Image analogies based filters for composite distortions

Milan Tuba, and Jelena Z. Tasic

***Abstract*—**Images from cameras on remote locations are often subject to multiple distortions caused by damaged optics or undesirable movements, while medical images originated by alternative sources like ultra-sound or x-rays are subject to natural noise and tissue penetration distortions. Information about exact sources of such distortions and their parameters are usually not available. This paper presents a method that uses examples to develop a single image processing filter that can restore images subject to multiple distortions without determining components of that combined distortion. Proposed method is based on image analogies and it requires one correct image before distortions and one image of the same object under distortions. The feedforward artificial neural networks and the resilient propagation learning algorithm are used to generate a single filter that successfully reverses all distortions and can be applied to all images from the same distorted source. Our method achieved very good results on different test examples so it can be considered a promising tool for multiple distortions corrections.

***Keywords*—**Image filters, Image analogies, Image processing, Artificial neural networks.

I. Introduction

DIGITAL images are nowadays used almost everywhere. Digital image processing is among most important research topics since for most applications some image processing is required. In many applications it is of crucial importance since without it images would be useless. Examples are many medical images originated by alternative sources like ultra-sound or x-rays [1], [2] or images from cameras on remote locations where it is impossible to correct problems, like damaged optics or undesirable movement, on the spot. If we know the sources and the nature of the distortions we can design mathematical tools, usually filters, to counter mentioned distortions [3]. However, if there are many such distortions and we do not have precise information about their nature and parameters, the problem becomes more difficult.

In this paper we propose a method of designing image processing filters based on image analogies. If we have one „good" image, before distortions and the same image with distortions, we develop a filter using neural networks that transforms the distorted image to the original without distortions. A single filter reverses all the distortions without separating them or even trying to determine them. Such a filter can later be used to process all images from the same source.

The rest of this paper is organized as follows. Section II is a review of image analogies research. Section III describes our proposed approach to design a single filter for removal of multiple distortions, while section IV examines the quality of our proposed algorithm on various examples where its robustness is proven.

II. Related Work

This paper's general idea – learning filters on the basis of a given sample – is based on Hertzmann's article [4] where the term image analogies is introduced: „Given a pair of images A and A' (the unfiltered and filtered source images, respectively), along with some additional unfiltered target image B, synthesize a new filtered target image B' such that:

$$A : A' = B : B' \quad (1)$$

Fig. 1 A : A' = B : B'

Fig. 2 B and expected B'

The similarity metric is based on approximation of Markov random field model and uses pixel values for learning of a chosen filter. The statistics of the selected pixel neighborhoods is used for calculating the relationship between the original and resulting pair of images. This algorithm uses a multi-layered representation of the images A, A' and B (Gaussian

Manuscript received October 22, 2012. Revised April 29, 2013.

The research was supported by the Ministry of Science, Republic of Serbia, Project No. III-44006

M. Tuba is with the Faculty of Computer Science, Megatrend University of Belgrade, Serbia, e-mail: tuba@ieee.org

J. Z. Tasic is with the Faculty of Mathematics, University of Belgrade, Serbia, e-mail: jelenatasicsd@yahoo.com.

pyramid), together with a feature vector as well as the approximate nearest neighbor method for speeding up the mapping process. Hertzmann uses the same framework for curve analogies [5], learning of curve style on the basis of a given example.

As opposed to Hertzmann's algorithm [4] which enables learning of different kinds of filters, other algorithms specialize in narrower spectrum of filters but hoping to achieve better results. Freeman proposed an algorithm [6] which offers a way of guessing a range of high frequencies that are missing in the image. Algorithm takes training images as references with the aim of learning image sharpening. For the training data the algorithm uses a set of images of low resolution and the corresponding images of high resolution. The algorithm [7] solves the problem of improving the quality of personal photographs by using the favorite set of personal photographs as a training set. The algorithm [8] is using Hertzmann's idea of image analogies where a pair of images is used for training. This algorithm uses semi-supervised learning technique, Markov's random model in order to secure global coexistence, and image quilting technique in order to secure a local coexistence. In the article [9] a method that improves a visual quality of satellite images by using the examples of super-resolution techniques is described.

Greyscale coloring of the images is a difficult problem. Levin [10] proposes a method based on a simple assumption: neighboring pixels in the space that have a similar intensity should also be of similar colors. This algorithm does not require the precise segmentation of the image. It is necessary that the user marks the color of each region of the image that is to be to colored. As opposed to this approach Irony's [11] algorithm finds several points on the image which the algorithm had colored correctly. Then Levin's approach is applied, as though the points are those that the user marked.

