A learning procedure for a fuzzy system: application to obstacle avoidance

Jelena Godjevac
EPFL Microcomputing Laboratory
IN-F Ecublens
CH-1015 Lausanne
Jelena.Godjevac@di.epfl.ch

Abstract

The goal of this work is to propose a learning procedure for fuzzy systems. Fuzzy systems are able to treat uncertain and imprecise informations. They have a capability to express knowledge in the form of linguistic rules. Their drawbacks are caused mainly by the difficulty of defining accurate membership functions and lack of a systematic procedure for the transformation of the expert knowledge into the rule base. Neural networks have the ability to learn but both knowledge extraction and knowledge representation are difficult. First, a neuro-fuzzy architecture is proposed. A learning procedure based on the stochastic approximation method is described. The methodology is the supervised learning method developed in the field of neural networks. In order to discuss the validity of the proposed method, three numerical examples are presented and it is shown that the proposed neuro-fuzzy system have the ability to learn.

It is applied to the obstacle avoidance problem of a mobile robot. As an experimental platform, the Khepera mobile robot is used. To conclude, the application of a neuro-fuzzy controller for Khepera is discussed.

Keywords

Neuro - Fuzzy Systems, Stochastic Approximation Method, Supervised Learning, Mobile Robots.

1 Introduction

Fuzzy systems have the attribute of expressing knowledge in the form of *linguistic rules*. They offer a possibility to implement expert human knowledge and experience. Their main drawback is the lack of a systematic methodology for their design. Usually, tuning parameters of membership functions is a time consuming task. *Neural network* learning techniques can automate this process, significantly reduce development time, and result in a better performance. The merge of neural networks and fuzzy logic led to the creation of *neuro-fuzzy controllers* which are one of the most popular research fields today. There are some very interesting architectures for neuro-fuzzy controllers proposed by Jang [Jan92] and Nomura [Nom92].

In this paper, we propose a simple neuro-fuzzy controller and a learning method for it. The idea is to simplify fuzzy system in order to linearise equations between variables in the system.

A learning procedure is a parameter estimation problem. There are several methods described in the literature [Ast71]. On of the most widely used algorithm is *least-mean square* (LMS) algorithm based on the idea of stochastic approximation method [Lju83]. It is used in adaptive signal processing for adjusting the weights in a linear adaptive system. It was derived and first applied in that context by Widrow and Hoff [Wid85].

The learning procedure that we present is based on the *stochastic approximation method* for the adjustment of parameters of a nonlinear system. The methodology corresponds to the *supervised learning*, developed in the field of neural networks. Supervised learning procedures are, in general, the application of adaptive algorithms and the scheme of a closed-loop adaptive system (Figure 1). We will define the response signal which is assumed to represent the desired output of the adaptive system. In this case it will be the fuzzy system. The error signal is the difference between the desired output and the actual output of the system. Using it, an adaptive algorithm adjusts the parameters of the system, thus altering its response characteristics by minimizing a

In Proceedings of the *International Symposium on Fuzzy Logic*, Zürich 1995, pages C 142 - 148

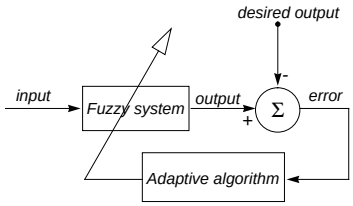


Figure 1. Closed loop adaptation

measure of the error, thereby closing the performance loop. We applied this method to the emulation of three nonlinear functions and to the obstacle avoidance of a Khepera mobile robot [MON94].

2 Neuro-fuzzy controller

In this section we describe a learning procedure for a neuro-fuzzy controller represented in the Figure 2. The starting point is the Takagi - Sugeno

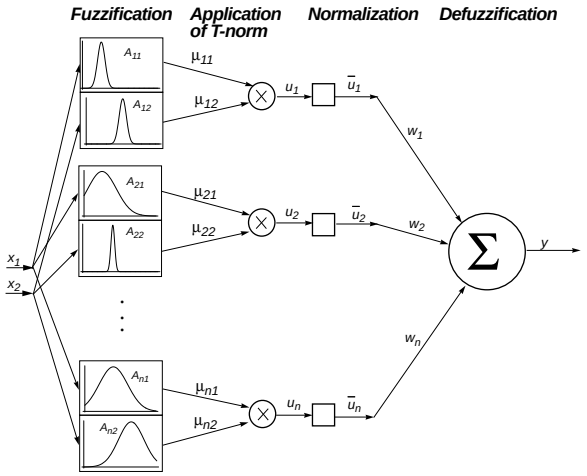


Figure 2. Neuro-fuzzy controller

method [TAK83] where output membership functions are defined as singletons.

