International Journal of Management and Fuzzy Systems
2016; 2(4): 31-37
http://www.sciencepublishinggroup.com/j/ijmfs
doi: 10.11648/j.ijmfs.20160204.11

Tuning of Systematic Fuzzy Model by Using Evolutionary Algorithm for Control of General Systems

Alireza Rezaee

Department of System and Mechatronics Engineering, Faculty of New Sciences and Technologies, University of Tehran, Tehran, Iran

Email address:
arrezaee@ut.ac.ir

To cite this article:
Alireza Rezaee. Tuning of Systematic Fuzzy Model by Using Evolutionary Algorithm for Control of General Systems. *International Journal of Management and Fuzzy Systems.* Vol. 2, No. 4, 2016, pp. 31-37. doi: 10.11648/j.ijmfs.20160204.11

Received: September 10, 2016; **Accepted:** October 29, 2016; **Published:** December 27, 2016

Abstract: This paper proposes a systematic fuzzy model (SFM) to control of general system. SFM model is parted to multiple parts such as: a single parameter in formulation of reasoning; a linear relationship between input and output as a result; the use of evolutionary programming for the selection of the appropriate system parameters and a fuzzy clustering algorithm. Unlike traditional methods of inference mechanism to select a priori reasoning mechanism; SFM model can adjust its parameters using evolutionary programming. To vary the degrees of linear functions of the fuzzy rules, a set of equations describes the system's input and output locally. Thus, this model can take advantage of the properties of linear systems. Fuzzy rules, the fuzzy c- means clustering algorithm and proper selection of the cluster centers by using evolutionary algorithm have been investigated. Finally, this system has been tested and validated on both controlled robot arm joint.

Keywords: Robot, Arm, Fuzzy Clustering, Evolutionary Programming

1. Introduction

Fuzzy models describe system by establishing relations between the relevant variables in the form of IF-THEN rules. Traditionally, a fuzzy model is built by using expert knowledge in the form of linguistic rules. Recently, there is an increasing interest in obtaining fuzzy models from measured data. Different approaches have been proposed for this purpose, such as fuzzy relational modeling [1], neural-network training techniques [2] and product-space clustering [3-4]. In this paper an improved fuzzy modeling system is developed. The focus of this work is on the process of system identification for fuzzy modeling. The problem of system identification can be divided into two parts: Structure identification and parameter identification. In the structure identification stage, input-output relations in terms of IFTHEN rules are specified. In this paper, fuzzy clustering is considered as an approach of rule generation in fuzzy modeling. In this direction, fuzzy c-mean clustering is used and the objective of this technique is improved by proper choosing of the number and the level of fuzziness of clusters. A method for assignment of initial cluster center locations is also introduced. The parameter identification consists of derivation of the optimum inference parameters and adjustment of membership functions. We use an evolutionary algorithm for choosing optimum inference parameters. Another aspect of the proposed method is combination of nonlinear consequent and linear antecedent based on Takagi-Sugeno-Kang (TSK) approach to fuzzy modeling. In this approach, the consequents of fuzzy rules are linear functions of the antecedent variables, describing the system with a set of local linear input-output relations. Thus, the model can take advantages of linear system properties in the consequent. The rest of the paper is organized as follows: section 2 presents rule generation with fuzzy clustering methods. Fuzzy inference parameter optimization is presented in section 3. Section 4 discusses about fuzzy inference output parameter optimization. In section 5, a case study is presented for testing and verifying the proposed approach. Finally, the discussions and conclusions are appeared in section 6.

2. Rule Generation Using Fuzzy Clustering Methods

A clustering algorithm partitions a given data set into subsets, called clusters, with the aim of minimizing the difference of the objects in the same cluster and maximizing

the difference between the objects in different clusters. For fuzzy clustering from the available data sequence identify X and y: where, X is input matrix and y is output vector.

Fuzzy clustering in Cartesian product-space X...Y is applied to partition the training data into characteristic regions. Combining X and y form the data set Z to be clustered:

$$Z = [X\ y] \quad (1)$$

Given the training data Z and the number of the clusters c, the fuzzy c-means clustering algorithm is applied, which computes the fuzzy partition matrix U. The fuzzy sets in the antecedent of the rules are obtained from the partition matrix U, where 0,1.. ik u is the membership degree of the data object zk in cluster ci. The ith row of U contains a point wise definition of a multidimensional fuzzy set. One dimensional (1-D) fuzzy set Aij are obtained from the multidimensional fuzzy sets by projections onto the space of the input variables xj.

