A NEURO-FUZZY EMBEDDED SYSTEM FOR INTELLIGENT ENVIRONMENTS

Javier Echanobe, Ines del Campo and Guillermo Bosque
Department of Electronics, University of the Basque Country, Spain

Keywords: Intelligent environments, Embedded systems, Neuro-Fuzzy, FPGA.

Abstract: Intelligent Environments are endowed with a large number of non-intrusive, embeded electronic systems, such as sensors, microprocessors, actuators, etc. These electronic systems must exhibit intelligent abilities in order to learn and adapt from the users' habits and preferences. In this paper, we propose a Neuro-Fuzzy electronic embedded system to control several ambient parameters of an intelligent environment, such as temperature, illumination, volume of the sound, and aroma. In particular the PWM-ANFIS model has been selected which provides learning/adaptation features and also fuzzy reasoning. The system is implemented on a reconfigurable device (i.e., FPGA) leading to a small, compact and very efficient electronic system.

1 INTRODUCTION

An Intelligent Environment, also known as Ambient Intelligence Environment, is characterized by its ubiquity, transparency and intelligence (ISTAG, 2001)(ESTO, 2003)(ISTAG, 2005). It must be ubiquitous in the sense that the environment is endowed with a multitude of embedded electronic systems such as sensors, microprocessors, communication devices, actuators, etc. It must be also transparent so the users do not note the presence of the electronic infraestructure in question. Finally, the system requires intelligent abililties in order to process - in a smart way - the large amount of information coming from the sensors, to be aware of the context, and also to learn and adapt from the habits and necessities of the users.

Neuro-Fuzzy Systems are a class of systems provided with some intelligent features. In particular, the Neuro-Fuzzy System model known as ANFIS (Adaptive Neuro-Fuzzy Inference System) (Jang, 1993) has been widely used in different application scopes such as Robotics, Control Systems, Automotive, Pattern Recognition, etc. (Jang, 1993) (Jang and Sun, 1995), providing quit good results. The main feature of an ANFIS system relies on its ability to integrate in an efficient way knowledge representation, reasoning, and learning. That is, on the one hand it is able to describe a problem by means of linguistic rules of the form "IF-THEN" and to perform reasoning using fuzzy logic. On the other hand, it also exhibit learning performances typical of Neural-Networks. Hence, an ANFIS system can adapt its behaviour (i.e., its response) from a set of learning samples in order to handle new different siuations. Both reasoning and learning attributes constitute the powerful characteristics of Neuro-Fuzzy systems to be used in Intelligent Environments.

On the other hand, reconfigurable devices like FPGAs are a class of efficient electronic devices which are widely used in an increasing number of areas. Moreover, they are usually the main component of many embedded systems. They provide both high performance because they are hardware based and high flexibility because they can be easily reconfigured in order to modify or improve their functionality. In addition, the increasing complexity and integration of electronic devices have led to powerful new FPGA device families with a huge amount of resources, able to accommodate all the components of a typical embedded system (e.g., processor cores, memory blocks, peripherals, dedicated hardware, etc.), on a single chip, commonly referred to as system-on-a-programable-chip (SOPC). Hence, FPGA are appropiate devices to be used in Intelligent Environments where there could be a great number of such systems integrated in all the space in a non-intrusive way.

In this paper, we propose the development over reconfigurable devices (i.e., FPGAs) of an embedded electronic system for the intelligent control of an in-

Echanobe J., del Campo I. and Bosque G. (2009).
A NEURO-FUZZY EMBEDDED SYSTEM FOR INTELLIGENT ENVIRONMENTS.
In *Proceedings of the International Joint Conference on Computational Intelligence*, pages 559-564
DOI: 10.5220/0002321105590564

habited environment. In particular, an ANFIS-like hybrid Neuro-Fuzzy architecture has been designed and implemented. The system is provided with adaptation and learning features so it can fit the necessities and preferences of the users of the environment, including changes that can arise after a long period of time.

