to 3-D. If we consider that $x(n_1, n_2, n_3)$ and $h(n_1, n_2, n_3)$ are the 3-D input data and the 3-D impulse filter to have sizes equal to $M_1 \times M_2 \times M_3$ and $L_1 \times L_2 \times L_3$, respectively, then their linear convolution $y(n_1, n_2, n_3)$ is of size $N_1 \times N_2 \times N_3 = (M_1+L_1-1)\times(M_2+L_2-1)\times(M_3+L_3-1)$ with M_i, L_i and N_i being integers and $i = 1,2,3$. The two input data need to be padded with zeros to $y(n_1, n_2, n_3)$ size in order to avoid wrap around errors. The padded inputs are then used to calculate the 3-D linear convolution as follows:

$$\begin{aligned} y(n_1,n_2,n_3) &= x(n_1,n_2,n_3) ***h(n_1,n_2,n_3) \\ &= \text{3-D Inverse DHT}\{[X(k_1,k_2,k_3) \otimes\otimes\otimes H_{\text{ev}}(k_1,k_2,k_3)] \\ &\quad + [X(N_1-k_1,N_2-k_2,N_3-k_3) \otimes\otimes\otimes H_{\text{od}}(N_1-k_1,N_2-k_2,N_3-k_3)]\} \end{aligned} \tag{3}$$

where $***$ is the symbol for 3-D linear convolution, and $\otimes\otimes\otimes$ stands for 3-D point by point multiplication. $H_{\text{ev}}(\cdot,\cdot,\cdot)$ and $H_{\text{od}}(\cdot,\cdot,\cdot)$ refer to even and odd parts of $H(\cdot,\cdot,\cdot)$, respectively and are given by

$$H_{\text{ev}}(k_1,k_2,k_3) = [H(k_1,k_2,k_3) + H(N_1-k_1,N_2-k_2,N_3-k_3)]/2 \tag{4}$$

$$H_{\text{od}}(k_1,k_2,k_3) = [H(k_1,k_2,k_3) - H(N_1-k_1,N_2-k_2,N_3-k_3)]/2. \tag{5}$$

An application of the 3-D convolution in medical image processing for the dose calculation in radionuclide therapy is demonstrated in [25]. The 3-D convolution is carried out using the 3-D DHT. The 3-D DHT is calculated via the row-column approach using the 1-D radix-2 FHT.

2) Relationship Between 3-D DHT and 3-D DFT: The 3-D DHT is closely related to the 3-D DFT, where the real and imaginary parts of the 3-D DFT are equal to the even part and the negative of the odd part of the 3-D DHT, respectively.

$$F(k_1,k_2,k_3) = \left[\frac{H(k_1,k_2,k_3)+H(-k_1,-k_2,-k_3)}{2} - j\,\frac{H(k_1,k_2,k_3)-H(-k_1,-k_2,-k_3)}{2}\right] \tag{6}$$

where $F(k_1, k_2, k_3)$ is the 3-D Fourier transform, and $H(k_1, k_2, k_3)$ is the 3-D Hartley transform. Therefore, we can easily calculate the 3-D complex Fourier transform once the 3-D Hartley transform is calculated.

III. Fast Algorithm for the 3-D Discrete Hartley Transform

Although many algorithms have been developed for fast calculation of the 1-D and 2-D DHTs [1], [2], [4], [8]–[13], algorithm development for m-D DHT in three and more dimensions has not been given similar attention. Hence, the m-D DHT is usually calculated using the row-column approach. However, true multidimensional algorithms can be more efficient than the row-column approach and need to be developed.

In this paper, the development and derivation of a 3-D radix-$2 \times 2 \times 2$ algorithm is introduced, which calculates the 3-D DHT directly. The transform size should be a power of $2 \times 2 \times 2$. In this algorithm, the 3-D Hartley of size $N \times N \times N$-point is divided into eight $N/2 \times N/2 \times N/2$-point 3-D DHTs. In the next stage of the algorithm, each $N/2 \times N/2 \times N/2$-point 3-D DHT is further divided into eight $N/4 \times N/4 \times N/4$-point 3-D DHTs, and the process continues until we get $2 \times 2 \times 2$ transforms. Hence, this algorithm is based on divide-and-conquer procedure applied in three dimensions. For simplicity and without loss of generality, let $N_1 = N_2 = N_3 = N$; then, $X(k_1, k_2, k_3)$ can be written as

$$X(k_1,k_2,k_3) = \sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1}\sum_{n_3=0}^{N-1} x(n_1,n_2,n_3)$$

$$\times \operatorname{cas}\left(\frac{2\pi}{N}(n_1k_1+n_2k_2+n_3k_3)\right)$$
$$= \sum_{n_1:\text{even}}\sum_{n_2:\text{even}}\sum_{n_3:\text{even}} + \sum_{n_1:\text{even}}\sum_{n_2:\text{even}}\sum_{n_3:\text{odd}}$$
$$+ \sum_{n_1:\text{even}}\sum_{n_2:\text{odd}}\sum_{n_3:\text{even}} + \sum_{n_1:\text{even}}\sum_{n_2:\text{odd}}\sum_{n_3:\text{odd}}$$
$$+ \sum_{n_1:\text{odd}}\sum_{n_2:\text{even}}\sum_{n_3:\text{even}} + \sum_{n_1:\text{odd}}\sum_{n_2:\text{even}}\sum_{n_3:\text{odd}}$$
$$+ \sum_{n_1:\text{odd}}\sum_{n_2:\text{odd}}\sum_{n_3:\text{even}} + \sum_{n_1:\text{odd}}\sum_{n_2:\text{odd}}\sum_{n_3:\text{odd}}. \quad (7)$$

Therefore, the general decomposition formula is

$$\begin{aligned}
&X(k_1,k_2,k_3)\\
&= D_{000}(k_1,k_2,k_3) + \left[D_{001}(k_1,k_2,k_3)\cos\frac{2\pi k_3}{N}\right.\\
&\quad \left. + D_{001}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_3}{N}\right]\\
&\quad + \left[D_{010}(k_1,k_2,k_3)\cos\frac{2\pi k_2}{N}\right.\\
&\quad \left. + D_{010}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_2}{N}\right]\\
&\quad + \left[D_{011}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_2+k_3)\right.\\
&\quad + D_{011}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\\
&\quad \left.\times \sin\frac{2\pi}{N}(k_2+k_3)\right]\\
&\quad + \left[D_{100}(k_1,k_2,k_3)\cos\frac{2\pi k_1}{N}\right.\\
&\quad + D_{100}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\\
&\quad \left.\times \sin\frac{2\pi k_1}{N}\right]\\
&\quad + \left[D_{101}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_3)\right.\\
&\quad + D_{101}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\\
&\quad \left.\times \sin\frac{2\pi}{N}(k_1+k_3)\right]\\
&\quad + \left[D_{110}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2)\right.\\
&\quad + D_{110}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\\
&\quad \left.\times \sin\frac{2\pi}{N}(k_1+k_2)\right]\\
&\quad + \left[D_{111}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2+k_3)\right.\\
&\quad + D_{111}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\\
&\quad \left.\times \sin\frac{2\pi}{N}(k_1+k_2+k_3)\right] \quad (8)
\end{aligned}$$

