International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:3, No:2, 2009

Algorithms for the Fast Computation of PWL and PHL Transforms

Fituri H Belgassem, Abdulbasit Nigrat, Seddeeq Ghrari
Higher Institute of Electronics Engineering, P.O.Box 38645 Beni-Walid, Libya
Email: fhbelgasem@yahoo.com

***Abstract*—**In this paper, the construction of fast algorithms for the computation of Periodic Walsh Piecewise-Linear PWL transform and the Periodic Haar Piecewise-Linear PHL transform will be presented. Algorithms for the computation of the inverse transforms are also proposed. The matrix equation of the PWL and PHL transforms are introduced. Comparison of the computational requirements for the periodic piecewise-linear transforms and other orthogonal transforms shows that the periodic piecewise-linear transforms require less number of operations than some orthogonal transforms such as the Fourier, Walsh and the Discrete Cosine transforms.

***Keywords*—**Piecewise linear transforms, Fast transforms, Fast algorithms.

I. Introduction

FAST algorithms are used to greatly reduce the number of computations required to determine the transform coefficients as compared to direct computation of the transform. The main idea of efficient or fast computational algorithms is the ability to subdivide the total computational task into a series of computational steps such that partial results from initial steps can be repeatedly utilized in subsequent steps. These algorithms result from matrix manipulations such as sparse matrix factorization and matrix transpose [1],[2]. For the discrete transforms which are based on piecewise-linear functions such as PWL and PHL transforms, the methods of matrix factorization and transposition do not give satisfactory results due to the non-orthogonality and non-symmetry properties of the matrices of these transforms. This also implies that a completely different fast algorithm is required for the computation of the inverse transform as opposed to orthogonal transforms where the fast algorithm of the inverse transform usually has a similar structure as that of the forward algorithm with minor modifications. Fast algorithms for the computation of piecewise-linear transforms are based on the reordering of the transform matrix in such a way as to obtain sub-matrices having a structure identical to Walsh-Hadamard matrices.

II. Periodic Piecewise-Linear Transforms

1. PWL Transform: The Periodic Walsh Piecewise-Linear PWL functions which are the basis functions of the PWL transform are obtained by integrating periodic Walsh functions [5]. The matrix equation of the forward and inverse PWL transforms may are given by:
(a) Forward transform

$$[C(N)]=[-2^{-(k+1)}][PWL(N)][X(N)] \quad (1)$$

(b) Inverse transform

$$[X(N)]=[IPWL(N)][C(N)] \quad (2)$$

where [C(N)] -vector of PWL coefficients
[X(N)] -vector of sampled signal.
[PWL(N)]-matrix of forward transform.
[IPWL(N)]-matrix of inverse transform. $[-2^{-(k+1)}]$-diagonal matrix of normalization.

2. PHL Transform: The set of Periodic Haar Piecewise Linear (PHL) functions [6] is obtained by integrating the well known set of Haar functions. The PHL functions are the basis functions for the PHL transform. The matrix form of the forward and inverse PHL transforms are defined as follows:
(a) Forward transform

$$[C(N)]=\left[-\frac{1}{2^{k+1}}\right][PHL(N)][X(N)] \quad (3)$$

(b) Inverse transform

$$[X(N)]=[IPHL(N)][C(N)] \quad (4)$$

where [C(N)]-vector of PHL coefficients
[X(N)] - vector of sampled signal.
[PHL(N)]-matrix of forward transform. [IPHL(N)] - matrix of inverse transform.
$\left[-\frac{1}{2^{k+1}}\right]$-diagonal matrix of normalization.

