
COMPARATIVE STUDY OF FUZZY CONTROL, NEURAL NETWORK CONTROL AND NEURO-FUZZY CONTROL

Jelena Godjevac

*Microcomputing Laboratory
Swiss Federal Institute of Technology
IN-F Ecublens, 1015 Lausanne
Switzerland*

ABSTRACT

The goal of this work is to compare fuzzy, neural network and neuro-fuzzy approaches to the control of mobile robots. The first part of this paper is devoted to the formal framework of fuzzy controllers. Results of an example of their use for a mobile robot are discussed. As an experimental platform, the Khepera mobile robot is used. The same example is studied using artificial neural networks. For that purpose, fundamentals of artificial neural networks are outlined. Similarities and differences between fuzzy systems and neural networks are discussed as well as the respective advantages and drawbacks, and reasons for merging these two approaches are developed. Three models of fuzzy neurons, the learning methods and an architecture of neuro-fuzzy controller are presented. A learning procedure for the controller is described. To conclude, the application of a neuro-fuzzy controller on Khepera is discussed.

1 INTRODUCTION

The word *robot* (robota) has Slavic origins and means work. Robots are built to replace human beings in performing tasks that humans cannot or prefer not to do. They can clean dangerous areas in nuclear plants, transport goods, clean floors, search for mines or harvest crops in fields. Present technology is far too deficient to make robots accomplish these complicated tasks. This paper surveys very simple controllers based on *fuzzy logic* and/or *neural networks* for the *control of mobile robots*. The purpose is to investigate those solutions most likely to be found in nature which, notwithstanding an astonishing simplicity

can produce complex behaviours. For instance, in spite of a relatively simple nervous system, an ant may show many different behaviours. In the same way as animals in their natural environment, real mobile robots have to achieve tasks in unpredictable conditions.

Fuzzy logic was one of the major developments of *Fuzzy Set Theory* and was primarily designed to represent and reason with knowledge that cannot be expressed by quantitative measures. The main idea of algorithms based on fuzzy logic –loosely called *fuzzy systems* or *fuzzy controllers*– is to imitate the human reasoning process to control ill-defined or hard-to-model plants. Fuzzy inference systems model the qualitative aspects of human knowledge through *linguistic if-then rules* [ZAD65]. They have to capture the imprecision of the reasoning process without using exact quantitative analysis. We do not pretend the fuzzy logic approach to be a methodology borrowed from biology or having biological foundations. It is a rigorous mathematical branch offering solutions for control. An example of *fuzzy control* is shown in Section 3.1. We report experiments with a small mobile robot, Khepera, developed in our laboratory.

Neural networks were developed as an attempt to realise simplified mathematical models of brain-like systems. The key advantage is their ability to learn from examples instead of requiring an algorithmic development from the designer. One of the main motivations of the interest in neural networks is to build machines with the intelligence of biological organisms. In Section 4.3, we describe a neural network approach applied to the Khepera robot.

The main drawback of fuzzy controllers is the lack of a systematic methodology for their design. Usually, tuning parameters 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. Lotfi Zadeh proposed the name *soft computing* for these techniques [ZAD94]. The principal constituents of soft computing are: fuzzy logic, neural networks and probabilistic reasoning. An example of a neuro-fuzzy controller is presented in Section 6 as well as its implementation on Khepera.

2 FUZZY CONTROL

Humans, when making decisions tend to work with vague or imprecise concepts which can often be expressed linguistically. One of the ways of modelling this decision making process has been proposed by Zadeh [ZAD65] and is based on the *Theory of Approximate Reasoning* which enables certain classes of linguistic statements to be treated mathematically. First investigations by Prof. Zadeh [ZAD73] concerned how to use mathematical tools to represent a human language and human knowledge. He was the first to introduce the fuzzy set theory in the field of control. He proposed that all problems in which the data, the objectives and the constraints are too complex, or too ill-defined to admit a precise mathematical analysis have to be treated by approximate (fuzzy) solutions. Fuzzy control has received a lot of attention since it was applied for the first time by Mamdani and Assilian [MAM75].

The main argument in favour of fuzzy control is that conventional mathematical tools are not well suited for dealing with ill-defined and uncertain systems that are typically difficult to model. Some authors argue that fuzzy controllers are suitable where a precise mathematical model of the process being controlled is not available [KIC78, LI88]. However, it is not possible to build a controller which does not utilise environmental information.

2.1 Linguistic variables and membership functions

Linguistic variables represent process states and control variables. We can use *fuzzy sets* to represent them. Every fuzzy set can be represented by its *membership function*. If the referential set is a finite set, membership values are discrete values defined in the range $[0,1]$. If the referential set is an infinite set, we can represent these values as a continuous membership function. In general, the shape of a membership function depends on the application and can be monotonous, triangular, trapezoidal or bell-shaped as shown in Figure 1.

One of the first steps in every fuzzy application is to define the *universe of discourse* (dynamic range) for every linguistic variable. Every fuzzy set on a universe of discourse represents one *linguistic value* or *label*. Figure 2 illustrates one example of the universe of the discourse for the linguistic variable speed. Linguistic values, which define this variable are: NEGATIVE BIG, NEGATIVE MEDIUM, etc.

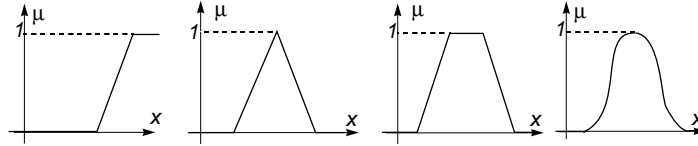


Figure 1 Different shapes of membership functions: monotonous, triangular, trapezoidal and bell-shaped

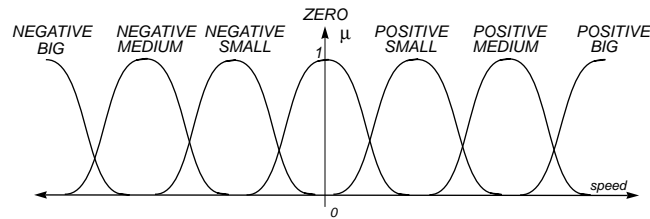


Figure 2 Universe of discourse for linguistic variable: speed

2.2 Notion of linguistic rule

As mentioned above, the principal idea of fuzzy logic systems is to express human knowledge in the form of linguistic if-then rules. Every rule has two parts:

- antecedent part (premise), expressed by *If...*
- consequent part, expressed by: *then...*

The antecedent part is the description of the state of the system which should turn on the rule, and the consequent is the action that the operator who controls the system must take. There are several forms of if-then rules. The general one is:

If a set of conditions is satisfied *then* a set of consequences can be inferred.