where

$$\begin{aligned}
&D_{\alpha\beta\delta}(k_1,k_2,k_3)\\
&\quad = \sum_{n_1=0}^{N/2-1}\sum_{n_2=0}^{N/2-1}\sum_{n_3=0}^{N/2-1} x(2n_1+\alpha,2n_2+\beta,2n_3+\delta)\\
&\qquad \times \operatorname{cas}\left(\frac{2\pi}{N}(2n_1k_1+2n_2k_2+2n_3k_3)\right)\\
&\alpha\beta\delta = \{000,\ 001,\ 010,\ 011,\ 100,\ 101,\ 110,\ 111\} \quad (9)
\end{aligned}$$

and the sums are recognized as eight $((N/2)\times(N/2)\times(N/2))$-point 3-D DHTs. The process is repeated until only $(2\times2\times2)$-point 3-D DHTs remain. A single butterfly is illustrated in Fig. 1, which simultaneously computes $X(k_1,k_2,k_3), X(k_1,k_2,k_3+N/2), X(k_1,k_2+N/2,k_3), X(k_1,k_2+N/2,k_3+N/2), X(k_1+N/2,k_2,k_3), X(k_1+N/2,k_2,k_3+N/2), X(k_1+N/2,k_2+N/2,k_3)$, and $X(k_1+N/2,k_2+N/2,k_3+N/2)$ as follows:

$$\begin{aligned}
&X\left(k_1,k_2,k_3+\frac{N}{2}\right)\\
&\quad = D_{000}(k_1,k_2,k_3) - \left[D_{001}(k_1,k_2,k_3)\cos\frac{2\pi k_3}{N}\right.\\
&\qquad \left. + D_{001}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_3}{N}\right]\\
&\qquad + \left[D_{010}(k_1,k_2,k_3)\cos\frac{2\pi k_2}{N}\right.\\
&\qquad \left. + D_{010}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_2}{N}\right]\\
&\qquad - \left[D_{011}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_2+k_3)\right.\\
&\qquad \left. + D_{011}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_2+k_3)\right]\\
&\qquad + \left[D_{100}(k_1,k_2,k_3)\cos\frac{2\pi k_1}{N}\right.\\
&\qquad \left. + D_{100}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_1}{N}\right]\\
&\qquad - \left[D_{101}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_3)\right.\\
&\qquad \left. + D_{101}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_3)\right]\\
&\qquad + \left[D_{110}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2)\right.\\
&\qquad \left. + D_{110}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_2)\right]\\
&\qquad - \left[D_{111}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2+k_3)\right.\\
&\qquad + D_{111}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\\
&\qquad \left.\times \sin\frac{2\pi}{N}(k_1+k_2+k_3)\right] \quad (10)\\
&X\left(k_1,k_2+\frac{N}{2},k_3\right)\\
&\quad = D_{000}(k_1,k_2,k_3) + \left[D_{001}(k_1,k_2,k_3)\cos\frac{2\pi k_3}{N}\right.
\end{aligned}$$

$$
\begin{aligned}
&+ D_{001}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_3}{N}\Big] \\
&-\Big[D_{010}(k_1,k_2,k_3)\cos\frac{2\pi k_2}{N} \\
&+ D_{010}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_2}{N}\Big] \\
&-\Big[D_{011}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_2+k_3) \\
&+ D_{011}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2}{N}(k_2+k_3)\Big] \\
&+\Big[D_{100}(k_1,k_2,k_3)\cos\frac{2\pi k_1}{N} \\
&+ D_{100}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_1}{N}\Big] \\
&+\Big[D_{101}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_3) \\
&+ D_{101}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_3)\Big] \\
&-\Big[D_{110}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2) \\
&+ D_{110}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_2)\Big] \\
&-\Big[D_{111}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2+k_3) \\
&+ D_{111}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right) \\
&\times\sin\frac{2\pi}{N}(k_1+k_2+k_3)\Big] \qquad (11)
\end{aligned}
$$

$$
\begin{aligned}
&X\left(k_1,k_2+\frac{N}{2},k_3+\frac{N}{2}\right) \\
&= D_{000}(k_1,k_2,k_3)-\Big[D_{001}(k_1,k_2,k_3)\cos\frac{2\pi k_3}{N} \\
&+ D_{001}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_3}{N}\Big] \\
&-\Big[D_{010}(k_1,k_2,k_3)\cos\frac{2\pi k_2}{N} \\
&+ D_{010}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_2}{N}\Big] \\
&+\Big[D_{011}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_2+k_3) \\
&+ D_{011}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_2+k_3)\Big] \\
&+\Big[D_{100}(k_1,k_2,k_3)\cos\frac{2\pi k_1}{N} \\
&+ D_{100}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_1}{N}\Big] \\
&-\Big[D_{101}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_3) \\
&+ D_{101}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_3)\Big] \\
&-\Big[D_{110}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2) \\
&+ D_{110}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_2)\Big] \\
&+\Big[D_{111}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2+k_3) \\
&+ D_{111}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right) \\
&\times\sin\frac{2\pi}{N}(k_1+k_2+k_3)\Big] \qquad (12)
\end{aligned}
$$

$$
\begin{aligned}
&X\left(k_1+\frac{N}{2},k_2,k_3\right) \\
&= D_{000}(k_1,k_2,k_3)+\Big[D_{001}(k_1,k_2,k_3)\cos\frac{2\pi k_3}{N} \\
&+ D_{001}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_3}{N}\Big] \\
&+\Big[D_{010}(k_1,k_2,k_3)\cos\frac{2\pi k_2}{N} \\
&+ D_{010}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_2}{N}\Big] \\
&+\Big[D_{011}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_2+k_3) \\
&+ D_{011}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_2+k_3)\Big] \\
&-\Big[D_{100}(k_1,k_2,k_3)\cos\frac{2\pi k_1}{N} \\
&+ D_{100}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_1}{N}\Big] \\
&-\Big[D_{101}(k_1,k_2,k_3)\cos\frac{2}{N}(k_1+k_3) \\
&+ D_{101}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_3)\Big] \\
&-\Big[D_{110}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2) \\
&+ D_{110}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_2)\Big] \\
&-\Big[D_{111}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2+k_3) \\
&+ D_{111}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right) \\
&\times\sin\frac{2\pi}{N}(k_1+k_2+k_3)\Big] \qquad (13)
\end{aligned}
$$

$$
\begin{aligned}
&X\left(k_1+\frac{N}{2},k_2,k_3+\frac{N}{2}\right) \\
&= D_{000}(k_1,k_2,k_3)-\Big[D_{001}(k_1,k_2,k_3)\cos\frac{2\pi k_3}{N} \\
&+ D_{001}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_3}{N}\Big] \\
&+\Big[D_{010}(k_1,k_2,k_3)\cos\frac{2\pi k_2}{N} \\
&+ D_{010}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_2}{N}\Big] \\
&-\Big[D_{011}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_2+k_3)
\end{aligned}
$$