2.1 Linguistic rules

Assume that a fuzzy controller has m inputs $x_1, x_2, \ldots, x_m$ and one output y and that we defined n linguistic rules in the form:

$\boldsymbol{R_i}$: **If** x_1 is A_{i1} **and** x_2 is A_{i2} **and** ... **and** x_m is A_{im} **then** y is w_i $\quad i = 1, \ldots, n$

where i is the index of the rule, A_{ij} is a fuzzy set for i-th rule and j-th linguistic variable defined over the universe of discourse for j-th variable and w_i is a real number that represents a consequent part.

2.2 Membership functions

Every membership function in this controller is defined as a Gaussian function as shown in Figure 3. The definition of the membership values is given by

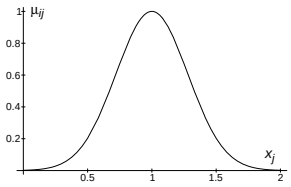


Figure 3. Membership function ($a_{ij} = 1, b_{ij} = 0.28$)

the eq. (1). For every input the universe of discourse is defined as a set of uniformly distributed membership functions (See Table 1).

$$\mu_{ij} = e^{\frac{-(x_j - a_{ij})^2}{2b_{ij}^2}} \tag{1}$$

2.3 Fuzzy inference

For the application of the rules we need to define a fuzzy inference, and in this case, we will take product operator as T-norm. It means that the firing strength of every rule is:

$$u_i = \mu_{i1}\mu_{i2}\cdots\mu_{im} \tag{2}$$

The consequent part of the rules are crisp values, and the evaluation of a centre of gravity is given by:

$$\begin{aligned} y &= \frac{\sum_{i=1}^{n} u_i w_i}{\sum_{i=1}^{n} u_i} \\ &= \frac{u_1}{\sum_{i=1}^{n} u_i} w_1 + \frac{u_2}{\sum_{i=1}^{n} u_i} w_2 + \cdots + \frac{u_n}{\sum_{i=1}^{n} u_i} w_n \\ &= \sum_{i=1}^{n} \overline{u_i} w_i \end{aligned} \tag{3}$$

A general scheme of this system with two inputs and one output is represented in Figure 2. This presentation is analogous to the architecture of a feed-forward artificial neural network.

The first layer performs a fuzzification for each linguistic rule. Outputs from this layer are membership values (eq. 1) and they are fed into the next layer which performs a T-norm operation –product– (eq. 2). The result is a firing strength for each rule and in the next layer, firing strengths are normalized (eq. 3). The last layer computes the overall output as the weighted sum of the incoming signals.

2.4 Computational aspects

Suppose that the parametric model of one nonlinear system is given in the following form:

$$\frac{dx}{dt} = f(x, u, z, t)$$

$$y = g(x, u, z, t)$$

where x is a state vector, u an input vector, z a parameter vector and y is the output vector. All solutions to parameter estimation problems consist in finding the extremum of *criterion (loss) function* V considered as a function of the parameters of the unknown system. The function we minimize is:

$$V(z) = \frac{1}{2}\left(y(t) - y_d(t)\right)^2 \tag{4}$$

where y_d is the *desired output* provided by an expert. The minimization of this function can be done in several ways [AST71]. All of them use the scheme suggested by Robbins and Monro in [ROB51]. Here we will use the criterion (eq. 4) and apply the method of stochastic approximation to identify the parameters of the fuzzy system. It is an iterative procedure given by:

$$z(t+1) = z(t) - \Gamma \nabla_z V\left[z(t)\right]$$

where z is the vector of parameters to adapt, Γ is the predefined constant and $-\nabla_z V$ is the notation for the gradient of V with respect to z:

$$\nabla_z V = \left[\frac{\partial V}{\partial z_1}, \cdots, \frac{\partial V}{\partial z_n}\right]$$