Zadeh was the first to introduce a notion of fuzzy rule [ZAD73] of the form:

R_1 : *If* x is A , *then* y is B .

A and B are linguistic values or labels characterized by appropriate membership functions of fuzzy sets defined on the universe of discourse of the linguistic variables x and y respectively. Every control rule is implemented by a *fuzzy implication (fuzzy relation)* and is defined as follows:

$$\mu_{R_1} = \mu_{A \rightarrow B}(x, y) = \mu_A(x) \rightarrow \mu_B(y) \quad (1.1)$$

where $\mu_{R_1} = \mu_{A \rightarrow B}(x, y)$ is the *truth value* for the statement \mathbf{R}_1 and μ_A and μ_B are the *levels of membership (membership values)* of the variables x and y in fuzzy sets A and B respectively.

There are many ways in which the fuzzy implication may be defined. Zadeh suggested the *compositional rule of inference* [ZAD73]. Nearly 40 distinct fuzzy implication functions have been described in the literature [LEE90, ZIM90].

Takagi and Sugeno [TAK83] proposed a form which has fuzzy sets only in the premise part of the rule, and the consequent part is described by a non-fuzzy equation of the input variable:

$$\mathbf{R}_2: \text{If } x \text{ is } A, \text{ then } y \text{ is } k_1x^2 + k_2x + k_3.$$

Like in Zadeh's rule, A is the linguistic label. k_1, k_2, k_3 are predefined constants.

The generalization of a control rule which has two conditions in the antecedent part has the following form:

$$\mathbf{R}_3: \text{If } x \text{ is } A \text{ and } y \text{ is } B, \text{ then } z \text{ is } C.$$

Following the previous definition (eq. 1.1) we can conclude that the truth value of this statement is:

$$\mu_{R_3} = \mu_{A \text{ and } B \rightarrow C}(x, y, z) = [\mu_A(x) \text{ and } \mu_B(y)] \rightarrow \mu_C(z) \quad (1.2)$$

where A, B and C are fuzzy sets defined on the universes of discourse for x, y and z . μ_A, μ_B and μ_C are the levels of membership of the variables x, y and z in fuzzy sets A, B, C respectively.

For the fuzzy intersection (*and* operator) Zadeh [ZAD65] suggested the *min* operator. Generally, intersection operators are called *T-norms* and union operators are called *T-conorms*.

2.3 Reasoning based on fuzzy logic vs. reasoning based on Boolean logic

In order to present the idea of fuzzy control to the reader, we will give a very simple toy problem from the field of mobile robotics.

Suppose that the goal of our experiment is to control a mobile robot shown in Figure 3 using a trivial algorithm. The robot is equipped with a single sensor

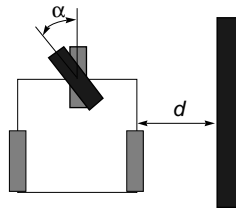


Figure 3 Mobile robot

which gives information about the distance between the robot and the obstacle. The first step in the design of a controller is to define its inputs and outputs i.e. linguistic variables. In this example, we consider a one input/one output controller: the input is a distance between the robot and the obstacle (d) and the output is the steering angle of the robot's wheel (α).

If we apply the Boolean logic approach, we can establish the following linguistic rules:

- **If** the distance between the robot and the obstacle is less than 10 cm, **then** steer for $+10^\circ$.
- **If** the distance between the robot and the obstacle is more than 10 cm, **then** steer for -10° .
- **If** the distance between the robot and the obstacle is 10 cm, **then** go straightforward.

These Boolean rules are intended to result in a wall following behaviour. The universes of discourse for these two linguistic variables are shown in Figure 4. Since we are in the binary domain, membership functions are crisp. This algorithm is quite primitive and the robot will unavoidably oscillate.

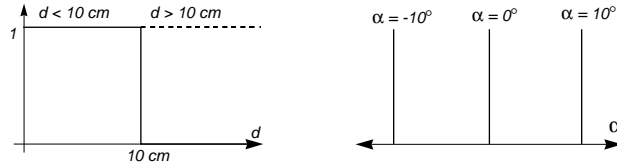


Figure 4 Universe of discourse for the distance and the steering angle

For the fuzzy controller, we define the following linguistic rules:

- **If** the distance between the robot and the obstacle is less than 10 cm, **then** turn to the left (α is negative).
- **If** the distance between the robot and the obstacle is more than 10 cm, **then** turn to the right (α is positive).
- **If** the distance between the robot and the obstacle is nearly 10 cm, **then** keep the direction (α is around 0).

In fact, with these rules, we defined three linguistic values for the linguistic variable distance: More than 10 cm, Nearly 10 cm (optimal distance) and Less than 10 cm. For the steering angle, we have chosen three linguistic values: Negative for negative angles, Zero for going straightforward and Positive for positive angles.

With this controller, we expect to have a more “intelligent” behaviour. The universes of discourse for the distance and the steering angle are shown in Figure 5. The chosen membership functions have triangular and monotonous

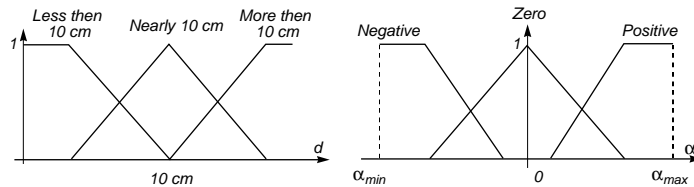


Figure 5 Universes of discourse for the distance and the steering angle

shapes. We can calculate the control value by applying the fuzzy implication we mentioned in Section 2.2.

Suppose the sensor gave 5 cm for the distance. This distance belongs to the fuzzy set Less than 10 cm with a degree of truth 0.7, or 70% (Figure 6), and

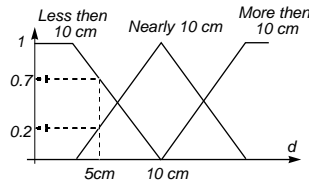


Figure 6 Fuzzification

we can say that the truth value of the statement It is true that the distance is less than 10 cm is 0.7. At the same time, it is true but only 0.2, or 20%, that this distance is Nearly 10 cm.

