Image Processing Using Cellular Neural Networks Based on Multi-Valued and Universal Binary Neurons

Igor Aizenberg and Constantine Butakoff

Neural Networks Technologies Ltd., 155, Bialik str., Ramat-Gan, 52523, Israel
e-mail: igora@netvision.net.il igora@nnt-group.com (IA); cbutakoff@yahoo.com cbutakoff@nnt-group.com (CB)

Abstract

Multi-valued and universal binary neurons (MVN and UBN) are the neural processing elements with the complex-valued weights and high functionality. It is possible to implement an arbitrary mapping described by partially defined multiple-valued function on the single MVN. An arbitrary mapping described by partially defined or fully defined Boolean function, which can be non-threshold, may be implemented on the single UBN. The quickly converging learning algorithms exist for both types of neurons. Such features of the MVN and UBN may be used for solving the different problems. One of the most successful applications of the MVN and UBN is their usage as basic neurons in the Cellular Neural Networks (CNN). It opens the new effective opportunities in nonlinear image filtering and its applications to noise reduction, edge detection and solving of the super resolution problem. A number of experimental results are presented to illustrate the performance of the proposed algorithms.

Keywords: neural networks, nonlinear filtering, image processing

1. Introduction

The intensive development of the neural networks during last years makes their application to image processing, analysis and recognition very attractive. E.g., many image filtering algorithms in spatial domain may be reduced to the same operation, which has to be performed over all the pixels of the processed image. For example, many of the algorithms of linear and nonlinear filtering in spatial domain are reduced to the processing within a local window around each pixel of the image. We will concentrate here on the nonlinear filtering algorithms that are reduced to the local processing within a window around each pixel and based on the nonlinear transformation of the result of linear convolution with the weighting kernel (template). The main operations in this case are: noise reduction, frequency correction (extraction of image details) and edge detection. A preliminary report on the following aspects has been presented at the 2000 IEEE International Workshop on Neural Networks for Signal Processing [1].

Since processing within a local window around the pixel is not recursive, it may be organized simultaneously for all the pixels, independently on each other. So it is natural to organize this process using some appropriate kind of neural network. The most appropriate neural network for solving of these problems is the Cellular Neural Network (CNN). CNN has been introduced in [2] as a special high-speed parallel neural structure for image processing and recognition. CNN concept has been intensively developed. Many results from simple filters for binary images [2-3] to algorithms for color image processing [4] were carried out. The appearance of CNN progressed in three fields: non-linear dynamic systems, neural networks and image processing. We omit here the first field, and consider the second and the third ones.

A revolutionary role of the CNN in neural networks is first of all in its local connectivity. The neural networks considered before the paper [2] usually were the fully connected networks (e.g., Hopfield network), or multi-layered neural networks with full connections between neurons of neighboring layers (e.g., multi-layered perceptron) [5]. A CNN concept supposes a cellular structure of the network: each neuron is connected only with the neurons from its nearest neighborhood (see Fig. 1). It means that the corresponding inputs of each neuron are connected with outputs of neurons from the nearest rxr neighborhood, and on the other hand, the output of each neuron is connected with the inputs of neurons from the same neighborhood. The neurons of such a network are also often called cells.

Depending on the type of neurons that are basic elements of the network, it is possible to distinguish continuous-time CNN (CTCNN) [2], discrete-time CNN (DTCNN) [6] (oriented especially on binary image processing), CNN based on multi-valued neurons (CNN-MVN) [7, 8] and CNN based on universal binary neurons (CNN-UBN) [8, 9, 10]. CNN-MVN makes possible processing, which is defined by some multiple-valued threshold functions, and CNN-UBN allows processing defined not only by threshold, but also by arbitrary Boolean function. These properties of the CNN-MVN and CNN-UBN will be used here to develop a concept of the nonlinear cellular neural filtering (NCNF) [1, 10].

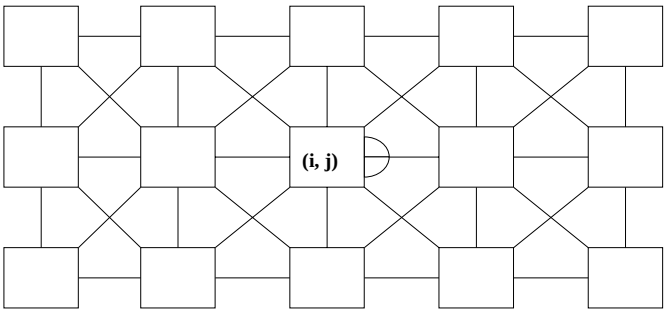


Fig. 1. CNN of a dimension 3x5 with the local connections in a 3x3 neighborhood: each neuron is connected with 8 neurons around it and with itself

We would like to consider below some effective image processing algorithms that should be implemented using CNN-MVN and CNN-UBN. There are the algorithms that belong to the family of nonlinear cellular neural filters.

- The first group of algorithms includes multi-valued filters (MVF) for noise reduction and frequency correction.
- The second one includes filters for the precise edge detection.
- We will also consider how MVF should be effectively used for solving the super resolution problem.

All the problems to be considered here are very important for different tasks of applied image processing: medical imaging, satellite imaging, graphic art, etc.

Additionally to the references mentioned above, the following CNN applications to image processing should be noted: implementation of order statistic nonlinear filtering [11], morphological operations [12], image segmentation [13, 14] and image halftoning [15].

Since the final purpose of this work is to present some new effective solutions in nonlinear image filtering we would like to make here a brief and concentrated observation of the considered problem. We will not present here a comprehensive observation of the nonlinear filtering. The reader, who is interested, should address to the latest monographs devoted to this problem [16-19]. We want to mention here the most important and key issues of the nonlinear cellular neural filtering (NCNF).

The purpose of the filtering operation is assumed to be an effective elimination or attenuation of the noise that is corrupting the signal. We will consider here 2D signals (images) and therefore 2D filters. Let us consider N x M image corrupted by noise. The simplest spatial domain filtering operation is the mean filter and it is defined as

$$\hat{B}_{ij} = \frac{1}{(2m+1)(2n+1)} \sum_{\substack{i-n\le k\le i+n \\ j-m\le l\le j+m}} B_{kl}\,, \tag{1}$$

where B_{kl} are the values of the input signal, $\hat{B}_{ij}$ is the corrected value of the ij^{th} pixel. The range of indices k and l defines the filter window. It is evident, but should be mentioned that mean filter is a linear operation. A more effective linear filter is a simple low-pass filter, which may be considered as a generalization of the mean filter (1):

$$\hat{B}_{ij} = w_0 + \sum_{\substack{i-n\le k\le i+n \\ j-m\le l\le j+m}} w_{kl} B_{kl}\,, \tag{2}$$

where $w_0 \ge 0,\ w_{kl} > 0$. If $w_0 = 0$, and $w_{kl} = \dfrac{1}{(2n+1)(2m+1)}$ then filter (2) is transformed to the filter (1).

Computing simplicity of the linear filters (1) and (2) makes them very attractive. But often it is impossible to obtain nice results with them. They reduce a noise, but not completely, and simultaneously they smooth a signal. May be the simplest nonlinear filter, which is more complicated from the computational point of view, but usually more effective for the noise reduction is a simple median filter, which is defined as

$$\hat{B}_{ij} = \underset{\substack{i-n\le k\le i+n \\ j-m\le l\le j+m}}{MED}\ B_{kl}\,, \tag{3}$$

where *MED* defines a median value within the n x m filter window.

A fundamental difference between the behavior of linear and nonlinear filters is the following. The impulse response (that is, the output signal when the input signal is equal to 1 at time instant 0 and zero otherwise) of a nontrivial time-invariant linear filter clearly cannot be a zero-valued sequence [16]. It means that a linear filter always will smooth the

signal. A nonlinear filter preserves signal carefully, and usually is more effective for noise removal. Different types of nonlinear filters were elaborated and investigated, e.g., *L*-filters [20], rank-order filters [20]. The reader who is interested may address also to the weighted median filters [21], stack filters [16, 17], etc.

It is easy to see from eq. (1) - (3) that a problem of the image filtering in the spatial domain is reduced to the replacement of the current value of brightness B_{ij} in *ij*-th pixel by the value $\widehat{B}_{ij}$, which is better from some point of view.

Let us consider the linear filters (1) and (2). The arithmetic mean is a most popular, but not a single mean. There is a number of other widely used types of mean, e.g., geometric, harmonic, L_p. But calculation of these types of mean is connected with nonlinear operations. A general form of nonlinear mean filters has been considered recently [16, 22]. It is (for 1D filter for simplicity):

$$B_i = g^{-1}\left(\frac{\sum_{k=-n}^{n} w_k g(B_k)}{\sum_{k=-n}^{n} w_k} \right),$$

where $g(x)$ is the following function $g\colon R \to R$:

$$g(x) = \begin{cases} x & \textit{the arithmetic mean} \\ 1/x & \textit{the harmonic mean} \\ \ln x & \textit{the geometric mean} \\ x^p,\ p \in R \setminus \{-1,\ 0,\ 1\} & \textit{the } L_p \textit{ mean} \end{cases}.$$

Since function $g(x)$ is used for averaging of the signal, a following question is natural: is it possible to define other nonlinear means, and therefore, other types of averaging, which may be used for the filtering? The main part of the paper gives a positive answer to this question.

The issue of the approach, which is developed here, is the application of the non-linearity of the neurons' activation functions. A convenience of the spatial linear filters implementation using cellular neural networks is an additional stimulus for the research in this direction. Introduction of the CNN based on MVN (CNN-MVN) [8, 23] and CNN based on UBN (CNN-UBN) [8, 9] opened a possibility for the neural implementation of the original nonlinear spatial domain filters, namely multi-valued filters (MVF) and cellular neural Boolean filters (CNBF). They should be considered as parts of the new nonlinear filters family. This family, which we would like to call as nonlinear cellular neural filters (NCNF), will be considered below.

2. Multi-Valued and Universal Binary Neurons

Universal Binary Neuron (UBN) performs mappings described by arbitrary Boolean function of *n* variables. Multi-Valued Neuron (MVN) performs mappings described by full-defined threshold or partial-defined *k*-valued function (function of *k*-valued logic), where *k* is in general an arbitrary positive integer.

Common basic mathematical background of the UBN and MVN is the following. An arbitrary Boolean function of *n* variables or *k*-valued function of *n* variables is represented by *n*+1 complex-valued weights $w_0, w_1, \ldots, w_n$ [8]:

$$f(x_1,\ldots,x_n) = P(w_0 + w_1 x_1 + \ldots + w_n x_n), \tag{4}$$

where $x_1, \ldots, x_n$ are the variables, which the performed function depends on (neuron inputs) and *P* is the activation function, which is defined in the following way.

