# Evolution of Recurrent Fuzzy Controllers

Carlos Kavka, Patricia Roggero and Javier Apolloni

LIDIC
Departamento de Informática
Universidad Nacional de San Luis
Ejército de los Andes 950
D5700HHW - San Luis - Argentina
Tel: 02652-420823 Fax: 02652-430224
e-mail: {ckavka,proggero,javierma}@unsl.edu.ar

**Abstract**

The main advantage of a recurrent architecture is the ability to store information from prior system states. A recurrent fuzzy controller includes hidden fuzzy variables which makes the controller more appropriate to deal with dynamic systems. We are currently investigating the effect of evolution of recurrent fuzzy controllers by applying the FV representation, which provides a set of advantages that can significatively benefit the quality of the knowledge insertion process.

## 1 Introduction

One of the most successful areas of application of fuzzy logic is control, where fuzzy controllers have proven to be very effective in the context of controlling complex ill defined processes [8]. A fuzzy controller is usually designed by representing the knowledge of a human expert with a set of linguistic variables and fuzzy rules [5]. However, there is still no systematic way to perform this process. A large number of methods to automate this task and to evaluate and fine tune the obtained fuzzy controllers have been proposed in the literature, with methods based on reinforcement learning, neural networks and evolutionary algorithms being the most successful ones [9] [11]. Particularly, the methods based on evolutionary algorithms, usually called *genetic fuzzy systems*, do not suffer from the local minimum problem, with affects most other methods.

In most genetic fuzzy systems [2], the incorporation of previous knowledge (knowledge insertion) consists in the definition of fuzzy sets for the input variables and fuzzy rules. The benefit for the evolution is that the algorithm can obtain significatively faster good controllers. However, the adverse effect is that a restriction on the output values and the partition of the input space is introduced in the evolutionary process, which can limit the quality of the solution. One interesting representation for fuzzy systems is the Fuzzy-Voronoi (FV) model proposed by Kavka et al. [6, 7], where the a-priori knowledge is introduced by specifying behavior in single points in the input domain without a strict specification of the area of application of the fuzzy rules. In this way, the adverse effect of imprecisely defined inserted knowledge is reduced to a minimum.

Recurrent fuzzy systems are rarely studied, even if an impressive amount of research has been done in the area of recurrent neural networks and neuro fuzzy systems [10]. Recurrent fuzzy systems were introduced by Gorrini et al. [4]. The main advantage of a recurrent architecture is the ability to store information from prior system states, which makes it more appropriate to deal with dynamic

systems. We are currently investigating the effect of evolution of recurrent fuzzy systems by applying the FV representation, which provides a set of advantages that can significatively benefit the quality of the knowledge insertion process.

The paper is organized as follows: section 2 introduces the main characteristics of the FV representation for fuzzy systems, section 3 presents details on recurrent fuzzy systems, and section 4 our current developments.

# 2 The Fuzzy Voronoi Partition

The Fuzzy Voronoi representation belongs to the class of approximative representations [1] for fuzzy systems, where each fuzzy rule defines its own fuzzy sets, being in this case determined by a single multivariate membership function. In this section, we are just covering the domain partition induced by the representation, without entering into details on other aspects. The interested reader is redirected to [6, 7] and the references therein.

## 2.1 Voronoi diagrams and Delaunay triangulation

The domain partition strategy is based on Voronoi diagrams. A Voronoi diagram induces a subdivision of the space based on a set of points called *sites*. Formally [3], a Voronoi diagram of a set of $n$ points $P$ is the subdivision of the plane into $n$ cells, one for each site in $P$, with the property that a point $q$ lies in the cell corresponding to a site $p_i$ if and only if the distance between $q$ and $p_i$ is smaller than the distance between $q$ and $p_j$ for each $p_j \in P$ with $j \neq i$. Figure 1 illustrates an example of a Voronoi diagram in $R^2$. The definition can be straightforward extended to $R^n$, with $n \geq 2$. A related

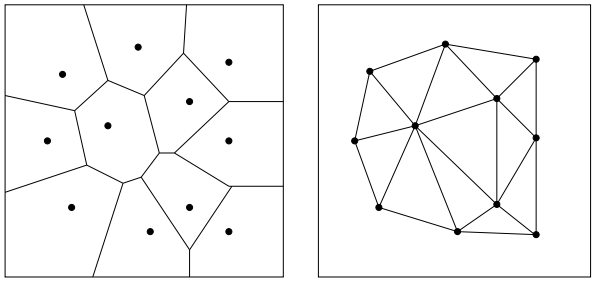


Figure 1: An example of a Voronoi diagram (left) and the corresponding Delaunay triangulation (right) for a set of points in $R^2$

concept that will be used in the paper is the so called Delaunay triangulation. A triangulation [3] of a set of points $P$ is defined as the maximal planar subdivision whose vertex set is $P$. A maximal planar subdivision $S$ is a subdivision such that no edge connecting two vertices can be added to $S$ without destroying its planarity. A triangulation $\mathcal{T}$ of a set of points $P$ is a Delaunay triangulation if and only if the circumcircle of any triangle in $\mathcal{T}$ does not contain a point of $P$ in its interior. A circumcircle of a triangle is defined as the circle that goes through its three summits. Figure 1 illustrates an example of a Delaunay triangulation in $R^2$.

