Improved FHT Algorithms for Fast Computation of the Discrete Hartley Transform

Mounir Taha Hamood, Lecturer
Electrical Engineering Department, University of Tikrit

Abstract
In this paper, by using the symmetrical properties of the discrete Hartley transform (DHT), an improved radix-2 fast Hartley transform (FHT) algorithm with arithmetic complexity comparable to that of the real-valued fast Fourier transform (RFFT) is developed. It has a simple and regular butterfly structure and possesses the in-place computation property. Furthermore, using the same principles, the development can be extended to more efficient radix-based FHT algorithms. An example for the improved radix-4 FHT algorithm is given to show the validity of the presented method. The arithmetic complexity for the new algorithms are computed and then compared with the existing FHT algorithms. The results of these comparisons have shown that the developed algorithms reduce the number of multiplications and additions considerably.

Keywords: Signal/Image processing, discrete Hartley transform (DHT), real-valued fast Fourier transform, (RFFT), fast transform algorithms.

تطوير خوارزميات جديدة للحساب السريع لتحويل هارتلي المنفصل

الخلاصة

تم في هذا البحث، الاستفادة من الخصائص التناظرية لتحويل هارتلي المنفصل (DHT) للحصول على خوارزمية سريعة جديدة من الجذر الثاني والتي تحتاج الى عمليات رياضية تساوي تلك العمليات المطلوبة لتحويل فورير ذو القيمة الحقيقة (RFFT). تتميز الخوارزمية المقترحة بسهولة التركيب ولها هيكل منتظم ويمكن تنفيذها باستخدام الحد الأدنى من الذاكرة.

تم تعميم الفكرة في الجزء الثاني من البحث لتشمل جميع الخوارزميات ذوات الجذور الأعلى، وقد قدمت خوارزمية الجذر الرابع كمثال عملي لتقييم كفاءة الطريقة المطورة. تم حساب العمليات الرياضية المطلوبة لتنفيذ الخوارزميات الجديدة وبعد ذلك تمت مقارنتها مع نظيراتها المعروفة وقد أظهرت نتيجة المقارنة بأن الخوارزميات المقدمة في هذا البحث تقلص العمليات الرياضية بشكل ملحوظ.

الكلمات الدالة:معالجة الأشارة والصور، تحويل هارتلي، تحويل فورير ذو القيمة الحقيقة، خوارزميات التحويل السريعة.

Introduction

The discrete Hartley transform (DHT) first introduced by Bracewell [1,2] plays an important role in signal and image processing applications. It is closely related to the real-valued discrete Fourier transform (RDFT)[3,4] with the comparable arithmetic complexity. However the advantage of DHT over the RDFT is that the DHT is an involuntary transform, therefore the forward transform is differing from the inverse transform only by the scale factor. Furthermore, for some applications such as convolution and correlation that required both the forward and the inverse transforms. The inverse RDFT still needs to deal with the complex values even for real-input sequence, while this not the case for the DHT. Therefore, the DHT has been proved to be faster than RDFT for these applications.

A lot of work has been done for the development of the fast Hartley transform (FHT) algorithms to reduce the arithmetic complexity and implementation cost. The first FHT algorithm was developed by Bracewell [5] that performs the DHT in a complexity proportional to $N\log_2 N$ using radix-2 approach. Sorenson *et al* [6] are developing a complete set of decimation-in-time (DIT) and in-frequency (DIF) algorithms using the index mapping approach, and verified that all (FFT) algorithms can also be applied to the computation of the FHT. Hou reported a decomposition method of the DHT computation[7]. Chan described a different algorithm by using the symmetric cosine structure (SCS)[8]. Other existing methods using fast algorithm of the discrete Fourier transform (DFT) [9] or the discrete cosine transform (DCT)[10]. All these algorithms are requiring the same number of multiplications although the twiddle factor operations can be implemented by using four multiplications and two additions or three multiplications and three additions.

In this paper, improved radix-based FHT algorithms are developed based on the symmetrical properties of the DHT matrix, with better arithmetic complexity than the existing algorithms. Next section will review the radix-2 FHT DIT algorithm and then an improved radix-2 algorithm will be proposed. Subsequently, the development is extended to the radix-4 improved algorithm, and follows that the development can be applied to any higher-radix FHT algorithms. Afterward, the performances of the developed algorithms are examined by analysing their arithmetic complexities and comparing them with the existing once. Finally a conclusion is given in the last section.