We apply the rules. The antecedent part of the first rule is satisfied but not 100%. It is true only 70%. This value is the firing strength of this rule. It means that the steering angle has to be Negative 70%. For the second rule, we will proceed in an analogous way. The distance is Nearly 10 cm, but only 20%. The robot has to keep the direction with this degree of truth. The premise part of the third rule is not satisfied so the rule is not used.

We can conclude that the robot has to steer 70% to the left and 20% to go straightforward. It means that we have to “modulate” the appropriate fuzzy sets with these two values. The methods mostly used are: *linear modulation* (Figure 7a) and *modulation by clipping* (Figure 7b).

There are several methods to find a crisp value of the command. Figures 7a and 7b illustrate two of them: *centre of gravity* and *mean of maxima*.

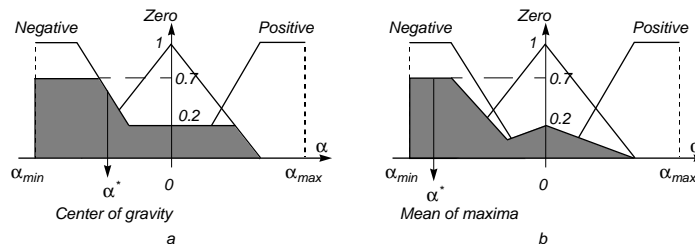


Figure 7 Fuzzy inference and defuzzification

Although trivial, this example is introduced to give a general idea of fuzzy reasoning and the design of a fuzzy controller.

2.4 Procedure for fuzzy reasoning

From the previous example, we summarize the steps in fuzzy reasoning:

1. **Fuzzification**: to each measure of an input variable is attributed a membership value for all the fuzzy sets defined over the universe of discourse of that variable.
2. **Application of the T-norm** (usually this operator is *min* or product) on the membership values of the premise part of the rules to get *firing strength* or the *weight* for each rule ($\mu_A(x)$ in eq. (1.1) or $[\mu_A(x) \text{ and } \mu_B(y)]$ in eq. (1.2)).¹
3. **Generation of the consequent value** ($\mu_B(y)$ in eq. (1.1) or $\mu_C(z)$ in eq. (1.2)) of each rule. It can be crisp or fuzzy.
4. **Defuzzification**: generate the crisp output values.

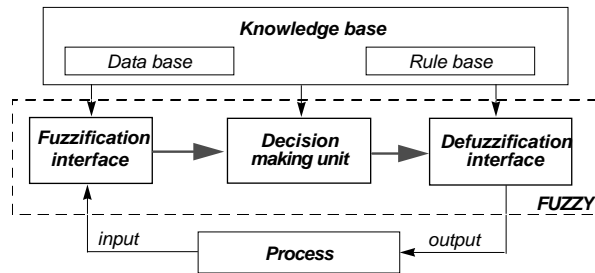


Figure 8 General structure of fuzzy inference system

This means that every fuzzy controller is composed of four principal blocks (Figure 8):

1. **Knowledge base** (parameters of membership functions and definitions of rules)

¹In the previous example we had only one input and we didn't need to apply the T-norm.

2. **Fuzzification interface** (transformation of crisp inputs into membership values)
3. **Decision making unit** (inference operations on the rules)
4. **Defuzzification interface** (transformation of the fuzzy result of the inference into a crisp output)

2.5 Advantages and disadvantages of fuzzy control

Standard controllers deal with systems that have a precisely defined mathematical model. In the design of a fuzzy controller one does not need to know a precise mathematical model of the plant to be controlled. If the model however exists, it can be used for the simulation and for the test of the control strategy. The main advantages of fuzzy controllers are:

- No need to have a mathematical model of the system.
- It is possible to implement expert human knowledge and experience using comprehensible linguistic rules.
- It is possible to control non-linear plants.
- Thanks to dedicated processors, it is possible to control fast processes.

One of the results of the fuzzy “boom” in 90’s was the general opinion that the fuzzy approach can solve all control problems and that classical control approaches will be soon abandoned. However, it is definitely not possible to solve all control problems using fuzzy logic because there are some very important drawbacks:

- There is no standard and systematic method for the transformation of the human knowledge or experience into the rule base of a fuzzy inference system, no general procedure for choosing the optimal number of rules, since a large number of factors are involved in the decision, e.g. performance of the controller, efficiency of computation, human operator behaviour, the choice of linguistic variables etc.
- Even when human operators exist, their knowledge is often incomplete and episodic, rather than systematic.

- It is not possible to show the stability of the controlled system, since the model is not known.
- It is not guaranteed that rules are coherent. It is possible to have a mismatch between the rules.
- Computing time could be long, because of the complex operations such as fuzzification and particularly defuzzification.

As a result, many researchers have been trying to automate the modelling process of fuzzy systems [SUN91]. The task can be divided in two parts:

- Structure identification of the system (related to finding a suitable number of rules and a proper partition of the feature space).
- Parameter identification (adjustment of the membership functions).

These two reasons have led to the idea of merging the neural network approach and fuzzy technology.

3 KHEPERA MOBILE ROBOT

We think that the only way to check out a control algorithm for a mobile robot is to apply it to the real robot. Simulation is a very nice approach, but the problem is that simulation is often not an approach that represents veraciously the processes from the real world.

Our work is based on experiments with a miniature robot named Khepera (Figure 9) 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. Its small size allows experiments to be performed in a small work area (Figure 10). In this configuration, Khepera is remote controlled by a work station through a serial link.

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, visualization software tools, etc.). The microcontroller includes all the

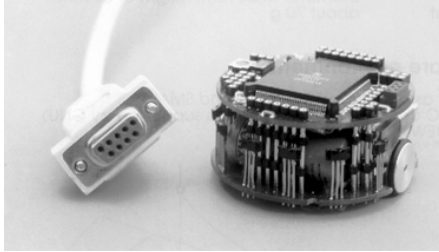


Figure 9 Khepera mobile robot

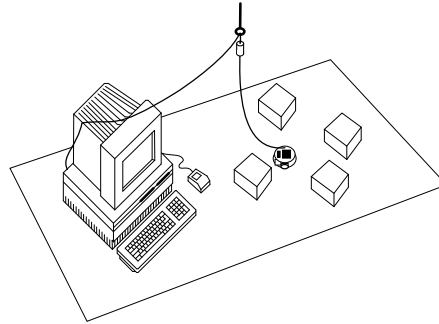


Figure 10 Khepera's working environment

features needed for easy interfacing with memories, with I/O ports and with external interruptions.

