Fuzzy Controller With Second Order Defuzzification Algorithm

Nirav Dharia[1], Jagadeesh Gownipalli[1], Okyay Kaynak[2] and Bogdan M.Wilamowski[1]
[1]University of Idaho, College of Engineering, Boise, U.S.A
[2] Bogazici University, Electrical and Electronic Engineering Department, Istanbul, Turkey

ABSTRACT

Fuzzy controllers can handle multiple inputs and outputs however size of the fuzzy table grows exponentially with each input added. In proposed approach two six bits input, (8 x 8), (6 x 6) and (4 x 4) fuzzy look up tables (LUT) were used each having 64, 36 and 16 values respectively. The system uses two inputs, one for the row and the other for the column, the upper three bits of each input are used for addressing the exact position of the nearest data point in the look up table, and the rest of the input is the information about the membership function. Eight adjacent points are used to approximate the required data points to generate the control surface. Second order method proposed in this paper was used to generate the control surface. The algorithm used for defuzzification will be discussed in detail. The advantage of this method over other defuzzification methods like the LUT and the triangular membership function approach is the efficient usage of a relatively smaller look up table, which minimizes the hardware requirement and ensures easier implementation. The error produced using this method is reduced considerably as compared to others; moreover it can also be extend into multi-dimensions easily with minimum number of inputs resulting in smoother control surfaces.

1 INTRODUCTION

Fuzzy Logic that Lotfi Zadeh introduced in 1965 [1] has been applied in a wide variety of disciplines such as system identification, classification, control and decision support. Stability analysis and systematic design are the most important issues for fuzzy control systems [5][6][14][15]. Fuzzy logic is an approach to reasoning where the rules of inference are approximated rather than finding exact solution. It is useful for manipulating information that is incomplete or imprecise. Fuzzy systems are universal approximation systems such as the Neural Network and can be used where the system can be classified using the Fuzzy If-Then rule [1]. This classification method differentiates Fuzzy Systems from other approximation methods. Fuzzy systems can be implemented by converting human knowledge to Fuzzy Rules [2][3][4]. Fuzzy systems are particularly useful for nonlinear system control [13]. Membership functions and fuzzy rules are selected differently for different systems and thus fuzzy systems are optimal for an application but not perfect. Fuzzy systems are used to provide solutions to control problems that cannot be described by complex mathematical models. In recent years, fuzzy systems have become very popular with the digital designer as a fuzzy system enables them to use non-linear controller for their application [2][3][4].

Using the established methods [1][8][11] for defuzzification, the control surfaces obtained are rough, which is not acceptable in the applications that require precise output and in order to overcome this difficultly we propose the use of the second order defuzzification algorithm, which reduces the error in the output surface and the control surface obtained is very smooth. In this concept three bits are used for defuzzification and the results obtained are much better then the ones obtained by the current methods using three bits defuzzification. Moreover using this approach one can easily extend this into multidimensional application [9][10]. Using this approach five-dimensional implementation on a FPGA [7] can be achieved (3x5=15 bit LUT address), as the size of the LUT is much smaller as compared to the ones used in other approaches. Larger dimensions are also possible if the inputs are well approximated by 2 bits. Let us consider the classical LUT [11] and triangular membership function approach [13] and then compare these results with the results obtained by using the second order defuzzification algorithm. We considered a three-dimension surface: s = cos (x) * sin (y) to compare the results obtained by the three methods. The required control surface is shown in the Fig. 1

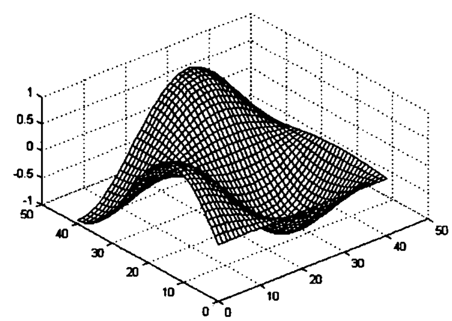

Fig. 1 Required control surface of function s=cos (x)*sin (y)

LUT approach

Using the classical LUT approach, using a 4x4 and 8x8 LUT the results obtained are shown in the Fig. 2 and Fig. 3,

0-7803-7278-6/02/$10.00 ©2002 IEEE

which shows that the surface obtained are very rough and are unacceptable in most applications.