1) For Multi-Valued Neuron:

$$P(z) = \exp(i2\pi j/k), \text{if } \ 2\pi j/k \le \arg(z) < 2\pi\,(j+1)/k, \tag{5a}$$

or with integer-valued output:

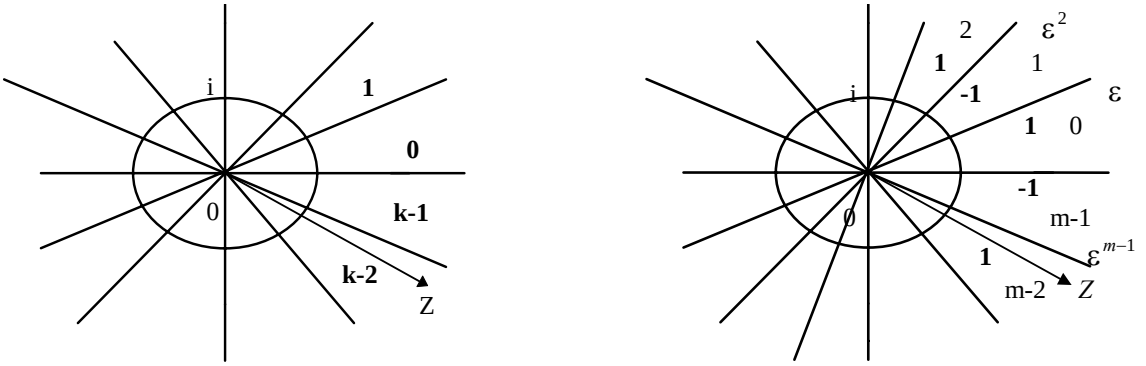
$$P(z) = j, \text{if } \ 2\pi j/k \le \arg(z) < 2\pi\,(j+1)/k, \tag{5b}$$

where j=0, 1, ..., k-1 are the values of the *k*-valued logic, i is the imaginary unity, $z = w_0 + w_1 x_1 + w_n x_n$ is the weighted sum, $arg(z)$ is the argument of the complex number z (values of the function and of the variables are coded by the complex numbers, which are k^{th} roots of unity: $\varepsilon^j = \exp(i2\pi\, j/k)$, $j \in \{0,1,\ldots,k\text{-}1\}$, i is an imaginary unity; in other words values of the *k*-valued logic are represented by k^{th} roots of unity: $j \to \varepsilon^j$);

2) For Universal Binary Neuron

$$P_B(z)=(-1)^j \text{ if } (2\pi j/m) \le \arg(z) < (2\pi\ (j+1)/m) \tag{6}$$

where m is a positive integer, j is a non-negative integer $0 \le j < m$, $z = w_0 + w_1x_1 + w_nx_n$ is the weighted sum, arg(z) is the argument of the complex number z. Estimation for m is $m \ge 2n$ and it is considered in [8]. Fig. 2 illustrates definition of the activation functions (5) and (6).



(a) MVN activation function (b) UBN activation function

Fig. 2 Application of the MVN and UBN activation functions to the weighted sum Z.

Evidently, functions (5) and (6) separate the complex plane into k sectors and m sectors, respectively.
There are at least two quickly converging learning algorithms for MVN [8]. One of them is based on the error-correction learning rule:

$$W_{l+1} = W_l + \frac{1}{(n+1)} C_l(\varepsilon^q - \varepsilon^s)\overline{X} \tag{7}$$

where W_l and W_{l+1} are current and next weighting vectors, $\overline{X}$ is the vector of the neuron's input values (complex-conjugated), ε is the primitive k-th root of unity (k is chosen from (5ab)), C_l is the scaling coefficient, q is a number of the desired sector on the complex plane("correct" sector), s is a number of the sector, where the actual value of the weighted sum fell to("incorrect" sector), n is a number of neuron's inputs. Learning for the UBN may also be reduced to the rule (7). For UBN q has to be chosen on each step based on the following rule: if the weighted sum Z gets into "incorrect" sector number s then both of the neighboring sectors are "correct" [8]:

$q = s - 1 \pmod m$, if Z is closer to $(s-1)$-th sector (8a)

$q = s+1 \pmod m$, if Z is closer to $(s+1)$-th sector. (8b)

3. MVN and UBN – based Cellular Neural Network

Cellular neural networks (CNN) were introduced by Chua and Yang in [2] and became a very appropriate computing model for the solution of different kinds of image processing problems. CNN local connectivity (see Fig. 1) is a key point for the implementation of the different spatial domain filtering algorithms.

The dynamics of Continuous time CNN (CTCNN) neuron (cell) is described by the following equation [2]:

$$v(i,j) = F\left[-\dot{z}(i,j) + \sum_{k=-r}^{r}\sum_{l=-r}^{r}(A(k,l)v(i+k,j+l)) + \sum_{k=-r}^{r}\sum_{l=-r}^{r}(B(k,l)u(i+k,j+l) + I \right]. \tag{9}$$

The dynamics of discrete time CNN (DTCNN) cell is described by the following equation [6]:

$$v(i,j,t)=F_d\left[\sum_{k=-r}^{r}\sum_{l=-r}^{r}(A(k,l)v(i+k,j+l,t\text{-}1)) + \sum_{k=-r}^{r}\sum_{l=-r}^{r}(B(k,l)u(i+k,j+l,t\text{-}1)) + I\right]. \quad (10)$$

Here (i, j) are the coordinates (number) of ij-th neuron, v is the neuron's output, u is the neuron's input, z is the neuron's state, r is the size of a neuron's nearest neighborhood (r x r), t, t+1 are the time slots t and t+1, respectively, A, B are the r x r matrices, which define synaptic weights (feedback and control templates respectively), I is a bias, F and F_d are the activation functions of a CTCNN and DTCNN neuron, respectively. For a CTCNN the activation function is the following:

$$F(z) = \frac{|z+1|-|z-1|}{2}. \quad (11)$$

For the DTCNN the activation function is the following:

$$F_d = \mathrm{sgn}(z). \quad (12)$$

The following remark should me made. A nonlinear D-template for the implementation of nonlinear filters (e.g., median and rank-order ones) has been introduced in [11, 24]). The equation (9) is transformed in this case as it is shown below:

$$v(i,j) = F\left[-z(i,j) + \sum_{k=-r}^{r}\sum_{l=-r}^{r}(A(k,l)v(i+k,j+l)) + \sum_{k=-r}^{r}\sum_{l=-r}^{r}(B(k,l)u(i+k,j+l) + \sum_{k=-r}^{r}\sum_{l=-r}^{r}\left(\hat{D}(k,l)\cdot\Delta V_{uzv}(t)\right) + I\right],$$

where $\hat{D}(k,l)$ is the generalized nonlinear term applied to ΔV, the voltage difference of either the input, state or output values.

The activation function (11) is a piecewise linear function. The activation function (12) is a simple *sign* function, and therefore DTCNN is based on the usual threshold neurons. Evidently, the linear part of the function (11) ensures the implementation of any algorithm of linear 2D filtering in spatial domain, which is reduced to the linear convolution with the weighting window, on the CTCNN cell. DTCNN ensures the implementation of binary image processing algorithms, which may be described by threshold Boolean functions.

Of course, the CTCNN and DTCNN open many possibilities to solve the different problems of image processing. Indeed, they enable the implementation of linear and nonlinear filters in spatial domain. Moreover it is a good way to implement the image processing algorithms described by threshold Boolean functions. On the other hand two problems should be mentioned here. The first one is that a derivation of corresponding templates is necessary for the implementation of any algorithm. Such a derivation may be achieved by several ways [6]: learning (using some learning algorithm to find the weighting templates, which perform the desired global mapping from input state to the corresponding desired output); design (e.g., programming the network to have some desired fixed points); mapping (e.g., simulation of biological phenomenon based on a set of equations). A problem is that many learning and design algorithms for CNN (see e.g., [6] for observation) often are reduced to the solution of differential equations, which may be a specific computing problem. The second problem is that the DTCNN can't be used anywhere for the implementation of the processing algorithms described by non-threshold Boolean functions. The non-threshold function can't be implemented with any learning algorithm on the threshold neural element. But the threshold neuron itself is a basic element of DTCNN. These two problems should not be considered as a disadvantage of CNN. Indeed, many interesting applied problems were solved within CTCNN and DTCNN traditional concept.

But the following ways for development of CNN paradigm are natural among other: 1) increasing of CNN functionality by using of MVN and UBN as basic neurons in the cellular network with the architecture presented in Fig. 1. This brings us to the CNN based on MVN and UBN (CNN-MVN and CNN-UBN); 2) a search for the new problems, which could not be solved within traditional CNN concept, but may be solved on CNN-MVN and CNN-UBN.

The efficiency of CNN-MVN and CNN-UBN is based on the following observations: 1) high functionality (because of universal functionality of UBN, and practically universal functionality of MVN); 2) quickly converging learning algorithms; 3) use of the non-linearity of MVN activation function (5) for the development of nonlinear filters.

Let us now define CNN-MVN and CNN-UBN. Consider a CNN with the obvious local connection structure (see Fig. 1) of a dimension N x M. Consider MVN or UBN as a basic neuron of such a network. Evidently, each cell of CNN-MVN (or CNN-UBN) will perform the following correspondence between neuron's inputs and output:

$$Y_{ij}(t+1) = F\left[w_0 + \sum_{k=-r}^{r}\sum_{l=-r}^{r} w_{kl}^{ij} x_{kl}^{ij}(t)\right] \quad (13)$$

where: ij are the two-dimensional coordinates (number) of a neuron; Y_{ij} is the neuron's output; w_{kl}^{ij} is the synaptic weight corresponding to the input of ij^{th} neuron, to which the signal from the output of kl^{th} neuron is transmitted (one may make a comparison to the elements of B-template in the traditional CNN (see (9) and (10)); x_{kl}^{ij} is the input of ij^{th} neuron, to which

the signal from the output of kl^{th} neuron is transmitted; F is the activation function of the neuron, which is defined by the equation (5) for MVN, and by the equation (6) for UBN; $\sum_{m=-r}^{r}$ means that each neuron is only connected with the neurons from its nearest r-neighborhood, as usual for CNN. Single weighting template of CNN-UBN and CNN-MVN may be written in the following form:

$$w_0;\quad W=\begin{pmatrix} w^{ij}_{i-r,j-r} & \cdots & w^{ij}_{i-r,j+r} \\ \cdots & w^{ij}_{ij} & \cdots \\ w^{ij}_{i+r,j-r} & \cdots & w^{ij}_{i+r,j+r} \end{pmatrix},$$

where W is a rxr matrix.

