Fifth IFSA World Congress (1993), 877-880

FUZZY CONTROL: DESIGNING VIA FUZZY MODELLING

Kaoru Hirota* Witold Pedrycz**

*Dept. of Instrument and Control Engineering
College of Engineering, Hosei University , Koganei-city, Tokyo 184, Japan

**Dept. of Electrical & Computer Engineering,University of Manitoba,
Winnipeg, Manitoba, Canada, R3T 2N2, pedrycz@eeserv.ee.umanitoba.ca

Abstract Fuzzy control algorithms are developed based on fuzzy models of systems. The control issues are posed as multiobjective optimization problems involving goals and constraints imposed on system's variables. Two basic design modes embrace on- and off-line control development. The first type of design deals with the time and state-dependent objectives and pertains to control determination based upon the current state of the system. The second design mode gives rise to explicit forms of fuzzy controller that is learned based on a given list of state-control associations. Both the fuzzy models as well as fuzzy controllers are realized as logic processors.

1. Introduction

The current methodology of development of fuzzy control algorithms and fuzzy controllers, in particular, is essentially experiment-driven. This error-and-trial attitude that is so dominant nowadays in various applications, is motivated by the two main reasons. On one hand, one can refer to a remarkable easiness with which a designer can modify the fuzzy control algorithm and experiment with it through a sequence of well-thought interactions with the environment. On the other side, one can encounter a lack of a suitable modelling platform that could make the overall design process more systematic and coherent. The primary requirement to achieve this goal is to set up an appropriate modelling environment which could preserve a conceptual compatibility and integrity throughout the fuzzy model and the control algorithm. By this form of compatibility we mean that the fuzzy model and the control algorithm should operate at the same level of information granularity, cf. [12] [5] [7] [8]. This implies that these two essential components should be perceived within the same range of precision satisfying a fundamental requirement of information compatibility .

The purpose of this study is to look at fuzzy control and the design of control algorithms from a general perspective of fuzzy system modelling .Firstly, we will briefly study a paradigm of fuzzy modelling. The very nature of this type of modelling as exclusively carried out through manipulation at the level of linguistic labels rather than numerical quantitates will be emphasized. Secondly, we will concentrate on designing control algorithms making a clear distinction between on-line and off-line control procedures. From the implementation point of view, the fuzzy models as well as the control algorithms will be conveniently completed in terms of logic-based neural networks.

2. Fuzzy modelling: logical approximation of mappings between fuzzy partitions.

Fuzzy models are developed at the level of linguistic labels. These labels, called also information granules, are commonly used to describe the spaces of the variables of the system [12] [7]. The level of generality of fuzzy models places them in between a class of models used for pure numerical processing and those AI constructs of qualitative modelling, cf. [10]. We will be concerned with fuzzy models of single input- single-output dynamical systems (the extension dealing with many-input models is fairly straightforward). The fuzzy partitions of the corresponding state and control space will be denoted by $\mathcal{X}$ and $\mathcal{U}$, respectively. Assuming that these spaces involve "n" and "m" linguistic labels we obtain

$$\mathcal{X} = \{X_1, X_2, \ldots, X_n\} \text{ and } \mathcal{U} = \{U_1, U_2, \ldots, U_m\}.$$

The elements of $\mathcal{X}$ and $\mathcal{U}$ should comply with some general requirements of semantic integrity that make them conceptually well sound. In general, they should "cover" the entire space, be unimodal, and sufficiently distinguishable. In particular, the elements of the fuzzy partitions (fuzzy sets) produced within fuzzy clustering satisfy the above requirements. Each datum in the state and control space, as it is viewed at the conceptual level of the fuzzy model, is represented in terms of these labels. This realization gives rise to the internal vector representation provided in the unit hypercubes of the input and state space

($\mathbf{x} \in [0,1]^n$ and $\mathbf{u} \in [0,1]^m$, respectively). This translation can be carried out in many ways; the use of possibility and necessity measures is quite common [11] [1]. More generally, one can look at this conceptual transformation as a sort of nonlinear and nonuniform linguistic data quantization.

