Aleksandr CARIOW, Dorota MAJORKOWSKA-MECH
WEST POMIERANIAN UNIVERSITY OF TECHNOLOGY, SZCZECIN,
49 Żołnierska St., 71-210 Szczecin, Poland

Basic Aspects of Designing a High-performance Processor Structure for Calculating a "true" Discrete Fractional Fourier Transform

Abstract

This paper presents a basic aspects of structural design of the high-performance processor for implementation of discrete fractional Fourier transform (DFrFT). The general idea of the possibility of parallelizing the calculation of the so-called "true" discrete Fourier transform on the basis of our previously developed algorithmic approach is presented. We specifically focused only on the general aspects of the organization of the structure of such a processor, since the details of a particular implementation always depend on the implementation platform used, while the general idea of constructing the structure of the processor remains unchanged.

Keywords: discrete fractional Fourier transform, parallelization of computations, hardware implementation, complexity reduction.

1. Introduction

The discrete fractional Fourier transform has been found many applications in various areas of digital signal and image processing such as signal and image recovery, image restoration and enhancement, image encryption and compression, pattern recognition, digital watermarking, radars [1-8]. The problem of a fast implementation of the DFrFT becomes extremely important for applications requiring real-time processing at high throughput especially in digital signal and image processing. Hence, to meet the stringent requirements to throughput, latency, and power-consumption constraints of real-time DSP systems, developing dedicated hardware implementations, such as application specific integrated circuits (ASICs) or field-programmable gate arrays (FPGAs), has of paramount importance [9-11]. In this paper, we focus on the basic aspects of structural implementation of the algorithmic solution, which was presented in [12], [13]. We named this type of algorithm as "true" DFRFT [13].

2. Preliminaries

A one-dimensional DFrFT can be expressed as a matrix-vector multiplication:

$$\mathbf{Y}_{N\times 1}^{(\alpha)} = \mathbf{F}_N^{(\alpha)}\mathbf{X}_{N\times 1} \tag{1}$$

where $\mathbf{X}_{N\times 1} = [x_0,\ x_1, \ldots, x_{N-1}]^{\mathrm{T}}$ is the input vector which has N elements, $\mathbf{F}_N^{(\alpha)}$ is the discrete fractional transform matrix and α is the fractional order.

The transform matrix of the DFrFT can then be defined as [3]

$$\mathbf{F}_N^{(\alpha)} = \left\| f_{k,n}^{(\alpha)} \right\| = \mathbf{Z}_N \mathbf{\Lambda}_N \mathbf{Z}_N^{\mathrm{T}} \tag{2}$$

Where $k, n = 0, 1, \ldots, N-1$, $\mathbf{\Lambda}_N$ is a diagonal matrix of size N, whose diagonal entries are the eigenvalues of the matrix $\mathbf{E}_N$, $\mathbf{Z}_N$ is the matrix whose columns are normalized mutually orthogonal, eigenvectors of the matrix $\mathbf{E}_N$ and $\mathbf{E}_N = \left\| w_N^{(k,n)} \right\|$ is the matrix of discrete exponential functions [14], $w_N^{(k,n)} = \exp(-j2\pi kn/N)$, and $j = \sqrt{-1}$ is an imaginary unit.

Direct calculation of matrix-vector product (1) requires N^2 multiplication and $N(N-1)$ additions. In some cases, this leads to a significant time delay. Below, we show how to reduce the computation time of DFrFT.

3. Algorithmic and structural aspects of DFrFT processor organization

We will write the matrix $\mathbf{F}_N^{(\alpha)}$ as a sum of three or two matrices and these matrices will have special structures as well. This trick may be useful to reduce the number of arithmetical operations when calculating a product of the matrix $\mathbf{F}_N^{(\alpha)}$ by a vector. The number of components of the sum depends on whether N is even or odd.

