Arbitrary-length Fast Hartley Transform without Multiplications

Zhenbing Liu, Shujie Jiang and Huihua Yang
School of Electronic Engineering and Automatation, Guilin University of Electronic Technology, Guilin, China
E-mail: zbliu@guet.edu.cn, 462832784@qq.com, yanghuihua@tsinghua.edu.cn

Jianguo Liu
State Key Laboratory for Multispectral Information Processing Technologies, Institute of Pattern Recognition and Artificial Intelligence, Huazhong University of Science and Technology, E-mail:jgliu@ieee.org

Abstract — Discrete Hartley transform (DHT) is an important tool in digital signal processing. In this paper, a multiplierless algorithm to compute discrete Hartley transforms is proposed, which can deal with arbitrary length input signals. The proposed algorithm can be implemented by integer additions of fixed points in binary system. Besides, an efficient and regular systolic array is designed to implement the proposed method, followed by the complexity analysis. Being different to other fast Hartley transforms, our algorithm can deal with arbitrary length signals and get high precision. The proposed method is easily implemented by hardware and very suited to a real-time processing.

Keywords — Discrete Hartley transform, Moments, Multiplierless

I. INTRODUCTION

DHTs (discrete Hartley transform) are very important tools for digital signal processing and analysis and many other engineering applications. A variety of methods for computing DHT have been proposed [3,8,9,12,14]. These methods can roughly be classified into two families. One family tries to reduce the number of multiplier and add operations, while another focuses on realizing DFT with efficient systolic or other parallel hardware architectures. Meantime, since it is highly computation-intensive, there is a strong need of dedicated processors for high-speed computation of the transform coefficients to meet the requirements of real-time signal processing and digital multimedia communication systems. However, the embedded and portal applications continue to impose serious limitations on the amount of hardware involved and the rate of energy consumption. To meet the challenges of ever-growing computational demand with minimal of hardware and power, several attempts have been made for fast implementation of DHT in bit-level as well as word-level VLSI structures.

Systolic architectures are one of the most popular VLSI structures for complete intensive digital signal processing applications due to their good properties of simplicity, modularity and regularity. For DHT, several CORDIC systolic architectures have been reported in [5,7,13]. Due to the good performance, the field-programmable gate array (FPGA) for implementation of DHT becomes more and more attractive [1,2]. Besides, the ROM-based structures presented in [9] by Guo, Liu and Jen are proved to be efficient.

But to achieve the highest efficiency, all the above methods require that the signal length \( N \) is highly composite, and the error of DHT is not considered. In this paper, based on our work in [4,15,16], we proposed a multiplierless algorithm for arbitrary length DHT, which transforms the multiplications of our moments-based DHT into additions by shifting digits and accumulation of integers. Based on the approach to the fast calculation of moments [4], systolic arrays to perform 1-D DHT are presented, followed by a complexity analysis.

The rest of the paper is organized as follows. We introduce our previous work about DHT briefly in section 2. We present an improved algorithm for DHT in section 3. Followed by complexity analysis, the systolic arrays are designed to compute DHT in section 4. Finally, we include our paper in section 5.

II. USING MOMENTS TO COMPUTE DHT

By using a modular mapping, DHT can be approximated by the sum of a finite sequence of discrete moments[4].

The DHT is defined for a real-valued sequence \( x(0), x(1), \ldots, x(N-1) \), as follows:

\[
X(k) = \sum_{n=0}^{N-1} x(n) \text{cas}(2\pi nk / N)
\]

(1)

\[
= \sum_{n=0}^{N-1} x(n)(\cos(2\pi nk / N) + \sin(2\pi nk / N))
\]

Let

\[
S(k,i) = \{ n | n = i \text{mod} N, n \in \{0,1,\ldots,N-1\} \}
\]

and
By using the periodic properties of the sine and cosine functions[15], Eq.(1) can be rewritten as:

\[
X(k) = \sum_{j=0}^{N-1} x_{j,k} (\cos(2\pi i / N) + \sin(2\pi i / N)) \quad (2)
\]

By Taylor extended law,

\[
\cos(2\pi i / N) = 1 - (2\pi i / N)^2 + \cdots + (-1)^r (2\pi i / N)^{2r} / (2r)! + R_{\cos}
\]

\[
\sin(2\pi i / N) = (2\pi i / N) - (2\pi i / N)^3 / 3! + \cdots + (-1)^r (2\pi i / N)^{2r+1} / (2r+1)! + R_{\sin}
\]

where \( R_{\cos} \) and \( R_{\sin} \) are Taylor remainder.