What is common and what is different with the traditional CNN? First of all the cellular architecture of network is an important common point. It is evident, and it will be shown below on many examples, that CNN-MVN and CNN-UBN are oriented towards the solution of the same problems of image processing and recognition, as traditional CNN. The main difference between the traditional CNN, and CNN-MVN, CNN-UBN is the use of complex-valued weights in the last ones. At first sight it is a disadvantage. But at the same time CNN-UBN and CNN-MVN use a single complex-valued weighting template, and the traditional CNN uses two real-valued weighting templates. So, from the point of view of the weights' complexity the traditional CNN, and CNN-UBN, CNN-MVN are equivalent.

To conclude this observation of CNN, it should be mentioned that CNN-UBN and CNN-MVN are not alternative to the traditional CNN. They may be considered as a nice supplement, which gives high efficiency in applications (to be considered below). On the other hand the elaboration of the new image processing algorithms has been initiated by the concept of CNN-UBN and CNN-MVN.

4. Nonlinear Cellular Neural Filtering

Two-dimensional nonlinear cellular neural filter (NCNF) may be defined in the following way:

$$\widehat{B}_{ij}=F(w_0+\sum_{\substack{i-n\le k\le i+n\\ j-m\le l\le j+m}} w_{kl}Y_{kl}) \tag{14}$$

Here m x n is a filter window, w_0, w_{kl} are the complex-valued weighting coefficients, $\widehat{B}_{ij}$ is a signal value in the ij^{th} pixel after filtering, definition of Y_{kl} and F depends on a type of the two NCNF subfamilies – multi-valued filters (MVF) and cellular neural Boolean filters (CNBF). Let us consider both subfamilies.

4.1. Multi-valued nonlinear filtering

Multi-valued nonlinear filters (MVF) have been introduced recently [1, 8, 9]. A specific complex non-linearity of the multi-valued neuron activation function [8] is a key for the development of a new filters family.

Let $0\le B\le k-1$ be a dynamic range of 2D signal. Let consider a set of the k^{th} roots of unity. We can put a root of unity to the correspondence with the real value B:

$$e^{B}=\exp(i2\pi B/k)=Y. \tag{15}$$

Thus, we have the univalent mapping $B\leftrightarrow\varepsilon^{B}$, where ε is a primitive k^{th} root of unity.

A two-dimensional *Multi-Valued Filter* (MVF) is a filter, which is defined by the following formula [10]:

$$\widehat{B}_{ij}=P(w_0+\sum_{\substack{i-n\le k\le i+n\\ j-m\le l\le j+m}} w_{kl}Y_{kl}) \tag{16}$$

where P is the activation function (5b) of the multi-valued neuron, Y_{kl} is obtained from B_{kl} by the equation (15), i,j are the coordinates of the filtered pixel, $n\times m$ is a filter window, w_{kl} are the weighting coefficients (complex-valued in general). Evidently, the equality (16) defines a class of filters. Each of them is defined by the corresponding set of the weighting coefficients (weighting template).

Let us compare the filter (16) to the linear filters (1) and (2). We can make the following conclusion: without the non-linear function P and complex addition (if Y will be replaced by B), and with $w_{kl} > 0$ filter (16) is transformed to the simple low-pass spatial domain linear filter. But the filter (16) is principally nonlinear. First of all the function P is nonlinear, but additionally the complex addition together with the transformation (15) defines a specific nonlinear averaging of the signal. A detailed mathematical investigation of the properties of non-linearity carried by complex addition and function P may be a subject of a special work. It should be mentioned here that impulse response of MVF is a zero-valued sequence, and this feature brings MVF together with other nonlinear filters.

One remark should be made before we will move to the applications of MVF. The integration of MVF with the different well-known nonlinear spatial domain filters leads to their promising modifications. Let us consider, e.g., a filter, which is defined by equation (16), but simultaneously connected with the weighted rank-order filter. The order-statistic filters and the weighted rank-order filters are based on the compromise between a pure nonlinear operation (ordering) and a pure linear operation (weighting). Actually e.g., a weighted rank-order filter may be defined by the same equation (2) as the low-pass filter, with an additional condition that addends under a sign Σ have to satisfy the rank-order criteria (the order-statistic of addends has to be in the limited neighborhood of the order-statistic of the filtered pixel). If we will implement a weighted rank-order filter, but together with the transformation (15), complex addition, and the function P applied to the weighted sum, we will obtain a *rank-order multi-valued filter* (ROMVF). MVF may be also connected with other different nonlinear filters in the same way.

Evidently, the most appropriate structure for the MVF implementation is CNN-MVN.

The most simple, but very effective template for the reduction of Gaussian or uniform noise is the following:

$$\mathrm{W} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & G & 1 \\ 1 & 1 & 1 \end{pmatrix} \tag{17}$$

where G is a parameter. The template (17) is designed from the following considerations. If $G = 0$ and all other weights are equal to 1 then a maximal signal averaging will be achieved. If $G > 1$ then the smoothing of the signal will be more light and the image boundaries will be preserved with more accuracy. At the same time for $G > 32$ the filter will be degenerated because it will not change the signal. Thus, to remove a noise with the impulsive component we have to recommend $0 \le G \le 1$. To remove the Gaussian or uniform noise, which is not mixed with the impulsive one our recommendation for G is the following: $1 \le G \le 32$.

Let us consider an example. It shows a reduction of zero-mean Gaussian noise with a dispersion equal to 0.3σ, and it is presented in Fig. 3. The results of rank-order filtering of the same images are given for comparison. The enhanced differences between noisy image and the resulting images are also shown. Table 1 contains the objective statistical characteristics of the original, corrupted, and filtered images.

Table 1. Statistical characteristics for the example given in Fig. 3

Image	"Rose"	"Noisy Rose"	MVF, G=8	MVF, G=16, 1 iteration	MVF, G=16, 2 iterations	ROMVF	ROF
ND (σ)	253.8	306.36	69.61	81.83	24.54	164.5	212.0
PSNR	-	24.69	27.50	26.90	27.50	26.30	22.91
SD	0	14.86	10.71	11.50	10.71	12.35	18.20

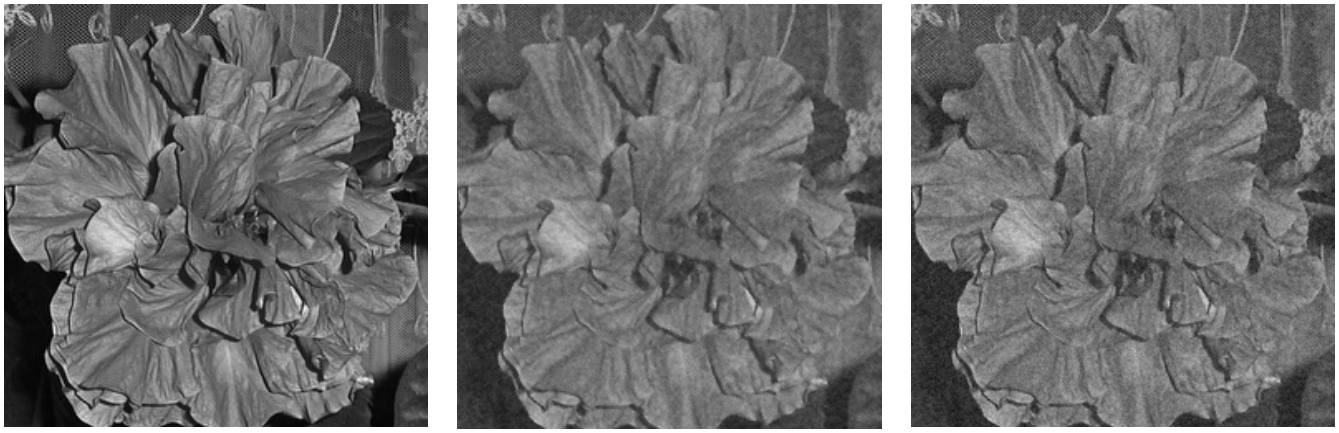
ND (σ) – estimation of white noise dispersion, PSNR – estimation of peak signal to noise ratio, SD – standard deviation from the original image.

The example shows advantages of MVF. The noise is reduced effectively by simple MVF, image boundaries are preserved effectively by rank-order MVF (ROMVF). All implementations of MFV give better results than the usual rank-order filter.

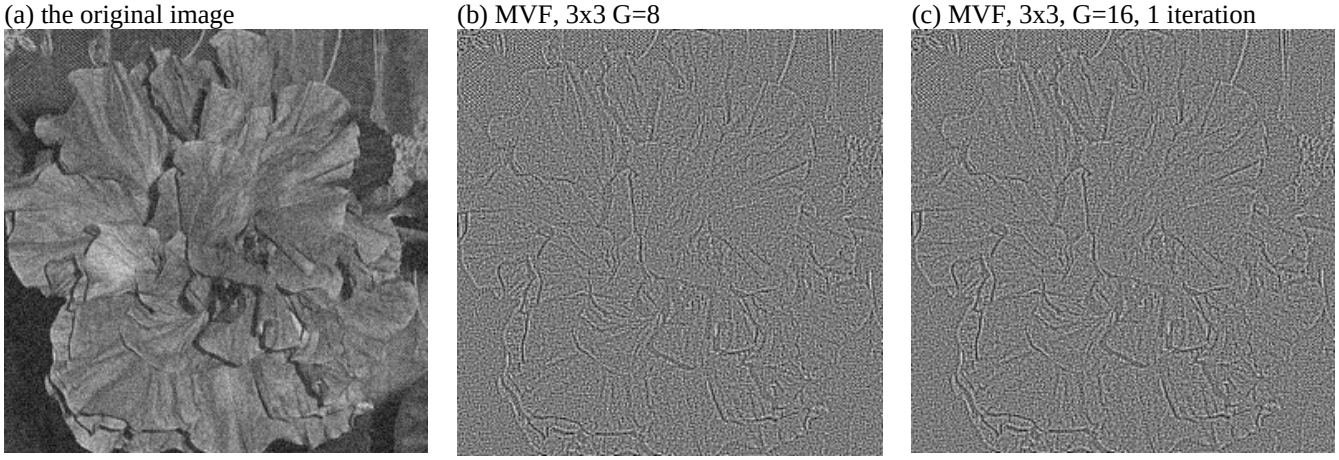
4.2. Multi-valued frequency correction

Multi-valued filters are also a highly effective mean for the frequency correction. A frequency correction problem may be considered as a specific "anti-filtering". Indeed, the filtering is a reduction of the high frequency. But it is natural to consider the inverse problem: amplification of the high and medium frequency. It is clear that by finding an appropriate solution of such a problem it is possible to solve the problems of image sharpening, extraction of details against a preventing background, local contrast enhancement, etc.