If N is even, the matrix $\mathbf{F}_N^{(\alpha)}$ can be written as a sum of three matrices

$$\mathbf{Y}_{N\times 1}^{(\alpha)} = \mathbf{F}_N^{(\alpha)}\mathbf{X}_{N\times 1} = \mathbf{A}_N^{(\alpha)}\mathbf{X}_{N\times 1} + \mathbf{B}_N^{(\alpha)}\mathbf{X}_{N\times 1} + \mathbf{C}_N^{(\alpha)}\mathbf{X}_{N\times 1} \tag{3}$$

where

$$\mathbf{A}_N^{(\alpha)} = \begin{bmatrix} f_{0,0}^{(\alpha)} & f_{0,1}^{(\alpha)} & \Lambda & f_{0,\frac{N}{2}-1}^{(\alpha)} & f_{0,\frac{N}{2}}^{(\alpha)} & f_{0,\frac{N}{2}-1}^{(\alpha)} & \Lambda & f_{0,1}^{(\alpha)} \\ f_{0,1}^{(\alpha)} & 0 & \Lambda & 0 & 0 & 0 & \Lambda & 0 \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ f_{0,\frac{N}{2}-1}^{(\alpha)} & 0 & \Lambda & 0 & 0 & 0 & \Lambda & 0 \\ f_{0,\frac{N}{2}}^{(\alpha)} & 0 & \Lambda & 0 & 0 & 0 & \Lambda & 0 \\ f_{0,\frac{N}{2}+1}^{(\alpha)} & 0 & \Lambda & 0 & 0 & 0 & \Lambda & 0 \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ f_{0,1}^{(\alpha)} & 0 & \Lambda & 0 & 0 & 0 & \Lambda & 0 \end{bmatrix},$$

$$\mathbf{B}_N^{(\alpha)} = \begin{bmatrix} 0 & 0 & \Lambda & 0 & 0 & 0 & \Lambda & 0 \\ 0 & f_{1,1}^{(\alpha)} & \Lambda & f_{1,\frac{N}{2}-1}^{(\alpha)} & 0 & f_{1,\frac{N}{2}+1}^{(\alpha)} & \Lambda & f_{1,N-1}^{(\alpha)} \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ 0 & f_{1,\frac{N}{2}-1}^{(\alpha)} & \Lambda & f_{\frac{N}{2}-1,\frac{N}{2}-1}^{(\alpha)} & 0 & f_{\frac{N}{2}-1,\frac{N}{2}+1}^{(\alpha)} & \Lambda & f_{1,\frac{N}{2}+1}^{(\alpha)} \\ 0 & 0 & \Lambda & 0 & 0 & 0 & \Lambda & 0 \\ 0 & f_{1,\frac{N}{2}+1}^{(\alpha)} & \Lambda & f_{\frac{N}{2}-1,\frac{N}{2}+1}^{(\alpha)} & 0 & f_{\frac{N}{2}-1,\frac{N}{2}-1}^{(\alpha)} & \Lambda & f_{1,\frac{N}{2}-1}^{(\alpha)} \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ 0 & f_{0,N-1}^{(\alpha)} & \Lambda & f_{1,\frac{N}{2}+1}^{(\alpha)} & 0 & f_{1,\frac{N}{2}-1}^{(\alpha)} & \Lambda & f_{1,1}^{(\alpha)} \end{bmatrix},$$

$$\mathbf{C}_N^{(\alpha)} = \begin{bmatrix} 0 & 0 & \Lambda & 0 & 0 & 0 & \Lambda & 0 \\ 0 & 0 & \Lambda & 0 & f_{1,\frac{N}{2}}^{(\alpha)} & 0 & \Lambda & 0 \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ 0 & 0 & \Lambda & 0 & f_{\frac{N}{2}-1,\frac{N}{2}}^{(\alpha)} & 0 & \Lambda & 0 \\ 0 & f_{1,\frac{N}{2}}^{(\alpha)} & \Lambda & f_{\frac{N}{2}-1,\frac{N}{2}}^{(\alpha)} & f_{\frac{N}{2},\frac{N}{2}}^{(\alpha)} & f_{\frac{N}{2}-1,\frac{N}{2}}^{(\alpha)} & \Lambda & f_{1,\frac{N}{2}}^{(\alpha)} \\ 0 & 0 & \Lambda & 0 & f_{\frac{N}{2}-1,\frac{N}{2}}^{(\alpha)} & 0 & \Lambda & 0 \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ 0 & 0 & \Lambda & 0 & f_{1,\frac{N}{2}}^{(\alpha)} & 0 & \Lambda & 0 \end{bmatrix}.$$

It is easy to see that the computation process (3) can be parallelized, which also facilitates the acceleration of calculations.