The rest of the paper is organised as follows: Section 2 introduces the proposed ANFIS-like neuro-fuzzy system. Section 3 shows an eficient heterogeneous hardware-software architecture to implement the PWM-ANFIS system. In Section 4, an embedded system intended for an Intelligent Environment is showed. Section 5 shows the implementation of the system which has been carried out on a FPGA device. Some details about the HW/SW requirements and also the performance of the system are presented here. Finally, some conclusions and future works are outlined.

2 ADAPTIVE NEURO-FUZZY INFERENCE MODEL

An ANFIS system (Jang, 1993) is a Fuzzy Inference System whose parameters (membership functions, strength of the rules, consequents, ...) are trained by means of Neural Network algorithms. The system can be viewed as a particular Neural Network that is functionally equivalent to a Fuzzy Inference System. Hence, it exhibits both the linguistic knowledge representation of fuzzy systems and the learning abilities of Neural Networks. On the other hand, note that as Neural Networks exhibit a notable parallelism in their structure, very efficient implementations can be achieved for ANFIS systems by means of hardware solutions (e.g. FPGAs based implementations (Omondi and Rajapakse, 2006)(del Campo et al., 2008)). Let us describe the structure of an ANFIS system:

Consider a rule based n-input fuzzy system with m antecedent functions per dimension. There are m^n rules where the j-th rule can be expressed as:

R_j : IF x_1 is $A^1_{j_1}$ and ... and x_n is $A^n_{j_n}$ THEN f is $c_{j_1 j_2 \cdots j_n}$

where $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, $\mathbf{x} \in R^n$ is the input vector; $A^1_{j_1}, \ldots A^n_{j_n}$ are linguistic labels associated with the membership functions $\mu_{ij}(x_i)$ (i.e., antecedents of the rules), y is the output variable, and $c_{j_1 j_2 \cdots j_n} \in R$ is a crisp consequent (i.e. a singleton). If the center of gravity defuzzification method is adopted, the output of the system is given by (1) being $w_j = A^1_{j_1}(x_1) \cdot A^2_{j_2}(x_2) \cdot \cdots \cdot A^n_{j_n}(x_n)$.

$$f = \frac{\sum_{j=1}^{m^n} w_j c_j}{\sum_{j=1}^{m^n} w_j} \quad (1)$$

This fuzzy system can be represented by a neural network (Figure 1) where the first layer computes the membership functions; the second layer computes the values w_j ($1 \leq j \leq m^n$); the third layer normalizes these values; the fourth layer computes $\overline{w}_j c_j$ ($1 \leq j \leq m^n$). Finally the output of the network is provided by the fifth layer which aggregates the overall output as the summation $f = \sum \overline{w}_j c_j$. It can be easily shown that this network performs the same function as the fuzzy system (Jang, 1993). Regarding the computational cost, it is required 1 division, $m^n \cdot (n+1)$ products, $2 \cdot m^n$ sums and $m \cdot n$ membership evaluations to perform an inference.

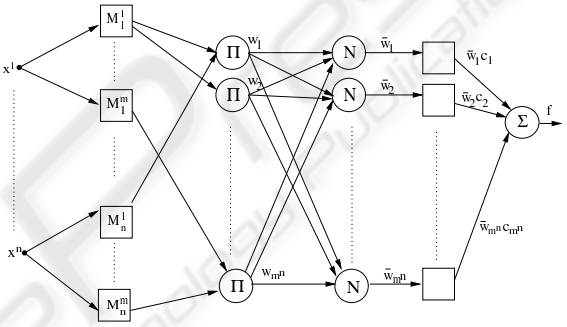

Figure 1: ANFIS Network structure.

