3-D VECTOR RADIX ALGORITHM FOR THE 3-D DISCRETE HARTLEY TRANSFORM

O. Alshibami and S. Boussakta

Institute of Integrated Information Systems,
School of Electronic and Electrical Engineering,
The University of Leeds, Leeds LS2 9JT, UK
e-mail: s_bouss@hotmail.com

ABSTRACT

The three-dimensional discrete Hartley transform (3-D DHT) has been proposed as an alternative tool to the 3-D discrete Fourier transform (3-D DFT) for 3-D applications when the data is real. The 3-D DHT has been applied in many three-dimensional image and multidimensional signal processing applications. This paper presents a fast three-dimensional algorithm for computing the 3-D DHT. The mathematical development of this algorithm is introduced and the arithmetic complexity is analysed and compared to related algorithms. Based on a single butterfly implementation, this algorithm is found to offer substantial savings in the total number of multiplications and additions over the familiar row-column approach.

1. INTRODUCTION

The fast Hartley transform (FHT) has attracted research interests in many applications [1-4]. These applications include spectrum analysis, signal and image processing [1], motion analysis [2], multidimensional filtering [3], and, recently, calculating the dose in radionuclide therapy using the 3-D DHT [4]. The multidimensional discrete Hartley transform (m-D DHT) is proposed as an alternative tool to the complex multidimensional discrete Fourier transform (m-D DFT) [5-10]. All the properties applicable to the m-D DFT apply to the m-D DHT. The advantage of the m-D DHT, over the m-D DFT, is that it is a real to real transform and it has the same inverse. Therefore, the m-D DHT is more suitable for multidimensional signal processing applications when the data is real.

Unlike the m-D DFT which is separable, m-D DHT can not be calculated using algorithms developed for the 1-D and 2-D cases [5,6,10] directly in the row-column approach. Hao and Bracewell [8] proposed the computation of the 3-D DHT through an intermediate transform, which is separable. From the intermediate transform, the 3-D DHT is calculated at the expense of more additions and multiplications [8,9]. Although, this method has the advantage of using algorithms developed for the 1-D case, proper multidimensional algorithms can be faster, more efficient and need to be developed.

In this paper, the 3-D vector-radix algorithm is developed for fast calculation of the 3-D Hartley transform. This algorithm involves elements from throughout the 3-D arrays rather than operating on individual rows and columns. The arithmetic complexity for this algorithm is analysed and compared to the familiar row-column approach. The new algorithm has been found to be faster and more efficient.

2. THREE DIMENSIONAL DHT

The 3-D discrete Hartley transform for an N×N×N 3-D data, $x(n_1,n_2,n_3)$, is defined by:

$$X(k_1,k_2,k_3)=\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1}\sum_{n_3=0}^{N-1}x(n_1,n_2,n_3)cas\left(\frac{2\pi}{N}(n_1k_1+n_2k_2+n_3k_3)\right) \quad (1)$$

$$k_1,k_2,k_3=0,1,2,...,N-1$$

where cas(ω)=cos(ω)+sin(ω).

The inverse transform is the same as the forward except for a scale factor $1/N^3$:

$$x(n_1,n_2,n_3)=\frac{1}{N^3}\sum_{k_1=0}^{N-1}\sum_{k_2=0}^{N-1}\sum_{k_3=0}^{N-1}X(k_1,k_2,k_3)cas\left(\frac{2\pi}{N}(n_1k_1+n_2k_2+n_3k_3)\right) \quad (2)$$

$$n_1,n_2,n_3=0,1,2,...,N-1$$

The scale factor $1/N^3$ can be split between the forward and the inverse transforms to make them exactly the same.

3. 3-D VECTOR-RADIX ALGORITHM

The multidimensional Hartley transform is usually computed using the row-column approach by adding a certain number of temporary arrays. The temporary arrays are computed using several 1-D FHTs applied over each dimension successively. The m-D DHT is then computed from the temporary arrays at the expense of some extra additions and halvings (multiplication by 0.5) [8,9].

In this paper, the 3-D vector-radix algorithm is introduced for direct calculation of the 3-D DHT with size $2^n\times2^n\times2^n$. This algorithm extends the idea behind the one-dimensional radix-2 FHT algorithm to the 3-D case.