In this case, the process of calculating the transformation can be represented as follows:

$$\mathbf{Y}_{N\times 1}^{(\alpha)} = \mathbf{\Sigma}_{N\times 3N}\mathbf{W}_{3N}\mathbf{P}_{3N\times N}\mathbf{X}_{N\times 1} \tag{4}$$

where

$$\mathbf{P}_{3N\times N} = \mathbf{1}_{3\times 1}\otimes\mathbf{I}_N = \left[\begin{array}{ccc} 1 & & \\ & \ddots & \\ & & 1 \\ \hline 1 & & \\ & \ddots & \\ & & 1 \\ \hline 1 & & \\ & \ddots & \\ & & 1 \end{array}\right],$$

$$\mathbf{W}_{3N} = \mathbf{A}_N^{(\alpha)}\oplus\mathbf{B}_N^{(\alpha)}\oplus\mathbf{C}_N^{(\alpha)} = \begin{bmatrix} \mathbf{A}_N^{(\alpha)} & \mathbf{0}_N & \mathbf{0}_N \\ \mathbf{0}_N & \mathbf{B}_N^{(\alpha)} & \mathbf{0}_N \\ \mathbf{0}_N & \mathbf{0}_N & \mathbf{C}_N^{(\alpha)} \end{bmatrix},$$

$$\mathbf{\Sigma}_{N\times 3N} = \mathbf{1}_{1\times 3}\otimes\mathbf{I}_N =$$

$$= \left[\begin{array}{ccc|ccc|ccc} 1 & & & 1 & & & 1 & & \\ & \ddots & & & \ddots & & & \ddots & \\ & & 1 & & & 1 & & & 1 \end{array}\right],$$

where $\mathbf{1}_{M\times N}$ is an $M\times N$ matrix of ones (a matrix where every element is equal to one), $\mathbf{O}_N$ is a zero matrix of order N, $\mathbf{I}_N$ is an identity $N\times N$ matrix and signs „⊗" and „⊕"denote tensor product and direct sum of two matrices, respectively[15, 16].

Fig. 1 shows a general structure of processing system for implementation of DFrFT for even N. The system contains 3 matrix-vector multiplier units, input bus, multiplexing bus, and summation unit. The dash-dotted line delineates a fragment of the system, which is responsible for combining and summing up partial products in expression (4), i.e. multiplication by the matrix $\mathbf{\Sigma}_{N\times 3N}$.

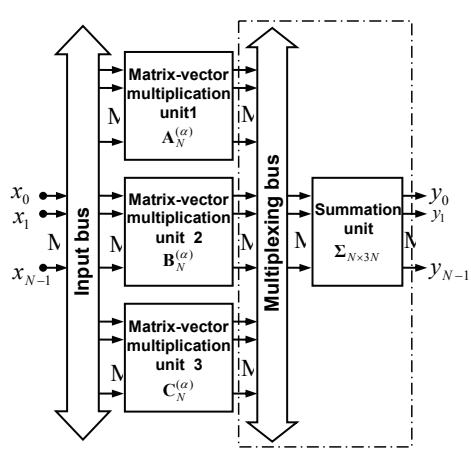

Fig. 1. General structure of DFrFT processing system for even length N

If N is even, the matrix $\mathbf{F}_N^{(\alpha)}$ can be written as a sum of two matrices

$$\mathbf{Y}_{N\times 1}^{(\alpha)} = \mathbf{F}_N^{(\alpha)}\mathbf{X}_{N\times 1} = \mathbf{A}_N^{(\alpha)}\mathbf{X}_{N\times 1} + \mathbf{B}_N^{(\alpha)}\mathbf{X}_{N\times 1} \tag{5}$$

