Transactions on Information and Communications Technologies vol 16, © 1996 WIT Press, www.witpress.com, ISSN 1743-3517

Edge Detection by Fuzzy Neural Network

S. Intajag* & K. Paithoonwatanakij**
**Dept. of Industrial Instrumentation Technology,*
Faculty of Engineering,
King Mongkut's Institute of Technology Ladkrabang,
Bangkok 10520, THAILAND. Email : s8621006@kmitl.ac.th
***Dept. of Electronic, Faculty of Engineering,*
King Mongkut's Institute of Technology Ladkrabang,
Bangkok 10520, THAILAND. Email : drkitti@crsc.kmitl.ac.th

Abstract

In this paper, the edge detection using fuzzy neural network is described. The input features are fuzzy sets and a learning algorithm employs fuzzified delta rule. To increase the efficiency during the training, the varied learning rate and the momentum is applied instead of fixed values. In addition, instead of pixel-based inputs, the texture-based inputs are fed into the fuzzy neural network to facilitate and determine the quality of an edge feature. Experimental results have been tested for the case of both step edges and real world images with noise. The performance of a fuzzy neural network edge detector is compared with the neural network and the traditional techniques such as Sobel, LoG, Gabor function, and relaxation.

1. Introduction

Neural networks and fuzzy set are both well suited for nonlinear and time-varying system, especially, they can estimate a function without an explicit analytical model of how outputs depend on inputs [1]. The aims of combining fuzzy set and neural networks are to speed up the learning and generalize input features to generate appropriate output values. The fuzzy neural network, abbreviated FNN, has three models [2] which are grouped via input signals and weight of a network. The first model has real number input signals but fuzzy set weights. The second model has fuzzy input values and weights are real number. The last model both the input signals and weights are fuzzy set.

For edge detection, the applied scheme is the second model that the inputs of FNN are fuzzy values and the normalized real number weights. The learning algorithm of FNN is the fuzzified delta rule [3]. Both the varied learning rate and the momentum [4] are applied instead of the arbitrary fixed values to reduced the convergence time and avoided the local minima. The learning rate is varied following the variance of each training pattern and the momentum factor is changed depending on the standard deviation of each epoch. The presented FNN is a four-layer topology with nine nodes for an input layer, twenty nodes for the 1st hidden layer, fifteen nodes for the 2nd hidden layer, and one node for the output layer.

The purpose of this paper is an edge detection which is one of an important task in computer vision. Edge detection is the front-end process of object recognition and image understanding systems. Most edge detection operators are usually based on one of the following approaches: gradient operators, Laplacian-of-Gaussian (LoG) operator, Gabor function, and relaxation. The gradient operator [5] such as Sobel, the edge pixels are defined to be those where the first-order derivative of the pixel intensity values exceeds an arbitrary threshold. The Laplacian-of-Gaussian operator [6] is based the edge pixels by the second-order derivative of pixel intensity values undergoes a zero crossing. In the case of Gabor function [7], an edge image is achieved by the carefully defined threshold. The edge detection by relaxation [8-10] is an iteration process that the probability of each pixel is calculated starting from an initial guess value to defined edge pixels. If neighboring pixels support this conjecture the probability value is increased and will be iterated until a stable probability.

In our scheme, both synthetic and natural images have been employed to compare the performance of the proposed algorithm with the four aforementioned traditional techniques and also neural network using the backpropagation learning method. An edge detector figure of merit [11] is quantitatively used for a performance metric to evaluate the edge images. This performance merit is defined by

$$P = \frac{1}{I_N} \sum_{i=1}^{I_A} \frac{1}{1 + ad^2} \tag{1}$$

where $I_N = MAX(I_I, I_A)$, I_I and I_A represent the number of ideal and actual edge map points, a is a scaling constant; and d is the separation distance of an actual edge point normal to a line of ideal edge points. The rating factor (P) is normalized so that $P = 1$ for a perfectly detected edge.