Proposed Algorithm

The DHT $X(k)$ for a real data sequence $x(n)$ of length N is defined for $0 \le k \le N-1$, as [1]:

$$X(k) = \sum_{n=0}^{N-1} x(n)\, cas(\theta nk) \qquad (1)$$

Where:

$cas(\theta) = cos(\theta) + sin(\theta), \quad \theta = 2\pi/N$

and the transform length N assumed to be an integer power of two. Fast algorithms may be developed for the DHT , they include radix-2 [5, 11], radix-4 [6,12] and split-radix [13,14] algorithms, where the decimation can be carried out either in time or in frequency.

Improved radix-2 FHT algorithm

According to the decimation in time radix-2 algorithm [5], Eq. (1) can be written as:

$$X(k) = X_{2n}(k) + X_{2n+1}(k)\cos(\theta k) + X_{2n+1}(N-k)\sin(\theta k) \qquad (2)$$

$$X(k+\tfrac{N}{2}) = X_{2n}(k) - X_{2n+1}(k)\cos(\theta k) - X_{2n+1}(N-k)\sin(\theta k) \qquad (3)$$

Where

$$X_{2n+l}(k) = \sum_{n=0}^{N/2-1} x(2n+l)\, cas(2\theta nk) \qquad (4)$$

For $l = 0,1$. $X_{2n}(k)$ and $X_{2n+1}(k)$ can be recognized as the (N/2)-point DHT for even $x(2n)$ and odd $x(2n+1)$ parts of $x(n)$. In order to ensure the in-place computation, the retrograde indexing scheme [15] must be applied for both kth and (N/2-k)th terms, as follows:

$$X(\tfrac{N}{2}-k) = X_{2n}(\tfrac{N}{2}-k) + X_{2n+1}(k+\tfrac{N}{2})\sin(\theta k) - X_{2n+1}(N-k)\cos(\theta k) \qquad (5)$$

$$X(N-k) = X_{2n}(\tfrac{N}{2}-k) - X_{2n+1}(k+\tfrac{N}{2})\sin(\theta k) + X_{2n+1}(N-k)\cos(\theta k) \qquad (6)$$

Therefore, four points are included in each butterfly to aviod overwriting an element that will be needed later [6]. Figure 1 shows the butterfly structure for this algorithm.

Equations (2) - (6) can be recognized as two separate DHTs, each of length (N/2)-point. Each of these can be further decomposed into another two (N/4)-point transforms which, in turn, can be decomposed further until two-point DHTs are obtained. Figure 2 illustrates, using signal flow graph notation, the processes involved in evaluating an eight-point (N=8) DHT using two four-point $H_1(k)$ and $H_2(k)$ transforms.

The first two stages of the transform do not involve multiplication operations by the twiddle factors, and can be fused leading to a four-point butterfly which can be implemented separately by the following matrices:

$$\begin{bmatrix} H_1(0) \\ H_1(1) \\ H_1(2) \\ H_1(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} x(0) \\ x(4) \\ x(2) \\ x(6) \end{bmatrix} \qquad (7)$$

$$\begin{bmatrix} H_2(0) \\ H_2(1) \\ H_2(2) \\ H_2(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} x(1) \\ x(5) \\ x(3) \\ x(7) \end{bmatrix} \qquad (8)$$

The next stage can be calculated by introducing a new (N/2)-point DHT, $\breve{H}_2(k)$ that processing $H_2(k)$ through the twiddle factor stage (sine and cosine values) as shown in Figure 2 and is given by:

$$\breve{H}_2(k) = H_2(k)\cos(\theta k) + H_2(N-k)\sin(\theta k) \qquad (9)$$

Therefore the output of $\breve{H}_2(k)$ can be computed by substituting (8) into (9) and noting that θ is equals to (π/4), we get:

$$\begin{bmatrix} \breve{H}_2(0) \\ \breve{H}_2(1) \\ \breve{H}_2(2) \\ \breve{H}_2(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 1 & 1 & -1 & -1 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix} \begin{bmatrix} x(1) \\ x(5) \\ x(3) \\ x(7) \end{bmatrix} \qquad (10)$$

Finally the desired DHT output is obtained by comibing $H_1(k)$ and $\breve{H}_2(k)$ transforms for $0 \le k \le 3$, in the last stage that involvs additions/subtraction only,

$$\begin{aligned} X(k) &= H_1(k) + \breve{H}_2(k) \\ X(k+\tfrac{N}{2}) &= H_1(k) - \breve{H}_2(k) \end{aligned} \qquad (11)$$

From (7) - (11), we obtain the signal flow graph of the new algorithm as shown in Figure 3.

