ADAPTIVE BEAMFORMING USING GENERALIZED TRIGONOMETRIC TRANSFORM DOMAIN LMS ALGORITHM

A Dissertation submitted towards the partial fulfilment of
the requirement for the award of degree of

Master of Technology in
Microwave and Optical Communication Engineering

Submitted by
Rishab Dhawan
2K14/MOC/14

Under the supervision of
Dr. Priyanka Jain
Assistant Professor

Department of Applied Physics and Department of Electronics & Communication Engineering

Delhi Technological University
(Formerly Delhi College of Engineering)

JUNE 2016

DELHI TECHNOLOGICAL UNIVERSITY

Established by Govt. of Delhi vide Act 6 of 2009
(*Formerly Delhi College of Engineering)*
SHAHBAD DAULATPUR, BAWANA ROAD, DELHI-110042

CERTIFICATE

This is to certify that the work which is being presented in the dissertation entitled "**Adaptive Beamforming using generalized Trigonometric Transform Domain LMS Algorithm**" is the authentic work of **Rishab Dhawan** under my guidance and supervision in the partial fulfillment of requirement towards the degree of Master of Technology in Microwave and Optical Communication Engineering jointly run by Department of Applied Physics and Department of Electronics and Communication in Delhi Technological University during the 2014-16.

As per the candidate declaration this work has not been submitted elsewhere for the award of any other degree.

Dr. Priyanka Jain	**Prof. Rajesh Rohilla**	**Prof. S. C. Sharma**
Supervisor	Head of Department	Head of Department
Assistant Professor	Deptt. of ECE	Deptt. of Applied Physics
Delhi Technological University	Delhi Technological University	Delhi Technological University

DECLARATION

I hereby declare that all the information in this document has been obtained and presented in accordance with academic rules and ethical conduct. This report is my own, unaided work. I have fully cited and referenced all material and results that are not original to this work. It is being submitted for the degree of Master of Technology in Engineering at the Delhi Technological University. It has not been submitted before for any degree or examination in any other university.

Rishab Dhawan

M. Tech, MOCE

2K14/MOC/14

ABSTRACT

In the recent years, the use of wireless technology has grown rapidly. This has led to increase in number of users and services. To meet Quality of Service (QoS) standards in reception and transmission and track a wider coverage area keeping in mind limited spectrum available, spatial processing was devised as a practical solution. Spatial processing technique is used to discriminate two signals on the basis of location with respect to an array of antennas using the knowledge of the attributes of the signal. In a multiple access system, some users may occupy the same frequency bands allotted to them, thus frequency domain filtering techniques cannot be used to separate signals coming from these users. Beamformers employ spatial processing technique to estimate the location of the desired and interfering signals and adjust their reception radiation pattern to provide maximum amplitude in the direction of desired signals and maximum rejection in the direction of interfering signals in the presence of Gaussian noise.

Adaptive Algorithms are used to adjust the radiation pattern by tuning the weights of an antenna array. The most commonly used algorithm for tuning the weights is Least Mean Sqaures algorithm. It is least computationally complex but it suffers from slow convergence of the beamformer output to the desired output. Many alternatives have been devised in the past. One such alternative is using Discrete Cosine Transform and Discrete Fourier Transform to decorrelate the input data which leads to faster convergence. In this thesis, generalized transform architecture is proposed which uses these transforms as well as Discrete Sine Transform and Discrete Hartley Transform and employed with conventional LMS algorithm to study the convergence of beamformer output using these transforms.

Mathematical analysis of Least Mean Square Algorithm and Transform Domain Least Mean Square Algorithm has been discussed. Both of these algorithms have been implemented in MATLAB and LabVIEW and the results have been analysed with varying input correlation parameters.

Keywords: Beamforming, LMS, Transform Domain LMS, DFT, DST, DCT, DHT

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my project supervisor, Asst. Professor Dr. Priyanka Jain, for her supervision, invaluable guidance, motivation and support throughout the extent of the project. I have benefitted immensely from her wealth of knowledge.

I would also like to thank Dr. Ajeet Kumar (Physics Department) and Dr. Neeta Pandey (ECE Department) for their precious suggestions, support and technical help during the course of this project.

My gratitude is extended to my colleagues and friends who have not been mentioned here personally in making this project a success.

Last but not least, I want to thank my parents, for inculcating good ethos, as a result of which I am able to do my post-graduation from such an esteemed institution. I would thank my friends for believing in my abilities and for always showering their invaluable support and constant encouragement.

Rishab Dhawan

M. Tech, MOCE

2K14/MOC/14

LIST OF PUBLICATIONS

JOURNAL PUBLICATIONS

- Rishab Dhawan, Priyanka Jain, "Unified Architecture for Discrete Trigonometric Transforms and their Inverse using Recursive filter structures", Springer Journal of Circuits, Systems and Signal Processing, 2016 (Under Review)

TABLE OF CONTENTS

LIST OF FIGURES

FIGURES

LIST OF TABLES

TABLES

LIST OF ABBREVIATIONS

QoS	:	Quality of Service
ASICs	:	Application Specific Integrated Circuits
SINR	:	Signal to Interference plus Noise Ratio
DOA	:	Direction Of Arrival
SOI	:	Signal Of Interest
SNOI	:	Signal Not Of Interest
SAR	:	Synthetic Aperture Radar
θ	:	Elevation Angle
ϕ	:	Azimuth Angle
LCMV	:	Linearly constrained minimum variance
TDLMS	:	Transform Domain Least Mean Square
ULA	:	Uniform Linear Array
x(n)	:	Input Signal received at the antenna array
y(n)	:	Beamformer Output
AOA	:	Angle of Arrival
θ_{d1}	:	Angle of Arrival of desired signal 1
θ_{t1}	:	Angle of Arrival of interference source 1
θ_{t2}	:	Angle of Arrival of interference source 2
τ_i	:	Time delay between two consecutive elements for i^{th} signal
θ_i	:	Angle of Arrival of i^{th} signal
v	:	Speed of signal
$\emptyset_i$	:	Serial Transmission Line
w_c	:	Sampling frequency in radians
f	:	Sampling frequency in hertz
$a(\theta_i)$	:	Steering vector
$s_i(n)$	:	Incoming signal
$N(n)$	:	Additive white Gaussian noise
w	:	Weights of beamformer
H	:	Hermitian operator
PDF	:	probability density function
RLS	:	Recursive Least Square

DST	:	Discrete Sine Transform
DCT	:	Discrete Cosine Transform
DFT	:	Discrete Fourier Transform
DHT	:	Discrete Hartley Transform
DWT	:	Discrete Wavelet Transform
OFDM	:	Orthogonal Frequency Division Multiplexing
KLT	:	Karhunen-Louve Transform
CEM	:	Computational Electromagnetics
MoM	:	Method of Moments
$\xi(n)$	:	Optimal Cost Function
E[]	:	Expectation operator
MoM	:	Method of Moments
MoM	:	Method of Moments
MoM	:	Method of Moments

CHAPTER 1

INTRODUCTION

1.1 Basic overview

In the recent years, the use of wireless technology has grown rapidly. This has led to increase in number of users and services. To meet Quality of Service (QoS) standards in reception and transmission and track a wider coverage area keeping in mind limited spectrum available, spatial processing was devised as a practical solution. Spatial processing may be defined as discriminating two signals on the basis of location with respect to an array of antennas using the knowledge of the attributes of the signal. In a multiple access system, some users may occupy the same frequency bands allotted to them, thus frequency domain filtering techniques cannot be used to separate signals coming from these users. This is where spatial discrimination idea is exploited. The desired and interfering signals have different spatial location origin, thus spatial filtering can be used at the receiver to separate these signals.

Beamforming is based on the idea of spatial processing. Beamformers are powerful processors or Application Specific Integrated Circuits(ASICs) employ spatial processing techniques to estimate the location of the desired and interfering signals and adjust their reception radiation pattern to provide maximum amplitude in the direction of desired signals and maximum rejection in the direction of interfering signals in the presence of Gaussian noise. This maximizes the Signal to Interference plus Noise Ratio (SINR). This is achieved using an array of antennas separated at a physical distance. The discrete incoming signal data received at these antennas are then scaled and linearly combined using various techniques listed in the following sections to obtain maximum reception in desired direction. Adaptive array antennas have two main advantages as discussed below [1].

The physical aperture of an antenna determines the spatial discrimination capability. The size of spatial aperture in multiples of wavelengths is important rather than absolute aperture size. Considering the case of a single antenna with continuous spatial aperture, the discrimination is proportional to the physical size. For high frequencies, the antenna sensor is practically feasible since the wavelength is small. However, for low frequency signals, the antenna sensor is generally not practically feasible since the physical size required is very large. An array of

sensors can be used which gives a much larger spatial aperture size as compared to a single physical antenna.

A second very important advantage of using an array of sensors is the flexibility of discrete spatial filtering. In real time systems, the spatial filtering function needs to be updated at regular intervals in order to maintain effective suppression of interfering signals. Changing the spatial filtering function for a continuous aperture antenna requires change in geometry or shape which is not feasible.

For a better understanding of how adaptive antennas work, let us consider a practical example of adaptive antenna-human analogy [2]. Assume two people conversing in a room free of any other speakers and noises as shown in figure 1.1. If the speaker (analogous to desired source) amongst the two moves, then the listener is able to judge the movement of the speaker due to different arrival times at the ears of the listener (analogous to antenna array). This processing is done by the brain of the listener (analogous to beamformer processor). The brain processes the time-delays between the voice received at each ear and constructively adds them to strengthen its amplitude and focus on its Direction Of Arrival (DOA). If additional speakers join the conversation, the brain of the listener can reject the other speakers and synchronize itself with the desired speaker. Also, since the listener has knowledge of direction of desired speaker, it can converse back by responding in the direction of desired speaker.

Electrical adaptive antennas use antenna array instead of two ears and a digital signal processor instead of human brain to perform these tasks. The antenna arrays sample the incoming signals, and the beamformer, which is nothing but a digital signal processor, process these spatial samples to adjust the weights of the array to produce a radiation pattern that maximum amplitude in the direction of Signal Of Interest (SOI) and ideally rejecting all the other undesired speakers or Signal Not Of Interest (SNOI).

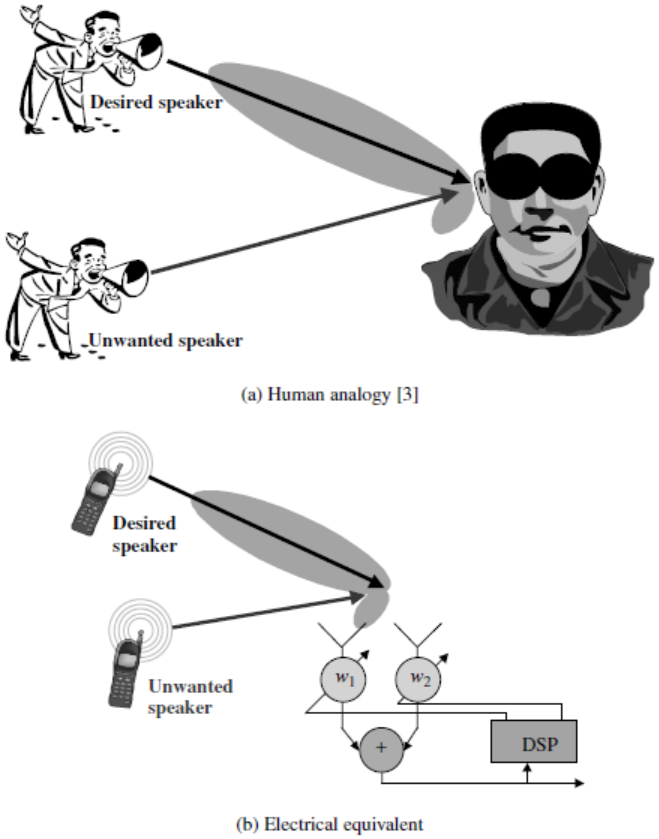

Fig 1.1 Adaptive Antenna-Human analogy

Beamformers have several applications in fields of

- Radar: synthetic aperture radar(SAR), phased array radar, air traffic control[3,4,5]
- Sonar: source localization and classification[3,4]
- Wireless Communications: cell sectoring, cell broadcasting, satellite communications, directional reception and transmission[5,6]
- Seismology: Oil mapping, Earth crust mapping[7,8]
- Imaging: Optical, Tomographic, Ultrasound, High Resolution Earth Imaging[9-11]
- Bio-medical: Cancer detection[11], Heart Monitoring
- Acoustics: Hearing Aids, Echo and Noise cancellation[12]

There are different types of array geometries (1D, 2D and 3D). Some commonly used ones are linear, circular and planar as shown in figure 1.2. The linear geometry suffers from mirror image ambiguity since its radiation pattern is symmetrical to the axis of antenna which is also known as the end-fire axis. The other two do not have this disadvantage. The radiation pattern of the

antenna array depends mainly on geometry, inter-element spacing and the amplitude and phase of the feed. There are some basic criteria regarding the inter-element spacing. If the inter-element spacing increases, the number of grating lobes increases which are nothing but direction of maximum radiation. For uniform (constant) inter-element spacing, the maximum spacing between the elements is half wavelength[1]. This yields no grating lobes. Grating lobes result in 'aliasing' where two distinct spatial location produces the same phase difference between consecutive elements of the array thus the array is not able to distinguish between the two distinct spatial locations. This ambiguity is shown in figure 1.2.d). Location here, in general, is a three dimensional quantity. We often are interested in DOA which may be a one dimensional (θ) or two dimensional quantity (θ, ϕ).

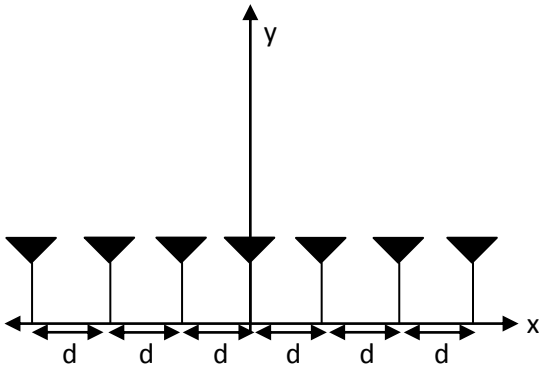

Fig 1.2.a Linear geometry array

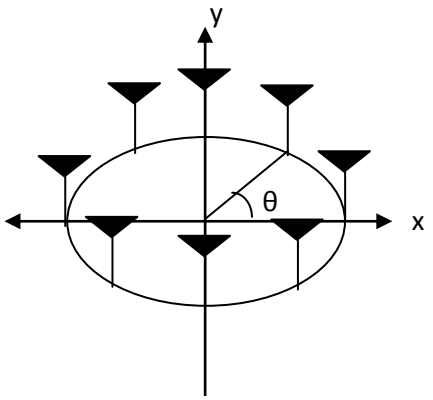

Fig 1.2.b Circular geometry array

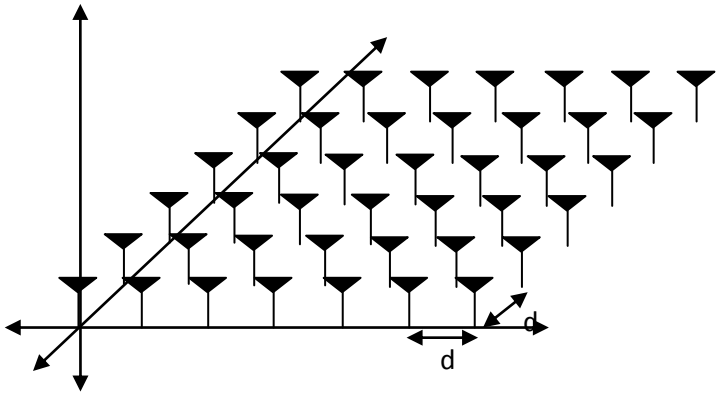

Fig 1.2.c Planar geometry array

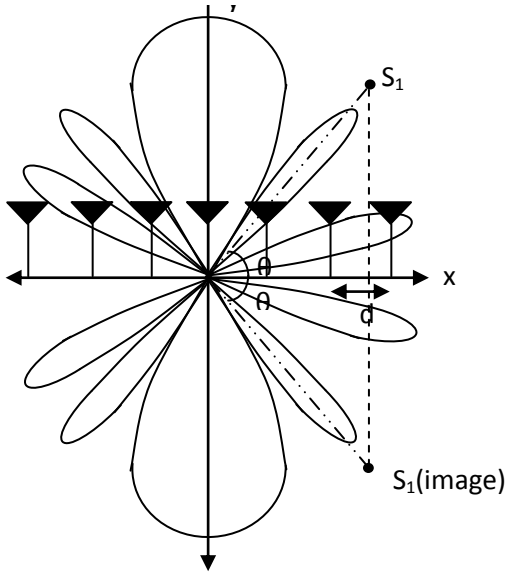

Fig 1.2.d Linear array ambiguity

In the past, various schemes have been proposed to select the weights of beamformer. Each of these schemes has their own characteristics and schemes. The simplest beamformer, a conventional beamformer, also known as a delay and sum beamformer, has equal weights. The phases are electronically controlled to steer the array in a particular direction [13]. Another type of beamformer is null steering. It places nulls in the DOA of the interfering signals. It provides strong rejection and it may be repeated for multiple interfering signals [13]. Conventional and null-steering beamformers require the knowledge of DOA of signals. Also, they do not maximize the SINR by estimated weights. Adaptive beamforming overcomes these limitations. The weights are selected by minimizing a particular output signal while the SNR is maximized. Adaptive array beamforming can be divided into two broad categories; element-space processing and beam-space processing [13]. The signal output of the elements is scaled to produce a desired output in element space processing while beam-space processing involves obtaining a beam output from each element and then scaling and combining these beams to produce desired array output. Beam-space antenna is less computational intensive but lack the versatility of element-space processing. Various beam-space processors have been studied, including howell-applebaum array[14,15] using the spread-spectrum technique which is based on the statistical approach of continuously varying the steering array to point the beam in the direction of desired signal and iteratively nullifying the jammers thus requiring a prior knowledge of DOA; multiple sidelobe canceller[16] where a main beam antenna is used with adaptive auxiliary elements to cancel the interference from different directions, which might lead to desired signal cancellation if the desired and interfering signals are correlated; partially adaptive array[17] where some elements are adaptively weighted whereas the rest are fixed weighted thus reducing complexity at the cost of degrading performance; digital beamforming[18,19] using conjugate gradient methods instead of stochastic gradient methods but requiring a prior knowledge of DOA; partitioned processor[20,21], multibeam antennas[22] where the inputs are decoupled to increase convergence rate of howell applebaum arrays; regenerative hybrid array[23] where a desired signal is used additionally to that of an applebaum array; generalized side-lobe cancellor [24] which is also known as linearly constrained minimum variance(LCMV) beamformer. It was first proposed by frost [25] and implemented by Griffith and jim[24]. More commonly, Least Mean Square (LMS) adaptive filter is used with discrete transforms to adapt weights and to increase the convergence rates by decorrelating the input data [26]. The degrees of freedom are directly

proportional to number of unwanted signals, rather than number of weights in element-space methods. Recently, robust arrays have been developed to deal with errors in input signals. In this thesis, the conventional beamformer has been used and an adaptive transform domain LMS(TDLMS) filter is employed to increase the convergence rate. The complexity of TDLMS is similar to that of original LMS schemes.

1.2 Beamformer System Model

Let us consider the example of a uniform linear array (ULA). The input signal x(n) and output signal y(n) is modeled. Consider an array of L elements as shown in figure 1.3, with an inter-element distance d and angle of arrival (AOA) of desired signals θ_{d1} and for interference source θ_{t1} and θ_{t2}. The elements are placed along the x-axis (axis of antenna) and first element is taken to be the reference element.

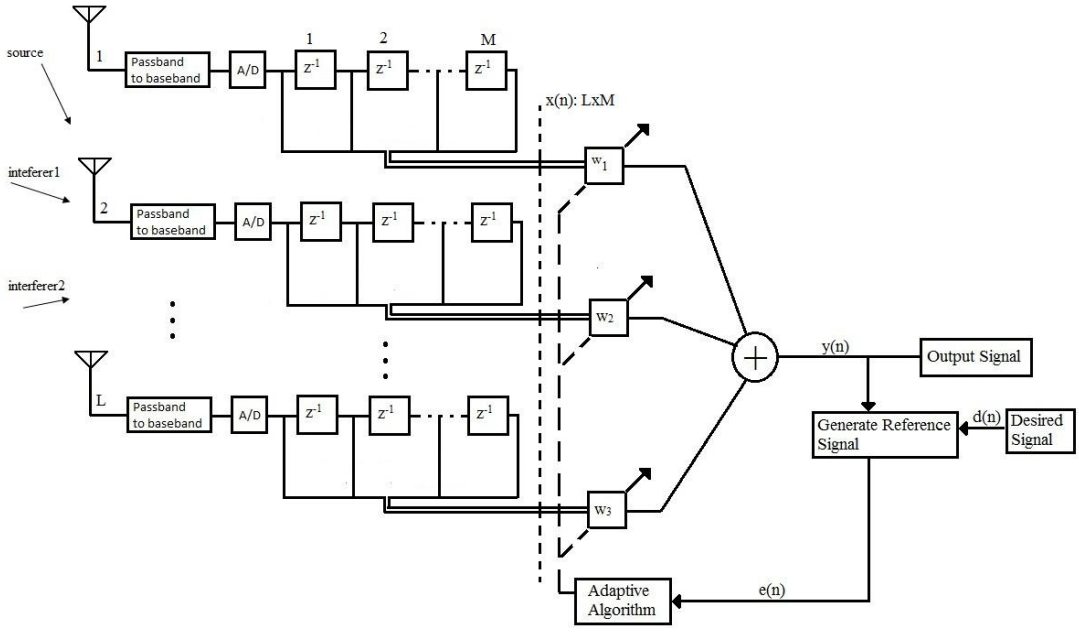

Fig 1.3 System model for ULA

The inter-element phase shift can be found out mathematically. First, time delay between two consecutive elements τ_i is found out. It is given by equation

$$\tau_i = d \cos \theta_i / v \tag{1.1}$$

Where v is the speed of signal (sound or electromagnetic waves)

The corresponding phase shift $\emptyset_i$ is given by equation

$$\emptyset_i = w_c.\tau_i \tag{1.2}$$

$$\emptyset_i = 2\pi f d \cos\theta_i / v \tag{1.3}$$

According to [27], the array output vector is given by

$$x(n) = a(\theta_i)s_i(n) \tag{1.4}$$

Where $a(\theta_i)$ is steering vector and given by

$$a(\theta_i) = \begin{bmatrix} 1 \\ \exp(-j\emptyset_i) \\ \exp(-2j\emptyset_i) \\ . \\ . \\ . \\ \exp(-(L-1)j\emptyset_i) \end{bmatrix}_{L\times 1} \tag{1.5}$$

And $s_i(n)$ is the incoming signal with M samples. For P sources, the total output vector is given by

$$x(n) = \sum_{i=1}^{P} a(\theta_i)s_i(n) \tag{1.6}$$

Assuming the environment to be noisy, we add white Gaussian noise to the signal. Hence, we get the signal model used in array processing

$$x(n) = \sum_{i=1}^{P} a(\theta_i)s_i(n) + N(n) \tag{1.7}$$

Where $N(n)$ is white Gaussian noise .