III. Fast Computation of the PWL Transform

Construction of a fast algorithm for the computation of the forward PWL transform requires ordering of the transform matrix in such a way so that sub-matrices of orders from (N/2

x N/2) to (2x2) having a structure identical to Walsh-Hadamard matrices for which algorithms already exists, will be obtained. The ordering which achieves this is the bit reversal order (BRO) of the columns of the transform matrix. After this operation the PWL matrix, for N=8, will have the following form:

$$[PWL(8)]_{BRO} = \left[\begin{array}{cc|cc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2 & 2 & 0 & 0 & 0 & 0 \\ 2 & 2 & -2 & -2 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & -2 & -2 & 2 & 2 \\ 2 & -2 & 0 & 0 & -2 & 2 & 2 & -2 \\ 0 & 0 & 2 & -2 & -2 & 2 & -2 & 2 \\ 2 & 2 & 2 & 2 & -2 & -2 & -2 & -2 \end{array}\right]$$

$$= \left[\begin{array}{cc|cc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & -2 & 0 & 0 & & & & \\ 0 & 0 & & & & & & \\ 2 & 2 & \multicolumn{2}{c|}{\boldsymbol{A}} & \multicolumn{4}{c}{\boldsymbol{0}} \\ \hline 0 & 0 & & & & & & \\ 2 & -2 & \multicolumn{2}{c|}{\boldsymbol{0}} & \multicolumn{4}{c}{\boldsymbol{B}} \\ 0 & 0 & & & & & & \\ 2 & 2 & \multicolumn{2}{c|}{-\boldsymbol{A}} & & & & \end{array}\right] \qquad (5)$$

where $\boldsymbol{A}=\begin{bmatrix} -2 & 2 \\ -2 & -2 \end{bmatrix}$; $\boldsymbol{B}=\begin{bmatrix} \boldsymbol{A1} & \boldsymbol{A2} \\ \boldsymbol{A} & \boldsymbol{A} \end{bmatrix}$;

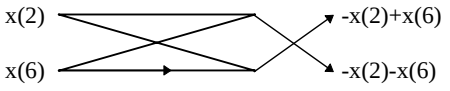
Sub-matrix ***A1*** is obtained by changing the sign of the second column of sub-matrix ***A*** , and ***A2*** is obtained by changing the sign of the first column of ***A*.** Fig..1 represents the flow graphs for sub-matrices ***A*** and ***B*** which are designated by FT(2) and FT(4), respectively. The coefficients of the PWL spectrum can be successively written as:

c(0)=x(0)+x(1)+x(2)+x(3)+x(4)+x(5)+x(6)+x(7)
c(1)=2x(0)-2x(4)
c(2)=2x(6)-2x(2)
c(3)=2x(0)+2x(4)-2x(2)-2x(6)
c(4)=2x(1)-2x(5)+2x(3)+2x(7)
c(5)=2x(0)-2x(4)-2x(1)+2x(5)+2x(3)-2x(7)
c(6)=2x(2)-2x(6)-2x(1)+2x(5)-2x(3)+2x(7)
c(7)=2x(0)+2x(4)+2x(2)+2x(6)-2x(1)-2x(5)-2x(3)-2x(7)

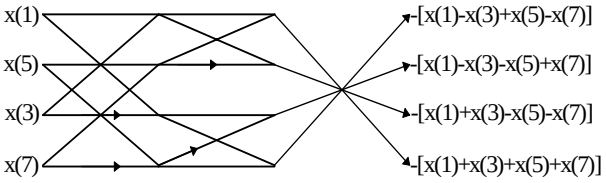
This leads to the signal flow graph for the fast PWL transform for N=8 shown in Fig.2. The arrows in the flow graph represent a negative operation (subtraction). The input data is arranged in a bit reversal order and the output spectral coefficients are obtained in natural order. This algorithm can be generalized for any order N, provided that it is an integer power of 2. It is seen that this algorithm has a similar structure to that proposed in [3]. It requires $\{\frac{N}{4}[(2\log_2 N)+\frac{N}{4}+1]\}$ additions and N normalizations.

The fast algorithm for the computation of the inverse PWL transform is based on the Paley's order of Walsh functions. The IPWL matrix for N=8 has the following form:

$$\left[IPWL(8)\right] = \begin{bmatrix} 1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 \\ 1 & -\frac{1}{2} & 0 & \frac{1}{2} & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & -1 & 0 & 0 & 0 \\ 1 & \frac{1}{2} & 0 & \frac{1}{2} & 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & 0 & -1 & 0 & 0 & 0 \\ 1 & \frac{1}{2} & 0 & -\frac{1}{2} & 1 & -1 & 1 & -1 \\ 1 & 0 & 1 & -1 & -1 & 0 & 0 & 0 \\ 1 & -\frac{1}{2} & 0 & -\frac{1}{2} & 1 & -1 & -1 & 1 \end{bmatrix} \qquad (6)$$



(a) FT(2).