2. Fuzzification

The edge points are determined by the edge possibility that compares with the neighboring points; when the intensity change immediately the possibility (grade or degree of membership) [12-14] of the edge will be high. Instead of using a pixel-based feature as input, the texture-based input is applied. In our proposed scheme, the three simple texture variables that consist of the entropy, dispersion, and the standard deviation is used on trial to detected edge pixels. Each variable is fuzzified [15] and fed into an FNN one at a time. The results of this trial are shown in the Figure 1.

(a) input (b) entropy (c) dispersion (d) standard deviation

Figure 1: The effect of each variable on edge detection.

Figure 1(d) has shown more a complete edge so the standard deviation has been chosen as an input feature. This texture-based input, the standard deviation value in 2×2 mask, is a fuzzy variable which is fuzzified with the S function [16]. The results from S function are the edge possibilities. An S function is given by

$$\mu_A(x) = \begin{cases} 0; & if\ x \le \min \\ 2\left(\dfrac{x-\min}{\max-\min}\right)^2; & if\ \min < x \le \text{mid} \\ 1-2\left(\dfrac{x-\min}{\max-\min}\right)^2; & if\ \text{mid} < x \le \max \\ 1; & if\ x > \max \end{cases} \tag{2}$$

Where x is a standard deviation, min and max denote the minimum and maximum values of standard deviation of pictures, and mid = (min + max) / 2. The membership values, $\mu_A(x) \in [0, 1]$ are the edge possibilities that are fed into an FNN to detect edges.

3. FNN Edge Detector

The topology of an FNN edge detector is shown in Figure 2. $X_p = \{x_1, x_2, x_3, ..., x_9\}$ represents a p^{th} input pattern vector. Each element is a local standard deviation value of 2×2 sub-image. Input vector X_p is fuzzified by Equation (2). The membership values of an edge possibility are fed into an input layer of a network. A next step is the learning algorithm of the FNN edge detector to define an edge pixel on an output layer.