Substituting them into Eq.(3), we get

\[
X(k) = x_{k,0} + \sum_{j=0}^{N-1} x_{j,k} \left[ (-1)^r (2\pi i / N)^{2r} / (2r)! + \cdots + (-1)^r (2\pi i / N)^{2r+1} / (2r+1)! \right] + R_{k,p}
\]

\[
= x_{k,0} + \sum_{i=0}^{2^{p+1}-1} a_i m_{k,i} + R_{k,p} \quad (3)
\]

where

\[
a_i = \begin{cases} (-1)^r (2\pi i)^{2r} / (2r)! & r = 2q \\ (-1)^r (2\pi i)^{2r+1} / (2r+1)! & r = 2q + 1 \\ \end{cases} N^r(2q)! / N^r(r)! \in N \bigcup \{0\} \subset N
\]

and \( m_{k,i} = \sum_{j=0}^{N-1} x_{j,k} i^r \) is the so-called moment.

If the item \( R_{k,p} \) is ignored, we have

\[
X(k) = x_{k,0} + \sum_{i=0}^{2^{p+1}-1} a_i m_{k,i}, \quad 0 \leq k \leq N-1 \quad (4)
\]

In effect, \( R_{k,p} \) converges to zero very rapidly and uniformly as \( p \) increases, so the approximation (4) can satisfy the accuracy requirement of most applications without computing too many terms. We have proved \( p \) can be expressed approximately in the form of \( \lfloor 2 \log_2 N / \log_2 \log_2 N \rfloor \) [4]. In this paper, \( \lfloor x \rfloor \) denotes the integer closest to \( x \).

Thus, the computation of \( X(k) \) using the approximation of (4) establishes the relationship between DHT and moments.

III. MULTIPLIERLESS DHT

In Eq. (2), there is a dot product of the moments with a constant vector \( (a_r) \) to compute. When \( N \) is large, \( (a_r) \) is too small to compute. We can resolve this problem and transform the product of floating-point into additions of integers by the following steps.

When \( N = 2^i \), multiplying \((a_r)\) by \( b_r = 2^{3\log_2 N - 10} \), we get \( [a_r \times b_r] < (-1)^r 86 \times 2^{3\log_2 N - 10} \) which are integers and can be represented as sums of distinct powers of 2:

\[
[a_r \times b_r] = \sum_{i=0}^{2^{p+1}-1} n_{i,r} 2^{i} \quad (5)
\]

where \( n_{i,r} = 0 \) or 1 for \( m = 1, \ldots, 2p+1 \), and \( i = 1, \ldots, t \).

Then, Eq. (4) can be rewritten as:

\[
x_{k,0} + \sum_{i=0}^{2^{p+1}-1} a_i m_{k,i} = x_{k,0} + m_{k,0} + \sum_{i=0}^{2^{p+1}-1} (a_i \times b_r) / 2^{3\log_2 N + 10} = x_{k,0} + m_{k,0} + \sum_{i=0}^{2^{p+1}-1} [a_i \times b_r] / 2^{3\log_2 N + 10} \quad (6)
\]

By doing so, all the products of floating-point constants with moments in Eq. (2) are eliminated, replaced by shifting the digits in binary system and accumulations of integers. Eq.(5) can be computed in advance once the signal length \( N \) is determined.

Since \( n_{r,i} = 0 \) or 1 and \( m_{r}(x) \) which can be performed by shifting digits of \( m_{r}(r) \) to the left \( \log N - 10 \) places in binary system are integers, the
computation of $\sum_{n=0}^{2^p-1} (-1)^n m_n^k(r)$ only involves accumulations of $m_n^k(r), \quad i = 0, 1, \ldots, p$. By shifting digits of $\sum_{n=0}^{2^p-1} (-1)^n m_n^k(r)$ to the left $t-i$ places in binary system, the multiplication of $2^{t-i}$ can be performed, and shifting $\sum_{n=0}^{2^p-1} (-1)^n m_n^k(r) \times 2^{-i}$ to the right $3 \log N + 20$ places in binary system, $\sum_{n=0}^{2^p-1} (-1)^n m_n^k(r) \times 2^{-i} / 2^{3 \log N + 20}$ can be performed.

It is obvious that less than $(2p+1)(t+1)+2$ additions of integers and $t+3+2p$ shifts are required to implement $X(k)$ with (6) all moments are produced.

When $N \neq 2^k$, $\log N$ is not an integer, but substituting $2^{3 \log N + 20}$ by $2^{3 \log N + 19}$, $2^{3 \log N - 10}$ by $2^{3 \log N - 11}$, and $2^k$ by $2^{3 \log N - 10}$ for $r = 1, \ldots, 2p+1$ in (5) and other above equations, we can get the same result, and the only difference is $t = \lfloor 2 \log N + 18 \rfloor$ here. For convenience, we let $t = 2 \log N + 18$ either when $N = 2^k$ or when $N \neq 2^k$.