In medical industry a substantial amount of data is being collected, while only a small number can be correctly analyzed. There is no reliable way of quickly segmenting a large amount of data. In the article [12] image analogies are used for resolving the problem of image segmentation. Medical images reviewed by the experts are used as training data.

In the article [13] a method is proposed for the automatic improvement of brightness and contrast by using supervised learning.

III. Implementation and Methodology of the Proposed Solution

Analogy is the basic reasoning process. People often use analogies without usually even being conscious about it. They use them for predicting or resolving different kinds of problems. Generalization based on a set of familiar examples is the central problem of machine learning. The basic premise of this paper comes from the essential idea of analogy presented by Hertzmann [4]. The idea for this work is to use neural networks for learning analogies.

A. Learning Technique and Training Data

As a program input two images are taken: the original image (model) and the filtered image (master). Neural networks learn model-master mapping, and after a given number of generations, the achieved result is displayed in the apprentice window.

For choosing a training data set two options are enabled: a choice of a number of a random set of pixels from the input images (model and master) or the manual selection of image surfaces to be used as training data. The colors on and around any given pixel *p* of the image model correspond to the colors on and around that same pixel *p* of the image master. It is necessary to select the training data set carefully since if it does not contain specific information, the desired result may not be achieved.

The learning procedure takes the achieved and the desired results and compares them, after which the network connection weights are corrected. The same set of data is processed several times during the network training until the connection weights are improved. The procedure is repeated until the desired similarity - fitness condition is satisfied when the program remembers the determined neural network.

B. Architecture

It is necessary to select the architecture, activation function and the algorithm for training the neural networks [14].

Feedforward neural networks have been chosen, which consist of the input, the hidden and the output layer of neurons, where the upper layers do not supply feedback to the lower ones. The network is structured in such a way that each neuron of the hidden layer receives the signals from all the neurons from the previous layer.

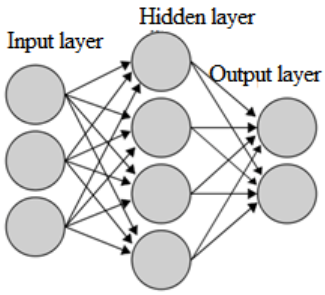

Fig. 3 Feedforward neural network

In addition to the pixels chosen as training data, the size of the retina is also given as a parameter which influences the number of neurons in the input and the hidden layers. If this parameter has value 3, the size of the retina is 3x3, the number of neurons in the input layer for a color image is 3*9, because a color image has 3 bands, with one component for red, one for green and one for blue. In that case, the number of the output neurons is 3. For a greyscale image and same retina the number of the input neurons would be 9, while the number of the output neurons would be 1. Both, the number of layers and the number of elements in each layer have impact on the efficiency of the neural network. The program offers an option

to choose the number of hidden layers, as well as the parameter for each hidden layer which shows how many times is the number of neurons in the hidden layer larger than the number of neurons in the input layer.

Each neuron consists of four basic functions: receipt of input, input processing, converting of the processed input into the output and synapse connections with other neurons. Network inputs are represented with the mathematical symbol *x(n)*, and each input information is multiplied by its weight *w(n)*. The results are added up (sum function):

$$Sum = \Sigma w_i x_i \quad (2)$$

and forwarded to the activation function which generates the result and then forwards it as the output.

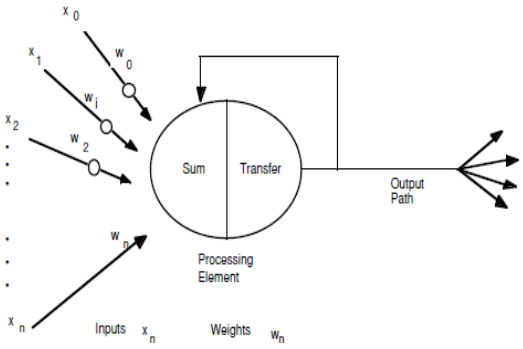

Fig. 4 Neuron in the ANN

C. Activation function

The activation function is used to scale the output data from the layers. The reason for using the activation function is the possibility for the neural network to learn a nonlinear function. If this function did not exist, the neurons from the hidden layers would not allow for more possibilities than the ordinary network that only consists of the input and the output – the perceptron.

For the activation function a sigmoid function has been chosen. The reason for using this function is the expected positive value as the output.

$$f(x) = 1/(1+e^{-x}) \quad (3)$$

The output of the sigmoid function is the value between 0 and 1.

D. Training algorithm

Resilient propagation has been chosen for the learning algorithm, as suggested by the authors of the *Encog3* neural networks software. The advantage of this algorithm is that it does not require setting of the parameters such as the learning rate or the momentum value. That is good because it is sometimes very hard to determine these parameters in search for the optimal solution. The algorithm can be easily swapped in the program, but the examples tested and presented here use the resilient propagation algorithm.