(b) FT(4).

Fig. 1 (a) Flow graph of sub-matrix *A.* (b) Flow graph of sub-matrix *B*

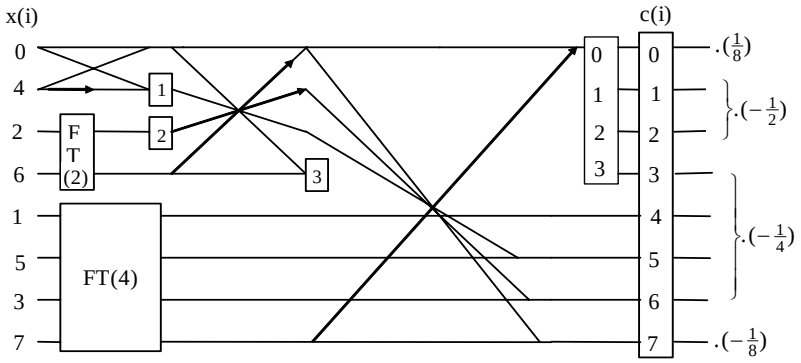


Fig.2 Flow diagram for the fast PWL transform, N=8.

Rearranging the rows of the above matrix according to the bit reversal rule gives:

$$[IPWL(8)]_{BRO} = \left[\begin{array}{cccc|cccc} 1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 1 & -1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 1 & -1 & -1 & 0 & 0 & 0 \\ \hline 1 & -\frac{1}{2} & 0 & \frac{1}{2} & 1 & 1 & 1 & 1 \\ 1 & \frac{1}{2} & 0 & -\frac{1}{2} & 1 & -1 & 1 & -1 \\ 1 & \frac{1}{2} & 0 & \frac{1}{2} & 1 & 1 & -1 & -1 \\ 1 & -\frac{1}{2} & 0 & -\frac{1}{2} & 1 & -1 & -1 & 1 \end{array}\right]$$

$$= \left[\begin{array}{cccc|cccc} 1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 1 & -1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & \multicolumn{2}{c|}{\multirow{2}{*}{$\boldsymbol{A}$}} & -1 & 0 & 0 & 0 \\ 1 & 0 & \multicolumn{2}{c|}{} & -1 & 0 & 0 & 0 \\ \hline 1 & -\frac{1}{2} & 0 & \frac{1}{2} & \multicolumn{4}{c}{\multirow{4}{*}{$\boldsymbol{B}$}} \\ 1 & \frac{1}{2} & 0 & -\frac{1}{2} & \multicolumn{4}{c}{} \\ 1 & \frac{1}{2} & 0 & \frac{1}{2} & \multicolumn{4}{c}{} \\ 1 & -\frac{1}{2} & 0 & -\frac{1}{2} & \multicolumn{4}{c}{} \end{array}\right] \quad (7)$$

where $\boldsymbol{A} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$; $\boldsymbol{B} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{A} \\ \boldsymbol{A} & -\boldsymbol{A} \end{bmatrix}$

In Fig.3 are shown the signal flow graphs for sub-matrices ***A*** and ***B***, represented by IT(2) and IT(4), respectively. Based on the IPWL matrix as given by Eqn.(6), the input sequence x(i) can be determined successively as follows:

x(0)=c(0)-c(1)-c(2)-c(4)

x(4)=c(0)+c(1)-c(2)-c(4)

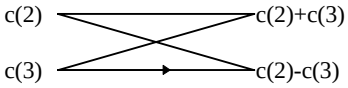
$x(2)=2c(2)+c(3)+\frac{1}{2}[x(0)+x(4)]$

$x(6)=2c(2)-c(3)+\frac{1}{2}[x(0)+x(4)]$ (2.61d)

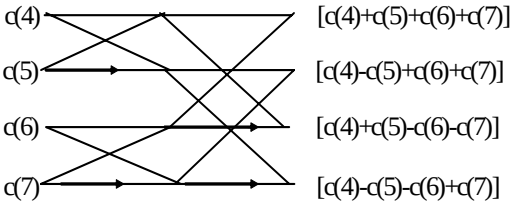
$x(1)=2c(4)+c(5)+c(6)+c(7)+\frac{1}{2}[x(0)+x(2)]$

$x(5)=2c(4)-c(5)+c(6)-c(7)+\frac{1}{2}[x(4)+x(6)]$

$x(3)=2c(4)+c(5)-c(6)-c(7)+\frac{1}{2}[x(2)+x(4)]$



(a) IFT(2).