It is abvious from Figure (3) that the developed algorithm requires two multiplications and 22 additions while for the standard radix-2 DHT algorithms [6, 11] the operation counts for N=8 are four multiplications and 26 additions. The reductions in the operations are due to the fact that at (θ=π/4) the twiddle factors are identical i.e. $cos(\theta)$=$sin(\theta)$. This redundancy is clearly shown on zeros in matrix (10) and it ocuurs at every stage staisfying the condition:

$$\theta = \left(\frac{\pi}{4}\right) i \qquad \text{for odd } i \qquad \ldots\ldots \quad (9)$$

Improved Higher-radix FHT algorithm

A view of this improvement is illustrated by the structure shown in Figure 4, which represents a partial signal flow graph extracted from the whole DHT graph at a specific length satisfying the condition given by (9). It can be proved that both of Figure 4a and Figure 4b are equivalents, as follows:

From Figure 4a, we have:

$$\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \ldots.(13)$$

and from Figure 4b,

$$\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \ldots\ldots\ldots\ldots.(14)$$

Hence (13) is identical to (14), which means that Figures 4a and 4b are also identical.

Using these results, the improvement of the FHT for any higher radix algorithms can be extended straight forward. As an example for this development is shown in Figure 5 which represents the 16-point signal flow graph of the improved radix-4 FHT algorithm.

Arithmetic Complexities

As shown from Figure (3), the improved radix-2 algorithm needs $log_2 N$ stages of butterfly computation. Each stage uses $(N-6)$ real multiplications and $(3N/2-6)$ real additions. In addition, two $(N/2)$ length DHTs have to be calculated, thus the whole improved radix-2 DHT complexity satisfies the recurrences:

$$M(N) = 2M(\tfrac{N}{2}) + N - 6 \qquad \ldots\ldots(12)$$

$$A(N) = 2A(\tfrac{N}{2}) + \tfrac{3N}{2} - 6 \qquad \ldots\ldots(13)$$

Where $M(N)$and $A(N)$ are the number of real multiplications and additions, respectively, needed by the proposed radix-2 algorithm for a length-N DHT, The computational complexities in (15) and (16) are recursive. To obtain the equations in a closed form, the initial values of these complexities are needed. In this case, the initial values can be the number of operations that are needed by length-8 DHTs, which are equal to $M(8) = 2$ and $A(8) = 22$. Solving (15) and (16) by repeated substitutions of the initial values, we get:

(10)

$$M(N) = N\log_2 N - \tfrac{7N}{2} + 6 \qquad (14)$$

$$A(N) = \tfrac{3N}{2}\log_2 N - \tfrac{5N}{2} + 6 \qquad (15)$$

(11) Based on (17) and (18), the number of operations of radix-2 DHT algorithms for different transform lengths N is shown in Table 1, in comparsions with the existing FHT algorithms. It should be noted that, using the proposed algorithm the saving in the arithmetic complexity are $(N-4)/2$ multiplicationsand $(N-4)$ additions.

In the same way, we can prove from Figure 5 that the arithmetic complexity of the improved radix-4 DHT algorithm satisfies the recurrences:

$$M(N) = 4M(\tfrac{N}{4}) + \tfrac{3N}{2} - 12 \qquad (16)$$

$$A(N) = 4A(\tfrac{N}{4}) + \tfrac{11N}{4} - 10 \qquad (17)$$

Solving (19) and (20) by repeated substitutions of the initial values $M(4) = 0$ and $A(4) = 8$, we get the closed form complexity of this algorithm, as follows:

$$M(N) = \tfrac{3N}{2}\log_4 N - \tfrac{15N}{6} + 4 \qquad (18)$$

$$A(N) = \tfrac{11N}{4}\log_4 N - \tfrac{19N}{12} + \tfrac{10}{3} \qquad (19)$$

Similarly, the arithmetic complexy given by (21) and (22) are compared with those of the radix-4 FHT algorithm [6, 12] for different transform lengths N as shown in Table (2). Clearly as shown from this table, using the proposed improved radix-4 FHT algorithm the saving in the arithmetic complexity are $(N-4)/6$ multiplications and $(N-4)/3$ additions respectively.

Conclusions

In this paper, an efficient radix-2 fast Hartley transform algorithm based on symmetrical properties of the DHT matrix has been developed. The algorithm has been extended to all power of two higher radix FHT algorithms. The advantages of the proposed algorithms are aimed that the operation counts are reduced by reducing the number of multiplications and additions without increasing the hardware complexity.