The dimensions of x(n) is L×M.

The array output is given by

$$y(n) = w^H.x(n) \tag{1.8}$$

The array factor is given as

$$AF = \sum_{i=1}^{L} w_i.exp(-j(i-1)2\pi f d \cos\theta / v) \tag{1.9.a}$$

$$0 \le \theta \le \pi \tag{1.9.b}$$

1.3 Beamformer classification

On the basis of how weights are chosen, beamformers are typically classified as data independent and data dependent. As the name suggests, the weights of a data independent beamformer does not depend on the spatially sampled input data and are specified for all signal and interference sources scenarios. The weights in a data dependent beamformer are chosen based on second order statistics (correlation matrix) of the input data to optimize the array response so as to maximize the SINR ratio at beamformer output. The interferers are suppressed by placing nulls in their direction.

Determining the second order statistics is a tedious computational task, but if the signal can be assumed using the probability density function (PDF), the statistics and the optimum weights can be determined from the available data. Now, since these statistics vary temporally due to moving signal and interfering sources, the weights can be adapted accordingly at regular intervals either by Block adaptation or by Continuous adaptation [26]:

- Block adaptation[28]: the weights are determined from statistics estimated using a block of data updated at regular time intervals.
- Continuous adaptation[26]: the weights are determined from statistics estimated using every new data sample

For a non-stationary environment, block adaptation is preferred. However, if number of adaptive weights is large (more than 50) or data is time-varying, continuous adaptation is preferred.
In the next section, we discuss about various adaptive algorithms used in adaptive beamforming.

1.4 Adaptive Algorithms

Various beamforming algorithms have been proposed in the literature, that include LMS based algorithms [71, 29] and Recursive Least Square (RLS) based algorithms [71]. The LMS algorithm finds the filter coefficients by minimizing the square of error signal which is nothing but difference between desired signal and the output signal. The RLS, on the other hand, reduces a weighted linear least square function. The LMS algorithm is less computationally complex [29], but its convergence rate is slow as compared to RLS algorithm when the input signals are correlated. Many variants have also been proposed in the past [30-32, 26]. The improved convergence rate comes at the cost of increased computational complexity. One such proposed method is transform domain LMS algorithm. The TDLMS employs orthogonal transforms that

reduces the correlation between the input signals thus trying to improve the convergence rate [26]. Power normalization is done on the transformed inputs to reduce eigen value spread of the input signal correlation matrix. The common orthogonal transforms that are used are Discrete Sine Transform (DST), Discrete Cosine Transform (DCT), Discrete Hartley Transform (DHT), Discrete Fourier Transform(DFT), Discrete Wavelet Transform(DWT), Slant Transform, Haar Transform. Among these transforms, DST, DCT, DHT, DFT are the least complex. The TDLMS was first proposed by Narayan et. al [26], in which DFT and DCT were employed In this thesis, the work has been concentrated on these transforms which have been implemented using a unified architecture. The transforms are employed to the conventional LMS algorithms for various input signals have different correlation coefficient values (ρ) where $0 < |\rho| \leq 1$ and the convergence is analyzed for each transform. The advantage of a unified architecture is that we can switch between the mentioned transforms for varying input, thus providing optimum performance for input signals whose statistics may change over time.

1.5 Thesis Objective

This thesis is aimed at

- Developing a unified architecture for above mentioned transforms and their inverse as well and implementing the same in MATLAB and LabVIEW.
- Implementing the conventional LMS algorithm in MATLAB and LabVIEW and analyzing the convergence of the algorithm in multiple simulated environments with input signals having different correlation coefficient values
- Employing the unified architecture with the conventional LMS algorithm for different environment and analyze the convergence of these algorithms for each transform.

1.6 Thesis Organization

The rest of the thesis has been organized has been organized as follows

Chapter 2 aims at developing a unified architecture for DST, DCT, DHT, DFT and their inverse as well and comparing the performance with previously developed unified architectures.

Chapter 3 includes the mathematical analysis of LMS algorithm. LMS algorithm has been studied and formulated. An iterative approach is followed to minimize the optimal cost function. The TDLMS algorithm has also been discussed and formulated.

Chapter 4 analyzes the simulation results of proposed beamformer system model using LMS adaptive algorithm and TDLMS adaptive algorithm in MATLAB and LabVIEW.

Chapter 5 summarizes the conclusion and future scope.

CHAPTER 2

IMPLEMENTATION OF UNIFIED ARCHITECTURE

2.1Introduction

In the digital domain, various types of discrete trigonometric transforms such as DFT [33], Z Transform, DCT [34], DST [35], DHT [36-37], Walsh Transform, Discrete Hadamard Transform and slant transform are used. These transforms have played a significant role in signal processing for a number of years, and therefore, transform coding continues to be a topic of interest in theoretical as well as for applied work in this field. DHT has been established as a potential tool for signal processing and communication applications e.g computation of convolution and deconvolution [38-39], interpolation of real valued signal [40], Optical Orthogonal Frequency Division Multiplexing (OFDM) [41], multicarrier modulation [42] and many other applications. DHT uses the transform kernel similar to that of DFT, except that it is a real valued transform. All the properties applicable to DFT such as convolution and shifting theorem apply to DHT as well. Due to energy compaction property, DCT and DST are most widely used transform in speech and image processing applications such as block filtering [43], transform domain adaptive filtering [44], digital signal interpolation [45], adaptive beamforming [46,47], image resizing [35,48,49], speech enhancement [50] and so forth. Both DST and DCT are good approximation to the statically optimal Karhunen-Louve Transform (KLT) [35]. It is found that in case of signal with high correlation coefficient, DCT based coding results in better performance but for low correlation signal, DST gives lower bit rate [35]. The above mentioned transforms (DFT/ DHT/ DST/ DCT) are also used in Computational Electromagnetics (CEM) in modeling radiation problems using Method of Moments (MoM) [51, 52, 66].

In the literature, there are many implementation methods proposed for unified discrete trigonometric transform computation [55-64]. Liu et. al. proposed a unified parallel lattice structure and an IIR filter structure for time recursive DXT [55]. In [57], a hardware efficient unified systolic architecture for sliding window DXT is designed based on CORDIC arithmetic unit. In [59, 60], a unified arrays for DXT computation are either based on one time even-odd portion of frequency samples or Clenshaw's recurrence formula. Hsiao et. al. presented the unified architecture based on systolic mapping of the common kernel operation with fewer

processing elements[58]. Cyclic convolution approach has been used in [61] for obtaining unified structure. In most of the developed unified methods, implementation of inverse transform in the unified form has not been shown. In this paper, we have proposed unified recursive structure that can be used for the computation of discrete trigonometric transforms (DFT/ DHT/ DST/ DCT) as well as their inverse.

The motivation of this work is to propose a simple architecture which can compute all these transforms can serve the purpose of a general purpose DSP chip. The proposed unified architecture can be used in applications such as High Speed Optical Networking (HSON)[69], Multitone-systems where high SNR is required[68], adaptive beamforming applications [46,47], bio-medical applications where real time spectral analysis of signals like ECG, EEG, digital ultrasonography, EMG, EGG can be done using DFT and DHT and data obtained can be archived and stored in compressed format using DCT and DST. A single FPGA chip that can perform all the above applications would be highly resource-efficient. The main advantage of this architecture is that the same FPGA chip can be used for inverse transformation as well. The above unified approach can be extended for other transforms to get a generalized orthogonal transform FPGA chip.

The basic structure of all the transforms DST/DHT/DST and DCT are almost equivalent and this property has been exploited in the design of unified architecture. Each of the transform has a unique data arrangement format. To address this problem, array-indexing architecture has been adopted for implementation.

2.2 Derivation of Forward Transforms

2.2.1 Discrete Sine Transform

The recursive structure for DST as shown in figure 2.1 has already been proposed by Jain *et. al.*[54]

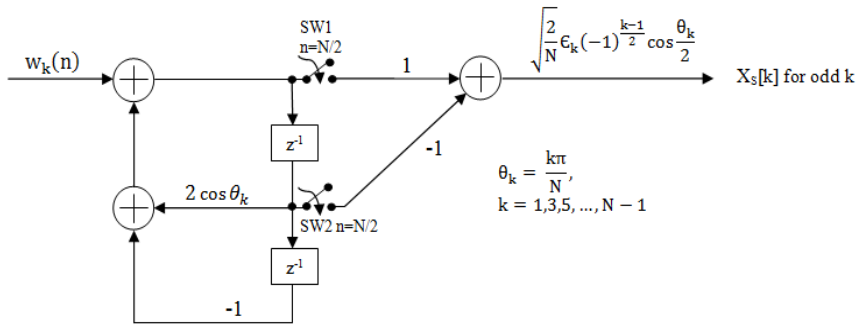

Fig 2.1.a Lattice structure implementation of DST odd coefficients

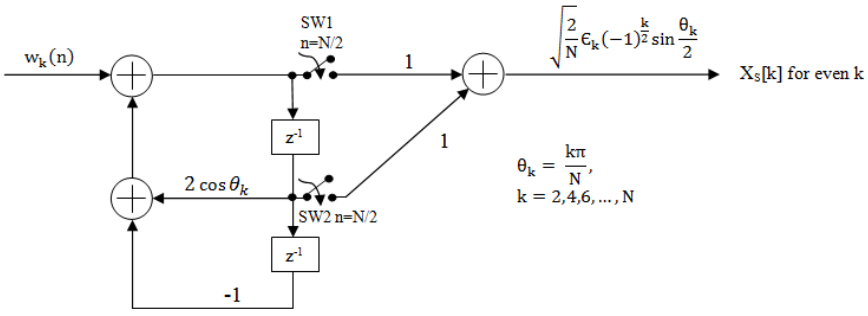

Fig 2.1.b Lattice structure implementation of DST even coefficients

2.2.2 Discrete Cosine Transform

The recursive structure for DCT as shown in figure 2.2 has already been proposed by *Wang et. al.*[55]

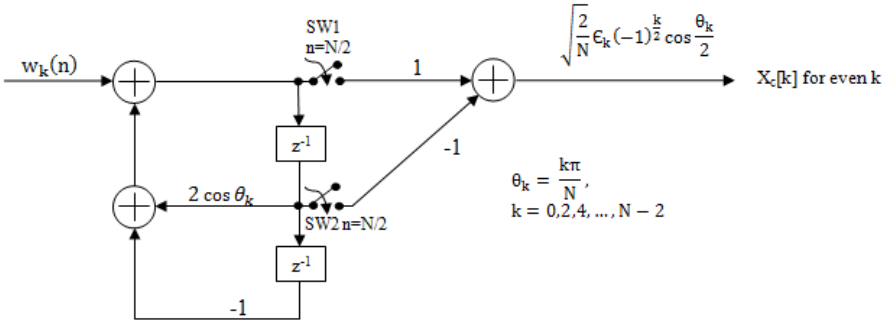

Fig 2.2.a Lattice structure implementation of DCT even coefficients

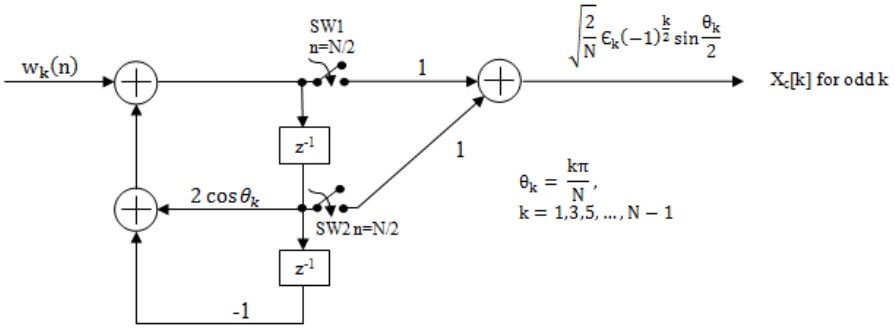

Fig 2.2.b Lattice structure implementation of DCT odd coefficients

2.2.3 Discrete Fourier Transform

The one dimensional DFT is given by

$$X_F[k] = \frac{1}{\sqrt{N}}\sum_{n=0}^{N-1} x(n)\acute{w}_N^{kn}, \qquad k = 0,1,2,\dots,N-1 \tag{2.1.a}$$

Where

$$\acute{w}_N^{kn} \triangleq \exp\left\{\frac{-i2\pi}{N}\right\} \tag{2.1.b}$$

or

$$X_F[k] = \frac{1}{\sqrt{N}}\sum_{n=0}^{N-1} x(n)\left\{\cos\frac{2nk\pi}{N} - i\sin\frac{2nk\pi}{N}\right\}, \qquad k = 0,1,2,\dots,N-1 \tag{2.2}$$

Assuming N is a multiple of 2 to get recursive structure. We fold the input sequence around N/2 and combine the n^{th} term and $(n+N/2)^{th}$ term in eq. (2.2). It can be rewritten as

$$X_F[k] = \frac{1}{\sqrt{N}}\sum_{n=0}^{\frac{N}{2}-1} W_k(n)\left\{\cos\frac{2nk\pi}{N} - i\sin\frac{2nk\pi}{N}\right\}, \qquad k = 0,1,2,\dots,N-1 \tag{2.3.a}$$

Where

$$W_k(n) = x(n) + (-1)^k x\left(n+\frac{N}{2}\right) \tag{2.3.b}$$

It can be observed that we require only half the summation terms in eq. (2.3.a) as compared to eq. (2.2)

Replacing n by $\left(\frac{N}{2}-1-n\right)$ in eq. (2.3. a), we obtain

$$X_F\,[k] = \frac{(-1)^k}{\sqrt{N}} A_{\frac{N}{2}-1}(k) \tag{2.4.a}$$

Where

$$A_j(k) = \sum_{n=0}^{j} w_k(j-n)\{\cos((n+1)\theta_k) + i\sin((n+1)\theta_k)\} \tag{2.4.b}$$

And

$$\theta_k = \frac{2k\pi}{N}, \qquad k = 0,1,2,\dots,N/2 \tag{2.4.c}$$

On solving eq. (2.4.b) as shown in Appendix A, we get the following recursive form.

$$A_j(k) = 2\cos\theta_k\, A_{j-1}(k) + w_k(j)\,\{\cos\theta_k + i\sin\theta_k\} - w_k\,(j-1) - A_{j-2}(k), \quad k = 0,1,2,\dots,N/2 \tag{2.5}$$

Thus, $A_{\frac{N}{2}-1}(k)$ is available when switches SW1, SW2, SW3 and SW4 as shown in Fig. 2.3 are closed at n = N/2. The output coefficients are obtained as real and imaginary part through the structure.

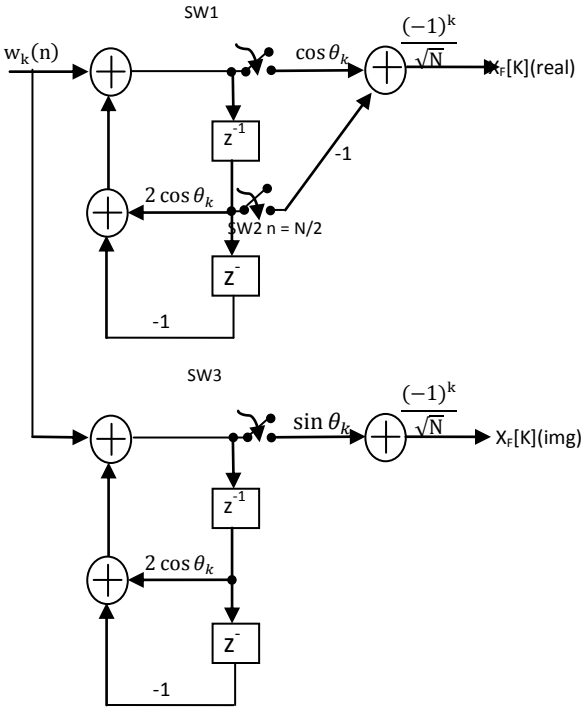

Fig 2.3 Lattice Structure implementation of DFT coefficients

For one specific value of k = k1, say, the computation of DFT coefficient X[k1] takes N/2 clock cycles. Therefore, we can calculate first N/2 + 1 coefficients of DFT using eq. (2.5) and the remaining coefficients (N/2 +1,…,N-1) can be obtained by using conjugate symmetry property of DFT [35].

2.2.4 Discrete Hartley Transform

The one dimensional DHT given by Bracewell [36] is given by

$$X_H[k] = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x(n) \operatorname{cas}\left(\frac{2\pi kn}{N}\right), \qquad k = 0,1,2 \ldots N-1 \tag{2.6.a}$$

Where

$$\operatorname{cas}(\theta) = \cos\theta + \sin\theta \tag{2.6.b}$$

or

$$X_H[k] = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x(n) \left\{\cos\frac{2nk\pi}{N} + \sin\frac{2nk\pi}{N}\right\}, \qquad k = 0,1,2 \ldots N-1 \tag{2.7}$$

Assuming N is a multiple of 2 to get recursive structure. We fold the input sequence around N/2 and combine the n^{th} term and $(n+N/2)^{th}$ term in eq. (2.7). It can be rewritten as

$$X_H[k] = \frac{1}{\sqrt{N}} \sum_{n=0}^{\frac{N}{2}-1} w_k(n) \left\{\cos\frac{2nk\pi}{N} + \sin\frac{2nk\pi}{N}\right\}, \qquad k = 0,1,2, \ldots, N-1 \tag{2.8.a}$$

Where

$$w_k(n) = x(n) + (-1)^k x\left(n + \frac{N}{2}\right) \tag{2.8.b}$$

It can be observed that we require only half the summation terms in eq. (2.8.a) as compared to eq. (2.7)

Replacing n by $\left(\frac{N}{2} - 1 - n\right)$ in eq. (2.8. a), we obtain

$$X_H[k] = \frac{(-1)^k}{\sqrt{N}} B_{\frac{N}{2}-1}(k) \tag{2.9.a}$$

Where

$$B_j(k) = \sum_{n=0}^{j} w_k(j-n)\{\cos((n+1)\theta_k) - \sin((n+1)\theta_k)\} \quad (2.9.b)$$

And

$$\theta_k = \frac{2k\pi}{N}, k = 0,1,2,\dots,N-1 \quad (2.9.c)$$

Again, from eq. (2.9.b) as shown in Appendix A, we get the following recursive form

$$B_j(k) = 2\cos\theta_k B_{j-1}(k) + w_k(j)\{\cos\theta_k - \sin\theta_k\} - w_k(j-1) - B_{j-2}(k), \quad k = 0,1,2 \dots N-1 \quad (2.10)$$

Switches SW1, SW2 and SW3 as shown in Fig. 2.4 are closed at n=N/2. Two such filter structures are used in parallel in proposed architecture to obtain two consecutive output coefficients at simultaneously. Thus, so all the coefficients for DHT are obtained after (N/2).(N/2) = $N^2/4$ clock cycles.

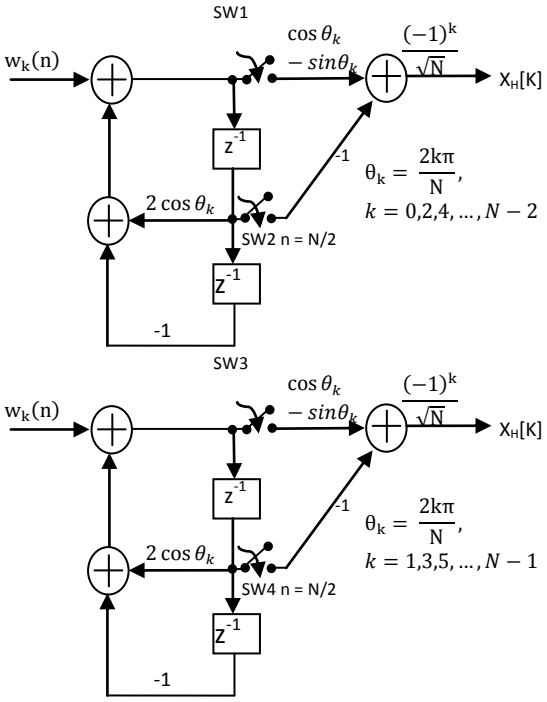

Fig 2.4 Lattice Structure implementation of DHT coefficients

2.3 Derivation of Inverse Transforms

2.3.1 Inverse Discrete Sine Transform

The recursive structure for IDST as shown in figure 2.5 has already been proposed by *Jain et. al.*[54]

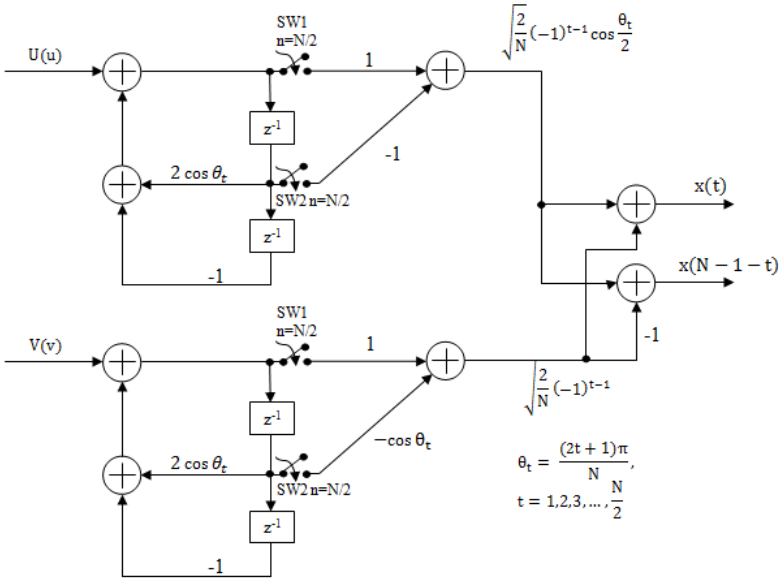

Fig 2.5 Lattice Structure implementation of IDST coefficients

2.3.2 Inverse Discrete Cosine Transform

The recursive structure for IDCT as shown in figure 2.6 has already been proposed by *Wang et. al.*[55]

