RECENT ADVANCES in MECHATRONICS

Proceedings of the 2nd International Conference on Recent Advances in Mechatronics
ICRAM'99, May 24–26, 1999, Istanbul, Turkey

Abstracts of Plenary Papers
Regular Papers

Editors:
Okyay Kaynak
Feza Kerestecioğlu
M. Önder Efe
Cem Ünsalan

The 2nd International Conference on Recent Advances in Mechatronics, Istanbul, Turkey, May 24-26, 1999

Do Fuzzy Controllers Have Advantages over Neural Controllers in Microprocessor Implementation

Bogdan M. Wilamowski and Jeremy Binfet
Department of Electrical Engineering
University of Wyoming
wilam@ieee.org and binfet@uwyo.edu

Abstract

The purpose of this document is to compare several controllers for the same desired control surface implemented in the popular HC11 micro-controller using various fuzzy and neural network architectures. Several neural network architectures were developed and optimized with a help of SNNS - Stuttgart Neural Network Simulator. The microprocessor code for all cases was obtained using the ICC11 C-compiler.

It was proven in the case of neural controller implemented on a microprocessor the code is simpler, much shorter, the processing time is comparable, and the control surfaces obtained with neural controllers are far superior. Control surfaces obtained from neural controllers also do not exhibit the roughness of fuzzy controllers that can lead to unstable or raw control.

The only drawback of neural controllers is that the design process is more complicated than that of fuzzy controllers. However, this difficulty can be easily overcome with proper design tools.

1. Introduction

In recent years, a significant amount of research has been devoted in the development of fuzzy controllers [2][3][4][7][8][14][15][16]. Fuzzy controllers are especially useful for nonlinear systems, which are difficult to describe by mathematical model. Fuzzy controllers are easy to implement. Membership functions and fuzzy rules are chosen arbitrarily and therefore fuzzy controllers are often good but not optimal. Fuzzy controllers can be significantly improved when they are tuned with neural network [5]or genetic algorithm [6].

2. Implementation of Fuzzy Controllers Using Microprocessors.

Microprocessors use primarily trapezoidal membership functions. In order to store the function only four bytes are required x1, x2, x3, and x4 (see Fig. 1). The triangular membership function is a special case of trapezoidal where x2=x3.

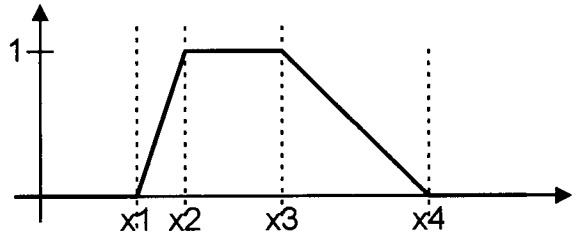


Fig. 1. Representation of the membership function in microprocessor.

Programming the fuzzy controller is relatively simple. It only requires the description of the rule table. For all combinations of input membership functions, a given output membership function must be assigned. The block diagram for Zadeh type [1] controllers is shown in Fig. 2, while required modifications for the Sugano-Tagagi controller are shown in Fig 3.

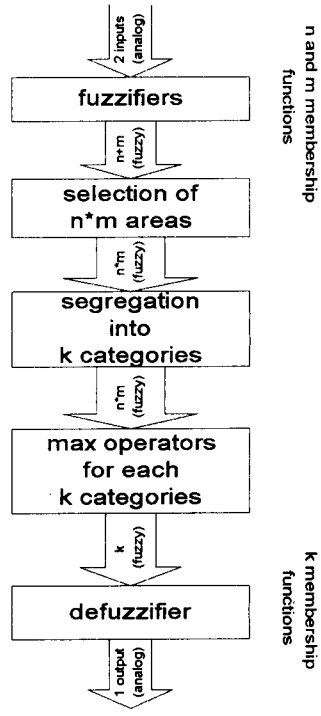


Fig. 2. Block diagram for Zadeh type controller

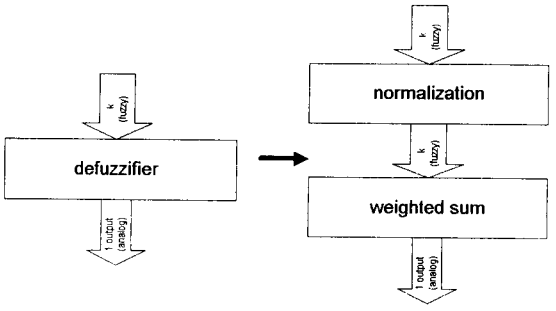


Fig. 3. In Tagagi-Sugeno controller normalization and weighted sum routines replace the defuzzification routine.