It should be noted that the developed algorithms offer comparable computational complexity with the corresponding real-FFT algorithms. Therefore, efficient hardware realization of these algorithms should be superior to current FHT/RFTT processors.

References

1. R. N. Bracewell, "Discrete Hartley Transform," *Journal of the Optical Society of America* vol. 73, pp. 1832-1835, 1983.
2. R. N. Bracewell, "The Hartley transform," Oxford University Press, Inc., 1986.
3. O. K. Ersoy and N. C. Hu, "Fast algorithms for the real discrete Fourier transform," in *International Conference on Acoustics, Speech, and Signal Processing, ICASSP-88.*, 1988, pp. 1902-1905 vol.3.
4. H. Sorensen, D. Jones, M. Heideman, and C. Burrus, "Real-valued fast Fourier transform algorithms," *IEEE Transactions on Acoustics, Speech and Signal Processing,* vol. 35, pp. 849-863, 1987.
5. R. N. Bracewell, "The fast Hartley transform," *Proceedings of the IEEE,* vol. 72, pp. 1010-1018, 1984.
6. H. Sorensen, D. Jones, C. Burrus, and M. Heideman, "On computing the discrete Hartley transform," *IEEE Transactions on Acoustics, Speech and Signal Processing.,* vol. 33, pp. 1231-1238, 1985.
7. H. S. Hou, "The Fast Hartley Transform Algorithm," *IEEE Transactions on Computers,* vol. C-36, pp. 147-156, 1987.
8. Y. H. Chan and W. C. Siu, "New fast discrete Hartley transform algorithm," *Electronics Letters,* vol. 27, pp. 347-349, 1991.
9. P. Duhamel and M. Vetterli, "Improved Fourier and Hartley transform algorithms: Application to cyclic convolution of real data," *IEEE Transactions on Acoustics, Speech and Signal Processing,* vol. 35, pp. 818-824, 1987.
10. H. Malvar, "Fast computation of discrete cosine transform through fast Hartley transform," *Electronics Letters,* vol. 22, pp. 352-353, 1986.
11. C. Kwong and K. Shiu, "Structured fast Hartley transform algorithms," *IEEE Transactions on Acoustics, Speech and Signal Processing.,* vol. 34, pp. 1000-1002, 1986.
12. K. J. Jones, "Design and parallel computation of regularised fast Hartley transform," *IEE Proceedings -Vision, Image and Signal Processing.,* vol. 153, pp. 70-78, 2006.
13. G. Bi, "Split radix algorithm for the discrete Hartley transform," *Electronics Letters,* vol. 30, pp. 1833-1835, 1994.
14. P. Soo-Chang and W. Ja-Ling, "Split-radix fast Hartley transform," *Electronics Letters,* vol. 22, pp. 26-27, 1986.

15. A. C. Erickson and B. S. Fagin, "Calculating the FHT in hardware," *IEEE Transactions on Signal Processing,* vol. 40, pp. 1341-1353, 1992.

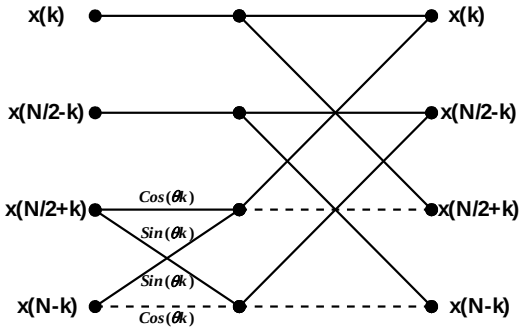

Figure. (1) An in-plce butterfly structrure of the radix-2 DHT, where $\theta = 2\pi/N$, soild and dotted lines stand for additions and subtraction respectively.

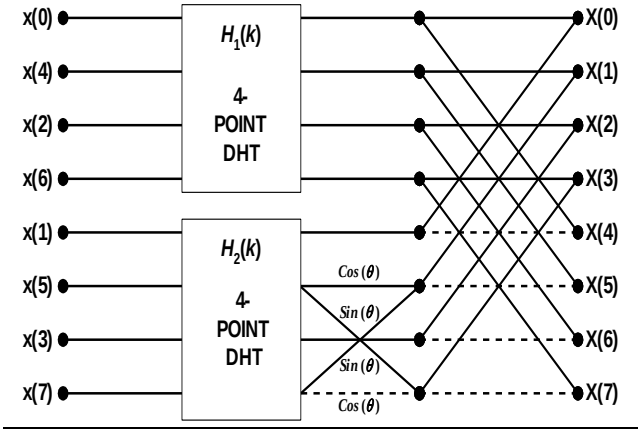

Figure. (2) Construction of an eight-point DHT from two four-point DHTs, where soild and dotted lines stand for additions and subtraction respectively.