where

$$\mathbf{A}_N^{(\alpha)} = \begin{bmatrix} f_{0,0}^{(\alpha)} & f_{0,1}^{(\alpha)} & \Lambda & f_{0,\frac{N-1}{2}}^{(\alpha)} & f_{0,\frac{N-1}{2}}^{(\alpha)} & \Lambda & f_{0,1}^{(\alpha)} \\ f_{0,1}^{(\alpha)} & 0 & \Lambda & 0 & 0 & \Lambda & 0 \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ f_{0,\frac{N-1}{2}}^{(\alpha)} & 0 & \Lambda & 0 & 0 & \Lambda & 0 \\ f_{0,\frac{N-1}{2}}^{(\alpha)} & 0 & \Lambda & 0 & 0 & \Lambda & 0 \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ f_{0,1}^{(\alpha)} & 0 & \Lambda & 0 & 0 & \Lambda & 0 \end{bmatrix},$$

$$\mathbf{B}_N^{(\alpha)} = \begin{bmatrix} 0 & 0 & \Lambda & 0 & 0 & \Lambda & 0 \\ 0 & f_{1,1}^{(\alpha)} & \Lambda & f_{1,\frac{N-2}{2}}^{(\alpha)} & f_{1,\frac{N+1}{2}}^{(\alpha)} & \Lambda & f_{1,N-1}^{(\alpha)} \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ 0 & f_{1,\frac{N-1}{2}}^{(\alpha)} & \Lambda & f_{\frac{N-1}{2},\frac{N-1}{2}}^{(\alpha)} & f_{\frac{N-1}{2},\frac{N+1}{2}}^{(\alpha)} & \Lambda & f_{1,\frac{N+1}{2}}^{(\alpha)} \\ 0 & f_{1,\frac{N+1}{2}}^{(\alpha)} & \Lambda & f_{\frac{N-1}{2},\frac{N+1}{2}}^{(\alpha)} & f_{\frac{N-1}{2},\frac{N-1}{2}}^{(\alpha)} & \Lambda & f_{1,\frac{N-1}{2}}^{(\alpha)} \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ 0 & f_{1,N-1}^{(\alpha)} & \Lambda & f_{1,\frac{N+1}{2}}^{(\alpha)} & f_{1,\frac{N-1}{2}}^{(\alpha)} & \Lambda & f_{1,1}^{(\alpha)} \end{bmatrix}$$

In this case, the parallelized process of calculating the transformation can be represented as follows:

$$\mathbf{Y}_{N\times 1}^{(\alpha)} = \mathbf{\Sigma}_{N\times 2N}\mathbf{W}_{2N}\mathbf{P}_{2N\times N}\mathbf{X}_{N\times 1} \tag{6}$$

where

$$\mathbf{P}_{2N\times N} = \mathbf{1}_{2\times 1}\otimes\mathbf{I}_N = \left[\begin{array}{ccc} 1 & & \\ & \ddots & \\ & & 1 \\ \hline 1 & & \\ & \ddots & \\ & & 1 \end{array}\right],\quad \mathbf{W}_{2N} = \begin{bmatrix} \mathbf{A}_N^{(\alpha)} & \mathbf{0}_N \\ \mathbf{0}_N & \mathbf{B}_N^{(\alpha)} \end{bmatrix},$$

$$\mathbf{\Sigma}_{N\times 2N} = \mathbf{1}_{1\times 2}\otimes\mathbf{I}_N = \left[\begin{array}{ccc|ccc} 1 & & & 1 & & \\ & \ddots & & & \ddots & \\ & & 1 & & & 1 \end{array}\right].$$

Fig. 2 shows a general structure of processing system for implementation of DFrFT for odd N. The system in this case contains only 2 matrix-vector multiplier units, input bus, multiplexing bus, and summation unit. The dash-dotted line delineates a fragment of the system, which is responsible for combining and summing up partial products in expression (6) i.e. multiplication by the matrix $\mathbf{\Sigma}_{N\times 2N}$.