(b) IFT(4)

.

Fig. 3 (a) Flow graph of sub-matrix *A*. (b) Flow graph of sub-matrix *B*.

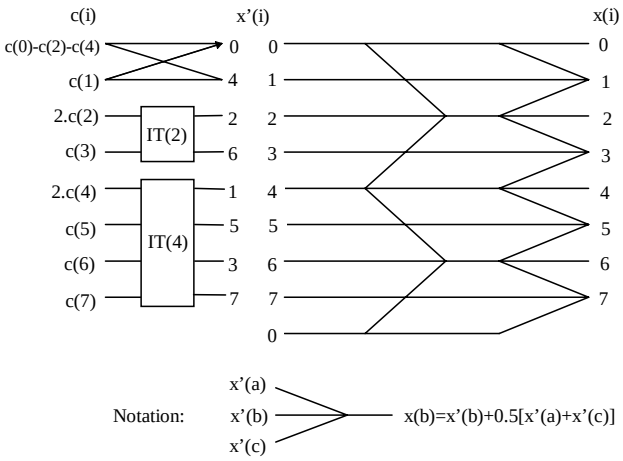


Fig. 4 Flow diagram for the fast IPWL transform, N=8.

IV. Fast Computation of the PHL Transform

$x(7)=2c(4)-c(5)+c(6)-c(7)+\frac{1}{2}[x(0)+x(6)]$

These equations are represented by the flow diagram of the IPWL transform , for N=8, shown in Fig.4. The transform coefficients c(2) and c(4) must be multiplied by 2. The output data sequence x(i); i=0,1, ... , N-1. is in natural order. This algorithm has the advantage that it is recursive and thus can be generalized to any order N (where N is an integer power of 2). It is seen that this algorithm requires less number of additions and multiplications than that given in [3]. The number of additions and multiplications needed are ($N\log_2 N$) and (N-2), respectively.

$$[PHL(8)]=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & -2 & 0 & 0 & 0 \\ \sqrt{2} & 0 & -2\sqrt{2} & 0 & \sqrt{2} & 0 & 0 & 0 \\ \sqrt{2} & 0 & 0 & 0 & \sqrt{2} & 0 & -2\sqrt{2} & 0 \\ 2 & -4 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -4 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -4 & 2 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 2 & -4 \end{bmatrix} \quad (8)$$

If we write the equations defining the PHL transform coefficients c(i), i=0,1,2,.. ..,N-1, including the multiplication by the normalization coefficients [4], we obtain

The forward PHL matrix of order 8 is given by [4]

$$c(0)=x(0)$$
$$c(1)=-\tfrac{1}{2}[2x(0)-2x(4)]$$
$$c(2)=-\tfrac{1}{4}[\sqrt{2}x(0)-2\sqrt{2}x(2)+\sqrt{2}x(4)]$$
$$c(3)=-\tfrac{1}{4}[\sqrt{2}x(0)+\sqrt{2}x(4)-2\sqrt{2}x(6)]$$
$$c(4)=-\tfrac{1}{8}[2x(0)-4x(1)+2x(4)]$$
$$c(5)=-\tfrac{1}{8}[2x(2)-4x(3)+2x(4)]$$
$$c(6)=-\tfrac{1}{8}[2x(4)-4x(5)+2x(6)]$$
$$c(7)=-\tfrac{1}{8}[2x(0)-4x(6)+2x(7)]$$

The above set of equations leads to the flow diagram shown in Fig.5, for N=8. It can be seen that the input data sequence, x(i), to the flow diagram is in natural order, however, the output coefficients , c(i), are in different order. Therefore a final reordering step is required at the end to obtain the natural order of the coefficients. Examination of the flow diagram for the PHL transform reveals that the number of operations required are (2N-3) additions, (N-2) normalizations, and (N-2) binary shifts, however, as known in digital implementation, binary shifts are faster than multiplications or additions.

