Non-Linear Image Representation Based on IDP with NN

ROUMEN KOUNTCHEV
Department of Radiocommunications
Technical University - Sofia,
Bul. Kl Ohridsky, 8
Sofia 1000, BULGARIA
rkountch@tu-sofia.bg
www. tu-sofia.bg

STUART RUBIN
Space and Naval Warfare Systems Center
San Diego (SSC-PAC)
USA
stuart.rubin@navy.mil

MARIOFANNA MILANOVA
Department of Computer Science
UALR, 2801 S. University Ave.
Little Rock, Arkansas 72204
USA
mgmilanova@ualr.edu

VLADIMIR TODOROV
T&K Engineering
Mladost 3, POB 12
Sofia 1712
BULGARIA
todorov_vl@yahoo.com

ROUMIANA KOUNTCHEVA
T&K Engineering
Mladost 3, POB 12
Sofia 1712
BULGARIA
kountcheva_r@yahoo.com

Abstract: - In this paper is offered a method for non-linear still image representation based on pyramidal decomposition with a neural network. This approach is developed by analogy with the hypothesis for the way humans do image recognition using consecutive approximations with increasing similarity. A hierarchical decomposition, named Inverse Difference Pyramid (IDP), is used for the image representation. The approximations in the consecutive decomposition layers are represented by the neurons in the hidden layers of the neural networks (NN). This approach ensures efficient description of the processed images and as a result – a high compression ratio. This new way for image representation is suitable for various applications (efficient compression, multi-layer search in image databases, etc.).

Key-Words: - Non-linear image representation, pyramidal decomposition, neural networks

1 Introduction

The solutions of the problems concerning the efficient still image representation depend on the application: medicine, digital libraries, electronic galleries, geographic information systems, documents archiving, digital communication systems, etc.

Two basic forms for digital image presentation are widely used – the primary (not compressed) and the secondary forms, obtained from the primary one and based on some kind of compression [1-6]. The primary form for digital image presentation is the matrix. The secondary forms are based on various techniques: multi-dimensional vectors, pyramids, linear prediction with fixed coefficients, linear orthogonal transforms, discrete wavelet transforms, fractal transforms, tree structures, algebraic models, models for visual information perception, etc.

Another form is the vector representation, used for image compression with vector quantization and for image analysis and recognition based on 3-dimensional color features, textural features, K-dimensional color histograms, multi-dimensional shape features, RST-invariant features, R-tree, etc.

The pyramidal representation describes the image with progressively increased resolution, which corresponds to the layers of the Gaussian-Laplacian Pyramid. The derivatives of this representation are the Reduced Sum/Difference pyramid; the S-transform pyramid, the Hierarchy-Embedded Differential Pyramid; the Least Square Pyramid, the Morphological Pyramid, etc. This group of pyramids is called over-complete because the data needed for the full pyramid representation is larger than that for the non-compressed image. The Orthogonal pyramids are non-over-complete: they are based on Wavelets or Contourlets functions and have higher efficiency and computational complexity than pyramids from the first group.

The spectral image representation is based on orthogonal transforms: statistical (Karhunen-Loeve Transform, Principle Component Analysis, Independent Component Analysis, Singular Value Decomposition) and determined (Discrete Fourier

Transform, Discrete Cosine Transform, Walsh-Hadamard Transform, Hartley Transform, Lapped Orthogonal Transform, Slant Transform, etc.). In this group could be included the new algebraic image transform based on 2D angular windowing functions, which is suitable for the synthetic shape local phase and orientation evaluation.

The knowledge-based models for image representation are used mostly in the systems for Visual Information Retrieval. The main approach for image representation used is the pyramid model of 4 layers, which contain correspondingly: the primary matrix, the features vectors, the description of the relations between the features and the semantic image structure. One more approach for image representation is the perceptual one, based on anisotropic filtration controlled by the Human Visual System (HVS) visual attention model.

The requirements on the object representation models are contradictory: minimum features number and invariability together with exact description, low computational complexity, etc. The methods, described above, solve these problems to some degree, but can't achieve the best balance, because they are not flexible enough (they do not involve learning and feedback procedures). The state-of-the-art analysis shows that there still exist unexplored possibilities for a wider cognitive approach in the creation of the object representation model and the context-based retrieval.