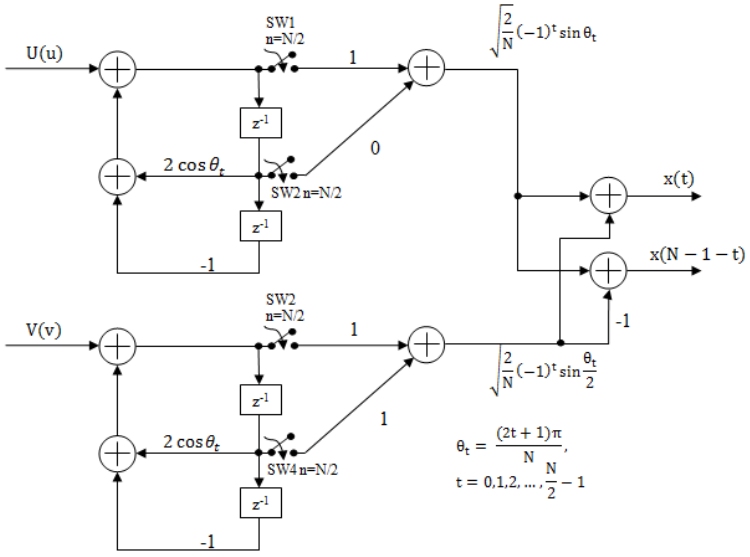

Fig 2.6 Lattice Structure implementation of IDCT coefficients

2.3.3 Inverse Discrete Fourier Transform

The formula for IDFT is given by

$$x(n) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X_F[k] \acute{w}_N^{-kn} \quad n = 0,1,2 \ldots N-1 \tag{2.11.a}$$

Where

$$\acute{w}_N^{kn} \triangleq \exp\left\{\frac{-i2\pi}{N}\right\} \tag{2.11.b}$$

Rewriting eqn. (2.11) using eqn. (2.11a)

$$x(n) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X_F[k] \left\{\cos\frac{2nk\pi}{N} + i\sin\frac{2nk\pi}{N}\right\}, \qquad n = 0,1,2 \ldots N-1 \tag{2.12}$$

Assuming, N is a multiple of 2 to get recursive structure. We fold the input sequence around N/2 and combine the k^{th} term and $(k+N/2)^{th}$ term in eq. (2.12). It can be rewritten as

$$x(n) = \frac{1}{\sqrt{N}} \sum_{k=0}^{\frac{N}{2}-1} W_n[k] \left\{\cos\frac{2nk\pi}{N} + i\sin\frac{2nk\pi}{N}\right\}, \qquad k = 0,1,2, \ldots, N-1 \tag{2.13.a}$$

where

$$W_n[k] = X_F[k] + (-1)^n X_F\left[k + \frac{N}{2}\right] \tag{2.13.b}$$

And

$$W_n[k] = W_n[k](real) + iW_n[k](imag) \tag{2.13.c}$$

$W_n[k]$ is complex. It can be observed that we require only half the summation terms in eq. (2.13.a) as compared to eq. (2.12)

Replacing k by $\left(\frac{N}{2} - 1 - k\right)$ in eq. (13. a), we obtain

$$x(n) = \frac{(-1)^n}{\sqrt{N}} \left(C_{\frac{N}{2}-1}(n) + D_{\frac{N}{2}-1}(n)\right) \tag{2.14.a}$$

Where

$$C_j(n) = \sum_{k=0}^{j} (W_n[j-k](real)) \cos((k+1)\theta_n) \tag{2.14.b}$$

And

$$D_j(n) = \sum_{k=0}^{j} (W_n[j-k](imag)) \sin((k+1)\theta_n) \quad (2.14.c)$$

And

$$\theta_n = \frac{2n\pi}{N}, n = 0,1,2, \dots, N/2 \quad (2.14.d)$$

Using symmetry property of IDFT [37], we can say that

$$x\big((N-n) \bmod N\big) = \frac{(-1)^n}{\sqrt{N}} \left(C_{\frac{N}{2}-1}(n) - D_{\frac{N}{2}-1}(n) \right) \quad (2.15)$$

On solving eq. (2.14.b), eq. (2.14.c) and eq. (2.14.d) as shown in Appendix A, we get the following recursive form.

$$C_j(n) = 2\cos\theta_n \, C_{j-1}(n) + (W_n[j](real))\cos\theta_n - (W_n[j-1](real)) - C_{j-2}(n), n = 0,1,2, \dots, N/2 \quad (2.16.a)$$

and

$$D_j(n) = 2\cos\theta_n \, D_{j-1}(n) + (W_n[j](imag))\sin\theta_n - (W_n[j-1](imag)) - D_{j-2}(n), \quad n = 0,1,2, \dots, N/2 \quad (2.16.b)$$

Switches SW1, SW2 and SW3 as shown in Fig. 2.7 are closed at n=N/2. Two output coefficients x(n) and x(N-n) are obtained after N/2 clock cycles.

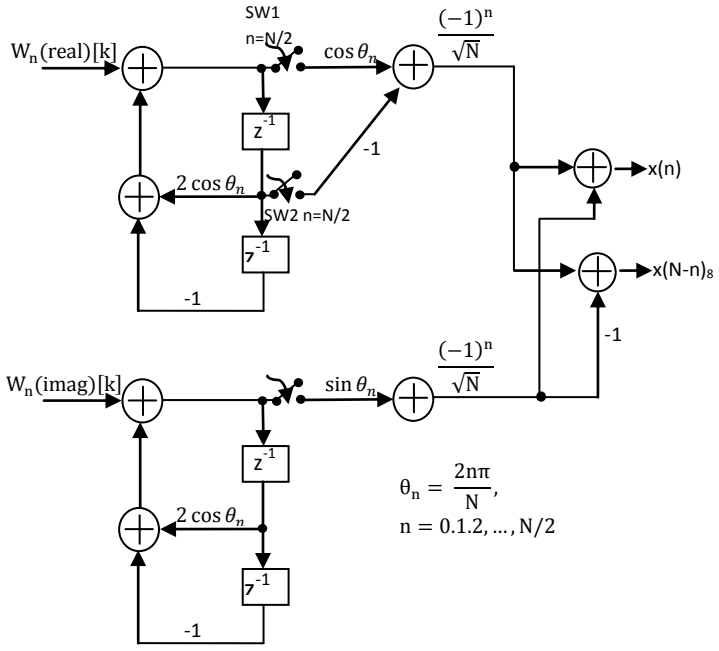

Fig 2.7 Lattice Structure implementation of IDFT coefficients

2.3.4 Inverse Discrete Hartley Transform

The one dimensional IDHT given by Bracewell [36] is given by

$$x(n) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X_H[k] \text{cas}\left(\frac{2\pi kn}{N}\right) \quad n = 0,1,2 \ldots N-1 \tag{2.17.a}$$

Where

$$\text{cas}(\theta) = \cos\theta + \sin\theta \tag{2.17.b}$$

We know DHT is self-invertible transform so the structure shown in fig. 2.8 can be used for computation of inverse transform with a modification that

$$W_n[k] = X_H[k] + (-1)^n X_H\left[k + \frac{N}{2}\right] \tag{2.18}$$

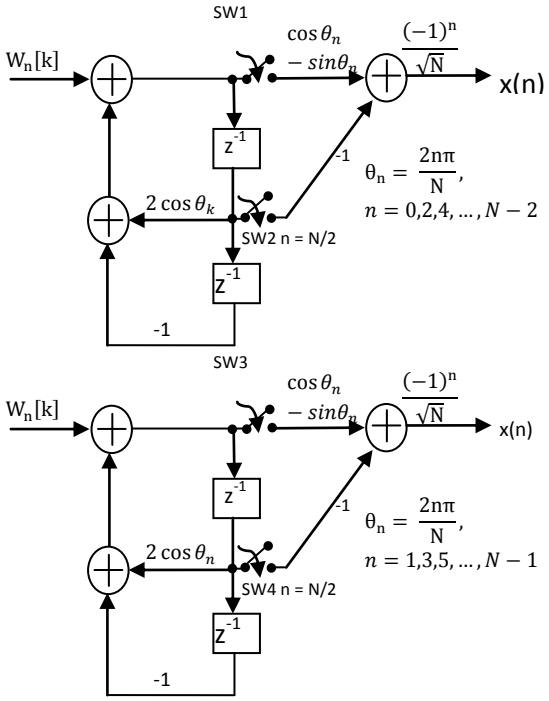

Fig 2.8 Lattice Structure implementation of IDHT coefficients

2.4 Unified Architecture of Discrete Trigonometric Transforms

Fig. 2.9 shows the proposed unified architecture to compute N point DST/DCT/DFT/DHT/IDST/IDCT/IDFT/IDHT. The input matrix, X1 and X2 are of dimension N, and consist of same data in case of DST, IDST, DCT, IDCT, DHT, IDHT and DFT since all of these transforms involve only real values. For IDFT, X1 consists of real coefficients and X2

consists of imaginary coefficients. The mode selection is done using 3 bit choice switch which selects mode as shown in table 2.1. It may be the observed that the block diagram is modular and consists of pre-processing unit, filter bank and post processing unit. Pre-processing has four sub-blocks namely, 'Mul1', 'Mul2', 'Mul3', 'angle θ_k'.

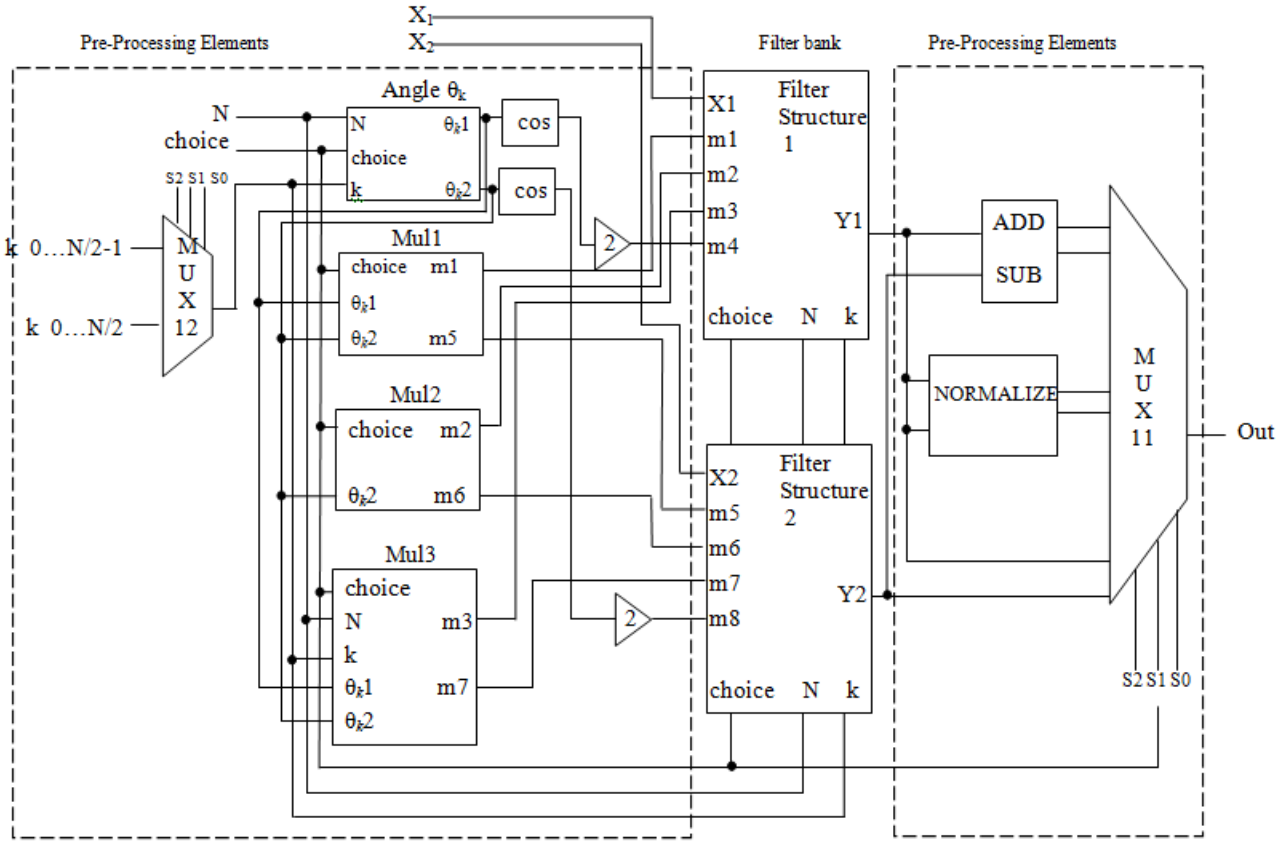

Fig. 2.9 Unified architecture of trigonometric transforms and their inverse

These sub-blocks are the logic boxes used to find the values of multipliers and angle coefficient for different modes. The filter bank is executed for input values of n, k = 0,1,2…(N/2-1) for DST, DCT, DHT, IDST, IDCT, IDHT and for input values of n, k = 0,1,2…(N/2-1) for DFT and IDFT. The output of filter bank is given as input to the post-processing unit to get output coefficients of the transforms and their inverse. The unified architecture yields two output coefficients in one clock. The post-processing and post-processing blocks have been explained in detail.

Here, Mux11 is used to select corresponding output manipulation as shown in table 2.2 and Mux12 is used to select iteration value of k. For every clock cycle and 'input k' value, the output coefficients are X[2k] and X[2k+1] for DCT, DHT, IDHT and for IDCT, the output coefficients

are x(n) and x(N-n-1) are obtained from post-processing output 1 and output 2 respectively. For DST, output coefficients X[2k+1] and X[2k+2] are obtained. For IDST, the output coefficients are x(n+1) and x(N-n) are obtained from post-processing output 1 and output 2 respectively. In the case of DFT, the filter structure1 gives the first (N/2+1) real coefficients of X [k] and second filter structure 2, gives the first (N/2+1) imaginary coefficients. The rest of the coefficients ((N/2+2), (N/2+3), (N/2+4),…,(N-1)) are found out using the symmetry property of DFT.

TABLE 2.1

MODE DESCRIPTION

Choice			Mode
S2	*S1*	*S0*	
0	0	0	DST
0	0	1	DCT
0	1	0	DFT
0	1	1	DHT
1	0	0	IDST
1	0	1	IDCT
1	1	0	IDFT
1	1	1	IDHT

TABLE 2.2

DECISION TABLE FOR MUX 11 AND MUX 12

Choice			**Mux 11**	**Mux12**
S2	*S1*	*S0*		
0	0	0	$X(2k+1) = \varepsilon_k Out1$, $X(2k+2) = \varepsilon_k Out2$	k = 0…N/2-1
0	0	1	$X(2k) = \varepsilon_k Out1$, $X(2k+1) = \varepsilon_k Out2$	k = 0…N/2-1
0	1	0	X(k)=Out1 + jOut2, X(N-1-t) = X*(t+1) for t=0…N/2-2	k = 0…N/2
0	1	1	X(2k) = Out1, X(2k+1) = Out2	k = 0…N/2-1
1	0	0	x(k+1) = Out1 + Out2, x(N-k) = Out1 - Out2	k = 0…N/2-1
1	0	1	x(k) = Out1 + Out2, x(N-1-k) = Out1 - Out2	k = 0…N/2-1
1	1	0	x(k) = Out1 + Out2, x(N-k) = Out1-Out2	k = 0…N/2
1	1	1	x(2k) = Out1, x(2k+1) = Out2	k = 0…N/2-1

In the case of IDFT, the output coefficients x(n) and x(N-n) are obtained from post-processing output 1 and output 2 respectively. The additional hardware used is 2 Mux, 2 Multiplier, 2 Adder. A lookup table can be used to compute the value of cos and sin blocks. Gain blocks of gain '2' can be modeled using right shift and left shift operations. The filter blocks consist of recursive structure shown in figures 2.1-2.8. Now, In the following section unified filter structure has been defined, which can compute all the transforms in a signal structure.

2.4.1 Recursive Filter Structure

Fig. 2.10 shows the filter block for proposed unified structure. X1 and X2 are the input arrays. A serial to parallel converter is used to obtain the input in an array form. The Mux9 and Mux10 are used to select input data for each section of parallel filter structure i.e. Wn1 and Wn2. For every

iteration of n i.e. n = 0,1,2…N/2-1, data value is selected corresponding to the mode as shown in table 2.3.

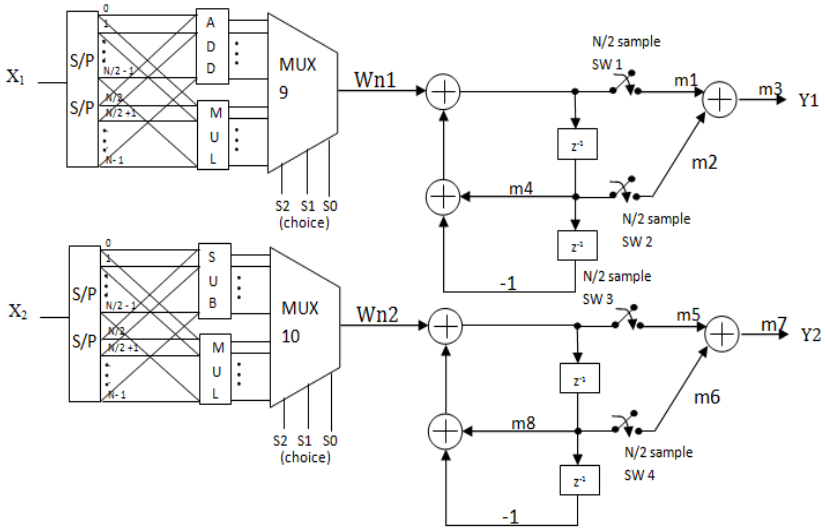

Fig. 2.10 Recursive filter structure for unified architecture

TABLE 2.3

DECISION TABLE FOR MUX 9 AND MUX 10

Choice			**Wn1**	**Wn2**
S2	*S1*	*S0*		
0	0	0	x1(n)+x1(N-n-1)	x2(n)-x2(N-n-1)
0	0	1	x1(n)+x2(N-n-1)	x2(n)-x2(N-n-1)
0	1	0	$x1(n)+(-1)^k$ x1(N/2+n)	x2(n)+$(-1)^k$x2(N/2+n)
0	1	1	x1(n)+x1(N/2+n)	x2(n)-x2(N/2+n)
1	0	0	ε_kX1(2*n)	ε_kX2(2n+1)
1	0	1	ε_kX1(2*n)	ε_kX2(2n+1)
1	1	0	X1(n)+$(-1)^k$X1(N/2+n)	X2(n)+$(-1)^k$X2(N/2+n)
1	1	1	X1(n)+X1(N/2+n)	X2(n)-X2(N/2+n)

After N/2 iterations, the switches SW1, SW2, SW3 and SW4 are closed and the final value is obtained as Y1 and Y2 after N/2 clock cycles In other words, full vector x(k) is computed in (N/2).(N/2) =$N^2/4$ clock cycles for DST, DCT, DHT, IDST IDCT, IDHT. Similarly for DFT and IDFT, N/2 computation cycles are needed for each coefficient and full vector x(k) is computed in (N/2).(N/2+1) =$N^2/4$ + N/2 clock cycles.

2.4.2 Block for calculating Angle θ_k

Fig. 2.11 shows the logic block for calculation of angle θ_k. Mux1 and Mux2 are used to select logic for calculation of angle for corresponding mode as shown in table 2.4. It may be observed that the above block uses 2 adders, 4 multipliers and 2 Mux. Gain blocks of gain '2' can be modeled using right shift and left shift operations.

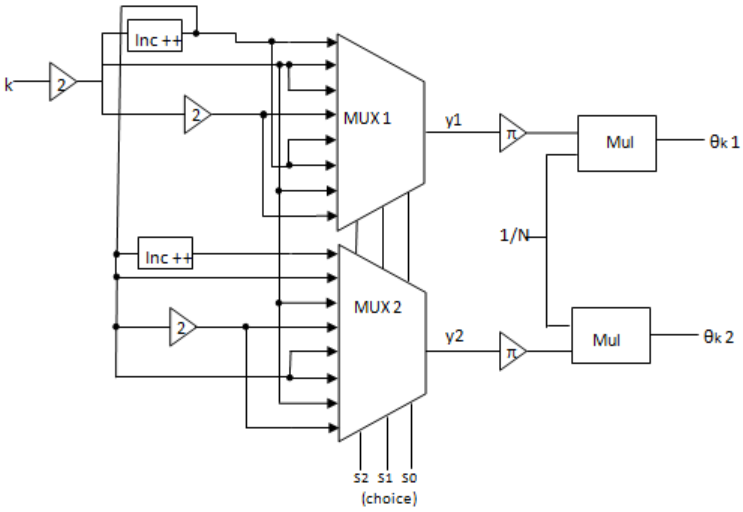

Fig. 2.11 Block for calculation of angle θ_k

TABLE 2.4

DECISION TABLE FOR MUX 1 AND MUX 2

Choice			y1	y2
S2	*S1*	*S0*		
0	0	0	2k+1	2k+2
0	0	1	2k	2k+1
0	1	0	2k	2k
0	1	1	4k	2(2k+1)
1	0	0	2k+1	2k+1
1	0	1	2k+1	2k+1
1	1	0	2k	2k
1	1	1	4k	2(2k+1)

2.4.3 Block for calculating Mul1

Fig. 2.12 shows the logic block for calculation of multiplier m1 and m5. Mux3 and Mux4 are used to select appropriate logic for calculation of multipliers for corresponding mode as shown in table 2.5. A lookup table can be used to compute the value of cos and sin blocks. The additional hardware used is 2 adder blocks.

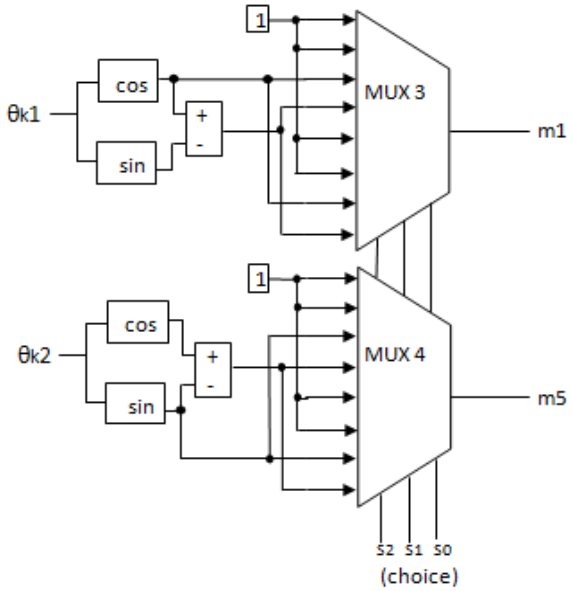

Fig. 2.12 Block for calculation of Mul1

TABLE 2.5

DECISION TABLE FOR MUX 3 AND MUX 4

Choice			m1	m5
S2	S1	S0		
0	0	0	1	1
0	0	1	1	1
0	1	0	$\cos\theta_k1$	$\sin\theta_k2$
0	1	1	$\cos\theta_k1-\sin\theta_k1$	$\cos\theta_k2-\sin\theta_k2$
1	0	0	1	1
1	0	1	1	1
1	1	0	$\cos\theta_k1$	$\sin\theta_k2$
1	1	1	$\cos\theta_k1-\sin\theta_k1$	$\cos\theta_k2-\sin\theta_k2$

2.4.4 Block for calculating Mul2

Fig. 2.13 shows the logic block for calculating multiplier m2 and m6. Mux5 and Mux6 are used to select appropriate logic for calculation of multiplier for corresponding mode as shown in table 2.6. A lookup table can be used to compute the value of cos block. No additional hardware is used.