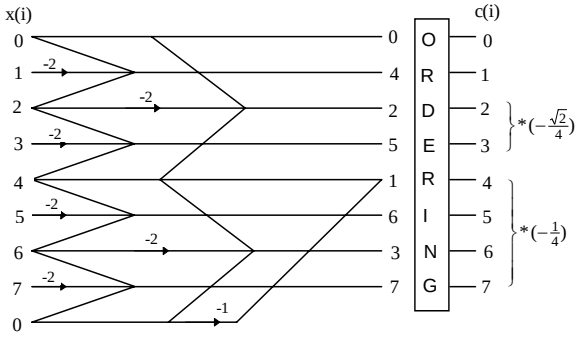


Fig.5 Flow diagram for the fast PHL ransform, N=8.

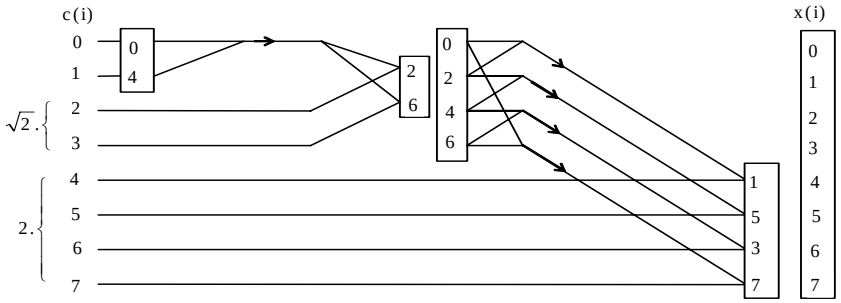
To construct the algorithm for the fast computation of the inverse PHL transform, the rows of inverse matrix are first reordered in a bit reversal order. For N=8 this yields;

$$\left[IPHL(8)\right]_{BRO} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & \frac{1}{2} & \sqrt{2} & 0 & 0 & 0 & 0 & 0 \\ 1 & \frac{1}{2} & 0 & \sqrt{2} & 0 & 0 & 0 & 0 \\ 1 & \frac{1}{4} & \frac{\sqrt{2}}{2} & 0 & 2 & 0 & 0 & 0 \\ 1 & \frac{3}{4} & 0 & \frac{\sqrt{2}}{2} & 0 & 0 & 2 & 0 \\ 1 & \frac{3}{4} & \frac{\sqrt{2}}{2} & 0 & 0 & 2 & 0 & 0 \\ 1 & \frac{1}{4} & 0 & \frac{\sqrt{2}}{2} & 0 & 0 & 0 & 2 \end{bmatrix} \quad (9)$$

The input data sequence, x(i), is then determined successively based on (9), as follows:

$$x(0) = c(0)$$
$$x(4) = c(0) + c(1)$$
$$x(2) = c(0) + \tfrac{1}{2}c(1) + \sqrt{2}c(2) = \sqrt{2}c(2) + \tfrac{1}{2}[x(0) + x(4)]$$
$$x(6) = c(0) + \tfrac{1}{2}c(1) + \sqrt{2}c(3) = \sqrt{2}c(3) + \tfrac{1}{2}[x(0) + x(4)]$$
$$x(1) = c(0) + \tfrac{1}{4}c(1) + \tfrac{\sqrt{2}}{2}c(2) + 2c(4) = 2c(4) + \tfrac{1}{2}[x(0) + x(2)]$$
$$x(3) = c(0) + \tfrac{3}{4}c(1) + \tfrac{\sqrt{2}}{2}c(2) + 2c(5) = 2c(5) + \tfrac{1}{2}[x(2) + x(4)]$$
$$x(5) = c(0) + \tfrac{3}{4}c(1) + \tfrac{\sqrt{2}}{2}c(3) + 2c(6) = 2c(6) + \tfrac{1}{2}[x(4) + x(6)]$$
$$x(7) = c(0) + \tfrac{1}{4}c(1) + \tfrac{\sqrt{2}}{2}c(3) + 2c(7) = 2c(7) + \tfrac{1}{2}[x(0) + x(6)]$$

The above equations can be represented by the flow diagram shown in Fig.6. The algorithm for the inverse PHL transform requires (3N/4) additions and (N-3) multiplications. Partial normalization is performed by post-multiplying the coefficients c(i), i=2,...,N-1 by factors which are a power of $\sqrt{2}$, for total of (N-2) normalizations are required. Note that the arrows in the flow diagrams of the IPHL transform indicate a multiplication by (1/2).