## 2.2 The FV induced partition

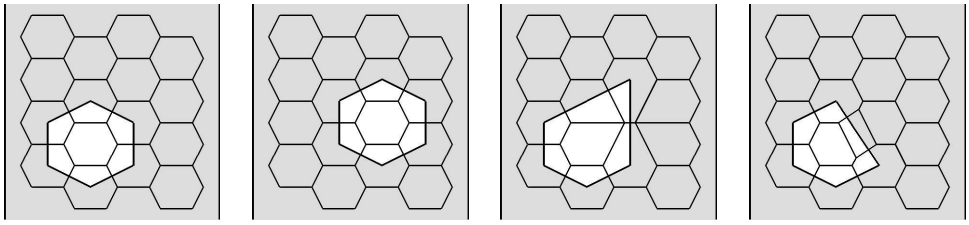
In the FV representation, the rule $R_k$ defines the joint fuzzy set by just specifying a point $p_k$ in the input domain. This point corresponds to the center of the Voronoi region $V_k$ associated to the rule.

The complete fuzzy system consisting of $\omega$ rules $R = \{R_1, R_2, \ldots, R_w\}$ defines a Voronoi diagram where the points $P = \{p_1, p_2, \ldots, p_\omega\}$ are the centers of the Voronoi regions $V = \{V_1, V_2, \ldots, V_\omega\}$.

The output produced by a fuzzy rule in the FV representation does not depend on the rule itself, it also depends on other rules defined in the database. The area of application $\mathcal{A}$ of a fuzzy rule $R_i$ is defined as the union of all Delaunay regions which contain the point $p_i$, center of the rule $R_i$. Formally:

$$\mathcal{A}(R_i) = \bigcup_{p_i \in D_j} D_j \quad D_j \in D = \{D_1, \ldots, D_\gamma\}. \tag{1}$$

where $p_i$ is the center of the rule $R_i$ and $D = \{D_1, \ldots, D_\gamma\}$ is the Delaunay partition induced by the points $P = \{p_1, \ldots, p_\omega\}$. The figure 2 shows an example of the application area of a single rule. The



(a) (b) (c) (d)

Figure 2: The diagram (a) shows the application area of a fuzzy rule and the diagram (b) shows the application area of a neighbor rule. The diagram (c) shows the application area of the first rule when the second rule is removed and the diagram (d) the application area of the first rule when a third rule is added between the first and the second rule. The diagrams correspond to a regular partition, where the unbounded rays of the open Voronoi regions are not shown

application area of a rule depends on the application areas of the other rules by its own definition. The application area of a rule is reduced if a new rule that defines a neighbor Voronoi region is added in the database. The application area of the rule $R_i$ is modified when the rule $R_{\omega+1}$ is added only if $\mathcal{A}(R_i) \cap \mathcal{A}(R_{\omega+1}) \neq \emptyset$. Inversely, the application area of a rule is enlarged if a rule that defines a neighbor Voronoi region is removed (see figure 2).

The evolutionary algorithm evolves individuals that represent fuzzy systems defined by a set of fuzzy rules synergically related, and not fuzzy systems defined with a set of independent fuzzy rules.

# 3 Recurrent fuzzy systems

The $k$-th rule of a fuzzy rule base of a $\sigma$ order recurrent fuzzy system looks like:

$$\begin{array}{ll} \textit{if} & s(t) \textit{ is } A_0^k \textit{ and } x_1 \textit{ is } A_1^k \textit{ and } \ldots \textit{ and } x_n \textit{ is } A_n^k \\ \textit{then} & s(t+1) \textit{ is } O_0^k \textit{ and } y^k \textit{ is } O_1^k \end{array}$$

where the $A_i^k$ are the antecedent fuzzy sets (or antecedent linguistic labels) associated to each input variable $x_i$, $y^k$ is the output variable, $O_i^k$ are the consequent fuzzy sets and $s(t)$ is the status of the system at time $t$. The system contains internal fuzzy variables than act like a short term memory. In this way, the output of a recurrent fuzzy system depends not only on the values of the current inputs, but also on the values of the previous system conditions:

$$y(t) = f(y(t-1), \ldots, y(t-\sigma), x_1(t-1), \ldots, x_n(t-1), \ldots, x_2(t-2), \ldots, x_n(t-\sigma))$$

Gorrini et al. [4] propose a gradient based learning algorithm to adapt the membership functions to model high order dynamic processes. Surmann et al. [12] uses genetic algorithms to optimize