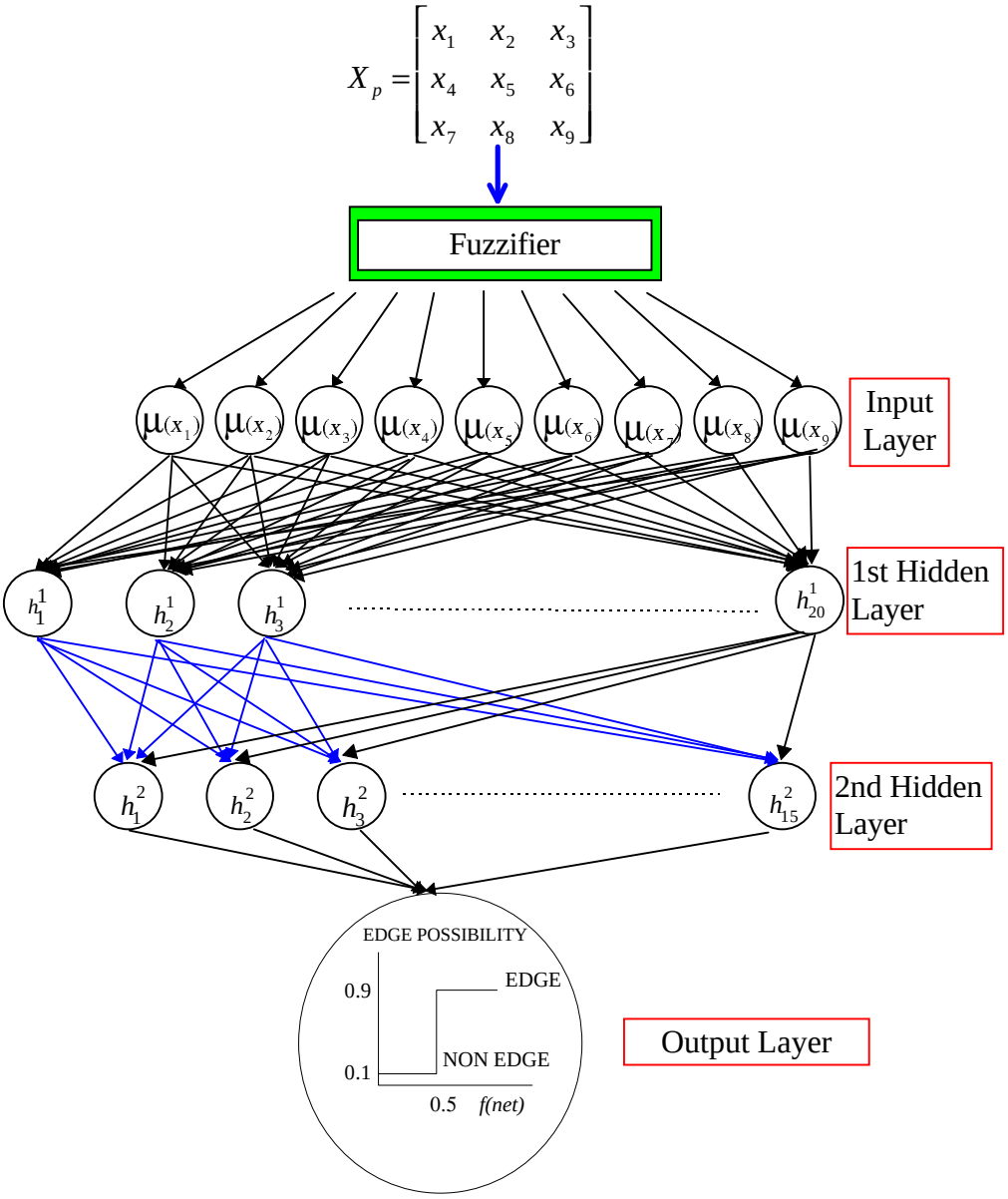


Figure 2: FNN edge detector topology.

The learning algorithm of an FNN is the fuzzified delta rule. A training set of this algorithm is $\overline{T}_p = \left\{\overline{X}_p, \overline{D}_p\right\}$; p = 1, 2, 3, ..., P. Where P is all patterns for the network training. For a bar over a symbol it represents a fuzzy set. D_p referred to the set of output target where $\overline{D}_p \in \{0.1, 0.9\}$ with 0.1 and 0.9 are denoted to non-edge and edge, respectively. The output node of each layer can be calculated with an activation function [17-19] which is defined by

$$\overline{o}_{pj} = f(\overline{y}_{pj}) = \frac{1}{1+e^{-\overline{y}_{pj}}} \tag{3}$$

where $\overline{y}_{pj} = \sum_{i=1}^{N} w_{ji}\overline{x}_{pi}$; N is a number of node on each layer. For w_{ji} denotes as a weight link from i^{th} input node to j^{th} output node.

The update weight equation is connected between layer at a time t that given by

$$w_{ji}(t+1) = w_{ji}(t) + \eta_p \delta_{pj} \overline{x}_{pi} + \alpha \Delta_p w_{ji}(t) \tag{4}$$

where η_p is a varied learning rate and is calculated by

$$\eta_p = \frac{E_p}{S}; \quad E_p = \sum_{\forall k} \left(\overline{o}_k - \overline{d}_k\right)^2 \tag{5}$$

where E_p is the sum square error between the desired output or expectation value of p^{th} pattern and the obtained activation on output layer, and S refer to the number of pattern being trained. The α in Equation (4) denotes a momentum value which is varied corresponding to the standard devation of each epoch and is expressed by

$$\alpha = \sqrt{\frac{E_p}{P}} \tag{6}$$

and the error term can be written as

$$\delta_{pj} = \begin{cases} (\overline{o}_{pj} - \overline{d}_{pj})\overline{o}_{pj}(1-\overline{o}_{pj}), & \text{for output layer} \\ \overline{o}_{pj}(1-\overline{o}_{pj})(\delta^*_{pj} w^*_{ji}), & \text{for other layers} \end{cases} \tag{7}$$

where δ^*_{pj} and w^*_{ji} are error term and weight on next layer respectively. For $\Delta_p w_{ji}(t\text{-}1)$ is the weight at the time t-1 and is given by