The fuzzy model is developed at the level of the linguistic labels with its exclusive goal of expressing rigorously relationships between the elements of $\mathcal{X}$ and $\mathcal{U}$ occurring in successive discrete time moments. From a formal point of view one can anticipate that the fuzzy model (FM) realizes a logic-inclined approximation between the unit hypercubes induced by the fuzzy partitions, cf. [3] [6] [9]. The perspicuous logical nature of this approximation will be reflected by the architecture of the approximation scheme being implemented as a logic processor, (LP) cf. [3] [9]. Let us remind that the logic processor is designed with the aid of AND and OR neurons being structured into a three-layer neural network. These logical neurons are governed by the following expressions,

-OR neuron:

$$y = \mathrm{OR}(\mathbf{x}; \mathbf{w})$$

namely,

$$y = \mathrm{OR}[x_1 \text{ AND } w_1, x_2 \text{ AND } w_2, \ldots, x_n \text{ AND } w_n]$$

where $\mathbf{w} = [w_1, w_2, \ldots, w_n] \in [0,1]^n$ describes a vector of its connections (weights). The standard implementation of the fuzzy set connectives standing there involves triangular norms, namely the OR and AND operators are realized by some s- and t-norms, respectively. This translates the above expression into the form,

$$y = \overset{n}{\underset{i=1}{S}} [x_i \, t \, w_i]$$

-AND neuron:

In the AND neuron, the OR and AND operators are utilized in a reversed order. We obtain

$$y = \mathrm{AND}(\mathbf{x}; \mathbf{w})$$

which expressed in the notation of the triangular norms reads as

$$y = \overset{n}{\underset{i=1}{T}} [x_i \, t \, w_i]$$

The architecture of the logic processor includes a single hidden layer that consists of the AND neurons, followed by the output layer being formed by the OR neurons, cf. [4]. The goal of the learning itself is to approximate the relationship between the corresponding elements of the hypercubes. The learning is usually completed in a supervised mode. The connections of the neurons are adjusted based on the gradient of the specified performance index.

The first-order dynamical model realized by the logic processor will be described as

$$\mathbf{x}(k+1) = \mathrm{LP}(\mathbf{u}(k), \mathbf{x}(k), \mathcal{W}) \qquad (1)$$

where $\mathcal{W}$ is used to embrace all the connections of the logic processor while the vectors $\mathbf{u}(k)$, $\mathbf{x}(k)$, and $\mathbf{x}(k+1)$ summarize the degrees of activation of the linguistic labels in $\mathcal{U}$ and $\mathcal{X}$ observed in successive time instances. The learning schemes utilized for the training of this class of the logic-based neural networks are well-documented; for details the reader is referred to [9].

3. Control in fuzzy models - problem formulation and architecture

In general, one can consider control activity as aiming at a maximal simultaneous satisfaction of control goals and constraints specified for the system. These control objectives, no matter whether they are given precisely or imprecisely, can be uniformly expressed as the elements in the corresponding fuzzy partitions. Let the goal defined in $\mathcal{X}$ be equal to $\mathbf{g}$ while the constraint given in $\mathcal{U}$ be expressed as $\mathbf{c}$. For the fuzzy model (1) the control task reads as the following vector optimization problem

$$\begin{aligned} &\underset{\mathbf{u}(k)}{\mathrm{Max}} \ \mathcal{F}_1(\mathbf{g}, \mathbf{x}(k+1)) \\ &\underset{\mathbf{u}(k)}{\mathrm{Max}} \ \mathcal{F}_2(\mathbf{c}, \mathbf{u}(k)) \end{aligned} \qquad (2)$$

where $\mathcal{F}_1$ and $\mathcal{F}_2$ are predicates describing referential relationships to be satisfied by the objectives. We will denote them by REF(.,.). Not elaborating now on their numerical character, let us stress that these operations may embrace aspects of matching two objects, determining their difference (dissimilarity), expressing a level of dominance or inclusion existing between them, etc. Generally, the objectives $\mathbf{g}$ and $\mathbf{c}$ could be functions of time as well as they may depend upon the current state variable.