In this paper, we use fuzzy c-means (FCM) algorithm for data clustering. For proper choosing of number of clusters (c) and weighting exponent (m), several methods have been investigated.

Let X={x1,…,xN} be the set of input data, c number of clusters, and m weighting exponent that demonstrates degree of fuzziness. An efficient method of choosing c is the parameter of cluster validity index (Scs). The Scs is defined as following [5]:

$$S_{cs} = \sum_{i=1}^{c}\sum_{k=1}^{N}(u_{ik})^m \left(\|x_k - \vartheta_i\|^2 - \|\vartheta_i - \bar{\vartheta}\|^2\right) \quad (2)$$

In the above expression, i v is the cluster center of ci and v is the total mean vector. For obtaining proper c, one should minimize parameter of Scs with respect to c. However, this requires that we know the proper value of m. For choosing m [6] and [7] introduce new parameters called total scatter matrix (ST) and K.

$$S_T = \sum_{k=1}^{N}(\sum_{i=1}^{c}(u_{ik})^m)(x_k - \bar{v})\,(x_k - \bar{v})^T \quad (3)$$

$$K=\text{trace}\left(\sum_{l=1}^{N}\left[\left(x_l - \frac{1}{N}\sum_{k=1}^{N}x_k\right)(x_l - \frac{1}{N}\sum_{k=1}^{N}x_k)^T\right]\right) \quad (4)$$

Therefore, for clustering a data set, a suitable value of m may be found in the mid-domain of K, i.e., K/2 [7]. For this purpose, a variation of ST is plotted as a function of m with the different values of c. However, this methodology gives a reliable domain for selecting m instead of a single value. For proper selection of number of clusters c, we use the subtractive clustering method [8].

The subtractive clustering method assumes each data point is a potential cluster center and calculates a measure of the likelihood that each data point would define the cluster center, based on the density of surrounding data points. Steps of the algorithm are as follows:

- Selects the data point with the highest potential to be the first cluster center,
- Removes all data points in the vicinity of the first cluster center (as determined by radii), in order to determine the next data cluster and its center location,
- Iterate on this process until all of the data is within radii of a cluster center, the subtractive clustering method is an extension of the mountain clustering method proposed by Yager [9]. For the convenience of our description, let us rewrite subtractive clustering method as follows:

Each datais in a set of data points as a candidate cluster center. Defining a measure as

$P_i = \sum_{j=1}^{n} e^{-\xi\|x_i - x_j\|^2}$ for each data point, where, $\xi = \frac{4}{r_a^{\frac{2}{m-1}}}, r_a$ is a positive constant; Pidenotes the function of the distance xi to any other data point in a data set. Each value of Pi is calculated and then xi with the maximum distance value Pi is taken as a first cluster center x_i^*. The next cluster center is calculated and determined by $P_i = P_i - P_l^* e^{-\sigma\|x_i - x_l^*\|^2}$ where, $\sigma = \frac{4}{r_b^2}, r_b = 1.5r_a$. The nearer a data point to x_i^*. Results in the smaller related Pi. Thus, the data point will have less possibility to be taken as the next cluster center. Then, the third data point is taken and calculated as above. This process is continued until $P_k^* < 0.15P_l^*$ where $P_i = P_i - P_k^* e^{-\sigma\|x_i - x_k^*\|^2}$. After selecting the proper value of c by means of subtractive clustering, we can find the proper value of m. For this purpose, the trace of total scatter matrix (ST) is plotted. A suitable value of m is that which gives a value of ST equal to K/2 [6] and [7].

Another problem in FCM algorithm arises from the fact that this algorithm may produce only local minimum or partial optimal points. Therefore, different initial guesses for cluster centers may lead to different optimum results. In order to efficiently obtain a preference for initial location of cluster prototypes, the subtractive clustering method is used, too.

Next, the values of parameters for parameter identification should be computed. In a fuzzy model, the parameters are those concerned with membership functions. So, after the clustering process and obtaining the membership functions, the trapezoidal membership functions are approximated. For this purpose we first obtain the trapezoidal fuzzy set parameters [at btctdt] by cutting the membership membership functions by a-cut of level a= 0.1 and 1-a. The points of intersections of a- cut are used as an initial guess for parameters [at btctdt]. Then, these parameters are optimized to minimize the difference between the initial fuzzy sets and trapezoidal membership functions. Here, SIMPLEX optimization scheme is implemented, which gives the global minimum point.