3.1 Mathematical Development

In this algorithm, the 3-D Hartley of size N×N×N-point is divided into eight N/2×N/2×N/2-point 3-D DHTs. In the next stage of the algorithm, each N/2×N/2×N/2-point 3-D DHT is further divided into eight N/4×N/4×N/4-point 3-D DHTs, and the process continues until we get 2×2×2 transforms. Hence, the 3-D discrete Hartley transform, $X(k_1,k_2,k_3)$, can be decomposed as:

$$\begin{aligned} X(k_1,k_2,k_3) &= X_{000}(k_1,k_2,k_3) + X_{001}(k_1,k_2,k_3) + X_{010}(k_1,k_2,k_3) \\ &+ X_{011}(k_1,k_2,k_3) + X_{100}(k_1,k_2,k_3) + X_{101}(k_1,k_2,k_3) \\ &+ X_{110}(k_1,k_2,k_3) + X_{111}(k_1,k_2,k_3) \end{aligned} \quad (3)$$

where

$$X_{\alpha\beta\delta}(k_1,k_2,k_3) = \sum_{n_1=0}^{N/2-1}\sum_{n_2=0}^{N/2-1}\sum_{n_3=0}^{N/2-1} x(2n_1+\alpha, 2n_2+\beta, 2n_3+\delta) \, cas\left(\frac{2\pi}{N}\left((2n_1+\alpha)k_1 + (2n_2+\beta)k_2 + (2n_3+\delta)k_3\right)\right) \quad (4)$$

$\alpha, \beta, \delta = 0 \ or \ 1$

For $\alpha\beta\delta = 000$, $X_{000}(k_1,k_2,k_3)$ can be written as:

$$\begin{aligned} X_{000}(k_1,k_2,k_3) &= \sum_{n_1=0}^{N/2-1}\sum_{n_2=0}^{N/2-1}\sum_{n_3=0}^{N/2-1} x(2n_1,2n_2,2n_3) \, cas\left(\frac{2\pi}{N}(2n_1k_1+2n_2k_2+2n_3k_3)\right) \\ &= D_{000}(k_1,k_2,k_3) \end{aligned} \quad (5)$$

where

$$D_{\alpha\beta\delta}(k_1,k_2,k_3) = \sum_{n_1=0}^{N/2-1}\sum_{n_2=0}^{N/2-1}\sum_{n_3=0}^{N/2-1} x(2n_1+\alpha, 2n_2+\beta, 2n_3+\delta) \, cas\left(\frac{2\pi}{N}(2n_1k_1+2n_2k_2+2n_3k_3)\right) \quad (6)$$

$\alpha, \beta, \delta = 0 \ or \ 1$

$D_{000}(k_1,k_2,k_3)$ is a 3-D DHT with size N/2×N/2×N/2. Similarly, $X_{001}(k_1,k_2,k_3)$ can be written as:

$$X_{001}(k_1,k_2,k_3) = \sum_{n_1=0}^{N/2-1}\sum_{n_2=0}^{N/2-1}\sum_{n_3=0}^{N/2-1} x(2n_1,2n_2,2n_3+1) \, cas\left(\frac{2\pi}{N}(2n_1k_1+2n_2k_2+(2n_3+1)k_3)\right) \quad (7)$$

Using the identity:

$$cas(m+n) = cas(m)\cos(n) + cas(-m)\sin(n) \quad (8)$$

$X_{001}(k_1,k_2,k_3)$ can be written as:

$$X_{001}(k_1,k_2,k_3) = D_{001}(k_1,k_2,k_3)\cos\left(\frac{2\pi}{N}k_3\right) + D_{001}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\left(\frac{2\pi}{N}k_3\right) \quad (9)$$

Again $D_{001}(k_1,k_2,k_3)$ and $D_{001}(N/2-k_1,N/2-k_2,N/2-k_3)$ are 3-D DHTs with size N/2×N/2×N/2. Following the same development, $X_{010}(k_1,k_2,k_3)$, $X_{011}(k_1,k_2,k_3)$, $X_{100}(k_1,k_2,k_3)$, $X_{101}(k_1,k_2,k_3)$, $X_{110}(k_1,k_2,k_3)$, and $X_{111}(k_1,k_2,k_3)$ can be developed as:

$$X_{010}(k_1,k_2,k_3) = D_{010}(k_1,k_2,k_3)\cos\left(\frac{2\pi}{N}k_2\right) + D_{010}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\left(\frac{2\pi}{N}k_2\right) \quad (10)$$

$$X_{011}(k_1,k_2,k_3) = D_{011}(k_1,k_2,k_3)\cos\left(\frac{2\pi}{N}(k_2+k_3)\right) + D_{011}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\left(\frac{2\pi}{N}(k_2+k_3)\right) \quad (11)$$