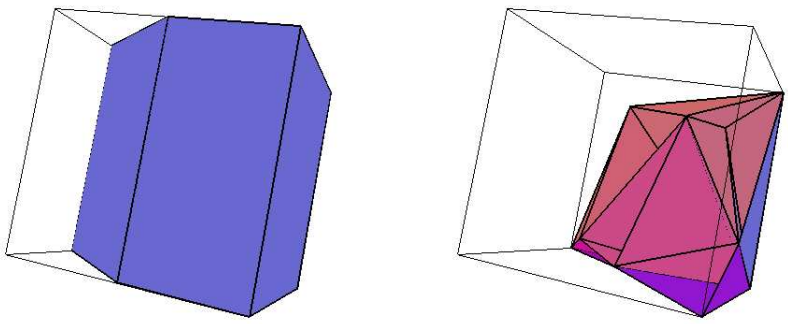
the width and position of the membership functions. Nurnberger [10] trains neural networks that represent hierarchical recurrent fuzzy systems.

# 4 FV based recurrent fuzzy systems

We are currently studying the application of evolution of recurrent fuzzy systems represented with the FV representation. One of the most important properties of the FV representation is the synergic relation between the rules, where the area of application of a rule is not determined by the rule itself, but depends also on the other rules. This synergic relation guarantees also the completeness property, which establishes that for very input there is at least a rule that is activated with a guaranteed minimum level of activation.

One of the most difficult problems faced with recurrent fuzzy systems is that it is not easy to define rules by using linguistic terms, with the possibility to define wrong a-priori knowledge increasing significatively. The algorithms mentioned in section 3 can deal with these problems, since they can update the membership functions by using gradient based algorithms. However, when using evolutionary algorithms, the bias introduced with apriori knowledge can introduce strong constraints in the search space.

In the FV representation, it is very easy to define rules that produce a certain value for a particular input point, without a specification of the area of application of the rule. Due to the completeness property, the whole domain is covered by the apriori rules, something that does not happen with most other representations. Evolution with FV based individuals that include the apriori rules produce new fuzzy systems that include the default behavior in the specified points. However, the application area of the rules (and the output produced in these areas) is modified by new rules in neighbor Voronoi regions. These changes in behavior are produced without an explicit modification of the apriori rules, which remain always the same. Just as an example, the figure 3 shows the change of the application area of an apriori rule during evolution for a problem with three input variables.



(a) (b)

Figure 3: The application area of an apriori rule before evolution and after evolution.

Recurrent fuzzy rules can be defined by using the same strategy. In particular, the definition of output values for a single point in the domain is as simple as in the non recurrent case. Experiments are being carried on the evolution of recurrent fuzzy controllers for avoiding moving obstacles. Note that a non recurrent controller cannot determine if an obstacle is moving, due to the static nature of its input/output behavior.

# References


[1] R. Babuška. Fuzzy modeling: Principles, methods and applications. In C. Bonivento, C. Fantuzzi, and R. Rovatti, editors, *Fuzzy Logic Control: Advances in Methodology*, pages 187–220. World Scientific, Singapore, 1998.

[2] O. Cordon, F. Herrera, and A. Peregrin. A practical study on the implementation of fuzzy logic controllers. *The International Journal of Intelligent Control and Systems*, 3:49–91, 1999.

[3] M. de Berg, M. van Kreveld, M. OVermars, and O. Schwarzkopf. *Computational Geometry, Algorithms and Applications*. Springer Verlag, 1998.

[4] V. Gorrini and H. Bersini. Recurrent fuzzy systems. In *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems*, pages 193–198, New Orleans, 1996.

[5] F. Hoffmann. Evolutionary algorithms for fuzzy control system design. *Proceedings of the IEEE, Special Issue on Industrial Innovations using Soft Computing*, 2001.

[6] C. Kavka and M. Schoenauer. Voronoi diagrams based function identification. In *Proceedings of the Genetic and Evolutionary Computation Conference*, 2003.

[7] C. Kavka and M. Schoenauer. Evolution of voronoi-based fuzzy controllers. Submitted to the International Conference on Parallel Problem Solving from Nature PPSN VIII, Birmingham, UK 2004.

[8] C. Lee. Fuzzy logic in control systems: Fuzzy logic controller - part i. *IEEE Transactions on Systems, Man and Cybernetics*, 20(2):404–418, March/April 1990.

[9] P. McQuesten. *Cultural Enhancement of Neuroevolution*. PhD thesis, The University of Texas at Austin, August 2002.

[10] A. Nurnberger. A hierarchical recurrent neuro-fuzzy model for system identification. *International Journal of Approximate Reasoning*, 32:153–170, 2003.

[11] A. Saffiotti. The uses of fuzzy logic in autonomous robot navigation. *Soft Computing*, 1(4):180–197, 1997.

[12] H. Surmann and M. Maniadakis. Learning feed-forward and recurrent fuzzy systems: a genetic approach. *Journal of Systems Architecture*, 47(7):649–662, August 2001.