2.5 The learning algorithm

In the fuzzy controller presented above, z is given by:

$$z = (a_{11}, \ldots, a_{nm}, b_{11}, \ldots, b_{nm}, w_1, \ldots, w_n)$$

There are three types of parameters to adapt: center values a_{ij}, width values b_{ij} and consequent values w_i. The number of parameters to adapt is $p = 2nm + n$. The vector which minimizes the loss function is given by:

$$\left(-\frac{\partial V}{\partial z_1}, -\frac{\partial V}{\partial z_2}, \cdots, -\frac{\partial V}{\partial z_p}\right) = 0$$

and the learning rule:

$$z_k(t+1) = z_k(t) - \Gamma \frac{\partial V(z)}{\partial z_k} \quad , \quad k = 1, \ldots, p \tag{5}$$

It follows from (eq. 1 - 5) that the equations for the adaptation of the parameters of fuzzy system are:

$$\begin{aligned} a_{ij}(t+1) \;=\; & a_{ij}(t) - \\ & \Gamma_a \frac{u_i}{\sum_{l=1}^{n} u_l}(y - y_d)\left(w_i - y\right) \\ & \frac{\left(x_j - a_{ij}(t)\right)}{b_{ij}^2} \end{aligned} \tag{6}$$

$$\begin{aligned} b_{ij}(t+1) \;=\; & b_{ij}(t) - \\ & \Gamma_b \frac{u_i}{\sum_{l=1}^{n} u_l}(y - y_d)\left(w_i - y\right) \\ & \frac{\left(x_j - a_{ij}(t)\right)^2}{b_{ij}^3} \end{aligned} \tag{7}$$

$$\begin{aligned} w_i(t+1) \;=\; & w_i(t) - \\ & \Gamma_w \frac{u_i}{\sum_{l=1}^{n} u_l}(y - y_d) \end{aligned} \tag{8}$$

The iterative procedure for the adaptation of parameters and for the minimization of the criterion function can be summarized as follows:

1. Initialization of parameters.
   - Consequent values w_i are random numbers.
   - Choice of antecedent parameters a_{ij} and b_{ij} (all membership functions on the universe of discourse are regularly distributed and have the same width).
2. Data input $(x_1, x_2, \ldots, x_m, y_d)$.
3. Fuzzy inference.
4. Adaptation of consequent parameters w_i.
5. Adaptation of parameters a_{ij} and b_{ij}.
6. Evaluation of the criterion function V.
7. Repeat the steps 3 to 7 until V is smaller than one threshold value.

This algorithm is more efficient if the steps 3 and 4 are repeated two times in the same iteration loop. It means that the weights w_i are adapted two times in one iteration. This method is similar to the method proposed in [NOM92].

2.6 Numerical examples

In order to test the learning capacities and the validity of the proposed method, three nonlinear functions were emulated. They have two inputs x_1, x_2