All fuzzy controllers were implemented on Motorola's 68HC711E9 microcontroller. This is a low cost, 8-bit microprocessor. The on-board features of the HC711 are 512 bytes of RAM and EEPROM and 12K bytes of UV erasable EPROM. The processor was used with an 8 MHz crystal, allowing an internal clock frequency of 2 MHz. A serial port along with an RS232 interface enables code to be down loaded from a computer. ICC11 for Windows V5 was the compiler used to program the HC711E9. It is capable of converting C or assembly code into the *.S19 file which is downloaded to the microprocessor. ICC11 also has a terminal window for interfacing with the HC711.

Table 1. Error comparison for various type of fuzzy controllers

	Approach used	error SSE	error MSE
1	Zadeh fuzzy controller with trapezoidal membership function (7*7 input and 7 output)	908.4	0.945
2	Zadeh fuzzy controller with triangular membership function (7*7 input and 7 output)	644.4	0.671
3	Zadeh fuzzy controller with Gaussian membership function (7*7 input and 7 output)	562.0	0.585
4	Tagagi-Sugeno fuzzy controller with trapezoidal membership function (7*7 input)	296.5	0.309
5	Tagagi-Sugeno fuzzy controller with triangular membership function (7*7 input)	210.8	0.219
6	Tagagi-Sugeno fuzzy controller with Gaussian membership function (7*7 input)	294.2	0.306

For all controllers shown in Fig. 5 to 10 the same rule table was used and only the shape of membership functions are different. Also, two different defuzzification processes were used. The first three examples used the Zadeh [1] approach and for the following three examples, the Tagagi-Sugeno [2] approach was implemented. All controllers were designed to emulate the control surface shown in Fig. 4. Three different membership functions were used: trapezoidal (Fig. 5 and 8), triangular (Fig. 6 and 9), and Gaussian (Fig. 7 and 10). Error comparisons are shown in Table 1. In that respect the Tagagi-Sugeno approach is far superior over the Zadeh one. The Tagagi-Sugeno algorithm has noticeably large memory requirements. The smoothest results are obtained for the Gaussian type membership functions. Unfortunately it is very difficult to implement Gaussian function on microprocessor. Computation of Gaussian function is very time consuming and it can be used only for slow controllers where time is not the critical issue.

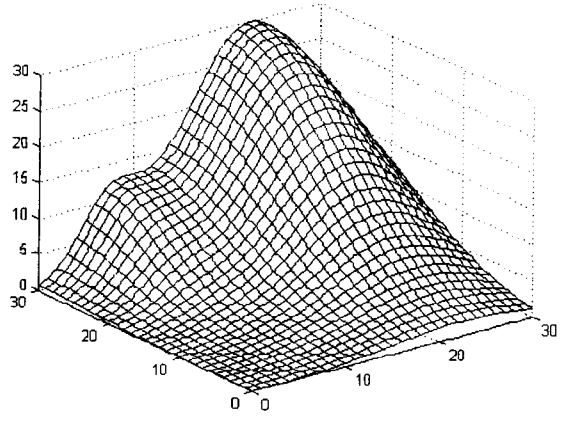


Fig. 4. Required control surface

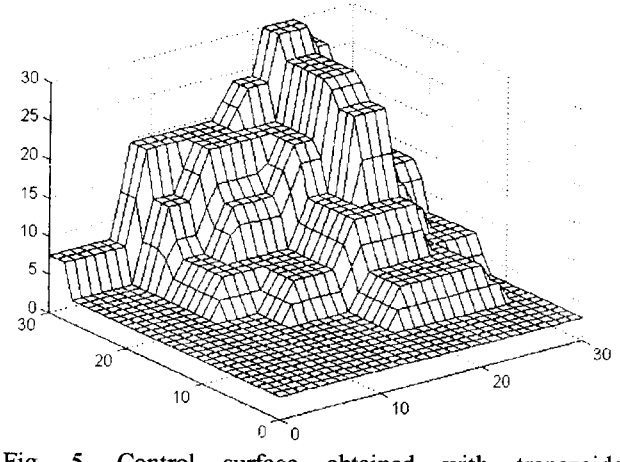


Fig. 5. Control surface obtained with trapezoidal membership functions and Zadeh approach.