To train the above network, a hybrid algorithm has been proposed (Jang, 1993). The algorithm is composed of a forward pass which is carried out by a Least Squares Estimator (LSE) process, followed by a backward pass which is carried out by a back propagation (BP) algorithm. The LSE computes the consequents and the back propagation adjusts the parameters of the antecedents. Although the back propagation algorithm could train the network alone, the training process can be accelerated with the LSE. This is possible due to the fact that the output of the network is linear in the consequent parameters.

Due to the huge amount of information and the large number of parameters that are involved in an Intelligent Environment, we have used a modified model called PWM-ANFIS (PieceWise Multilinear ANFIS) which we have proposed (del Campo et al., 2008) (Echanobe et al., 2008) to reduce the computational requirements in high dimensionality systems. In particular, we have shown how, by introducing some restrictions on the membership functions (i.e., antecedents), a much more simplified system can be obtained with hardly any loss of the learning and approximation capabilities. These restrictions basically involve the use of normalized triangular membership

functions overlapped by pairs. As a result, it is only required $2^n(2n+1)+n$ products and 2^n+2n sums to perform an inference. As we can see, this cost is drastically less than the initial one and it does not depend on the number of membership functions m. Moreover, as n and m increase, the difference between the two costs becomes huge.

3 HARDWARE/SOFTWARE ARCHITECTURE

Neural Networks contain two types of algorithms: On the one hand there are training algorithms (e.g. BP, LSE) that exhibit high irregularity, require high precision arithmetics and demand high computational cost. On the other hand, we have the computation of the network itself, which provides the network output for a given input (i.e., the inference of the rules in the ANFIS). This process implies regular and repetitive calculations which are suitable to be performed in parallel.

Taking into account these two different algorithms we propose an heterogeneous HW/SW architecture to implement the PWM-ANFIS (see Figure 2). HW implementations can exploit the parallel nature of the algorithms, therefore the computation of the network output will be implemented in the HW partition. Moreover, as it is shown in Figure 2, many PWM-ANFIS subsystems may be implemented all at once because many parameters must be simultaneously controlled in an Intelligent Environment. On the other hand, the training algorithms are implemented clearly better in software and hence they will be addressed in the microprocessor-based system. In fact, an analysis in detail of the hybrid training algorithm (LSE+BP) shows that a hardware implementation would be highly complex. The control of the global process and the input-output data handling is also carried out by this software partition. Although all the process can be developped in software, the benefits of the hardware parallelism are huge. For example, if we observe the complexity expresions in the previous section, we have that it is required 56 products and 14 sums to perform an inference in a 3-input system. The hardware partition can perform all these operations in parallel consuming many few clock cycles whereas a microprocessor would spent at least 70 (56+14) cycles. Moreover, if several systems are implemented simultaneously the difference is outstanding. In next sections, we will show the implementation of 4 systems in the same FPGA.

In particular we will develop our implementation over an FPGA device that integrates together with the reconfigurable hardware one (or more) micropocessor core. Hence, the Software partition will be addressed in this core while the HW partition will use the rest of the FPGA. Therefore, all the system is integrated on a single device with the corresponding advantages in size, power, and cost.

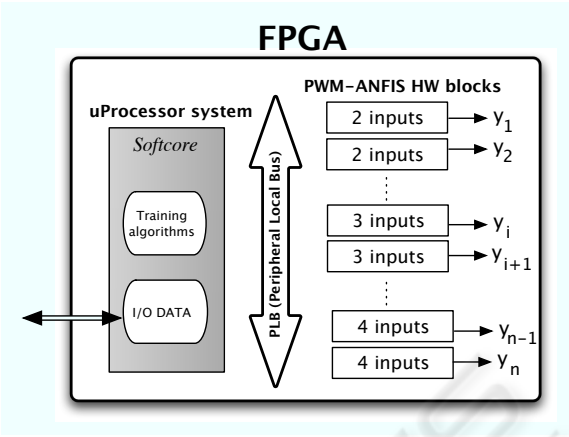

Figure 2: Architecture of the proposed system.