The main disadvantages of the image representation methods, described above, are the relatively poor use of the image content knowledge obtained through learning, and the too complicated cognitive structures used.

A group of methods for image representation, based on the use of artificial neural networks (NN) [7-15] had recently been developed. Unlike the classic methods, this approach is distinguished by higher compression ratios because the NN training is performed together with the coding. The results already obtained show that these methods can not successfully compete with the still image compression standards, JPEG and JPEG2000 [3]. For example, the Adaptive Vector Quantization (AVQ), based on SOM NN [8, 13], requires the use of code books of too many vectors, needed to ensure high quality of the restored image and this results in lower compression.

In this paper is presented one new approach for non-linear image representation based on pyramidal decomposition with neural network. This approach is based on the analogy with the hypothesis for the way humans do image representation using consecutive approximations with increasing resolution. Significant elements of the new representation include the use of feedback, which provides iterative change of the cognitive models' parameters in accordance with the data similarity results obtained.

2 General Principles of the IDP Decomposition

Mathematically the digital halftone image is usually represented as a matrix of size H×V, whose elements $b(i, j)$ correspond to the image pixels; i and j define the pixel position as a matrix row and column and b is the pixel brightness. The halftone image matrix [B(i,j)] is defined as:

$$[B(i,j)]=\begin{bmatrix} b(0,0) & b(0,1) & \cdots & b(0,H-1) \\ b(1,0) & b(1,1) & \cdots & b(1,H-1) \\ \vdots & \vdots & & \vdots \\ b(V-1,0) & b(V-1,1) & \cdots & b(V-1,H-1) \end{bmatrix} \quad (1)$$

The essence of the IDP decomposition for 8-bit grayscale images is presented as follows. First, the digital image is processed with two-dimensional (2D) Direct Orthogonal Transform (DOT) using a limited number of coefficients. The values of the coefficients, calculated in result of the transform, constitute the lowest pyramid level. The image is then restored with the Inverse Orthogonal Transform (IOT) using only the retained coefficients' values. The first (coarse) approximation of the original image is obtained, which is then subtracted pixel by pixel from the original one. The difference image, which is of same size as the original, is divided into 4 sub-images and each is then processed again with the 2D DOT. The values of the so-calculated coefficients constitute the second pyramid layer. The processing continues in similar way with the next pyramid layers. The set of coefficients of the orthogonal transform, retained for every pyramid layer, can be different and define the restored image quality. The image decomposition stops when the needed quality for approximating the image is obtained – usually earlier than the last possible pyramid layer. The values of the coefficients obtained as a result of the orthogonal transform from all pyramid layers are then sorted in accordance with their spatial frequency, scanned sequentially, and losslessly compressed.

The IDP decomposition used in practice, is usually "truncated", i.e. it starts from some of the higher layers and for this, the discrete original image

is divided into blocks (sub-images), represented as matrices $[B(2^n)]$ of size m×m ($m=2^n$). After that, each block is represented by an individual pyramid, whose elements are defined by the already described recursive calculations. The number p of the IDP layers is in the range $0 \le p \le n-1$. The case p = n-1 corresponds to complete pyramidal decomposition of maximum number of layers, for which the image is restored without error (all decomposition components are used).

In correspondence with the described principle, the matrix $[B_{k_0}]$ of one image block could be represented as a decomposition of (n+1) components:

$$[B_{k_0}] = [\tilde{B}_{k_0}] + \sum_{p=1}^{n-1} [\tilde{E}_{k_{p-1}}] + [E_{k_n}] \qquad (2)$$

for $k_p=1,2,..,4^pK$ and p=0,1,...,n-1.

Here k_p is the number of the sub-matrices of size $m_p \times m_p$ ($m_p=2^{n-p}$) in the IDP layer p; the matrices $[\tilde{B}_{k_0}]$ and $[\tilde{E}_{k_{p-1}}]$ are the corresponding approximations of $[B_{k_0}]$ and $[E_{k_{p-1}}]$; $[E_{k_n}]$ is the matrix, which represents the decomposition error in correspondence with Eq. (2), for the case, when only the first n components are used.