The underlying optimization problem of fuzzy control (2) will be studied within the architecture visualized in Fig.1 and composed of several functional blocks:

Referential blocks. These are utilized to generate the degrees of satisfaction of the relationships between the control objectives and the fuzzy sets describing the system. The basic idea realized there is concerned with the referential operations performed on fuzzy sets. The operations are defined pointwise for each coordinate of the unit hypercube.

The list of useful operations includes:

- similarity (equality). The degree of its satisfaction by x and a, x, a $\in$ [0,1] is embodied by considering the equality index defined as [6], EQ(x,a)=a $\equiv$ x

$$a \equiv x = \frac{1}{2}\left[(a \,\varphi\, x) \wedge (x \,\varphi\, a) + (\bar{a} \,\varphi\, \bar{x}) \wedge (\bar{x} \,\varphi\, \bar{a})\right]$$

where $\wedge$ stands for the minimum operation, overbar denotes complement, and φ is used to describe pseudocomplement (implication), $a\varphi x = \sup\{c \in [0,1] \mid atc \le x\}$.

- inclusion. The degree of inclusion of x in a, INCL(x,a), is expressed as

$$\mathrm{INCL}(x,a) = x \,\varphi\, a$$

-difference. The difference is viewed as a feature dual to the property of similarity, namely

$$\mathrm{DIFF}(x,a) = 1 - \mathrm{EQ}(x,a)$$

- dominance. The degree of dominance, DOM(x,a), defined as

$$DOM(x,a) = a \varphi x$$

represents a degree to which x dominates a.

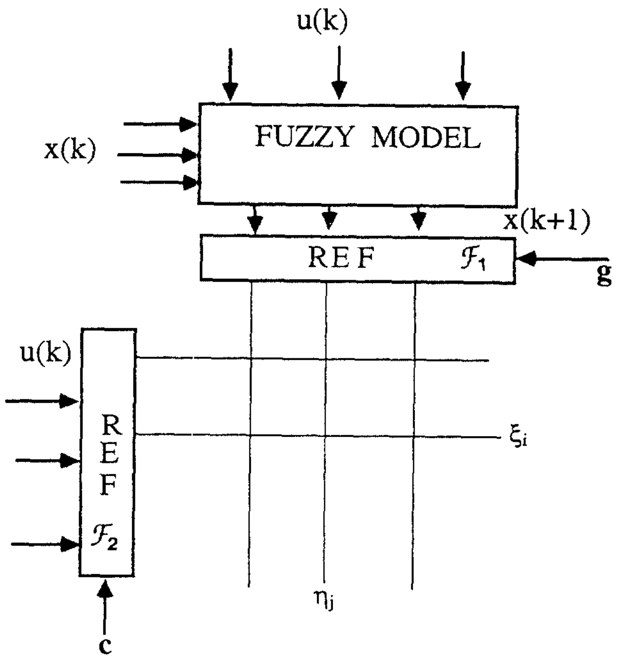

Fig.1. Architecture of model-based fuzzy control

-the fuzzy model (FM) serves as a standard single-step predictor determining outcomes ($\mathbf{x}$(k+1)) produced by $\mathbf{u}$(k) when the system is described by $\mathbf{x}$(k). The functional blocks $\mathcal{F}_1$ and $\mathcal{F}_2$ return the degrees of satisfaction of the objectives, which subsequently are plugged into the aggregation array. At the level of the array, we are looking for the global satisfaction of these objectives achieved at all the elements of $\mathcal{X} \times \mathcal{U}$.

aggregation array. The array as shown in the above figure aggregates the individual levels of satisfaction of the objectives. The value t_{ij} specifies a degree to which the i-th and j-th coordinate of the constraint and the goal are satisfied. This degree is determined by AND-ing the values of the pointwise referential computations produced by $\mathcal{F}_1$ and $\mathcal{F}_2$. The original vector optimization problem (2) is then replaced by its scalar version taking on the form,

$$\operatorname*{Min}_{u(k)} T \tag{3}$$

where

$$T=1-\frac{1}{nm}\sum_{i,j}^{n,m} t_{ij}$$

and $t_{ij}=\eta_i \text{AND} \xi_j$

4. Control determination

In this section we will exploit two conceptually distinct modes of fuzzy control determination such as on-line and off-line control computations.