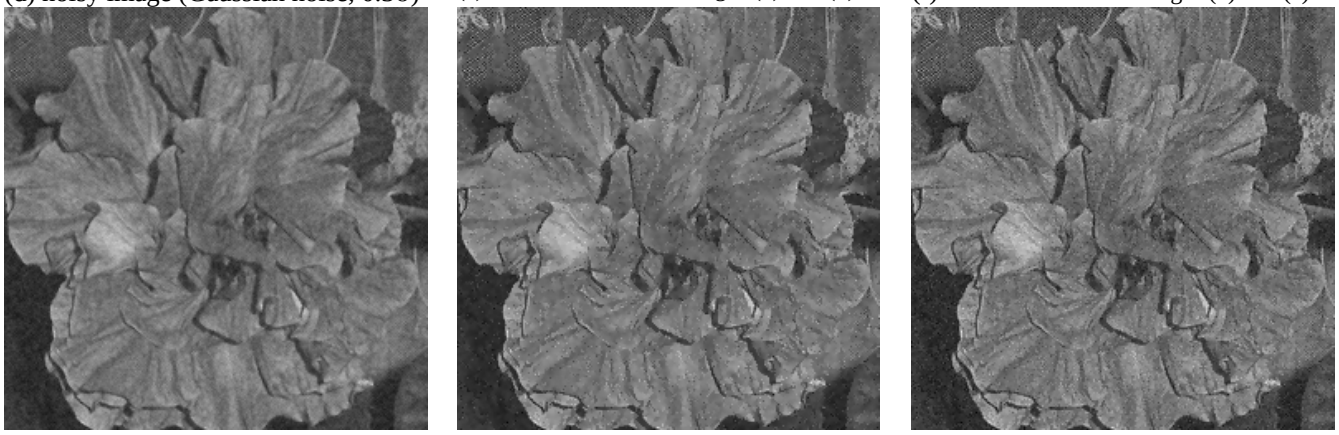
A natural solution to the frequency correction problem is a filtering in the frequency domain. It is clear that this approach is complicate from the computing point of view. Direct and inverse Fourier transformations, search for the appropriate filter mask, and multiplication of the spectrum by this mask require many operations and therefore it will take a long time. A very interesting idea [25] is the approximation of the corresponding filters in frequency domain by simple linear filters in spatial domain.



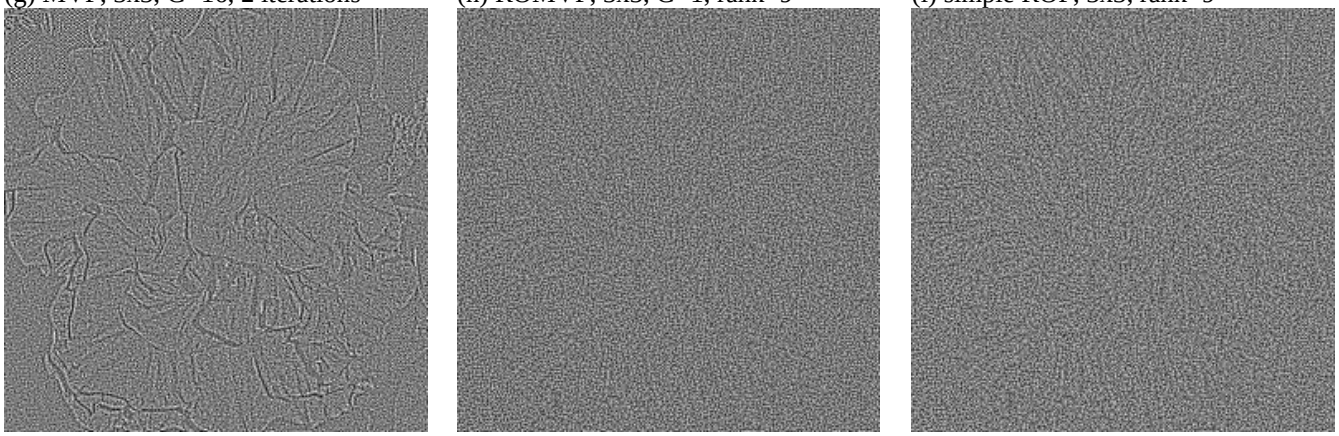
(a) the original image (b) MVF, 3x3 G=8 (c) MVF, 3x3, G=16, 1 iteration



(d) noisy image (Gaussian noise, 0.3σ) (e) difference between images (d) and (b) (f) difference between images (d) and (c)



(g) MVF, 3x3, G=16, 2 iterations (h) ROMVF, 3x3, G=1, rank=9 (i) simple ROF, 3x3, rank=9



(j) difference between images (d) and (g) (k) difference between images (d) and (h) (l) difference between images (d) and (i)

Fig. 3. Noise reduction using multi-valued filtering

The extraction, localization, and detection of important features on the complex image background were proposed to consider [25] as a problem of localization of objects or details of a given type on the image. Such a type has been defined as sizes of the objects. Then this idea has been developed in [9], where the solution of the problem using traditional CNN also has been considered. The following linear filters have been proposed for the solution of the global [9, 25] and high [9, 26] frequency correction problems, respectively

$$\widehat{B}_{ij}=G_1B_{ij}+G_2(B_{ij}-B_m)+G_3B_m+c \tag{18}$$

$$\hat{B}_{ij}=(G_1+G_2)B_{ij}-G_2B_m+c\,, \tag{19}$$

where B_m is a local mean value in a window around the pixel B_{ij}; B_{ij}, and $\widehat{B}_{ij}$ are the signal values in ij^{th} pixel before and after the processing, respectively; G_1 and G_3 define the correction of the low spatial frequencies, G_2 defines the correction of the high spatial frequencies, c is a mean value of the background after processing. It should be noted that filter (19) could be obtained from the filter (18). By the way, if $G_1=1$ then filter (19) coincides with the filter called "unsharp masking" in different issues (see, e.g., [26]).

Although the filters (18) and (19) give good results, they have some significant disadvantages. They are very sensitive to the choice of the weighting coefficients. Even minimal change of any of them may move a general dynamic range of the image to "white" or "black" side. At the same time the global image histogram may dramatically change. It means, that some information will be lost, because some pixels will be replaced by truly white or truly black ones. On the other hand even a small increment of the weighting coefficients often leads to the extremely high amplification of the high frequency and a noise may appear on the image as a result.

To break these disadvantages, linear filters (18) and (19) should be transformed to the nonlinear ones. Since filters (18) and (19) operate with the arithmetic mean, they should be generalized using nonlinear means. One of the natural solutions is implicated from the following considerations. Multi-valued filter (16) is a generalization of the linear filters (1)-(2). What will be a multi-valued generalization of the filters (18)-(19)? Taking into account (15), it is easy to obtain the following multi-valued generalization for the filter (18):

$$\hat{B}_{ij}=P\left[c+(G_1+G_2)Y_{ij}+(G_3-G_2)\sum_{Y(k,l)\in R_{ij}}Y_{kl}\right], \tag{20}$$

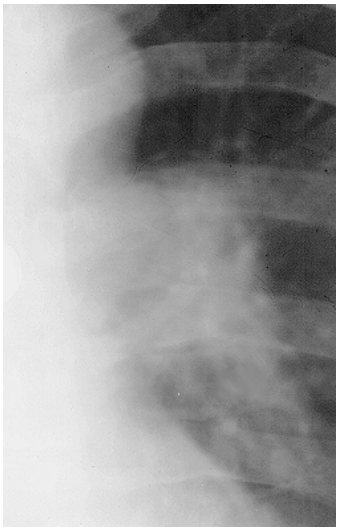
where Y_{kl} is obtained from B_{kl} by equation (15), R_{ij} is a local window around pixel Y_{ij}; B_{ij} (Y_{ij}) and $\hat{B}_{ij}$ are the signal values in ij^{th} pixel before and after processing, respectively, G_1, G_3 are the coefficients, which define the correction of the low frequencies, G_2 defines the correction of the high frequencies, c is a constant. Filter (20) solves a global frequency correction problem. Its application to the image leads to the extraction of details whose sizes are approximately equal to the sizes of the filter window. MVF for the high frequency correction may be obtained from (20) by putting $G_3=0$:

$$\hat{B}_{ij}=P\left[c+(G_1+G_2)Y_{ij}-G_2\sum_{Y_{kl}\in R_{ij}}Y_{kl}\right], \tag{21}$$

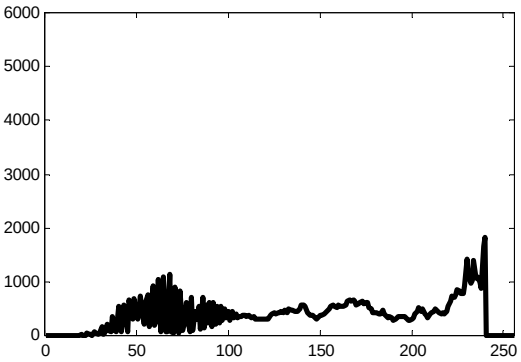
where Y_{kl} is obtained from B_{kl} by equation (15), R_{ij} is a local window around pixel Y_{ij}; B_{ij} (Y_{ij}) and $\hat{B}_{ij}$ are signal values in ij^{th} pixel before and after processing respectively, G_1 and G_2 are the correction coefficients, c is a constant.

Filters (20) and (21) are not so sensitive to the choice of the correction coefficients in comparison to the filters (18) and (19). They always preserve a global dynamic range of the image and its global histogram. Such a tolerance may be used for the processing of the images that contain many details of different sizes and different configuration. The following weighting templates implement filters (20) and (21) on the CNN-MVN, respectively:

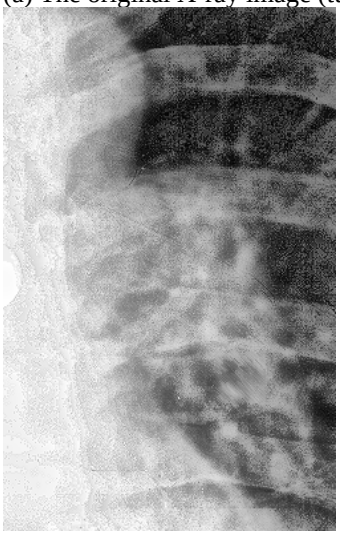
$$w_0=c;\ W=\begin{pmatrix} G_3-G_2 & \dots & \dots & G_3-G_2 \\ \cdot & & & \cdot \\ \cdot & \dots & G_1+G_3\ \dots & \cdot \\ \cdot & & & \cdot \\ G_3-G_2 & & \dots & G_3-G_2 \end{pmatrix} \tag{22}$$



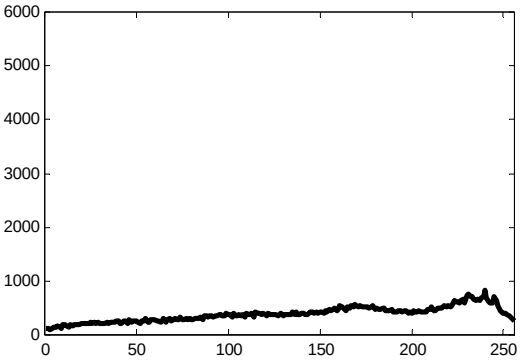
(a) The original X-ray image (tumor of lung)