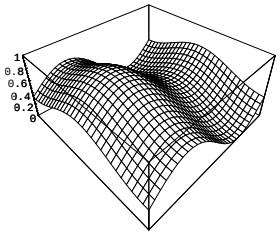
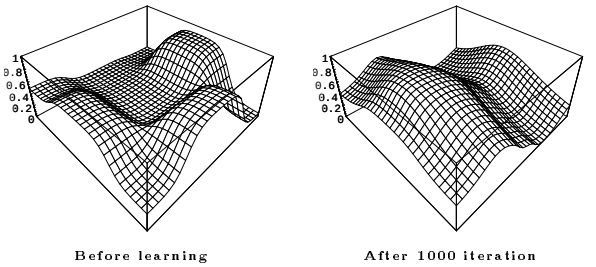
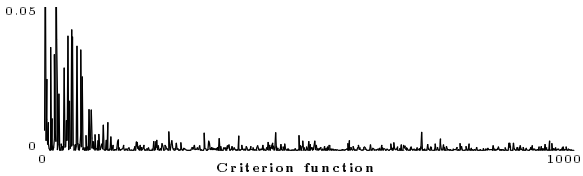
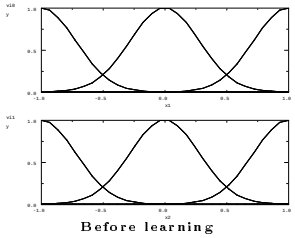
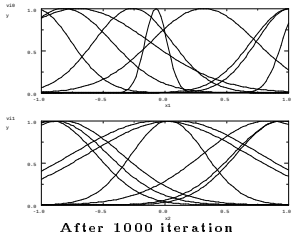
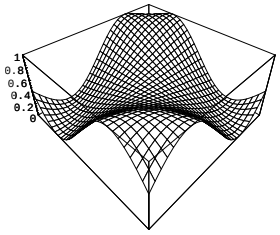
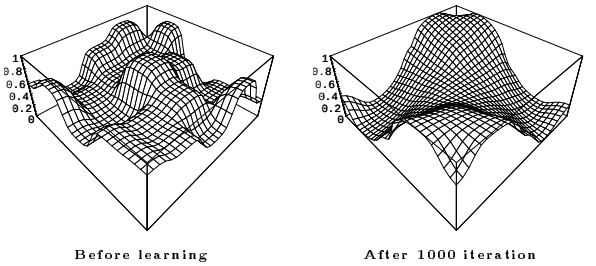
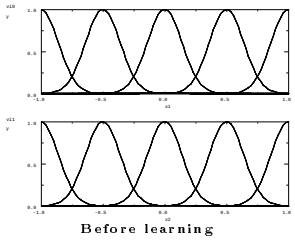
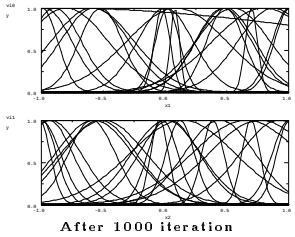
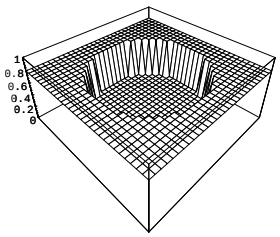
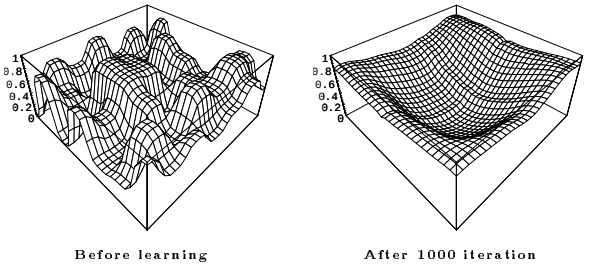
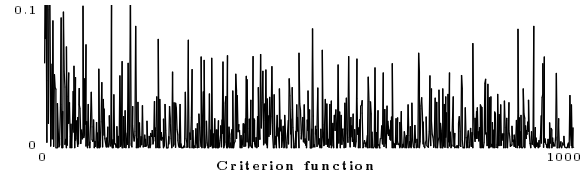
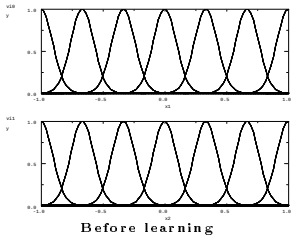
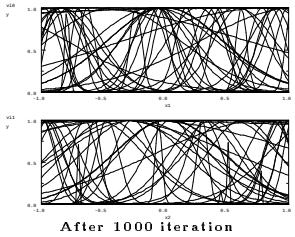
Function	Emulated function and Criterion function	Membership functions
$y = 4\sin(\pi x_1) + 2\cos(\pi x_2)$ 	 	 
$y = \sin(3x_1x_2)$ 	 	 
$y = \begin{cases} 0.8 \text{ for } x_1^2 + x_2^2 > 0.5 \\ 0.2 \text{ otherwise} \end{cases}$ 	 	 

Table 1. Results of simulation

and one output y and they are given by their analytical equations. (See Table 1).

The input - output data are random numbers within $[-1, 1]$, and output data y_d is normalized within $[0, 1]$. The results show that for the first function the criterion function V (squared error) decreases bellow 0.001 (0.1%) after only 200 iterations. For the second one the squared error decreases with almost same speed. For the third function the error stays about 5%. The reason is that the shape of membership functions is not appropriate for the approximation of the non-continuous functions.

Table 1 illustrates also the shape of emulated functions before learning and after 1000 iterations, as well as the distribution of membership functions before and after learning. The learning speed increases if we the number of membership functions is larger.