4 DEVELOPED SYSTEM

An Intelligent Environment can be understood as a collection of distributed devices (i.e., sensors and actuators) linked with one or more electronic control units (ECUs) (Schoitsch and Skavhaugh, 2006) (i.e., microprocessors, microcontrollers, FPGAs, ...) endowed with robust algorithms which implies ”smart” or ”intelligence” capabilities. In this work, we have carried out the design, implementation and verification of an embedded electronic system for one of this ECUs, following the architecture described in the previous section. We assume that some ambiental parameters values are provided by external sensors and that we can modify some others via actuators. In particular, we deal with 6 input parameters of a room or salle: External Temperature, Internal occupancy (i.e., persons inside the room), Internal Noise, External Light level, External Humidity and External Pollution. The output variables of the system are: Internal Temperature, Internal Light Level, Volume of the Sound (if music is being heard) and Intensity of an Air-Freshener or a Fragance. The control of these four variables are achieved by 4 PWM-ANFIS subsystems whose dependence on the input variables is showed in Table 1. Thus, we have two 2-input subsystems and two 3-input sub-systems. The choice of these variables, their dependences and the behaviour of the environment have been provided by an expert from an architecture and environment-design com-

pany. In particular, this knowledge has been expressed by means of linguistic rules of the form "IF-THEN". From this set of rules a Fuzzy System has been built up and from it, a collection of input-output data has been generated to train the PWM-ANFIS.

Table 1: Dependence between input and ouput environment parameters in the 4 PWM-ANFIS Subsystems.

Inputs	Subs.1 Temp.	Subs.2 Light	Subs.3 Volum.	Subs.4 Aroma
Occupancy	X		X	X
Ext. Light	X	X		
Ext. Temp.		X		
Pollution		X		X
Int. noise			X	
Humidity				X

In order to verify that the PWM-ANFIS system preserves good approximation capabilities despite the restrictions imposed on the antecedent functions, we have compared the output surfaces obtained a) by the PWM-ANFIS system and b) by using the target function derived from the set of linguistic rules (i.e., from the constructed fuzzy system). Also the training error curves have been represented to analyze the learning process. Two of the four sub-systems described above are showed here: A 2-input system and a 3-input system. In the following section we present these results.

4.1 2-Input PWM-ANFIS

As an example of a 2-input system, we show here the results obtained with the first sub-system of Table 1: the control of the internal Temperature with respect to the occuppancy and to the External Light Level. To train this system, 121 input-output data pairs $((x_1, x_2), f)$ are collected by taking 11 equally distributed values for both the Occupancy (x_1) and the External Light Level (x_2), and computing the Internal Temperature values (f) by the fuzzy system as we have already commented. The number of antecedent functions per input variable has been set to 5 so we obtain a total of 25 rules and consequents. Figure 3 shows the output surfaces for this example (all the variables have been normalized to the interval [0,1]): a) The target function to be approximated; b) The approximated function obtained with the PWM-ANFIS system once it has been trained. We can see how the approximated surface is quite similar to the target one. On the other hand, Figure 4 shows the results of the learning procedure. It can be seen how the shape and position of the triangular membership functions (antecedents) have been modified from an initial symetric distribution to fit the training data. On the other hand the evolution of the error value - during the training phase - between the PWM-ANFIS output and the target function is shown. As we can see, a very small value is obtained even after few iterations, so the learning ability is still quite good.