The matrix $[E_{k_{p-1}}]$ of the difference sub-block k_{p-1} in the IDP layer p is defined as:

$$[E_{k_{p-1}}] = [E_{k_{p-2}}] - [\tilde{E}_{k_{p-2}}], \qquad (3)$$

for p = 2,3,..., n-1. In this case p = 1:

$$[E_{k_0}] = [B_{k_0}] - [\tilde{B}_{k_0}] \qquad (4)$$

The matrix $[E_{k_{p-1}}]$ of the difference sub-block in the layer p is divided into 4^pK sub-matrices $[E_{k_p}]$ and for each is then calculated the corresponding approximating matrix $[\tilde{E}_{k_p}]$. The submatrices $[\tilde{E}_{k_p}]$ for $k_p=1,2,...,4^pK$ define the next decomposition component (p+1), represented by Eq. (2). It is necessary to calculate the new difference matrix for this and then perform the same operations again, following the already described order.

3 Image Representation with NN-Controlled IDP

The new method for image representation is based on one modification of the IDP decomposition, in which the direct and inverse transforms in all layers are performed by using 3-layer neural networks with error back propagation (BPNN) [7]. A 3-layer BPNN structure of the kind $m^2 \times n \times m^2$ was chosen for this application, as shown in Fig. 1. The input layer has m^2 elements, which correspond to the input vector components; the hidden layer has n elements for $n < m^2$, and the output layer has m^2 elements, which correspond to the output vector components. The input m^2-dimensional vector is obtained as a result of the transformation of the elements m (i, j) of each image block of size m × m into one-dimensional massif of length m^2.

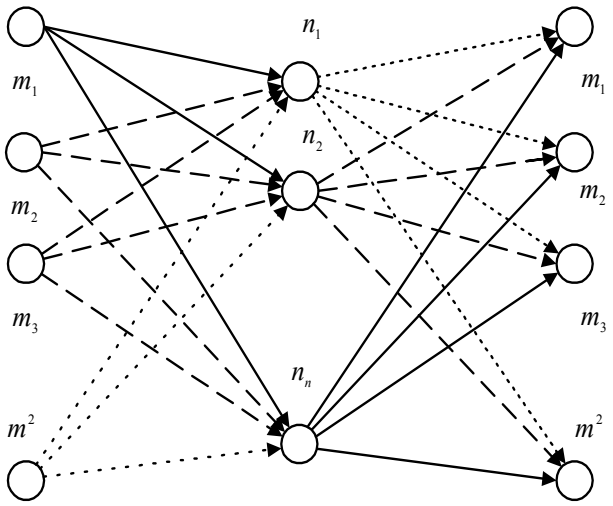

Fig. 1. A 3-layer BPNN with $n_n < m^2$ neurons in the hidden layer and m^2 neurons in the input and output layers

In order to obtain higher efficiency, the processed image is represented by the sequence of m^2-dimensional vectors $\vec{X}_1, \vec{X}_2, ..., \vec{X}_K$ are then transformed into the n-dimensional corresponding vectors, $\vec{h}_1, \vec{h}_2, ..., \vec{h}_K$. The components of the vectors $\vec{h}_k$ for k=1,2,..K, correspond to the neurons in the hidden layer of the trained 3-layer BPNN. In the output NN layer, the vector $\vec{h}_k$ is transformed back into the m^2-dimensional output vector $\vec{Y}_k$, which approximates the input vector, $\vec{X}_k$. The approximation error depends on the training algorithm and on the BPNN parameters. The training vectors, $\vec{X}_1, \vec{X}_2, ..., \vec{X}_K$ at the BPNN input for the decomposition layer p = 0 correspond to the starting image blocks. The algorithm of Levenberg-Marquardt (LM) was chosen for training [8], which ensures good efficiency in cases when high

accuracy is not required - i.e., it is suitable for the presented approach.

The parameters of the 3-layer BPNN define the relations between the inputs and the neurons in the hidden layer and between the neurons from the hidden and the output layers. These relations are represented by using weight matrices and vectors, which contain threshold coefficients and by functions for non-linear vector transforms. The relation between the input m^2-dimensional vector $\vec{X}_k$ and the corresponding n-dimensional vector $\vec{h}_k$ in the hidden BPNN layer for the IDP layer p = 0 is:

$$\vec{h}_k = f([W]_1 \vec{X}_k + \vec{b}_1) \text{ for } k = 1,2,..K, \quad (5)$$

where, $[W]_1$ is the matrix of the weight coefficients of size $m^2 \times n$, which is used for the linear transform of the input vector $\vec{X}_k$; $\vec{b}_1$ is the n-dimensional vector of the threshold coefficients in the hidden layer and f(x) is a linear activating sigmoid function defined by the relation:

$$f(x) = 1/(1 + e^{-x}). \quad (6)$$

In result the network performance becomes partially non-linear and this dependence is stronger when x is outside the range [-1.5, +1.5].