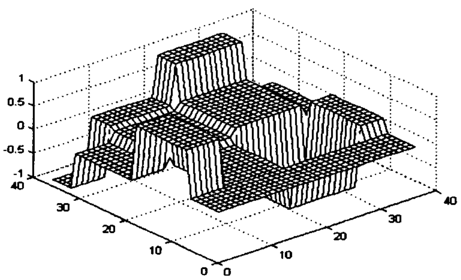

Fig. 2 Look up table results (4x4) LUT (2 bits per input)

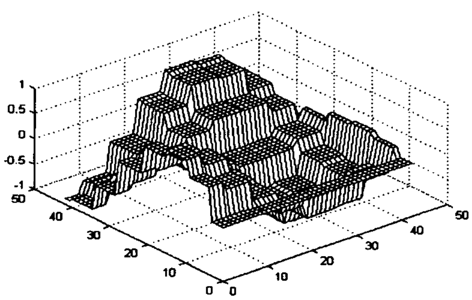

Fig. 3 Look up table results (8x8) LUT (3 bits per input)

In order to improve the results and reduce the error, Triangular membership function approach was used, which is discussed in the next section.

Triangular Membership Function Approach

We are using three bits inputs and thus there can be at most 8-membership functions. In this method as shown in the Fig. 4 we have used Tagagi Sugeno method [12] having 8 membership function for defuzzification of the input data and the information about the membership function is obtained from the lower input bits.

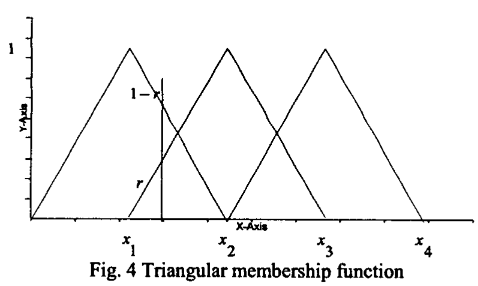

Fig. 4 Triangular membership function

The results obtained using 4x4 and 8x8 LUT [9] are shown in the Fig. 5 and Fig. 6

The results obtained using the triangular membership function were satisfactory but at the same time the error generated using this method is unacceptable for most applications. The error plots are shown in Fig. 7 to Fig. 8.

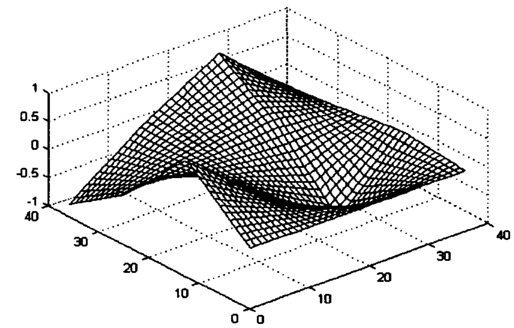

Fig. 5 Triangular approximation (4x4) LUT (2 bits per input)

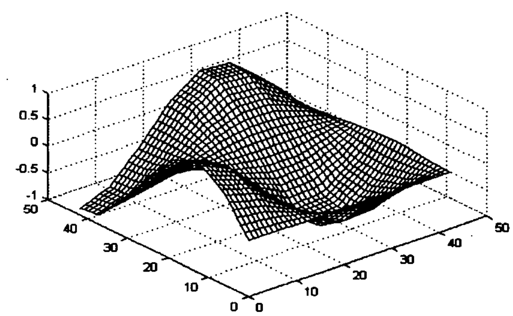

Fig. 6 Triangular approximation (8x8) LUT (3 bits per input)

