

# Towards Transparent Control of Large and Complex Systems

Jianwei Zhang, Alois Knoll and Ingo Renners

Faculty of Technology, University of Bielefeld

33501 Bielefeld, Germany

Tel.: ++49-521-106-2951

Fax: ++49-521-106-6440

E-mail: zhang|knoll|irenners@techfak.uni-bielefeld.de

<http://www.techfak.uni-bielefeld.de/techfak/ags/ti/>

**Abstract.** We first discuss the importance of making a controller interpretable and give an overview of the existing models and structures for that purpose. We then propose an approach to designing fuzzy controllers based on the B-spline model by learning. Unlike other normalised parameterised set functions for defining fuzzy sets, B-splines do not necessarily span membership values from zero to one but possess the property of “partition of unity”. B-splines can be automatically determined after each input is partitioned. Learning of a fuzzy controller based on B-splines is then equivalent to the adaptation of a B-spline interpolator. Parameters of the controller output of each rule can be rapidly adapted by gradient descent. Optimal placements of the non-uniform B-splines for specifying each input can be found by Genetic Algorithms. Through comparative examples of function approximation we show that training of such a fuzzy controller generally provides results with minimal error. The approach can be extended to the problems of high-dimensional input by combining neural networks with a fuzzy control model.

**Keywords:** Neuro-fuzzy system, Genetic Algorithms (GAs), interpretability, prediction, modelling, control

## 1 Introduction

Until recently, a large part of current science and technology was based on analytical methods which are usually specialised for pre-defined domains. For a human-being it is time-consuming to get acquainted with the model, to devise the model, even difficult to explain a model clearly to someone else. On this problem, physicist Richard P. Feynman mentioned “the way we have to describe nature is generally incomprehensible to us”. Nevertheless Albert Einstein believed “it should be possible to explain the laws of physics to a barmaid”.

Among the mechanisms to interpret the nature of a process like equations, tables, flow charts, stories, etc., fuzzy linguistic rules and relation descriptions are easy to understand. For building models with training data, certain types of fuzzy rule systems like the ANFIS-Model [4] and the B-spline model [14] have been developed which can approximate any low-dimensional input-output functions. There are good reasons for making such a controller model symbolically interpretable:

**International Journal of Computing Anticipatory Systems, Volume 7, 2000**

**Edited by D. M. Dubois, CHAOS, Liège, Belgium, ISSN 1373-5411 ISBN 2-9600179-9-4**

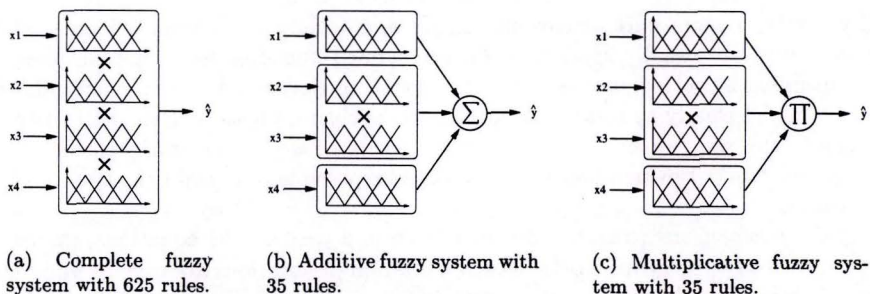
- Linguistic modelling provides a way of transferring skills from one human expert to other non-experts or to robots. The transferred rules and human knowledge can be used to accelerate the model-building and to patch the model in the case of data deficiency.
- Automatic learning of transparent models makes the analysis, validation and supervision in the model/controller development easier. By this way, a large part of design expenses can be shifted from humans to the computer.
- Transparent models will have wide applications in decision-support systems. In the next years, most of the control of complex systems will still be semi-autonomous. "Human-in-the-Loop" is based on compact and summarising descriptions of a system model.

New control and modelling approaches are enjoying the rapid increasing of computation power and memory brought by the computer technology. Advances have been made in the theory and applications of neural networks and fuzzy rule based systems to the control of physical systems which cannot be adequately modelled by linear differential equations. The fuzzy rule description of a system has the advantage over neural networks that it is not just a "black-box". Neuro-fuzzy models integrate automatic feature extraction and learning of membership functions into a fuzzy control structure. Together with optimisation algorithms, which deal with local minima, semi-automatic procedures can be designed for constructing non-linear models and controllers which can be understood by human users.