$$X_{100}(k_1,k_2,k_3) = D_{100}(k_1,k_2,k_3)\cos\left(\frac{2\pi}{N}k_1\right) + D_{100}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\left(\frac{2\pi}{N}k_1\right) \quad (12)$$

$$X_{101}(k_1,k_2,k_3) = D_{101}(k_1,k_2,k_3)\cos\left(\frac{2\pi}{N}(k_1+k_3)\right) + D_{101}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\left(\frac{2\pi}{N}(k_1+k_3)\right) \quad (13)$$

$$X_{110}(k_1,k_2,k_3) = D_{110}(k_1,k_2,k_3)\cos\left(\frac{2\pi}{N}(k_1+k_2)\right) + D_{110}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\left(\frac{2\pi}{N}(k_1+k_2)\right) \quad (14)$$

$$X_{111}(k_1,k_2,k_3) = D_{111}(k_1,k_2,k_3)\cos\left(\frac{2\pi}{N}(k_1+k_2+k_3)\right) + D_{111}\left(\frac{N}{2}-k_1,\frac{N}{2}-k_2,\frac{N}{2}-k_3\right)\sin\left(\frac{2\pi}{N}(k_1+k_2+k_3)\right) \quad (15)$$

Replacing (5), (9), and (10-15) into (3), leads to the general formula of the 3-D vector-radix algorithm for the 3-D DHT:

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

Combining eight points together gives the 3-D vector-radix algorithm butterfly as shown in (17) and Figure 1:

$$\begin{bmatrix} X(k_1,k_2,k_3) \\ X(k_1,k_2,k_3+N/2) \\ X(k_1,k_2+N/2,k_3) \\ X(k_1,k_2+N/2,k_3+N/2) \\ X(k_1+N/2,k_2,k_3) \\ X(k_1+N/2,k_2,k_3+N/2) \\ X(k_1+N/2,k_2+N/2,k_3) \\ X(k_1+N/2,k_2+N/2,k_3+N/2) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} X_{000}(k_1,k_2,k_3) \\ X_{001}(k_1,k_2,k_3) \\ X_{010}(k_1,k_2,k_3) \\ X_{011}(k_1,k_2,k_3) \\ X_{100}(k_1,k_2,k_3) \\ X_{101}(k_1,k_2,k_3) \\ X_{110}(k_1,k_2,k_3) \\ X_{111}(k_1,k_2,k_3) \end{bmatrix} \quad (17)$$

3.2. Arithmetic Complexity

Figure 1 shows a single butterfly for the 3-D vector radix algorithm for the 3-D DHT. It calculates eight points and involves 31 real additions and 14 real multiplications. This butterfly needs to be calculated (N×N×N)/8 times each stage. The 3-D transform needs $\log_2 N$ stages. Therefore, the calculation of the whole transform using a single butterfly requires:

$$\text{Additions} = (31/8)\, N^3 \log_2 N \quad (18)$$

and

$$\text{Multiplications} = (14/8)\, N^3 \log_2 N \quad (19)$$

Using several butterflies to remove trivial operations, the total number of multiplications and additions can be reduced quite significantly at the expense of increased complexity. For example, the first stage in the 3-D vector radix algorithm decimation in time can be calculated without multiplications and with only 24 real additions per butterfly as shown in Figure 2.

In the row-column approach, the calculation of the 3-D DHT starts with the calculation of a 2-D separable transform using algorithms developed for the 1-D DHT applied over each dimension. The 3-D DHT is then calculated from the separable transform at the expense of $3N^3$ extra additions, and more shifts (multiplications by 1/2). Ignoring the multiplications by (1/2) as they can be embedded with other calculations, the total numbers of arithmetic operations required to calculate an $N \times N \times N$ 3-D DHT using the row-column approach are:

$$\text{Additions} = (9/2)N^3\log_2N+3N^3 \quad (20)$$

and

$$\text{Multiplications} = 3N^3\log_2N \quad (21)$$

As shown in Figures 3 and 4, the new algorithm offers a substantial saving in both multiplications and additions.

4. CONCLUSION

As an alternative to computing the 3-D DHT transform using 1-D algorithms in row-column fashion, a proper three-dimensional algorithm has been introduced in this paper. This algorithm extends the idea behind the radix-2 in one-dimensional to the 3-D case reducing the number of both multiplications and additions. The arithmetic complexity for this algorithm has been analysed and compared to related algorithms. It has been found that, the new algorithm offers substantial savings over the familiar row-column approach. It should be noted that the comparison is based on a single butterfly implementation, using multiple butterflies to remove trivial operations will reduce arithmetic operations in both algorithms at the expense of increased complexity.

