JOURNAL OF APPLIED
COMPUTER SCIENCE
Vol. 21 No. 2 (2013), pp. 97-115

Image Filtering with Fast Parametrized Biorthogonal Transforms Implemented on a New GUI Research Aid System

**Dariusz Puchala, Bartłomiej Stasiak,
Kamil Stokfiszewski, Mykhaylo Yatsymirskyy**

*Institute of Information Technology
Łódź University of Technology
Wólczańska 215, 90-924 Łódź*

*bartlomiej.stasiak@p.lodz.pl,
dariusz.puchala@p.lodz.pl,
kamil.stokfiszewski@p.lodz.pl,
mykhaylo.yatsymirskyy@p.lodz.pl*

Abstract. *In this paper the authors show that fast parametrized biorthogonal transforms (FPBT) are well suited for adaptive generalized Wiener image filtering. Research results are obtained with a use of a new graphical user interface system for implementing various fast adaptive techniques, designed, implemented and published by the authors as a part of a project Innovative Economy Programme 2007-2013 „Platforma Informatyczna TEWI".*
Keywords: *biorthogonal transforms, fast algorithms, image filtering, image processing applications, graphical user interface.*

1. Introduction

Discrete biorthogonal transforms [1, 2] are a generalization of well known discrete orthogonal transforms [3, 4], which have been widely used in two-dimensional signal filtering, especially in image compression [5, 6, 7], image processing

and recognition [8, 9, 10] as well as in image watermarking algorithms [11]. The aforementioned generalization comes from the fact that in case of a pair of linear biorthogonal transforms **A** and **B** the only restriction imposed on the analysis and the synthesis transform matrices is that they have to be invertible, i.e. $det\,(\mathbf{A}) \neq 0$ and $det\,(\mathbf{B}) \neq 0$. The motivation for using such transforms, instead of orthogonal ones, is that it has been shown, both theoretically and practically, that for many signal filtering problems their optimal solutions are not the ones involving orthogonal matrix pairs. One example of such non-orthogonal optimal solutions is a generalized Wiener filtering problem [12], which will be discussed in the next section. The second example involves the problem of finding optimal pair of coding and decoding matrices in the task of transform image compression with scalar quantization. This problem is of great practical significance, since such image compression scheme is widely used in many of the contemporary image compression algorithms such as a JPEG method [13]. It has been shown theoretically in [14] that at least one of the optimal pair of coding and decoding matrices $\mathbf{A}_{opt}$ and $\mathbf{B}_{opt}$ is not orthogonal and that for the overall optimal transfer matrix $\mathbf{A}_{opt} \cdot \mathbf{B}_{opt} \neq \mathbf{I}$ holds, where **I** denotes the respective identity matrix. The above paragraph explains briefly only the first of the premises for biorthogonal transforms to be an interesting alternative to the orthogonal ones.

The second motivation for the one to be interested in using biorthogonal transforms is that the earlier work of the authors has been conducted to construct fast parametrized orthogonal transform adaptation algorithms, which lately have been generalized by the authors to biorthogonal cases, in many signal processing tasks, e.g. [2, 6, 9, 10, 15]. Those adaptation techniques developed by the authors in their earlier works [7, 16], use fast computational structures for calculation of various known discrete linear transforms (e.g. discrete sine, cosine, Fourier or Hartley transforms) which allow to obtain computational complexity of order $O(N log_2 N)$, while maintaining the potential to adapt to solutions, which are close to optimal ones.

Taking into consideration the above arguments in this paper the authors consider a problem of automatic adaptation of the biorthogonal parametrized transforms to a task of image filtering resulting in the reduction of noise level in natural images, i.e. generalized Wiener filtering. For this purpose the authors use the graphical user interface (GUI) system for designing and performing experiments in adaptive signal/image processing which has been originally developed and published by the authors as a part of the project *Innovative Economy Programme 2007-2013 „Platforma Informatyczna TEWI"*. Thus the paper is organized as fol-

lows: in the first part the problem of image filtering with the use of biorthogonal parametrized transforms is considered, the experiments are conducted and described in detail, finally the results are presented. In the second part of the article a brief but comprehensive description of the GUI research aid system, which has been designed by the authors and used during experiments, is described regarding it's most general functionality and capabilities. In the last section summary and conclusions are presented.

2. Image filtering with fast parametrized biorthogonal transforms

2.1. Generalized Wiener filtering for two-dimensional signals

In this paper authors use generalized Wiener filtering, formulated in [12], as the starting point for the adaptation of parametrized biorthogonal transforms of two-dimesional signals. A generalized Wiener filtering scheme for one-dimesional signals is presented in Fig. 1.

