Signal Processing 88 (2008) 2316–2322

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Radix-3 fast algorithms for polynomial time frequency transforms

Guoan Bi *, Yingtuo Ju

School of Electrical and Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798, Singapore

ARTICLE INFO

Article history:
Received 19 April 2007
Received in revised form
17 March 2008
Accepted 24 March 2008
Available online 3 April 2008

Keywords:
FFT
Polynomial-phase signals
Polynomial time frequency transform
Decimation-in-frequency

ABSTRACT

The polynomial time frequency transforms have been used as an effective tool to reveal the polynomial-phase information by converting a one-dimensional polynomial-phase signal in the time domain into a multi-dimensional output array in the frequency domain. To significantly reduce the prohibitive computational complexity for dealing with high order polynomial-phase signals, efficient fast algorithms are extremely important for any practical applications. Based on radix-3 decomposition techniques, this paper presents fast algorithms for any order of the polynomial-phase signals. It shows that the computational complexity, except that for twiddle factors, of the radix-3 algorithm is independent of the order of the polynomial time frequency transform. The proposed algorithms are simple in concept and achieve significant savings on computational complexity.

1. Introduction

Polynomial-phase signals (PPSs) have been used in many applications, such as pulse compression radar systems [1], synthetic aperture radar (SAR) imaging [2] and mobile communications [3,4]. The $(r+1)$th order polynomial time frequency transform (PTFT) is generally defined as

$$X(k_0,k_1,\ldots,k_r)=\sum_{n=0}^{N_0-1}x(n)W_{N_0}^{k_0n}W_{N_1}^{k_1n^2}\ldots W_{N_r}^{k_rn^{r+1}}$$
$$0\leqslant k_i\leqslant N_i-1,\ 0\leqslant i\leqslant r \tag{1}$$

where $W_N=\mathrm{e}^{-\mathrm{j}2\pi/N}$, $x(n)$ is the one-dimensional (1D) input sequence of N_0 points and N_i is the size of the ith dimension of $X(k_0,k_1,\ldots,k_r)$. In practice, it is often that $N_j\geqslant N_i\geqslant N_0$ for $j>i>0$ to achieve a satisfactory accuracy for parameter estimation [5]. It was shown that when dimensional sizes are different, a simple decomposition method can be applied to divide the overall computational tasks into many smaller ones whose dimensional sizes are the same [6]. Therefore, it is assumed in this paper that all dimensional sizes of the PTFT are the same.

The PTFT of $x(n)$ is equivalent to calculating $N_1,N_2,\ldots,N_r$ 1D fast Fourier transforms (FFTs) [7,8], which is similar to the row–column method for multi-dimensional DFTs. Although the algorithms of 1D FFT are available for use, such computation of the PTFT still requires a huge computational complexity. For a better computational efficiency, fast quadratic-phase transform [5] was proposed as an efficient algorithm for the 2nd order PTFT to reduce the computational complexity by a factor of $\log_2 N$ compared with that needed by directly using the 1D FFTs. The fast algorithm based on decimation-in-time decomposition [9] reduces the overall computational complexity by exploiting some properties of the PTFT. However, further reduction on computational complexity can be achieved because some properties of the PTFT are not fully utilized. Recent work [6,10,11] was reported to achieve a better computational efficiency for an arbitrary order of the PTFT whose dimensional size is a power of two.

It is noted that these reported fast algorithms support sequence length being a power of two only. Fast algorithms for other sequence lengths are also desired for applications that need sequence lengths other than a

* Corresponding author. Tel.: +65 6790 4823; fax: +65 6793 3318.
E-mail address: egbi@ntu.edu.sg (G. Bi).

0165-1684/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.sigpro.2008.03.018

power of two. It is well known that when the sequence length is not supported by the available fast algorithms, zero padding techniques have to be used to augment the input sequence to the next available size supported by the fast algorithm. This mismatch obvious wastes the computational resources and decreases the computational efficiency of the system. The availability of fast algorithms based on radix numbers other than two is very useful to minimize the possibility and the degree of the mismatch between the input sequence length and the transform sizes supported by the fast algorithm. Fast algorithms for discrete Fourier transform based on radices other than the power of two were reported in the literature (see [7,8], for example). Because the fast algorithms for PTFT are developed in the last few years only, there has been no reported fast algorithms that are specially designed for sequence lengths that are not a power of two.

This paper presents methods of efficiently decomposing the PTFT for any order of the PPSs whose dimensional size has a factor of a power of three. The proposed algorithm is general enough for dealing with any order of the PTFTs, has the regular computational structure, and also obtains a very good computational efficiency compared with most reported fast algorithms. By combining other factors in the sequence lengths, it shows that the proposed fast algorithm can be used to efficiently support many different sequence lengths. Similar to the development of fast algorithms for the discrete Fourier transform, the periodic and symmetric properties of the transform kernel are used for the design of fast PTFT algorithms. However, the PTFT has its unique properties to be used for minimizing the computational complexity, which will be demonstrated in the following sections.

The rest of this paper is organized as follows. The next section presents the radix-3 algorithms for the 3rd order PTFTs. In particular, details are presented for reusing the partially computed results to minimize the required computational complexity. With understanding of the fast algorithm for the 3rd order PTFT, Section 3 presents the radix-3 algorithm for any arbitrary order of PTFTs. In Section 4, the computational complexities required by the proposed algorithm are also analyzed and compared with other reported ones. Section 5 presents the conclusions.