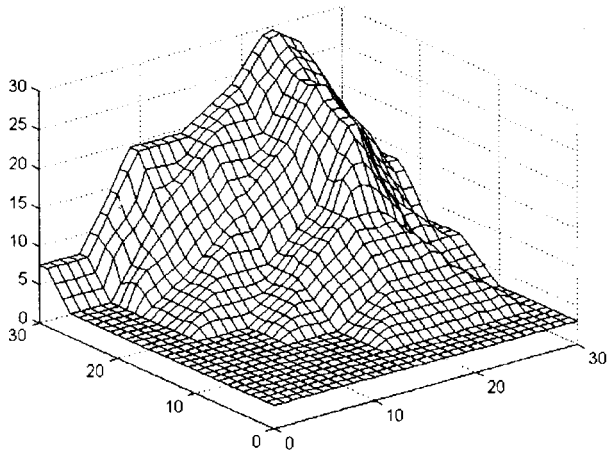


Fig. 6. Control surface obtained with triangular membership functions and Zadeh approach.

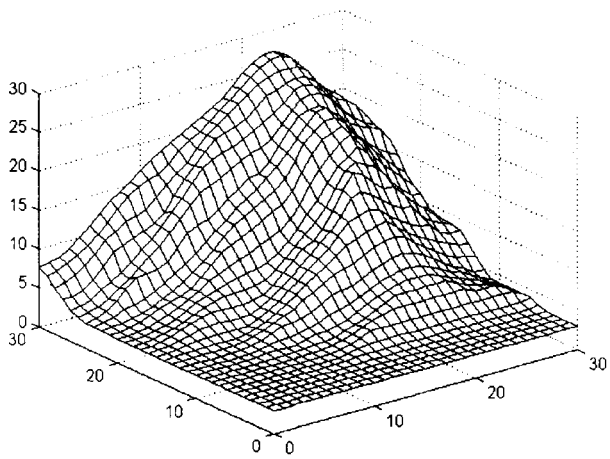


Fig. 7. Control surface obtained with Gaussian membership functions and Zadeh approach.

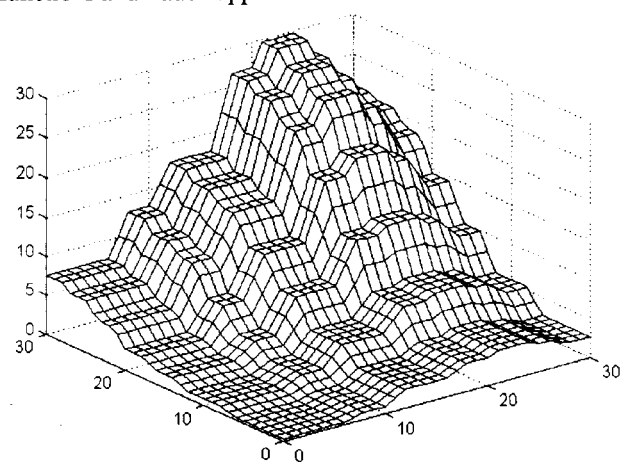
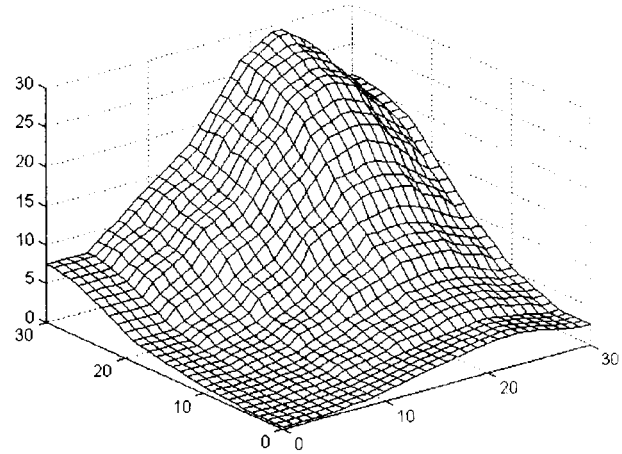


Fig. 8. Control surface obtained with trapezoidal membership functions and Tagagi-Sugeno approach



.Fig. 9. Control surface obtained with triangular membership functions and Tagagi-Sugeno approach.