Analysis of the structures of matrices $\mathbf{A}_N^{(\alpha)}$, $\mathbf{B}_N^{(\alpha)}$ and $\mathbf{C}_N^{(\alpha)}$ allows us to conclude that it is possible to reduce the number of arithmetic operations when calculating the matrix-vector products $\mathbf{A}_N^{(\alpha)}\mathbf{X}_{N\times 1}$, $\mathbf{B}_N^{(\alpha)}\mathbf{X}_{N\times 1}$ and $\mathbf{C}_N^{(\alpha)}\mathbf{X}_{N\times 1}$.

Thus, the proposed solutions make it possible to accelerate the implementation of the DFrFT not only by parallelizing computations, but also by reducing the number of required arithmetic operations.

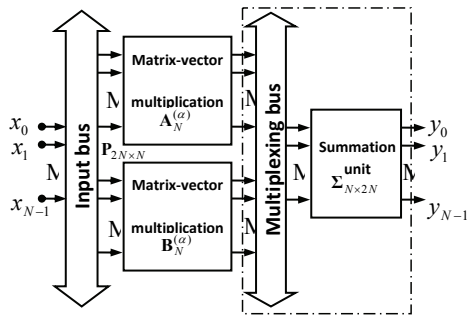

Fig. 2. General structure of DFrFT processing system for odd length N

In our previous work [12], we presented solutions leading to a reduction in the number of arithmetic operations in the calculation of the matrix-vector products under discussion. The corresponding results are presented in Table 1 and Table 2.

Tab. 1. Computation complexity of the proposed solution for even N

Operation type	$\mathbf{A}_N^{(\alpha)}\mathbf{X}_{N\times 1}$	$\mathbf{B}_N^{(\alpha)}\mathbf{X}_{N\times 1}$	$\mathbf{C}_N^{(\alpha)}\mathbf{X}_{N\times 1}$	Summing
additions	N-1	$N(N-2)/2$	N-2	$2N$-3
multiplications	N+1	$N(N-2)^2/2$	N-1	

Tab. 2. Computation complexity of the proposed solution for odd N

No.	$\mathbf{A}_N^{(\alpha)}\mathbf{X}_{N\times 1}$	$\mathbf{B}_N^{(\alpha)}\mathbf{X}_{N\times 1}$	Summing
additions	N-1	$(N-1)(N+1)/2$	N-1
multiplications	N	$(N-2)^2/2$	

4. Conclusions

In the paper, some aspects of the algorithmic and structural organization of the processor for computing of the fractional discrete Fourier transform were considered. It is shown that for processing even-length sequences, three matrix-vector multiplication units, two data buses, and one summing unit are necessary. To process sequences of odd length, only two three matrix-vector multiplication units, two data buses and one summing block are needed. The specific implementation of such a processor will depend on the chosen microelectronic platform. It can be a many-core VLSI, where the cores will calculate the corresponding vector-matrix products, or FPGA with embedded PC processor blocks which will be used as matrix-vector multipliers.