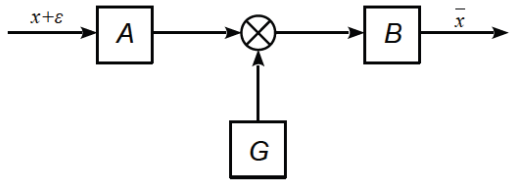

Figure 1. Generalized Wiener filtering scheme for one-dimesional signals

Here an input to the system consists of a sum of the N-element zero-mean signal vector $\mathbf{x}$ and the N-element zero-mean noise vector $\mathbf{e}$. It's assumed that the signal and noise vectors $\mathbf{x}$ and $\mathbf{e}$ are mutually uncorrelated. First, the input signal $\mathbf{x}+\mathbf{e}$ is multiplied by the $N \times N$ - element matrix $\mathbf{A}$ to the scheme's analysis transform domain represented by this matrix. In the second step the transformed signal is multiplied by the $N \times N$ - element filtering matrix $\mathbf{G}$, which performs the actual filtering. In the last step the filtered signal is transformed back to the time domain with the use of $N \times N$ - element matrix $\mathbf{B}$ representing the scheme's sythesis transform. This results in an N-element output signal $\bar{\mathbf{x}}$ which takes the following form $\bar{\mathbf{x}} = \mathbf{BGA}(\mathbf{x}+\mathbf{e})$. The aim of the filtering process is to find optimal forms of all of

the three matrices **A**, **B** and **G**, i.e. the analysis transform, the synthesis transform and the filtering matrix, respectively, such that the mean-square error between the original signal **x** and the output signal $\bar{\mathbf{x}}$ is minimized. Thus the problem can be formally stated as follows

$$\underset{\mathbf{A},\mathbf{B},\mathbf{G}}{argmin} \left\{ E \{ tr [(\mathbf{x} - \bar{\mathbf{x}}) (\mathbf{x} - \bar{\mathbf{x}})^T] \} \right\} \tag{1}$$

where $E\{\cdot\}$ denotes expectation operator, $tr[\cdot]$ stands for a matrix trace operator and $(\cdot)^T$ denotes matrix transposition. It has been shown in [12] that the solution of the problem (1) is obtained by any triple of matrices **A**, **B** and **G**, which obey the following necessary and sufficient condition

$$\mathbf{B}\,\mathbf{G}\,\mathbf{A} = \mathbf{R}_x (\mathbf{R}_x + \mathbf{R}_e)^{-1} \tag{2}$$

where $\mathbf{R}_x$ and $\mathbf{R}_e$ are the $N \times N$ autocorrelation matrices of the original signal **x** and noise signal **e**, respectively.

Generalization of the above results to two-dimensional case, i.e. the images, is straightforward. For this purpose one can use a matrix column stack operator $vec(\cdot)$, which takes an $N \times N$ - element matrix and converts it into an N^2 element column vector by stacking it's columns. For a two-dimensional $N \times N$, zero-mean original signal **X** and the respective noise signal **E** one can make a formal substitution in (1), which can be written as follows

$$\mathbf{x} = vec(\mathbf{X}) \quad \text{and} \quad \bar{\mathbf{x}} = \mathbf{B}\,\mathbf{G}\,\mathbf{A}\,(vec(\mathbf{X}) + vec(\mathbf{E})) \tag{3}$$

where, this time, the respective Wiener filtering matrices **A**, **B** and **G** are $N^2 \times N^2$ element linear transforms. Finally, making a following formal substitution in (2)

$$\mathbf{R}_x = E\{vec(\mathbf{X})\, vec(\mathbf{X})^T\} \quad \text{and} \quad \mathbf{R}_e = E\{vec(\mathbf{E})\, vec(\mathbf{E})^T\} \tag{4}$$

one obtains the desired conversion form two-dimensional to one-dimensional signal case. Having performed the above conversion all assumptions, considerations and results obtained for one-dimensional generalized Wiener filtering scheme also apply for the two-dimensional case.

Finally, it should be noted that in general case from the solution (2) of the generalized Wiener filtering problem it follows that at least one of the filtering matrices, either **A**, **B** or **G** has to represent a non-orthogonal transformation, which on behalf of (3) and (4) also holds for two-dimensional case.

2.2. Fast parametrized biorthogonal transforms

Fast parametrized biorthogonal transforms (FPBT) were introduced by two of the authors in paper [2]. In this paragraph main concepts and a brief introduction to fast parametrized biorthogonal transforms will be presented. As mentioned in [2] parametrized transforms, unlike transforms with fixed base vectors, have an advantage of the ability to adapt to statistical characteristics of signals. The construction of fast parametrized biorthogonal transforms relies on two facts. Firstly, there exist one and two-dimensional structurally unified fast algorithms of order $O(N log_2 N)$ for computation of many known fixed base discrete transforms with the use of a single unified structures, (see e.g. [16, 17, 18]). Secondly, many types of base operations along with respective adaptation rules for fast parametrized transforms can be constructed to reflect both – the particular restrictions on the transform itself (e.g. orthogonality [15] or biorthogonality [2]) and an adaptation goal function forms (see e.g. [7, 19]) – for a specified transform used in a particular application. For the purposes of this article authors have chosen to use a two-stage Walsh-Hadamard like transform structure, whose analysis part for the two-dimensional signal case is depicted in Fig. 2.

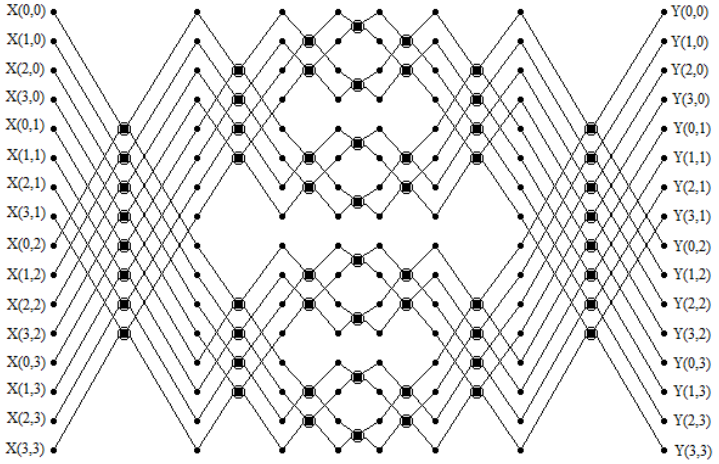

Figure 2. Two-stage Walsh-Hadamard like transform graph for 2D signals

Each butterfly operator „•” shown in Fig. 2 is schematically depicted in Fig. 3.