$$
\begin{aligned}
&+ D_{011}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_2+k_3)\Big]\\
&-\Big[D_{100}(k_1,k_2,k_3)\cos\frac{2\pi k_1}{N}\\
&+ D_{100}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_1}{N}\Big]\\
&+\Big[D_{101}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_3)\\
&+ D_{101}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_3)\Big]\\
&-\Big[D_{110}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2)\\
&+ D_{110}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_2)\Big]\\
&+\Big[D_{111}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2+k_3)\\
&+ D_{111}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\\
&\times\sin\frac{2\pi}{N}(k_1+k_2+k_3)\Big] \qquad (14)
\end{aligned}
$$

$$
\begin{aligned}
&X\left(k_1+\frac{N}{2},k_2+\frac{N}{2},k_3\right)\\
&= D_{000}(k_1,k_2,k_3)+\Big[D_{001}(k_1,k_2,k_3)\cos\frac{2\pi k_3}{N}\\
&+ D_{001}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_3}{N}\Big]\\
&-\Big[D_{010}(k_1,k_2,k_3)\cos\frac{2\pi k_2}{N}\\
&+ D_{010}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_2}{N}\Big]\\
&-\Big[D_{011}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_2+k_3)\\
&+ D_{011}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_2+k_3)\Big]\\
&-\Big[D_{100}(k_1,k_2,k_3)\cos\frac{2\pi k_1}{N}\\
&+ D_{100}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_1}{N}\Big]\\
&-\Big[D_{101}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_3)\\
&+ D_{101}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_3)\Big]\\
&+\Big[D_{110}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2)\\
&+ D_{110}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_2)\Big]\\
&+\Big[D_{111}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2+k_3)\\
&+ D_{111}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\\
&\sin\frac{2\pi}{N}(k_1+k_2+k_3)\Big] \qquad (15)
\end{aligned}
$$

$$
\begin{aligned}
&X\left(k_1+\frac{N}{2},k_2+\frac{N}{2},k_3+\frac{N}{2}\right)\\
&= D_{000}(k_1,k_2,k_3)-\Big[D_{001}(k_1,k_2,k_3)\cos\frac{2\pi k_3}{N}\\
&+ D_{001}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_3}{N}\Big]\\
&-\Big[D_{010}(k_1,k_2,k_3)\cos\frac{2\pi k_2}{N}\\
&+ D_{010}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_2}{N}\Big]\\
&+\Big[D_{011}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_2+k_3)\\
&+ D_{011}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_2+k_3)\Big]\\
&-\Big[D_{100}(k_1,k_2,k_3)\cos\frac{2\pi k_1}{N}\\
&+ D_{100}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi k_1}{N}\Big]\\
&+\Big[D_{101}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_3)\\
&+ D_{101}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_3)\Big]\\
&+\Big[D_{110}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2)\\
&+ D_{110}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\frac{2\pi}{N}(k_1+k_2)\Big]\\
&-\Big[D_{111}(k_1,k_2,k_3)\cos\frac{2\pi}{N}(k_1+k_2+k_3)\\
&+ D_{111}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\\
&\times\sin\frac{2\pi}{N}(k_1+k_2+k_3)\Big]. \qquad (16)
\end{aligned}
$$

To prove the validity of this algorithm, an example of an $8\times8\times8$ 3-D Hartley transform for a random 3-D input and its inverse calculated using the 3-D radix-$2\times2\times2$ algorithm is shown in Fig. 2. A C-program for the 3-D radix-$2\times2\times2$ algorithm is available upon request from the authors.

IV. Arithmetic Complexity and Comparison With Existing Algorithms

In this section, the arithmetic complexity of this algorithm is analyzed and compared with the most commonly used row-column approach based on similar algorithms.

A. Arithmetic Complexity of the Radix-2 × 2 × 2

The butterfly shown in Fig. 1 calculates eight points and needs 14 real multiplications and 31 real additions. The whole transform needs $\log_2 N$ stages. For an $(N\times N\times N)$ point 3-D FHT, the total number of real multiplications needed using one butterfly and $4/2$ implementation is $(7/4)N^3\log_2 N$, and the total number of real additions is $(31/8)N^3\log_2 N$, as shown in Table I. Using different butterflies to remove trivial multiplications and additions and reduce the arithmetic operations further, the total number of real multiplications and additions

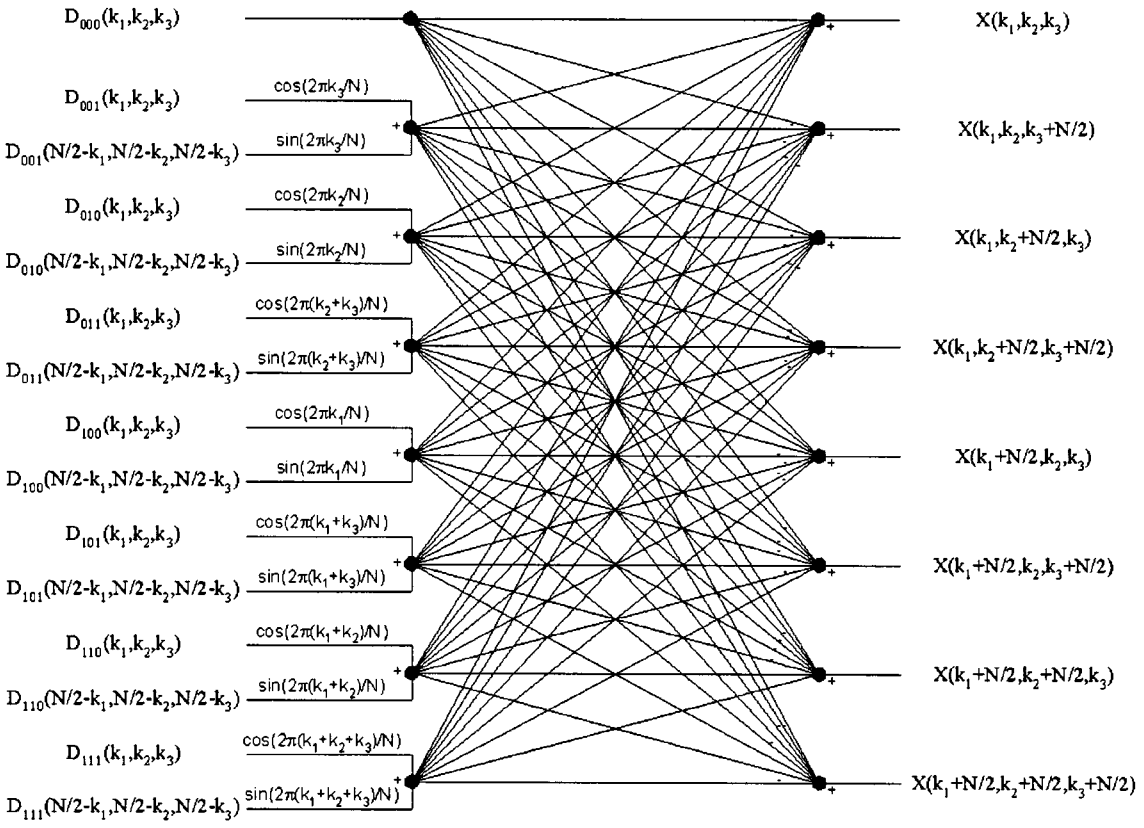

Fig. 1. One butterfly radix-2 × 2 × 2 algorithm for the 3-D Hartley transform.

can be reduced to $(7/4)N^3\log_2 N - (49/8)N^3 + (21/2)N^2$ and $(31/8)N^3\log_2 N - (21/8)N^3 + (7/2)N^2$, respectively, as shown in Table II.