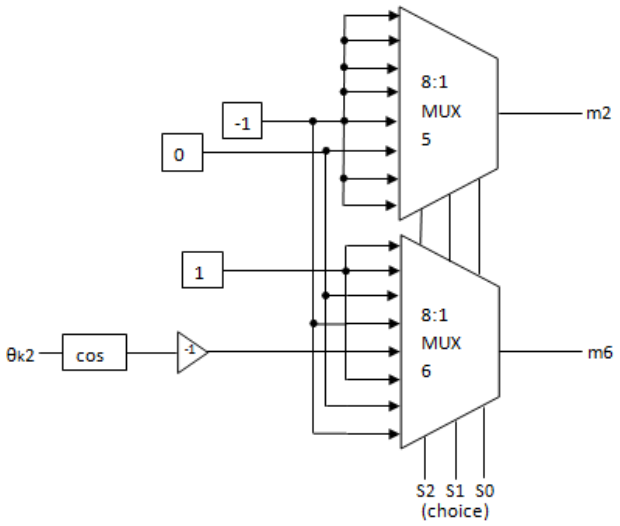

Fig. 2.13 Block for calculation of Mul2

TABLE 2.6

DECISION TABLE FOR MUX 5 AND MUX 6

Choice			m2	m6
S2	*S1*	*S0*		
0	0	0	-1	1
0	0	1	-1	1
0	1	0	-1	0
0	1	1	-1	-1
1	0	0	-1	$-\cos\theta_k 2$
1	0	1	0	1
1	1	0	-1	0
1	1	1	-1	-1

2.4.5 Block for calculating Mul3

Fig. 2.14 shows the logic block for calculating multiplier m2 and m6. Mux7 and Mux8 are used to select appropriate logic for calculation of multiplier for corresponding mode as shown in table 7. A lookup table can be used to compute the value of cos and sin blocks. The additional hardware used is 2 Mux and 6 Multipliers. Gain blocks of gain ‘2’ and ‘0.5’ can be modeled using right shift and left shift operations.

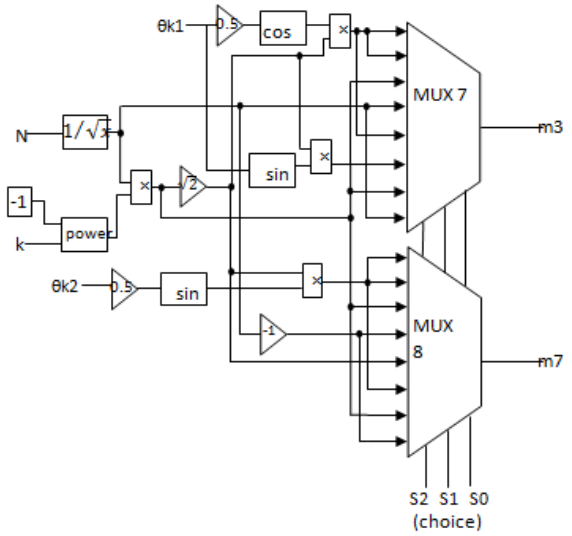

Fig. 2.14 Block for calculation of Mul3

TABLE 2.7

DECISION TABLE FOR MUX 7 AND MUX 8

Choice			m3	m7
S2	S1	S0		
0	0	0	$\sqrt{\frac{2}{N}}\epsilon_k(-1)^k\cos\frac{\theta_{k1}}{2}$	$\sqrt{\frac{2}{N}}\epsilon_k(-1)^k\sin\frac{\theta_{k2}}{2}$
0	0	1	$\sqrt{\frac{2}{N}}\epsilon_k(-1)^k\cos\frac{\theta_{k1}}{2}$	$\sqrt{\frac{2}{N}}\epsilon_k(-1)^k\sin\frac{\theta_{k2}}{2}$
0	1	0	$\frac{(-1)^k}{\sqrt{N}}$	$\frac{(-1)^k}{\sqrt{N}}$
0	1	1	$\frac{(-1)^k}{\sqrt{N}}$	$\frac{(-1)^k}{\sqrt{N}}$
1	0	0	$\sqrt{\frac{2}{N}}(-1)^k\cos\frac{\theta_{k1}}{2}$	$\sqrt{\frac{2}{N}}(-1)^k$
1	0	1	$\sqrt{\frac{2}{N}}(-1)^k\sin\theta_{k1}$	$\sqrt{\frac{2}{N}}(-1)^k\sin\frac{\theta_{k2}}{2}$
1	1	0	$\frac{(-1)^k}{\sqrt{N}}$	$\frac{(-1)^k}{\sqrt{N}}$
1	1	1	$\frac{(-1)^k}{\sqrt{N}}$	$\frac{(-1)^k}{\sqrt{N}}$

The proposed structure is simulated in Simulink and coded in the MATLAB.

2.5 Implementationof Unified Architecture in Simulink

Fig. 2.14 shows the implementation of unified architecture in MATLAB Simulink.

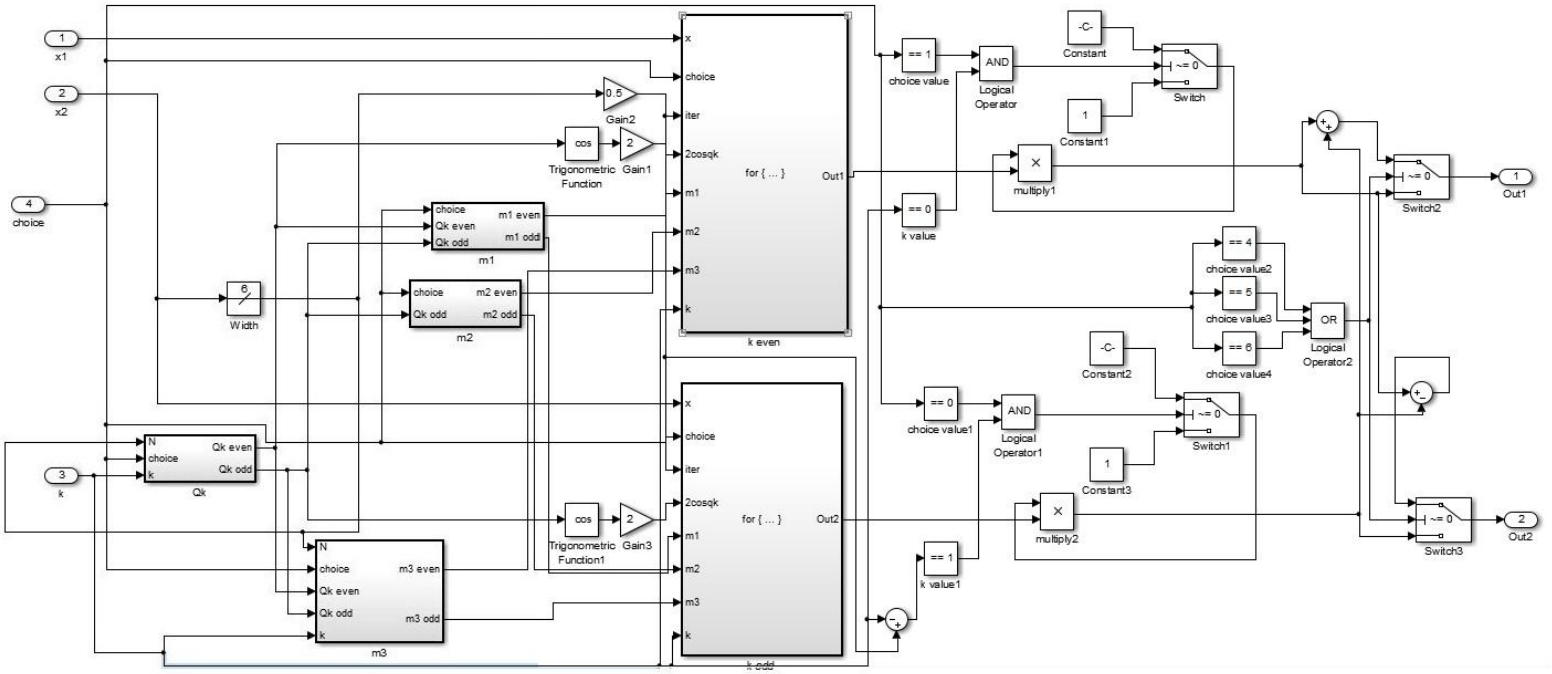

Fig. 2.15 Simulink implementation of proposed unified architecture

2.6 Performance Analysis

The proposed unified architecture has been presented with the aim of realizing the trigonometric transform (DFT/ DHT/ DST/ DCT) and their inverse. It can be seen from fig. 2.10 that after pre-processing, basic structure of all the transforms and their inverse are almost similar and requires only real multiplication per sample in the feedback path and one real multiplication in the feed-forward path(m1, m5) as m2 and m6 are unity, except for IDFT where two multiplications are involved. The constant multiplication with m1, m5, m2, m6, m3, m7 is required after N/2 clock cycles and it is multiplied once per sample. Hence, total real multiplications for DHT/IDHT and DFT/IDFT are (N/2 + 2) and (N + 4) respectively. In case of DST/ DCT/ IDCT, value of m1, m5, m2, m6 is unity hence only (N/2 + 1) multiplications are required. In case of IDST, (N/2 + 2) multiplications are required as m6 is not unity. In table 2.8, the number of multipliers and adders, throughput and latency and other aspects are compared with other existing architectures. It is seen from fig 2.10 that the structure constitutes resonance with poles at $\pm e^{j\theta_k}$, i.e. the conjugate poles are lying on the unit circle.

TABLE 2.8

COMPARISON OF DIFFERENT UNIFIED STRUCTURES

	No. of multipliers	**No. of adders**	**Latency**	**Throughput**	**Type**	**Limitation on Transform size N**	**Communication**	**I/O operation**
Proposed	10	8	N/2	1 for DFT 2 for others	DST, DCT, DFT, DHT, IDST, IDCT, IDFT, IDHT	Even	Local	PISO
Chiper *et. al.,* 2005[64]	2	2N+3	-	2	DCT,DST, IDCT, IDST	Prime Number greater than 2	-	-
Das and Banerjee, 2002[62]	-	2N(N-1)	-	-	DST, DCT, DFT, DHT	-	Local	-
Maharatna *et. al,* 2001.[56]	-	4N(N-1)	-	-	DST, DCT, DFT, DHT	-	Global	-
Hsiao, 2000[58]	$\log_2 N$	$\log_2 N+2$	2N	1 for DFT 2 for others	Radix-2 DST, DCT, DFT, DHT	Power of 2	Global	PISO
	$\log_4 N$	$3\log_4 N+2$			Radix-4 DST, DCT, DFT, DHT	Power of 4		
Fang and Wu, 1997[59]	N/2 + 4	N + 3	2N for DFT 3N for DST/DCT	1	DST, DCT, DFT, DHT	Even	-	-
Pan and Park, 1997[60]	N	2N	N	2	DCT, DST, DHT	Even	-	-
Kar and Rao, 1996[57]	0	2N	-	2	DST, DCT, DFT, DHT	-	-	-
Liu-Chiu1, 1993[55]	6N-4	5N-1	N	2	DST, DCT	No	Local	SIPO
Liu-Chiu2, 1993[55]	4N	5N-1	2N	2	DST, DCT	No	Local	SISO

TABLE 2.9

COMPARISON OF NUMBER OF MULTIPLIERS/ ADDERS

N	**8**	**16**	**32**	**64**	**128**
Proposed	10/8	10/8	10/8	10/8	10/8
Hsiao, 2000[58]	3/5	4/6	5/7	6/8	7/9
	-/-	2/8	-	3/11	-
Fang and Wu, 1997[59]	8/11	12/19	20/35	36/67	68/131
Kar and Rao, 1996[57]	0/16	0/32	0/64	0/128	0/256
Liu-Chiu1, 1993[55]	44/39	92/79	188/159	380/319	764/639
Liu-Chiu2, 1993[55]	32/39	64/79	128/159	256/319	512/639
Pan and Park, 1997[60]	8/16	16/32	32/65	64/128	128/256
Chiper *et. al.,* 2005[64]	2/19	2/35	2/67	2/131	2/259
Das and Banerjee, 2002[62]	-/112	-/480	-/1984	-/8064	-/32512
Maharatna *et. al,* 2001.[56]	-/224	-/960	-/3968	-/16128	-/65024

TABLE 2.10

COMPARISON OF NUMBER OF MULTIPLICATIONS/ ADDITIONS

Transform	**Proposed**		**Proposed N = 8**		**Prots'ko, 2014[61]**	
	Multiplication	*Addition*	*Multiplication*	*Addition*	*Multiplication*	*Addition*
DST	N/2 + 2	N	6	8	8	37
DCT	N/2 + 2	N	6	8	8	33
DHT/IDHT	N/2 + 2	N	6	8	-	-
DFT	N + 4	2N - 2	12	14	-	-
IDST	N/2 + 3	N	7	8	8	37
IDCT	N/2 + 2	N	6	8	8	37
IDFT	N + 4	2N - 1	12	15	-	-

These resonators are having resonance frequency of θ_k and sharper peak at resonances [65]. It can be seen from table 2.8 that the proposed unified architecture is efficient in terms of hardware. Table 2.10 shows the comparison of computational complexity of the proposed unified architecture with other algorithms. It can be seen that the proposed algorithm not only reduces the number of multiplication but also reduces the total number of operations. The main advantage of the proposed architecture is that it can also compute the inverse of transforms in a unified structure form which have not been computed in the existing papers. Moreover, any desired output coefficient is computed independently.

CHAPTER 3
LMS ALGORITHM

3.1 Mathematical Derivation of LMS Algorithm

According to wiener filter theory, under the assumptions of linearity, time invariance and additive white Gaussian noise, the mean square error provides optimum estimate of cost function. Most of the adaptive algorithms work on the principle of minimization of mean square error (MMSE) for adaptation of weights. The weights are adjusted iteratively to minimize the square of error function.

The LMS is the most basic adaptive algorithm. It was first proposed by Widrow and hoff [29] based on the steepest descent approach. The weights are adjusted using a step size factor and the gradient of error signal. It is the least computationally intensive algorithms but it suffers from a major disadvantage of slow convergence [70]. A large value of step size leads to faster convergence but also yields large residual errors and can make the system unstable sometimes. A small value of step size leads to low residual errors but it also slows the convergence. The optimum value of step size is dependent on the eigenvalue spread variance of incoming signal [29, 71, 72].

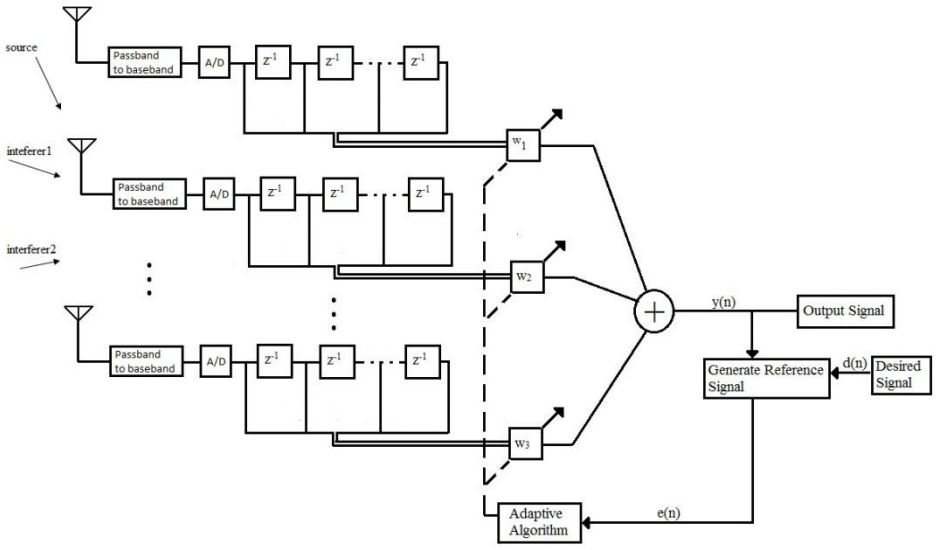

Fig. 3.1 Beamformer system model using conventional LMS algorithm

Let us consider the example shown in figure 3.1. We assume an L element array. The error function is given by[70]

$$e(n) = d(n) - y(n) \tag{3.1}$$

d(n): reference signal

y(n): beamformer output

we know that the output of beamformer is given as

$$y(n) = w^H(n)x(n) \tag{3.2}$$

x(n): L×M received signal

H: Hermitian operator(conjugate transpose)

The optimal cost function is given as

$$\xi(n) = E[e^2(n)] \tag{3.3}$$

E[]: Expectation operator

$$\xi(n) = E\left[\left(d(n) - y(n)\right)^2\right] \tag{3.4}$$

$$\xi(n) = E\left[\left(d(n) - w^H(n)x(n)\right)^2\right] \tag{3.5}$$

$$\xi(n) = E\left[\left(d(n) - w^H(n)x(n)\right).\left(d(n) - w^H(n)x(n)\right)^*\right] \tag{3.6}$$

$$\xi(n) = E\left[\left(d(n) - w^H(n)x(n)\right).\left(d^*(n) - w(n)x^H(n)\right)\right] \tag{3.7}$$

$$\xi(n) = E[d^2(n)] - r^H(n)w(n) - w^H(n)r(n) + w^H(n)R(n)w(n) \tag{3.8}$$

r(n): cross-correlation matrix of desired signal d(n) and received signal x(n) i.e. d*(n)x(n)

R(n): auto-correlation matrix of received signal x(n) i.e. $x(n)x^H(n)$

To minimize the equation 3.8, we differentiate with respect to $w^H(n)$ and then equate it to zero

$$\frac{\partial\left(\xi(n)\right)}{\partial w^H(n)} = -r(n) - r(n) + 2R(n)w_{opt}(n) = 0 \tag{3.9}$$

This gives

$$w_{opt}(n) = R^{-1}(n)r(n) \tag{3.10}$$

Thus, computation of optimal weights requires knowledge of cross-correlation matrix and auto-correlation matrix. The calculation of inverse of auto-correlation matrix is computationally intensive when the system comprises of large number of weights or when weights are complex. Therefore, instead of computing the weight vector using the above, we calculate it iteratively using training algorithms. The weights are adjusted according to the equation

$$w(n+1) = w(n) - \mu \frac{\partial(\xi(n))}{\partial w^H(n)} \tag{3.11}$$

μ: step size

Substituting equation 3.9 in equation 3.11

$$w(n+1) = w(n) - \mu\left[2R(n)w_{opt}(n) - 2r(n)\right] \tag{3.12}$$

$$w(n+1) = w(n) - 2\mu\left[x(n)x^H(n)w_{opt}(n) - d^*(n)x(n)\right] \tag{3.13}$$

$$w(n+1) = w(n) - 2\mu x(n)\left[x^H(n)w_{opt}(n) - d^*(n)\right] \tag{3.14}$$

$$w(n+1) = w(n) + 2\mu e^*(n)x(n) \tag{3.15}$$

Where

$$e(n) = d(n) - w^H(n)x(n) \tag{3.16}$$

LMS is the least computational intensive algorithm. However, it is not possible to improve the convergence rate and lower the steady state error at the same time. Thus the basic LMS algorithm requires some modifications. One such modification is variable size LMS in which a large step size is used in the beginning of the iterations so as to achieve faster convergence and then, it is gradually reduced when the error signal reduces to small value so as to minimize the steady state error signal. However, it is fairly complex to keep a track of varying step size.

Another way of improving the convergence speed and steady state error simultaneously is by using Transform Domain Least Mean Square (TDLMS). The eigen values of correlation matrix are related to power spectral density of the frequency bins of the received signal. More the excitation of PSD, faster will be the convergence [71]. The orthogonal transforms partitions the input signal into different frequency bands and normalization process equalizes the energy

content in these bands. The input signal is transformed and then normalized. We see the mathematical formulation of TDLMS in next section

3.2 Mathematical Derivation of TDLMS Algorithm

In this section, we modify the conventional LMS algorithm. The inputs are first transformed using an N×N orthogonal transform and then the transformed inputs are normalized by the square root of their estimated power. The block diagram is depicted in fig. 3.2. It can be seen that the antenna inputs are first decorrrelated using an orthogonal transform. The conventional LMS is modified to account for power normalization. In further sections, we analyze the convergence of some transform domain LMS algorithms which include DFT-LMS, DCT-LMS, DHT-LMS, DST-LMS for inputs having different values of ρ where $0< |\rho|<1$.