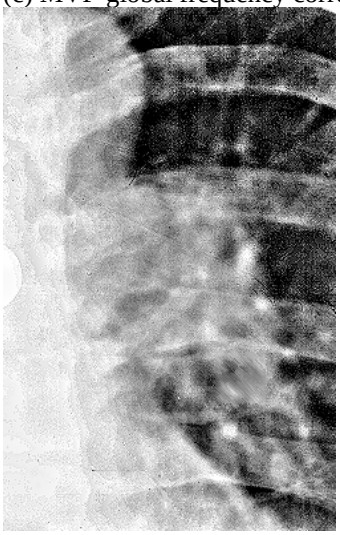
(b) Histogram of the original image (a)



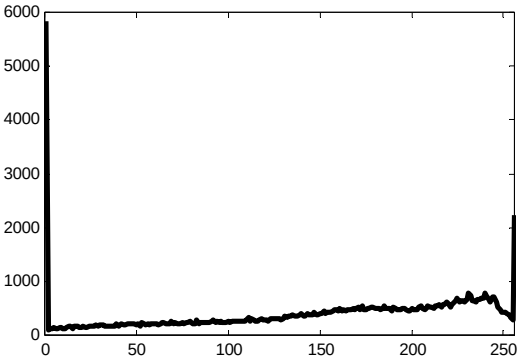
(c) MVF-global frequency correction, eq. (20), 35x35



(d) Histogram of the image (c) after MVF-global frequency correction. This histogram does not contain any peak



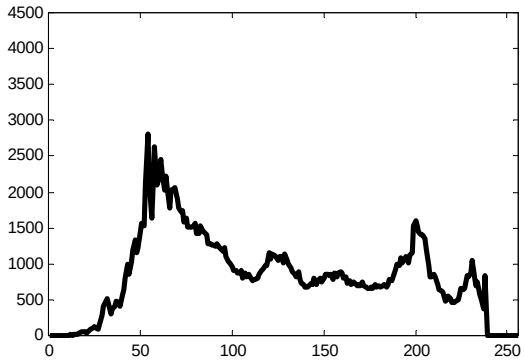
(e) Linear global frequency correction, eq. (18), 35x35 (many pixels became truly white or truly black)



(f) Histogram of the image (e) after linear global frequency correction. High peaks in “0” and “255” are clearly visible. It means that many pixels have become truly white or truly black. As a result, information in the corresponding pixels is lost

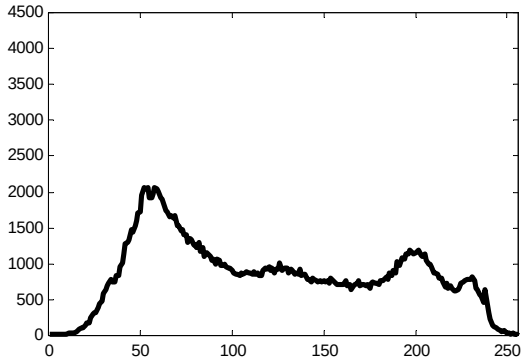
Fig. 4. MVF-global frequency correction and its comparison to linear global frequency correction

(a) The original image “Sydney”

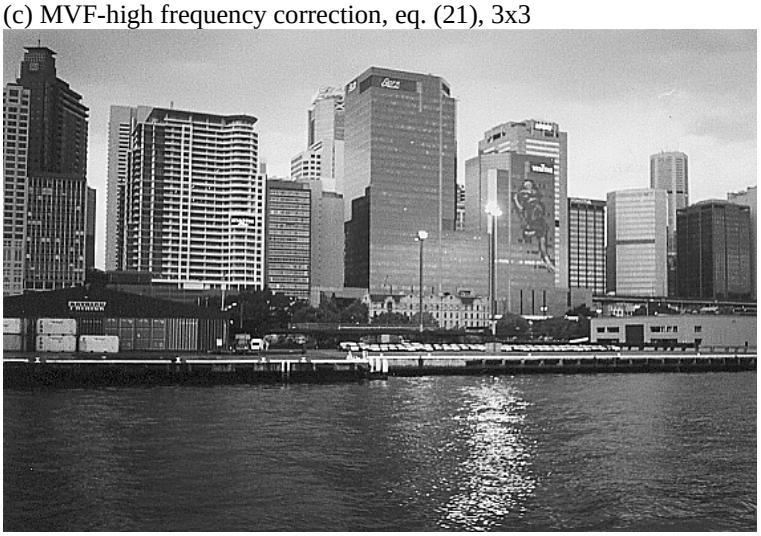


(b) Histogram of the original image (a)

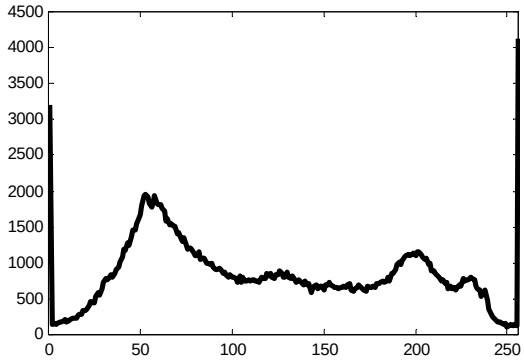
(c) MVF-high frequency correction, eq. (21), 3x3



(d) Histogram of the image (c) after MVF-high frequency correction. This histogram does not contain any peak



(e) Linear high frequency correction, eq. (19), 3x3 (many pixels became truly white or truly black)



(f) Histogram of the image (e) after linear high frequency correction. High peaks in “0” and “255” are clearly visible. It means that many pixels have become truly white or truly black. As a result, information in the corresponding pixels is lost

Fig. 5. MVF-high frequency correction and its comparison to linear high frequency correction

$$w_0=c;\ W=\begin{pmatrix} -G_2 & \dots & \dots & -G_2 \\ \cdot & & & \cdot \\ \cdot & \dots & G_1 \dots & \cdot \\ \cdot & & & \cdot \\ -G_2 & & \dots & -G_2 \end{pmatrix}. \tag{23}$$

Filter (20) extracts image details with the sizes approximately equal to the sizes of the filter window. Filter (21) extracts the smallest image details. Thus, the most appropriate size for the window of the filter (21) is 3x3. Estimations for the values of the weighting coefficients in (20) and (22) are the following: $\frac{nm}{2} \le G_2 < nm$, $G_2 - 1 < G_3 < G_2$, $\frac{nm}{2} - G_3 < G_1 < 2nm$, where *n* x *m* is a filter window. Estimations for the values of the weighting coefficients in (21) and (23) are the following: $\frac{nm}{2} \le G_1 < 2nm$, $0 < G_2 < 1$. Let us consider some examples of image processing using filters (20) and (21) and their comparison to the results obtained by filters (18) and (19) (see Fig 4 and Fig. 5).

The presented examples show high efficiency of the MVF frequency correction for image sharpening and extraction of image details. They also show advantages of the MVF method in comparison to the linear one. The linear frequency correction leads to extremely high amplification of the high frequency. As a result, many truly white and black pixels appear at the image. Some details become sharper, but some other disappear, due to dramatic change of the global image histogram. The MVF frequency correction is free of these disadvantages. It is very important for the applications, for example, in medical imaging, satellite imaging, digital photography, graphic arts, etc.

4.3. Cellular Neural Boolean Filtering

Idea of cellular neural Boolean filtering (CNBF) has been proposed in [9]. This idea is based on a separate processing of the image binary planes using special spatial-domain filter defined by some Boolean function and further integration of the resulting binary planes into the resulting image. This method has been called a cellular neural Boolean filtering in [8, 10]. A key issue for the method was ability of CNN-UBN to implement spatial domain filter described by arbitrary Boolean function [8, 9] (not only threshold Boolean function as for DTCNN [6]).

Let B_{kl} be a signal value in the ij^{th} pixel and $B \in \{0, 1, ..., 2^k - 1\}$. Let us split the image onto *k* binary planes directly, without any thresholding. Let B_{ij}^s be a binary value of the signal corresponding to s^{th} binary plane, Y_{ij}^s is the same value transformed to the Boolean alphabet {1, -1}. Let *F* in (14) be the UBN activation function (6). Taking into account (6), (14) is transformed to the following form:

$$\hat{B}_{ij}^s = P_B\Big(w_0 + \sum_{\substack{i-n\le k\le i+n \\ j-m\le l\le j+m}} w_{kl} Y_{kl}^s\Big), \tag{24}$$

The equation (24) establishes CNBF for a binary plane. CNBF for the gray-scale image or for the color channel of the color image should be written in the following form:

$$\hat{B}_{ij} = \bigoplus_{s=0}^{k-1} \left[P_B\Big(w_0 + \sum_{\substack{i-n\le k\le i+n \\ j-m\le l\le j+m}} w_{kl} Y_{kl}^s\Big) \right], \tag{25}$$

where $\bigoplus_{s=0}^{k-1}$ denotes the integration of *k* binary planes from 0^{th} until $(k\text{-}1)^{\text{th}}$ into the gray-scale image (color channel). Equations (24) and (25) may be also written in the following form, respectively:

$$\hat{B}_{ij}^s = f\left(\underset{\substack{i-n\le k\le i+n \\ j-m\le l\le j+m}}{B_{k_1 l_1}^s, B_{k_1 l_2}^s, \dots B_{k_n l_m}^s} \right), \tag{26}$$

$$\bar{B}_{ij} = \bigoplus_{s=0}^{k-1} \left[f\left(\underset{\substack{i-n\le k\le i+n \\ j-m\le l\le j+m}}{B^s_{k_1 l_1}, B^s_{k_1 l_2}, \ldots B^s_{k_n l_m}} \right) \right], \tag{27}$$

where f is a Boolean function, which defines spatial domain filter with n x m window for the binary image (binary plane). Evidently, (24) describes the implementation of the Boolean function (26) using UBN, or CNN-UBN cell (one may compare the equation (24) to the equation (13)).

The most impressive application of CNBF is a precise edge detection, which has been proposed in [9] and then developed (see, e.g., [1, 8, 10]). Many of the different edge detection algorithms are well known. The most known group of edge detectors is based on the convolution with zero-sum kernels [26, 27]. Some of these algorithms are even classical (e.g., Laplacian, Sobel, Prewitt). These algorithms are also classified as convolutions with the 1st order derivative kernels (e.g. Previtt) and 2nd order derivative kernels (e.g. Laplacian). Another group of edge detectors is based on a local statistical analysis [19, 28]. A problem is that all of the mentioned algorithms do have a common and significant disadvantage: they are usually very sensitive to the significant brightness jumps and unable to detect the edges corresponding to the small brightness jumps. It means that all these algorithms give good results while applied to the images with a good contrast. The application of the same algorithms to the images with a poor contrast or to the images with many brightness jumps of a different level is possible, but the results often are not satisfactory. Other problems are the detection of the edges corresponding to the details of a complicated form and edge detection on the noisy images.