B. Comparison With Existing Algorithms

In this section, we first compare this algorithm with the most commonly used row-column approach based on 1-D radix-2, 1-D split radix using $4/2$ implementation, 1-D hybrid DHT/FFT using $3/3$ implementation, and then with known papers for 3-D DHT.

1) Comparison With the Row-Column Approach Based on Radix-2 Using $4/2$ Implementation: Owing to the difficulty of developing fast algorithms in three and more dimensions, the 3-D DHT is usually computed using the row-column approach [12], [13], [25]. The 3-D DHT is not separable ($\text{cas}(m+n+s) \neq \text{cas}(m)\text{cas}(n)\text{cas}(s)$), and hence, the multidimensional Hartley transform, in three and more dimensions, is usually computed by adding a certain number of temporary arrays; these arrays are computed using several 1-D FHTs applied over each dimension [12], [13], [25]. The multidimensional Hartley transform is then computed from the temporary arrays at the expense of some extra additions and halvings (multiplication by 0.5). Ignoring the halvings, the number of real multiplications required to calculate an $N\times N\times N$ 3-D DHT, using the 1-D radix-2 algorithm in a row-column approach, is $3N^3\log_2 N$, and the total number of real additions is $(9/2)N^3\log_2 N + 3N^3$. Using different butterflies to remove trivial operations and reduce the arithmetic complexity further, the total number of real multiplications and additions can be reduced to $3N^3\log_2 N - (21/2)N^3 + 18N^2$ and $(9/2)N^3\log_2 N - (9/2)N^3 + 6N^2 + 3N^3$, respectively. Tables I and II show a comparison between the row-column approach and the 3-D radix-2 × 2 × 2 algorithm, in terms of numbers of multiplications and additions per point, using a single and multiple butterflies.

From Tables I and II, it is clear that great savings in the number of multiplications and additions can be achieved when using the 3-D radix-2 × 2 × 2 algorithm. In counting the arithmetic operations in this section, we used the so-called $4/2$ implementation. Depending on the type of system in use, a certain number of multiplications can be traded for additions using the so-called $3/3$ implementation. This can be done for both algorithms and saves approximately one multiplication in four but increases the number of additions by the same amount, and hence, we have omitted the tables for $3/3$ in this section.

In general, the number of real multiplications and additions required for the multidimensional (m-D) Hartley transform using the radix-2 × 2 × 2 and $4/2$ implementation are approximately

$$\text{Mults}_1 = \frac{2^m - 1}{2^{m-1}} N^m \log_2 N \tag{17}$$

and

$$\text{Adds}_1 = \left(m + \frac{2^m - 1}{2^m}\right) \times N^m \log_2 N. \tag{18}$$

On the other hand, the number of real multiplications and additions for m-D FHT using the row-column approach based on radix-2 and $4/2$ implementation are

$$\text{Mults}_2 = mN^m \log_2 N \tag{19}$$

and

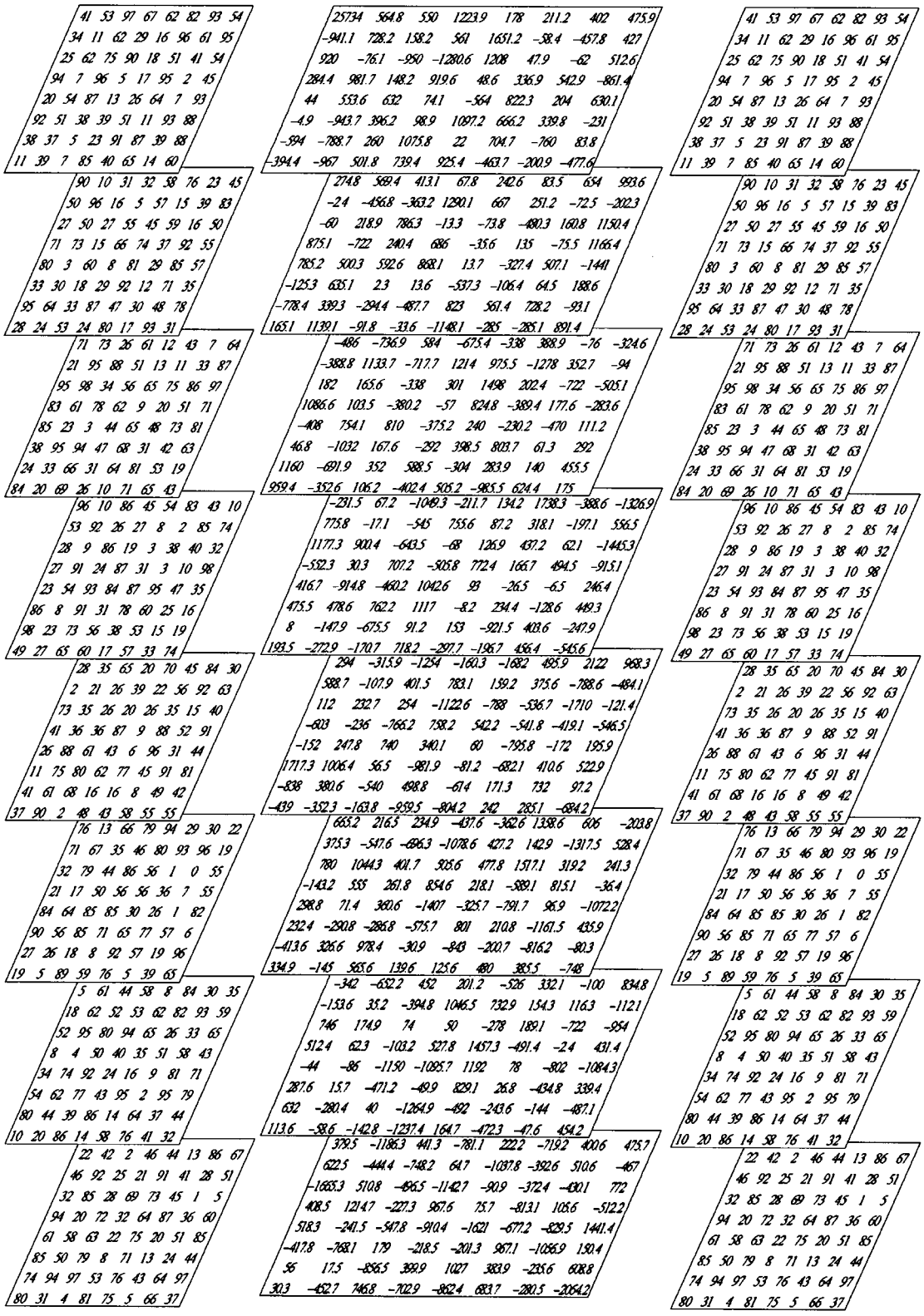

Fig. 2. Forward and inverse 3-D Hartley transform. (a) Three-dimensional input image. (b) 3-D forward Hartley transform. (c) 3-D inverse Hartley transform.