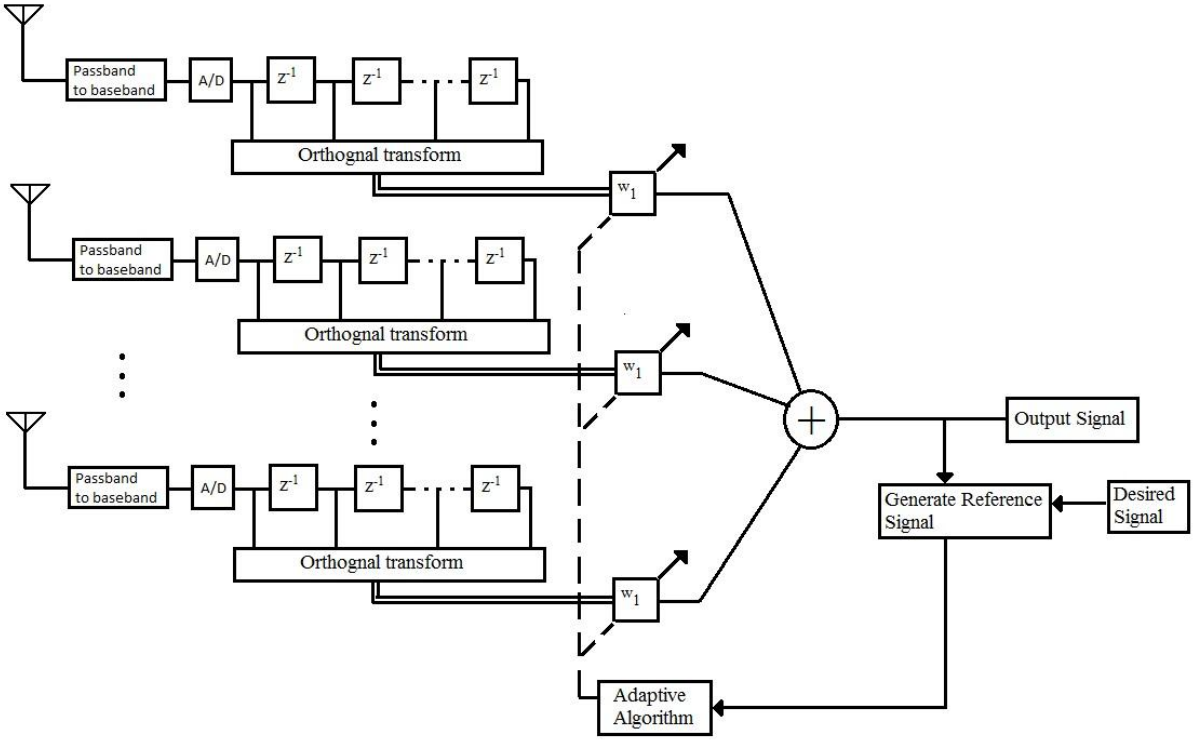

Fig. 3.2 Beamformer system model using TDLMS algorithm

The forward orthogonal transforms that have been studied are

$$\boldsymbol{T_N} = \begin{Bmatrix} \mathrm{X_s}\,[\mathrm{k}] = \sqrt{\frac{2}{\mathrm{N}}}\epsilon_{\mathrm{k}} \sum_{\mathrm{n}=1}^{\mathrm{N}} \mathrm{x(n)} \sin\frac{(2\mathrm{n}-1)\mathrm{k}\pi}{2\mathrm{N}} & \mathrm{k} = 1,2,3 \ldots \mathrm{N} \\ \mathrm{X_C}[\mathrm{k}] = \sqrt{\frac{2}{\mathrm{N}}}\epsilon_{\mathrm{k}} \sum_{\mathrm{n}=0}^{\mathrm{N}-1} \mathrm{x(n)} \cos\frac{(2\mathrm{n}+1)\mathrm{k}\pi}{2\mathrm{N}} & \mathrm{k} = 0,1,2 \ldots \mathrm{N}-1 \\ \mathrm{X_F}\,[\mathrm{k}] = \frac{1}{\sqrt{\mathrm{N}}} \sum_{\mathrm{n}=0}^{\mathrm{N}-1} \mathrm{x(n)} \left\{\cos\frac{2\mathrm{nk}\pi}{\mathrm{N}} - \mathrm{i}\sin\frac{2\mathrm{nk}\pi}{\mathrm{N}}\right\} & \mathrm{k} = 0,1,2 \ldots \mathrm{N}-1 \\ \mathrm{X_H}\,[\mathrm{k}] = \frac{1}{\sqrt{\mathrm{N}}} \sum_{\mathrm{n}=0}^{\mathrm{N}-1} \mathrm{x(n)} \left\{\cos\frac{2\mathrm{nk}\pi}{\mathrm{N}} + \sin\frac{2\mathrm{nk}\pi}{\mathrm{N}}\right\} & \mathrm{k} = 0,1,2 \ldots \mathrm{N}-1 \end{Bmatrix} \tag{3.17}$$

$$\boldsymbol{T_N^{-1}} = \begin{Bmatrix} \mathrm{x(n)} = \sqrt{\frac{2}{\mathrm{N}}} \sum_{\mathrm{k}=1}^{\mathrm{N}} \epsilon_{\mathrm{k}} \mathrm{X_S[k]} \sin\frac{(2\mathrm{n}-1)\mathrm{k}\pi}{2\mathrm{N}} & \mathrm{n} = 1,2,3 \ldots \mathrm{N} \\ x(\mathrm{n}) = \sqrt{\frac{2}{\mathrm{N}}} \sum_{\mathrm{k}=0}^{\mathrm{N}-1} \epsilon_{\mathrm{k}} \mathrm{X_C[k]} \cos\frac{(2\mathrm{n}+1)\mathrm{k}\pi}{2\mathrm{N}} & \mathrm{n} = 0,1,2 \ldots \mathrm{N}-1 \\ \mathrm{x(n)} = \frac{1}{\sqrt{\mathrm{N}}} \sum_{\mathrm{k}=0}^{\mathrm{N}-1} \mathrm{X_F}\,[\mathrm{k}] \left\{\cos\frac{2\mathrm{nk}\pi}{\mathrm{N}} + \mathrm{i}\sin\frac{2\mathrm{nk}\pi}{\mathrm{N}}\right\} & \mathrm{n} = 0,1,2 \ldots \mathrm{N}-1 \\ \mathrm{x(n)} = \frac{1}{\mathrm{N}} \sum_{\mathrm{k}=0}^{\mathrm{N}-1} \mathrm{X_H[k]} \left\{\cos\frac{2\mathrm{nk}\pi}{\mathrm{N}} + \sin\frac{2\mathrm{nk}\pi}{\mathrm{N}}\right\} & \mathrm{n} = 0,1,2 \ldots \mathrm{N}-1 \end{Bmatrix} \tag{3.18}$$

The transformation T_N is applied to every column of input matrix. It results in output vector u_n.

$$u(n) = \boldsymbol{T_N}[x(n)] \tag{3.19}$$

The transformed input is then normalized. The power normalization matrix can be obtained using the following equation

$$P_N = diag[\sigma^2(i), i = 0,1,2, \ldots N-1] \tag{3.20}$$

Here, $\sigma^2(i)$ is the individual estimated power of i^{th} element for each antenna's transformed input. It is obtained using the following equation.

$$\sigma^2(i) = \gamma\sigma^2(i) + (1-\gamma)|u(i)|^2 \tag{3.21}$$

γ: smoothening factor used to obtain power matrix.

The modified LMS equation is given as [26, 73]

$$w_T(n+1) = w_T(n) + 2\mu e^*(n) P_N^{-1} u(n) \tag{3.22}$$

Where

$$e(n) = d(n) - {w_T}^H(n) u(n) \tag{3.23}$$

w_T are the transformed weights

Modifying the above equation so that it can be obtained in the form of conventional LMS algorithm, we do the following assumption

$$\acute{w}_T(n) = P_N^{1/2} w_T(n) \tag{3.24}$$

$$z(n) = P_N^{-1/2} u(n) \tag{3.25}$$

The final LMS equation thus obtained is

$$\acute{w}_T(n+1) = \acute{w}_T(n) + 2\mu e^*(n) z(n) \tag{3.26}$$

Where

$$e(n) = d(n) - {\acute{w}_T}^H(n) z(n) \tag{3.27}$$

The solution of the above equation converges to $\acute{w}_T(n)$ which is given by

$$\acute{w}_T(n) = P_N^{1/2} \boldsymbol{T_N} [w(n)] \tag{3.28}$$

Thus, the weiner solution in time domain is obtained as

$$w(n) = \boldsymbol{T_N^{-1}} P_N^{-1/2} [\acute{w}_T(n)] \tag{3.29}$$

In the next chapter, we simulate the block diagrams shown in figure 2.3, figure 3.1, 3.2 and compare the results with previous work.

CHAPTER 4

RESULTS AND ANALYSIS

4.1 Overview

In this chapter, the implementation and analysis of the block diagrams shown in figures 2.9, 3.1, and 3.2 has been discussed. Since LabVIEW provides faster execution results and more versatility in other features, we shift our work to LabVIEW. Firstly, we implement and simulate the unified architecture of discrete trigonometric transforms in LabVIEW. After that, the conventional beamformer has been implemented and analyzed in both MATLAB and LabVIEW to verify the results from both the tools. Lastly, the implementation and analysis of conventional beamformer using TDLMS algorithm has been discussed in detail for different inputs and other parameters.

4.2 Implementation of Unified Architecture for Discrete Transforms in LabVIEW

Figure 5.2 shows the LabVIEW implementation of unified architecture already discussed in chapter 2 figure 2.9. The other blocks have been shown subsequently.

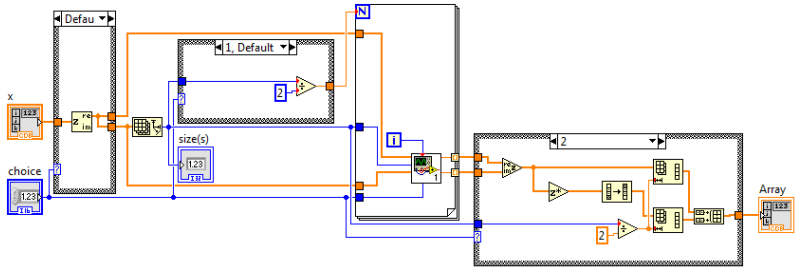

Fig. 4.1 LabVIEW implementation of unified architecture for discrete transforms

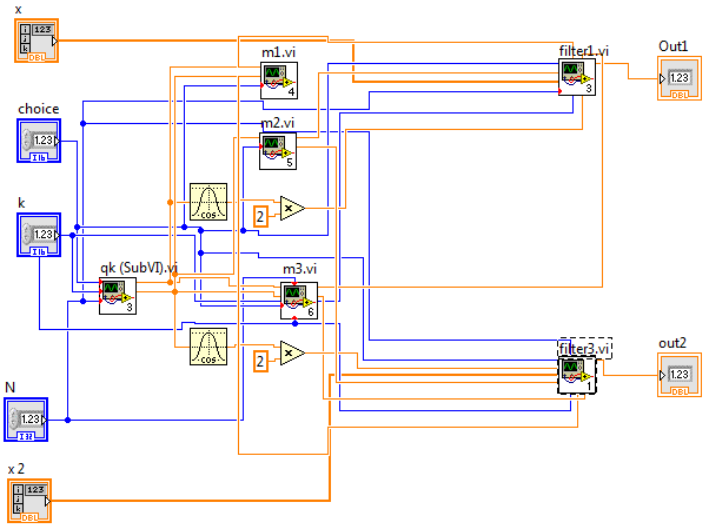

Fig. 4.2 Main sub block

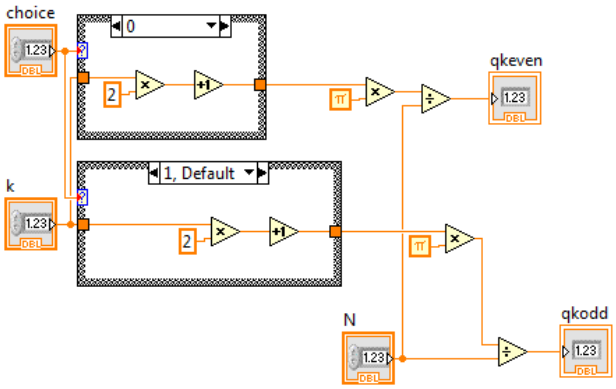

Fig. 4.3 Q_k sub block

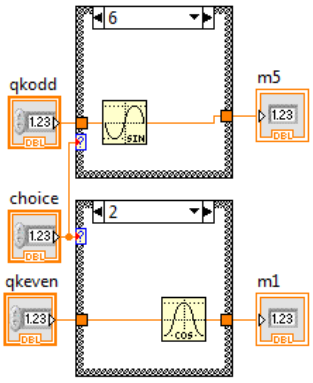

Fig. 4.4 Mul1 sub block

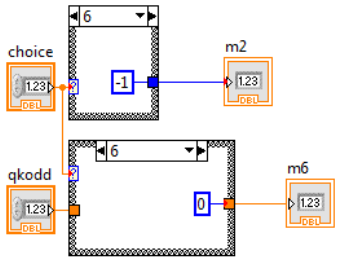

Fig. 4.5 Mul2 sub block

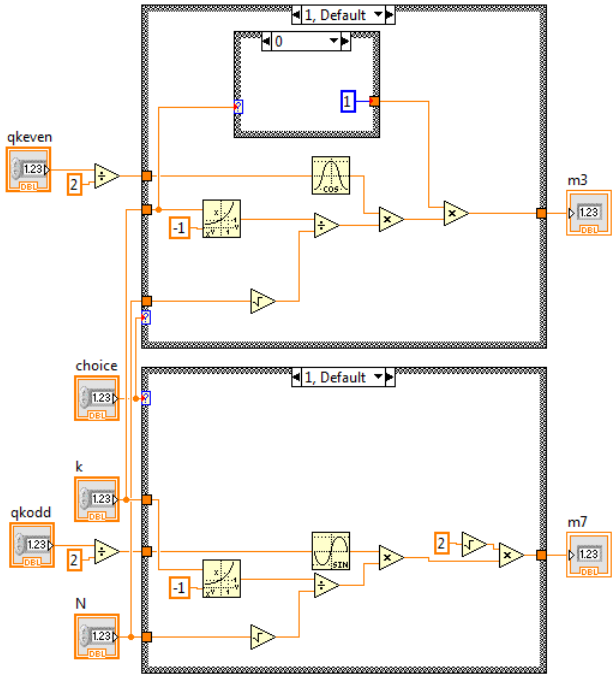

Fig. 4.6 Mul3 sub block

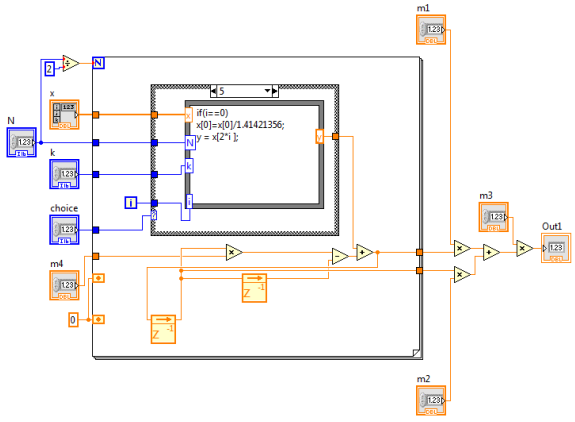

Fig. 4.7 Filter1 sub block

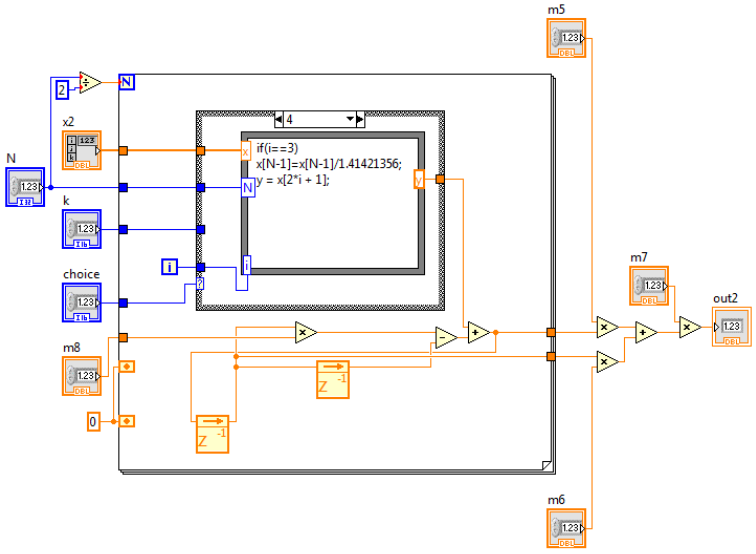

Fig. 4.8 Filter2 sub block

The simulation results have been verified with the results obtained in chapter 2 and have been shown in subsequent figures.

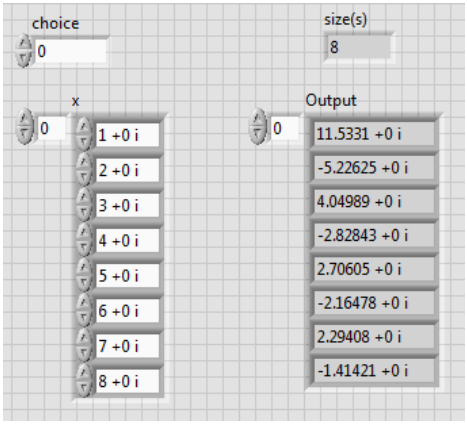

Fig. 4.9 Simulation output for DST

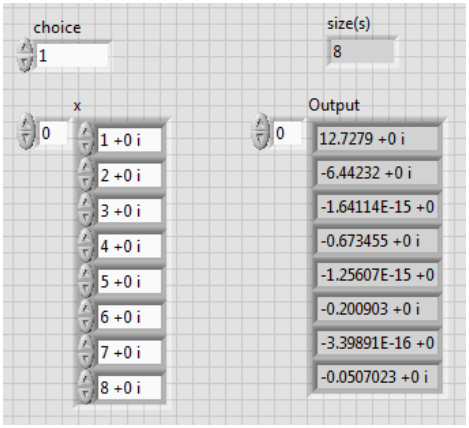

Fig. 4.10 Simulation output for DCT

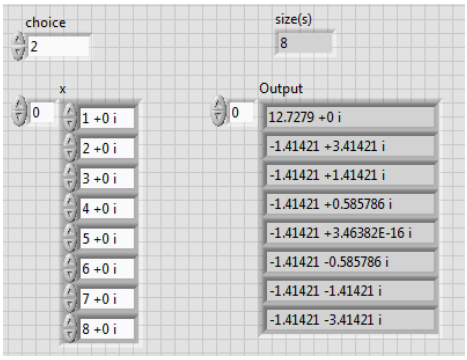

Fig. 4.11 Simulation output for DFT

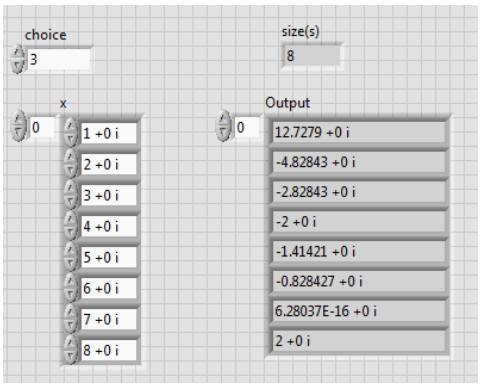

Fig. 4.12 Simulation output for DHT

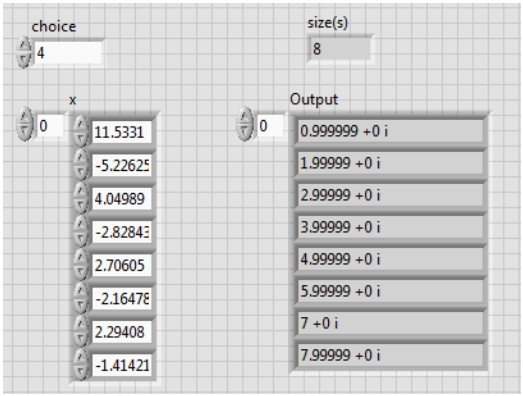

Fig. 4.13 Simulation output for IDST

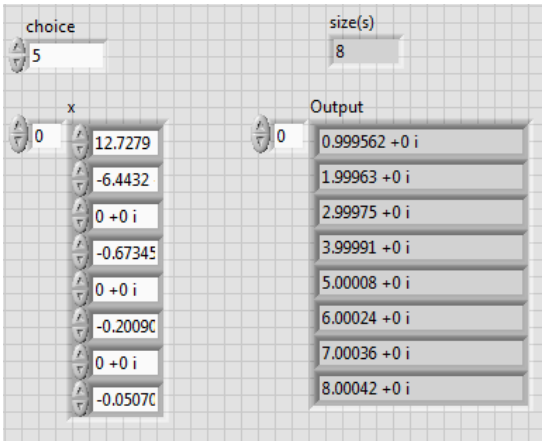

Fig. 4.14 Simulation output for IDCT

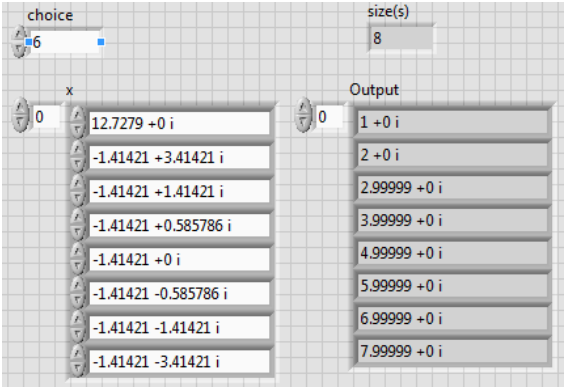

Fig. 4.15 Simulation output for IDFT

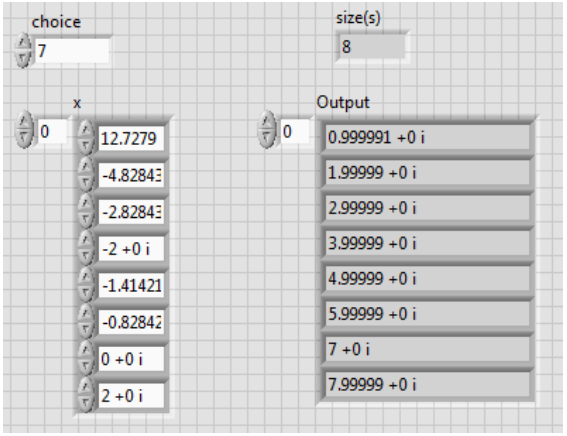

Fig. 4.16 Simulation output for IDHT

4.3 Implementation of Conventional and Adaptive TDLMS Beamformer in MATLAB

In this section, we examine the simulation results through modeling the block diagram shown in figure 3.1 and 3.2 in MATLAB. Sampling frequency is set as 3300Hz and distance between each antenna is set such that the corresponding phase difference is of pi radians. The input signals 'signal1' and 'singal2' are assumed to be sinusoidal and a white Gaussian noise is added to the system such that the resulting SNR is 20 dB. The conventional LMS algorithm stated in equation 3.15 and equation 3.16 have been implemented. The step-size chosen was 0.004. Also, different adaptive TDLMS algorithm has been implemented to compare the simulation results. The value of gamma lies in the range of 0 to 0.1 and is assumed to be 0.04. The unified architecture for discrete transform is used to decorrelate the input data. The resulting array factor for conventional LMS and adaptive TDLMS algorithm has been plotted as shown in figure 4.17 for two different values of correlation coefficient between input signals with the application of each transform discussed in chapter 2. Also, the learning curve for conventional LMS algorithm and adaptive TDLMS has been plotted as shown in figure 4.18.