One of the good alternative solutions is the edge detection using CNBF. Let us consider a background for the design of the Boolean functions, which implement the corresponding CNBF. The following key items should be taken into account:

1) when we say that the edge in some pixel of binary image is detected, it means that the brightness value in such a pixel differs from the brightness value of at least one of the other pixels from the n x n window around it. If the isolated pixels should be removed from the consideration, then the following supplement to this condition should be added: the brightness value in such a pixel must be equal to the brightness value of at least one of the other pixels from the n x n window around it. We will not consider this case further;
2) the upward brightness jumps (from 0 to 1, if 1 is the value of brightness in the analyzed pixel), and the downward ones (from 1 to 0, if 0 is the brightness value in the analyzed pixel) are not equivalent in general;
3) it follows from the item 2 that the next three cases of edge detection should be considered to obtain the most general picture of the jumps of brightness: a) separate detection of edges corresponding to the upward brightness jumps; b) separate detection of edges corresponding to the downward brightness jumps; c) joint detection of edges corresponding to the upward and downward brightness jumps.

The Boolean functions used for the CNBF edge detection with a window 3x3, are presented in [9, 10]. For example, the edge detection corresponding to the upward brightness jumps detection within a 3x3 window is implemented using the following Boolean function:

$$f\begin{bmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{bmatrix} = x_5 \,\&\, (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee \bar{x}_6 \vee \bar{x}_7 \vee \bar{x}_8 \vee \bar{x}_9). \tag{28}$$

.

This function may be easily implemented on the UBN. The learning algorithm (7)-(8) requires a few iterations to converge, starting from any random vector. The example of precise edge detection using the filter defined by (27)-(28) and the comparison to other edge detectors are presented in Fig. 6.

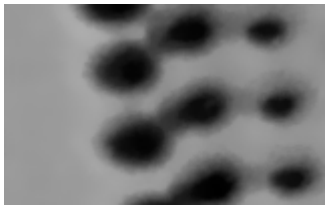
In [8] CNBF with a 3x3 window for the precise edge detection in a particular direction has been also considered.

At the same time it is very attractive to consider the precise edge detection using CNBF with a 5x5 window. It is important for the detection of edges corresponding to the details of a complicated form.

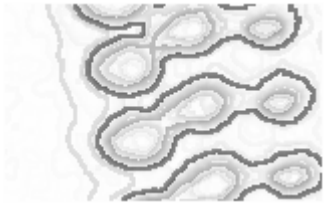
Evidently, it is possible to design many functions for the edge detection with a 5x5 window in the different particular directions. For example, the following function may be considered:

Upward jumps, South-East

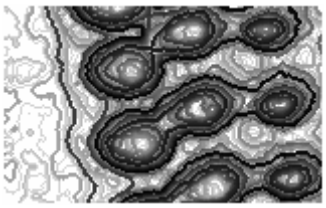
$$f\left[\begin{array}{ccccc} x_1 & x_2 & x_3 & x_4 & x_5 \\ \hline x_6 & x_7 & x_8 & x_9 & x_{10} \\ x_{11} & x_{12} & \boldsymbol{x}_{13} & x_{14} & x_{15} \\ x_{16} & x_{17} & x_{18} & \boldsymbol{x}_{19} & x_{20} \\ \hline x_{21} & x_{22} & x_{23} & x_{24} & \boldsymbol{x}_{25} \end{array}\right] = (x_{13} \,\&\, x_{19} \,\&\, x_{25}) \,\&\, (\overline{x_{14}} \vee \overline{x_{18}}) \,\&\, (\overline{x_{20}} \vee \overline{x_{24}})$$



(a) Input image from electronic microscope

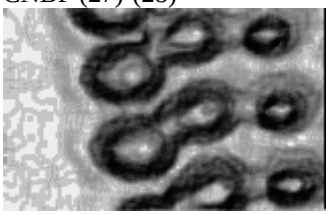


(b) Edge detection using CNBF (27)-(28)[1]

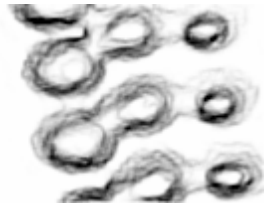


(c) Image (b) with the enhanced contrast

(d) Edge detection using Sobel operator: edges are smoothed



(e) Edge detection using the algorithm based on a local statistics analysis [19]: edges are smoothed



(f) Edge detection using the variance (2^{nd} order) enhancement based on [28]: edges are smoothed

Fig. 6. Precise edge detection using CNBF with a 3x3 window

The function, which implements CNBF for complete precise edge detection corresponding to the upward brightness jumps:

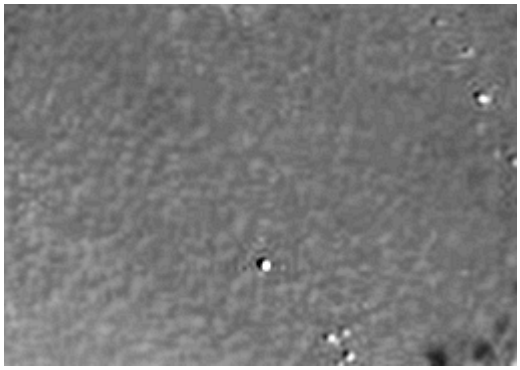
$$f_{up}\begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 \\ x_6 & x_7 & x_8 & x_9 & x_{10} \\ x_{11} & x_{12} & x_{13} & x_{14} & x_{15} \\ x_{16} & x_{17} & x_{18} & x_{19} & x_{20} \\ x_{21} & x_{22} & x_{23} & x_{24} & x_{25} \end{bmatrix} = \begin{aligned} &[(x_{13} \,\&\, x_7 \,\&\, x_1) \,\&\, (\overline{x_8} \vee \overline{x_{12}}) \,\&\, (\overline{x_2} \vee \overline{a_6})] \vee \\ &[(x_{13} \,\&\, x_8 \,\&\, x_3) \,\&\, (\overline{x_7} \vee \overline{x_9}) \,\&\, (\overline{x_2} \vee \overline{a_4})] \vee \\ &[(x_{13} \,\&\, x_9 \,\&\, x_5) \,\&\, (\overline{x_8} \vee \overline{x_{14}}) \,\&\, (\overline{x_4} \vee \overline{a_{10}})] \vee \\ &[(x_{13} \,\&\, x_{14} \,\&\, x_{15}) \,\&\, (\overline{x_9} \vee \overline{x_{19}}) \,\&\, (\overline{x_{10}} \vee \overline{a_{15}})] \vee \\ &[(x_{13} \,\&\, x_{19} \,\&\, x_{25}) \,\&\, (\overline{x_{14}} \vee \overline{x_{18}}) \,\&\, (\overline{x_{20}} \vee \overline{a_{24}})] \vee \\ &[(x_{13} \,\&\, x_{18} \,\&\, x_{23}) \,\&\, (\overline{x_{19}} \vee \overline{x_{17}}) \,\&\, (\overline{x_{22}} \vee \overline{a_{24}})] \vee \\ &[(x_{13} \,\&\, x_{17} \,\&\, x_{21}) \,\&\, (\overline{x_{12}} \vee \overline{x_{18}}) \,\&\, (\overline{x_{16}} \vee \overline{x_{22}})] \vee \\ &[(x_{13} \,\&\, x_{12} \,\&\, x_{11}) \,\&\, (\overline{x_1} \vee \overline{x_{17}}) \,\&\, (\overline{x_6} \vee \overline{x_{16}})]. \end{aligned} \tag{29}$$

Evidently, it consists of the disjunction of the functions that implement edge detection in all the directions.

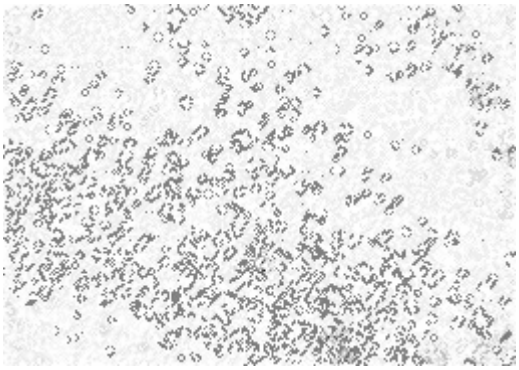
Fig. 7 presents the results of precise edge detection using CNBF with a 5x5 window. This example is a very nice illustration of how it is possible to detect the sea streams in the different directions using the presented technique.

It is clear from the example that CNBF with a 5x5 window is more effective than CNBF with a 3x3 window for detection of the edges corresponding to the details of a complicated form (one may compare images in Fig. 7e and 7f). It is also more effective than a classical solution for the edge detection in a particular direction (one may compare images in Fig. 7c and 7d).

[1] Here and further the edged images are given with the inverted palette (“black” on “white”)

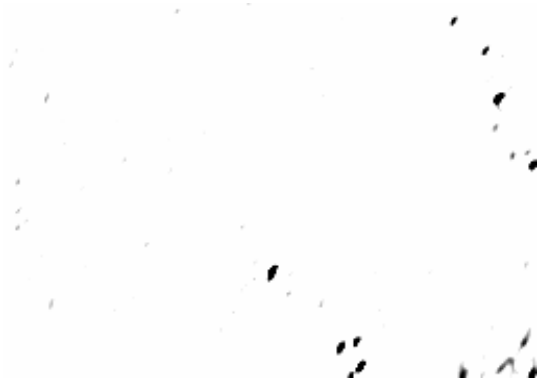


(a) The original image: sea surface

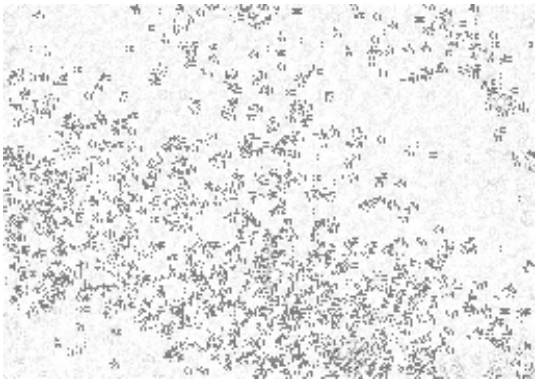


(b) North-East+North-West+ South-West+South-East

(c) North-East+South-West



(d) For comparison: North-East+South-West using the zero-kernel convolution presented in [27]



(e) All directions (filter (27)+(29))

(f) Edge detection using CNBF with a 3x3 window (filter (27)-(28))

Fig. 7. Precise edge detection using CNBF with a 5x5 window

5. Application of the Multi-Valued Filters to the Solution of the Super-Resolution Problem

The idea of the Fourier spectrum extrapolation to reach a super-resolution effect has been proposed in [29, 30]. The approach to solution of the super-resolution problem, which will be developed here, has been proposed in [7] and then considered in [8]. This approach is based on the iterative approximation of the Fourier, Cosine or Walsh image spectra in the highest frequency domain. We will develop this approach here.