(Note: Arrows indicate a multiplication by 1/2)

Fig. 6 Flow diagram for the fast inverse PHL transform, for N=8.

V. Comparison of Computational Complexity

Computational complexity is one of the most important factors in evaluating a transform. High computational complexity leads to high implementation cost. For the purpose of comparison, the computational requirements of different transforms are listed in Table I. Table I shows the number of additions required for different transforms.

Note that all operations in Table I are real except the multiplications and additions of the FFT which are complex. It can be seen that PHL requires the least number of additions after the IPHL transform, and also requires the least number of normalizations and no multiplications are required, at the expense of some binary shifting operations (which are known to be much faster than additions and multiplications in digital implementation). Therefore the computational complexity of the PHL algorithm is simpler than other transforms considered in this comparison. The IPHL transform requires the least number of multiplications and less number of additions than the other transforms and the same number of normalizations as that of the PHL transform. The IPWL algorithm requires less number of additions and multiplications than that given in [3].

TABLE I
COMPUTATIONAL REQUIREMENTS FOR DIFFERENT TRANSFORMS

Transform	Additions	Multiplications	[illegible]	Binary shifts
PWL	$(N/4)[(2\log_2 N)+(N/4)+11]$		N	
IPWL	$N\log_2 N$	$N-2$		
PHL	$2N-3$		$N-2$	$N-2$
IPHL	$(3N/4)$	$N-3$	$N-2$	
FFT	$N\log_2 N$	$(N/2)\log_2 (N/2)$	N	
WH	$N\log_2 N$		N	
DCT	$(3N/2)(\log_2 N)-N+1$	$(N/2)\log_2 N$		

VI. Conclusion

Methods for constructing the algorithms for the fast computation of the PWL and PHL transforms have proposed. It has been shown that the proposed algorithms are characterized by the simple computational complexity. Comparison analysis of fast algorithms is has considered. Results of this analysis have shown that the computational complexity of the PWL and PHL algorithms are generally simpler than some orthogonal transforms.

References

[1] Brigham E. O. , The Fast Fourier Transform, Prentice-Hall, Englewood Cliffs, NJ, 1974.
[2] Elliot D., Rao K. R. , Fast Transforms : Algoritms, Analysis, Applications, Academic Press, 1982
[3] Dziech A, Kwater M "Fast Transforms Based on Piecewise-Linear Functions" IEEE Workshop on ASSP, Beijing 1986.
[4] A. Dziech, F. Belgassem, S. Aboukres, A. Nabout "Periodic Haar Piecewise Linear Transform" Proceedings of the CESA'96 IMACS/IEEE-SMC Multiconference, Lile-France, July 1996, pp.157-160.
[5] Dziech A, Pardyka I "Shape Approximation using Fast Piecewise Linear Transforms" AMSE Review, vol. 8, No.1,1988, pp.19-30.
[6] Dziech A, Belgassem F, Ammar K, B. Bushofa " One-dimensional Haar Piecewise Linear Series and Transform" Proceedings of the 32nd. Science Week, University of Damascus, Damascus, Syria, Nov. 1992, pp.515-522.
[7] Dziech A , Pardyka I "Shape descriptors based on Fast Piecewise Linear Transforms" AMSE Int conference on Modelling and Simulation, Karlsrhe,1987.
[8] Paul C R, Koch R W "On Piecewise-Linear Basis Functions and Piecewise-Linear Signal Expansion" IEEE Trans. Acoust. Speech Signal Process.,ASSP-22, No.4 , Aug.1974, pp.263-268.
[9] Lee. P, Huang F "Restructured Recursive DCT and DST Algorithms" IEEE Trans. on Signal Processing, vol. 42, No.7, July 1994, pp 1600-1609.