4.3.1 MATLAB Code

```
clc;
clear all;
close all;


f=3300;
v=330;
lamda=v/f;
d=lamda/2;
phi=d*2*pi*f/v;
N=10;
n_b=100 ; % number of samples
thetad1=80 ; %AoA
thetad2=155 ; %AoA
theta_d1=thetad1*pi/180;
theta_d2=thetad2*pi/180;
sv1=exp(-1i*phi*cos(theta_d1)* [0:N-1]).' ;  %steering vector for the desired
signal
sv2=exp(-1i*phi*cos(theta_d2)*[0:N-1]).' ;
%svn=exp(1i*phi*cos(70*pi/180)*[0:N-1])' ;
signal1 = cos(1200*[0:n_b-1]);
signal2 = 0.2*cos(1200*[0:n_b-1])+0.8*sin(1200*[0:n_b-1]);
x1 = sv1*signal1;
```

```
x2 = sv2*signal2;
signal = awgn(x1 + x2,20);


%---------------------------LMS 1-----------------------------------------
mu = 0.004;
e1 = zeros(N,n_b);
w1 = zeros(N,n_b);
d = cos(1200*[0:n_b-1]);
dd = ones(N,1)*d;
for k = 1:n_b
    e1(:,k) = dd(:,k)-(w1(:,k).')*signal(:,k);
    w1(:,k+1)=w1(:,k)+2*mu*e1(:,k).*conj(signal(:,k));
end
%-------------------------------------------------------------------------

%signal_dct = dct2(signal);
corr_coeff = corr(signal1.', signal2.')
choice = 3;
signal_dct = zeros(N,n_b);
for i = 1:n_b
signal_dct(:,i) = my_code(signal(:,i),choice);
end

%---------------------------LMS 2-----------------------------------------
mu = 0.004;
e2 = zeros(N,n_b);
w2 = zeros(N,n_b);
power = zeros(N,1);
beta = 0.05;
d=cos(1200*[0:n_b-1]);
powersqrt = zeros(1,N);
dd = ones(N,1)*d;
for k = 1:n_b
    power = beta*(signal_dct(:,k).*conj(signal_dct(:,k))) + (1-beta)*power;
    for i=1:N
        powersqrt(i) = sqrt(power(i));
    end
    e2(:,k) = dd(:,k)-(w2(:,k).')*(signal_dct(:,k)./transpose(powersqrt));

w2(:,k+1)=w2(:,k)+2*mu*e2(:,k).*(conj(signal_dct(:,k))./transpose(powersqrt))
;
end
%-------------------------------------------------------------------------

w11 = w1(:,n_b+1);
w2(:,n_b+1) = w2(:,n_b+1).*transpose(powersqrt);
w3 = my_code(w2(:,n_b+1),(choice+4));
theta = pi/n_b:pi/n_b:pi;


AF1 = zeros(1,length(theta));
e11 = zeros(1,n_b);
for i = 1:N
    AF1 = AF1 + w11(i)'.*exp(1j*(i-1)*phi*cos(theta));
```

```
    e11 = e11 + e1(i,:);
end
e11 = e11/N;

AF2 = zeros(1,length(theta));
e22 = zeros(1,n_b);
for i = 1:N
    AF2 = AF2 + w3(i)'.*exp(1j*(i-1)*phi*cos(theta));
    e22 = e22 + e2(i,:);
end
e22 = e22/N;

hold;
plot(theta*180/pi,abs(AF1),'-r');
plot(theta*180/pi,abs(AF2),'--b');
xlabel('AOA (deg)');
ylabel('|AF_n|');
figure;
hold on;
plot(abs(e11), '-r');
plot(abs(e22), '--b');
xlabel('iterations');
ylabel('|Error_n|');
hold off;
```

4.3.2 Performance Analysis of Conventional Beamformer vs Adaptive TDLMS Beamformer

4.3.2.1 Array Factor

Array factor can be described as the gain of the antenna plotted against a scan angle which is Angle of Arrival (AOA) in our case. It can also be termed as the radiation pattern. For a ULA, the radiation pattern obtained is 1-D. Figure 4.17 illustrates the array factor for conventional beamformer and adaptive TDLMS beamformer for two different values of correlation coefficient ρ. The desired signal is set to 'signal 1', hence the maximum gain should be at the direction of 'signal 1', i.e. 80 degrees. We first discuss the results obtained for low auto-correlation value of input signal i.e. ρ 0.2302.

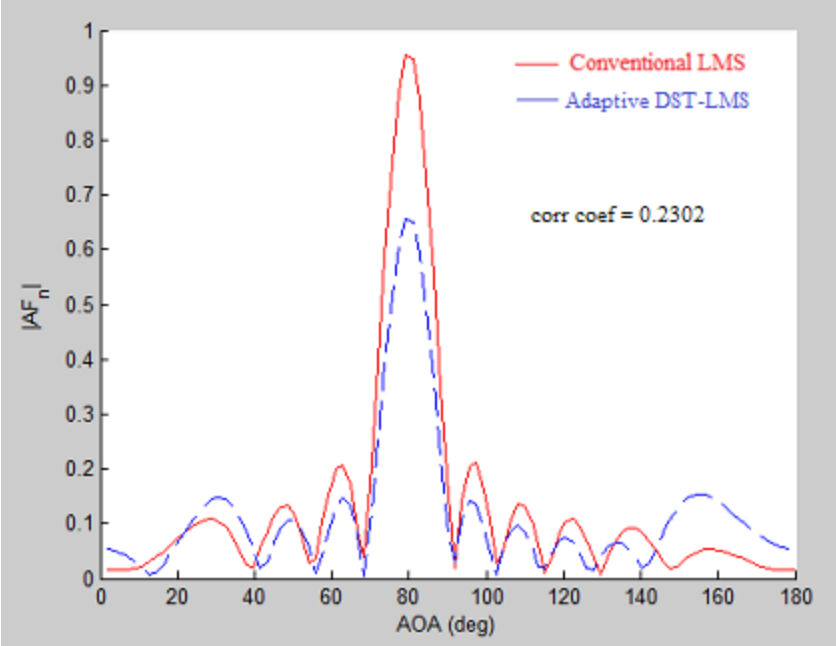

Fig 4.17.a Array Factor for DST-LMS for ρ 0.2302

From figure 4.17.a, we observe that for both conventional and DST-LMS algorithm, the maximum gain is at the right angle. However, the gain obtained using weights computed by conventional LMS is more as compared to the gain obtained using weights computed by DST-LMS. The side-lobe level is lower for DST-LMS.

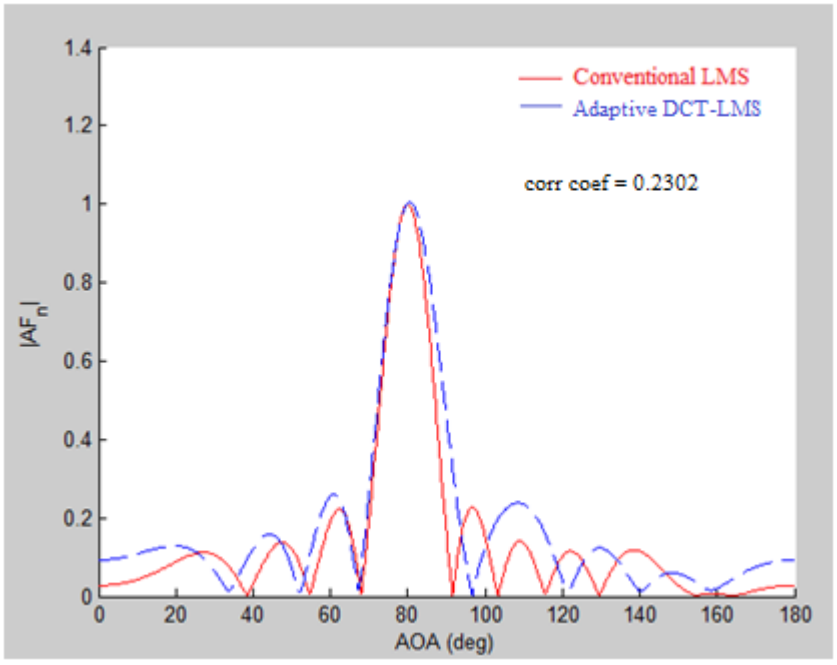

Fig 4.17.b Array Factor for DCT-LMS for ρ 0.2302

From figure 4.17.b, we observe that for both conventional and DCT-LMS algorithm, the maximum gain is at the right angle. Also, the gain provided by both the algorithms is the same. However, the side-lobe level is more in the case of DCT-LMS.

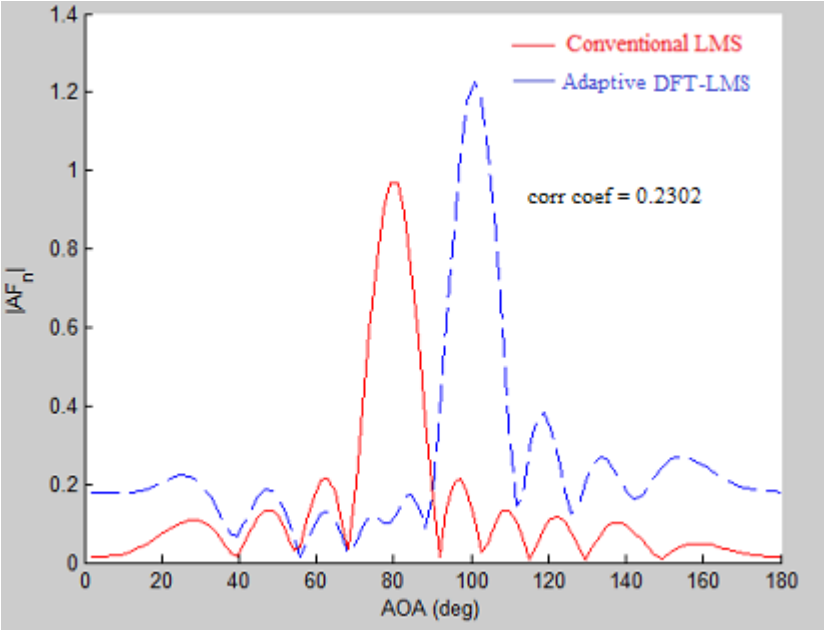

Fig 4.17.c Array Factor for DFT-LMS for ρ 0.2302

From figure 4.17.c, we observe that for DFT-LMS algorithm, the maximum gain is at an angle that is equal to 180-desired arrival angle. This can be adjusted using additional processing block. The gain obtained using DFT-LMS is more as compared to conventional LMS. However, the side-lobe level is much more as compared to conventional LMS.

From figure 4.17.d, we observe that for both conventional and DHT-LMS algorithm, the maximum gain is at the right angle. Also, the gain provided by both the algorithms is the same. However, the side-lobe level is more in the case of DHT-LMS.

We now discuss the results obtained for high auto-correlation value of input signal i.e. ρ 0.9693.

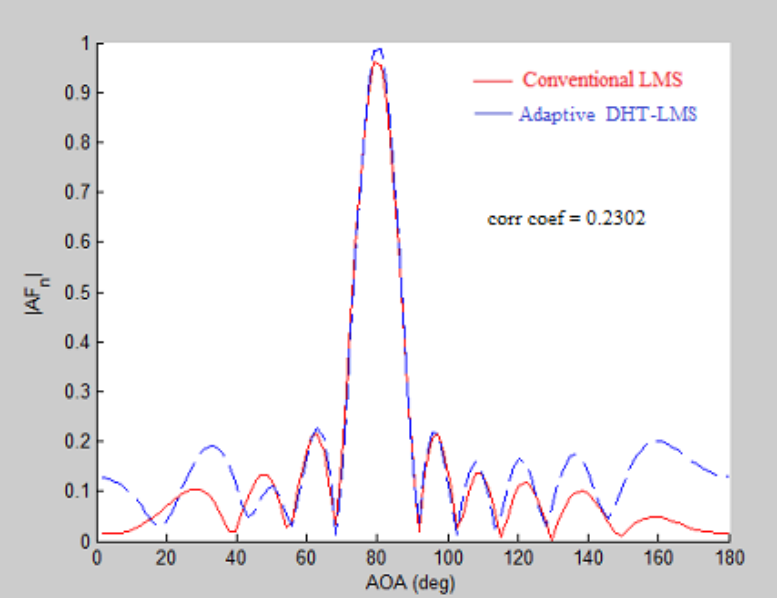

Fig 4.17.d Array Factor for DST-LMS for ρ 0.2302

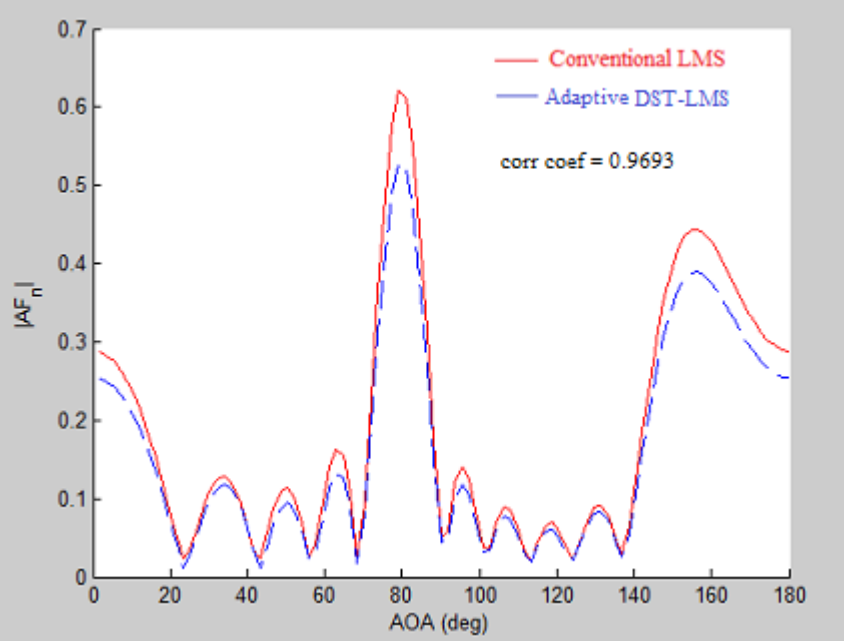

Fig 4.17.e Array Factor for DST-LMS for ρ 0.9693

Similar to the results obtained from figure 4.17.a, we observe from figure 4.17.e that for both conventional and DST-LMS algorithm, the maximum gain is at the right angle. However, the gain obtained using weights computed by conventional LMS is more as compared to the gain obtained using weights computed by DST-LMS. The side-lobe level is lower for DST-LMS.

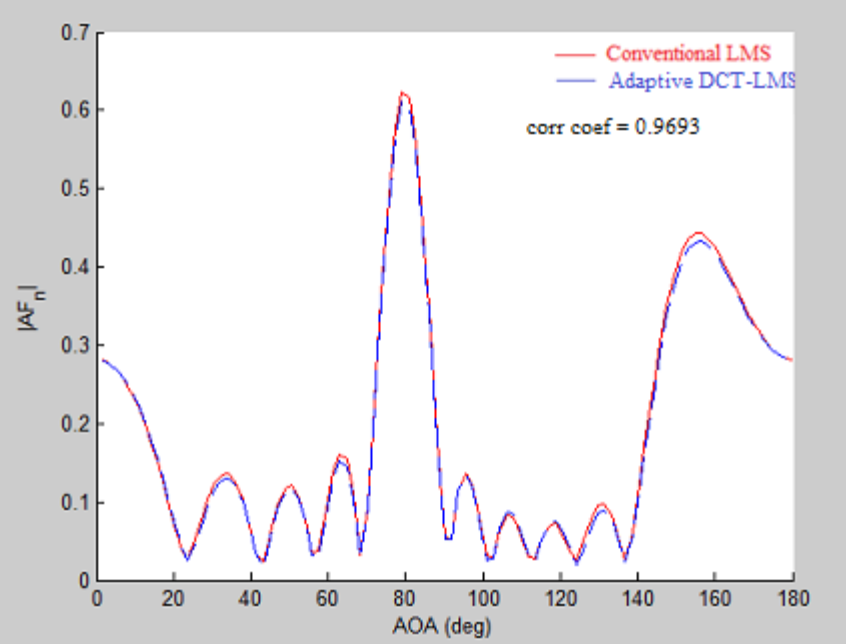

Fig 4.17.f Array Factor for DCT-LMS for ρ 0.9693

From figure 4.17.f, we observe that for both conventional and DST-LMS algorithm, the maximum gain is at the right angle. The overall gain is the same for both the algorithms.

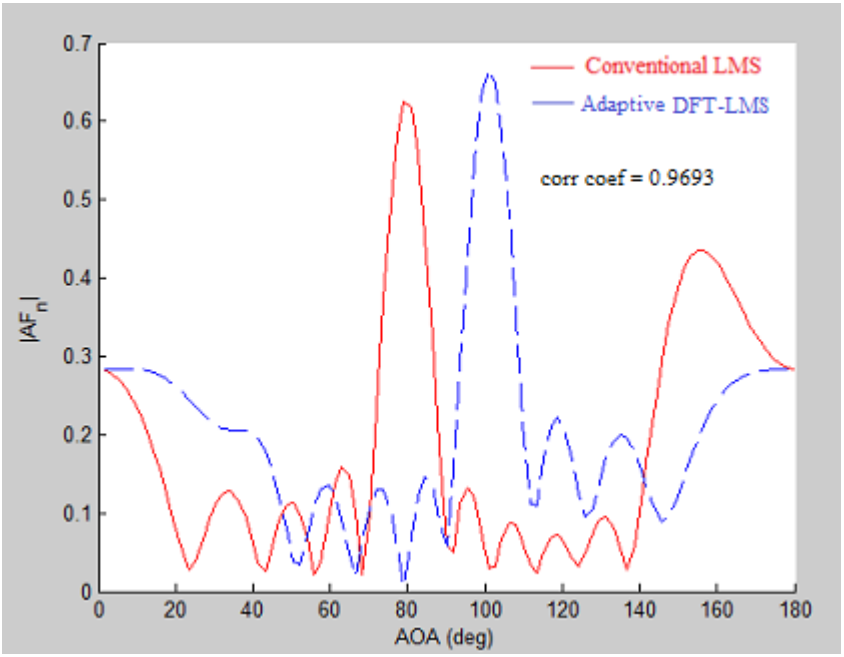

Fig 4.17.g Array Factor for DFT-LMS for ρ 0.9693

Similar to the results obtained from figure 4.17.c, we observe from figure 4.17.g that for DFT-LMS algorithm, the maximum gain is at an angle that is equal to 180-desired arrival angle. This can be adjusted using additional processing block. The gain obtained using DFT-LMS is more as compared to conventional LMS. However, the side-lobe level is much more as compared to conventional LMS.

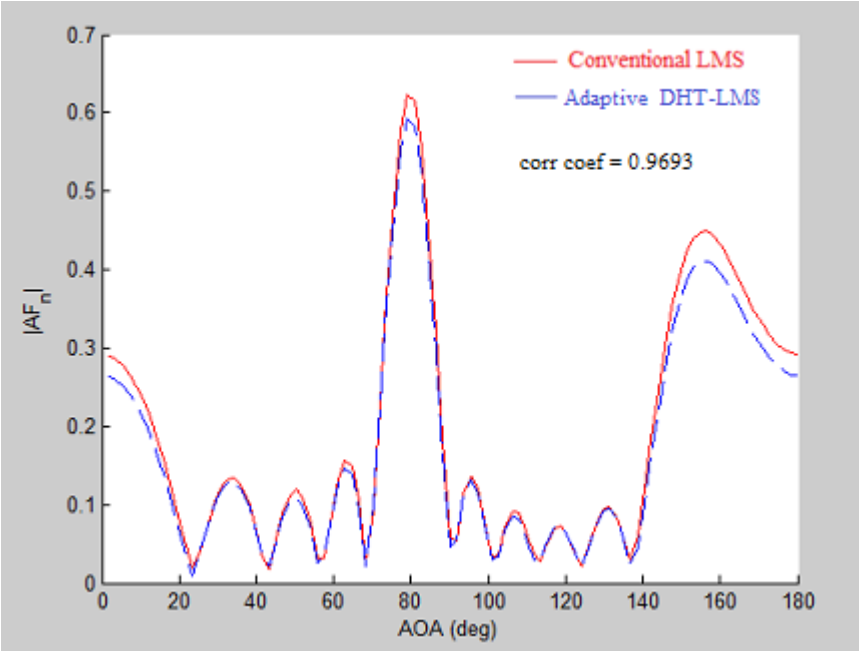

Fig 4.17.h Array Factor for DHT-LMS for ρ 0.9693

From figure 4.17.f, we observe that for both conventional and DST-LMS algorithm, the maximum gain is at the right angle. The overall gain is the same for both the algorithms.

4.3.2.2 Learning Rate

Learning rate can be described as how fast the error signal converges to zero value, i.e. how fast the output of beamformer converges to the desired output. Figure 4.18 illustrates the learning curve for conventional beamformer and adaptive TDLMS beamformer for two different values of correlation coefficient ρ. We first discuss the results obtained for low auto-correlation value of input signal i.e. ρ 0.2302.

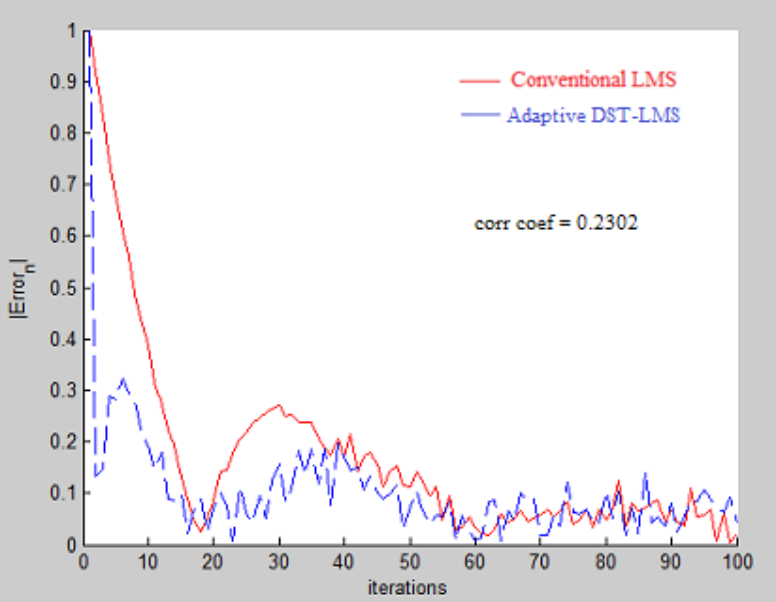

Fig 4.18.a Learning curve for DST-LMS for ρ 0.2302

From figure 4.18.a, we observe that the learning curve for DST-LMS converges faster than that of conventional LMS. Also, the steady state error for both the algorithms is similar.

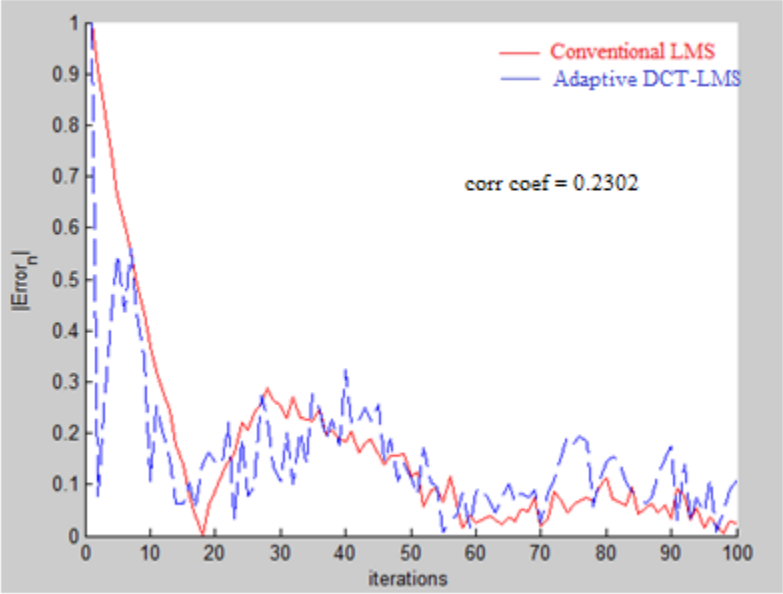

Fig 4.18.b Learning curve for DCT-LMS for ρ 0.2302

From figure 4.18.b, we observe that the learning curve for DCT-LMS converges similar to that of conventional LMS. However, the steady state error is more in case of DCT-LMS.