$$\text{Adds}_2 = \frac{3}{2} m N^m \log_2 N + m N^m. \tag{20}$$

Therefore, using the 3-D radix-$2 \times 2 \times 2$ will save approximately

$$S_{\text{Mults}}\% = 100 \times \left(1 - \frac{2^m - 1}{m 2^{m-1}}\right) \tag{21}$$

and

$$S_{\text{Adds}}\% = 100 \times \left(1 - \frac{2\left(m + \frac{2^m - 1}{2^m}\right) \times \log_2 N}{3m \log_2 N + 2m}\right). \tag{22}$$

The savings in the number of multiplications and additions increases with the increase of the transform dimensions, as shown

TABLE I
COMPARISON BETWEEN THE ROW-COLUMN APPROACH AND THE 3-D RADIX-2 × 2 × 2 USING ONE-BUTTERFLY AND 4/2 IMPLEMENTATION

Transform Size	Row-column approach (based on radix-2 algorithm)			3-D radix-2×2×2 approach		
	Mults. /point	Adds. /point	Mults.+Adds. /point	Mults. /point	Adds. /point	Mults.+Adds. /point
$2^3\times2^3\times2^3$	9	16.5	25.5	5.25	11.625	16.875
$2^4\times2^4\times2^4$	12	21	33	7	15.5	22.5
$2^5\times2^5\times2^5$	15	25.5	40.5	8.75	19.375	28.125
$2^6\times2^6\times2^6$	18	30	48	10.5	23.25	33.75
$2^7\times2^7\times2^7$	21	34.5	55.5	12.25	27.125	39.375
$2^8\times2^8\times2^8$	24	39	63	14	31	45
$2^9\times2^9\times2^9$	27	43.5	70.5	15.75	34.875	50.625
$2^{10}\times2^{10}\times2^{10}$	30	48	78	17.5	38.75	56.25
$2^{11}\times2^{11}\times2^{11}$	33	52.5	85.5	19.25	42.625	61.875
$2^{12}\times2^{12}\times2^{12}$	36	57	93	21	46.5	67.5

TABLE II
COMPARISON BETWEEN THE ROW-COLUMN APPROACH AND THE 3-D RADIX-2 × 2 × 2 USING 4/2 IMPLEMENTATION AFTER REDUCING THE ARITHMETIC OPERATIONS USING MULTIPLE BUTTERFLIES

Transform Size	Row-column approach (based on radix-2 algorithm)			3-D radix-2×2×2 approach		
	Mults. /point	Adds. /point	Mults.+Adds. /point	Mults. /point	Adds. /point	Mults.+Adds. /point
$2^3\times2^3\times2^3$	0.75	12.75	13.5	0.4375	9.437	9.8755
$2^4\times2^4\times2^4$	2.625	16.875	19.5	1.5313	13.094	14.6253
$2^5\times2^5\times2^5$	5.0625	21.1875	26.25	2.9531	16.860	19.8121
$2^6\times2^6\times2^6$	7.7813	25.5938	33.3751	4.5391	20.680	25.2191
$2^7\times2^7\times2^7$	10.6406	30.0469	40.6875	6.207	24.527	30.734
$2^8\times2^8\times2^8$	13.5703	34.5234	48.0937	7.916	28.389	36.305
$2^9\times2^9\times2^9$	16.5352	39.0117	55.5469	9.6455	32.257	41.9025
$2^{10}\times2^{10}\times2^{10}$	19.5176	43.5059	63.0235	11.3853	36.128	47.5133
$2^{11}\times2^{11}\times2^{11}$	22.5088	48.0029	70.5117	13.1301	40.0017	53.1318
$2^{12}\times2^{12}\times2^{12}$	25.5044	52.5015	78.0059	14.8776	43.8759	58.7535

in Table III. However, the complexity of developing multidimensional algorithms increases with the transform dimension as well, but the result needs to be developed only once, and the savings achieved make the calculation worthwhile.

2) Comparison With the Row-Column Approach Based on Split Radix Using 4/2 Implementation: The 1-D split radix algorithm is known to be one of the best 1-D FHT algorithms for reducing the number of multiplications and additions [2], [14]. However, it has a more complex structure and indexing scheme than other methods, and hence, the arithmetic advantage could not be turned into a similar speed advantage for real implementations [3], [5]–[7]. Therefore, it is difficult to make this comparison, and it would perhaps be better to compare the row-column approach based on 1-D split radix with the 3-D split radix Hartley transform algorithm. However, the 3-D split radix Hartley transform algorithm has yet to be developed, and hence, we will use the radix-2 × 2 × 2 results developed in this paper as a guide only. The number of arithmetic operations for the 1-D split radix Hartley transform using 4/2 implementation and multiple butterflies to remove trivial operations is reported and programmed in [2]. The total number of multiplications needed to calculate the 3-D DHT using this algorithm is

$$2N^3\log_2 N-(19/3)N^3+9N^2+(N^2/3)(-1)^{\log_2 N}$$

and the total number of additions is

$$4N^3\log_2 N-(5/3)N^3+9N^2+(5N^2/3)(-1)^{\log_2 N}.$$

TABLE III
SAVINGS WHEN USING THE 3-D RADIX-2 × 2 × 2 ALGORITHM OVER THE ROW-COLUMN APPROACH, IN PERCENTAGE AS A FUNCTION OF DIMENSIONS

Dimensions	3	4	5	6	7	8	9	10
S_{Mults}%	41.67	53.13	61.25	67.19	71.65	75.1	77.82	80.02
S_{Adds}% N=16	26.19	29.46	31.79	33.48	34.76	35.74	36.52	37.15
S_{Adds}% N=32	24.02	27.39	29.78	31.53	32.84	33.85	34.65	35.3
S_{Adds}% N=64	22.5	25.94	28.38	30.16	31.5	32.53	33.35	34.01
S_{Adds}% N=128	21.38	24.86	27.34	29.14	30.5	31.55	32.38	33.05
S_{Adds}% N=256	20.51	24.04	26.54	28.37	29.74	30.8	31.64	32.31

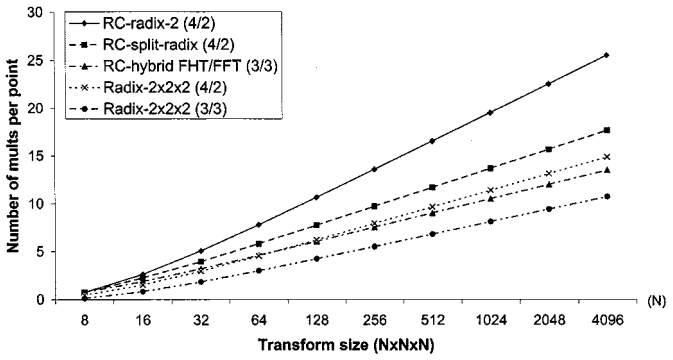

Fig. 3. Comparison between the 3-D radix-2 × 2 × 2 algorithm and the row-column (RC) approach based on radix-2 (RC-radix-2), split-radix (RC-split-radix) using $4/2$ implementation and hybrid FHT/FFT (RC-hybrid FHT/FFT) using $3/3$ implementation in terms of multiplications.