4.1. On-line computations

The thrust of this form of the computations is to determine control $\mathbf{u}$ such that (3) is minimized for the fixed goal and constraint. The optimization problem is thus specified in the form,

$$\min_{\mathbf{u} \in [0,1]^m} Q$$

The iterative scheme reads as

$$\mathbf{u}(\text{iter}+1)= \mathbf{u}(\text{iter})-\alpha \frac{\partial Q}{\partial \mathbf{u}(\text{iter})}$$

$\alpha \in [0,1]$. Control $\mathbf{u}$ is iteratively adjusted following the gradient of Q. The main advantage of on-line control resides with an evident flexibility in expressing both the goal and constraint sets as being time and state dependent, namely $\mathbf{g}=\mathbf{g}(k,\mathbf{x}(k))$, $\mathbf{c}=\mathbf{c}(k,\mathbf{x}(k))$. Once the goal or constraint are modified, the relevant control has to be recomputed. This, unfortunately, could be associated with a substantial computational burden.

4.2. Off-line control

The leading concept of on-line control is to synthesize a control algorithm through a collection of precomputed associations between some states and control actions. In this sense these associations can be aggregated into an explicit single control architecture (fuzzy controller).

The development of off-line control consists of two phases:
(i) in the first step one acquires associations between some states $\mathbf{x}$(k) and their resulting control. Denote a family of these cases (associations) by $\mathcal{R}$.
(ii) subsequently, the elements of $\mathcal{R}$ are used to learn a closed form of the control algorithm expressed as $\mathbf{u}= \Phi(\mathbf{x})$ where Φ is a mapping between the state and control space.

In the following design we will assume that the goal and constraint are time invariant and they do not depend on the current state of the system. Both the above phases (i)-(ii) can be realized in several ways. Here we will concentrate on one of them originating within the general stream of fuzzy neurocomputations [9].

The associations can embrace either specific fuzzy sets of state and control (that are usually quite convincing from a numerical end of the design) or could be selected in a general form guided eventually by some additional criteria. Regarding the specific type of fuzzy sets used in the construction of the controller, one can select $\mathbf{x}$ and $\mathbf{u}$ viewed as singletions, say [0 0 ...0 1 0...0] . The relevant associations as built between $\mathbf{x}$'s and $\mathbf{u}$'s selected in this way can be determined quite easily. The computations become reduced to a straightforward enumeration of the edges of the unit hypercube of control and picking up the best singleton (namely the one yielding minimum of (3)). The inevitable drawback of this method is that the produced associations might be characterized by a relatively high level of the performance index Q. This is obvious considering the coarseness of the control actions taken into account during this search.

Another option worth exploring is to carry out a regular on-line control computations for **x** provided. These computations usually produce associations with lower values of Q (as we are not restricted to {0,1} membership values in control). In comparison to the first procedure, this option is definitely more computationally demanding.

Despite the character of the state fuzzy sets, the first phase provides us with the associations

$$\mathcal{R}= \{ \mathbf{x}_l, \mathbf{u}_l, \lambda_l \}$$

l=1,2,...,N, where λ_l stands for the confidence level attached to the l-th association and taken as a straightforward complement of the performance index, $\lambda_l=1-Q$. Obviously, the higher λ_l , the more significant the association. It should be emphasized that the states used to generate these associations should be representative, so that the resulting control mapping can be general enough and capable of providing meaningful actions even for a significant diversity of the environment.