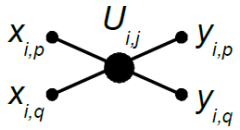

Figure 3. Single butterfly operator of fast parametrized transform structure

Here $\mathbf{U}_{i,j}$ denotes 2×2 matrix of the j-th butterfly operator inside a given layer of the structure from Fig. 2, i stands for a layer number, while $x_{i,p}$, $x_{i,q}$ and $y_{i,p}$, $y_{i,q}$ are the inputs and the outputs of the operator respectively, where p and q denote connection indices. Among many possible operator models (see section 4) for the purpose of this article authors have chosen the operator type with four parameters whose matrix along with it's operation are defined by the following equations

$$\mathbf{U}_{i,j} = \begin{bmatrix} a_{i,j} & b_{i,j} \\ c_{i,j} & d_{i,j} \end{bmatrix} \quad , \quad \begin{bmatrix} y_{i,p} \\ y_{i,q} \end{bmatrix} = \mathbf{U}_{i,j} \begin{bmatrix} x_{i,p} \\ x_{i,q} \end{bmatrix} \quad . \tag{5}$$

It can be easily verified [2], that for the above operator definition and transform structure shown in Fig. 2 number of additions $\mathcal{N}_+$ and multiplications $\mathcal{N}_*$ needed to calculate the transform's outputs are equal to

$$\mathcal{N}_+ = 2\,N\,(\,log_2 N - 0.5\,) \quad , \quad \mathcal{N}_* = 4\,N\,(\,log_2 N - 0.5\,) \tag{6}$$

where N denotes the total number of inputs/outputs of the structure depicted in Fig. 2. The values of $\mathcal{N}_+$ and $\mathcal{N}_*$ given in (6) imply immediately that the considered transform is fast in terms of computational complexity. It can also be easily verified [6, 15] that for the steepest descent adaptation technique of the considered parametrized transform and the mean-square-error (MSE) between the obtained and the demanded outputs of the transform taken as the adaptation criterion, the local adaptation rule for a single butterfly operator takes the form

$$\hat{z}_{i,j} = z_{i,j} + \eta \frac{\partial}{\partial z_{i,j}} \left(\begin{bmatrix} y_{i,p} \\ y_{i,q} \end{bmatrix}^T \mathbf{U}_{i,j} \begin{bmatrix} x_{i,p} \\ x_{i,q} \end{bmatrix} \right) \tag{7}$$

where $z_{i,j}$ stands for an appropriate parameter of the $\mathbf{U}_{i,j}$ butterfly operator, $\hat{z}_{i,j}$ is the parameter's new value and $\eta > 0$ is the adaptation step. Form the above assumptions it follows immediately that in order to adapt the transform's parameters one may use a standard error-backpropagation method (see e.g. [7]) with the local adaptation rule (7) for each of the butterfly operators.

3. Experimental results

In order to verify the effectiveness of FPBT several experiments were conducted regarding generalized Wiener filtering, described in paragraph 2.1, for natural images. The complete data flow graph used during adaptation and latter testing of the obtained FPBT's was constructed out of two structures. Both structures were identical to that depicted in Fig. 2 and corresponded to the analysis and synthesis transforms i.e. matrices **A** and **B** in Fig. 1, respectively. For a particular experiment setup that was used there was no need to optimize the filtering matrix **G** form Fig. 1 separately. Despite the lack of explicit reference to the matrix **G** in the adaptation scheme, the results still maintained the ability of being potentially optimal, since for biorthogonal transform pair **A** and **B** the considered filtering matrix **G** could successfully be incorporated into one or both of the mentioned transforms during the adaptation process. The data flow diagram of the structure used during the adaptation step is depicted in Fig. 4.

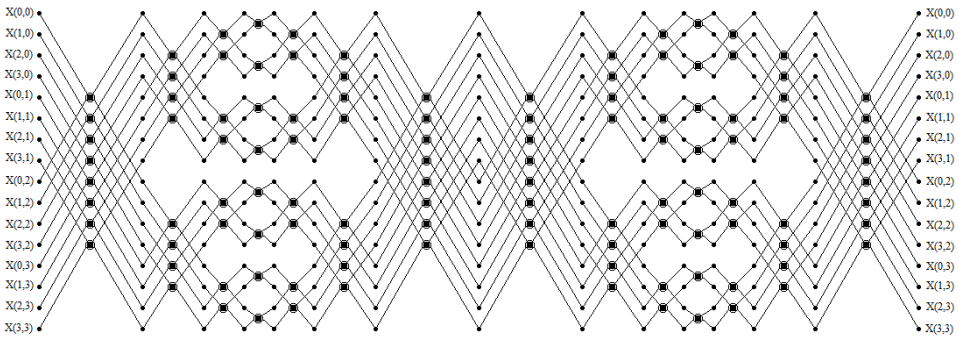

Figure 4. FPBT's data flow graph used during experiments

During adaptation process 512×512 - pixel, 8 - bit gray-scale natural images were used, which are shown in Fig 5. For all of three variants of the experiment the input training patterns consisted of the `Lena` image, partitioned into 8×8 pixel blocks and corrupted (in the sense of addition) with white Gaussian noise, for which the standard deviations σ_k, $k = 1, 2, 3$ were equal to $30, 20$ and 15, respectively. The pattern demanded at the output of the optimized structure was the respective 8×8 - pixel block of the original `Lena` image. A standard steepest gradient descend method was invoked with formula (7) used for adaptation of the