The relation between the n-dimensional vector $\vec{h}_k$ of the hidden layer and the m^2-dimensional BPNN vector $\vec{Y}_k$ from the IDP layer p = 0, which approximates $\vec{X}_k$, is defined in accordance with Eq. (5) as follows:

$$\vec{Y}_k = f([W]_2 \vec{h}_k + \vec{b}_2) \text{ for } k = 1,2,..K \quad (7)$$

where $[W]_2$ is a matrix of size $n \times m^2$ representing the weight coefficients used for the linear transform in the hidden layer of the vector $\vec{h}_k$, and $\vec{b}_2$ is the m^2-dimensional vector of the threshold coefficients for the output layer. Unlike the pixels in the halftone images, whose brightness is in the range $[0,255]$, the components of the input and output BPNN vectors are normalized in the range $x_i(k), y_i(k) \in [0,1]$ for i=1, 2,.., m^2. The components of the vector, which represents the neurons in the hidden layer $h_j(k) \in [0,1]$, for j=1, 2,.., n are placed in the same range because they are defined by the activating function, $f(x) \in [0,1]$. The normalization is necessary, because it enhances the BPNN efficiency [8].

The image representation with IDP-BPNN comprises two consecutive stages:

1) BPNN training;
2) Coding of the obtained output data.

For the BPNN training in the IDP layer p = 0, the vectors $\vec{X}_k$ are used as input and reference ones, which are compared to the corresponding output vectors. The comparison result is used to correct the weight and the threshold coefficients so as to obtain a minimum MSE (mean square error). The training is repeated until the MSE value for the output vectors becomes lower than some predefined threshold.

The vectors obtained after dividing the difference block $[E_{k_{p-1}}]$ (or sub-block) into 4^pK sub-blocks and their transformation into corresponding vectors are used for the training of the 3-layer BPNN in the next IDP layers (p > 0). The BPNN training for each layer p > 0 is performed in the way already described for the layer p = 0.

In the second stage, the vectors in the hidden BPNN layers for all decomposition layers are coded losslessly with entropy coding [6]. The block diagram of the pyramid decomposition for one block of size m×m with 3-layer BPNN for decomposition layers p = 0, 1, 2 and entropy coding/decoding is shown in Fig. 2.

When the BPNN training is finished for each layer, p are defined for the corresponding output weight matrix $[W]_p$ and the threshold vector $[b]_p$. The coded data, which are later transferred to the decoder, comprise the following information.

- The *vector of the threshold coefficients* for the neurons in the output NN layer (common for all blocks in the layer p);
- The *matrix of the weight coefficients,* which represents relations between the neurons in the hidden layer towards the output BPNN layer (common for all blocks in the layer p);
- The *vector of the neurons in the hidden BPNN layer*, personal for each block in the layer p.

In the decoder is performed the entropy decoding (ED) of the compressed data. After that the BPNN in the layer p is initialized setting the values of the threshold coefficients for the neurons in the output layer and of the weight coefficients for the neurons, connecting the hidden and the output layers.

At the end of the decoding, the vector of neurons in the hidden BPNN layer, has each block transformed into the corresponding output vector. The so obtained output vectors are used for the restoration of the processed image.

4 IDP-BPNN Algorithm

The IDP-BPNN algorithm for grayscale images comprises the steps, given below:

Coding:

Step 1. The input halftone image is represented as a matrix of size H×V, 8bpp;

Step 2. The input image matrix is divided into K blocks of size m×m ($m=2^n$). The value of m is selected so that to retain as much as possible the correlation between the block pixels (for big images of size 1024×1024 or larger, this block is usually 16×16 or 32×32, and for smaller images it is 8×8);

Step 3. The matrix of every block (sub-block) of $m^2/2^p$ elements in the layer p is transformed into the input vector of size $(m^2/2^p)\times 1$. The so obtained 4^pK input vectors build a matrix of size $(m^2/2^p)\times 4^pK$, which is used for the BPNN training and as a matrix of the reference vectors, which are then compared with the BPNN output vectors;

Step 4. The matrix used for the BPNN training is normalized – transforming its range [0,255] into [0,1];

Step 5. The training function of the NN is defined;

Step 6. The BPNN working function is set, i.e., the mse function;

Step 7. The criterion for the end of the BPNN training is defined by setting the MSE threshold value or by setting the possible maximum number of training cycles.