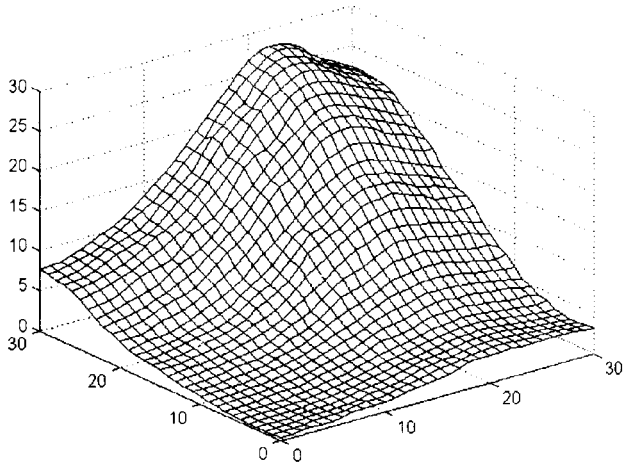


Fig. 10. Control surface obtained with Gaussian membership functions and Tagagi-Sugeno approach.

3. Implementation of Neurocontrollers Using Microprocessors.

Neural network implementations usually require computations of sigmoidal functions [7][8][12][13]

$$f(net) = \frac{1}{1+\exp(-net)} \tag{1}$$

for unipolar neurons, or

$$f(net) = \tanh(net) = \frac{2}{1-\exp(-2net)} - 1 \tag{2}$$

for bipolar neurons. This function is relatively difficult to compute and such implementation on a microprocessor is difficult. If the Elliott function is used:

$$f(net) = \frac{net}{1+|net|} \tag{3}$$

instead of the sigmoidal, then the computations are relatively simple and the results are almost as good as in the case of sigmoidal function. Fig. 11 shows comparison of sigmoidal and Elliot functions.

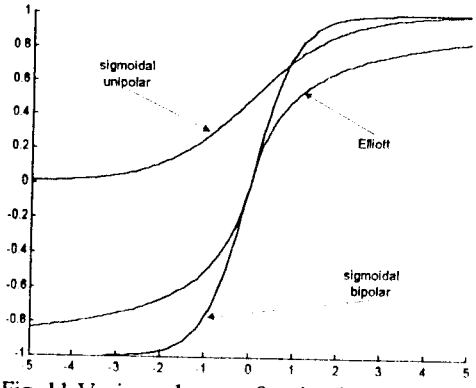


Fig. 11 Various shapes of activation functions

Neural controllers were also implemented on Motorola's 68HC711E9 microcontroller with the code written in C language. Block diagram of neurocontroller is shown in Fig. 12.

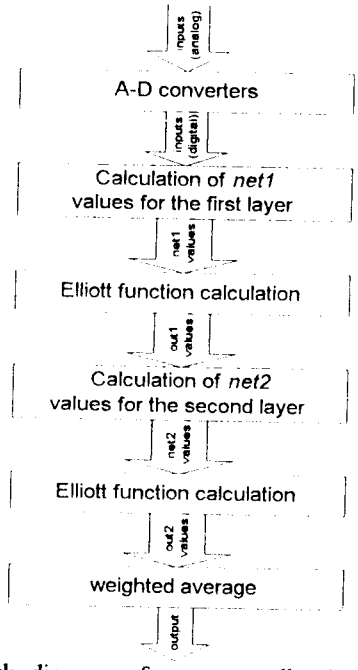
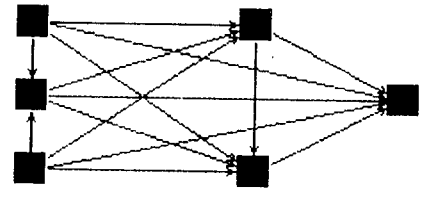