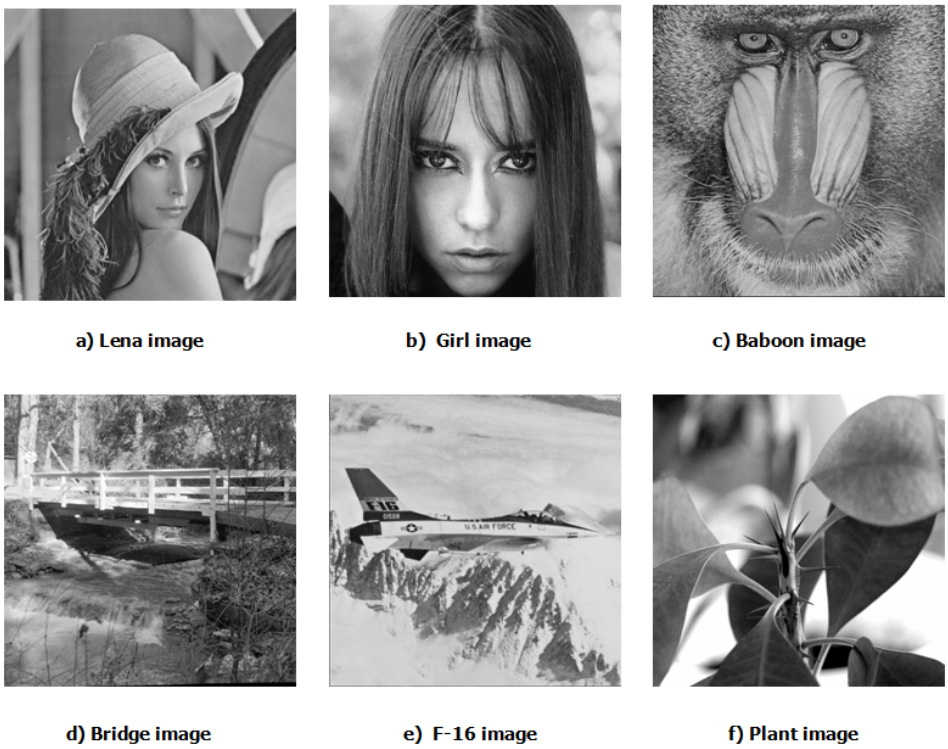

Figure 5. Training image a) and test images a), b), c), d) and f)

transforms' parameters and the mean-square-error measure (MSE) between the obtained and the demanded output was used as an optimization criterion. During tests all of the gathered images were used, including the `Lena` image which, as the only image among all others, served as a source to generate training patterns during adaptation process. Tests were held by inputting all test images' 8×8 pixel fragments to the two-dimensional generalized Wiener filtering scheme with the adapted earlier fast biorthogonal transforms. The results obtained with FPBTs were compared against results obtained with optimal filtering scheme [12] taking the advantage of signal dependent Karhunen-Loève transform (KLT) which was computed for the training (`Lena`) image only. The results, given in terms of the PSNR measure according to formula (8), calculated for the corrupted/filtered

images and their original counterparts are presented below in Tab. 1, 2 and 3.

$$\begin{aligned} \mathrm{PSNR} &= 10 \cdot \log_{10} \frac{255^2}{\mathrm{MSE}}, \\ \mathrm{MSE} &= \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \left(f_{ij} - \hat{f}_{ij} \right)^2 . \end{aligned} \tag{8}$$

In definition (8) N stands for horizontal/vertical experimental images' dimensions while f_{ij} and $\hat{f}_{ij}$ denote the original and the corrupted/filtered image's pixels respectively.

Table 1. Results of the 1-st experiment, Gaussian noise standard deviation $\sigma_1 = 30$

PSNR [db] / Image	**Lena**	**Girl**	**Baboon**	**Bridge**	**F-16**	**Plant**
NOISED	19.07	19.28	19.04	19.17	19.23	19.42
FBPT	27.46	26.75	21.27	22.92	25.55	27.80
KLT	27.97	27.25	21.80	23.60	26.28	28.20

Table 2. Results of the 2-nd experiment, Gaussian noise standard deviation $\sigma_2 = 20$

PSNR [db] / Image	**Lena**	**Girl**	**Baboon**	**Bridge**	**F-16**	**Plant**
NOISED	22.01	22.16	22.02	22.10	22.05	22.33
FBPT	28.89	27.99	21.94	23.87	26.87	29.19
KLT	29.50	28.56	22.71	24.78	27.88	29.74

Table 3. Results of the 3-rd experiment, Gaussian noise standard deviation $\sigma_3 = 15$