$$\Delta_p w_{ji}(t) = w_{ji}(t) - w_{ji}(t-1) \tag{8}$$

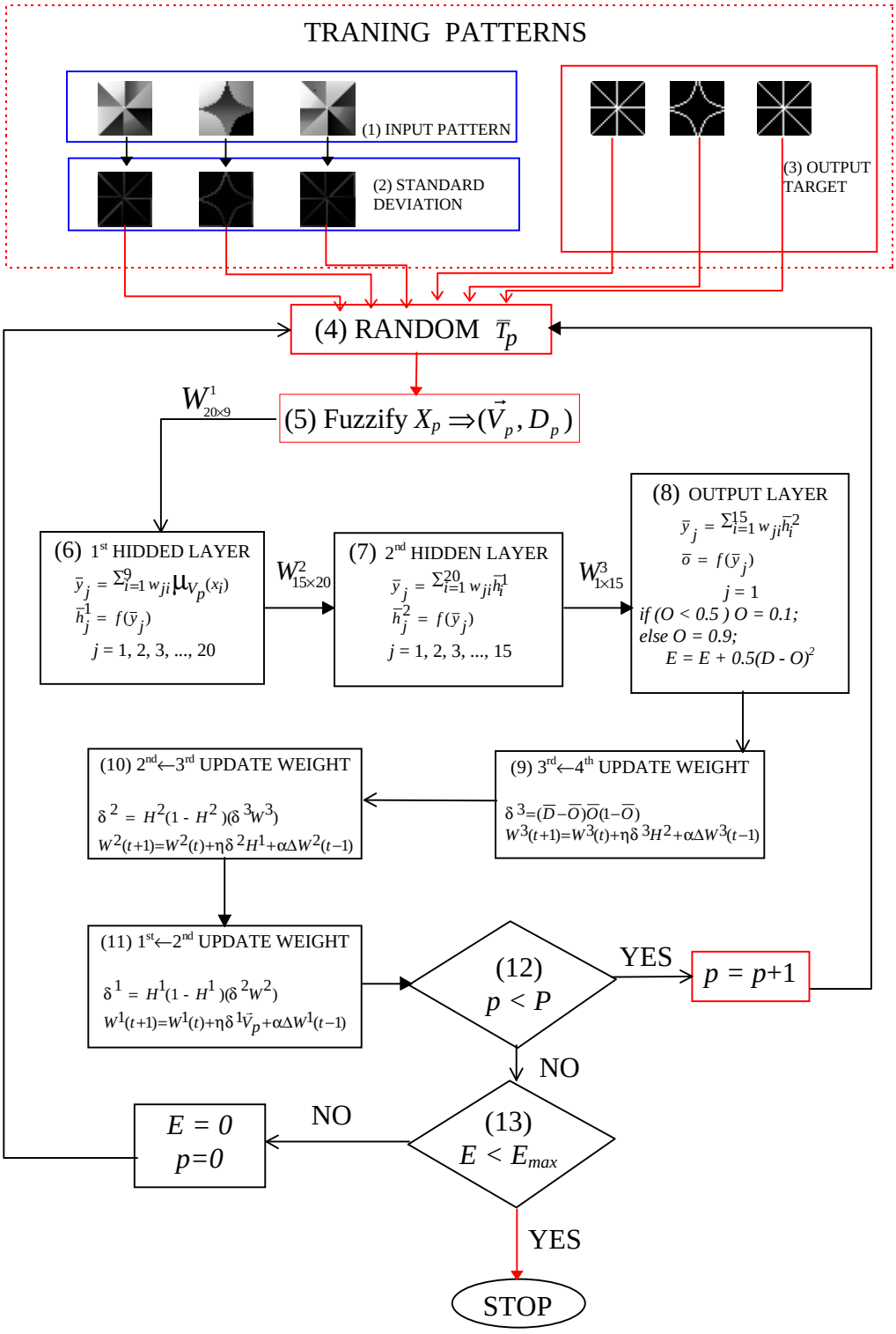


Figure 3 : FNN edge detector training diagram.