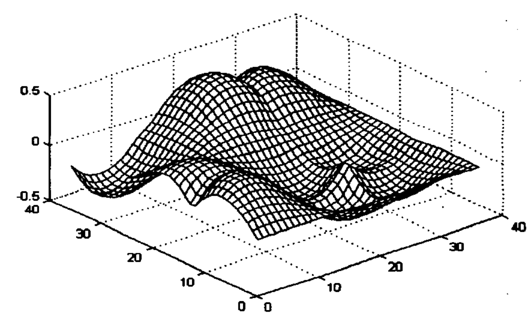

Fig. 7 Error surface for the Triangular membership function approach (4x4) LUT

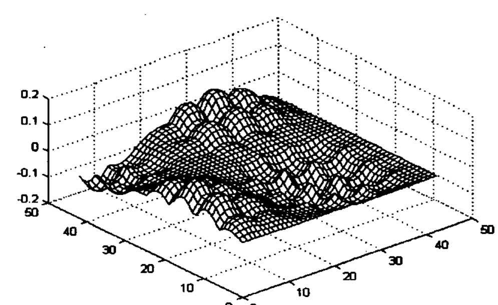

Fig. 8 Error surface for the Triangular membership function approach (8x8) LUT

From the above plots we observe that the error reduces as the size of the LUT [9] increases but at the same time the hardware implementation and computational complexities increases. Using the second order defuzzification algorithm proposed in this paper we get better results for a 3-bit input with a substantial reduction in errors. This method will be discussed in details in the next section.

2 SECOND ORDER DEFUZZIFICATION ALGORITHM (SODA)

To explain this concept we shall first derive the equations and then consider one dimension example before moving on to the 2-dimension approach.

Algorithm in 1-D

Second order equation is of the form

$$y(x) = x(ax + b) + c \tag{1}$$

First derivative of the above equation

$$\frac{\partial y}{\partial x} = 2ax + b \tag{2}$$

Second derivative of the above equation

$$\frac{\partial^2 y}{\partial x^2} = 2a \tag{3}$$

Lets say that the required point 'x' is between x_2 and x_3 as shown in the Fig. 9 this method then takes x_2 as an index and calculates the value of x at that point. Here we consider 4 points x_1, x_2, x_3 and x_4, which are adjacent to the required point 'x'. The corresponding Y values (y_1, y_2, y_3, y_4) are obtained from the LUT. Now we find the intermediate points between (y_1, y_2), (y_2, y_3) and (y_3, y_4) using the following equations.

$$y_{12} = 0.5(y_1 + y_2) \tag{4}$$

$$y_{23} = 0.5(y_2 + y_3) \tag{5}$$

$$y_{34} = 0.5(y_3 + y_4) \tag{6}$$

Using numerical first order differentiations we get

$$\frac{\partial y}{\partial x} = \frac{y_3 - y_2}{\Delta x} \tag{7}$$

Using numerical second order differentiations we get

$$\frac{\partial^2 y}{\partial x^2} = \frac{2y_{23} - y_{12} - y_{34}}{\Delta x^2} \tag{8}$$

Substituting equations (4), (5) and (6) into (8) we get

$$\frac{\partial^2 y}{\partial x^2} = 0.5\frac{y_1 - y_2 - y_3 + y_4}{\Delta x^2} \tag{9}$$

Equating equations (2) and (3) with (7) and (9) respectively we get

$$y(x) = c + x[b + a(1 - x)] \tag{10}$$

Where

$$c = y_2 \tag{11}$$

$$b = \frac{y_3 - y_2}{\Delta x} \tag{12}$$

$$a = 0.25\frac{y_1 - y_2 - y_3 + y_4}{\Delta x^2} \tag{13}$$

In the case of the digital implementation, the most significant bits are used as identification of the membership function (address in LUT) and remaining bits as a fuzzy variable. Now Δx=1 and equations (11) to (13) simplifies to