V. COMPLEXITY ANALYSIS

According to section 2 and 3, our 1-D DHT can be implemented by the following procedures ( \text{Input initial } p = \lfloor 2 \log_2 N / \log_2 \log_2 N \rfloor, t = \lfloor 2 \log N + 18 \rfloor )

1. Compute $x_{ij}$ and $n_{ij}$ using (2.3) and (3.1);
2. Compute $m_{n_{ij}}$ using our method in [7];
3. Compute $X(k)$ with (3.2).

Next we analyze the complexity of the algorithm. Step 1 constitutes a preprocessing step which involves equations (2.3) and (3.1). It is noteworthy that $[a_i \times 2^n]$, $S_u$ and $n_{ij}$ are real numbers only with relation to $N$ and can be obtained in advance, so these computations are not part of the real-time system. It remains to consider the generation of $x_{ij}$. In [4], we introduce a method to compute all $x_{ij}$ with $N^2 - \sum_{k=0}^{N-1} N / \gcd(k,N)$ integer additions.

To compute $m_{n_{ij}}$, we use the $p$-network method presented in [4], which require less than $(2p+2)(2p+3)(N-2)N / 2$ integer additions. The $p$-network shown in Fig.1 maps the vector $(1, x, x^2, \ldots, x^p)$ into $(1, (1+x), (1+x)^2, \ldots, (1+x)^p)$. We denote it by $F_p$: $F_p(1, x, x^2, \ldots, x^p)$

$= (1, (1+x), (1+x)^2, \ldots, (1+x)^p)$

In general $F_p^{-1}(1, x, x^2, \ldots, x^p) = (1, (n+1+x), (n+1+x)^2, \ldots, (n+1+x)^p)$

By substitution, $F_p^{-1}(1,1,1,\ldots,1) = (1,n,n^2,\ldots,n^p)$

Let $A_i = (a_i, a_i, a_i, \ldots, a_i)$ which is a $p+1$ dimensional vector, then we get

$F_p(F_p(F_p(\cdots(F_p(A_1)+A_2)+A_3)+\cdots+A_k)+A_i)$

$= F_p^{-1}(A_i) + F_p^{-1}(A_{i-1}) + \cdots + F_p^{-1}(A_2) + F_p^{-1}(A_1) + A_i$

$= (\sum_{i=1}^{n} a_i \sum_{i=1}^{n} a_i^2 \cdots \sum_{i=1}^{n} a_i^{ip})$

For emphasis, the above equation is rewritten as

$\text{Moment}(a_1, a_2, \ldots, a_i, a_i)$

$= F_p(F_p(\cdots F_p(F_p(A_1)+A_2)+A_3)+\cdots+A_k)+A_i$

$= (\sum_{i=1}^{n} a_i \sum_{i=1}^{n} a_i^2 \cdots \sum_{i=1}^{n} a_i^{ip})$

From section 3, the computation of (6) involves $(2p+1)(2 \log N + 18)N + 2N$ integer additions and $(2 \log N + 18) + 2p + 2N$ shifts.

Our algorithm seems to require a larger number of additions than many fast Hartley transforms, but it is easily implemented by systolic arrays efficiently because it only involves integer additions and shifts.

Besides, it has the following advantages. There are no multiplications in our method, which is superior to the $O(N \log_2 N)$ in classical FFT. Exponential functions have been replaced by simple polynomial functions and all multiplications have been changed into integer additions, reducing the computational cost and memory requirement. It can accommodate data samples of arbitrary length and produces nice accuracy and convergence property. If the sampled data are real numbers, the additions involved in our method are all real operations while in other methods they are complex operations.

© 2013 ACADEMY PUBLISHER
VI. SYSTOLIC ARRAYS FOR THE DHT

Since the proposed algorithm involves only integer operations, it is very suited to systolic implementation for a real-time processing. The general network for implementing it is shown in Fig.2. It consists of the moment generator arrays [4] and the shift and accumulation arrays.
The moments generator formed of $N-2$ sections of Pascal triangles with a row of adder-latch could be used to generate the 1-D moments. The shift and accumulation array (Fig. 3) receives $m_k(r)$ and computes (6), where $m_k(r)$ can be performed by shifting digits of $m_k(r)$ produced by the moment generator arrays to the left.
\[ \log N^{-1} - 10 \text{ places. The numbers in square brackets that are below the horizontal line denote the amount of time delay to keep synchronous pace in Fig.2.} \]

The moments generator was formed by \( N - 2 \cdot 2p + 1 \)-network which contains \( pN \) latches and \( 2p(2p + 1)N \) adder-latches [4]. The shift and accumulation array (Fig.2) can compute (6) once the moments are produced. It is noted that all the \( \sum_{r=1}^{m} (-j)^r n_{r} m_{k}^{r}(r) \) are produced in the same time. This array was formed by about \( pt \log(pt) \) adder-latches and some shifters and the execution time is about \( \log(p([2 \log N + 18] + 2)) \).