PSNR [db] / Image	**Lena**	**Girl**	**Baboon**	**Bridge**	**F-16**	**Plant**
NOISED	25.02	25.07	25.01	25.06	25.00	25.26
FBPT	30.62	29.53	23.06	25.26	28.70	30.85
KLT	31.15	30.16	23.75	26.09	29.61	31.34

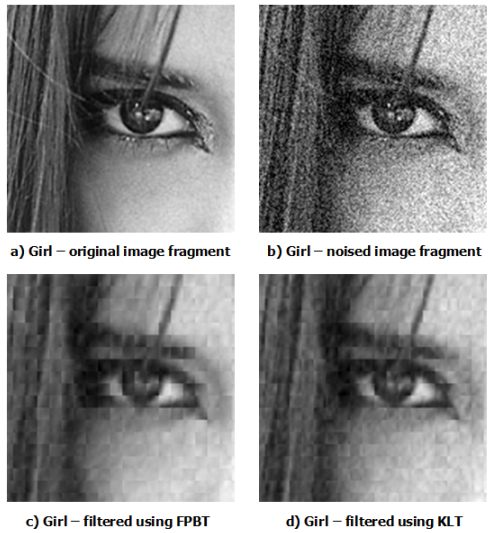

Figure 6. Girl image fragments - original, noised and filtered, $\sigma = 30$

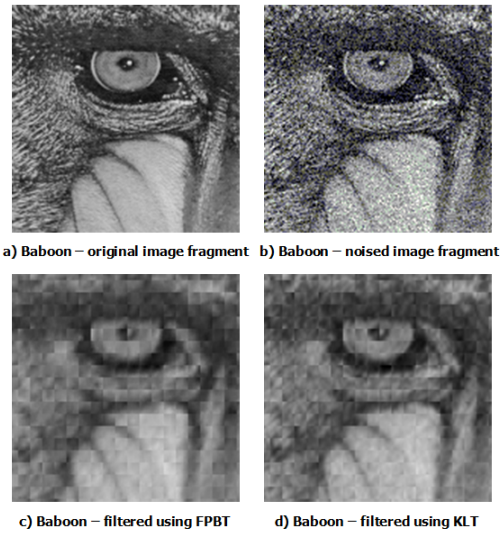

Figure 7. Baboon image fragments - original, noised and filtered, $\sigma = 30$

In Fig. 6 and Fig. 7 selected testing images' fragments are shown. The original image fragments are presented in a) subfigures, image fragments corrupted with Gaussian white noise with standard deviation $\sigma_1 = 30$ in b) subfigures and fragments filtered with the use of adapted FPBTs and KLTs in c) and d) subfigures respectively. From the results gathered in Tab. 1, 2 and 3 one can clearly see that the PSNR values of the images filtered with the use of FPBTs are close to those obtained in the Wiener filtering scheme with the use of Karhunen-Loève transform. Secondly, it's worth noting that for all noise levels the biggest absolute PSNR values for images filtered with the use the FPBTs and the KLTs are obtained for the `Lena` images and `Plant` images, which comes as no surprise in both cases since the first of the mentioned images was used as the training pattern for both of the compared transforms and the second of the mentioned images clearly has the closest statistical characteristics to the one possessed by the `Lena` training image. It's also has to be noted that blocking artifacts arise after the filtering process in both cases i.e. for Karhunen-Loève transforms and the adapted fast parametrized biorthogonal transforms, which is a consequence of the fact that the both transform types were adapted with the use of 8×8 – pixel blocks.

4. A new GUI research aid system

In this section authors describe the most general functionality and capabilities of the new GUI research aid system, which has been designed by the authors and used during the conducted experiments. The system was created and published by the authors as a part of the project *Innovative Economy Programme 2007-2013 „Platforma Informatyczna TEWI"*.

The graphical user interface (GUI) research aid system was created for the purpose of visual designing and automating the conduction of the experiments involving general fast parametric transforms including biorthogonal ones.

4.1. System structure

The presented system has been implemented in `C++` language under *Linux* operating system with the use of the standard `gcc` compiler and can be easily ported to other operating systems such as *Windows* OS family. The whole project is divided into two parts. The first part consists of basic tools that enable creation, adaptation and testing of the various types of fast parametrized transforms, Applications which are contained in this part are all command line tools that enable

flexible setup of the the conducted experiments based on the command line options. Main tools available to the user in the first part of the system are

- `test_FonNet` – a basic module for adaptation and tests performed on the adapted fast parametrized transforms
- `test_FonNetGenerator` – a basic module for generation of the transform structures

The second part of the system contains GUI tools that encapsulate the functionality of the command line applications and enable the user to visually create, design and control the conduction of the experiments regarding general fast parametric transforms. Main tools available to the user in the second part of the system are

- `ButterflySimualtor` – a GUI application that enables visual creation of complex data processing diagrams, consisting of the functional blocks, which include visual creation of data flow graphs of various fast parametrized transforms
- `testWxProperties` – a GUI application for easy editing the parameters of the functional blocks of the mentioned data processing diagrams created in `ButterflySimualtor`

For the `ButterflySimualtor` and the `testWxProperties` applications selected views of their main windows are shown in Fig. 8 and Fig. 9.