3. Fuzzy Inference Parameter Optimization

This research uses the following linguistic fuzzy model, which consist of a set of fuzzy rules, each describing a local input-output relation in a linear form [10].

$$R_i : IF\ x_l\ is\ A_{il}\ \ AND \ldots AND\ \ x_n\ \ is\ A_{in}\ THEN\ \ \hat{y}_i = a_i x + b_i i = 1,2,\ldots \quad (5)$$

Where, i R is the ith rule, X=[x1 ,..., x n] is the vector of input variables, Ai1 ,..., A in are fuzzy sets defined in the antecedent space $\hat{y}_i$ is the rule output, c denotes the number of rules in the rule base, and isr means "is related to". The aggregated output of the model $\hat{y}$ is calculated by Basic Defuzzification Distribution (BADD) method [11]:

$$\hat{y} = \frac{\sum_{i=1}^{c} \beta_i^{\alpha}(x)\hat{y}_i}{\sum_{i=1}^{c} \beta_i^{\alpha}(x)} \quad (6)$$

Obviously, BAAD is essentially a family of parameterized defuzzification methods. By continuously varying in the real interval, it is possible to have more appropriate mapping from fuzzy set to a crisp value, depending on the system behavior. In the above formula $\beta_i(x)$ is degree of firing of the ith rule:

$$\beta_i(x) = T_{DP}(A_{i1}, \ldots, A_{in}) \quad (7)$$

Here, parametric method of Dubios&Prade [12] is used, for calculating degree of firing of rules in the formula (7):

$$T_{DP}(a,b) = \frac{ab}{\max(a,b,\gamma)} \quad (8)$$

In the above formula a and b are two fuzzy sets and γ is the parameter between zero and one.

For proper choosing of a and γ, an evolutionary programming (EP) [13] is implemented, since it does not demand a rich knowledge of the system behavior. Suppose that the consequent parameters i a and i b are determined by the process of determination of them explained in section 4. We apply the following EP procedure (EPI) for obtaining proper parameters:

1) Generate an initial population of individuals randomly and set k=1. Each individual is a pair of real-valued vectors $(w_i, \eta_i), \forall i \in \{1, \ldots, \mu\}$ where w_i are system parameters, i.e., a, γ, and η_i^s are variance vectors for Gaussian mutations (also known as strategy parameters in self-adaptive EA's). Each individual corresponds to a fuzzy model.

2) Each individual $(w_i, \eta_i), i \in \{1, \ldots, \mu\}$ creates a single offspring (w_i', η_i'):

$$\eta_i'(j) = \eta_i(j)\exp(\tau' N(0,1) + \tau N_j(0,1)) \quad (9)$$

$$w_i'(j) = w_i(j) + \eta_i'(j)N_j(0,1) \quad for\ j = 1, \ldots, n \quad (10)$$

Where, $w_i(j), w_i'(j), \eta_i$ and η_i' denote the jth component of the vectors w_i, w_i', η_i and $\eta_i', N(0,1)$ is a normally distributed one-dimensional random number with mean zero and variance equal to one $N_j(0,1)$ indicates that a random number isgenerated anew for each value of j, and parameters τ and $\tau'$are commonly set to $and\ \left(\sqrt{2\sqrt{n}}\right)^{-1}$ and $\left(\sqrt{2n}\right)^{-1}$ respectively.

3) Determine the fitness of every individual, including all parents and offsprings, based on the training error. Here, different error functions may be used. We use the performance index (PI) as a training error:

$$PI = \frac{\sum_{i=1}^{N}(y^i - \hat{y}^i)^2}{N} \quad (11)$$

4) Conduct pairwise comparison over the union of parents (w_i, η_i) and offsprings (w_i', η_i'), $\forall i \in \{1, \ldots, \mu\}$. For each individual, q opponents are chosen uniformly random from all parents and offsprings. For each comparison, if individual's fitness is not smaller than opponent's, it receives a "win". Select μindividuals individuals out of (w_i, η_i) and (w_i', η_i'), ,$\forall i \in \{1, \ldots, \mu\}$, that have most wins to form the next generation.

5) Stop if the halting criterion is satisfied (o.k.), otherwise put k=k+1 and go to step 2.