Let $f(x, y)$ be a two-dimensional signal or discrete image of a *n x n* sizes, without loss of generality. Let A be a spatial domain, on which the signal is defined, $\tilde{A} \subset A$ be its subdomain. The function

$$F(u,v) = \Phi\left[f(x, y)\right]; u,v \in \{0,1,...,n-1\} ,$$

where Φ is Fourier, Cosine, or Walsh (ordered by Walsh) transform, is a spectrum of the signal $f(x, y)$. The super-resolution problem can be separated into two problems: extrapolation of spectrum to the domain

$u, v \in \{0,1,...,n-1,n,...,2n-1\}$ and interpolation of signal on the whole domain A. To solve these two problems, we will consider the following iterative process [8].

To initiate the iterative process, we need to extend the signal from domain $\tilde{A}$ to domain A. So let us build the first approximation to the resulting signal:

$$g(x,y) = \begin{cases} f(x,y), \text{if } (x,y) \in \tilde{A} \\ s(x,y), \text{if } (x,y) \in A \setminus \tilde{A}, \end{cases} \tag{30}$$

where $s(x,y)$ can be the uniform noise with a small dispersion and mean equal to the mean of $f(x,y)$. $s(x,y)$ can also be the result of interpolation of the signal $f(x,y)$ to the whole domain A. It should be noted that it was proposed in [29, 30] to use a zero-constant as $s(x,y)$. But this could not be recognized as a good solution because the results of the following Fourier spectrum extrapolation leads to the resulting image, which is very close to the result of interpolation.

For simplicity and without loss of generality $g(x,y)$ may be considered as $f(x,y)$, corrupted by additive uniform noise, and furthermore we know, that $f(x,y)$ is corrupted only in the domain $A \setminus \tilde{A}$. This assumption is true also, when $s(x,y)$ is the result of interpolation. Since $f(x,y)$ and $s(x,y)$ were obtained separately in this case, they have a different statistics in general. This feature leads to the noisy visual effect. So we can reformulate our problem of the super-resolution as a problem of noise reduction and correction of the highest frequencies. But if we take a closer look, we can see that the spectrum of original signal $f(x,y)$ is also known. It can be used for particular reconstruction of the spectrum of $g(x,y)$.

Thus let us assume that $f_1(x,y) = g(x,y)$. The n^{th} approximation $f_n(x,y)$ for $f(x,y)$ can be obtained in the following way:

$$F_{n-1}(u,v) = \begin{cases} F(u,v); u,v \in \{0,1,..., n-1\} \\ \Phi[f_{n-1}(x,y)]; u,v \in \{n,...,2n-1\} \end{cases} \tag{31}$$

$$g_{n-1}(x,y) = \Phi^{-1}[F_{n-1}(u,v)] \tag{32}$$

$$f_n(x,y) = \begin{cases} f(x,y), \text{if } (x,y) \in \tilde{A} \\ g_{n-1}(x,y), \text{if } (x,y) \in A \setminus \tilde{A} \end{cases} \tag{33}$$

Expressions (31) - (33) define the iterative process, which allows to obtain the best approximation for the signal $f(x, y)$ and its spectrum $F(u,v)$. It has been proven [29] for continuous signals that

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |F(u,v) - F_n(u,v)|^2 \, du dv < \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |F(u,v) - F_{n-1}(u,v)|^2 \, du dv .$$

This inequality leads us to the following conclusion

$$\lim_{n\to\infty} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |F(u,v) - F_n(u,v)|^2 \, du dv = 0 .$$

It means that $\lim_{n\to\infty} F_n(u,v) = F(u,v)$. In this case the following is also true: $\lim_{n\to\infty} f_n(x,y) = f(x,y)$. The last conclusion means that the iterative process defined by (31)-(33) is finite, and it means that this process is converging. The practice shows that this iterative process always converges quickly. Usually the process requires no more than 7-8 iterations for the stabilization. As a result, we obtain the following:

$$f_n(x,y) = \begin{cases} f(x,y), \text{if } (x,y) \in \tilde{A} \\ f(x,y) + \tilde{s}(x,y), \text{if } (x,y) \in A \setminus \tilde{A}, \end{cases} \tag{34}$$

where $\tilde{s}(x,y)$ is a noisy component and

$$f_n(x,y) - f_{n-1}(x,y) = \begin{cases} 0, \text{if } (x,y) \in \tilde{A} \\ \varepsilon, \text{if } (x,y) \in A \setminus \tilde{A}, \end{cases} \tag{35}$$

where ε is close to zero.

According to the equation (34) $\tilde{f}(x,y) = f_n(x,y)$ contains an additive noise in $A \setminus \tilde{A}$ subdomain. It means that to complete a process of spectrum and signal restoration, we have to remove the noise and to correct the highest frequencies, which may be distorted during the noise removal. There are many different filters, which may be used to solve this problem. The best filter to apply, both from the point of view of noise reduction, quality and preservation of the image boundaries, is MVF defined by (16). So, if $MVF[h(x,y)]$ is the filter (16) applied to the function $h(x, y)$, and $MVF_{fc}[h(x,y)]$ is the filter (19) applied to $h(x, y)$, we obtain the following formulas, which define the final correction of the super-resoluted image:

$$S(x,y) = \begin{cases} f(x,y), \text{ if } (x,y) \in \tilde{A} \\ MVF\,[\tilde{f}(x,y)], \text{if } (x,y) \in A / \tilde{A} \end{cases} \tag{36}$$

$$\tilde{S}(x,y) = \begin{cases} f(x,y), \text{ if } (x,y) \in \tilde{A} \\ MVF_{fc}[S(x,y)], \text{if } (x,y) \in A / \tilde{A} \end{cases} \tag{37}$$

According to the equations (36) and (37), the signal is preserved in the subdomain $\tilde{A}$, where it is originally defined.

It is also possible to extrapolate the image spectrum without the use of the original one in (31). To go in this way, the following correction has to be made in (31) - (33):

$$F_{n-1}(u,v) = \begin{cases} \Phi[f_{n-1}(x,y)]; u,v \in \{0,1,..., n-1\} \\ 0; \qquad u,v \in \{n,...,2n-1\} \end{cases} \tag{38}$$

This technique requires a few iterations more for the convergence of the iterative process (31) – (33), but at the same time a quality of the resulting image often can be better, especially for the Cosine spectrum.

Another important improvement to the proposed super-resolution algorithm is the following. It is well known that spatial limitation of the discrete images and therefore limitation of their discrete spectral representation leads to the appearance of so-called boundary effects at the results of frequency domain filtering. These boundary effects become apparent as series of stripes, which are often clearly visible at the regions of image boundaries. To eliminate this effect, it is possible to use a symmetrical extension of the original image in all directions (even extension of the image). Of course, this approach requires more resources and leads to the increase of the processing time. At the same time a quality improvement of the resulting image will be achieved.

Let us illustrate the proposed algorithm by the following example. The original 1024x1024 image (see Fig. 8a) has been subsampled four times using the extraction of the low-frequency part from its Fourier spectrum. Then the obtained 256x256 image (Fig. 8b) has been interpolated using several algorithms (the best result of interpolation is achieved using Lanczos interpolating filter, see Fig. 8c) and upsampled using several variations of the proposed super-resolution algorithm (the best result is presented in Fig. 8d). It is clearly visible that a visual quality of the image obtained using the super-resolution technique is better than a quality of the image obtained by interpolation. The image in Fig. 8d is sharper than one in Fig. 8c, and it is very difficult to detect some difference between the result of the super-resolution procedure and the original image. Of course, these images do not coincide with each other, but the standard deviation between them is minimal. The standard deviations between the original image and images obtained by different interpolation algorithms are the following:

- Bicubic interpolation – 3.98
- Sinc (Lanczos) interpolation (Fig. 8c) – 3.86
- Super-resolution (algorithm [30] with Fourier transform) – 3.93
- Super-resolution, algorithm described here, Cosine transform, MVH_{fc}, without even extension – 3.45
- Super-resolution, algorithm described here, Cosine transform, MVH_{fc}, with even extension (Fig. 8d) – 3.10.

The obtained results show efficiency of the proposed algorithm. Fig. 9 presents a comparison between average values of the Cosine spectrum amplitudes corresponding to the original (Fig. 8a), interpolated (Fig. 8c) and supersampled (Fig. 8d) images. It is clear from this comparison, why the super-resolution algorithm described here is better than the interpolation. We see that the presented algorithm restores a higher frequency part of the image spectra more effectively than the interpolation. The amplitude of the supersampled image spectrum is very close to the amplitude of the original image

spectrum especially in the highest frequency domain, while the amplitude of the interpolated image spectrum is close to zero.