From the fast parametrized transforms researcher's point of view, among many of the system's capabilities, one of the major advantages is the ability to construct and visually manipulate data flow graphs of fast parametrized transforms (FPTs) of various kinds. The user has to his/her disposal a whole collection of possible butterfly operators and processing units of other types with the ability to easily add his or her new operator/processing units types. A list of already implemented operators and processing unit types is comprehensive for most of the research tasks, the mentioned list is depicted in Fig. 10. Since a whole description of the presented system capabilities is beyond the scope of this paper, the authors have decided to show a part of an exemplary application showing the process of creation of the connection scheme for two-stage 8-point fast parametrized biorthogonal transform based on BOON (Basic Operation Orthogonal Neuron) operators [20].

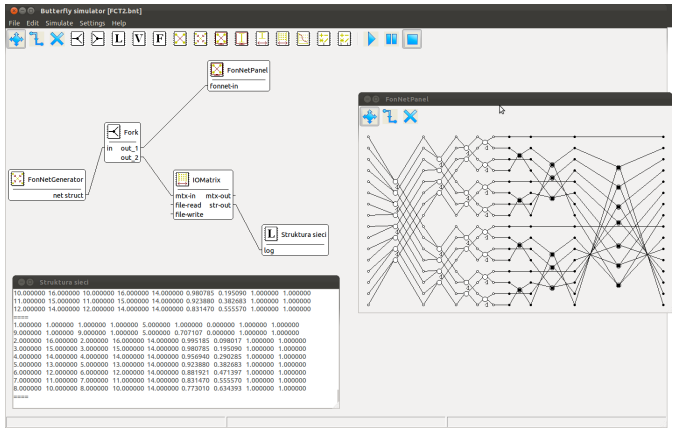

Figure 8. `ButterflySimualtor` - selected view of the main window

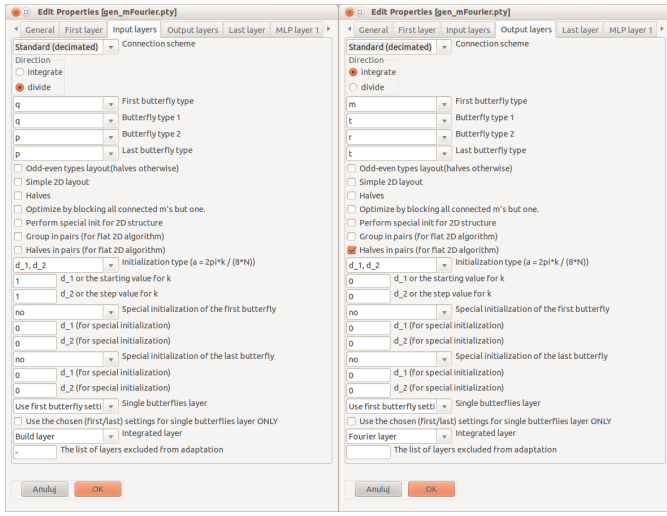

Figure 9. `testWxProperties` - selected view of the main window

4.2. Exemplary application

In this paragraph an exemplary application will be presented briefly describing the process of creation of the connection scheme for two-stage 8-point fast parametrized biorthogonal transform based on BOON butterfly operators [20].

Operation type (number)	Description	Symbol
`c` (0)	Connection between layers	
`p` (1)	Orthogonal butterfly with two weights: $\begin{bmatrix} u & w \\ -w & u \end{bmatrix}$	
`q` (2)	Orthogonal butterfly with two weights: $\begin{bmatrix} u & w \\ w & -u \end{bmatrix}$	
`t` (3)	Orthogonal butterfly with one weight: $\begin{bmatrix} 1 & t \\ -t & 1 \end{bmatrix}$ (for algorithms with tangent multipliers)	
`r` (4)	Orthogonal butterfly with one weight: $\begin{bmatrix} t & 1 \\ 1 & -t \end{bmatrix}$ (for algorithms with tangent multipliers)	
`m` (5)	Multiplication (single weight)	
`b` (6)	Multiplication (single weight) without adaptation	
`s` (7)	Multiplication + bias (two weights)	

Operation type (number)	Description	Symbol
`a` (8)	Multiplication + bias (two weights) + non-linear activation function	
`i` (9)	Bias + non-linear activation function	
`n` (10)	Input vector norm: square root of the sum of squares of both inputs (Warning: one output)	
`x` (11)	Input vector norm: square root of the sum of squares of both inputs + the second output ("phase" information)	
`d` (12)	Orthogonal butterfly with one weight: $\begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix}$	
`e` (13)	Orthogonal butterfly with one weight: $\begin{bmatrix} \cos\alpha & \sin\alpha \\ \sin\alpha & -\cos\alpha \end{bmatrix}$	
`f` (14)	Full matrix 2×2 (4 weights): $\begin{bmatrix} u & w \\ v & z \end{bmatrix}$	

Figure 10. A list of available butterfly operators and processing units of other types

Biorthogonal transform consists of L layers. Each layer is represented by a matrix defining the connection scheme in this layer. A single row of the matrix refers to a single butterfly of BOON type and it contains 9 values:

```
in1, in2, out1, out2, type, w1, w2, w3, w4;
```

where

- `in1` is the index of the first input,
- `in2` is the index of the second input,

– `out1` is the index of the first output,

– `out2` is the index of the second output,

– `type` is the butterfly operation type,

– `w1, w2, w3, w4` are connection parameters.