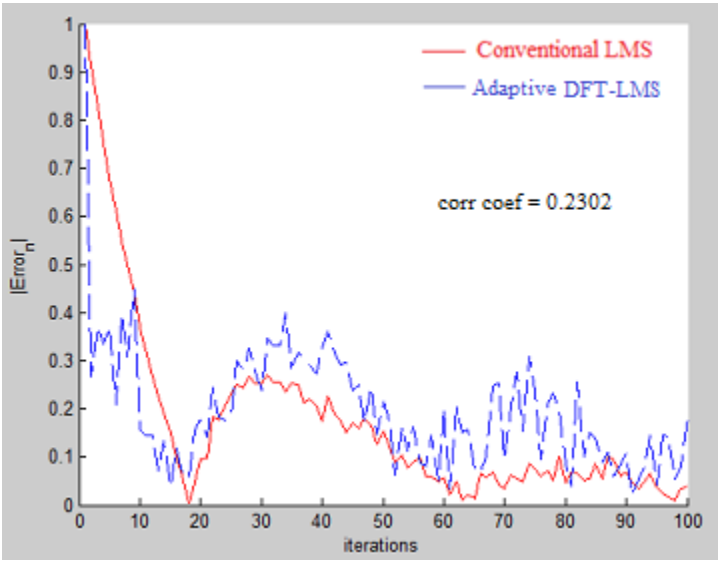

Fig 4.18.c Learning curve for DFT-LMS for ρ 0.2302

From figure 4.18.c, we observe that the learning curve for DST-LMS converges faster than that of conventional LMS. However, the steady state error is much more for DFT-LMS algorithm.

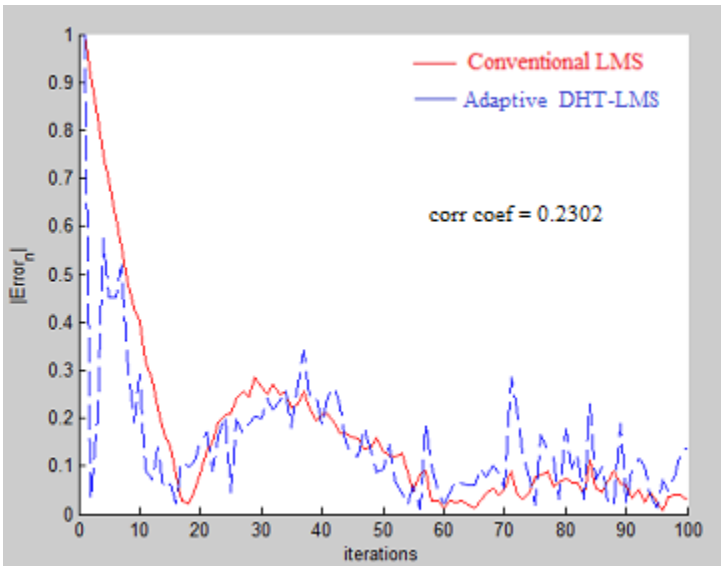

Fig 4.18.d Learning curve for DHT-LMS for ρ 0.2302

From figure 4.18.d, we observe that the learning curve for DHT-LMS is similar to that obtained using DCT-LMS shown in figure 4.18.b. The steady state error is similar or both conventional-LMS and DHT-LMS.

We now discuss the results obtained for low auto-correlation value of input signal i.e. ρ 0.2302.

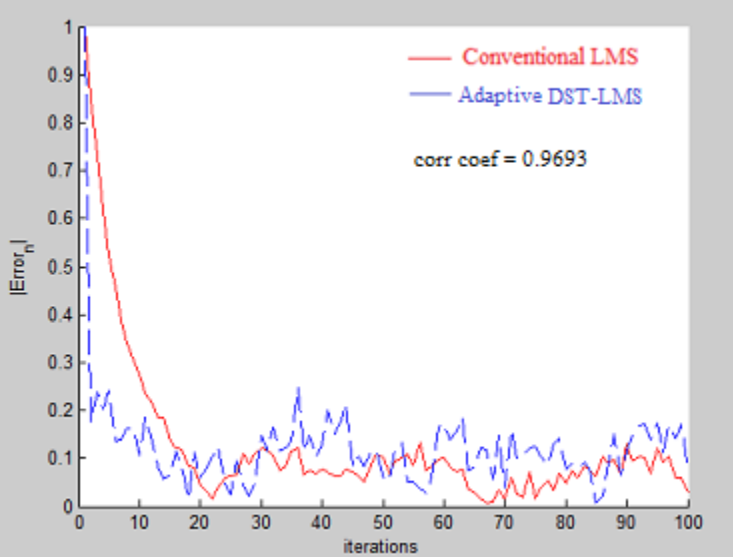

Fig 4.18.e Learning curve for DST-LMS for ρ 0.9693

From figure 4.18.e, we observe that the learning curve for DST-LMS converges similar to that of conventional LMS. However, the steady state error for DST-LMS is more as compared to conventional LMS.

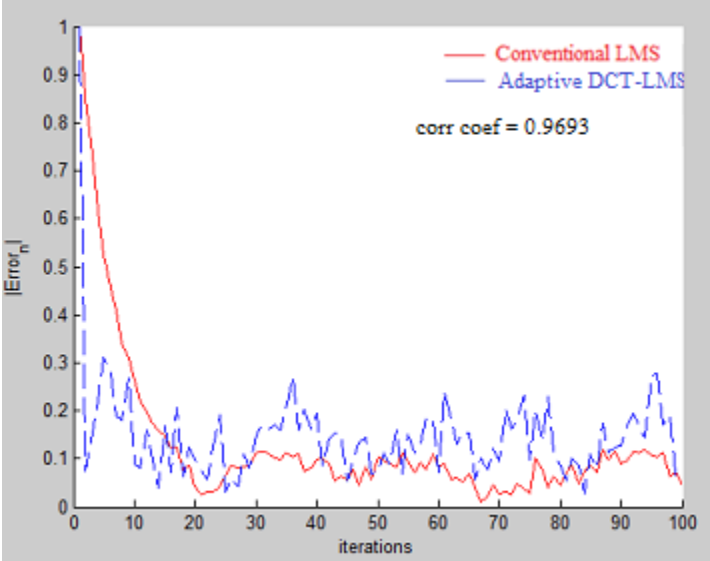

Fig 4.18.f Learning curve for DCT-LMS for ρ 0.9693

From figure 4.18.f, we observe that the learning curve for DCT-LMS converges faster than that of conventional LMS. However, the steady state error for DCT-LMS is more as compared to conventional LMS.

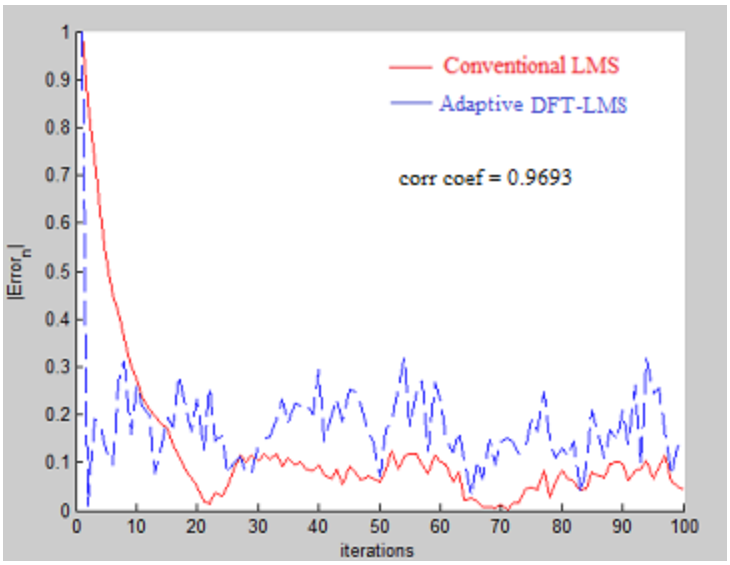

Fig 4.18.g Learning curve for DFT-LMS for ρ 0.9693

From figure 4.18.g, we observe that the learning curve for DFT-LMS converges faster than that of conventional LMS. However, the steady state error for DFT-LMS is much more as compared to conventional LMS.

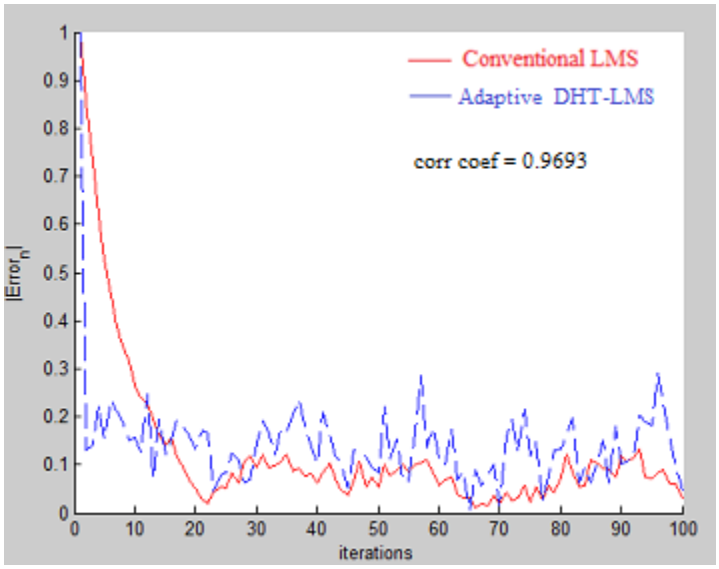

Fig 4.18.h Learning curve for DHT-LMS for ρ 0.9693

From figure 4.18.h, we observe that the learning curve for DHT-LMS converges almost similar to that of conventional LMS. However, the steady state error for DHT-LMS is much more as compared to conventional LMS.

4.4 Implementation of Conventional Beamformer and Adaptive TDLMS Beamformer in LabVIEW

In this section, we model the block diagram shown in figure 3.1 and 3.2 in MATLAB. The sampling frequency is set as 3300Hz. The input signals, ‘signal1’ and ‘signal2’ are assumed to be sinusoidal and white Gaussian noise is added such that resulting SNR is 10 dB. The conventional LMS algorithm stated in equation 3.15 and equation 3.16 have been modeled in figure 4.19. The step size used is 0.004. The simulation results for conventional LMS algorithm have been shown in figure 4.20. The different adaptive TDLMS algorithms have been modeled in figure 4.21 in addition to conventional LMS algorithm and the corresponding results are shown in figure 4.22.

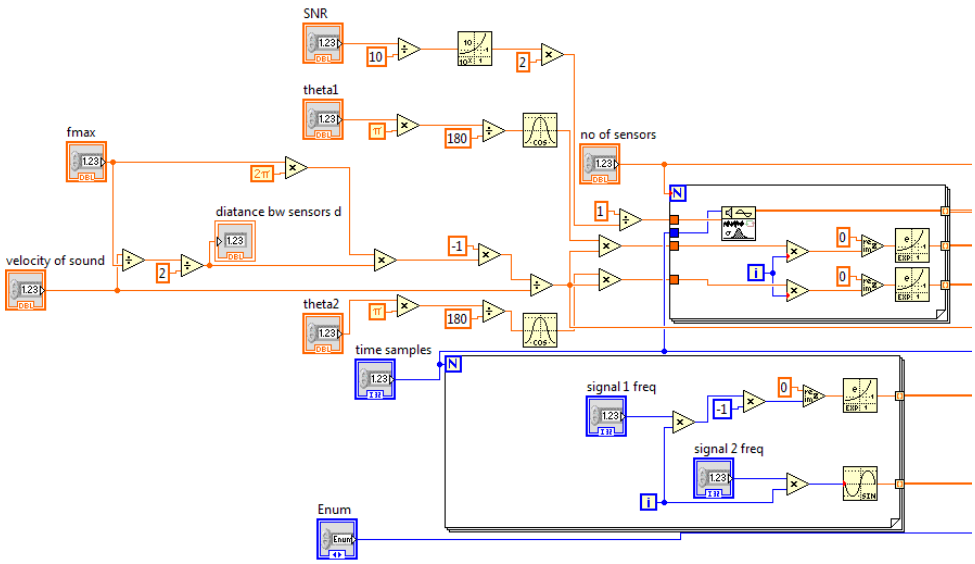

Fig. 4.19.a LabVIEW implementation of Conventional Beamformer

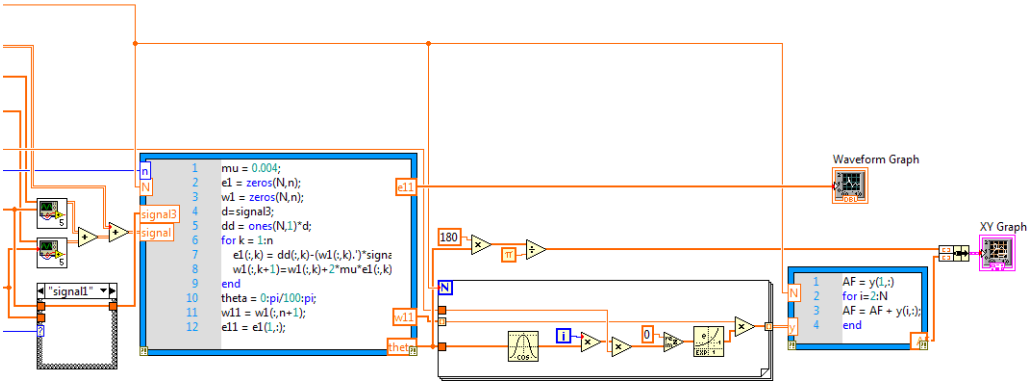

Fig. 4.19.b LabVIEW implementation of Conventional Beamformer(contd.)

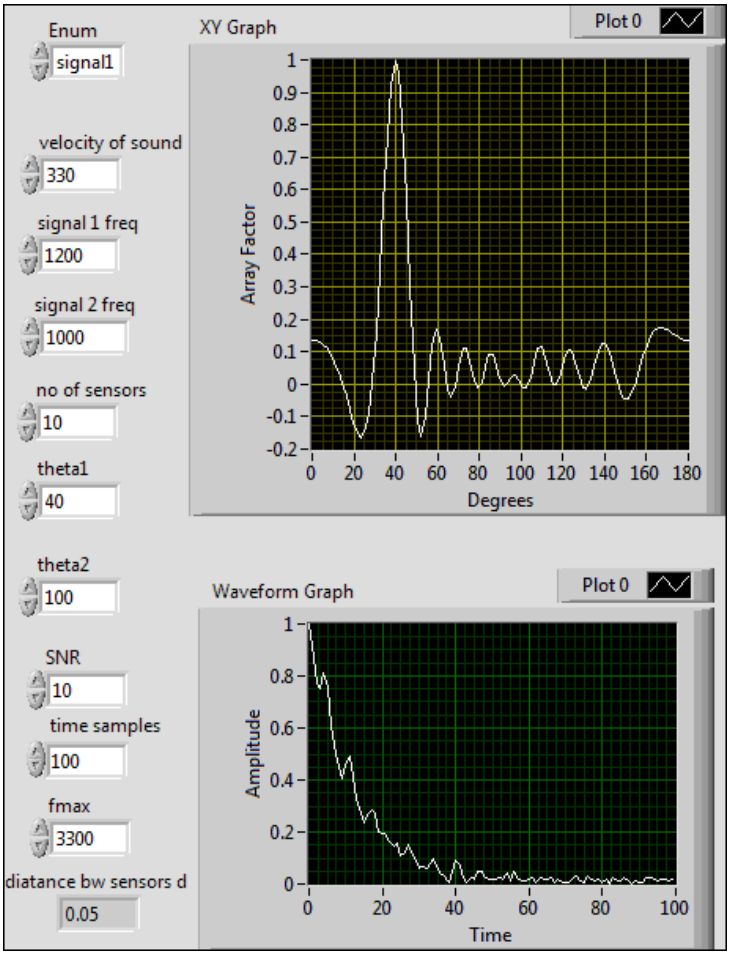

Fig. 4.20 Simulation output of Conventional Beamformer

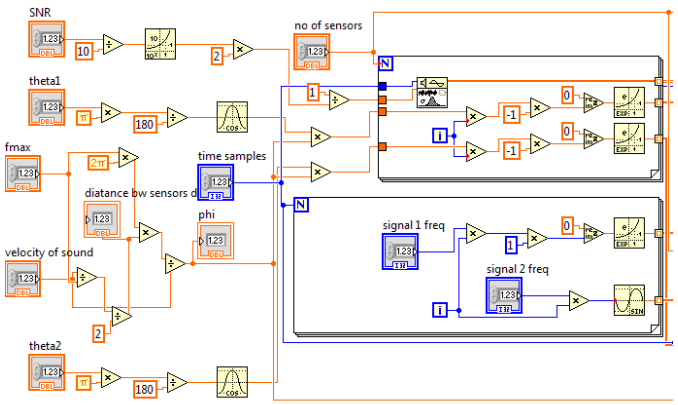

Fig. 4.21.a LabVIEW implementation of tdlms beamformer

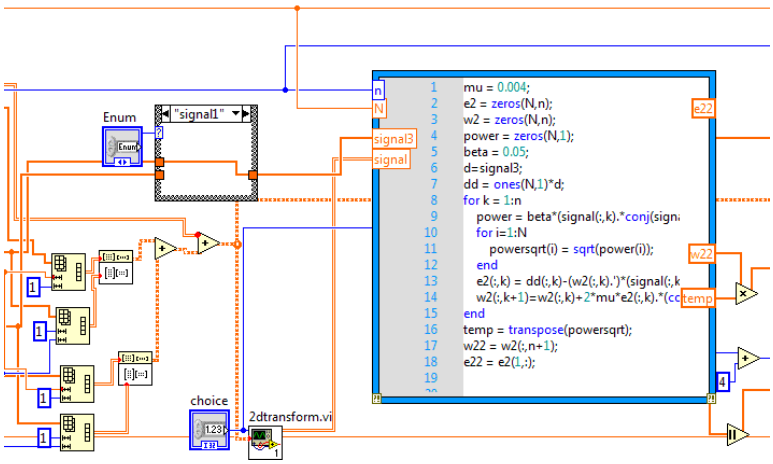

Fig. 4.21.b LabVIEW implementation of tdlms beamformer(contd.)

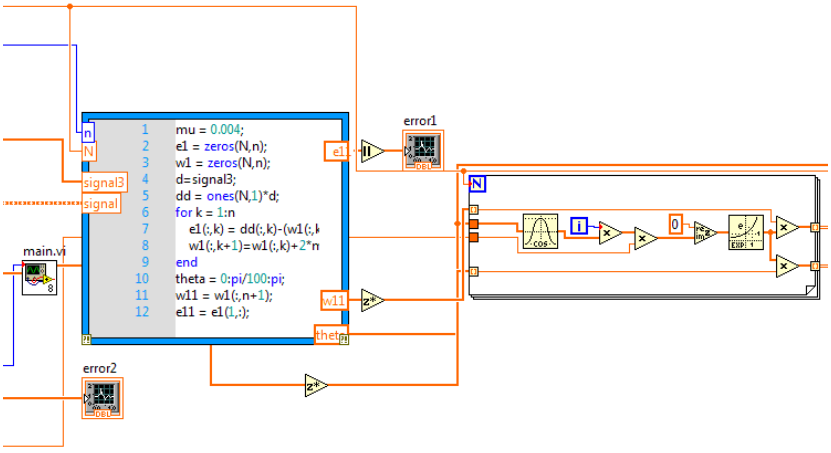

Fig. 4.21.c LabVIEW implementation of tdlms beamformer(contd.)

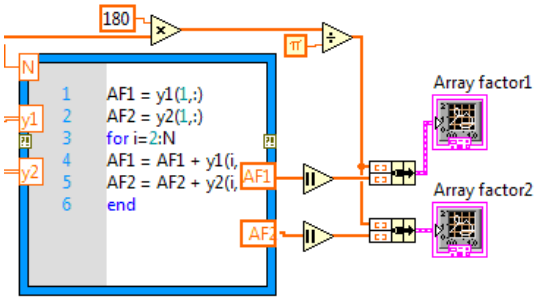

Fig. 4.21.d LabVIEW implementation of tdlms beamformer(contd.)

The simulation results are as follows

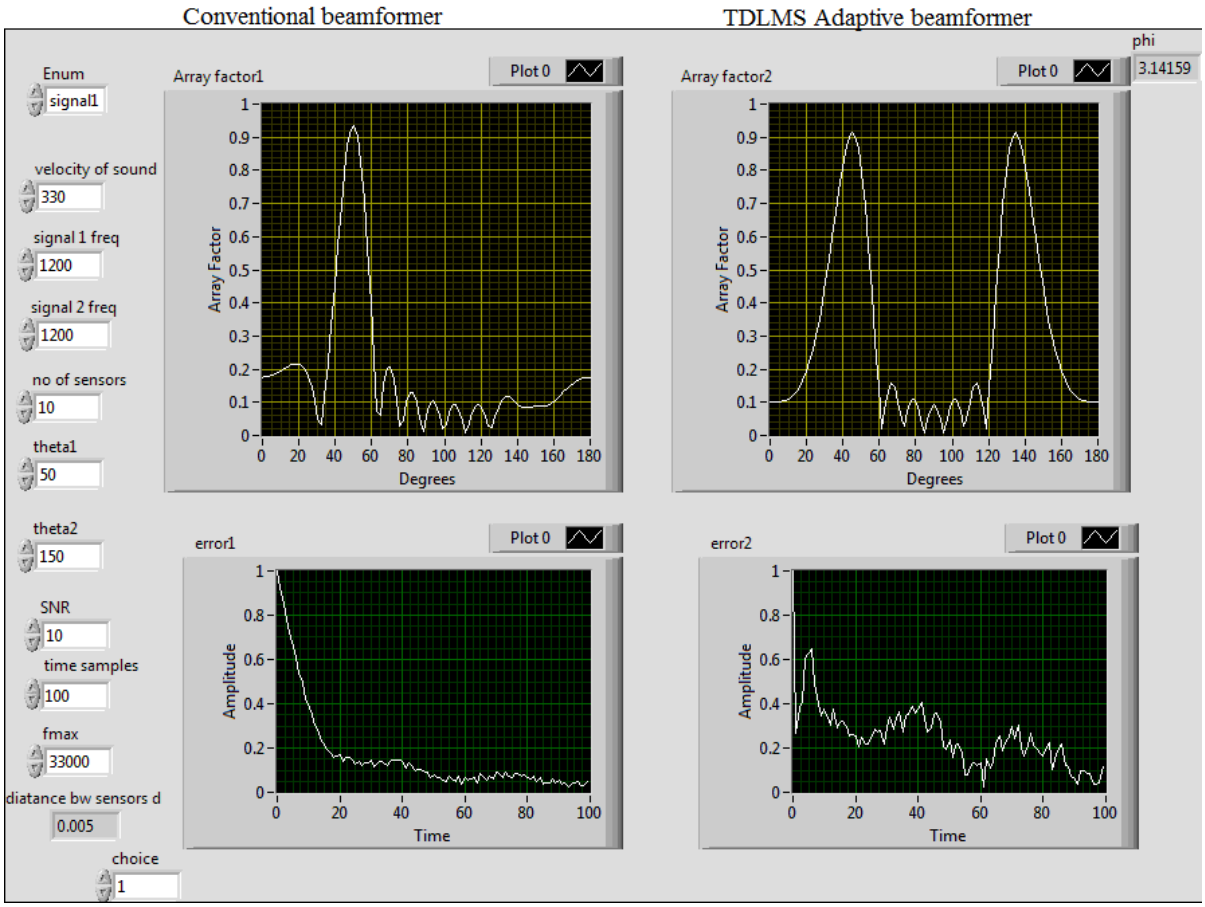

Fig. 4.22 Simulation results of tdlms beamformer vs conventional beamformer

CHAPTER 5
CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

The problem of optimizing the learning curve in a beamformer has been discussed a lot in the literature. Numerous adaptive algorithms have been theorized and implemented based on some criteria like MMSE or the least-square. The LMS, which was implemented on the basis of MMSE criteria, suffered from slow convergence issues and trade-off between convergence and steady state error. Much research work has been done in this area to improve the convergence rate without increasing the complexity of the computation. One such algorithm i.e. TDLMS was proposed since it has a similar complexity to that of conventional LMS algorithm, and it provided better convergence results.