5. ACKNOWLEDGEMENT

The authors are pleased to acknowledge the financial support from the Nuffield Foundation.

References

1. C. H. Paik and M. D. Fox, “Fast Hartley transforms for image processing”, IEEE Trans. on Medical Imaging. vol. 7, pp. 149-153, June 1988.
2. S. A. Mahmoud, “Motion analysis of multiple moving objects using Hartley transform”, IEEE Trans. on Systems, Man Cybern. vol. 21, pp. 280-287, 1991.
3. K. S. Knudsen and L. T. Bruton, “Mixed multidimensional filters”, Proc. 33rd Midwest Symp. on Circuits and Systems, 1991, pp. 80-83.
4. 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”, Medical Physics. vol. 25, (11), pp. 2226-2233, 1998.
5. R.N. Bracewell, The discrete Hartley transform. Oxford University Press, 1986.
6. H. V. Sorensen, D. L. Jones, C. S. Burrus and M. T. Heidman, “On computing the discrete Hartley transform”, IEEE Trans. on Acoustics, Speech, and Signal Processing. vol. 33, (4), pp. 1231-1238, 1985.
7. R. N. Bracewell, O. Buneman, H. Hao and J. Villasenor, “Fast two-dimensional Hartley transform”, Proc. of the IEEE. vol. 74, (9), pp. 1282-83, September 1986.
8. H. Ho and R. N. Bracewell, “A three-dimensional DFT algorithm using the fast Hartley transform”, Proc. of the IEEE. vol. 75, (2), pp. 264-266, February 1987.
9. O. Buneman, “Multidimensional Hartley transforms”, Proc. of the IEEE. vol. 75, (2), pp. 267, February 1987.
10. R. Kumaresan and P. K. Gupta, “Vector-radix algorithm for a 2-D discrete Hartley transform”, Proc. of the IEEE. vol. 74, (5), pp. 755-757, May 1986.

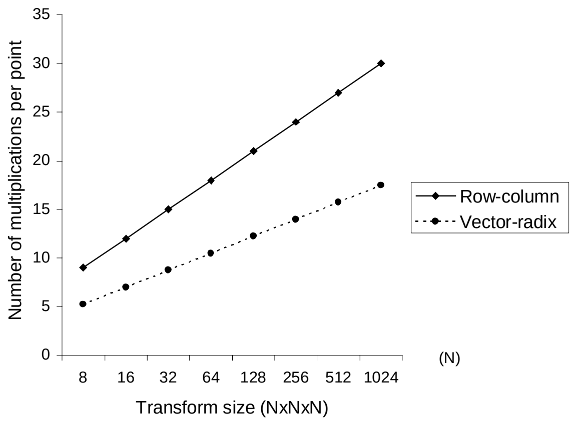

Figure 3. Comparison between the row-column and the 3-D vector-radix algorithms (number of multiplications per point).

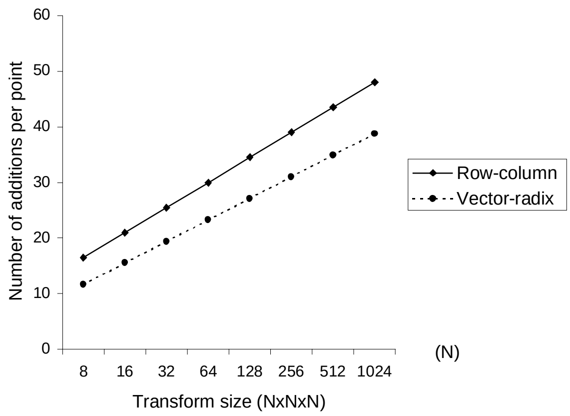

Figure 4. Comparison between the row-column and the 3-D vector-radix algorithms (number of additions per point).