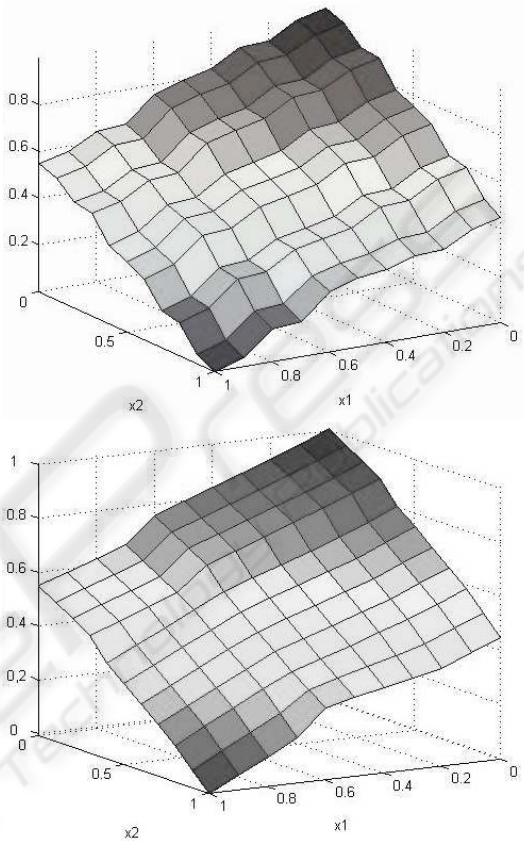

Figure 3: Output surfaces for Internal Temperature $y1$ regarding the Occupancy level (x1) and the External Light level(x2). Target surface (up) and PWM-ANFIS surface (down).

4.2 3-Input PWM-ANFIS

To illustrate a 3-input system we show here the subsystem used to control the Internal Illumination with respect to the External Illumination (x_3) , External Temperature (x_4), and the External Pollution (x_5). Again, 11 equally distributed values are taken for every input variable so a total of 1331(11x11x11) input-output data pairs are collected to train the system. In addition, five antecedents are set per dimension, leading to a total of 125 rules. Figure 5 shows the ouput surfaces for a fixed value of the External Pollution (i.e., so it can be represented). Again we can see that both surfaces are quite similar so the system also presents here good approximation performances. With regard to the learning process, the results are

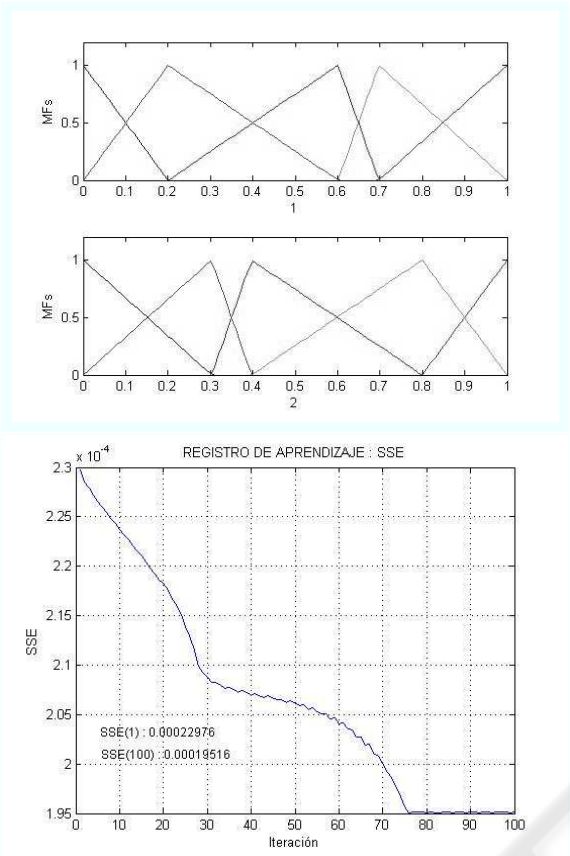

Figure 4: Example 1: triangular antecedents after training (up) and training error evolution (down).