(a) The original image

(b) 4-times original image downsampling

(c) Sinc (Lanczos) interpolation of the image (b)

(d) Super-resolution of the image (b) (4-times upsampling): sinc interpolation as starting approximation; Cosine transform; input image even extension; model (38) for the input image spectrum (without its preservation); 8 iterations for the process (31)-(35); MVF high frequency correction (37)

Fig. 8. Super-resolution: downsampling and upsampling, comparison with the interpolation

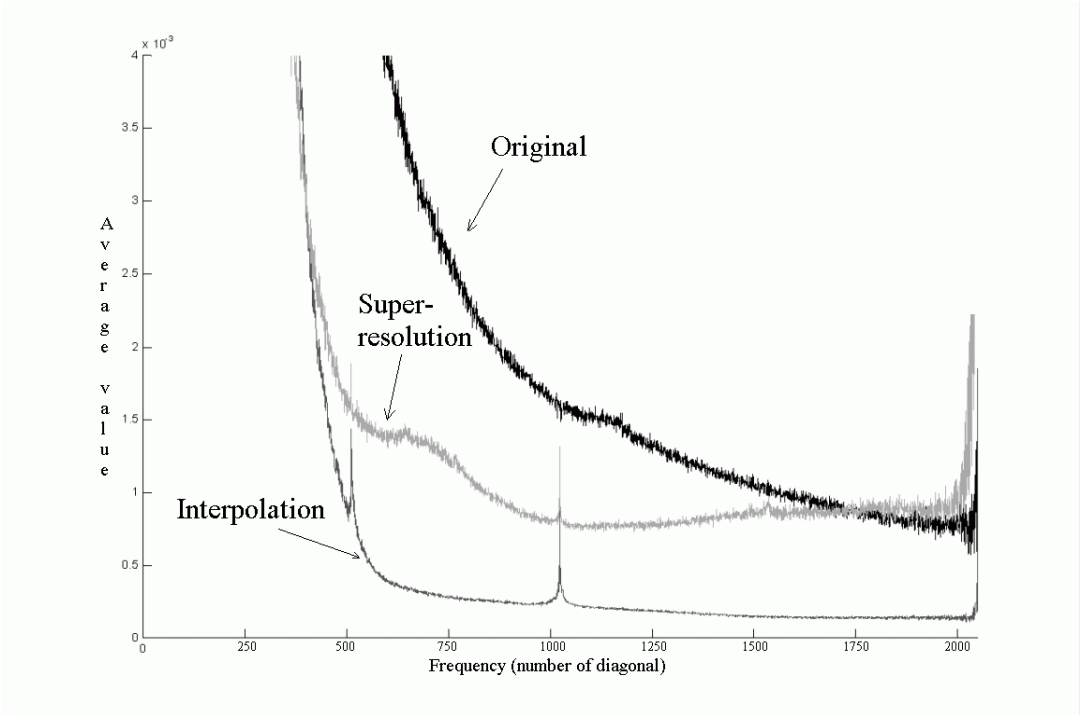


Fig. 9 Comparison between average values of the Cosine spectrum amplitude corresponding to the original (Fig. 8a), interpolated (Fig. 8b) and super-resoluted images (Fig. 8c)

6. Conclusions

We have presented the cellular neural networks based on the multi-valued and universal binary neurons (CNN-MVN and CNN-UBN). The results presented in this paper show high efficiency of CNN-MVN and CNN-UBN in image processing applications.

On the other hand, consideration of these neural networks involved development of a new family of nonlinear filters – nonlinear cellular neural filters. As it was shown, these filters are very effective and useful for the solution of such problems as noise reduction, frequency correction and edge detection. Multi-valued filter (MVF) may be successfully used for the noise reduction and for the solution of the frequency correction problem (extraction of image details). MVF comparisons to the order-statistic filters show its advantages for noise reduction. At the same time it was shown that MVF could be effectively combined with other nonlinear filters (for example, with the same order-statistic filters) to obtain better results. The comparisons of MVF frequency correction algorithms to unsharp masking also show advantages of MVF for extraction of image details and image sharpening.

It is shown that cellular neural Boolean filters (CNBF) are a very good mean for the solution of edge detection problem. The precise edge detection algorithm based on CNBF with a 5x5 window is presented. Advantages of precise edge detection in comparison to the classical edge detectors are considered.

It is also shown that the described filters may be used for the solution of the super-resolution problem, to overcome the results obtained by interpolation algorithms and the results obtained without the use of multi-valued filters.

ACKNOWLEDGEMENTS

The work was completely supported by Neural Networks Technologies Ltd. (Israel). Special thanks to Mr. David Halfon, the President of this company.

REFERENCES

1. I. Aizenberg, N.Aizenberg, T.Bregin., C.Butakoff. and E.Farberov “Image Processing Using Cellular Neural Networks Based on Multi-Valued and Universal Binary Neurons”, *Proceedings of IEEE 2000 International Workshop on Neural Networks for Signal Processing, Sydney, Australia, December 13-15, 2000*, IEEE Computer Society Press, 2000, pp. 557-567.

2. L.O.Chua and L.Yang, "Cellular neural networks: Theory", *IEEE Transactions on Circuits and Systems.* Vol. CAS-35, No 10, 1988, pp. 1257-1290.
3. T.Roska and J.Vandewalle (ed.) *"Cellular Neural Networks",* New York: John Wiley & Sons, 1993.
4. C.-C.Lee and.J.Pineda de Gyvez "Color Image Processing in a Cellular Neural -Network Environment", IEEE *Trans. On Neural Networks,* vol. NN-7, No 5, 1996, pp. 1086-1088.
5. S.Haykin *Neural Networks. A Comprehensive Foundation. Second Edition.* New York: Macmillan College Publishing Company, 1999.
6. H.Harrer and J.A.Nossek "Discrete-time cellular neural networks", *International Journal of Circuit Theory and Applications*, vol. CTA-20, 1992, pp. 453-467.
7. I.Aizenberg and N.Aizenberg "Application of the neural networks based on multi-valued neurons in image processing and recognition", *SPIE Proceedings*, Vol. 3307, 1998, pp. 88-97.
8. I.Aizenberg, N.Aizenberg and J.Vandewalle "*Multi-valued and universal binary neurons: theory, learning, applications*". Boston/Dordrecht/London: Kluwer Academic Publishers, 2000.
9. I.Aizenberg "Processing of noisy and small-detailed gray-scale images using cellular neural networks" *Journal of Electronic Imaging*, vol. EI-6, No 3, 1997, pp. 272-285.
10. I.Aizenberg, N.Aizenberg, S.Agaian, J.Astola and K.Egiazarian "Nonlinear cellular neural filtering for noise reduction and extraction of image details", *SPIE Proceedings*, Vol. 3646, 1999, pp.100-111.
11. C.Rekeczky, T.Roska and A.Ushida "CNN Based Self-Adjusting Nonlinear Filters" *Proc. of the Fourth IEEE International Workshop on Cellular Neural Networks and their Applications*, Seville, Spain, June 1996, pp. 309-314.
12. A.Zarandy,. A.Stoffels, T.Roska, and L.O.Chua "Morphological Operations on the CNN Universal Machine", *Proceedings of the Fourth International Workshop on Cellular Neural Networks and their Applications,* Seville, Spain, June 24-26, 1996, pp. 151-156.
13. T.Szirany, J.Zerubia and D.Geldreich "Cellular Neural Network for Markov Random Field Image Segmentation", *Proceedings of the Fourth International Workshop on Cellular Neural Networks and their Applications,* Seville, Spain, June 24-26, 1996, pp.139-144.
14. D.L.Vilarino, D.Cabello, M.Balsi and V.M.Brea "Image Segmentation Based on Active Contours using Descrite-Time Cellular Neural Networks", *Proceedings of the Fifth IEEE International Workshop on Cellular Neural Networks and their Applications*, London, UK, April 14-17, 1998, pp. 331-336.
15. K.R.Crounse, T.Roska, and L.O.Chua "Practical Halftoning on the CNN Universal Machine", *Proceedings of the Fifth IEEE International Workshop on Cellular Neural Networks and their Applications,* London, UK, April 14-17, 1998, pp. 337-342.
16. J.Astola, and P.Kuosmanen *Fundamentals of non-linear digital filtering.* New York: CRC Press, Boca Raton, 1997.
17. S.Agaian, J.Astola and K.Egiazarian *Binary polynomial transforms and nonlinear digital filters.* New York/Boston/Hong-Kong: Marcel Dekker, Inc.,1995.
18. I.Pitas and A.N.Venetsanopoulos *Nonlinear digital filters: Principles and Applications*. Boston: Kluwer Academic Publishers, 1990.
19. X.Z.Sun, and A.N.Venetsanopoulos "Adaptive Schemes for Noise Filtering and Edge Detection by use of Local Statistics", *IEEE Transactions on Circuits and Systems*, vol. CAS-35, 1988, pp. 57-69.
20. A.C.Bovik, T.S.Huang, and D.C.Mudson A Generalization of Median Filtering Using Linear Combinations of Order Statistics', *IEEE Transactions on Acoustic, Speech, Signal Processing* vol. ASSP-31, pp. 1342-1349, 1983.
21. B.I.Justusson Median Filtering: Statistical Properties. *Topics in applied Physics - 43, Two-dimensional Digital Signal Processing II,* (Ed. T.S.Huang), Berlin: Springer-Verlag, 1981, pp. 161-196.
22. I.Pitas, and A.N.Venetsanopoulos Nonlinear Mean Filters in Image Processing. *IEEE Transactions on Acoustic, Speech, Signal Processing*, vol. ASSP-34, 1996, pp. 573-584.
23. N.Aizenberg and I.Aizenberg "CNN Based on multi-valued neuron as a model of associative memory for gray-scale images", *Proceedings of the Second IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-92),* Technical University Munich, Germany, October 14-16, 1992, pp.36-41.
24. B.E.Shi "Order Statistic Filtering with Cellular Neural Networks", *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications*, Rome, Italy, December 1994, pp.441-444.
25. T.P.Belikova "Simulation of the linear filters in the problems of medical diagnostics", *Digital Image and Fields Processing in the Experimental Researches,* (I. Ovseevich - Ed.), Moscow: Nauka Publisher House, 1990, pp. 130-152.
26. W.KPratt "*Digital Image Processing. Second Edition*" New York: John Wiley & Sons, 1992.
27. J.R.Jensen "*Introductory Digital Image Processing*: A Remote Sensing Perspective" Englewood Cliffs, New Jersey: Pretince-Hall, 1996.
28. R.M.Haralick "Statistics and Structural Approach to Texture", *Proceedings IEEE*, vol. 67, No 5, 1979, pp. 786-804.
29. M.Shaker Sabri and W.Steenart "An approach to band-limited signal extrapolation: the extrapolation matrix", IEEE *Trans.Circuits Syst*., vol. CAS-25, No 2, 1978, pp. 74-78.
30. H. Stark "*Applications of optical Fourier transforms*", New York: Academic Press, 1982.

AUTHORS:

IGOR N. AIZENBERG

Dr. Igor Aizenberg graduated from the State University of Uzhgorod (USSR) with M.D. in 1982, received Ph.D. in the Computing Center of the Academy of Sciences of the USSR (Moscow, USSR) in 1986. In 1982-1990 he was with the Institute for Information Transmission Problems of the Academy of Sciences of the USSR (Moscow) as a researcher. In 1990-1996 and 1998-1999 he was with the Department of Cybernetics in the State University of Uzhgorod (USSR-Ukraine) as Associate Professor (Docent). In 1996-1998 he was with the Department of Electrical Engineering (ESAT) of the Catholic University of Leuven (K.U. LEUVEN, Belgium) as a visiting researcher. Since 1999 he is a head of the research department of the company "Neural Networks Technologies Ltd." (Israel). He is the author of about 80 papers and one book in the field of neural networks and their applications in image processing and pattern recognition. He also holds 4 patents.

CONSTANTINE V.BUTAKOFF

Constantine Butakoff graduated from the State University of Uzhgorod (Ukraine) with M.D. in 1999. In 1998-1999 he was with the company "Neurotechnologies" (Ukraine) as a researcher-programmer. Since 1999 he is with the company "Neural Networks Technologies Ltd." (Israel) as a researcher. He is the author of 7 papers in the field of neural networks and their applications in image processing and pattern recognition.