Training is the way in which neural networks learn how to improve the weights of synaptic connections in order to achieve the desired result. A training set of input data and the ideal output for each input is forwarded to the training algorithm. The resilient propagation training algorithm goes through series of iterations and after each one it attempts to lower the error rate for a certain degree. The error rate represents a percent of difference between the real output and the output produced by the training algorithm. A gradient of error is calculated for each connection in the neural network. There is no global update parameter; rather for each value of the weight matrix there is a separate delta value. These values are at first randomly initiated at very low values. They are updated in accordance with delta values after each iteration. The magnitude of the gradient is used to determine how delta values should change so that each weight matrix can be individually trained [15]. The training algorithm is:

$\forall i,j: \Delta_{i,j}(t) = \Delta_0$
$\forall i,j: \frac{\partial E}{\partial w_{ij}}(t-1) = 0$
do
Compute gradient $\frac{\partial E}{\partial w}(t)$
For all weights {
if $(\frac{\partial E}{\partial w_{ij}}(t-1) * \frac{\partial E}{\partial w}(t) > 0)$ **then**{
$\Delta_{i,j}(t) = minimum\,(\Delta_{ij}(t-1) * \eta^+, \Delta_{max})$
$\Delta w_{ij}(t) = -sign\,(\frac{\partial E}{\partial w_{ij}}(t) * \Delta_{ij}(t))$
$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t)$
$\frac{\partial E}{\partial w_{ij}}(t-1) = \frac{\partial E}{\partial w_{ij}}(t)$
}
else if $(\frac{\partial E}{\partial w_{ij}}(t-1) * \frac{\partial E}{\partial w}(t)) < 0$ **then**{
$\Delta_{i,j}(t) = maximum\,(\Delta_{ij}(t-1) * \eta^-, \Delta_{min})$
$\frac{\partial E}{\partial w_{ij}}(t-1) = 0$
}
else if $(\frac{\partial E}{\partial w_{ij}}(t-1) * \frac{\partial E}{\partial w}(t)) = 0$ **then**{
$\Delta w_{ij}(t) = -sign\,(\frac{\partial E}{\partial w_{ij}}(t) * \Delta_{ij}(t))$
$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t)$
$\frac{\partial E}{\partial w_{ij}}(t-1) = \frac{\partial E}{\partial w_{ij}}(t)$
}
}
while (converges)

The program used for experiments is written in JRuby programming language, version 1.9. JRuby is a Java implementation of the Ruby programming language, which enables usage of all the Java libraries by writing the Ruby code. It is a dynamic and object orientated programming language.

The proposed software uses *Encog3* neural networks. *Encog3* is an advanced framework for neural networks and machine learning. It contains classes for creating different

types of networks, and also supports classes for normalizing and processing data for a specific type of networks. *Encog3* enables multithreading programming and works in many integrated development frameworks, such as Eclipse or NetBeans. It is distributed as a JAR file. The software is developed in NetBeans programing framework. Resulting filter is saved as a neural network in the .eg format. The software supports different image formats such as .jpg, .gif, .png.

E. The Process of Learning the Chosen Filter

With the running of the program the original image model is loaded first, after that the chosen filter image – master is loaded. Each image appears in the separate window. After every 500 epochs a new image is displayed in the apprentice window, the attempts of the neural network to learn the master set on the basis of the model set. In case the program is interrupted or the set similarity – fitness is satisfied, the result is displayed in the apprentice window. All the parameters whose values can be changed (and were initially set at assumed values) are displayed in separate window.

$tolerrable_err is a global variable whose value is initially set at 0.1. This variable represents the permitted error and is the condition for stopping the program. The value of the error is in the interval je between 0 and 255 in the levels of grey. The run can also be stopped before the error becomes smaller of equal to the permitted value.

$taining_set_size is the global variable representing the number of the training data. It is initially set to 5000, which is the number of randomly chosen image pixels. This is one method of choosing data for the purposes of training. The other method for choosing data is not arbitrary (Fig. 5). By clicking on a pixel from the model or master image, there appears a square of the size set through the global variable *$domain_width*. This value is initially set at 38, which represents a square the size of 38x38 pixels. In case that this method of setting the training data is chosen, the size of the variable *$training_set_size* is ignored.