The system shown in the Table 1, function 1 is simulated in [NOM92] with a neural network and the number of learning iterations is more than 35000. In that case, the neural net consisted of three layers including two units in the input layer, sixteen units in the hidden layer and one unit in the output layer. The learning method is the back-propagation algorithm.

The advantage of the proposed learning method for a fuzzy system is that the learning speed is higher than that of a conventional back-propagation type neural networks. Unlike neural nets, this fuzzy system has a capability to express the knowledge acquired from input-output data in the form of fuzzy inference rules.

3 Khepera mobile robot

Suppose that the first goal is to do obstacle avoidance with a mobile robot. Whatever the mobile robot has to perform, it has to avoid obstacles. Follow a guide, transport objects, go to a goal... All these actions are supposed not to damage the robot's hardware. Nor its environment!

Khepera is a miniature mobile robot (Figure 4) developed in our lab [MON94]. It has cylindric shape, measuring 55 mm in diameter and 30 mm in height. Its weight is only 70 g.

The basic configuration of Khepera is composed of the CPU and of the sensory/motor boards. The CPU board is a complete 32 bit machine including a 16 MHz microcontroller, system and user memory, analogue inputs, extension busses and a serial link allowing a connection to different host machines (terminals, visualisation software tools, etc.). The sensory/motor board includes two DC motors coupled with incremental sensors, eight analogue infrared (IR) proximity sensors (S0...S7 on Figure 5) and

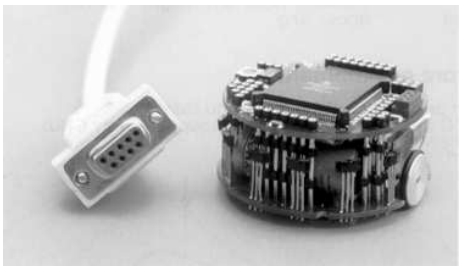


Figure 4. Khepera mobile robot

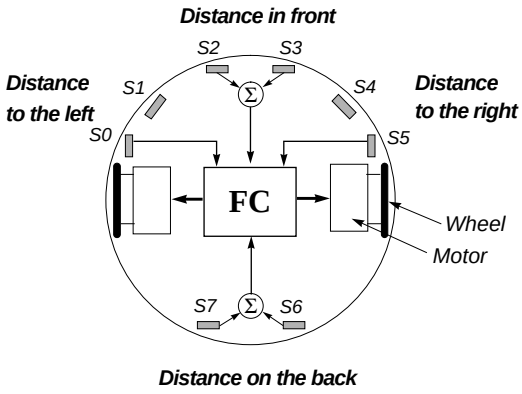


Figure 5. Position of the sensors on the Khepera mobile robot

on-board power supply. IR sensors are composed of an emitter and of an independent receiver. The dedicated electronic interface is built with multiplexers, sample/hold's and operational amplifiers. This allows the measurement of the absolute ambient light and the estimation, by reflection, of the relative position of an object from the robot. This estimation gives, in fact, an information about the distances between the robot and the obstacles.

What can one do with a robot that has 8 not very accurate sensorsΓ Apply fuzzy logicΓ It can be one of the ideas. There are some others: genetic algorithms [FLO94], subsumption architecture [BRO86], distributed adaptive control [VER92], learning with neural networks [GAU94, ZRE94], artificial intelligence approaches [BAS93]. The simple fuzzy controller for the obstacle avoidance for Khepera is described in details in [GOD95]. The controller has 4 inputs and 2 outputs as shown in Figure 5. The robot avoids obstacles very successfully. The critical problem is that the number of possible rules is very large and it is very difficult to make appropriate definitions of parameters for membership functions. Moreover, a very important factor is the big computational time and this was one of the main reasons

that led us to simplify the fuzzy controller and to develop a learning procedure for it.

3.1 Application of the neuro-fuzzy controller to the Khepera

We applied the learning procedure described in Section 2.5 to the obstacle avoidance problem of the Khepera mobile robot. The goal was to adjust the parameters of membership functions. The robot started with 625 basic rules and non-adjusted membership functions. During the learning phase, the adaptation is done each time the robot encounters an unknown situation.

In the first step we have limited the learning method to the adaptation of parameters with a fixed rule base. After learning, we suppressed all rules not used in the supervised learning phase. The problem is that some membership functions "disappeared" from the universe of discourse or overlapped with other functions. It was necessary to make an analysis of the obtained results and to suppress useless functions and rules.