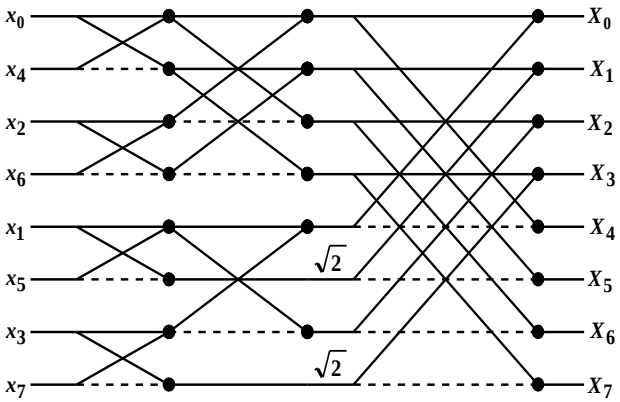

Figure.(3). Signal flow graph of 8-point improved radix-2 FHT algorithmsolid and dot lines stand for addition and subtraction respectively

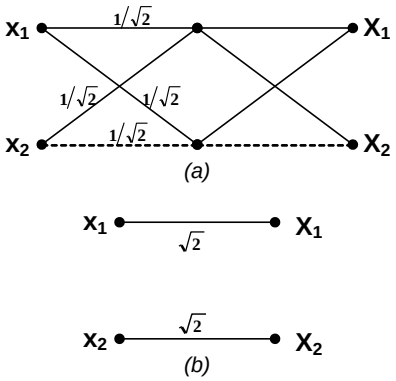

Figure.(4) Partial signal flow graph for the (a) DHT algorithm and (b) Improved DHT algorithm.

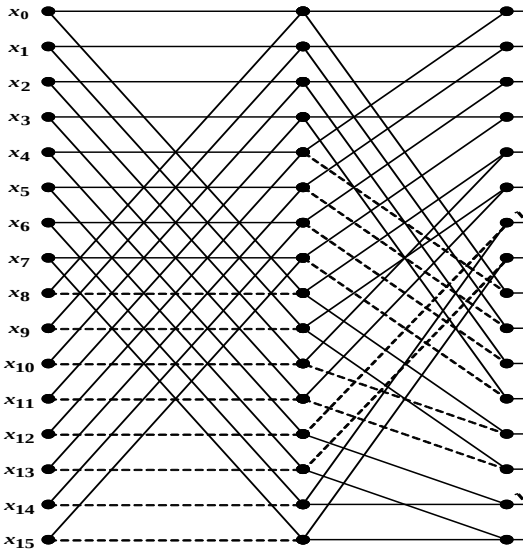

Figure.(5) Signal flow graph of 16-point improved radix-4 FHT algorithmsolid and dot lines stand for addition and subtraction respectively and $C_L^n = Cos(\frac{\pi n}{L}), S_L^n = Sin(\frac{\pi n}{L})$ **.**

Table (1)
comparison of radix-2 FHTs arithmetical complexities

Length	Radix-2FHT algorithms[5, 11]			Proposed Radix-2 FHT algorithm		
N	M(*N*)	A(*N*)	Total	M(*N*)	A(*N*)	Total
8	4	26	30	2	22	24
16	20	74	94	14	62	76
32	68	194	262	54	166	220
64	196	482	678	166	422	588
128	516	1154	1670	454	1030	1484
256	1284	2690	3974	1158	2438	3596
512	3076	6146	9222	2822	5638	8460
1024	7172	13826	20998	6662	12806	19468
2048	16388	30722	47110	15366	28678	44044
4096	36868	67586	104454	34822	63494	98316

Table (2)
comparison of radix-4 FHTs arithmetical complexities

Length	Radix-4 FHT algorithm [6]			Proposed Radix-4 FHT algorithm		
N	M(*N*)	A(*N*)	Total	M(*N*)	A(*N*)	Total
4	0	8	8	0	8	8
16	14	70	84	12	66	78
64	142	450	594	132	430	562
256	942	2498	3440	900	2414	3314
1024	5294	12802	18096	5124	12462	17586
4096	27310	62466	89776	26628	61102	87730