At the end of the training, the following information is saved in a special file:

- The neurons of the hidden layer, which in general are different for every block (sub-block);
- The threshold coefficients for the output layer;
- The matrix of the weight coefficients between the hidden and the output BPNN layers.

Step 8. The data, described in Step 7 is losslessly coded (entropy coding) and is saved in a special file, which contains the compressed data for each of the consecutive layers.

Step 10. The decomposition stops when the module of the approximation error (Eq. 3) becomes smaller than the pre-defined threshold value, or when the highest possible (or set) layer is processed.

Step 11. The coded data for all decomposition layers is gathered in a common file.

Decoding:

Step 1. The common file is loaded in the decoder;

Step 2. The data, which corresponds to the consecutive layers, is separated;

Step 3. For every decomposition layer p are decoded: the values of the neurons in the hidden layer for each block (sub-block), the threshold coefficients and the matrix of the weight coefficients for the corresponding output BPNN layer;

Step 4. The components of the vector for each block (sub-block) in the output BPNN layer are restored;

Step 5. The output BPNN vector is transformed into the block (sub-block) matrix;

Step 6. The range [0, 1] of the matrix elements is transformed back into [0, 255];

Step 7. The matrices from all decomposition layers $p = 0,1,..,p_{max}$ are summed, and in result the restored image is obtained.

For the image representation in accordance with the IDP-BPNN method was developed new format, which contains information about the 3 main BPNN components for every layer, as follows:

- The vector of the values of the neurons in the hidden layer – personal for each block/sub-block;
- The vector of the threshold coefficients for the output layer – common for all blocks/sub-blocks;
- The matrix of the weight coefficients for the output layer - common for all blocks/sub-blocks.

5 Experimental results

The presented IDP-BPNN algorithm was simulated with MATLAB. The simulation (coding/decoding) was accomplished by following the already described steps. For the experiments with the IDP-BPNN algorithm were used test images of size 224×352, 8 bpp (i.e. 78 848 B). Some of the original test images are shown in Fig. 3 and the corresponding images obtained after the processing – in Fig. 4.

In the initial IDP layer p = 0 the image was divided into K blocks of size 8×8 pixels, (K=1232). At the BPNN input for the layer p = 0 was passed the training matrix of the input vectors of size 64×1232 = 78 848. In the hidden BPNN layer, the size of each input vector was reduced from 64 to 8.

The restoration of the output vector in the decoder was performed using these 8 components, together with the vector of the threshold values and the matrix of the weight coefficients in the BPNN output layer. For the layer p = 0, the size of the data obtained was 83 456 B, i.e. - larger than that of the original image (78 848 B). As it was already pointed out, the data has high correlation and this ensures efficient compression with entropy coding. For example, the compressed data size for the same

layer (p=0) of the test image "*Grayscale_forest010032.bmp"* is 1510 B (the result is given in Table 1). Taking into account the size of the original image, the compression ratio CR=52,21 is calculated.

The Peak Signal to Noise Ratio (PSNR) for the first test image *Grayscale_forest010032.bmp* for p=0 (Table 1) is PSNR = 23,45 dB. In the same table are given the compression ratios obtained with IDP-BPNN for other 18 test images of same size (224×352). It is easy to see that for the mean compression ratio CR = 52,13 is obtained PSNR>22,52 dB, i.e. the visual quality of the restored test images is suitable for various applications. Better image quality is obtained when the next pyramid layers are added.

In Table 2 are given the results for the group of 18 test images of the kind "*forest*" after IDP-BPNN compression, implemented with MATLAB. The results obtained (Table 2) show that IDP-BPNN method performance for group of similar test images surpasses that for single images (Table 1). In the same table are given the results obtained using the JPEG 2000-based compression for the same test images. The results show that the CR and the PSNR are a little higher for IDP-BPNN than for JPEG 2000.