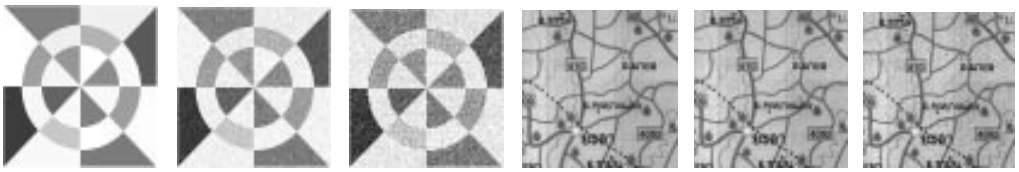
On the output layer of Figure 2, An edge pixel is defined with a threshold that equal to 0.5.

Figure 3 illustrates the training algorithm of the FNN edge detector that the fuzzified delta rule has been employed for learning. Both input patterns and output targets are 32×32 synthetic images, *W*s denote the weight matrixes, and E_{max} represents the acceptance maximum error value which is equal to 0.01.

4. Experimental Results and Comparison

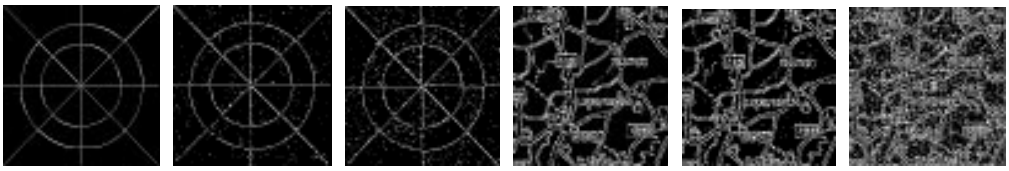
This section presents several results that illustrate the effectiveness of our proposed edge detection algorithm, by compared with the four traditional techniques namely Sobel, LoG, Gabor function, and relaxation on two images. The first group of images, shown in Figure 4(a), (b) and (c), is synthetic images that contain only step edges. The map images, shown in Figure 4(d), (e) and (f), are the second group images. These scanned images contain several types of edges including both continuous and step edges. The performance comparison of edge detectors based on two criteria: its ability to detect an edge, and the accuracy of localization the spatial position of those detected edges. The robustness of the edge detectors is also tested by applying them with Gaussian noise corrupted versions of each image that is illustrated in Figure 4(b), (c), (e) and (f), with noise of variance σ^2 = 25, 50, 15 and 30 respectively.

The comparisons are shown both qualitative and quantitative. A qualitative comparison is expressed in the Figure 5 to Figure 10 for each algorithm. Figure 5, (a), (b), (c), (d), (e) and (f), show the results of an FNN edge detector on the synthesis and map images of Figure 4. The results of an edge detection by neural network (NN) with backpropagation learning [4] are shown in Figure 6, (a) - (f). In Figure 7 to Figure 10, (a) - (f), are the results of the four aforementioned traditional techniques; Sobel, LoG, Gabor and relaxation



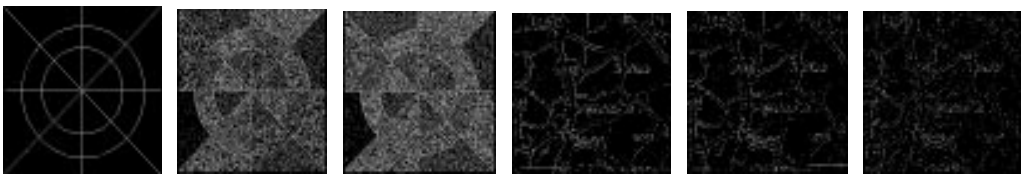
(a) (b) (c) (d) (e) (f)

Figure 4 : Synthetic Image (a-c), Map Image(d-f); (a) without noise, (b) with noise of variance σ^2 =25, (c) with noise of variance σ^2 =50 (d) without noise, (e) with noise of variance σ^2 =15, (c) with noise of variance σ^2 =30.