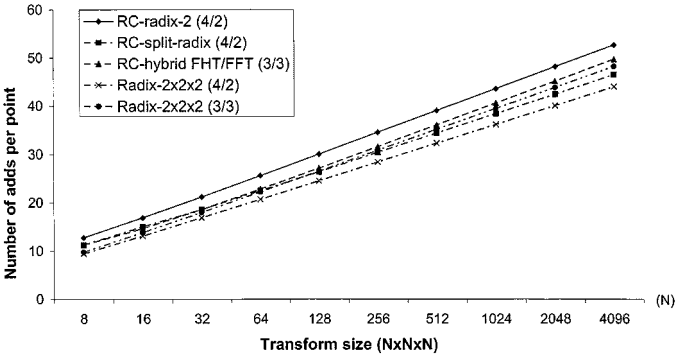

Fig. 4. Comparison between the 3-D radix-2 × 2 × 2 algorithm and the row-column (RC) approach based on radix-2 (RC-radix-2), split-radix (RC-split-radix) using $4/2$ implementation and hybrid FHT/FFT (RC-hybrid FHT/FFT) using $3/3$ implementation in terms of additions.

As shown in Figs. 3–5 and Table IV, the radix-2 × 2 × 2 involves less arithmetic operations than the row-column based split radix approach.

3) Comparison With the Row-Column Approach Based on Hybrid FHT/FFT Using $3/3$ *Implementation:* Depending on the type of system in use, a certain number of multiplications

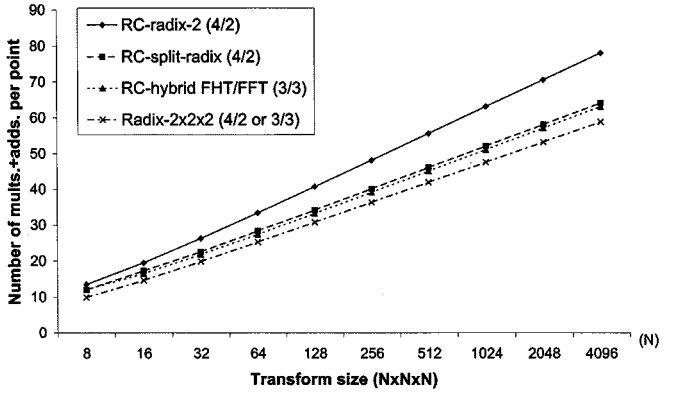

Fig. 5. Comparison between the 3-D radix-2 × 2 × 2 algorithm and the row-column (RC) approach based on radix-2 (RC-radix-2), split-radix (RC-split-radix) using $4/2$ implementation and hybrid FHT/FFT (RC-hybrid FHT/FFT) using $3/3$ implementation in terms of multiplications+additions.

can be traded for additions using the so-called $3/3$ implementation [4], [9], [14], [30]. This can saves approximately one multiplication in four but increases the number of additions by nearly the same amount. This can be beneficial on systems where multiplication is much slower than additions. On systems where the time required for additions and multiplications is equal, the $4/2$ implementation is faster than and more preferable to the $3/3$ implementation [6].

The hybrid FHT/FFT is one of the 1-D butterfly-style algorithms that achieves a very low arithmetic complexity [14]. Based on the hybrid FHT/FFT and related algorithms, the total number of arithmetic operations for the calculation of the 3-D DHT using the $3/3$ implementation is

$$(3/2)N^3\log_2 N - (9/2)N^3 + 6N^2 \text{ multiplications} \quad \text{and}$$
$$(9/2)N^3\log_2 N - (9/2)N^3 + 18N^2 \text{ additions.}$$

In addition, the total number of multiplications and additions for the radix-2 × 2 × 2 using the $3/3$ implementation is given as $(21/16)N^3\log_2 N - (161/32)N^3 + (77/8)N^2$ multiplications and $(69/16)N^3\log_2 N - (119/32)N^3 + (35/8)N^2$ additions.

TABLE IV
COMPARISON BETWEEN THE ROW-COLUMN APPROACH BASED ON SPLIT-RADIX AND THE 3-D RADIX-2 × 2 × 2 USING 4/2 IMPLEMENTATION AFTER REDUCING THE ARITHMETIC OPERATIONS USING MULTIPLE BUTTERFLIES

Transform Size	Row-column approach (based on split-radix algorithm)			3-D radix-2×2×2 approach		
	Mults. /point	Adds. /point	Mults.+Adds. /point	Mults. /point	Adds. /point	Mults.+Adds. /point
$2^3\times2^3\times2^3$	0.75	11.25	12	0.4375	9.437	9.8755
$2^4\times2^4\times2^4$	2.25	15	17.25	1.5313	13.094	14.6253
$2^5\times2^5\times2^5$	3.9375	18.5625	22.5	2.9531	16.860	19.8121
$2^6\times2^6\times2^6$	5.8125	22.5	28.3125	4.5391	20.680	25.2191
$2^7\times2^7\times2^7$	7.7344	26.3906	34.125	6.207	24.527	30.734
$2^8\times2^8\times2^8$	9.7031	30.375	40.0781	7.916	28.389	36.305
$2^9\times2^9\times2^9$	11.6836	34.3477	46.0313	9.6455	32.257	41.9025
$2^{10}\times2^{10}\times2^{10}$	13.6758	38.3438	52.0195	11.3853	36.128	47.5133
$2^{11}\times2^{11}\times2^{11}$	15.6709	42.3369	58.0078	13.1301	40.0017	53.1318
$2^{12}\times2^{12}\times2^{12}$	17.6689	46.3359	64.0049	14.8776	43.8759	58.7535

TABLE V
COMPARISON BETWEEN THE ROW-COLUMN APPROACH BASED ON HYBRID FHT/FFT AND THE 3-D RADIX-2 × 2 × 2 USING 3/3 IMPLEMENTATION AFTER REDUCING THE ARITHMETIC OPERATIONS USING MULTIPLE BUTTERFLIES