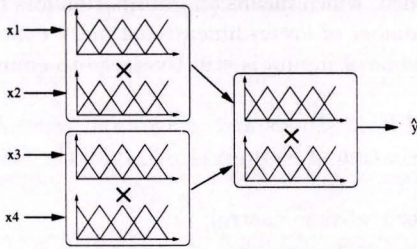
## 2 State of the Art

### 2.1 Structure of Interpretable Models

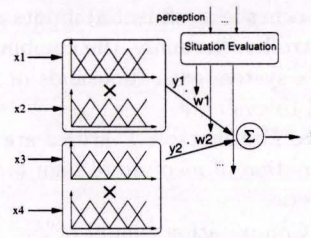
Some variances of fuzzy rule system are: additive, multiplicative and hierarchical. Fig. 1 and Fig. 2 illustrate these structures. The solution with hierarchical structure assumes that the input information can be classified into groups [12]. Within each group the inputs determine an intermediate variable, they can be decoupled from inputs of other groups. To realise such a grouping, usually heuristics based on the fusion of physical sensors have to be applied.



**Fig. 1.** Fuzzy system with 4 inputs and each with 5 linguistic terms.



(a) Hierarchical fuzzy system with four inputs, each with 5 linguistic terms – resulting in 75 rules.



(b) Behaviour blending using a two-step hierarchy. “Situation Evaluation” uses rules to determine the weight of each monolithic controller [15].

**Fig. 2.** Hierarchical fuzzy systems.

Classical fuzzy controller of the Mamdani type [6] is based on the idea of directly using symbolic rules for diverse control tasks. As application areas grow, the *systematic* design of an optimal fuzzy controller becomes more and more important. Another important type of fuzzy controllers is based on the TSK (Takagi-Sugeno-Kang) model [9]. Recently, TSK type fuzzy controllers have been used for function approximation and supervised learning [11]. However, it is pointed out that the TSK model is a black-box based on multi-local-model. In the following section, we describe an approach that can build membership functions (MFs) for linguistic terms of the IF-part systematically, then adapt the control actions of the THEN-part through learning and the shape and position of the IF-part MFs through genetic optimisation. Our approach is based on the B-spline model which can be classified as zero-order TSK model. However, we define linguistic terms for input variables with B-spline basis functions and for output variables with singletons. Such a method requires fewer parameters than other set functions such as trapezoid, Gaussian function, etc. The output computation is very simple and the interpolation process is transparent. We also achieved good approximation capabilities and rapid convergence of the B-spline controllers.

## 2.2 Automatic Information Filtering

The explosive amount of information can be efficiently utilised if the maximal information content is maintained while the dimension of the input data is reduced. The currently used methods for dealing with large systems are:

**Input Selection** This concept uses an experimental method to find the most important input variables among a large number of them [4]. All the combinatorial possibilities of the low-dimensional fuzzy model are considered and approximately tested. The selected inputs which enable the best result are viewed as the most influential ones to build an exact neuro-fuzzy model. This concept is simple to implement and the generated model is interpretable. But there are two disadvantages with this concept. First, all

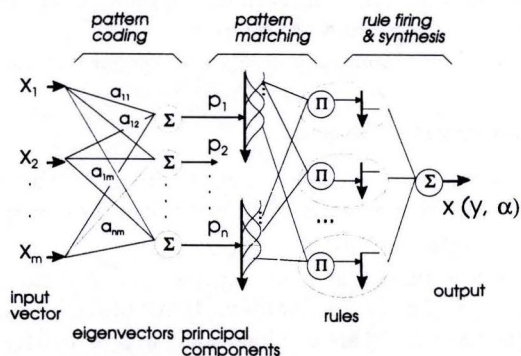
the other less influential inputs are discarded, which means an information loss for the controller. Secondly, the combinational number of lower-dimensional fuzzy controllers for a system with thousands or even millions of inputs is still too large to enumerate and to evaluate.

**Feature Extraction** Features are extracted from signals and images and represent information of medium to high level. The selection of features should fulfil the following criteria:

- Computation should be fast enough for real-time control;
- Feature variables should be as independent as possible;
- The feature vector should contain adequate information for determining the system;
- The features should be stably observable during the whole process.

If a feature possesses any meanings like geometry, attributes, intermediate variables, scenario, etc., it can be associated with some symbols, frames, stories, etc. These features are mostly selected manually from experience of human experts. Neural networks can be used as automatic methods for computing features, e.g. using linear combinatorators to determine *output relevant features* (ORFs) [18].

**Scenario Based Subspace Projection** Depending on how “local” the measuring data are and, therefore, how similar the observed sensor patterns appear during variations within a given situation, a more or less small number of eigenvectors calculated by a principal component analysis (PCA) can provide a sufficient summary of the state of all input variables (see the left part of Fig. 3). Our experimental results [18] show in the case of very high input dimensions, an effective dimension reduction can be achieved by projecting the original input space into the eigenspace [17]. The number of input variables in time series which possess in most cases a large degree of similarity, can be reduced using this method. Since this approach does not use any expensive computations for feature extraction, it can be called “appearance-based” or “scenario-based” approach. Eigenvectors can be partitioned by covering them with linguistic terms (the right part of Fig. 3). In our implementation, fuzzy controllers constructed according to the B-spline model are used [14].