$$c = y_2 \tag{14}$$

$$b = y_3 - y_2 \tag{15}$$

$$a = 0.25(y_1 - y_2 - y_3 + y_4) \tag{16}$$

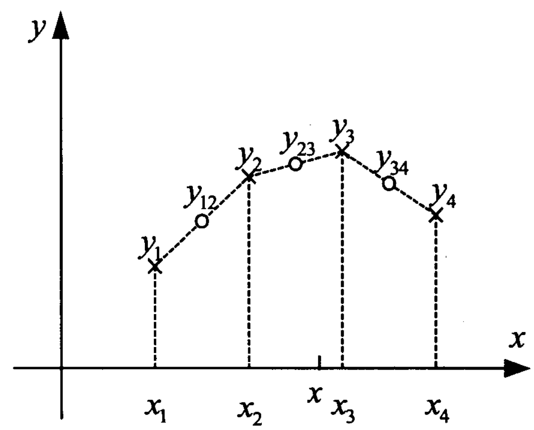

Fig 9 Second Order Defuzzification Algorithm in 1-D

One dimension example

We considered a simple $y = \sin(x)$ function and generated the plots obtained by the direct LUT [9] method Fig. 13 and linear approximation method Fig. 14. The dotted line shows the plot of the original function.

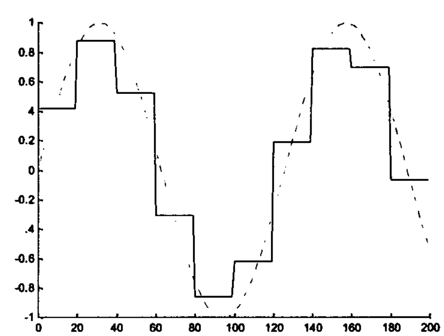

Fig. 10. Data stored in LUT

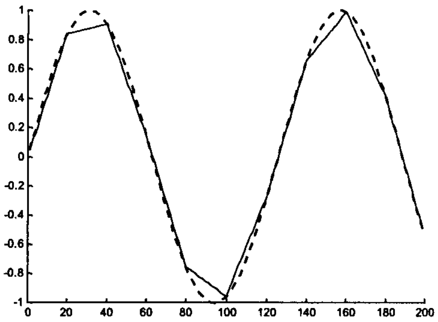

Fig. 11 Linear approximation

The above Figs. 10 and 11 shows that the surface obtained by the current two methods are not smooth. Now applying the concept of the SODA explained in the next section we have plot the curve shown in Fig. 12. As we can see the obtained surface is very smooth and almost similar to the required surface. The error plot shown in Fig. 13 proves the error obtained by this method is very small.

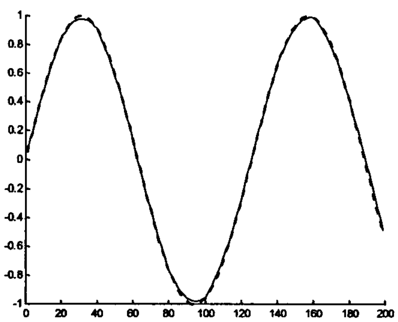

Fig. 12 Second Order Defuzzification for 1-D

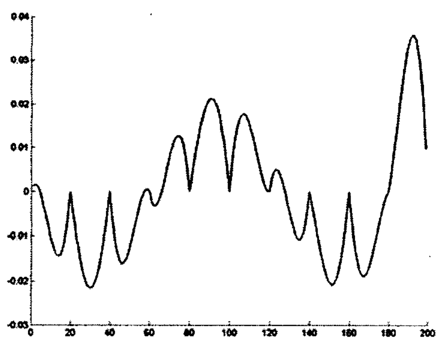

Fig. 13 Error using Second Order method in 1-D

Extension to 2-Dimension

Special approach has to be taken to extend the concept into multi-dimension. The fuzzification interface takes two 6 bits input from an analog to digital converter. The higher 3-bits are the address bits and the lower bits are used to determine the membership index of the input fuzzy groups. The membership function used here is the symmetric triangular membership function as shown in the following Fig. 4. The points marked in the Fig. 14, represent the data points in the LUT [9] and the point marked 'X' is the input.