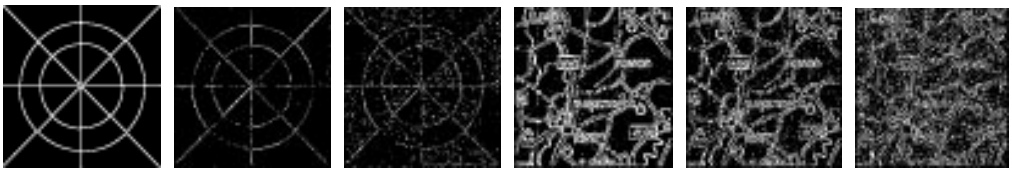
(a) (b) (c) (d) (e) (f)

Figure 5 : Edge image using FNN edge detector on synthetic and map images.



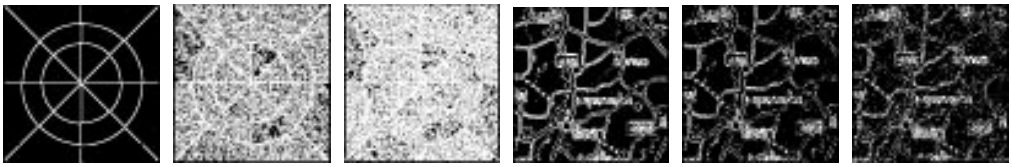
(a) (b) (c) (d) (e) (f)

Figure 6 : Edge image using neural network on synthetic and map images.



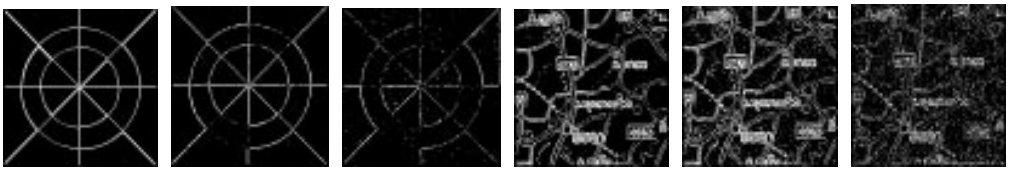
(a) (b) (c) (d) (e) (f)

Figure 7 : Edge image using Sobel on synthetic and map images.



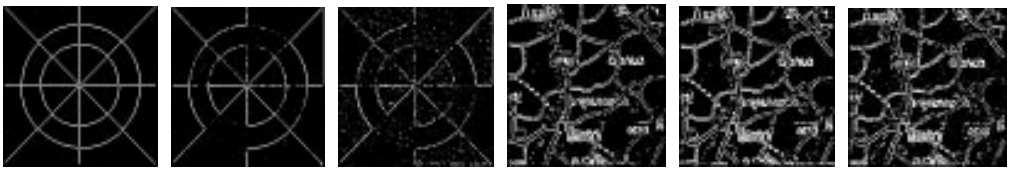
(a) (b) (c) (d) (e) (f)

Figure 8 : Edge image using LoG with 3×3 mask and $\sigma = 1.06$ on synthetic and map images.



(a) (b) (c) (d) (e) (f)

Figure 9 : Edge image using Gabor function with σ=0.9, ω=1.11 on synthetic and map images.



(a) (b) (c) (d) (e) (f)

Figure 10 : Edge image using relaxatoin with $d^* = 0.1$ and $q^* = 0.15$ on synthetic and map images.

respectively. For a quantitative comparison, the figure of merit P (Equation (1)) for each of the studied method can be calculated for the case of synthetic images and map images. In the case of map image that is a natural image, the figure of merit could not be easily computed, but in our experiment the ideal edge I_i is calculated from Robert edge operator [11] and the results are summarized in Table 1 and Table 2 for synthetic and map image respectively. In the case of added noise images, Table 1 and 2, the relaxation method achieves a good performance in comparison with the FNN but it is very time consuming to identify the optimum variables d^* (the constant employs to adjust confidence value) and q^* (the lowest bound confidence value) using trial-and-error.