**Fig. 3.** The task-based mapping can be interpreted as a neuro-fuzzy model. The input vector consists of all measurable system influential factors, e.g. many thousands of pixels of an image.

### 3 Constructing Fuzzy Controllers with B-Splines

#### 3.1 Basis Concept

Our B-spline model provides an ideal implementation of CMAC proposed by Albus [1]. B-spline models employ piecewise polynomials as MFs. The universe of discourse of each input is divided into a number of subintervals, where each subinterval is delimited by breakpoints called *knots* which determine the appearance and position of each B-spline (Fig. 4).

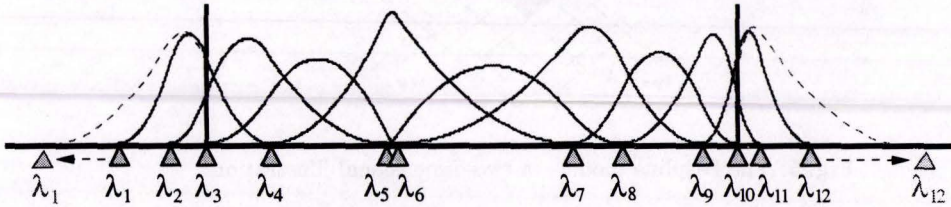


Fig. 4. Nine B-splines of order 3 defined over 12 non-uniformly distributed knots ( $\lambda_1, \dots, \lambda_{12}$ ).

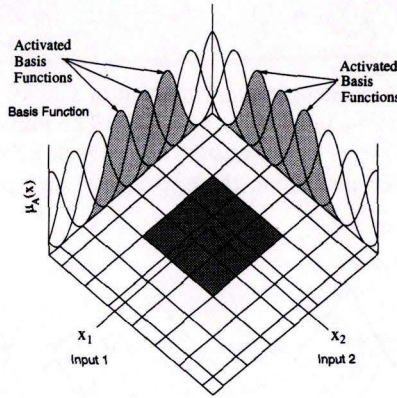
Figure 5 illustrates the partition of a two-dimensional B-spline model with 8 MFs on each uniformly subdivided input interval and the activated B-splines (slightly shaded) for a given input. Since learning one new part of the input space affects only a given number of controller response values (darkly shaded area of figure 5), fast on-line learning can be devised. Due to these advantages, B-spline models are proposed to be applied in control systems and will be denoted as B-spline Fuzzy Controllers (BFC)[14]. By using the B-spline model the approximation ability is only limited by the number of knots distributed over the input intervals. Regarding that most observed data are disturbed to a certain degree, the over-fitting problem may occur. GA optimised B-spline models are a promising approach to find sparse models, which are able to bridge the gap between high bias and high variance of a model.

#### 3.2 Definition of B-Splines

The B-spline  $N_{i,k+1}$  of degree  $k$  is recursively defined with knots  $\lambda_1, \dots, \lambda_{i+k+1}$  (see Fig. 6). Therefore  $m$  knots  $\lambda_i (i = 1, \dots, m)$  form  $l = m - k$  B-splines (Fig. 4). Using this  $l$  B-splines as linguistic terms, the minimum input value of a BFC is determined by  $a = \lambda_k$  and the maximum input value by  $b = \lambda_{m-k+1}$ , constituting the valid input interval to  $[a, b]$ . Thus the required parameters to define the structure of a normal B-spline model are:

$$Parameter(BFC) = \sum_{j=1}^n (m - 2 + 1) = \sum_{j=1}^n (m - 1) \quad (1)$$

The Term “-2” is needed since the two outermost knots does not influence the valid input interval  $[a_j, b_j]$  and the term “+1” is used because this parameter contains  $k_j$ .