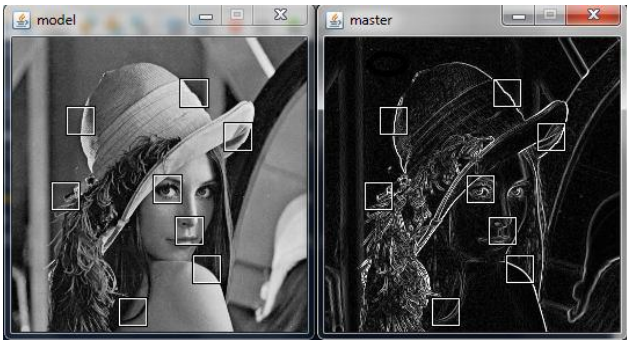

Fig. 5 Selecting characteristic points for learning process

The global variable *$hidden_layers* represents a sequence of hidden layer quotients. The quotient number determines the number of hidden layers, while the number of neurons in this layer is equal to the number of neurons in the input layer multiplied by the quotient itself. One hidden layer was initially set at value 1.5. Recalling the *ma_retina* function we can change the size of the retina, i.e. the number of neurons in the input layer.

Fig.s 6 and 7 illustrate the process of learning. Upper right corner is target image. Fig. 6 in the lower left corner shows initial results after only 20 iterations with error 1.222, while Fig. 7 is the result after 2500 iterations with error only 0.024

Fig. 6 Result of applied filter (lower left) and target (upper right) after 20 iterations

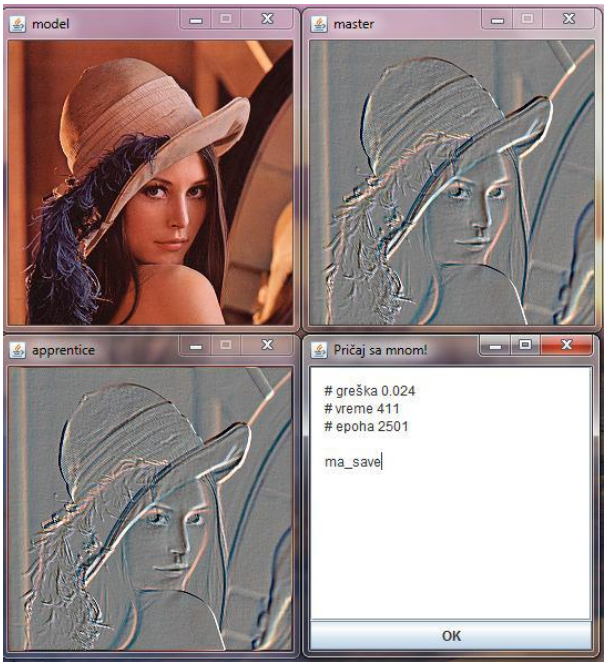

Fig. 7 Result of applied filter (lower left) and target (upper right) after 2500 iterations

The *ma_run* function starts the learning process by clicking the OK button. After the achieved similarity, the result can be saved by recalling the *ma_save* function. What is saved is the neural network of the learnt filter which can be applied to any other selected image any number of times.

IV. Experiments

In this section the results achieved by using the proposed method with our software are presented. For each selected filter the result of learning it will be shown first, followed by the results of application of that filter to newly selected images.

The simplest filters are the operations on one pixel. They are defined as a function carried out on each pixel of the image, independently from the other pixels of that same image. Some of the point processes are: converting a color image into a greyscale one, increase and reduction of contrast or brightness, the image negative and the threshold. As examples of the point process two filters are selected: the negative and the threshold.

A. Negative

The negative filter is particularly suited for improving the white and grey details surrounded by dark regions of the image, especially when black surfaces predominate. Fig. 8 shows the result of learning the "invert" filter.

The obtained result is shown in the apprentice window. The error of the obtained result is 0.01, and the number of iterations is 965. Size of the retina was 3x3.

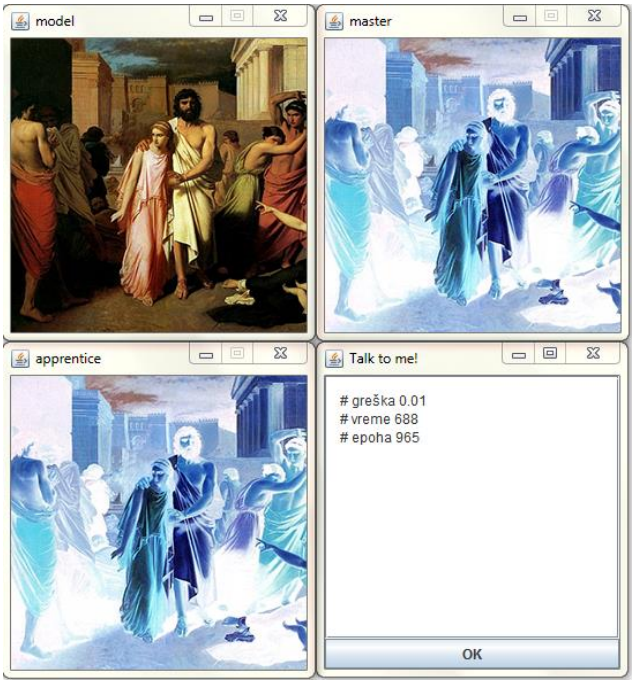

Fig. 8 Learning of the filter "negative", upper row: original and the negative, lower row: "artificial negative" by our filter

Fig. 9 illustrates the application of our "negative" filter to a different image.