The sensory/motor board includes two DC motors coupled with incremental sensors, eight analogue infra-red (IR) proximity sensors ($S_0 \dots S_7$ on Figure 11) and on-board power supply. IR sensors are composed of an emitter and of an

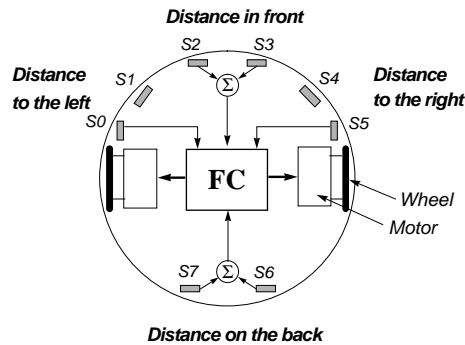


Figure 11 Position of the sensors on the Khepera mobile robot

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 sensors? Apply fuzzy logic? 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 [GAU93, ZRE94], artificial intelligence approaches [BAS93]. In next section, we will describe a very simple fuzzy controller for this robot.

3.1 A simple fuzzy controller for obstacle avoidance

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!

The aim of our first "fuzzy" experiment was to design a simple controller for Khepera with a minimal number of the linguistic rules. The simple fuzzy controller could be designed in the following way:

Suppose that we take all the available informations. The controller would have 8 inputs and 2 outputs. The inputs are the distances between the robot and the obstacles and the outputs are the motor speeds. For every input we can define 2 membership functions (Small distance, Big distance) and for every output 3 membership functions (Backward, Stop, Forward). It means that there are 2304 ($2^8 * 3^2$) possible rules and a complete fuzzy engine needs 256 (2^8) rules! It is obvious that the design of such a controller without a precise methodology is not possible. In its absence, one is forced to reduce the number of inputs (the number of outputs cannot be reduced) and simplify the fuzzy inference engine.

We have chosen to deal with four inputs: distance to the right, to the left, in front and on the back of the robot (Figure 11). The structure of the implemented controller is shown in Figure 12. For each input we defined three membership functions, therefore classifying the distances as: Big, Average and Small. Each motor speed is represented by seven linguistic variables: Backward Fast, Backward Medium, Backward Slow, Stop, Forward Slow, Forward Medium and Forward Fast.

The linguistic rules can be defined in an almost intuitive way. The first very necessary rule is: *If* there are no obstacles around a robot, *then* it has to go

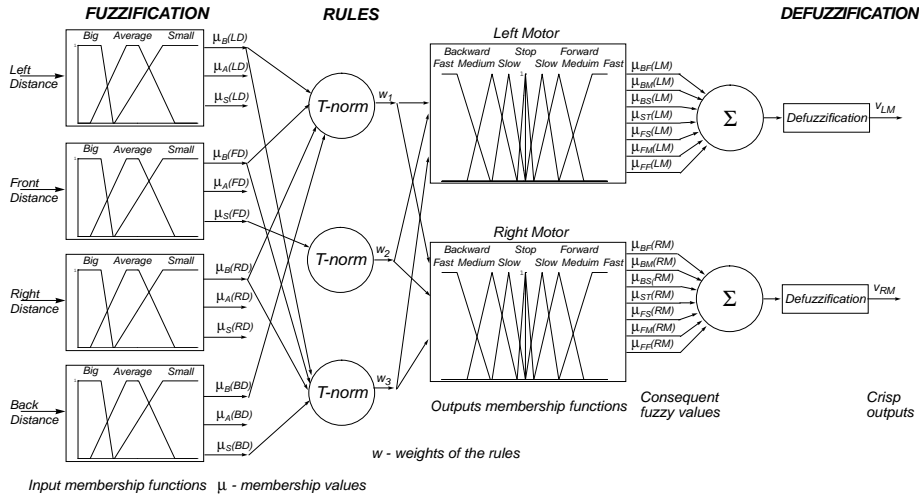


Figure 12 Structure of a fuzzy controller with 3 rules

straightforward. The second might be for example: **If** there is an obstacle in front of the robot, **then** it has turn to the left.

The minimum number of necessary rules is three but of course, the obtained behaviour is very primitive. These rules are represented in Table 1 and can be read in the following way:

Rule 1: **If** the distance to the left is Big **and** the distance in front is Big **and** the distance to the right is Big **and** the distance on the back is Big **then** left motor speed is Forward Medium and right motor speed is Forward Medium.

	Distances				Motor speeds	
	Left	Front	Right	Back	Left Motor	Right Motor
Rule 1:	Big	Big	Big	Big	Forward Medium	Forward Medium
Rule 2:		Small			Forward Medium	Backward Medium
Rule 3:	Big	Big	Big	Small	Forward Fast	Forward Fast

Table 1 Linguistic rules for the fuzzy controller

Rule 2: *If* the distance in front is Small (other distances are not considered) *then* left motor speed is Forward Medium and right motor speed is Backward Medium.

Rule 3: *If* the distance to the left is Big *and* distance in front is Big *and* distance to the right is Big *and* distance on the back is Small *then* left motor speed is Forward Fast and right motor speed is Forward Fast.

The robot avoids obstacles turning always in the same way. It turns while the back sensors do not “see” the obstacle. This algorithm allows the robot to move without a specific goal in a simple environment. It means that it cannot get out of a dead end for example.

The next step was the building of a more complicated controller by adding linguistic rules. With 16 rules, Khepera avoids obstacles very successfully turning to the left and to the right. Unfortunately, tuning by hand is very difficult when the controller has more than 16 or 20 rules because of possible mismatching of the rules. An automatic method for the choice of the rules and the parameters is necessary.

The critical problem of this algorithm 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 algorithm.