**Fig. 5.** The B-spline model – a two-dimensional illustration.

The most important properties of B-splines, with respect to neuro-fuzzy modelling are:

- Recursion:

$$N_{i,p+1}(x) = \frac{x-\lambda_i}{\lambda_{i+p-1}-\lambda_i} N_{i,p}(x) + \frac{\lambda_{i+p}-x}{\lambda_{i+p}-\lambda_{i+1}} N_{i+1,p}(x)$$

$$N_{i,1}(x) = \begin{cases} 1, & \text{if } x \in [\lambda_i, \lambda_{i+1}] \\ 0, & \text{otherwise} \end{cases}$$

- Positivity:

$$N_{i,k} \geq 0 \text{ for all } x$$

- Local support:

$$N_{i,k} = 0 \text{ if } x \notin [\lambda_i, \lambda_{i+k}]$$

- Partition of unity:  $\sum_{i=1}^l N_{i,k}(x) = 1$

Each  $n$ -dimensional rectangle ( $n > 1$ ) of the lattice is covered by the  $j^{\text{th}}$  multivariate B-spline  $N_k^j(x)$  which is formed by taking the tensor product of  $n$  univariate B-splines:

$$N_k^j(x) = \prod_{j=1}^n N_{i_j, k_j}^j(x_j) \quad (2)$$

Therefore the shape of each B-spline, and thus the shape of multivariate ones (Fig. 7), is implicitly set by their order and their given knot distribution on each input interval.

### 3.3 Use B-Splines as Fuzzy Controller

In [14], we showed that under several slightly modified conditions, the computation of the output of such a fuzzy controller is equivalent to that of a *general B-spline hypersurface*.

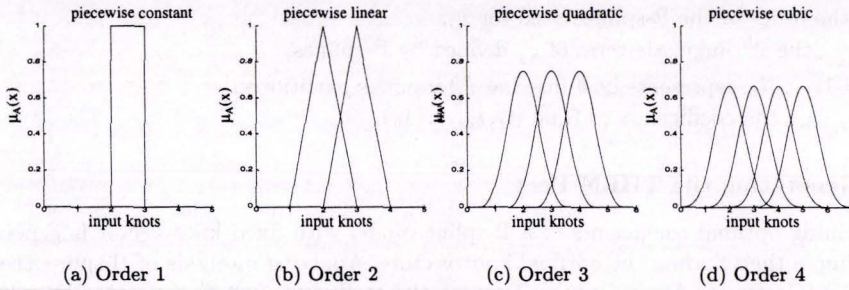


Fig. 6. Univariate B-splines of order 1-4.

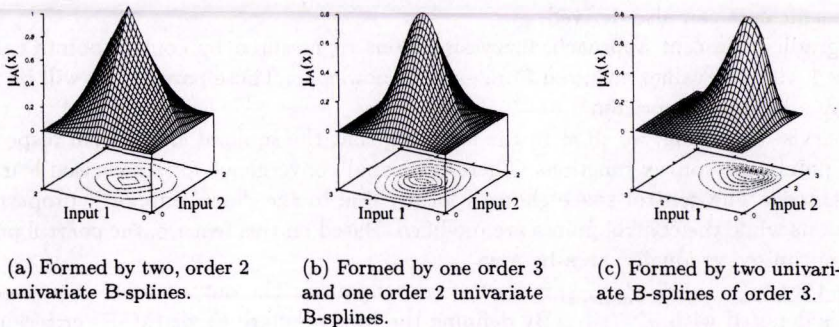


Fig. 7. Bivariate B-splines formed by taking the tensor product of two univariate B-splines.

The output of a Single Input Single Output (SISO) BFC is simply the unique representation of a B-spline  $s(x)$ ,  $x \in [a, b]$ :

$$y = \sum_{i=1}^l c_i N_{i,k}(x) \quad (3)$$

in which  $c_i$  are called *B-spline coefficients of  $s(x)$*  (also denoted as *weights*, *control points* or *de Boor points*) and  $l$  denotes the number of B-splines.

The output of a Multiple Input Single Output (MISO) B-spline network can be computed straightforward by merging (2) and (3):

$$y = \sum_{i_1=1}^{l_1} \cdots \sum_{i_n=1}^{l_n} \left( c_{i_1 i_2 \dots i_n} \prod_{j=1}^n N_{i_j, k_j}^j(x_j) \right) \quad (4)$$

where:

-  $x_j$ : the  $j^{\text{th}}$  input ( $j = 1, \dots, n$ ),

- $k_j$ : the order of the B-splines used for  $x_j$ ,
- $N_{i_j, k_j}^j$ : the  $i^{\text{th}}$  linguistic term of  $x_j$  defined by B-splines,
- $i_j = 1, \dots, l_j$ : represents how fine the  $j^{\text{th}}$  input is partitioned,
- $c_{i_1, i_2, \dots, i_n}$ : the coefficients of Rule  $(i_1, i_2, \dots, i_n)$ .

### 3.4 Generating the THEN-Part

Determining optimal coefficients of a B-spline model with fixed knot vector is generally more simple than finding the optimal knot vectors. Applying methods of B-spline theory in CAGD (Computer Aided Graphic Design), the coefficients can be estimated by solving an overdetermined linear system. Another method based on gradient descent can also be used. Although an iterative solution, this learning method is conceptually more easily understandable. Based on this learning method, unsupervised learning procedure of B-spline coefficients can also be derived.