Table 1: Comparison of figures of merit on synthetic image

Noise (σ^2)	Soble operator	LoG	Gabor function	relaxation	NN	FNN
No noise	0.28	0.65	0.57	0.68	0.84	1.00
25	0.27	0.10	0.27	0.65	0.24	0.29
50	0.19	0.09	0.25	0.43	0.20	0.25

Table 2: Comparison of figures of merit on map image

Noise (σ^2)	Soble operator	LoG	Gabor function	relaxation	NN	FNN
No noise	0.48	0.35	0.57	0.56	0.13	0.71
15	0.44	0.33	0.54	0.62	0.12	0.60
30	0.33	0.32	0.36	0.56	0.09	0.40

5. Conclusion

In this paper we have presented an FNN for edge detection. The learning algorithm of FNN is the fuzzified delta rule and employs two adjustable parameters instead of fixed values during the training: the first is a learning rate that varied following the variance of each training pattern, and the second is a momentum factor which is varied depending on a standard deviation of each epoch. From the compared results with both simulated images and 0-255 grey-level real images, the FNN for edge detection has demonstrated better performance than the neural network and also the traditional edge operators with consistency and more accuracy to identify edge data. The major advantages of this algorithm are the speed up in convergence time, higher accuracy for edge position and least thickness.

Acknowledgment

The authors would like to thank the referees for their valuable suggestions.

References

1. Kosko, B. *Neural Networks and Fuzzy Systems*, Prentice Hall, Englewood Cliffs, New Jersey., 1992.
2. Buckley, J. J., &Yoichi Hayashi Fuzzy neural networks, Chapter 11, *Fuzzy Set, Neural Networks, and Soft Computing*, ed. R. R. Yager & L. A. Zadeh, pp. 233-249, Van Nostrand Reinhold, New York, 1994.
3. Yoichi Hayashi, Buckley, J. J., & Czogala, E. Fuzzy neural network with fuzzy signals and weights, *Int. J. Intelligent Syst.*, 1993, 8, 527-537.
4. Intajag, S., *Edge Detection using Fuzzy and Neural Network Model*, Master Thesis, King Mongkut's Institute of Technology Ladkrabang, Bangkok, 1995, (in Thai).
5. Gonzalez, R. C. & Woods, R. E. *Digital Image Processing*, USA: Addison-Wesley, 1992.
6. Marr, D. & Hildreth, E. Theory of edge detection, *Proc. R. Soc. Lond.*, 1980, 207, 187-217.
7. Mehrota, R, Namudur, K. R. & Ranganathan, N. Gabor filter-based edge detection, *Pattern Recognition*, 1992, 25-12, 1479-1494.
8. Sonka, M., Hlavac, V., &Boyle, R., *Image Processing Analysis and Machine Vision*, UK: Chapman and Hall Computing, 1993.
9. Ballard, D. H. & Brown, C. M. *Computer Vision*, Prentice-Hall, Inc., New Jersey, 1982.
10. Rosenfeld, A. Iterative methods in image processing, *Pattern Recognition*, 1978, 10, 181-187.
11. Pratt, W. K. *Digital Image Processing*, Second Edition, USA: Jon Wiley & Sons, Inc., 1991.
12. Intajag, S. & Paithoonwattanakij, K. Automatic image enhancement using fuzzy rules, *J. Natl. Res. Council Thailand*, 1994, 26-2, 41-54.
13. Dubois, D. & Prade, H. *Fuzzy Set and Systems: Theory and Application*, Academic Press, New York, 1980.
14. Dubois, D. & Prade, H. *Possibility Theory: An Approach to computerized Processing of Uncertainty*, Plenum Press, New York, 1988.
15. Zadeh, L. A. Outline of a new approach to analysis of complex systems and decision process, *IEEE Trans. Syst. Man Cybern.*, 1973, SMC-3-1, 28-44.
16. Zadeh, L. A. Calculus of fuzzy restrictions, Chapter 1, *Fuzzy Set and Their Applications to Cognitive and Decision Processes*, ed L. A. Zadeh, K. S. Fu, K. Tanaka, & M. Shimura, pp. 1-39, Academic Press, London, 1975.
17. Kempka, A. A. Activating neural networks : Part I, *AI Expert*, June, 1994.
18. Kempka, A. A. Activating neural networks : Part II, *AI Expert*, Aug., 1994.
19. Rumelhart, D. E., McClelland, J. L. & the PDP Research Group, *Parallel Distributed Processing; Exploration in the Microstructure of Cognition, Volume 1: Foundation*, MIT Press, Massachusette, USA., 1986.