Fig. 12 Block diagram of neurocontroller implemented on Motorola 68HC711E9

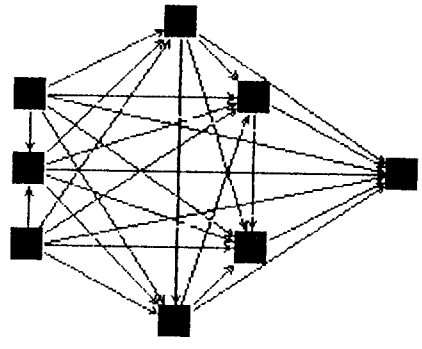
During a design process of a fuzzy controller, the designer must know what output should be expected for given input values. More precisely, what the output value is for a given combination of input membership functions. The exact same information can be used to train the neural network. This of course must be done be specially written program, or by using ready software. In our case we have used Stuttgart Neural Network Simulator SNNS [18]. This software is available free of charge form http://www.informatik.uni-stuttgart.de/ipvr/bv/projekte/snns/snns.html and it may run in both platforms UNIX and Windows. The Elliott activation function also implemented in the program SNNS (Stuttgart Neural Network Simulator). Using the dedicated software and the proper architecture, the required weights for each neurons can be found. First, the pattern file has then loaded. SNNS then trains a network for the desired control surface. Many network configurations were tested. The goal was to keep the network as simple as possible while achieving the lowest possible error. Different types of networks that were tested include a) multiple neurons in one hidden layer, b) multiple neurons in cascade and c) multiple neurons in multiple hidden layers. RProp was the training algorithm used to train the networks. It proved to have the fastest convergence time and provided the lowest errors.

For the given control surface shown in Fig. 4, several different controllers that are shown in Table 2 were implemented in the Motorola 68HC711E9 microprocessor. In order to simplify the computation for the neural architectures with limited microprocessor functions, the Elliot function (3) was used instead of the traditional sigmoidal activation function. Several neural network architectures were developed and optimized with a help of SNNS - Stuttgart Neural Network Simulator. The microprocessor code for all cases was obtained using the ICC11 C-compiler. Of course more optimal code can be written directly in assembly language, but relative ratio of codes and processing times should be similar.

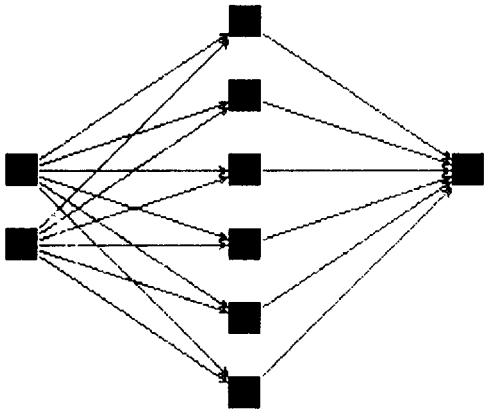
Depending on the complexity of neural networks, various levels of accuracy were obtained. Fig. 13 shows implemented neural network architectures and Fig. 14 shows obtained control surfaces for these architects



(a)

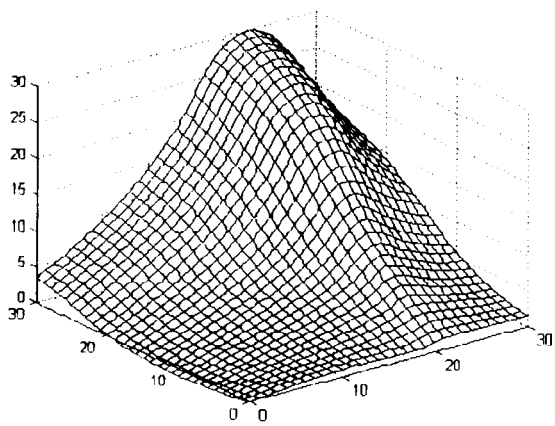


(b)