Transform Size	Row-column approach (based on hybrid FHT/FFT algorithm)			3-D radix-2×2×2 approach		
	Mults. /point	Adds. /point	Mults.+Adds. /point	Mults. /point	Adds. /point	Mults.+Adds. /point
$2^3\times2^3\times2^3$	0.75	11.25	12	0.109	9.766	9.875
$2^4\times2^4\times2^4$	1.875	14.625	16.5	0.82	13.805	14.625
$2^5\times2^5\times2^5$	3.188	18.563	21.751	1.832	17.98	19.812
$2^6\times2^6\times2^6$	4.594	22.781	27.375	2.994	22.225	25.219
$2^7\times2^7\times2^7$	6.047	27.141	33.188	4.231	26.503	30.734
$2^8\times2^8\times2^8$	7.523	31.57	39.093	5.506	30.798	36.304
$2^9\times2^9\times2^9$	9.012	36.035	45.047	6.8	35.102	41.902
$2^{10}\times2^{10}\times2^{10}$	10.506	40.518	51.024	8.103	39.411	47.514
$2^{11}\times2^{11}\times2^{11}$	12.003	45.009	57.012	9.411	43.721	53.132
$2^{12}\times2^{12}\times2^{12}$	13.501	49.504	63.005	10.721	48.032	58.753

As shown in Table V and Figs. 3–5, it is clear that the radix-2 × 2 × 2 Hartley transform involves fewer multiplications and additions than the row-column approach based on hybrid DHT/FFT algorithm. In addition, unlike the row-column approach, radix-2 × 2 × 2 does not involve matrix transpose. This is very promising because we are comparing a relatively new algorithm, which has the potential to be improved with the row-column approach based on well-optimized 1-D algorithms.

4) Comparison With Known Papers in 3-D DHT: In [26], the authors proposed the calculation of the 3-D Hartley transform using an index scheme to map the 3-D DHT into a series of 1-D, 2-D and 3-D convolutions and then use 1-D multiplication free FNT processors in a row-column fashion to calculate these convolutions and, hence, to calculate the 3-D DHT. The method achieves a very low multiplicative complexity and takes advantage of the low-cost implementation of the FNT processors [27]. This method is useful when FNT processors are available.

In [28], the authors proposed a quite complex algorithm for calculating the 2-D and 3-D Hartley transforms. The algorithm involves the calculation of both 1-D DHT and FFT transforms. In total, $2N^2$ N-point DFTs and N^2N-point DHT plus $2(N-2)N^2$ intermediate additions are needed for the calculation of 3-D DHT. The 1-D DHTs and 1-D FFTs are calculated using a combination of prime factors of real series and Winograd algorithms. This is quite complex for practical implementation, and it is not economical to implement it in software (it requires the implementation of both 1-D DHT

and 1-D FFT plus a relatively complex indexing scheme) and is exceedingly difficult to implement in hardware. The algorithm reduces the number of multiplications at the expense of increasing the number of additions. The total number of arithmetic operations in this algorithm is more than in the proposed radix-2 × 2 × 2 algorithm. For example, the total number of operations (additions+multiplications) is 42.19/point for a transform of size 252 × 252 × 252, whereas in our algorithm, the corresponding figure is 36.305/point for a transform size of 256 × 256 × 256. Furthermore, the proposed radix-2 × 2 × 2 algorithm has a simpler indexing scheme and better regular structure and can be implemented using even a single butterfly.

In another paper about the multidimensional Hartley transform [29], the authors proposed the calculation of multidimensional Hartley transforms using the 1-D complex Fourier transform in a row-column fashion and retrograde indexing. In general, the arithmetic complexity of this approach is $(N/2+1)$ 1-D complex Fourier transforms for each dimension of the m-D DHT. For example, if we apply this approach to the calculation of the 3-D Hartley transform, $3(N/2+1)$ complex 1-D FFTs will be needed, plus a special indexing scheme (retrograde indexing) plus a matrix transpose as shown in [29, Fig. 1]. Therefore, this approach is less efficient than the row-column approach, based on 1-D FHT, which we are using for comparison. Similar to the FNT-based approach, this method can be useful if it is required to use existing 1-D FFT hardware to calculate the m-D DHT.

However, all these papers avoided the subject of developing proper multidimensional Hartley transform algorithms. They use different index mapping schemes in order to map the 3-D problem into 1-D and then use other 1-D transforms for the calculation of the multidimensional Hartley transform in row-column fashion. This demonstrates that algorithm development of the Hartley transform for 3-D and higher dimensions is still relatively new and unexploited and requires more development.

In our paper, we proposed the 3-D radix-2 × 2 × 2 and compared it with the most commonly used row-column approach and other published work for 3-D DHT. Compared with the row-column approach based on the same algorithm, in the 1-D case, the radix-2 × 2 × 2 is found to reduce both the number of multiplications and additions significantly. Compared with the row-column approach based on 1-D split-radix and 1-D hybrid DHT/FFT, the radix-2 × 2 × 2 still involves fewer arithmetic operations, has no matrix transpose, and has better structure and indexing scheme. In addition, the proposed radix-2 × 2 × 2 is more efficient than other published work in 3-D DHT [26]–[29] and better suited for the calculation of 3-D DHT. This is very encouraging because we are comparing a relatively new algorithm that may be improved with optimized algorithms, as in the 1-D case.

V. Conclusion

The multidimensional Hartley transform is introduced as an alternative tool to the multidimensional Fourier transform for real data applications. It has been applied in many applications in digital image and multidimensional signal processing and communications. The multidimensional Hartley transform in three and more dimensions is usually calculated using the row-column approach. However, proper 3-D algorithms can be more efficient and need development. In this paper, we have introduced the concept and derivation of the 3-D radix-2 × 2 × 2 algorithm for fast calculation of the 3-D DHT. An example, which shows the validity of this algorithm, for forward and inverse 3-D Hartley transform, calculated using the 3-D radix-2 × 2 × 2, has been given. The arithmetic complexity for this algorithm has been analyzed and compared with the row-column approach. It has been found that the total number of multiplications and additions is reduced, as shown in Tables I–V and Figs. 3–5. In addition, the radix-2 × 2 × 2 is simple to implement and has a regular structure, and unlike the row-column approach, it does not require matrix transpose. This justifies our work and may lead to more development of fast algorithms in three and more dimensions.

References

[1] R. N. Bracewell, *The Hartley Transform*. Oxford, U.K.: Oxford Univ. Press, 1986.

[2] H. V. Sorensen, D. L. Jones, C. S. Burrus, and M. T. Heideman, "On computing the discrete Hartley transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, pp. 1231–1238, 1985.

[3] A. Ganapathiraju, J. Hamaker, J. Picone, and A. Skjellum. A Comparative Analysis of FFT Algorithms. Mississippi State Univ., Mississippi State. [Online]. Available: http://www.isip.msstate.edu/publications/journals/ieee_sp/1997/paralle l_dsp/paper_v7.pdf

[4] V. Britanak and K. R. Rao, "The fast-generalized discrete Fourier transforms: A unified approach to discrete sinusoidal transforms computation," *Signal Process.*, vol. 79, pp. 135–150, 1999.

[5] P. R. Uniyal, "Transforming real-valued sequences: Fast Fourier versus fast Hartley transform algorithms," *IEEE Trans. Signal Processing*, vol. 42, pp. 3249–3254, Nov. 1994.

[6] M. Popovic and D. Servic, "A new look at the comparison of the fast Hartley and Fourier transforms," *IEEE Trans. Signal Processing*, vol. 42, pp. 2178–2182, Aug. 1994.

[7] M. Balducci, A. Ganapathiraju, J. Hamaker, J. Picone, A. Choudary, and A. Skjellum, "Benchmarking of FFT algorithms," in *Proc. Eng. New Century*, 1997, pp. 328–330.