Fig. 9 Application of our filter "negative" to different image

An interesting example is double application of our "negative" filter which should reverse back to the original image (Fig. 10).

Fig. 10 Application of our "negative" two times

It is remarkable how good the result is considering that our filter does not "know" anything about concept of "negative"; it is a result of neural network learning about some pixels.

B. Thresholding

The segmentation of the image relates to the procedure of dividing the image into regions with similar attributes. Of all the attributes brightness is used most often in the case of greyscale images, whereas the color is used for the color images. During the image analysis segmentation is one of the first and most important steps. One of the methods for segmentation is determining of the brightness threshold [16], [17]. Fig. 11 shows the result of learning the threshold filter. For the size of the retina 3x3 was set.

The obtained result is shown in the apprentice window. The error of the achieved result is 0.222, and the number of iterations is 11,506. The time needed for learning of the desired filter was 8441 seconds.

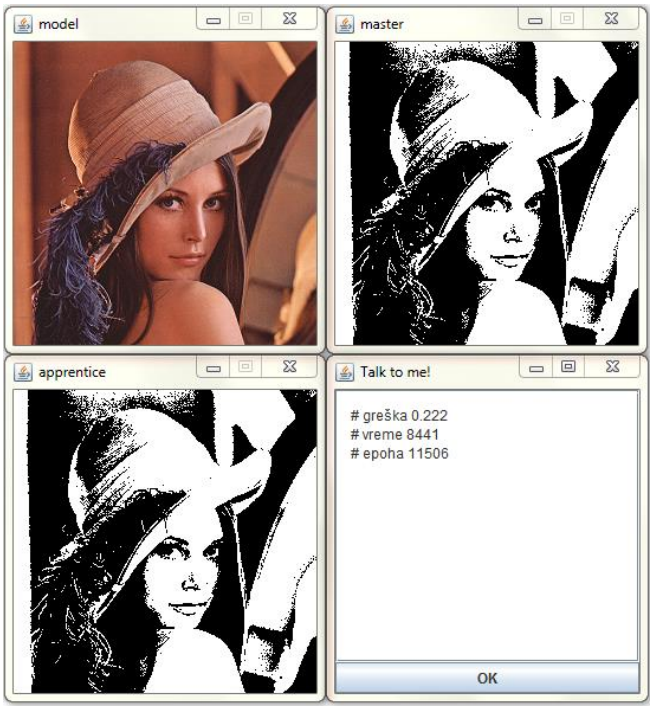

Fig. 11 Learning of the Threshold filter

Application of the obtained threshold filter is illustrated on two examples in Fig. 12.

Fig. 12 Application of the Threshold filter

C. Edge

The main goal of edge detection is deriving the important information from the image which can enable us to perform the computer interpretation and image analysis [18]. The edges correspond to significant changes of image intensity. Intuitively, the edge is a set of connected pixels located on the border between two regions. There are many filters for edge detection and some of them are: Roberts cross, Sobel, Prewitt, Kirsh, Canny and the Laplacian filter [19], [20]. Fig. 13 shows learning of the edge detection filter.

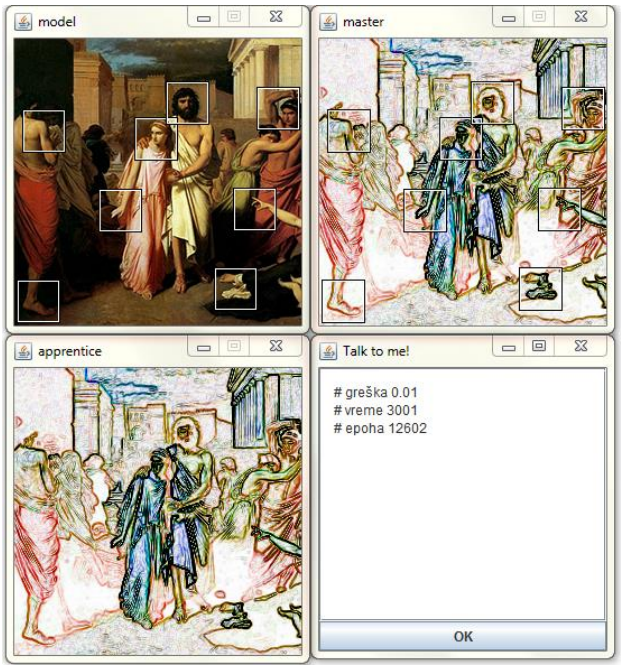

Fig. 13 Learning of the filter "edge"

The obtained result is shown in the apprentice window. The error for this result is 0.01, and the number of iterations is 12,602. The retina used is 5x5.