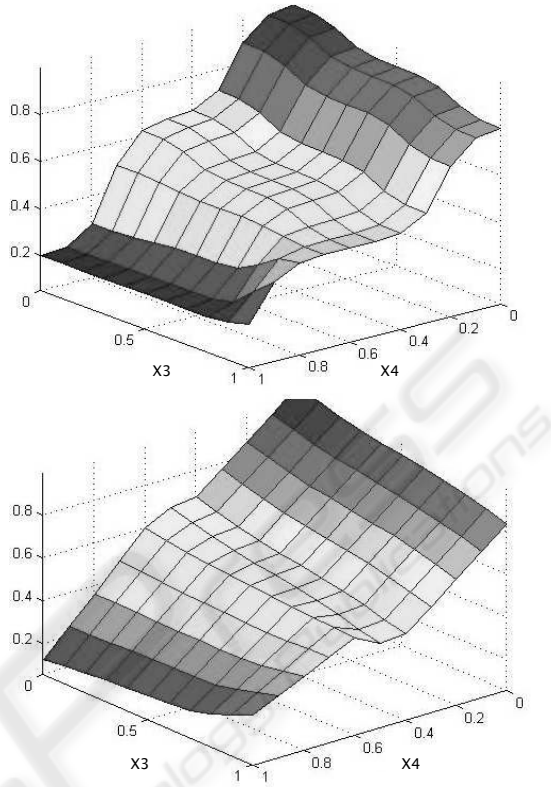

Figure 5: Output surfaces for Internal Illumination *y*2 regarding the External Illumination (x3) and the External Temperature (x4), for a fixed pollution value. Target surface (up) and PWM-ANFIS surface (down).

also satisfactory and again a very small error value is obtained after a small number of iterations (see Figure 6).

These two examples show the good approximation capability of the PWM-ANFIS. Obviously, the generic ANFIS (i.e., unrestricted gaussian membership functions) provides, general speaking, better approximation capability. However, the degree of approximation can be arbitrarily set in our system by taking a higher number of antecedents per dimension, *m*. Moreover, higher *m* does not increase the complexity of the system as it was shown in Section 2.

5 SYSTEM IMPLEMENTATION

The system has been implemented in a XILINX's FPGA, in particular over a XC2VP30 device from the VIRTEX-II Pro familly. Such a device is an intermediate one in the VIRTEX-II Pro familly and integrates a PowerPC-based microprocessor system, 13.696 Logic Blocks (Slices), 136 multipliers, embedded memory and some other resources. In addition, the ISE and EDK design tools, also from XILINX, have been used to design, synthetise, simulate and verify the implementation. The device resources used by the implementation are the following: On the one hand, each one of the two 2-input PWM-ANFIS subsystems requires 137 Slices (1%) and 10 multipliers (7%). On the other hand, each one of the two 3-input PWM-ANFIS subsystems requires 828 Slices (6 %) and 35 multipliers (25%). The complete system - including the HW/SW interface and the RAM memory for the software partition - makes use of 1375 Slices (10%), 40 multipliers (29%) and 64 RAM blocks (47%). The operation frequency of the system is 100MHz both for the microprocessor and for the clock of the hardware partition. Moreover, due to the high parallel degree achieved for the hardware design, the evaluation of every PWM-ANFIS (i.e., the inference of the rules for a given input) is achieved in just seven clock cycles (70ns). If these inferences were performed solely in software it would be necessary at least 196 cycles (152 products and 44 sums), as it is derived from expresions in Section 2.

Concerning the precision of the computations, the system provides its responses in a 32-bit signed data