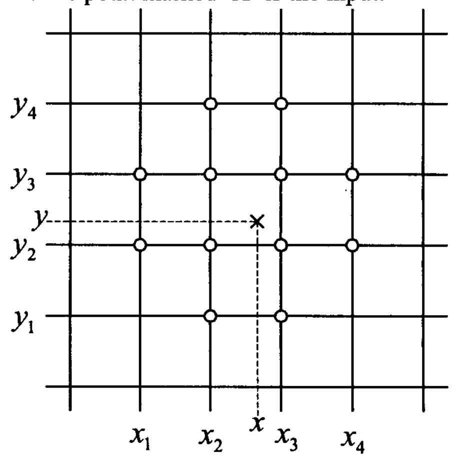

Fig. 14 Twelve points used for calculation surrounding the input point

Now in this algorithm the approximation is done both in the X and Y direction, using the 8 adjacent points shown in the Fig. 14. Following the algorithm discussed in the previous section and using the following equations (17) and (18) four points are approximated from the 8 adjacent points and are shown on the dotted lines in Fig. 15.

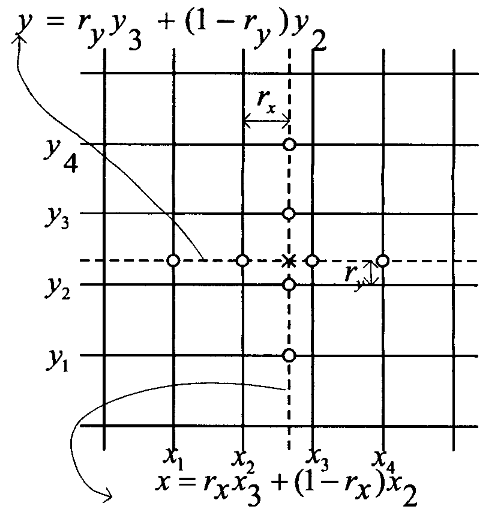

Fig. 15 Reduction of twelve points from the Fig. 17 to 8 points for calculating first and second order derivative.

$$x = r_x x_3 + (1 - r_x) x_2 \tag{17}$$

$$y = r_y y_3 + (1 - r_y) y_2 \tag{18}$$

Now these 4 points are approximately in line with the required input point, in both the directions. Thus the second order defuzzification algorithm applied to single dimension is applied in both X and Y direction separately following the same procedure explained in the previous section and 2 points are obtained. Taking the average of these two points we obtain the final result and the entire procedure is followed for all the input data elements. Several tests were carried out for testing the SODA and the results are listed in the following section.

Extension to multi-Dimension

However the above concept can be extended to multi-dimensions applying SODA in 'n' dimensions individually and approximating 'n' points from which the final point is calculated. The application of SODA in multi-dimension is much more complex as compared to its application in 1 and 2-dimension case.

Experimental Results

The sample surface is s = cos (x) * sin (y) as discussed in the Introduction and the original surface is shown in Fig. 1. We first plot the data points that are stored in the 4x4 and 8x8 LUT shown in figures 16 and 17 respectively.

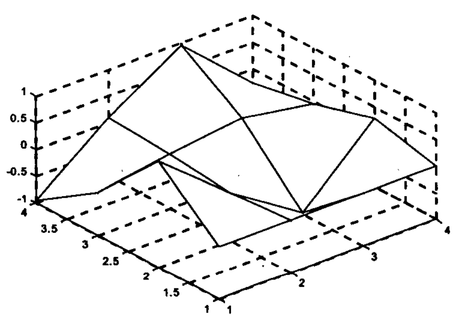

Fig. 16 Stored data in the 4x4 look up table (2 bits per input)

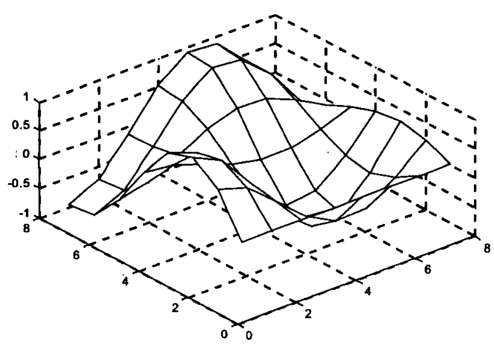

Fig. 17 Plot of the data stored in the 8x8 look up table (3 bits per input)

Results obtained by using the proposed algorithm are shown in the Figs. 18 and 19 using the 4x4 and 8x8 LUT [9] respectively. The results were obtained by carrying out MATLAB simulations.