Table 1. Results obtained for 18 test images from the group named "forest" after IDP-BPNN compression (78 848 Bytes for each original image).

Image No.	CR	PSNR [dB]	Bits per pixel (bpp)	Compressed file size [B]
1	52.21	23.45	0.1532	1510
2	52.42	23.05	0.1526	1504
3	51.53	19.10	0.1552	1530
4	50.54	17.14	0.1583	1560
5	52.35	22.71	0.1528	1506
6	52.35	19.72	0.1528	1506
7	53.06	22.28	0.1508	1486
8	52.49	23.40	0.1524	1502
9	52.85	31.63	0.1514	1492
10	51.80	21.55	0.1544	1522
11	52.21	21.92	0.1532	1510
12	52.21	22.28	0.1532	1510
13	52.35	19.87	0.1528	1506
14	51.73	19.50	0.1546	1524
15	52.43	22.31	0.1526	1504
16	52.08	23.67	0.1536	1514
17	52.49	28.07	0.1524	1502
18	51.20	23.63	0.1563	1540
Mean	52.13	22.52	0.1535	1513

Thus, the results obtained show higher compression ratio CR = 63,21 for a group of images than the mean compression ratio CR=52,13 obtained after individual processing of the same test images.

The NN architecture used for the experiments comprises 64 neurons in the input layer, 8 neurons in the hidden layer, and 64 neurons in the output layer for the zero decomposition level. The chosen ratio for the input vectors was correspondingly: 80% for Training; 10 % for Validation, and 10% for Testing.

Table 2. Results obtained for a group of 18 test images of the group "forest" after IDP-BPNN compression (1419264 B for the whole group).

Image	Image size	CR	PSNR [dB]	File size [B]
Original	6336×224	-	-	-
IDP-BPNN	6336×224	63.21	21,35	22 452
JPEG2000	6336×224	63.00	21,02	22 528

In Fig. 5 are shown the restored images of the test image "Boy" after compression using 5 methods: IDP-BPNN, JPEG and JPEG 2000. MATLAB (im2jpeg, imwrite) and Lura Smart Compress (JPEG and JPEG 2000) were used for these experiments.

The results obtained for several test images are given in Table 3.

In Fig. 6 are shown the results (restored images and enlarged parts) for the test image "Boy" after compression with JPEG 2000 and IDP-BPNN. The conditions in both cases are similar: CR = 60 and PSNR = 29 dB. It is easy to notice that for same compression and PSNR the image processed with IDP-BPNN is not as blurred as that, processed with JPEG 2000, i.e. the quality evaluation with PSNR in this case does not correspond to human perception.

The IDP-BPNN method is suitable for coding of uncompressed images represented in RGB sampling format 4:4:4. In this case, it is necessary to transform the image first into the YC_rC_b sampling format 4:2:0 and then apply the IDP-BPNN on each component individually. The decoding is performed in reverse order. The experiments with such images confirmed the efficiency of the method.

In Fig. 7 are shown the results for the color test image "Lena" (512×512 pixels, 24 bpp) – the restored images obtained for same compression (CR>60) with the IDP-BPNN method and with JPEG.

The experiments with high-resolution (satellite) images are very promising because they give high compression ratio for retained visual quality (PSNR

higher than 30 dB) after decoding. For example, an image of size 8192×8192 pixels was restored with retained visual quality after CR > 120 [16], while equivalent visual quality was obtained for CR<100 with JPEG 2000.

6 Conclusion

In this paper is presented one new approach for non-linear image representation based on NN-controlled pyramidal image decomposition. The method is similar to the way in which humans recognize images – starting with a coarse approximation and continuing with successively closer approximations until the needed result is obtained. Another analogy is provided by the training procedure – just like humans, the method requires some training before recognition – the better the training, the better the recognition results.

The method is asymmetric (the coder is more complicated than the decoder) and this determines it mostly in application areas, which do not require real time processing – i.e. applications, for which the training time is not crucial.

The experimental results show that for the same compression, the image approximations obtained using the IDP-BPNN method have better visual quality than the compression standard JPEG 2000. The method could be successfully applied for the efficient representation of images and for layered image search in large databases.