We made some experiments with Takagi and Sugeno’s method [TAK83] where output membership functions were defined as singletons (see Section 2.2). The behaviour of the robot was not changed in comparison with the previously described method. The only difference was the computing time which decreased significantly. The centre of gravity method takes around 70% of computational time of the fuzzy controller.

This approach allows a possibility to realise very interesting behaviours of the robot by simply changing linguistic rules. The robot can follow the edge of a wall, approach (or run away from) a light source and so on.

4 NEURAL NETWORKS

The aim of neural networks is to model networks of biological neurons in the brain. From the neuroscientist's point of view, these models are extremely simplified. However, we use them because we hope that they give some insight into principles of biological "computation".

Artificial neural network models have many names in the literature like: connectionist models, parallel distributed processing models, neuromorphic systems, associative networks, etc. Their structure is based on our present understanding of biological nervous systems. In fact, they are parallel structures composed of many computational elements connected by links with variable weights.

4.1 Biological neuron

There are about 10^{11} neurons of different types in a brain. The main parts of a neuron are: *cell body* or *soma*, *dendrites* and *axon* (Figure 13). The cell

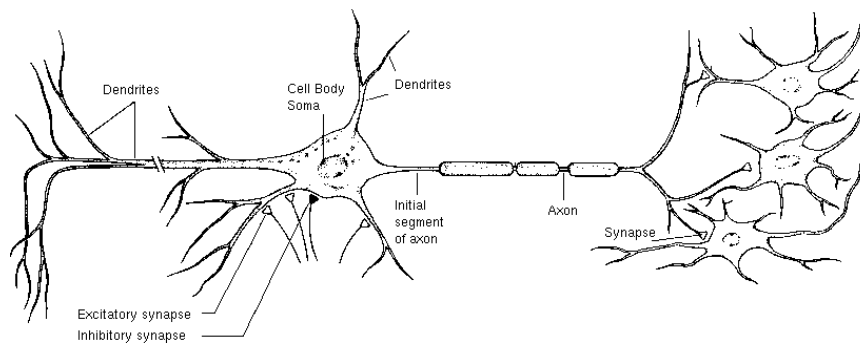


Figure 13 Biological neuron

nucleus is located in the soma, while dendrites are fibres connected to it. The axon is one long fibre that is the extension of the cell body. It branches into strands and substrands. At the end of these are *synapses* to other neurons. One axon makes typically a few thousand of *excitatory* or *inhibitory* synapses with other neurons.

The process of transmitting a signal from one neuron to another is chemically complex and is beyond the scope of this paper. Shortly, the receiving cell's *electrical potential* raises or lowers depending on the incoming signal. When

this potential reaches a *threshold*, an *action potential* of fixed strength and duration is sent down the axon. The neuron “fires” and the action potential is transmitted through the axon to the synapses with other cells. After a *refractory period*, the cell can fire again [HER91].

4.2 Model of artificial neuron

The first model of an artificial neuron was proposed in 1943 by McCulloch and Pitts [McC43]. Although it is a very simple computation unit, it is the basic element of connectionist neural networks.

In the neuron in Figure 14, x_j ($j = 1, \dots, n$) is the *input stimuli* that can be

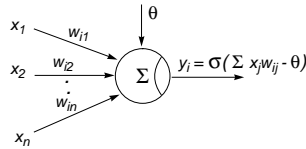


Figure 14 Model of artificial neuron

either 1 or 0 representing the state of neuron j as firing or nonfiring respectively. The *synaptic weight* w_{ij} can be excitatory or inhibitory and represents the strength of synapse that connects the neuron j to the neuron i . θ is a *threshold*. The neuron potential is:

$$p_i = \sum_{j=1}^n x_j w_{ij}$$

This neuron sums n weighted inputs and passes the result through a nonlinearity (or activation function). The node is characterized by an internal threshold or offset θ and by the type of the nonlinearity. Figure 15 illustrates some common types of nonlinearities. The most employed is a sigmoid function because it is considered to be very close to the input/output function of a real neuron [BLA90].

It is obvious that this model does not imitate the real neuron. Why then all this excitement about neural networks in the last 40 years? The most important reason is their ability to learn and to generalise to new situations. Once a neural network has been trained for a set of data, it can interpolate and produce answers for the cases not present in the training set.

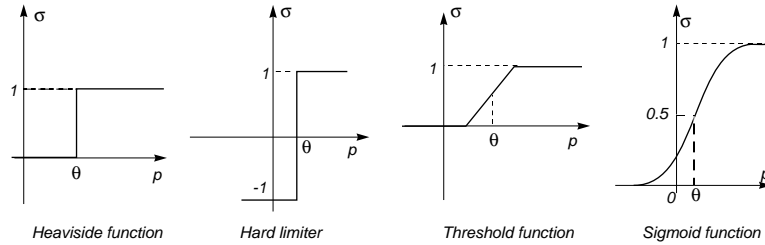


Figure 15 Nonlinearities in the model of artificial neuron

Neural networks are specified by their *topology* or *connection matrix*, *node characteristics* and *training* or *learning rules*. These rules specify an initial set of weights and indicate how weights should be adapted during learning to improve performance. This can be done in two ways:

- **Supervised learning.** The weights are adjusted on the basis of comparison of the output of the network with the desired (or correct) answer. It means that we “teach” the network to perform a desired computation.
- **Unsupervised learning.** This method is used when the learning goal is not defined at all in terms of specific correct examples. The only available information is in the correlation of the input data. The network has to create categories from these correlations and produce output signals corresponding to the input category [HER91].

4.3 Application to the Khepera mobile robot

A very simple neural network has been implemented on Khepera. It is based on Braitenberg’s idea to directly connect sensors with actuators [BRA84] as shown in Figure 16. Every connection has a positive (excitatory) or negative (inhibitory) weight. The vehicle can perform different kinds of behaviours depending on the value of the weight and the kind of sensors. It can be attracted or repulsed from the light source, it can follow another vehicle, avoid obstacles and so on.

To allow a comparison with the fuzzy controller applied to the same robot, we will consider the problem of obstacle avoidance [MON93]. A learning procedure has not been applied because of the simplicity of the task. Tuning of weights

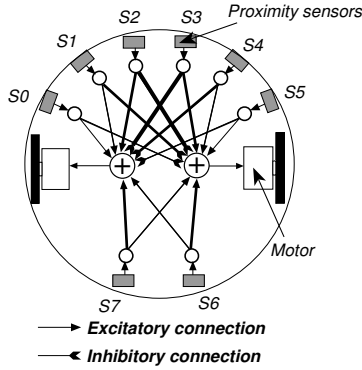


Figure 16 Braitenberg vehicle

has been done by hand. The only problem was to find the proper weights for all connections.