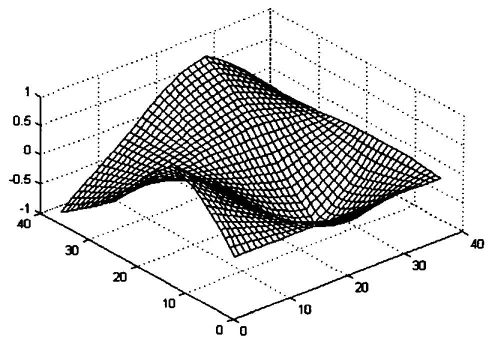

Fig. 18 Second Order Defuzzification (4x4) (2 bits per input)

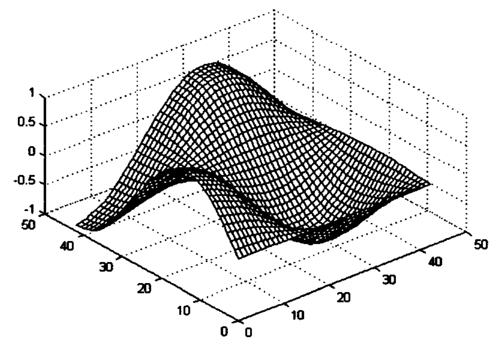

Fig. 19 Second Order Defuzzification (8x8) (3 bits per input)

As we can see from the above figures the surface obtained is very smooth.

3 CONCLUSION

The control surface obtained by using fuzzy controllers is usually raw and thus they are not applicable in applications where precision control is required but by using the proposed SODA method we observe that control surface Figs. 18 and 19 obtained is very smooth and relatively closer to the original stored surface Fig. 1 as compared to the control surfaces obtained by using LUT approach Figs. 2 and 3, Triangular Membership function approach Figs. 5 and 6. This can also be proved by comparing the error plots obtained for the Triangular Membership function approach Figs. 7 to 8 and SODA Figs. 20 to 21.

Using the presented approach there is a considerable reduction in the size of the Look Up Table as compared to the Classical Look Up table method and thus it saves a lot of space, reduces computational complexities and makes hardware implementation easier.

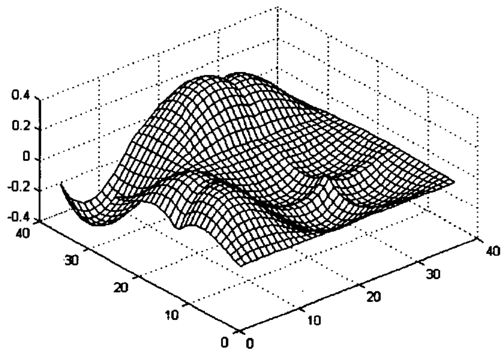

Fig. 20 Error surface for the proposed SODA method (4x4) LUT (2 bits per input)

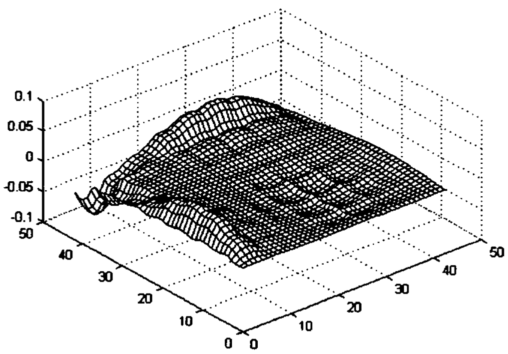

Fig. 21 Error surface for the proposed SODA method (8x8) LUT (3 bits per input)