In the gradient descent approach, fuzzy singletons represented by control points can be initialised with the values acquired from expert knowledge. These parameters will be fine-tuned by a learning algorithm.

For supervised learning, we show in the following that the squared errors with respect to control points are convex functions. Therefore, rapid convergence for supervised learning is guaranteed. The control space changes locally due to the "local support" property of B-functions while the control points are modified. Based on this feature, the control points can be optimised gradually, area-by-area.

Assume that  $(x_1^r, \dots, x_n^r, y_{desired}^r)$  is a set of training data. The output value computed by a BFC is denoted with  $y_{computed}^r$ . By defining the Mean-Square Error (MSE) criterion as:

$$E = \frac{1}{2} \cdot (y_{computed} - y_{desired})^2 \equiv \text{MIN}, \quad (5)$$

the derivation of each coefficient  $c_{i_1, \dots, i_k}$  is:

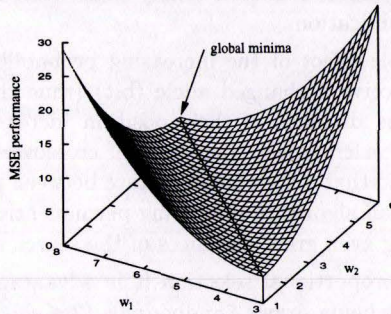
$$\Delta c_{i_1, \dots, i_k} = -\epsilon \frac{\partial E}{\partial c_{i_1, \dots, i_k}} = \epsilon (y_{computed}^r - y_{desired}^r) \cdot \prod_{j=1}^n N_{i_j, n_j}(x_{i_j}^r), \quad (6)$$

where  $\epsilon$  denotes the *learning rate*. Since the second partial derivation of  $c_1^r, \dots, c_k^r$  is always positive, the error function (5) is convex in weight space. If the inputs  $(x_1^r, \dots, x_n^r)$  are *linearly independent* there exists, due to the convexity of the MSE performance surface, only one global minimum and no local minima. On the other hand, if the autocorrelation matrix is singular, which occurs when the inputs  $x_r$  are *linearly dependent*, there exists an infinite number of global minima (Fig. 8), but still no local minima, in weight space.

### 3.5 Adaptive Modelling of the IF-Part

Based on the granularity of the input space and the distribution of extrema in the control space (if known), the fuzzy sets can be initialised using the recursive computation of B-splines. These fuzzy sets based on non-uniform B-splines can be further adapted during the generation of the whole system by using GAs.

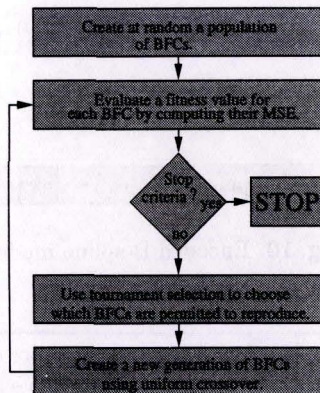




**Fig. 8.** A singular MSE performance surface.

By freeing the knots, the task of finding control points and accurate knot vectors to fit training data becomes a non-linear minimisation problem. To solve this problem we follow a strategy of problem splitting. We first consider the underlying model  $\delta(\lambda)$  of the controller and then compute the control points to minimise  $\delta(c)$ . Instead of using constrained least-square methods (constrained because of avoiding to “ride” on the gradient edge of coincident knots [5]), we try to estimate the knots by using GAs, because GAs are both theoretically and empirically proven to provide the means for efficient search, even in complex spaces [2]. Therefore each individual, in the example each B-spline controller with its special knot distribution, represents one point in search space.

**The Genetic Algorithms** We applied the basic GA introduced by Holland [3] with some modifications as follows (Fig. 9):



**Fig. 9.** Flowchart of the applied genetic algorithm.

- We used *gray coding* instead of standard binary code while representing coded chromosomes, a common modification.
- To bypass the undesirable effect of the increasing probability along the descendent chromosome-string to receive a changed allele (bit) (thus the conjointly heredity of genes decreases when the distance of their position increases) while using *n*-point crossover, we used *uniform crossover*. This kind of crossover has no positional and a high distributional bias, so that a high blending rate between participant chromosomes is granted. This leads to an algorithm producing permanently solutions which explore new locations by bridging even great distances of the search space.
- Instead of using fitness-proportional selection it is advantageous to use *tournament selection*. This selection schema draws  $\xi$  individuals ( $2 \leq \xi \leq \mu$ ) with a probability  $\frac{1}{\mu}$  from the current population and copies the individual with the best fitness into the mating pool. Besides saving computational power as a result of no need to sort the population (as in ranking based selection schemes), it is easier to bias the *takeover time*.