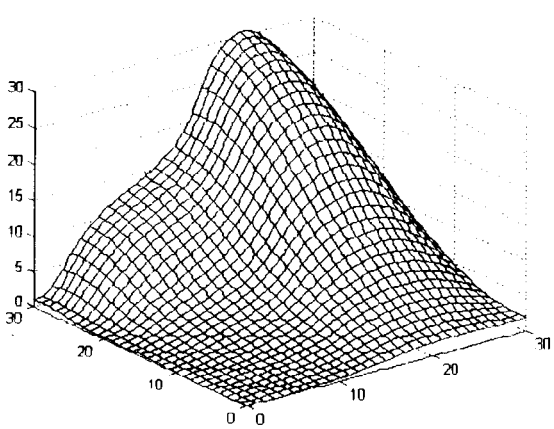


(c)

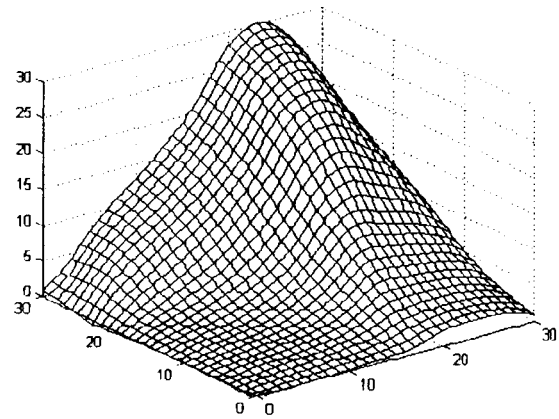
Fig. 13 Various neural network architectures developed for the required control surface of Fig. 4: (a) with 3 hiden neurons, (b) with 5 hidden neurons, and (c) with 6 hidden neurons organized in one hidden layer.



(a)



(b)



(c)

Fig. 14 Control surfaces of neuralcontroller: (a) with 3 hiden neurons, (b) with 5 hidden neurons, and (c) with 6 hidden neurons organized in one hidden layer.

Table 2. Error comparison for various type of neural controllers

	Approach used	error SSE	error MSE
1	Neural network with 3 neurons in cascade (Fig 13 (a))	0.5559	0.000578
2	Neural network with 5 neurons in cascade (Fig 13 (b))	0.0895	0.000093
3	Neural network with 6 neurons in one hidden layer (Fig 13 (c))	0.2902	0.000302

4. Comparison of fuzzy and neural approach

It is shown that it is much simpler to implement a desired control surface in a microprocessor using a neural network structure rather than using a fuzzy type of controller. The only drawback of neural controllers is that the design process is more complicated than that of fuzzy controllers. However, this difficulty can be easily overcome with proper design tools.

It was proven that in the case of neural controller implementation using a microprocessor, the code is simpler, much shorter, the processing time is comparable, and the control surfaces obtained with neural controllers are far superior. Control surfaces obtained from neural controllers also do not exhibit the roughness of fuzzy controllers that can lead to unstable or raw control.

5. Conclusion

Fuzzy controllers do have several advantages such as simple rule based design, but they usually produce relatively raw control surfaces, which are not acceptable for precision control. This obstacle can be overcome by several means. Instead of the triangular or trapezoidal

membership functions, Gaussian-like functions could be used. Better results are also possible with Tagagi-Sugeno fuzzy controllers.

One severe disadvantage of a fuzzy system is its limited ability of handling problems with multiple inputs. Fuzzy systems work well with two inputs. However, with an increase in the number of inputs, the size of the rule table grows exponentially. Because of this, three inputs is a practical limit. In the case of neural networks, the number of inputs can be much larger.

Table 3. Comparison of various fuzzy and neural controllers

	Type of controller	length of code in bytes	processing time (ms)	Error MSE
1	Zadeh fuzzy controller with trapezoidal membership function (7*7 input and 7 output)	2324	1.95	0.945
2	Zadeh fuzzy controller with triangular membership function (7*7 input and 7 output)	2324	1.95	0.671
3	Zadeh fuzzy controller with Gaussian membership function (7*7 input and 7 output)	3245	39.8	0.585
4	Tagagi-Sugeno fuzzy controller with trapezoidal membership function (7*7 input)	1502	28.5	0.309
5	Tagagi-Sugeno fuzzy controller with triangular membership function (7*7 input)	1502	28.5	0.219
6	Tagagi-Sugeno fuzzy controller with Gaussian membership function (7*7 input)	2845	52.3	0.306
7	Neural network with 3 neurons in cascade (Fig 13 (a))	680	1.72	0.000578
8	Neural network with 5 neurons in cascade (Fig 13 (b))	1070	3.3	0.000093
9	Neural network with 6 neurons in one hidden layer (Fig 13 (c))	660	3.8	0.000302

6. References

[1] L. A. Zadeh, Fuzzy sets. *Information and Control,* New York, Academic Press vol **8**, pp. 338-353, 1965.