The future development of the method is aimed at the preliminary definition of the basic NN parameters (i.e. the training to be done in advance). This is to be accomplished for some large image classes (medical images, texts, natural pictures, faces, fingerprints, etc.). This will solve the difficulties due to the main disadvantage of the IDP-BPNN method – the need for long training time and will permit real-time applications.

Acknowledgement

This work was supported by the National Fund for Scientific Research of the Bulgarian Ministry of Education and Science, Contract VU-I 305.

References:

[1] K. Rao, P. Yip, Eds. *The Transform and Data Compression Handbook.* CRC Press LLC, 2001.

[2] M. Barni, Ed., *Document and Image Compression*. CRC Press Taylor and Francis Group, 2006.

[3] T. Acharya, P. Tsai, *JPEG2000 Standard for Image Compression*, John Wiley and Sons, 2005.

[4] R. Gonzales, R. Woods, *Digital Image Processing*, Prentice Hall, 2002.

[5] R. Kountchev, V. Haese-Coat, J. Ronsin. Inverse Pyramidal Decomposition with Multiple DCT, *Signal Processing: Image Communication*, Elsevier 17, pp. 201-218, 2002.

[6] D. Salomon, *Data Compression*, Springer, 2004.

[7] St. Perry, H. Wong, L. Guan. *Adaptive Image Processing: A Computational Intelligence Perspective*, CRC Press LLC, 2002.

[8] Y. H. Hu, J. N. Hwang (Eds.), *Handbook of Neural Network Signal Processing*, CRC Press LLC, 2002.

[9] R. Dony, S. Haykin, Neural Network Approaches to Image Compression, *Proceedings of the IEEE*, vol. 23, No. 2, pp. 289-303, 1995.

[10] A. Namphol, et al., Image Compression with a Hierarchical Neural Network*, IEEE Trans. on Aerospace and Electronic Systems*, vol. 32, No. 1, pp. 327-337, 1996.

[11] S. Kulkarni, B. Verma, M. Blumenstein, Image Compression Using a Direct Solution Method Based Neural Network*, 10-th Australian Joint Conference on AI*, Perth, Australia, pp. 114-119, 1997.

[12] J. Jiang. Image Compressing with Neural Networks - A survey, *Signal Processing: Image Communication*, Elsevier, vol. 14, No. 9, pp. 737-760, 1999.

[13] N. Kouda, et al., Image Compression by Layered Quantum Neural Networks, *Neural Processing Letters*, *16*, pp. 67-80, 2002.

[14] N. Hikal, R. Kountchev. A method for digital image compression with IDP decomposition based on 2D SOFM VQ. *International Journal on Graphics, Vision and Image Processing (GVIP),* Special Issue on Image Compression, www.icgst.com, 2007, pp. 37-42.

[15] V. Cherkashyn, R. Kountchev, D.-C. He, R. Kountcheva. Adaptive Image Pyramidal Representation, *IEEE Symposium on Signal Processing and Information Technology* (ISSPIT'08), Sarajevo, Bosnia and Herzegovina, Dec. 18-19, 2008, pp. 441-446.

[16] V. Cherkashyn, D.-C. He, R. Kountchev, Compression of High-resolution Satellite Images with Pyramidal Neural Network, *International Journal: Neural networks and Applications,* International Science Press, India (in press).