Fig. 14 shows the application of the learnt filter to a newly selected image.

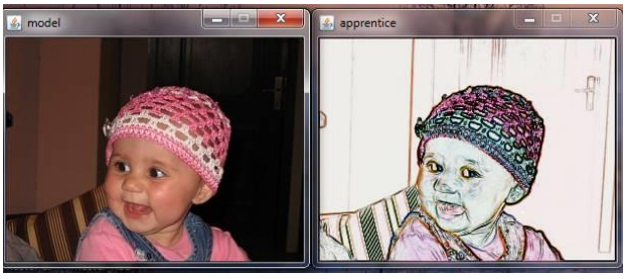

Fig. 14 Application of our filter "edge" to a different image

Another variant of the edge detection method that was mentioned is Sobel filter. Fig. 15 shows learning process for the Sobel filter, while Fig. 16 shows application of that filter to different images.

Fig. 15 Learning of the Sobel filter

Fig. 16 Application of our "Sobel" filer substitute

Yet another version of edge enhancement is Glowing edges filter where our method, again without any knowledge about applied transformation, generates very successful filter (Fig.s 17 and 18).

Fig. 17 Learning of Glowing edge

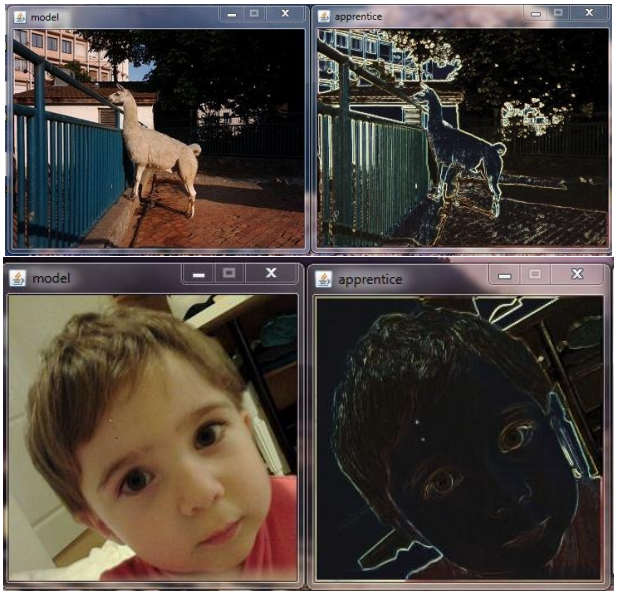

Fig. 18 Application of the Glowing edge

D. Deblur

The interpretation of the image depends on its sharpness, the possibility of isolating the information contained in the image. The sharper images show more detail. However, it is not easy to define what the sharp image is. It could be defined as an image that looks like a natural scene, but what appears natural to the human eye is difficult to calculate. Usually, a blurry image is defined as one that has lost the high frequency information. The images produced by using a photo camera are often blurry. Fig. 19 shows learning of the deblur filter. Retina 5x5 was used. The training data has not been arbitrarily chosen; rather, the image surfaces were selected for the purpose:

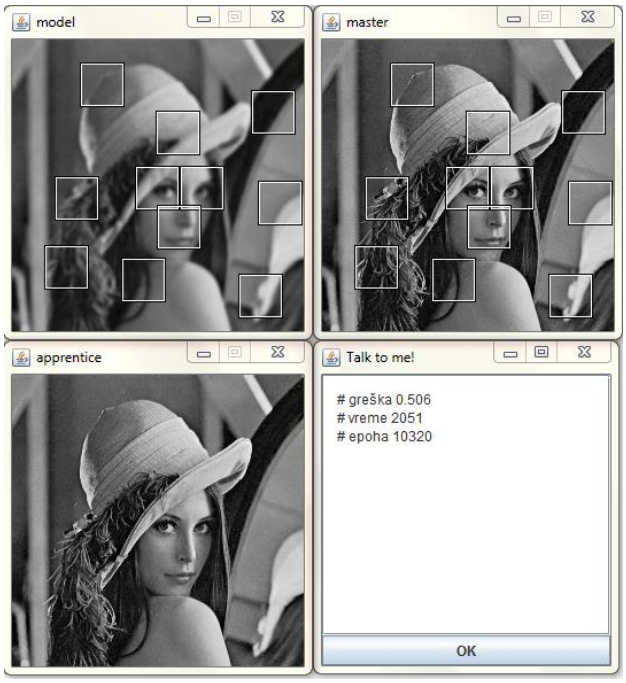

Fig. 19 Learning of the filter "deblur"

The result is shown in the apprentice window, the error is 0.506, and the number of iterations is 10,320. Fig. 20 illustrates the application of this filter to newly selected image.