2. Fast algorithm for the 3rd order PTFT

The 3rd order PTFT of a 1D length-N input sequence $x(n)$ is defined as

$$X(k_0,k_1,k_2)=\sum_{n=0}^{N-1}x(n)W_N^{k_0n+k_1n^2+k_2n^3} \tag{2}$$

where $W_N=\mathrm{e}^{-\mathrm{j}2\pi/N}$, and $k_i=0,1,\ldots,N-1$ for $i=0,1$ and 2. When $N=M\times 3^p$, where $p>1$ and M is a positive integer, it can be easily verified, by using the periodic property of W_N, that the computation task defined in (2) can be expressed as

$$\begin{aligned}X(3k_0+l_0,3k_1+l_1,3k_2+l_2)&=\sum_{n=0}^{N-1}x(n)W_N^{(3k_0+l_0)n+(3k_1+l_1)n^2+(3k_2+l_2)n^3}\\&=\sum_{n=0}^{N-1}x(n)W_N^{l_0n+l_1n^2+l_2n^3}W_{N/3}^{k_0n+k_1n^2+k_2n^3}\end{aligned} \tag{3}$$

where $0\leqslant k_0,k_1,k_2\leqslant N/3-1$ and $l_i\in\{0,1,2\}$ for $i=0,1$, and 2. When $l_0=l_1=l_2=0$, we have

$$X(3k_0,3k_1,3k_2)=\sum_{n=0}^{N/3-1}\left\{\sum_{m=0}^{2}x\left(n+\frac{mN}{3}\right)\right\}\times W_{N/3}^{k_0n+k_1n^2+k_2n^3} \tag{4}$$

which is a 3rd order length-($N/3$) PTFT. For other combinations of l_0,l_1 and l_2, (3) can be rewritten into

$$\begin{aligned}X(3k_0+l_0,3k_1+l_1,3k_2+l_2)&=\sum_{n=0}^{N/3-1}\left\{\sum_{m=0}^{2}x\left(n+\frac{mN}{3}\right)\times W_N^{l_0(n+mN/3)+l_1(n+mN/3)^2+l_2(n+mN/3)^3}\right\}\\&\quad\times W_{N/3}^{k_0n+k_1n^2+k_2n^3}\end{aligned} \tag{5}$$

When $N>3$, (5) can be simplified, by using the periodic property of W_N, into

$$\begin{aligned}X(3k_0+l_0,3k_1+l_1,3k_2+l_2)&=\sum_{n=0}^{N/3-1}\left\{x(n)W_N^{l_0n+l_1n^2+l_2n^3}+x\left(n+\frac{N}{3}\right)\times W_N^{l_0(n+N/3)+l_1(n+N/3)^2+l_2(n+N/3)^3}\right.\\&\quad\left.+x\left(n+\frac{2N}{3}\right)W_N^{l_0(n+2N/3)+l_1(n+2N/3)^2+l_2(n+2N/3)^3}\right\}\times W_{N/3}^{k_0n+k_1n^2+k_2n^3}\end{aligned} \tag{6}$$

$$\begin{aligned}&=\sum_{n=0}^{N/3-1}\left\{x(n)W_N^{l_0n+l_1n^2+l_2n^3}+x\left(n+\frac{N}{3}\right)\times W_N^{l_0(n+N/3)+l_1(n^2+2N/3)+l_2n^3}\right.\\&\quad\left.+x\left(n+\frac{2N}{3}\right)W_N^{l_0(n+2N/3)+l_1(n+4N/3)+l_2n^3}\right\}\times W_{N/3}^{k_0n+k_1n^2+k_2n^3}\end{aligned} \tag{7}$$

Eq. (7) is obtained from (6) after eliminating the terms, in $(n+iN/3)^2$ and $(n+iN/3)^3$ for $i=1$ and 2, that are wholly divisible by N. Therefore, (7) can be further expressed as

$$\begin{aligned}X(3k_0+l_0,3k_1+l_1,3k_2+l_2)&=\sum_{n=0}^{N/3-1}\left\{x(n)+x\left(n+\frac{N}{3}\right)W_N^{(l_0+2l_1n)N/3}+x\left(n+\frac{2N}{3}\right)W_N^{(2l_0+l_1n)N/3}\right\}\\&\quad\times W_N^{l_0n+l_1n^2+l_2n^3}W_{N/3}^{k_0n+k_1n^2+k_2n^3}\\&=\sum_{n=0}^{N/3-1}\left\{x(n)+x\left(n+\frac{N}{3}\right)W_N^{(l_0+2l_1n)N/3}+x\left(n+\frac{2N}{3}\right)W_N^{-(l_0+2l_1n)N/3}\right\}W_N^{l_0n+l_1n^2+l_2n^3}\\&\quad\times W_{N/3}^{k_0n+k_1n^2+k_2n^3}\\&=\sum_{n=0}^{N/3-1}[u_3(n,l_0,l_1,l_2)W_N^{l_0n+l_1n^2+l_2n^3}]W_{N/3}^{k_0n+k_1n^2+k_2n^3}\end{aligned} \tag{8}$$