The set of associations $\mathcal{R}$ will be used next in building an explicit control mapping between **x** and **u**. This construct constitutes the objective of the second phase of the design. The mapping can be implemented in various ways. Here we concentrate on the architecture of the logic processor, cf. Section 2. Its architecture bears a strong resemblance to standard fuzzy controllers. More formally we will write it down as

$$\mathbf{u}= LP(\mathbf{x}; \mathbf{w})$$

where **w** stands for a family of the connections of the processor. The parametric learning of the LP is completed based on the set of associations $\mathcal{R}$. The minimized performance index is of the form of a weighted sum of the distances,

$$Q=\sum_{l=1}^{N} \| \mathbf{u}_l-LP(\mathbf{x}_l,\mathbf{w})\| \lambda_l$$

that includes λ_l's as the relevant weighting factors of the corresponding cases in $\mathcal{R}$. This allows us to discount for weaker associations and force them to have less significant impact on the results of the learning procedure .

The architecture of the logic processor reflects the topology of any fuzzy controller in the sense that:

- the hidden layer (AND neurons) builds a collection of the conditions of the rules out of the subconditions appearing at the inputs of the network,

-the output layer (OR neurons) combines the conditions supporting the individual control actions. Hence each " if-then" control statement reads as

AND neuron $\swarrow$ OR neuron $\swarrow$

if

(subcond$_{1i}$ **and** subcond$_{2i}$ **and**... **and** subcond$_{ni}$) **or**

(subcond$_{1j}$ **and** subcond$_{2j}$ **and**... **and** subcond$_{nj}$) ...

then control$_r$

5.Conclusions

We have proposed a systematic and algorithmic way of designing fuzzy control algorithms. The essential point of the entire design procedure relies on the availability of the relevant fuzzy model. In contrast to broadly utilized operator-oriented approach to fuzzy control, the one discussed now is primarily model-oriented and shares all the advantages available within this methodology. In particular, an extensive characteristics of closed-loop control can be worked out in terms of fuzzy stability, fuzzy controllability, etc., where all these well-known terms of control engineering are moved from their original numerical niches and reformulated in the language of the linguistic quantities. The construction of fuzzy models and fuzzy controllers exploits fuzzy neural networks (logic processors). Their learning is completed following standard gradient-like methods; some other techniques like genetic algorithms [2] could be found beneficial as well.

Acknowledgment

Support from the Natural Sciences and Engineering Research Council of Canada and MICRONET is gratefully acknowledged.

6.References

1. D. Dubois, H. Prade, Possibility Theory - An Approach to Computerized Processing of Uncertainty, Plenum Press, New York, 1988.
2. D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, Reading, MA, 1989.
3. K. Hirota, W. Pedrycz, "Fuzzy logic neural networks: design and computations", Int. Joint Conf. on Neural Networks, Singapore, 18-21 November 1991.
4. W. Pedrycz, Neurocomputations in relational systems, IEEE Trans. on Pattern Analysis and Machine Intelligence, 13, 1991, 289-296.
5. W. Pedrycz, Fuzzy set framework for development of a perception perspective, Fuzzy Sets and Systems, 37, 1990, 123-137.
6. W. Pedrycz, Direct and inverse problem in comparison of fuzzy data, Fuzzy Sets and Systems, 34, 1990, 223-236.
7. W. Pedrycz, Selected issues of frame of knowledge representation realized by means of linguistic labels, Int. J.of Intelligent Systems, 7, 1992, 155-170.
8. W. Pedrycz, Fuzzy modelling: fundamentals, construction and evaluation, Fuzzy Sets and Systems, 41, 1991, 1-15.
9. W. Pedrycz,Fuzzy Control and Fuzzy Systems, 2nd edition, Research Studies Press/J.Wiley, Taunton/New York, 1993.
10. M. Sugeno, T. Yasukawa, A fuzzy-logic-based approach to qualitative modeling, IEEE Trans. on Fuzzy Systems, 1, 1993, 7-31.
11. L. A. Zadeh, Fuzzy sets as a basis for a theory of possibility, Fuzzy Sets and Systems, 1, 1978, 3-28.
12. L. A. Zadeh, Fuzzy sets and information granularity, in: M. M. Gupta, R.K. Ragade, R. R. Yager, eds., Advances in Fuzzy Set Theory and Applications, North Holland, Amsterdam, 3-18, 1979.