Fig. 20 Application of our filter "deblur" to a different image

E. Multiple Distortions

The advantage of using our proposed method is the possibility to learn a filter which is a combined application to the image of larger number of filters. Image lena.png is taken as an example on which many filters have been applied: solarize, brightness, contrast and blur filter (Fig. 21).

Fig. 21 Learning of the combined filter: "solarize", "brightness", "contrast" and "blur"

The result is shown in the apprentice window. The error is 0.1, and the number of iterations is 9,000. Fig. 22 shows the application of the learnt filter lena_mix.eg to a newly selected image:

Fig. 22 Application of our filter which replaces combination "solarize", "brightness", "contrast" and "blur" to different image.

F. Coloring

Greyscale coloring is a difficult problem. It is very challenging to color an image if some information about the image is not already available. The same objects can be colored differently. Coloring natural objects is also problematic. The leaves are green during the spring and brown in the autumn. Fig. 23 shows learning of this filter.

Fig. 23 Learning of Coloring

Fig. 25 shows the application of the learnt filter on a newly selected image while Fig. 24 shows the original image. This appears to be a good solution; however, the problem occurs when the expected image does not contain the color nuances of the trained pair

Fig. 24 Original image

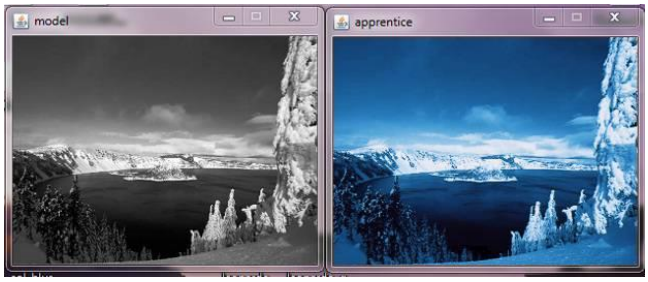

Fig. 25 Application of Coloring

V. Conclusion

We developed a method based on image analogies and neural networks learning to determine a single filter that can reverse combined effect of multiple distortions. Even with only one pair of images for training it is surprising how many different kinds and combinations of filters can be successfully

determined by our method. The proposed solution for image analogy using neural networks has proved to be a good method for learning different kinds of filters, such as: threshold, conversion of a color image into a grey-scale one, image inversion, enlargement and reduction of contrast and brightness, detecting, blurring and sharpening of edges and also all kinds of their combinations. The method enables natural image transformations instead of the selection of different filters and their adjustments. The user can simply choose the desired effect and reproduce the it on new images. The proposed software also has its limitations: the filters that include larger distortions and those that depend on the context of the image itself.

REFERENCES

[1] Chai, H. Y., Wee, L. K., Supriyanto, E.: Ultrasound images edge detection using anisotropic diffusion in canny edge detector framework, 2011, WSEAS Transactions on Biology and Biomedicine 8 (2) , pp. 51-60

[2] Anitha, M., Selvy, P.T., Palanisamy, V.: WML detection of brain images using fuzzy and possibilistic approach in feature space, 2012, WSEAS Transactions on Computers 11 (6) , pp. 180-189

[3] Nikola S. Rankovic, Milan Tuba: Improved Adaptive Median Filter for Denoising Ultrasound Images, Proceedings of the 6th European Computing Conference (ECC'12), Prague, Czech Republic, Sept 2012, pp. 169-174

[4] Hertzmann, A., Jacobs, C.E., Oliver, N., Curless, B., Salesin, D. H., Image Analogies, Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'01), New York, August 2001, pp. 327-340.

[5] Hertzmann, A., Oliver, N., Curless B., Seitz, S. M., Curve Analogies, Proceedings of the 13th Eurographics Workshop on Rendering (EGRW '02), Pisa, Italy, June 26-28, 2002, pp. 233- 246.

[6] Freeman, W.T., Jones, T.R., Pasztor, E.C., Example-based superresolution, IEEE Computer Graphics and Application, 22 (2), 2002, pp. 56-65.

[7] Joshi, N., Matusik, W., Adelson, E.H., Kriegman, D. J., Personal photo enhancement using example images, ACM Transactions Graphics, 29(2), 12, 2010, pp. 1-15.

[8] Cheng, L., Vistwanathan, S.V.N., Zhang, X., Consistent Image Analogies using Semi-supervised Learning, IEEE Computer Vision and Pattern Recognition, Anchorage, June 23-28, 2008, pp. 1-8.