In this thesis, we first proposed a unified architecture for discrete trigonometric transforms that is efficient in terms of hardware and computations as discussed in chapter 2 for DST, DCT, DFT and DHT. A unified architecture was chosen since these transforms are least computationally complex, and moreover, each one of have them have their own advantages. The DCT yields better decorrelation results for large values of ρ ($0< \rho<1$) whereas, DST results in better decorrelation for small values of ρ. The DFT is the basic transform that has been used since a long time. The DHT provided an alternative to DFT since it did not involve complex coefficients.

Next, we implemented TDLMS beamformer using the unified architecture. Since the input signal varies continuously thus resulting in different values of ρ at different instants, we used the unified architecture for transforms. The DCT-LMS can be used when the autocorrelation in the input signal is high. The DST-LMS can be used when the autocorrelation in the input signal is low. Thus, we exploit the properties of different transforms to obtain the best resulting for different values of autocorrelation coefficient. Also, the DFT-LMS provides more gain when compared to other algorithms, however the convergence is slow and steady state error is high, thus it is not that useful for most applications. The DHT-LMS yields the same results in terms of array factor gain as DCT-LMS except for slow convergence and high steady state error, thus it also isn’t useful for most applications.

5.2 Future Scope

The research in the field of transform domain adaptive algorithms has great scope. To conclude, we would like to discuss a number of areas that still need further investigation.

- In this thesis, we have discussed the conventional LMS algorithm and employed discrete transforms to the conventional LMS algorithm to study the convergence performance. Many robust algorithms have been proposed in the past that has not been studied in this thesis. These robust algorithms can deal with errors in input signal which might yield accurate results in the implementation of beamformer.
- The step size used in the LMS algorithm was not variable. A variable step size can be used with transform domain LMS which might lead to better and more accurate results.
- A practical implementation of beamformer has also not been discussed. We believe that practical demonstration of the system would lead to more accurate results.

REFERENCES

[1] B. D. Van Veen, K.M. Buckley, “Beamforming: a versatile approach to spatial filtering”, IEEE ASSP Magazine, April 1998.

[2] Balanis, Constantine A. Antenna Theory: Analysis and Design. 3rd ed. Hoboken, NJ: John Wiley, 2005.

[3] I. Bekkerman, J. Tabrikian, “Target Detection and Localization Using MIMO Radars and Sonars”, IEEE Transactions on Signal Processing, vol. 54, no. 10, pp. 3873-3883, Oct. 2006

[4] Jianping Yao, “Microwave Photonics”, Journal of Lightwave Technology, vol. 27, no. 3, pp. 314-355, Feb. 2009.

[5] P. A. Rosen, S. Hensley , I. R. Joughin, F. K. Li, S. N. Madsen, E. Rodriguez and R. M. Goldstein, “Synthetic aperture radar interferometry”, IEEE Proceedings, vol. 88, no. 3, pp. 333-382, March 2000.

[6] Hayssam Dahrouj, Wei Yu, “Coordinated beamforming for the multicell multi-antenna wireless system”, IEEE Transactions on Wireless Communications, vol. 9, no.5, pp. 1748-1759, May 2010.

[7] J. C. Chen, Kung Yao and R. E. Hudson, “Source localization and beamforming”, IEEE Signal Processing Magazine, vol. 19, no. 2, pp. 30-39, Mar 2002.

[8] J. A. TenCate , T. G. Muir , A. Caiti , A. Kristensen , J. F. Manning , J. A. Shooter , R. A. Koch and E. Michelozzi, "Beamforming on seismic interface waves with an array of geophones on the shallow sea floor", IEEE J. Ocean. Eng., vol. 20, no. 4, pp. 300-310, 1995

[9] J. J. Lee, R. Y. Loo, S. Livingston, V. I. Jones, J. B. Lewis, Huan-Wun Yen, G. L. Tangonan, M. Wechsberg, “Photonic wideband array antennas”, IEEE Transactions on Antennas and Propagation, vol. 43, no. 9, pp. 966-982, Sep 1995.

[10] Johan Fredrik Synnevag, Andreas Austeng, Sverre Holm, “Adaptive Beamforming Applied to Medical Ultrasound Imaging”, IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, vol. 54, no. 8, pp. 1606-1613, Aug 2007.

[11] Xu Li, E. J. Bond, B. D. Van Veen, S. C. Hagness, "An overview of ultra-wideband microwave imaging via space-time beamforming for early-stage breast-cancer detection", IEEE Antennas and Propagation Magazine, vol. 47, no. 1, pp. 19-34, Feb 2005.

[12] B. Widrow, J. R. Glover, J. M. McCool, J. Kaunitz, C. S. Williams, R. H. Hearn, J. R. Zeidler, Jr. Eugene Dong,R. C. Goodlin, "Adaptive noise cancelling: Principles and applications", IEEE Proceedings, vol. 63, no. 12, pp. 1692-1716, Dec. 1975.

[13] L. C. Godara, "Application of antenna arrays to mobile communications, part II: beam-forming and direction-of arrival considerations," Proceedings of the IEEE, vol. 85, no. 8, pp. 1195–1245, 1997.

[14] S. Applebaum, "Adaptive arrays," IEEE Transactions on Antennas and Propagation, vol. 24, no. 5, pp. 585–599, 1976.

[15] S. P. Applebaum and D. J. Chapman, "Adaptive arrays with main beam constraints," IEEE Transactions on Antennas and Propagation, vol. 34, pp. 650–662, 1986.

[16] Multiple sidelobe cancellor

[17] B. D. van Veen and R. A. Roberts, "Partially adaptive beam former design via output power minimization," IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 35, no. 11, pp. 1524–1532, 1987.

[18] T. K. Sarkar and N. Sangruji, "Adaptive nulling system for a narrow-band signal with a look-direction constraint utilizing the conjugate gradient method," IEEE Transactions on Antennas and Propagation, vol. 37, no. 7, pp. 940–944,1989.

[19] H. Chen, T. K. Sarkar, S. A. Dianat, and J. D. Brule, "Adaptive spectral estimation by the conjugate gradient method," IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 34, no. 2, pp. 272–284, 1986.

[20] A. Cantoni and L. C. Godara, "Fast algorithms for time domain broadband adaptive array processing," IEEE Transactions on Aerospace and Electronic Systems, vol. 18, no. 5, pp. 682–699, 1982.

[21] C. W. Jim, "A comparison of two LMS constrained optimal array structures," Proceedings of the IEEE, vol. 65, no. 12, pp. 1730–1731, 1977.

[22] J. T. Mayhan, "Adaptive nulling with multiple-beam antennas," IEEE Transactions on Antennas and Propagation, vol. 26, no. 2, pp. 267–273, 1978.

[23] C.-C. Yeh, W.-D. Wang, T.-H. S. Chao, and J. Mar, “Least squares regenerative hybrid array for adaptive beamforming,” IEEE Transactions on Antennas and Propagation, vol. 38, no. 4, pp. 489–497, 1990.

[24] L. J. Griffiths and C. W. Jim, “An alternative approach to linearly constrained adaptive beam forming,” IEEE Transactions on Antennas and Propagation, vol. 30, no. 1, pp. 27–34, 1982.

[25] O. L. Frost III, “An algorithm for linearly constrained adaptive array processing,” Proceedings of the IEEE , vol. 60, no. 8, pp. 926–935, 1972.

[26] S. Narayan, A. M. Peterson, and M. J. Narasimha, “Transform domain LMS algorithm,” IEEE Trans. Acoust. Speech, Signal Process., vol. ASSP-31, no. 3, pp. 609–615, Jun. 1983.

[27] H. L. Van Trees, Detection, estimation, and modulation theory, optimum array processing, John Wiley & Sons, 2004.

[28] M. Dentino, J. McCool, and B. Widrow, “Adaptive filtering in the frequency domain," Proc. of the IEEE, vol. 66, pp. 1658{1659, Dec. 1978.

[29] B. Widrow and M. E. Hoff,Jr., “Adaptive switching circuits," tech. rep., Stanford Electron. Labs., Stanford, CA, June 1960.

[30] T. Aboulnasr, K. Mayyas, “A robust variable step-size LMS-type algorithm: analysis and simulations”, IEEE Transactions on Signal Processing, vol. 45, no. 3, pp. 631-639, Mar 1997.

[31] D. T. M. Slock, “On the convergence behavior of the LMS and the normalized LMS algorithms”, IEEE Transactions on Signal Processing, vol. 41, no. 9, pp. 2811-2825, Sep 1993.

[32] G. Long, F. Ling, J. G. Proakis, “The LMS algorithm with delayed coefficient adaptation”, IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 37, no. 9, pp. 1397-1405, Sep 1989.

[33] G J.W. Cooley and J.W.Tukey, An algorithm for the machine computation of complex Fourier series, Math. Comput. (19) (1965) 297-301.

[34] N. Ahmed, T. Natarajan and K.R. Rao, Discrete Cosine Transform, IEEE trans. Computers (C-23) (1974) 90-93.

[35] A.K. Jain A fast Karhunen-Loeve Transform for a class of random processes, IEEE trans. Commun. (24) (1976) 1023-1029.

[36] R.N. Bracewell, "The Hartley Transform", Oxford University Press, 1986.

[37] R. N. Bracewell, Discrete Hartley Transform, J. Opt. Soc. Amer. (3-12) (1983) 1832-1835.

[38] P. Duhamel, M. Vetterli, Improved Fourier and Hartley transform algorithms: Applications to cyclic convolutions of real data, IEEE Trans. on Acoustics, Speech and Signal Processing (35) (1987) 818-824.

[39] Cheng Li-Zhi, Tong Li, Jiang Zeng-Rong, Real transform algorithm for computing discrete circular deconvolution, IEEE Int. Conf. on Signal Processing (1) (1996) 166-169.

[40] C.-C. Tseng and S.-L. Lee, Design of digital IIR integrator using discrete Hartley transform interpolation method, in Proc. IEEE Int. Symp. Circuits Syst. (2009) 2181–2184.

[41] M. S. Moreolo, R. Munoz, and G. Junyent, Novel power efficient optical ofdm based on hartley transform for intensity-modulated direct-detection systems, J. Lightw. Technol. (28-5) (2010) 798–805.

[42] M. S. Moreolo, J. M. Fabrega, L. Nadal, G. Junyent , FHT-based architectures for multicarrier modulation in direct detection and coherent optical systems, in Proc. Int. Conf. on Transp. Opt. Netw. (ICTON) (2011).

[43] S. A. Martucci, R. M. Mersereau, New approaches to block filtering of images using symmetric convolution and the DST or DCT, IEEE ISCAS '93, Chicago, IL (1993) 259-262.

[44] S. K. Zhao, Z. H. Man, S. Y. Khoo, and H. R. Wu, Stability and convergence analysis of transform-domain LMS adaptive filters with second-order autoregressive process, IEEE Trans. Signal Process. (57-1) (2009) 119–130.

[45] T. M. Lehmann, C. Gonncr, and K. Spitzer, Survey: interpolation methods in medical image processing, IEEE Trans. Med. hug. (18) (1999) 1049-1075.

[46] S.-J. Chirn, C.-Y. Sung, The performance of hybrid adaptive beamforming algorithm for jammers suppression, IEEE trans on Antennas and Propagation (42) (1994) 1223-1231.

[47] M.A. Tinati, A. Rastegarnia, T. Y. Rezaii, A transform domain based Least Mean Mixed-Norm algorithm to improve adaptive beamforming, IEEE ICEE, Lahore (2008) 1-4.

[48] J.-J. Ding, Y.-W. Huang, P.-Y. Lin, S.-C. Pei, H.-H. Chen, and Y.-H. Wang, Two-dimensional orthogonal DCT expansion in trapezoid and triangular blocks and modified JPEG image compression, IEEE Trans. Image Process. (22-9) (2013) 3664–3675.

[49] Y. S. Park and H. W. Park, Design and analysis of an image resizing filter in the block-DCT domain, IEEE Trans. Circuits Syst. Video Technol., (14-2) (2004) 274–279.

[50] Huijun Ding, Ing Yann Soon, and Chai Kiat Yeo, A DCT-Based Speech Enhancement System With Pitch Synchronous Analysis, IEEE Transactions on audio, speech, and language processing (19-8) (2011).

[51] N. L. Tsitas, Modelling of computational electromagnetics problems by using discrete trigonometric transforms matrices, IEEE URSI International Symposium on Electromagnetic Theory(EMTS) (2010) 958-961.

[52] K. F. Warnick and W. C. Chew, Accuracy of the method of moments for scattering by a cylinder, IEEE Trans. Microw. Theory Tech. (48) (2000) 1652–1660.

[53] J. L. Wang, C. B. Wu, B. D. Liu, J. F. Yang, Implementation of discrete cosine transform and its inverse by recursive structures, IEEE Worshop on Signal Processing System, SiPS 99, Taipei (1) (1999) 120-130.

[54] P. Jain, B. Kumar, S. B. Jain, Discrete sine transform and its inverse-realization through recursive algorithms, International Journal of Circuit Theory and Applications (36) (2007) 441-449.

[55] K. J. R. Liu, C.-T. Chiu, Unified parallel lattice structures for time-recursive discrete cosine/sine/ Hartley transforms, IEEE Transactions on Signal Processing (41) (1993) 1357-1377.

[56] K. Maharatna, A. S. Dhar, and S. Banerjee, A VLSI array architecture for realization of DFT, DHT, DCT and DST, Signal Process. (81) (2001) 1813–1822.

[57] D.C. Kar, V. V. Bapeswara Rao, A CORDIC-based unified systolic architecture for sliding window applications of discrete trsansforms, IEEE Transactions on Signal Processing, (44) (1996) 441-444.

[58] Shen-Fu Hsiao, Wei-Ren Shiue, Low cost unified architectures for the computation of discrete trigonometric transforms, IEEE International Conference on Acoustics, Speech and Signal Processing, 2000. ICASSP ’00, Istanbul (6) (2000) 3299-3302.

[59] Wen-Hsein Fang, Ming-Lu Wu, An efficient unified systolic architecture for the computation of discrete trigonometric transforms, IEEE International Symposium on Circuits and Systems, 1997. ISCAS ’97 (3) (1997) 2092-2095.

[60] Sung Bam Pan, Rae-Hong Park, Unified systolic arrays for computation of the DCT/DST/DHT, IEEE Transactions on Circuits and Systems for Video Technology (7) (1997) 413-419.

[61] Ihor Prots’ko, The Algorithm and Structures for Efficient Computation of Type II/III DCT/ DST/ DHT Using Cyclic Convolutions, International Journal of Signal Processing Systems (2-2) (2014) 119-127.

[62] B. Das and S. Banerjee, Unified CORDIC-based chip to realise DFT/DHT/DCT/DST, IEEE Computers and Digital Techniques, Proceedings (149-4) (2002) 121-127.

[63] M.Amiri, R.Ebrahimi Atani, S. Mirzakuchaki, M.Mahdavi, Design and Implementation of 50 MHz DXT Coprocessor, 10th Euro micro Conference on Digital System Design, Lubeck, Germany, 2007.

[64] D. F. Chiper, M. N. S. Swamy, M. O. Ahmad, and T. Stouraitis, Systolic algorithms and a memory-based design approach for a unified architecture for the computation of DCT/DST/IDCT/IDST, IEEE Trans. Circuits Syst. I, Reg. Papers (52-6) (2005) 1125–1137.

[65] P. A. Lynn, Online digital filters for biological signals: Some fast designs for a small computer, Med . Biol. Eng. Comput. (15-5) (1977) 534–540.

[66] H. T. Anastassiu, Fast, simple and accurate computation of the currents on an arbitrarily large circular loop antenna, IEEE Trans. Antennas Propag. (54) (2006) 860–866.

[67] Proakis JG, Manolakis DG. Digital Signal Processing (3rd edn). Prentice-Hall: Englewood Cliffs, NJ, U.S.A., 1996.

[68] M. Elhadad, et al., Application of trigonometric transforms in discrete multi tone systems, in Computer Engineering and Systems, 2009. ICCES 2009. International Conference, (2009) 171-176.

[69] M. S. Moreolo, V. Sacchieri, G. Cincotti, and G. Junyent, Trigonometric transforms for high-speed optical networks: all-optical architectures and optical OFDM, J. Netw. (5) (2010) 1248–1253.

[70] E. Eweda, "Comparison of RLS, LMS, and sign algorithms for tracking randomly time-varying channels," IEEE Trans. on Signal Processing, vol. 42, no.11, pp. 2937-2944, 1994.

[71] S. Haykin, Adaptive Filter Theory. 4th ed. Englewood Cliffs, NJ: Prentice Hall, 2002.

[72] B. Widrow and J. M. McCool, “Stationary and nonstationary leaming characteristics of the LMS filter," Proc. of the IEEE, vol. 64, pp. 1151{1162, Aug. 1976.

[73] D. I. Kim and P. D. Wilde, “Performance analysis of the DCT-LMS adaptive filtering algorithm," Signal Processing, vol. 80, pp. 1629-1654, Aug. 2000.

APPENDIX-A

A. FORMULA FOR RECURSIVE STRUCTURE FOR DFT

From eq. (4a), we have

$$A_j(k) = \sum_{n=0}^{j} w_k(j-n)\{\cos((n+1)\theta_k) + i\sin((n+1)\theta_k)\} \quad \text{(A.1)}$$

Using identities eq. (A2a) and (A2b) in eq. (A1)

$$\cos r\theta_k = 2\cos(r-1)\theta_k \cos\theta_k - \cos(r-2)\theta_k \quad \text{(A.2.a)}$$

$$\sin r\theta_k = 2\sin(r-1)\theta_k \cos\theta_k - \sin(r-2)\theta_k \quad \text{(A.2.b)}$$

$$A_j(k) = \sum_{n=0}^{j} w_k(j-n)\{2\cos(n\theta_k)\cos\theta_k - \cos((n-1)\theta_k)\} + i\sum_{n=0}^{j} w_k(j-n)\{2\sin(n\theta_k)\cos\theta_k - \sin((n-1)\theta_k)\} \quad \text{(A.3)}$$

Redistributing the terms in eq. (A.3), we get

$$A_j(k) = 2\cos\theta_k\left\{\sum_{n=0}^{j} w_k(j-n)\cos(n\theta_k)\right\} + i2\cos\theta_k\left\{\sum_{n=0}^{j} w_k(j-n)\sin(n\theta_k)\right\} - \sum_{n=0}^{j} w_k(j-n)\cos((n-1)\theta_k) - i\sum_{n=0}^{j} w_k(j-n)\sin((n-1)\theta_k) \quad \text{(A.4)}$$

Replacing n by n-1 in the first two terms of eq. (A.4) and n by n-2 in the last two terms of eq. (A.4)

$$A_j(k) = 2\cos\theta_k\left\{\sum_{n=-1}^{j-1} w_k(j-1-n)\cos\big((n+1)\theta_k\big)\right\} + i2\cos\theta_k\left\{\sum_{n=-1}^{j-1} w_k(j-1-n)\sin((n+1)\theta_k)\right\} - \sum_{n=-2}^{j-2} w_k(j-2-n)\cos((n+1)\theta_k) - i\sum_{n=-2}^{j-2} w_k(j-2-n)\sin((n+1)\theta_k) \quad \text{(A.5)}$$

$$A_j(k) = 2\cos\theta_k \left\{ w_k(j) + \sum_{n=0}^{j-1} w_k(j-1-n)\cos\big((n+1)\theta_k\big) \right\} - w_k(j)\cos\theta_k - w_k(j-$$

$$1) - \sum_{n=0}^{j-2} w_k(j-2-n)\cos\big((n+1)\theta_k\big) + i2\cos\theta_k \left\{ \sum_{n=0}^{j-1} w_k(j-1-n)\sin\big((n+ \right. \quad \text{(A.6)}$$

$$\left. 1)\theta_k\big) \right\} + iw_k(j)\sin\theta_k - i\sum_{n=0}^{j-2} w_k(j-2-n)\sin((n+1)\theta_k)$$

$$A_j(k) = 2\cos\theta_k \left\{ \sum_{n=0}^{j-1} w_k(j-1-ń)\cos\big((ń+1)\theta_k\big) + i\sum_{n=0}^{j} w_k(j-1-ń)\sin\big((ń+ \right.$$

$$\left. 1)\theta_k\big) \right\} + w_k(j)\{\cos\theta_k + i\sin\theta_k\} - w_k(j-1) - \left\{ \sum_{n=0}^{j-2} w_k(j-2-ń)\cos\big((ń+1)\theta_k\big) + \right. \quad \text{(A.7)}$$

$$\left. i\sum_{n=0}^{j} w_k(j-2-ń)\sin\big((ń+1)\theta_k\big) \right\}$$

$$\sum_{n=0}^{j-1} w_k(j-1-n)\cos\big((n+1)\theta_k\big) +$$

$$i\sum_{n=0}^{j-1} w_k(j-1-n)\sin\big((n+1)\theta_k\big) \text{ can be expressed as } A_{j-1}(k)$$

$$\sum_{n=0}^{j-2} w_k(j-2-n)\cos\big((n+1)\theta_k\big) +$$

$$i\sum_{n=0}^{j-2} w_k(j-2-n)\sin\big((n+1)\theta_k\big) \text{ can be expressed as } A_{j-2}(k)$$

$$A_j(k) = 2\cos\theta_k A_{j-1}(k) + w_k(j)\{\cos\theta_k + i\sin\theta_k\} - w_k(j-1) - A_{j-2}(k) \quad \text{(A.8)}$$