**Chromosome Encoding for Knot Placement** To minimise  $\delta(\lambda)$  each individual consists of *n* knot vectors, where *n* is the problem dimension. Each encoded knot vector consists of 32 knots and a so-called *activation string* of 32 bit length. Which knots are in use to define the current model is encoded through the activation string. Activated knots are represented by 1 and inactivated knots are represented by 0.

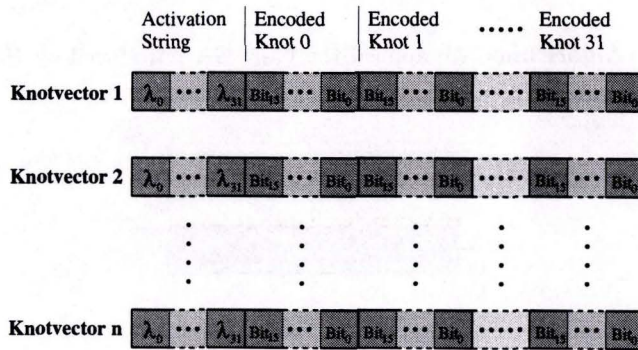
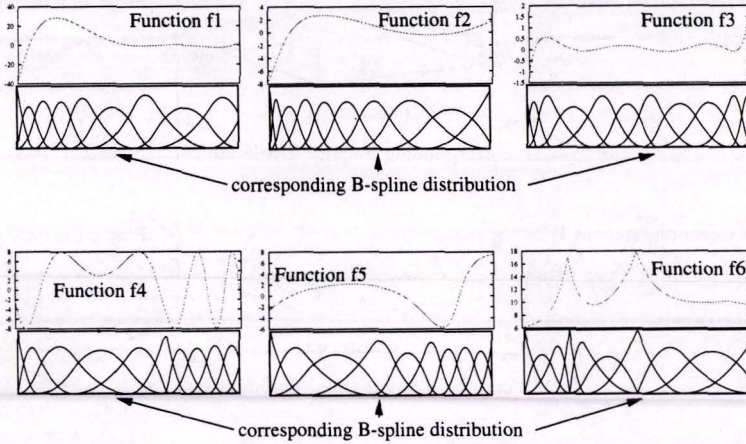


Fig. 10. Encoded B-spline model.

Every knot is encoded by 16 bit (see figure 10) and therefore each knot can be placed on its respective input interval  $[a, b]$  with an accuracy of  $\frac{1}{2^{16}} \times (b - a)$ . The fitness values for each individual are simply computed by determining the control points of the controller. Using these control points the mean square error is evaluated and the fitness for one individual is set equal to  $\frac{1}{MSE}$ .



**Fig. 11.** One dimensional functions and their encountered B-spline model.

### 3.6 Numerical Results

For comparison the above described GA was applied on one and two-dimensional functions also used in [8] (for function definitions see [8]). These one-dimensional and two-dimensional functions are depicted in Fig. 11 and 12 respectively.

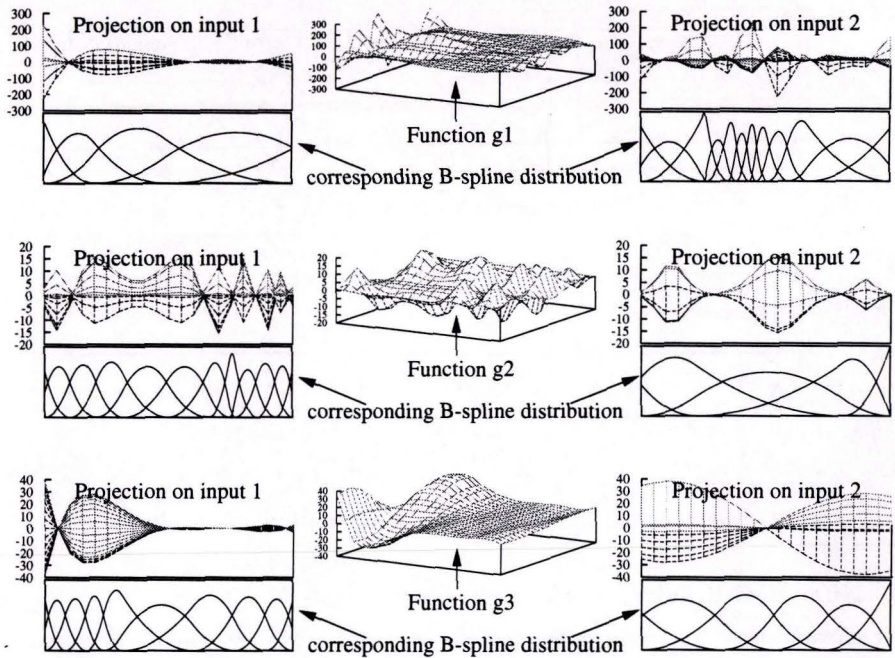
In these examples, parameter settings were chosen as  $\text{crossover\_probability}=0.75$ ,  $\text{mutation\_probability}=0.0005$ ,  $\mu = 40$  and  $\xi = 3$ . A maximum generation index of 200 was used as stop criterion. The MSE of each adapted BFC represents the average MSE of 3 runs. The comparative results are summarised in Tab. 1.