If one examines the matrix shown in Table 2 and even if the principle of the

Sensor:	S0	S1	S2	S3	S4	S5	S6	S7	Threshold
Motor 1	0.2	0.2	0.3	-0.72	-0.46	-0.2	0.26	0.15	0.30
Motor 2	-0.26	-0.51	-0.77	0.31	0.2	0.2	0.15	0.26	0.30

Table 2 Matrix of weights for Braitenberg's vehicle

algorithm is known, it is not possible to define which behaviour the robot will perform. From this point of view the fuzzy controller is superior because the behaviour is defined by a set of linguistic easily comprehensible rules. However the computational time for neural networks is shorter than that for fuzzy controllers.

4.4 Similarities between neural networks and fuzzy systems

Neural network technology and fuzzy theory were developed at the same time. Their similarities are often discussed in the literature [Kos92, Tak90]. So far, the main direction of the research has been on automatic design and fine-tuning

of the membership functions used in fuzzy control through learning by neural networks.

Neural networks and fuzzy systems are highly parametric parallel structures. In neural nets, both knowledge extraction and knowledge representation are difficult. On the other hand, the advantage of fuzzy systems is that knowledge is represented in the form of the comprehensive linguistic rules.

Fuzzy systems are able to treat the uncertain and imprecise informations. The weak points of fuzzy controllers are caused mainly to the difficulty of defining accurate membership functions and lack of the systematic procedure for the transformation of the expert knowledge into the rule base. The shape of the membership function in fuzzy systems and the threshold function in neural networks are similar. *Multiply-add* operation of artificial neurons is very close to *max-min* operation of approximate reasoning.

These reasons lead to the idea of merging these two approaches. There are two possible methods: a method by which individual merits are combined, and another by which analogies between these are overlapped. One possibility is to endow learning functions to fuzzy logic systems, or to conduct pattern processing before fuzzy logic is applied. The other is to incorporate fuzzy logic and linguistic rules into the structure of neural networks.

5 FUZZY NEURONS

There are a lot of *fuzzy neurons* proposed in the literature. Also, some learning and adaptation mechanisms for the proposed neurons are given [GOD94, GUP91, JAN92]. We will focus on only three of them.

5.1 Basic structure of fuzzy neuron

The theoretical structure of fuzzy neuron is shown in Figure 17. Since it has to be able to cope with fuzzy informations [GUP91], the inputs are fuzzy sets x_1, \dots, x_n on the universes of discourse U_1, \dots, U_n , respectively. These fuzzy sets can be labelled by linguistic terms like: warm, high, large. . . First, they are weighted, but in a different way than in neural networks. After, the inputs are aggregated by the fuzzy operation which is, in general the fuzzy implication function.

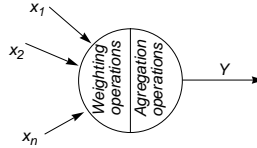


Figure 17 General model of fuzzy neuron

5.2 Models of fuzzy neurons

The principal motivation for building fuzzy neurons is to have models that behave in much the same way as biological neurons [GUP91]. Even if the motivation is the same as in the field of neural networks, we do not agree that these simplified models imitate biological neurons because natural structure is much more complicated than the models presented below.

The first model is an attempt to design the linguistic if-then rule. The experi-

<p>Model 1 Model of the linguistic if-then rule</p>		<p>Rule: If X_{1i} and X_{2i} and ... and X_{ni} then Y_i</p> $Y_i = X_{1i} \circ X_{2i} \circ \dots \circ X_{ni}$
<p>Model 2 Fuzzy extension of the non-fuzzy neuron</p>		<p>Fuzzification</p> $\mu(x_1, x_2, \dots, x_n) = \mu_1(x_1) \otimes \mu_2(x_2) \otimes \dots \otimes \mu_n(x_n)$
<p>Model 3 Extension of the non-fuzzy neuron</p>		<p>Fuzzy inference</p> $Y = X'_1 \otimes X'_2 \otimes \dots \otimes X'_n$ $X'_i = G_i(X_i), i = 1, 2, \dots, n$

Table 3 Models of fuzzy neurons

ence of the neuron is stored in the fuzzy operator \circ which is usually a T-norm or a T-conorm [GUP91]. It is composed from the current inputs and the past experiences. There are currently few, if any, learning methods proposed in the literature.

The second model is the fuzzy extension of the non-fuzzy neuron. It has n fuzzy or non-fuzzy inputs and the weighting operations are replaced by membership functions. These operations are *synaptic*. The results are membership values of the corresponding input in the fuzzy set. These values are then aggregated and the aggregation operation is called *somatic*. It means that the output, considered as a level of confidence or degree of truth, is a crisp value from the interval $[0,1]$. The aggregating operation \otimes may be one of the operators such as *min* or *max*. Learning method for this kind of neuron is proposed in [GOD94].

The third fuzzy neuron is also an extension of a non-fuzzy neuron. It has fuzzy inputs X_1, \dots, X_n . Each input undergoes a synaptic operation which results in another fuzzy set. All modified fuzzy sets X'_1, \dots, X'_n are aggregated to produce another n -dimensional fuzzy set Y . The operator \otimes may be one of the T-norms or T-conorms. Similarly to the first model, learning method for this one is not proposed in the literature.

5.3 Learning methods

The aim of a learning method is to train a system to perform the desired computation by iterative adjustment of the synaptic weights. The proposed adaptation mechanisms for neurons described above can be *synaptic* or *somatic* [GOD93]. They are based on the supervised neural network learning scheme.

Synaptic adaptation means that all the inputs are constantly modified by synaptic operations and then forwarded to the neuron body. These operations are simple in the non-fuzzy case (multiplication), but in fuzzy neurons they are very complex. If the membership functions and inputs are triangular, the proposed modification are shown in Figure 18. The dotted lines represent the fuzzy inputs before modification, and the solid ones after modification.