The fifth field (`type`) defines the butterfly operation type according to the types shown in Fig. 10. Dependent on the operation type, some of the other fields may be unused. For example, the connection of type `m` requires only the fields: `in1, out1` and `w1`; the values of the remaining fields are ignored. It should also be noted that all the four parameters `w1` – `w4` are necessary only in the case of the last operation type `(f)`. The sequence of L matrices, representing the consecutive layers of the parametrized transform is written to text files (with repeated `'='` sign as matrix separator). Exemplary text file is given below

```
/*-------------------------------------------------------------------------*/
  Exemplary configuration text file for transform structure
  based on 8-point discrete cosine transform of type II (FCT2)
/*-------------------------------------------------------------------------*/

====
0.00000
0.00000
0.00000
1.00000
1.00000
1.00000
====
1.00000 2.00000 1.00000 5.00000 2.00000 1.00000 1.00000 1.00000 1.00000
3.00000 4.00000 2.00000 6.00000 1.00000 1.00000 1.00000 1.00000 1.00000
5.00000 6.00000 3.00000 7.00000 2.00000 1.00000 1.00000 1.00000 1.00000
7.00000 8.00000 4.00000 8.00000 1.00000 1.00000 1.00000 1.00000 1.00000
====
1.00000 2.00000 1.00000 3.00000 2.00000 1.00000 1.00000 1.00000 1.00000
3.00000 4.00000 2.00000 4.00000 1.00000 1.00000 1.00000 1.00000 1.00000
5.00000 6.00000 5.00000 7.00000 2.00000 1.00000 1.00000 1.00000 1.00000
7.00000 8.00000 6.00000 8.00000 1.00000 1.00000 1.00000 1.00000 1.00000
====
1.00000 2.00000 1.00000 2.00000 2.00000 1.00000 1.00000 1.00000 1.00000
3.00000 4.00000 3.00000 4.00000 2.00000 1.00000 1.00000 1.00000 1.00000
5.00000 6.00000 5.00000 6.00000 2.00000 1.00000 1.00000 1.00000 1.00000
7.00000 8.00000 7.00000 8.00000 2.00000 1.00000 1.00000 1.00000 1.00000
====
1.00000 1.00000 1.00000 1.00000 5.00000 1.00000 0.00000 1.00000 1.00000
2.00000 1.00000 2.00000 1.00000 5.00000 0.70710 0.00000 1.00000 1.00000
3.00000 3.00000 3.00000 3.00000 5.00000 1.00000 0.00000 1.00000 1.00000
4.00000 3.00000 4.00000 3.00000 5.00000 0.70710 0.00000 1.00000 1.00000
5.00000 5.00000 5.00000 5.00000 5.00000 1.00000 0.00000 1.00000 1.00000
```

```
6.00000 5.00000 6.00000 5.00000 5.00000 0.70710 0.00000 1.00000 1.00000
7.00000 7.00000 7.00000 7.00000 5.00000 1.00000 0.00000 1.00000 1.00000
8.00000 7.00000 8.00000 7.00000 5.00000 0.70710 0.00000 1.00000 1.00000
====
1.00000 1.00000 1.00000 1.00000 5.00000 1.00000 0.00000 1.00000 1.00000
3.00000 1.00000 3.00000 1.00000 5.00000 0.70710 0.00000 1.00000 1.00000
2.00000 4.00000 2.00000 4.00000 1.00000 0.92387 0.38268 1.00000 1.00000
5.00000 5.00000 5.00000 5.00000 5.00000 1.00000 0.00000 1.00000 1.00000
7.00000 5.00000 7.00000 5.00000 5.00000 0.70710 0.00000 1.00000 1.00000
6.00000 8.00000 6.00000 8.00000 1.00000 0.92387 0.38268 1.00000 1.00000
====
1.00000 1.00000 1.00000 1.00000 5.00000 1.00000 0.00000 1.00000 1.00000
5.00000 1.00000 5.00000 1.00000 5.00000 0.70710 0.00000 1.00000 1.00000
2.00000 8.00000 2.00000 8.00000 1.00000 0.98078 0.19509 1.00000 1.00000
3.00000 7.00000 3.00000 7.00000 1.00000 0.92387 0.38268 1.00000 1.00000
4.00000 6.00000 4.00000 6.00000 1.00000 0.83146 0.55557 1.00000 1.00000
```

At the beginning of the file a binary column vector is written (a matrix of size $L \times 1$), each element of which stores information if a given layer can be adapted during the training stage (`1`) or not (`0`). An example of a transform structure based on 8-point cosine transform of type II (FCT2) is presented below in Fig. 11.

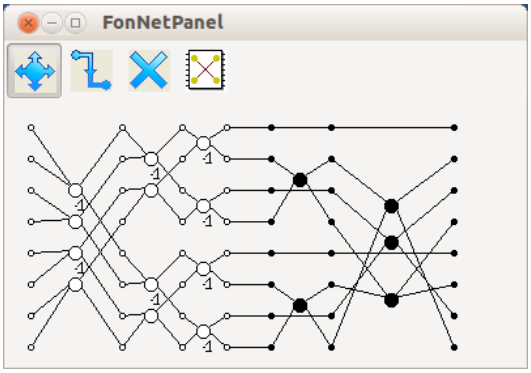

Figure 11. An example of the network based on two-stage FCT2 transform

The structure is based on a scheme of two-stage algorithm, where the first stage is not subjected to adaptation. Due to this fact the symbols of the operations in the first three layers are displayed as „empty". It should be also noted that the butterflies of type 1 and 2 (`p` and `q`) have two inputs; therefore the matrices representing the first three layers have 4 rows each. The next layer contains single connections only (of type 5: `m`) so its matrix has 8 rows. In the case of the two last layers an intermediate situation occurs.