5. References

[1] Ozaktas H. M., Zalevsky Z. and Kutay M. A.: The Fractional Fourier Transform with Applications in Optics and Signal Processing. Wiley, Chichester, 2001.
[2] Ozaktas H.M., Ankan O., Kutay M.A., Bozdagi G.: Digital computation of the fractional Fourier transform. IEEE Trans. Signal Process. 1996, 44(9), pp. 2141-2150. doi:10.1109/78.536672.
[3] Pei S. C., Yeh M. H.: Discrete fractional Fourier transform. in: Proc. IEEE Internat. Symp. on Circuits and Systems, 1996, pp. 536-539.
[4] Jindal N., Singh K.: Image and video processing using discrete fractional transforms. Signal, Image and Video Processing 8(8), 1543-1553 (2014). doi:10.1007/s11760-012-0391-4.
[5] El-Mashed M.G., Zahran O., Dessouky M.I., El-Kordy M., Abd El-Samie F.E.: Synthetic aperture radar imaging with fractional Fourier transform and channel equalization, Digital Signal Processing, 2013, vol. 23, No 1, pp. 151-175.
[6] Djurovic I., Stankovic S., Pitas I.: Digital watermarking in the fractional Fourier transformation domain. J Network and Computer Appl, 2001, 24, pp. 167—173.
[7] Pei S. C., Ding J. J.: Closed-form discrete fractional and affine Fourier transforms. IEEE Trans. Signal Process, 2000, 48(5), pp. 1338-1353. doi:10.1109/78.839981.
[8] Bultheel A., Martinez-Sulbaran H.E.: Computation of the Fractional Fourier Transform. Applied and Computational Harmonic Analysis.2004, 16(3), pp. 182-202.
[9] Irfan M., Zheng M. L., Shahzad H.: Review of computing algorithms for discrete fractional Fourier transform 2013, 6(11), pp. 1911-1919.
[10] Tao R., Liang G., Zhao X.: An efficient FPGA-based implementation of fractional Fourier transform algorithm, J. Signal Processing Systems 60(1), 47-58 (2010). doi:10.1007/s11265-009-0401-0A.
[11] Prasad M. V. N. V., Ray K. C. and Dhar A. S.: FPGA implementation of Discrete Fractional Fourier Transform, in Proc. IEEE Int. Conf. Signal Processing and Communication (IEEE-SPCOM), Bangalore, 2010, pp. 1-5.
[12] Ray K. Ch., Prasad M. V. N. V., Dhar A. S.: An Efficient VLSI Architecture for Computation of Discrete Fractional Fourier Transform. Journal of Signal Processing Systems. 2017. pp. 1–12.
[13] Majorkowska-Mech D., Cariow A.: A Low-Complexity Approach to Computation of the Discrete Fractional Fourier Transform, Circuits, Systems, and Signal Processing,2017, 36 (10), pp. 4118-4144.
[14] Majorkowska-Mech D., Cariow A.: An algorithm for computing the "true" discrete fractional Fourier transform. Advances in Soft and Hard Computing. 2018(in print).
[15] Cariow A.: Strategies for the Synthesis of Fast Algorithms for the Computation of the Matrix-vector Products. Journal of Signal Processing Theory and Applications 3(1), 1-19 (2014). doi:10.7726/jspta.2014.1001.
[16] Granata J., Conner M., Tolimieri R.: The Tensor product: A mathematical programming language for FFTs and other fast DSP operations., IEEE Signal Processing Magazine, 1992. No 1, pp.41-48.
[17] Regaliat P. A. and Mitra S. K.: Kronecker Products, Unitary Matrices and Signal Processing Applications, Siam Review, 1989, 31, No. 4, pp. 586-613.

Received: 10.12.2017 *Paper reviewed* *Accepted: 01.02.2018*

Prof. Aleksandr CARIOW, DSc, PhD

He received the Candidate of Sciences (PhD) and Doctor of Sciences degree (DSc or Habilitation) in Computer Sciences from LITMO of St. Petersburg, Russia in 1984 and 2001, respectively. In September 1999, he joined the faculty of Computer Sciences and Information Technology at the West Pomeranian University of Technology, Szczecin, Poland, where he is currently a professor in the Department of Computer Architectures and Teleinformatics. His research interests include digital signal processing algorithms, VLSI architectures, and data processing parallelization.

e-mail: acariow@wi.zut.edu.pl

Dorota MAJORKOWSKA-MECH, PhD

Dorota Majorkowska-Mech received the MA degree in physics in 1994, and the MA degree in mathematics in 1996 from the University of Szczecin. In 2008 she obtained the PhD in the field of computer science at West Pomeranian University of Technology in Szczecin, where she currently works as an assistant-professor in the Department of Computer Architectures and Teleinformatics. Dr. Majorkowska-Mech is interested in using discrete orthogonal transforms in digital signal processing.

e-mail: dmajorkowska@wi.zut.edu.pl