To prove the validity of the method further we went through the entire procedure to obtained the control surface and the error plot using a 6x6 LUT as shown in Figs. 22 and 23 which shows good results as compared to the surfaces obtained by using the 4x4 and 8x8 LUT shown in Figs. 18 to 21

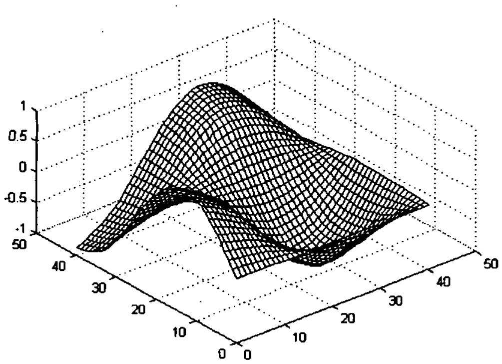

Fig. 22 Second Order Defuzzification (6x6) (3 bits per input)

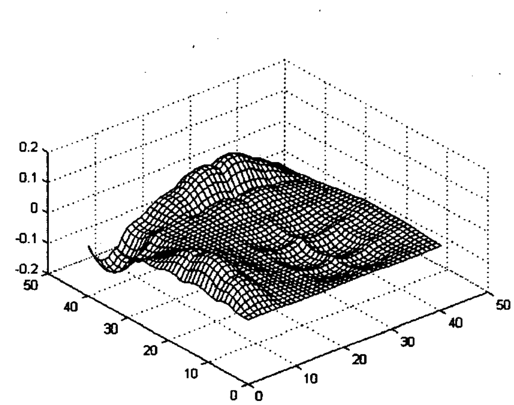

Fig. 23 Error surface for the proposed SODA method (6x6) LUT (3 bits per input)

4 REFERENCES

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

[2] B. M. Wilamowski and Jeremy Binfet " Microprocessor Implementation of Fuzzy Systems and Neural Networks ", International Joint Conference on Neural Networks (IJCNN'01), pp. 234-239, Washington DC, July 15-9, 2001.

[3] Mike McKenna and B. M Wilamowski, " Implementing a Fuzzy System on a Field Programmable Gate Array ", International Joint Conference on Neural Networks (IJCNN'01), pp. 189-194, Washington DC, July 15-19, 2001.

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

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

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

[7] Patyra, M.J.; Grantner, J.L.; Koster, K. "Digital fuzzy logic controller: design and implementation" IEEE Transactions on Fuzzy Systems, Volume: 4 Issue: 4 Page(s): 439 -459, Nov. 1996

[8] Hao Ying "General SISO Takagi–Sugeno Fuzzy Systems with Linear Rule Consequent Are Universal Approximators", IEEE Transaction on Fuzzy System, vol. 6, no. 4, Page(s): 582 -587 NOVEMBER

[9] D. Tikk and P.Baranyi,"Comprehensive Analysis of a New Fuzzy Rule Interpolation Method IEEE Transaction on Fuzzy System, vol. 8, no. 3, Page(s): 281 -296,JUNE 2000 281

[10] T. Kavli, "ASMOD: An algorithm for adaptive spline modeling of observation data," Int. J. Contr., vol. 58, no. 4, pp. 947–968, 1993.

[11] Tan H., R. Sandige, and B. Wilamowski, "Hardware implementation of PLD-based fuzzy logic controllers using a look-up table technique" presented at ANNIE'94 - Artificial Neural Networks in Engineering, St. Louis, Missouri, USA, November 13-16, 1994; also in Intelligent Engineering Systems Through Artificial Neural Networks vol. 4, pp. 89-94, ed. C. H. Dagli, B. R. Fernandez, J. Gosh, R.T. S. Kumara, ASME PRESS, New York 1994

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

[13] 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.

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

[15] Output feedback robust H/sup /spl infin// control of uncertain fuzzy Dynamic systems with time varying delay Kap Rai Lee; Jong Hae Kim; Eun Tae Jeung; Hong Bae Park Page(s): 657 –664.