[8] G. Bi, "New split-radix algorithm for the discrete Hartley transform," *IEEE Trans. Signal Processing*, vol. 45, pp. 297–302, Feb. 1997.

[9] E. A. Jonckheere and C. Ma, "Split-radix fast Hartley transform in one and two dimensions," *IEEE Trans. Signal Processing*, vol. 39, pp. 499–503, Feb. 1991.

[10] R. Kumaresan and P. K. Gupta, "Vector-radix algorithm for a 2-D discrete Hartley transform," in *Proc. IEEE*, vol. 74, May 1986, pp. 755–757.

[11] N. C. Hu and F. F. Lu, "Fast computation of the two-dimensional generalized Hartley transforms," in *IEE Proc. Inst. Elect. Eng., Vision, Image, Signal Process.*, vol. 142, Feb. 1995, pp. 35–39.

[12] R. N. Bracewell, O. Buneman, H. Hao, and J. Villasenor, "Fast two-dimensional Hartley transform," in *Proc. IEEE*, vol. 74, Sept. 1986, pp. 1282–1283.

[13] H. Hao and R. N. Bracewell, "A three-dimensional DFT algorithm using the fast Hartley transform," in *Proc. IEEE*, vol. 75, Feb. 1987, pp. 264–266.

[14] P. Duhamel and M. Vetterili, "Improved Fourier and Hartley transform algorithms: Application to cyclic convolution of real data," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 818–824, June 1987.

[15] T. M. Wong and C. P. Kwong, "Adaptive filtering using Hartley transform and overlap-save method," *IEEE Trans. Signal Processing*, vol. 39, pp. 1708–1711, July 1991.

[16] A. Bernardi and R. N. Bracewell, "Updating the power spectrum of real data by the Hartley method," in *Proc. IEEE*, vol. 75, July 1987, pp. 964–965.

[17] J. L. Wu and J. Shiu, "Discrete Hartley transform in error control coding," *IEEE Trans. Signal Processing*, vol. 39, pp. 2356–2359, Oct. 1991.

[18] C. L. Wang, C. H. Chang, J. L. Fan, and J. M. Cioffi, "Discrete Hartley transform based multicarrier modulation," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 5, pp. 2513–2516, 2000.
[19] R. Saatcilar, S. Ergintav, and N. Canitez, "The use of Hartley transform in geophysical applications," *Geophys.*, vol. 55, pp. 1488–1495, 1990.
[20] J. I. Agbinya, "Fast interpolations algorithm using fast Hartley transform," in *Proc. IEEE*, vol. 75, Apr. 1987, pp. 523–524.
[21] G. T. Heydt and K. J. Olejniczak, "The Hartley series and its application to power quality assessment," *IEEE Trans. Industry Applications*, vol. 29, pp. 522–527, May–June 1993.
[22] S. A. Mahmoud, "Motion analysis of multiple moving objects using Hartley transform," *IEEE Trans. Syst., Man, Cybern.*, vol. 21, pp. 280–287, Jan.-Feb. 1991.
[23] C. H. Paik and M. D. Fox, "Fast Hartley transforms for image processing," *IEEE Trans. Med. Imag.*, vol. 7, pp. 149–153, June 1988.
[24] K. S. Knudsen and L. T. Bruton, "Mixed multidimensional filters," in *Proc. 33rd Midwest Symp. Circuits Syst.*, vol. 1, 1991, pp. 80–83.
[25] A. Erdi, E. Yorke, M. Loew, Y. Erdi, M. Sarfaraz, and B. Wessels, "Use of the fast Hartley transform for three-dimensional dose calculation in radionuclide therapy," *Med. Phys.*, vol. 25, pp. 2226–2233, Nov. 1998.
[26] S. Boussakta and A. G. J. Holt, "Fast multidimensional discrete Hartley transform using Fermat number transform," in *Proc. Inst. Elect. Eng. G Circuits, Devices, Syst.*, vol. 135, Dec. 1988, pp. 253–257.
[27] ——, "Calculation of the discrete Hartley transform via the Fermat number transform using a VLSI chip," in *Proc. Inst. Elect. Eng. G Circuits, Devices, Syst.*, vol. 135, June 1988, pp. 101–103.
[28] P. K. Meher, J. K. Satapathy, and G. Panda, "Fast computation of multidimensional discrete Hartley transform," *Electron. Lett.*, vol. 28, pp. 1077–1078, June 1992.
[29] T. Bortfield and W. Dinter, "Calculation of multidimensional Hartley transforms using one-dimensional Fourier transforms," *IEEE Trans. Signal Processing*, vol. 43, pp. 1306–1310, May 1995.
[30] N. C. Hu, H. I. Chang, and O. K. Ersoy, "Generalized discrete Hartley transforms," *IEEE Trans. Signal Processing*, vol. 40, pp. 2931–2940, Nov. 1992.

Said Boussakta (M'90) received the "Ingenieur d'Etat" degree in electronic engineering from the National Polytechnic Institute of Algiers (ENPA), Algiers, Algeria in 1985 and the Ph.D degree in electrical engineering (signal and image processing) from the University of Newcastle upon Tyne, Newcastle upon Tyne, U.K., in 1990.

From 1990 to 1996, he was with the University of Newcastle upon Tyne as Senior Research Associate in digital signal and image processing. From 1996 to 2000, he was with the University of Teesside, Teesside, U.K., as Senior Lecturer in communications. He is currently at the Institute of Integrated Information Systems, University of Leeds, Leeds, U.K., where he is lecturing on modern communications networks, communications systems, and signal processing. His research interests are in the areas of digital communications, coding and cryptography, digital signal/image processing, and fast DSP algorithms. He has authored and co-authored a number of publications.

Dr. Boussakta is a Member of the IEEE Signal Processing, Communications and Computer Societies and of the IEE.

Osama Hamoud Alshibami (S'98) received the B.Sc. degree in computer engineering from Amman University, Amman, Jordan, in 1996 and the M.Sc. degree in advanced manufacturing systems from University of Teesside, Teesside, U.K., in 1998. He is now pursuing the Ph.D. degree at the Institute of Integrated Information Systems, School of Electronic and Electrical Engineering, University of Leeds, Leeds, U.K.

His research interests include fast computational algorithms for digital signal processing applications in one and multi-dimensions, image processing, video compression, and communication systems.

Mohammed Yunis Aziz (S'99) received the M.Sc. degree in electronics from the University of Newcastle Upon Tyne, Newcastle Upon Tyne, U.K., in 1994.

From 1994 to 1996, he was an electronics design engineer at British Gas Research Center, Newcastle Upon Tyne. From 1996 to 1999, he was working as a control design engineer at Siemens plc. From 1999 to 2001, he was a Research Assistant and a Ph.D. student at the University of Teesside, Teesside, U.K., and Leeds University, Leeds, U.K. He is currently a Teaching Assistant and Researcher at the University of Leeds. His research interests are parallel processing algorithms, architectures and applications, embedded signal processing systems, and image processing.