Somatic adaptation implies that the modifications are applied to the structure of the fuzzy neuron. These modifications include: changing the rule base, changing the membership functions assigned to the fuzzy terms in every rule and changing the aggregation functions.

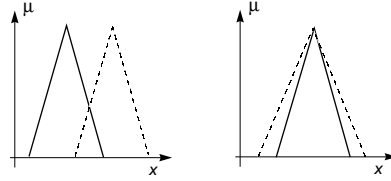


Figure 18 Changes of membership functions during learning phase

These types of adaptation can be applied in supervised learning procedures with well known methods like the gradient descent algorithm.

6 NEURO-FUZZY CONTROLLER

Usually, neuro-fuzzy controllers are fuzzy inference systems implemented under the framework of neural networks. The idea of the design of neuro-fuzzy system is to exploit the advantages of fuzzy logic, as an important branch of fuzzy set theory that emulates human reasoning process, and a neural network's ability to learn.

Learning methods for fuzzy controllers are usually the application of conventional adaptive control techniques. Since these techniques are used in neural networks, as for example the least square algorithm, adaptive fuzzy controllers are named neuro-fuzzy networks or neuro-fuzzy controllers. One of the interesting architectures for neuro-fuzzy controller has been proposed by Jang [JAN92].

In this section we will describe one learning method for a fuzzy controller [NOM92]. The starting point is the Takagi-Sugeno method where output membership functions are defined as singletons.

6.1 Linguistic rules

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

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 , $i = 1, \dots, n$*

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.

6.2 Membership functions

Every membership function in this controller is defined as symmetric and triangular and has to be partially differentiable (Figure 19). The definition of the

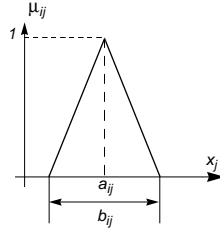


Figure 19 Membership function

membership values is given by the eq. (1.3). For every input the universe of discourse is defined. We use 5 fuzzy sets as uniformly distributed triangular membership functions.

$$\mu_{ij} = \begin{cases} 1 - \frac{2|x_j - a_{ij}|}{b_{ij}} & , \text{ for } a_{ij} - \frac{b_{ij}}{2} < x_j < a_{ij} + \frac{b_{ij}}{2} \\ 0 & , \text{ otherwise} \end{cases} \quad (1.3)$$

6.3 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:

$$u_i = \mu_{i1}\mu_{i2} \cdots \mu_{im} \quad (1.4)$$

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

$$y = \frac{\sum_{i=1}^n u_i w_i}{\sum_{i=1}^n u_i} = \frac{u_1}{\sum_{i=1}^n u_i} w_1 + \frac{u_2}{\sum_{i=1}^n u_i} w_2 + \cdots + \frac{u_n}{\sum_{i=1}^n u_i} w_n = \sum_{i=1}^n \bar{u}_i w_i \quad (1.5)$$

A general scheme of this system with two inputs and one output is represented in Figure 20. This presentation is analogous to the architecture of a feed-

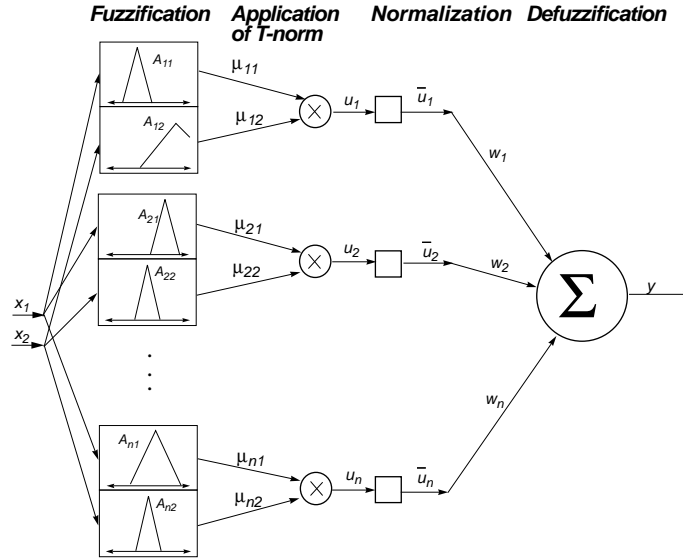


Figure 20 Neuro-fuzzy controller

forward artificial neural network. The first layer performs a fuzzification for each linguistic rule and it is a layer of neurons of model 1 (Table 3). Outputs from this layer are membership values (eq. 1.3) and they are fed into the next layer which performs a T-norm operation –product– (eq. 1.4). The result is a firing strength for each rule and in the next layer, firing strengths are normalized (eq. 1.5). The last layer computes the overall output as the weighted sum of the incoming signals.

6.4 Computational aspects

Suppose that the parametric model of one non-linear 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. The learning problem is a parameter estimation problem. All solutions to parameter estimation problems consist in finding the extremum of *criterion (loss) function* V considered a function of the parameters of the unknown system. The function to minimize is:

$$V(z) = \frac{1}{2} (y(t) - y_d(t))^2 \quad (1.6)$$

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. 1.6) and apply the *method of stochastic approximation* to identify the parameters of the fuzzy system. It is an iterative procedure given by:

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

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 :

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

6.5 The learning algorithm

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

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

The number of parameters to estimate is $p = 2nm + n$. The vector which minimizes the loss function is given by:

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

and the learning rule:

$$z_k(t+1) = z_k(t) - \Gamma \frac{\partial V(z)}{\partial z_k}, \quad k = 1, \dots, p \quad (1.7)$$

It follows from (eq. 1.3 - 1.7) that the equations for the adaptation of the parameters of fuzzy system are:

$$a_{ij}(t+1) = a_{ij}(t) - \Gamma_a \frac{u_i}{\sum_{l=1}^n u_l} (y - y_d) (w_i - y) \operatorname{sgn}(x_j - a_{ij}(t)) \frac{2}{b_{ij} \mu_{ij}(x_j)}$$

$$b_{ij}(t+1) = b_{ij}(t) - \Gamma_b \frac{u_i}{\sum_{l=1}^n u_l} (y - y_d) (w_i - y) \frac{1 - \mu_{ij}(x_j)}{b_{ij} \mu_{ij}(x_j)}$$

$$w_i(t+1) = w_i(t) - \Gamma_w \frac{u_i}{\sum_{l=1}^n u_l} (y - y_d)$$

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

1. Initialisation 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 $(x_1, x_2, \dots, x_m, y_d)$.
3. Fuzzy inference.
4. Adaptation of consequent parameters w_i .
5. Repeat once the step 3 with new values for w_i and step 4.
6. Adaptation of parameters a_{ij} and b_{ij} .
7. Evaluation of the criterion function V .
8. Repeat the steps 3 to 7 until V is smaller than one threshold value.