[2] T. Takagi and M. Sugeno, Derivation of Fuzzy Control Rules from Human Operator's Control Action. *Proc. of the IFAC Symp. on Fuzzy Inf. Knowledge Representation and Decision Analysis,* pp. 55-60, July 1989.

[3] Kosko B., (1993) *Fuzzy Thinking, The New Science of Fuzzy Logic.* Hyperion, New York.

[4] Passino, K. M., S. Yurkovich, Fuzzy Control, Addison-Wesley, 1998.

[5] Tapkan, Baskin I. and Bogdan M. Wilamowski, "Trainable Functional Link Neural Network Architecture", presented at ANNIE'95 - Artificial Neural Networks in Engineering, St. Louis, Missouri, USA, November 12-15, 1995; also in Intelligent Engineering Systems Through Artificial Neural Networks vol. 5, pp. 185-190, ASME PRESS, New York 1995.

[6] Cupal, J. J. and B. M. Wilamowski, "Selection of Fuzzy Rules Using a Genetic Algorithm," proceedings of Word Congress on Neural Networks, San Diego, California, USA, vol. 1, pp. 814-819, June 4-9, 1994.

[7] Wilamowski B. M. "Neuro-Fuzzy Systems and its applications" tutorial at 24th IEEE International Industrial Electronics Conference (IECON'98) August 31 - September 4, 1998, Aachen, Germany, vol. 1, pp. t35-t49.

[8] Wilamowski, B. M., "Neural Networks and Fuzzy Systems" chapters 124.1 to 124.8 in The Electronic Handbook. CRC Press 1996, pp. 1893-1914.

[9] Fahlman, S. E. and C. Lebiere. 1990. "The Cascade Correlation Learning Architecture", *Adv. Ner. Inf. Proc. Syst.*, 2, D. S. Touretzky Ed. Los Altos, CA: Morgan, Kaufmann pp. 524-532.

[10] Ota, Y. and B. M. Wilamowski, " CMOS Implementation of a Voltage-Mode Fuzzy Min-Max Controller", Journal of Circuits, Systems and Computers, vol. 6, No 2, pp. 171-184, April 1996.

[11] Choi J., B.J.Sheu, and J.C.F. Chang, (1994) A Gaussian Synapse Circuit for Analog VLSI Neural Networks. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. **2**, no. 1, pp. 129-133.

[12] Hornik, K., "Multilayer Feedforward Networks are Universal Approximators," Neural Networks, v.2, pp 359-366, 1989.

[13] Funahashi, K., "On the Approximate Realization of Continuous Mappings by Neural Networks," Neural Networks, v.2, pp 183-192, 1989.

[14] Wang, L., Adaptive Fuzzy Systems and Control, Design and Stability Analysis, PTR Prentice Hall, 1994.

[15] Efe, M. O., O. Kaynak, "A Novel Computationally Intelligent Architecture for Identification and Control of Nonlinear Systems," The 1999 IEEE Int. Conf. on Robotics and Automation, ICRA'99, May 10 – 15, Detroit, Michigan, U.S.A.

[16] Efe, M. O., A. B. Koku, O. Kaynak, "Comparison of Soft Computing and Conventional Methodologies in Control of Servo Systems", Int. Conference on Industrial Electronics, Control and Instrumentation, IECON'98, August 31 – September 4, Aachen, Germany, v.1, pp 75-80, 1998.

[17] Wilamowski B. M. and R. C. Jaeger, "Neuro-Fuzzy Architecture for CMOS Implementation" accepted for *IEEE Transaction on Industrial Electronics.*

[18] Stuttgart Neural Network Simulator SNNS http://www.informatik.uni-stuttgart.de/ipvr/bv/projekte/snns/announce.html