By searching for an optimal configuration of B-splines in two or more dimensions, it is important to consider how many B-splines are used for each input. In the examined two-dimensional cases we used the simple constraint of allowing no more than 64 rules. By using more input dimensions, the problem of finding (sub)optimal knot numbers becomes more relevant because of the exponentially increasing combination possibilities and thus an exponentially increasing search space.

## 4 Examples of Linguistic Interpretation

### 4.1 Approximation of a One-Dimensional Function

Intuitively, the basis functions can be efficiently used if they are placed at the positions where the corresponding system output changes dramatically. Depending on the smoothness of the original system input-output function and the demands on modelling precision, different orders of B-splines can be chosen. Optimal placements of the non-uniform B-splines for specifying each input can be found by genetic algorithms described above [16]. Fig. 13 illustrates a one-dimensional function and the generated membership functions for interpretation. Tab. 4.1 shows the rule table. Our method works in multi-dimensional cases as well.



**Fig. 12.** Two dimensional functions and their encountered B-spline model.

#### 4.2 Minimising the Number of Rules

Too large number of rules will not only result in the over-fitting problem, but also the lost of interpretability of the model. A psychological survey revealed that the number seven (plus or minus 2) is the human capacity limit for information processing [7]. By using linguistic modifiers like “between”, “at most”, “at least”, etc. and using an optimal partition algorithm [10], the rule base can be reduced to the minimum number, see the example shown in Tab. 3.

#### 4.3 Using Scenario Data

One main advantage of the neuro-fuzzy system in comparison with other adaptive systems like the multi-layer perceptron is the interpretability of the controller’s function. Since we can transform the projections in the eigenspace back into the original input image space, the control rules can be given an interpretation as IF-THEN rules, where the *Antecedent* is a back-transformed scenario image and the *Consequent* is controller output [17].


The following example illustrates the rules for a two-dimensional controller, whose task is to guide the robot hand using hand-camera images to the optimal grasping position over a screw. Each input variable with four linguistic terms. Therefore there are  $4 \cdot 4 = 16$  rules altogether. Two of these 16 automatically learned rotation control rules look as follows:


Function	Rules	Used Membership Function			
		Equidistant B-splines	Adapted B-splines	Best of [5]	Worst of [5]
$f_1$	12	0.02	0.007	0.08	0.7
$f_2$	12	0.0005	0.00003	0.02	0.3
$f_3$	12	0.0008	0.000048	0.002	0.03
$f_4$	12	4.9	0.04	0.1	10
$f_5$	12	0.04	0.0002	0.01	1
$f_6$	12	0.6	0.012	0.1	0.4
$g_1$	64	766	26.36 (60 rules)	9	26
$g_2$	64	10.91	2.8 (60 rules)	7	19
$g_3$	64	5.78	0.01 (60 rules)	1.2	6

**Table 1.** MSE results from [8] in comparison to results of an equidistant distributed BFC and results of a GA modified BFC.

IF x IS	VT	T	VS	S	L	VL	H	VH
THEN y IS	-13	12	-1	11	-15	15	-17	16

**Table 2.** Rules for system modelling - VT: "very tiny", T: "tiny", VS: "very small", S: "small", L: "large", VL: "very large", H: "huge", VH: "very huge".

IF the scenario is  THEN rotate the gripper  $-5^\circ$

IF the scenario is  THEN rotate the gripper  $4^\circ$

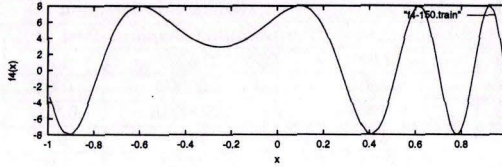
## 5 Future Research Themes

Integration of neural networks, fuzzy systems, genetic algorithms, chaos theory with the classical probability theory and control methods will play a central role in the future research. Among many topics, we list some important ones:

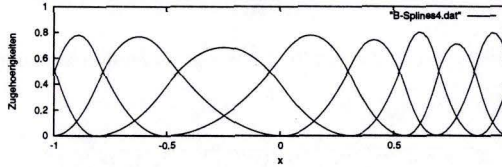
- Optimal input selection: factor analysis/synthesis, tracking focus of interests.

y \ x	VS	S	M	L	VL
VS	1	2	2	3	3
S	2	2	2	3	3
M	5	2	3	3	3
L	5	5	5	7	7
VL	5	5	5	7	7

**Table 3.** (2D) Rule table after the automatic optimal partition with the approach in [10].



(a) The input-output profile.