This method is very similar to the method of gradient descent for the Perceptron [RUM89].

6.6 Application of the neuro-fuzzy controller to the Khepera

We applied this algorithm to the obstacle avoidance problem of the Khepera mobile robot [GOD95]. The goal was to adjust the parameters of membership functions using a supervised learning procedure. 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.

This method is very interesting because it has the ability to express the knowledge acquired from input-output data in the form of linguistic rules.

7 CONCLUSION

In this paper some basic notions of fuzzy controllers and neural networks have been presented. The application of these methods is described in the case of the obstacle avoidance behaviour. As an experimental platform we used the Khepera mobile robot. It is shown that a simple fuzzy controller can be applied for a realization of very simple tasks. Even if the number of rules is small, the robot shows interesting behaviour.

The same problem is treated in the frame of a neural network based approach. We presented a simple neural net which enabled the robot to have different kinds of behaviour. The problem is that the extraction of knowledge from the neural network is impossible.

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 led to the idea of supplying fuzzy controllers with learning functions. This was implemented on a controller based on the Takagi-Sugeno method. The learning procedure is the stochastic approximation method known in the field of identification of systems which corresponds to supervised learning of neural networks. The results show that fuzzy systems have an ability to learn and that the methods used in the neural network field can be applied. The method is tested on Khepera and the experiments show that the robot can learn by adjusting the parameters of the controller.

Acknowledgements

The author would like to thank André Guignard, Edo Franzi and Francesco Mondada for the design of Khepera and for the help in testing the algorithms presented in this paper. She also gratefully acknowledge the constructive criticism on the manuscript and the constant support of Jean-Daniel Nicoud, Paolo Ienne, Pero Subašić, Laurent Tettoni and Andrea Gees.

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.
- [BLA90] François Blayo. *Une implantation systolique des algorithmes connexionnistes*. PhD Thesis N° 904, École Polytechnique Fédérale de Lausanne, Lausanne, 1990.
- [BRA84] Valentino Braitenberg. *Vehicles*. MIT Press, 1984.
- [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.
- [GAU93] Ph. Gaussier and S. Zrehen. Emergence of behaviors on a mobile robot: Learning with neural networks. In *Learning days in Jerusalem*, Jerusalem, 1993.
- [GOD93] Jelena Godjevac. State of the art in the neuro fuzzy field. Technical Report 93/25, École Polytechnique Fédérale de Lausanne - DI, April 1993.
- [GOD94] Jelena Godjevac. Comparison between classical and fuzzy neurons. In *EUFIT*, volume 3, pages 1326–1330, Aachen, Germany, September 1994.
- [GOD95] Jelena Godjevac. A learning procedure for a fuzzy system: application to obstacle avoidance. to be published ISFL '95, Zurich, Switzerland, 1995.

- [GUP91] M. M. Gupta and J. Qi. On fuzzy neuron models. In *Int. Joint Conf. on Neural Networks, IEEE+INNS*, volume 2, pages 431 – 435, Seattle, July 1991.
- [HER91] John Hertz, Anders Krogh, and Richard G. Palmer. *Introduction to the Theory of Neural Computation*. Santa Fe Institute Studies in Sciences of Complexity. Addison-Wesley, Redwood City, Calif., 1991.
- [JAN92] J.-S. Roger Jang. Anfis, adaptive-network-based fuzzy inference systems. *IEEE Trans. on systems, Man and Cybernetics*, 1992.
- [KIC78] W. J. M. Kickert and E. H. Mamdani. Analysis of a fuzzy logic controller. In *Fuzzy Sets and Systems 1*, pages 29–44. North Holland Publishing Company, 1978.
- [KOS92] Bart Kosko. *Neural networks and fuzzy systems*. Prentice-Hall, 1992.
- [LEE90] C. C. Lee. Fuzzy logic in control systems: fuzzy logic controller - part 1 and part 2. *IEEE Trans. on systems, Man and Cybernetics*, 20 (2):404–435, 1990.
- [LI88] Y. F. Li and C. C. Lau. Development of fuzzy algorithms for servo systems. In *IEEE International Conference on Robotics and Automation*, Philadelphia, Pennsylvania, April 1988.
- [MAM75] E. H. Mamdani and S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *Int. J. Man-Machine Studies*, 7:1–13, 1975.
- [MCC43] McCulloch and Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematica Biophysics*, 5:115–133, 1943.
- [MON93] Mondada F. and E. Franzi. Biologically inspired mobile robot control algorithms. In *NFP-PNR 23 Symposium*, Zurich, Switzerland, October 1993.
- [MON94] Francesco Mondada, Edoardo Franzi, and Paolo 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.

- [RUM89] David E. Rumelhart and James L. McClelland. Learning internal representations by error propagation. In *Parallel distributed processing*. MIT Press, Cambridge, Mass., 1989.
- [SUN91] C.-T. Sun and J.-S. Jang. Fuzzy modeling based on generalized neural networks and fuzzy clustering objective functions. In *Proc. of the 30th Conference on Decision and Control*, pages 2924–2929, December 1991.
- [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.
- [TAK90] H. Takagi. Fusion technology of fuzzy theory and neural networks, survey and future directions. In *Proc. of the International Conf. on Fuzzy Logic & Neural Networks*, pages 13–26, Iizuka, Japan, July 1990.
- [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.
- [ZAD65] Lotfi Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.
- [ZAD73] Lotfi Zadeh. Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Trans. on systems, Man and Cybernetics*, 3(1):28–44, January 1973.
- [ZAD94] Lotfi Zadeh. Fuzzy logic, neural networks and soft computing. *Fuzzy Systems, Communications of the ACM*, 37(3):77–84, March 1994.
- [ZIM90] H.-J. Zimmermann. *Fuzzy Sets Theory - and Its Applications*. Kluwer Academic Publishers, 1990.
- [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.