[9] Li, M., Wang, F., Liu, X., Jin, H., Super-Resolution Rendering for Digital Earth Applications, Proceedings of Sixth International Symposium on Digital Earth: Models, Algorithms, and Virtual Reality, Beijing, China, September 9-12, 2009, pp. 784005-784005-6

[10] Levin, A., Lischinski, D., Weiss, Y., Colorization using optimization, ACM Transactions On Graphics, 23(3), 2004, pp. 689-694.

[11] Irony R., Cohen, C., Lischinski, D., Colorization by Example, Proceedings Of Eurographics Symposium on Rendering, 29 June – 1 July, Konstanz, Germany, 2005, pp. 201-210.

[12] Lackey, J. B., Colagrosso M.D., Supervised segmentation of visible human data with image analogies, Proceedings of the International Conference on Machine Learning, Models, Technologies and Applications, Las Vegas, June 21-24, 2004, pp. 843-847.

[13] Bychkovsky, V., Paris, S., Chan, E., Durand, F., Learning Photographic Global Tonal Adjustment with a Database of Input/Output Image Pairs, IEEE Computer Vision and Pattern Recognition, Providence, June 20-25, 2011, pp. 97-104.

[14] Bharathi, P.T. , Subashini, P.: Optimization of image processing techniques using neural networks - A review, WSEAS Transactions on Information Science and Applications 8 (8) , pp. 300-328

[15] Igel, C., Toussaint, M., Weishui, W., Rprop Using the Natural Gradient, Trends and applications in constructive approximation, 151, 2005, pp. 259-272.

[16] Brajevic I., Tuba M., Bacanin N.: Multilevel Image Thresholding Selection Based on the Cuckoo Search Algorithm, Proceedings of the 5th International Conference on Visualization, Imaging and Simulation (VIS '12), Sliema, Malta, Sept 2012, pp. 217-222

[17] Brajevic I., Tuba M.: Multilevel Image Thresholding Selection Using the Modified Seeker Optimization Algorithm, Proceedings of the 1st International Conference on Computing, Information Systems and Communications (CISCO '12), Singapore City, Singapore, May 2012, pp.258-263

[18] Yusoff, I.A., Isa, N.A.M.: Two-dimensional clustering algorithms for image segmentation, 2011, WSEAS Transactions on Computers 10 (10), pp. 332-342

[19] Baterina, A.V., Oppus, C.: Image edge detection using ant colony optimization, 2010, WSEAS Transactions on Signal Processing 6 (2) , pp. 58-67

[20] Jovanovic R., Tuba M., Dana Simian: Parallelization of the Local Threshold and Boolean Function Based Edge Detection Algorithm Using CUDA, Proceedings of the World Congress: Applied Computing Conference (ACC'12), University of Algarve, Faro, Portugal, May 2012, pp. 157-161

Milan Tuba is Provost for Mathematical, Natural and Technical Sciences at Megatrend University of Belgrade. He received B.S. in mathematics, M.S. in mathematics, M.S. in computer science, M.Ph. in computer science, Ph.D. in computer science from University of Belgrade and New York University.

From 1983 to 1994 he was in the U.S.A. at Vanderbilt University in Nashville and Courant Institute of Mathematical Sciences, New York University and later as an assistant professor of electrical engineering at Cooper Union Graduate School of Engineering, New York. During that time he was the founder and director of Microprocessor Lab and VLSI Lab, leader of scientific projects and supervisor of many theses. From 1994 he was associate professor of computer science and Director of Computer Center at University of Belgrade, Faculty of Mathematics, and from 2004 also a Professor of Computer Science and Dean of the College of Computer Science, Megatrend University Belgrade. He was teaching more than 20 graduate and undergraduate courses, from VLSI design and Computer architecture to Computer networks, Operating systems, Image processing, Calculus and Queuing theory. His research interest includes mathematical, queuing theory and heuristic optimizations applied to computer networks, image processing and combinatorial problems. He is the author of more than 100 scientific papers and a monograph. He is coeditor or member of the editorial board or scientific committee of number of scientific journals and conferences.

Prof. Tuba is member of the ACM since 1983, IEEE 1984, New York Academy of Sciences 1987, AMS 1995, SIAM, IFNA . He participated in many WSEAS Conferences with plenary lectures and articles in Proceedings and Transactions.

Jelena Z. Tasic is a graduate student at University of Belgrade, Faculty of Mathematics, Department of Computer science, where she also received B.S. in Computer Science in 2009 and M.S. in Computer Science in 2012.

She is currently working asa programer. She is the coauthor of three research papers and has participated in three international conferences. Her current research interest includes image processing and applications.

Ms. Tasic participated in WSEAS conferences.