Fig.3 The shift and accumulation array

- \( \square \): latch  \( \bigcirc \): adder-latch
- \( \otimes \): shifter

\[ 1/2^{2 \log N + 20} \]

\[ X(k) \]
The scheduling of the dataflow is
\[
\sum_{i=0}^{t} \left( \sum_{r=1}^{2^{\log_{2}(N+20)}} \left( -1 \right)^{r} n_{r} m_{r}(r) \right) \times 2^{-i} / 2^{\log_{2}(N+20)}
\]
such that produced by the shift and accumulation array moves from left to right, and accumulates with \( x_{l,0} + m_{l}(0) \) to produce the \( X(k) \). The total execution cycle periods of the systolic arrays are
\[
N + (p + 1)(N - 2) + 2 \log(p[2\log_{2}(N + 18) + 2])
\]
. The first term \( N + (p + 1)(N - 2) \) is for producing moments \([5] \) and log(\( 2\log_{2}(N + 18) + 2 \)) for \((6) \). Since log(\( p[2\log_{2}(N + 18) + 2] \)) \(<(p + 1)(N - 2) \), the main time of the whole arrays, i.e., \((p + 1)(N - 2) \), is for producing the moments. The total number of adders is \( pt \log(pt) + p(p - 1)N/2 - pN \).

ACKNOWLEDGMENT

This work has been supported by the National Natural Science Foundation of China under contract Nos. 61105064 and 6172053, the Natural Science Foundation of Guangxi province of China under contract No. 2012GXNSFAA053230 and 2013GXNSFBA019279, Program for Excellent Talents Innovation Project of Guangxi Graduate Education Institutions([2011]40), Program for Excellent Talents in Guangxi Higher Education Institutions under contract Nos. 2011GXNSFBA009279, 2013GXNSFBA019279, Program for Excellent Talents Innovation Project of Guangxi Graduate Education Institutions([2011]40), Innovation Project of Guangxi Graduate Education under contract Nos. YCSZ2012073 and 201110590811M24.

REFERENCES


Zhenbing Liu received the Ph.D. (2010) degrees in Control science and engineering from Huazhong University of Science and Technology, China. He joined the School of Engineering and Automation at Guili University of Electronic Technology as an assistant professor in 2010, and is currently vice professor. His research interests include pattern recognition, machine learning, and image processing. He has published over 10 refereed papers in these areas, including those published in prestigious journals and top international conferences. He is also a frequent reviewer for various international journals and conferences.

Shujie Jiang, is pursing her master degrees in Guilin University of Electronic Technology since 2012. Her main research direction is image processing and computer vision.

Huihua Yang, Ph.D., professor of Guilin University of Electronic Technology(GUET), adjunct professor of Beijing University of Posts and Telecommunications, deputy secretary general of the Chinese Council of Near-Infrared Spectrum, ACM and CCF member. He received his bachelor's degree (1994) and master's degree (1997) at Huazhong University of Science and Technology in 1994 and 1997, respectively. He received Ph.D. degree at East China University of Science and Technology in 2005. He participated to work in department of computer science of GUET in 1997. He was a postdoctoral at analysis center of Tsinghua University from July 2005 to July 2007. Currently, he leads some major projects, such as the JT-G Green Channel inspection system, the algorithm software for crude oil fast evaluation funded by RIPP of Sinopec, a Guangxi Natural Science Fund on NIR Internet of Things, etc. He has completed several projects including one funded by National Science Foundation of China on modeling of NIR spectrum. He has published more than 30 papers in journals or conferences at home and aboard. His main research directions include intelligent information processing, machine learning, data mining, cloud computing, near-infrared spectroscopy and its application.
Jianguo Liu was born in Hubei, China in 1952. He received his B.S. degree in math from Wuhan University of Technology in 1983, M.S degree in math from Huazhong University of Science and Technology in 1984, both in Wuhan, Hubei province of China, and Ph.D degree in computer science from Hong Kong University in 1996 in Hongkong. He was a visiting scholar with the Medical Image Processing Group, Department of Radiology, University of Pennsylvania, Philadelphia, from December 1998 to July 2004. He is currently Professor and Deputy Director of the Key Laboratory of the State Education Ministry for Image Processing and Intelligent Control, Huazhong University of Science and Technology in Wuhan. His interests include signal processing, image processing, parallel algorithm and structure, and pattern recognition. Prof. Liu is a member of IEEE.