(b) The generating non-uniform B-splines as linguistic terms.

**Fig. 13.** Automatic model building using B-splines – a one-dimensional example function.

- Extension of sensor capability by using upto-date information technologies: software robots in WWW, linguistic modelling of human perception and sensor fusion so that information which is difficult to measure, incomplete or noisy can be perceived.
- Increasing the capability and quality of reinforcement signals and fitness evaluation of the learning system.
- Integrating symbolic sparse coding, granular computing, fuzzy set, rough set to enable the arbitrary transition between digital measurements and concepts.
- Learning from analytical models and generalise.

We believe that by solving the above problems properly we can

- find reliable models for traffic control, biological process, climate prediction, environment evolution.
- equip large amount of industrial machines and consumer products with higher Machine Intelligence Quotient (MIQ [13]) to make them easier to use, more adaptable in new environments, to show higher performance and to save resources.
- build robots which can program themselves using complex sensor patterns in the way a human will do.

## References

1. Albus, J. S. (1975). A new approach to manipulator control: The Cerebellar Model Articulation Controller (CMAC). *Transactions of ASME, Journal of Dynamic Systems Measurement and Control*, 97:220–227.

2. Goldberg, D. (1989). *Genetic Algorithms in Search Optimization, and Machine Learning*. Addison-Wesley, Reading, MA.
3. Holland, J. (1975). *Adaption in Neural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.
4. Jang, J.-S. R., Sun, C.-T., and Mizutani, E. (1997). *Neuro-Fuzzy and Soft Computing*. Prentice Hall.
5. Jupp, D. (1975). The 'lethargy' theorem - a property of approximation by  $\gamma$ -polynomials. *Journal of Approximation Theory*, 14:204-217.
6. Mamdani, E. H. (1993). Twenty years of fuzzy control: Experiences gained and lessons learned. *IEEE International Conference on Fuzzy Systems*, pages 339-344.
7. Miller, G. A. (1956). The magical number seven, plus or minus two; some limits on our capacity for processing information. *The Psychological Review*, 63(2):81-97.
8. Mitaim, S. and Kosko, B. (1996). What is the best shape of a fuzzy set in function approximation. In *IEEE International Conference on Fuzzy Systems*, pages 1237-1243.
9. Takagi, T. and Sugeno, M. (1985). Fuzzy identification of systems and its application to modelling and control. *IEEE Transactions on System, Man and Cybernetics*, SMC-15(1):116-132.
10. V., S. and Gorpinevich, A. (1993). Minimum dissection of a rectilinear polygon with arbitrary holes into rectangles. *Discret & Computational Geometry*, 9:57-79.
11. Wang, L. (1994). *Adaptive Fuzzy Systems and Control*. Prentice Hall.
12. Wang, L.-X. (1998). Universal approximation by hierarchical fuzzy systems. *Fuzzy Sets and Systems*, 93:223-230.
13. Zadeh, L. A. (1999). From computing with numbers to computing with words - from manipulation of measurements to manipulation of perceptions. *IEEE Transactions on Circuits and Systems -I*, 45(1):105-119.
14. Zhang, J. and Knoll, A. (1998). Constructing fuzzy controllers with B-spline models - principles and applications. *International Journal of Intelligent Systems*, 13(2/3):257-285.
15. Zhang, J. and Knoll, A. (1999). *Integrating deliberative and reactive strategies via fuzzy modular control*. In "Fuzzy logic techniques for autonomous vehicle navigation", edited by A. Saffiotti and D. Driankov, Springer.
16. Zhang, J., Knoll, A., and Renners, I. (1999a). Efficient learning of non-uniform B-splines for modelling and control. In *International Concerence on Computational Intelligence for Modelling, Control and Automation, Viena*, pages 282-287, Viena.
17. Zhang, J., Knoll, A., and Schmidt, R. (2000). A neuro-fuzzy control model for fine-positioning of manipulators. *Journal of Robotics and Autonomous Systems*. (to appear).
18. Zhang, J., Knoll, A., and Schwert, V. (1999b). Situated neuro-fuzzy control for vision-based robot localisation. *Robotics and Autonomous Systems*, 28:71-82.