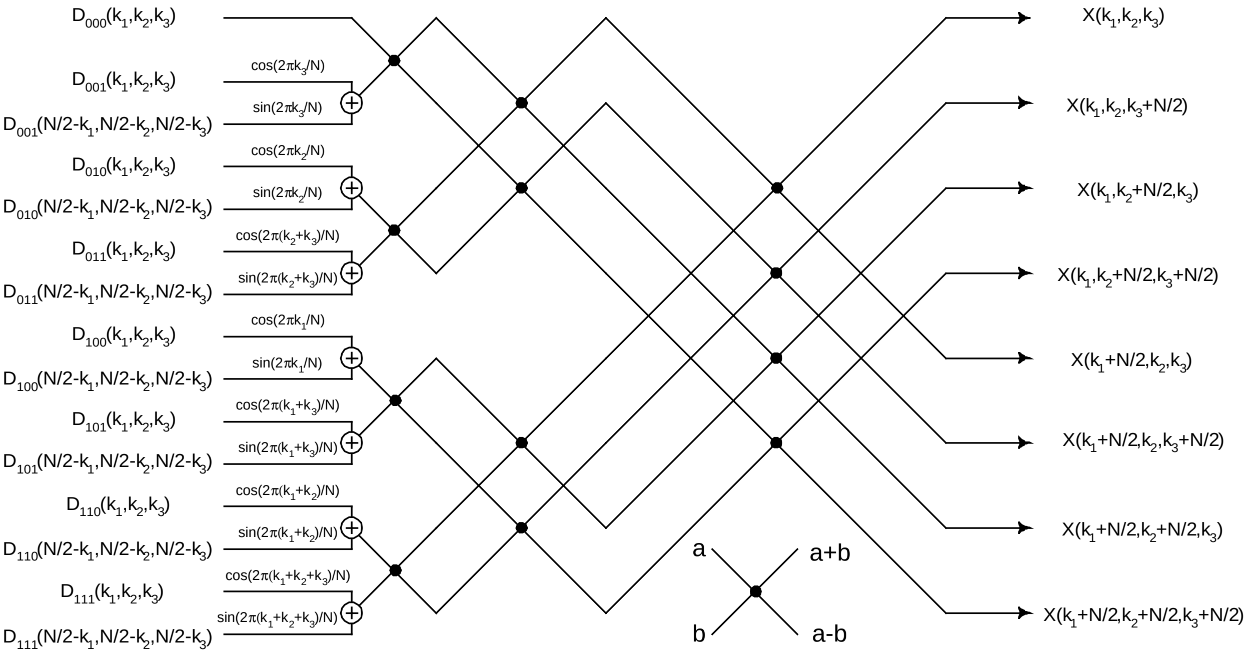

Figure 1. One 3-D butterfly in the 3-D vector-radix algorithm for 3-D DHT.

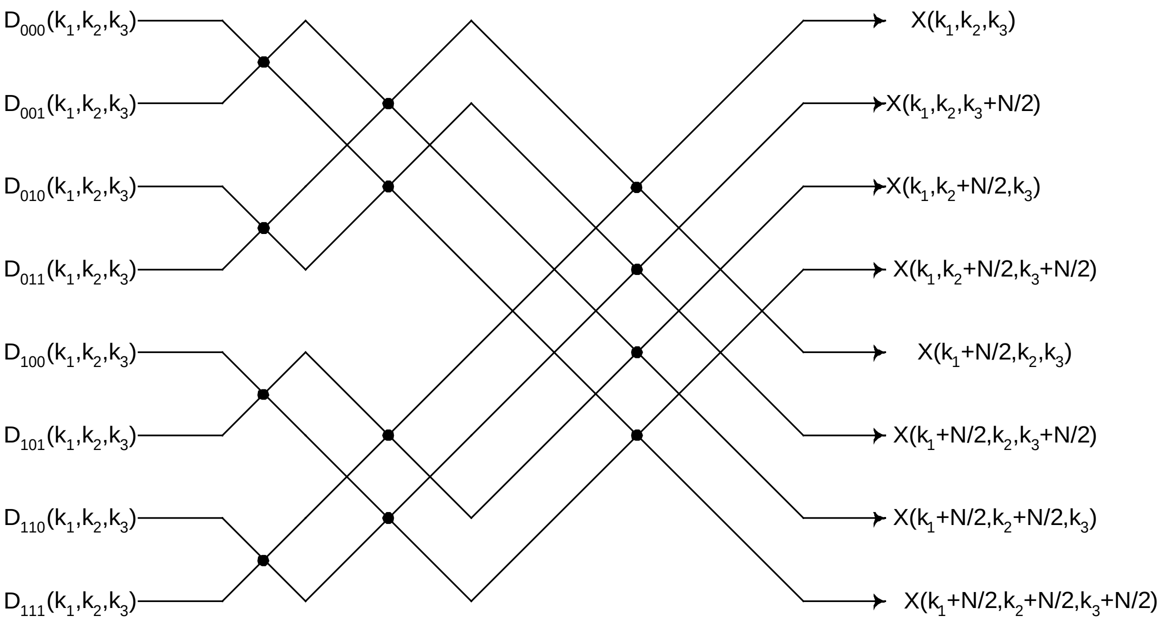

Figure 2. One 3-D butterfly with no multiplications in the 3-D vector-radix algorithm for 3-D DHT.