4. Fuzzy Consequent Parameter Optimization

Least squares method identifies the consequent parameters from given data, on the assumption that the input partition is given. The normalized activation degree of the rth rule for the kth input pattern by:

$$\emptyset_r(k) = \frac{\beta_r^{\alpha}(k)}{\sum_{r=1}^{c} \beta_r^{\alpha}(k)} \quad (12)$$

and combining these into matrix φ:

$$\emptyset = \begin{pmatrix} \emptyset_1(1) & \cdots & \emptyset_c(1) \\ \vdots & \ddots & \vdots \\ \emptyset_1(N) & \cdots & \emptyset_c(1) \end{pmatrix} \quad (13)$$

The output of fuzzy model computed over all input patterns (k=1…N) can be equivalent to:

$$y = \emptyset\theta \quad (14)$$

where,

$$\theta = [p_{01}, \ldots., p_{0c}]^T, p_{0i} = [a_i, b_i]^T \quad (15)$$

and

$$y = [y(1), \ldots, y(N)]^T \quad (16)$$

For the given y (the vectors of output reference values), the output parameters can be estimated by:

$$\theta = [\emptyset^T\emptyset]^{-1}\emptyset^T y \quad (17)$$

It should be noted that for each step of evolutionary programming, the candidate values for parameters α and γ and their related values and the calculated value of $.\hat{y}$ are chosen.

Finally, the performance index (PI) of the model is evaluated. If PI satisfies the evolutionary programming criteria, the values of α and γ are selected and the algorithm is ended; otherwise the above procedure is repeated.

5. Application Example

In this section, the proposed FSM is implemented to model

the inversekinematics of the two-joint planar robot arm shown in Figure 1 [14]. This problem involves learning to map from an end point Cartesian position (x, y) to joint angles (θ_1, θ_2) The forward kinematics equations from (θ_1, θ_2).to (x, y) are straightforward:

$$\begin{cases} x = l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) \\ y = l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2) \end{cases} \quad (18)$$

Where, l1 and l2 are arm lengths, and θ_1 and θ_2 are their respective angles (see Figure 1). However, the inverse mappings from (x, y) to (θ_1, θ_2) are not clear:

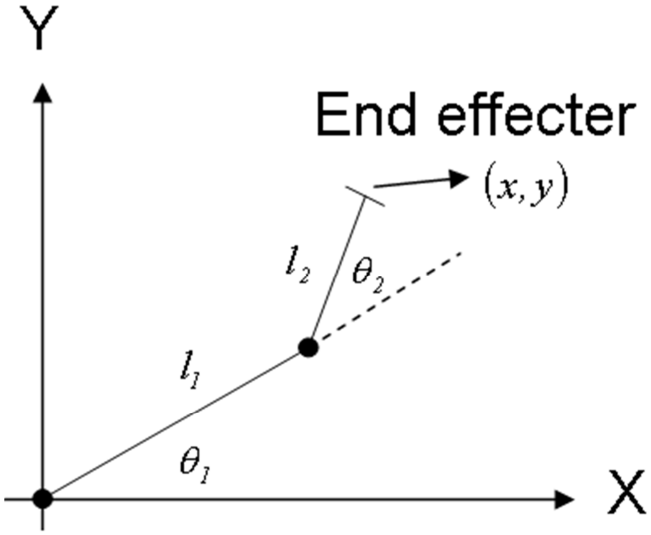


Figure 1. *Two-joint planar robot.*

$$\begin{cases} \theta_2 = cos^{-1}\left[\frac{x^2+y^2-l_1^2-l_2^2}{2l_1 l_2}\right] \\ \theta_1 = tan^{-1}\left(\frac{y}{x}\right) - tan^{-1}\left[\frac{l_2 sin\theta_2}{l_1 + l_2 cos\theta_2}\right] \end{cases} \quad (19)$$

Figure 2 demonstrates the forward mapping from$(\theta_1, \theta_2)$to (x,y) (the first row) and the inverse mapping from (x,y) to (θ_1, θ_2)(the second row). Here, it is assumed that l1=10,l2=7 and the value of $\theta_2$are restricted to$[0, \pi]$

Even though it is possible to find the inverse mapping algebraically, the solutions are not generally available for multi-joint robot arm in 3-D space. Instead of using the Equation (19) directly, we use two fuzzy modeling system to learn these inverse mappings.