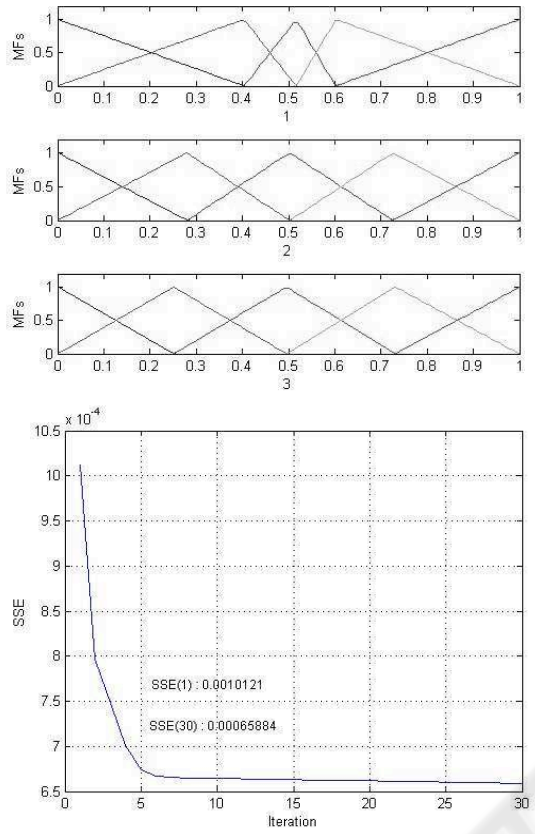

Figure 6: Example 2: triangular antecedents after training (up) and training error evolution (down).

format which is sufficient precision for this kind of application.

6 CONCLUSIONS

An efficient Embedded Electronic System for Intelligent Environments has been developped. The system is intended for the control of some environment parameters inside a room such as temperature, illumination etc. The system is based on a Neuro-Fuzzy model which provides features such as learning, adapting and fuzzy information processing. In particular a PWM-ANFIS Neuro-Fuzzy model has been adopted which is a computational-efficient version of the well known ANFIS model. First, some numerical simulations were carried out in order to validate the model, and the results obtained show an appropiate response. Finally, an efficient heterogeneous HW/SW implementation has been realised and tested. Training algorithm is excuted in software due to its irregularity and the required precission while the neural network computation is addressed in hardware to exploit its inherent parallelism. This implementation is performed over an FPGA device (i.e., a XILINX's VIRTEX-II Pro) and integrates in the same device four PWM-ANFIS models, each one controlling one environment parameter. It also integrates a microproccesor subsystem which executes the learning procedure and the input-output data handling. The whole design works at a frequency of 100MHz and all the four PWM-ANFIS systems perform simultaneously the rule inferences in just 70ns. To improve even more the efficiency of the proposed system, future work will be devoted to make use of the Dynamic Partial Reconfiguration techniques so that the size and the power consumption of the systems can be optimized.

ACKNOWLEDGEMENTS

This work was supported in part by the Basque Country Government under Grants GIC07/138-IT-353-07 and S-PE08UN49.

REFERENCES

del Campo, I., Echanobe, J., Bosque, G., and Tarela, J. (2008). Neuro-fuzzy modeling and control. *IEEE Transactions on Fuzzy Systems*, 16(3):761–778.

Echanobe, J., del Campo, I., Bosque, G., and Tarela., J. (2008). An adaptive neuro-fuzzy system for efficient implementations. *Information Sciences*, 178:2150–2162.

ESTO (2003). *Science and Technology Roadmapping: Ambient Intelligence in Everyday Life (AmI@Life)*. European Science and Technology Observatory (ESTO).

ISTAG (2001). *Scenarios for Ambient Intelligence in 2010*. IST Advisory Group (ISTAG), European Commission Community Research.

ISTAG (2005). *Ambient Intelligence: from vision to reality*. IST Advisory Group (ISTAG), European Commission Community Research.

Jang, J. . R. (1993). Anfis: Adaptive-network-based fuzzy inference system. *IEEE Trans. Systems, Man, and Cybernetics*, 23(3):665–685.

Jang, J. . R. and Sun, C.-T., M. (1995). Neuro-fuzzy modeling and control. *Proceedings of the IEEE*, 83(3):378–406.

Omondi, A. R. and Rajapakse, J. C. (2006). *FPGA Implementations of Neural Networks*. Springer.

Schoitsch, E. and Skavhaugh, A. (2006). Embedded intelligence. *ERCIM News*, 67:14–15.