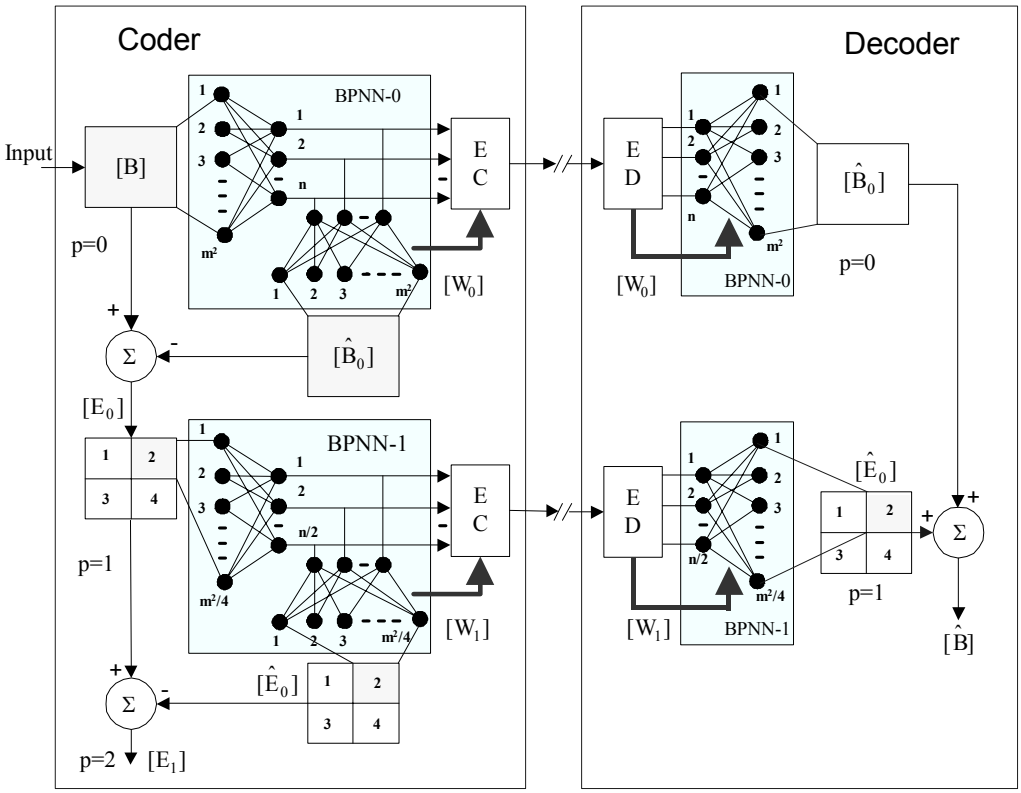

Fig. 2. Block diagram of the 2-layer inverse pyramidal image decomposition with 3-layer BPNN. Here $[b]_p$ – the vector of the threshold coefficients in the output layer for $p=0,1,2$; $[W]_p$ – the matrix of the weight coefficients between the hidden and the output BPNN layer for $p=0,1,2$.

Fig. 3. The original test images

Fig.4. The restored images after IDP-BPNN compression whit mean compression ratio CR = 52,13

Original

IDP-BPNN: p=0;CR=60.4;PSNR=29

JPEG (im2jpeg):CR=29.6;PSNR=28.89

JPEG (imwrite): CR=28.4 PSNR=29.34

LURA JPEG: CR=28.4 PSNR=29.33

LURA JPEG2000: CR=50 PSNR=29.15

Fig. 5. The restored image “BOY”, after compression with 5 methods

a b

c d

Fig. 6. The test image “Boy” after processing with JPEG 2000 (a,b) and IDP-BPNN(c,d)

a. Original test image "Lena" 512 x 512 pixels, 24 bpp
b. IDP - BPNN CR = 65,22; PSNR = 29.90 dB
c. JPEG CR = 66,49; PSNR = 28.97 dB)

Fig. 7 Results for the test image *Lena_color.bmp* (512×512 pixels, 24 bpp)

Table 3. Results for several test images after compression with different methods performed with MATLAB and Lura Smart Compress.

Image	AIDP-BPNN		MATLAB						LuraWave SmartCompress			
			JPEG (im2jpeg)		JPEG (imwrite)		JPEG 2000		Lura JPEG		Lura JPEG2000	
	CR	PSNR	CR	PSNR	CR	PSNR	CR	PSNR	CR	PSNR	CR	PSNR
Boy	60.40	29.05	29.6	28.99	28.47	29.34	25.23	28.97	28.48	29.33	50.04	29.15
Fruit	60.29	32.89	30.53	33.01	31.67	32.78	32.64	32.69	31.67	32.78	60.00	33.11
Vase	60.18	26.83	37.66	26.72	35.18	27.01	35.75	27.20	35.18	27.00	70.04	27.07
Clown	60.01	31.81	30.43	31.88	31.36	31.88	31.71	31.47	31.37	31.88	60.03	31.87
Peppers	60.23	30.94	37.63	31.17	36.80	31.17	38.39	30.70	36.81	31.16	80.02	30.85
Text1	60.23	18.69	19.09	19.00	20.12	18.89	17.86	18.35	22.37	18.23	30.02	18.21
Lena	59.57	29.15	30.48	29.53	30.75	29.53	32.10	29.20	30.75	29.52	60.03	29.31