To sum up, a brief exemplary application shows only a fraction of the functionality and flexibility of the implemented system. For an interested reader it's worth mentioning that a full documentation of the presented GUI research aid system is available on the „*Platforma Informatyczna TEWI*" user service located at `http://tewi.p.lodz.pl/Windchill`.

5. Conclusions

In this paper authors presented results of applying fast parametrized biorthogonal transforms to the Wiener image filtering scheme and compared them with the Karhunen-Loève transform, which is less efficient in terms of computational complexity than FPBTs. The obtained results show that the fast parametrized biorthogonal transforms are well suited for performing image filtering task in the sense of popular objective measures for classes of images with similar statistical characteristics. In the second part of the paper authors presented a brief description of a new GUI research aid system, which was implemented by the authors as a part „*Platforma Informatyczna TEWI*" project and used for evaluation of the obtained experimental results. The system was designed for the general purposes of digital signal processing realized with aid of fast parametric transforms, including biorthogonal ones, and has proved to be a very elastic, functional and helpful tool in performing digital signal processing experiments.

References

[1] Yatsymirskyy, M., *A Novel Matrix Model of Two Channel Biorthogonal Filter Banks*, Metody Informatyki Stosowanej, 2011, pp. 205–212, (in Polish).

[2] Yatsymirskyy, M. and Puchala, D., *Fast Parametrized Biorthogonal Transforms*, Przegląd Elektrotechniczny, Vol. 4, 2012, pp. 123–125.

[3] Ahmed, N. and Rao, K. R., *Orthogonal Tranforms for Digital Signal Processing*, Springer-Verlag, Berlin, Heidelberg, New York, 1975.

[4] Strang, G. and Nguyen, T., *Wavelets and filter banks*, Wellesley-Cambridge Press, 1999.

[5] Jayant, N. S. and Noll, P., *Digital Coding of Waveforms*, Prentice Hall, Engelwood Cliffs, NJ, 1984.

[6] Yatsymirskyy, M. and Puchala, D., *Fast Neural Networks Learning Techniques for Signal Compression*, Przegląd Elektrotechniczny, Vol. 1, 2010, pp. 189–191.

[7] Stokfiszewski, K. and Szczepaniak, P. S., *An Adaptive Fast Transform Based Image Compression*, Artificial Intelligence and Soft Computing - ICAISC, Lecture Notes in Computer Science, Vol. 5097, 2008, pp. 874–88.

[8] Zieliński, T., *From theory to digital signal processing*, WKŁ, 2009, (in Polish).

[9] Stasiak, B., *Two-dimensional Fast Orthogonal Neural Network for Image Recognition*, Lecture Notes in Computer Science, 2009, pp. 653–660.

[10] Stasiak, B. and Yatsymirskyy, M., *On Feature Extraction Capabilities of Fast Orthogonal Neural Networks*, Lecture Notes in Computer Science, 2007, pp. 27–36.

[11] Lipiński, P., *Watermarking software in practical applications*, Bulletin of the Polish Academy of Sciences: Technical Sciences, Vol. 59, No. 1, 2011, pp. 21–25.

[12] Pratt, W. K., *Generalized Wiener Filtering Computation Techniques*, IEEE Transactions on Computers, Vol. C-21, No. 7, 1972, pp. 636–641.

[13] *Digital compression and coding of continuous-tone still images*, Tech. Rep. T.81, International Telecommunication Union, 1992.

[14] Diamantaras, K. I. and Strintzis, M. G., *Optimal Transform Coding in the Presence of Quantization Noise*, IEEE Transactions on Image Processing, Vol. 8, No. 11, 1999, pp. 1508–1515.

[15] Yatsymirskyy, M. and Stasiak, B., *Fast orthogonal neural networks for 2D signal processing*, Przegląd Elektrotechniczny, Vol. 2, 2007, pp. 167–170.

[16] Yatsymirskyy, M. and Wiechno, T., *Two-stage Fast Fourier and Hartley Transform of Real Data Sequence*, Przegląd Elektrotechniczny, Vol. R. 86, No. 1, 2010, pp. 41–43.

[17] Yatsymirskyy, M. and Puchala, D., *Fast Time-Decimated Algorithms of Discrete Two-Dimensional Cosine and Sine Transforms of Type II and type IV*,

Modelling and Information Technologies, Institute of Modelling Problems in Power Engineering, Ukrainian Academy of Sciences, Kiev, Ukraine, , No. 30, 2005, pp. 138–149.

[18] Bouguezel, S., Ahmad, M., and Swamy, M., *New Parametric Discrete Fourier and Hartley Transforms, and Algorithms for Fast Computation*, IEEE Transactions On Circuits And Systems, Vol. 58, No. 3, 2011, pp. 562–575.

[19] Puchala, D., *Approximating the KLT by Maximizing the Sum of Fourth-Order Moments*, IEEE Signal Processing Letters, Vol. 20, No. 3, 2013, pp. 193–196.

[20] Stasiak, B. and Yatsymirskyy, M., *Fast Orthogonal Neural Networks*, In: Proc. of 8th International Conf. on Artificial Intelligence and Soft Computing (ICAISC'06), Lecture Notes in Computer Science 4029, 2006, pp. 142–149.