4 Conclusion

The advantage of fuzzy systems is that knowledge is represented in the form of the easily comprehensible linguistic rules. On the other hand, neural networks have the ability to learn. This leads to the idea of supplying fuzzy controllers with learning rules. This was implemented on a controller based on the Takagi-Sugeno method. The learning procedure is the stochastic approximation method and the adaptation corresponds to supervised learning of neural networks.

To test the validity of the proposed procedure, three nonlinear functions were emulated. The results show that fuzzy systems have an ability to learn and that the learning methods used in neural network field can be applied. The advantage of the proposed learning method for a fuzzy system is that the learning speed is higher than that of a conventional back-propagation type neural networks. Unlike neural nets, this fuzzy system has a capability to express the knowledge acquired from input-output data in the form of fuzzy inference rules.

We applied this procedure to the obstacle avoidance problem on the Khepera mobile robot. The goal was to adjust the parameters of membership functions using supervised learning procedure. The robot learned very successfully to avoid obstacles.

Acknowledgements

The author would like to thank André Guignard, Edo Franzi and Francesco Mondada for the design of Khepera and Yves Chenval for the design of Packlib environment. She also gratefully acknowledge the constructive criticism on the manuscript and the constant support of Laurent Tettoni, Paolo Ienne and Jean - Daniel Nicoud.

References

[Ast71] K.J. Astrom and P. Eykhoff. System identification - a survey. *Automatica*, 7:123–162, 1971.

[Bas93] A. Basso, F. Mondada, and C. Castelfranchi. Reactive goal activation in intelligent autonomous agent architectures. In *AAI93 symposium on Abstract Intelligent Agents*, Rome, January 1993.

[Bro86] R. Brooks. A robust layered control system for a mobile robot. *IEEE Robotics and automation*, RA-2:14–23, March 1986.

[Flo94] D. Floreano and F. Mondada. Automatic creation of an autonomous agent: Genetic evolution of a neural-network driven robot. In *SAB94*, Brighton, 1994.

[Gau94] Ph. Gaussier and S. Zrehen. A topological neural map for on-line learning: Emergence of obstacle avoindance. In *Proceedings of SAB, From Animals to Animats 94*, pages 182–190, Brighton, 1994.

[God95] J. Godjevac. Comparative study of fuzzy control, neural networks and neuro-fuzzy control. In Da Ruan, editor, *The Stimulating Role of Fuzzy Set Theory in Mathematics*, chapter 13. Kluwer Academic, June 1995. In Press.

[Jan92] J.-S. Roger Jang. Anfis, adaptive-network-based fuzzy inference systems. *IEEE Trans. on systems, Man and Cybernetics*, 1992.

[Lju83] Ljung, L. and Soderstrom, T. *Theory and Practice of Recursive Identification*. Prentice Hall signal processing series. The MIT Press, Cambridge Massachusetts, London England, 1983.

[Mon94] F. Mondada, E. Franzi, and P. Ienne. Mobile robot miniaturization: A tool for investigation in control algorithms. *Informatik*, pages 17–20, February 1994.

[Nom92] H. Nomura, I. Hayashi, and N. Wakami. A learning method of fuzzy inference rules by descent method. In *Proceedings of IEEE Int. Conf. on Fuzzy Systems*, pages 203–210, San Diego, 1992.

[ROB51] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[TAK83] T. Takagi and M. Sugeno. Derivation of fuzzy control rules from human operator's control actions. In *Proc. of the IFAC Symp. on Fuzzy Information, Knowledge Representation and Decision analysis*, pages 55–60, July 1983.

[VER92] P. F. M. J. Verschure, B. J. A. Koese, and R. Pfeifer. Distributed adaptive control: The self-organization of structured behavior. *Robotics and Autonomous Agents*, 9:181–196, 1992.

[WID85] Widrow, B. and Stearns, S. D. *Adaptive Signal Processing*. Prentice Hall signal processing series. Englewood Cliffs, N.J., 1985.

[ZRE94] S. Zrehen and Ph. Gaussier. Why topological maps are useful for learning in an autonomous agent. In *Proceedings PerAc*, Lausanne, September 1994. IEEE Press.