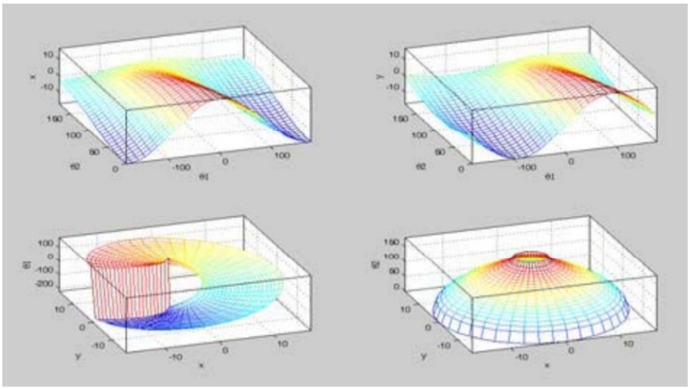


Figure 2. *Forward and inverse Kinematics of a Two-joint Planar for* (θ_1, θ_2) *to (x,y) and vice versa.*

From the first quadrature, 229 training data pairs of form (x, y, θ_1) and (x, y, θ_2) are collected, to train two fuzzy systems, respectively (this data can be find in traininv m-file in MATLAB). The parameters of two fuzzy models $(a_{ij}, b_i, c_{ij}, d_i, \alpha, \gamma)$ are available in tables 1 and 2. In Figures 4, 5 membership functions of input variables (A_{ij}, B_{ij}) are shown. Numbers of rules of fuzzy systems are 7. The generated fuzzy rules are as bellow:

$$Rule1:\ IF\ x\ is\ A_{11}\ AND\ y\ is\ A_{12}\ THEN\ \theta_1 = [a_{11} a_{12}][x\ \ y]^T + b_1$$

$$Rule2:\ IF\ x\ is\ A_{21}\ AND\ y\ is\ A_{22}\ THEN\ \theta_1 = [a_{21} a_{22}][x\ \ y]^T + b_2$$

$$Rule3:\ IF\ x\ is\ A_{31}\ AND\ y\ is\ A_{32}\ THEN\ \theta_1 = [a_{31} a_{32}][x\ \ y]^T + b_3$$

$$Rule4:\ IF\ x\ is\ A_{41}\ AND\ y\ is\ A_{42}\ THEN\ \theta_1 = [a_{41}a_{42}][x\ \ y]^T + b_4 \quad (20)$$

$$Rule5:\ IF\ x\ is\ A_{51}\ AND\ y\ is\ A_{52}\ THEN\ \theta_1 = [a_{51}a_{52}][x\ \ y]^T + b_5$$

$$Rule6:\ IF\ x\ is\ A_{61}\ AND\ y\ is\ A_{62}\ THEN\ \theta_1 = [a_{61}a_{62}][x\ \ y]^T + b_6$$

$$Rule7:\ IF\ x\ is\ A_{71}\ AND\ y\ is\ A_{72}\ THEN\ \theta_1 = [a_{71}a_{72}][x\ \ y]^T + b_7$$

$$Rule1:\ IF\ x\ is\ B_{11}\ AND\ y\ is\ B_{12}\ THEN\ \theta_2 = [c_{11}c_{12}][x\ \ y]^T + d_1$$

$$Rule2:\ IF\ x\ is\ B_{21}\ AND\ y\ is\ B_{22}\ THEN\ \theta_2 = [c_{21}c_{22}][x\ \ y]^T + d_2$$

$$Rule3:\ IF\ x\ is\ B_{31}\ AND\ y\ is\ B_{32}\ THEN\ \theta_2 = [c_{31}c_{32}][x\ \ y]^T + d_3$$

$$Rule4:\ IF\ x\ is\ B_{41}\ AND\ y\ is\ B_{42}\ THEN\ \theta_2 = [c_{41}c_{42}][x\ \ y]^T + d_4 \quad (21)$$

$$Rule5:\ IF\ x\ is\ B_{51}\ AND\ y\ is\ B_{52}\ THEN\ \theta_2 = [c_{51}c_{52}][x\ \ y]^T + d_5$$

$$Rule6:\ IF\ x\ is\ B_{61}\ AND\ y\ is\ B_{62}\ THEN\ \theta_1 = [c_{61}c_{62}][x\ \ y]^T + d_6$$

The antecedent spaces of the above models, which are two dimensional space of the inputs x and y, are partitioned into seven fuzzy subspaces. The above model can be regarded as a quasi-linear system (i.e., a linear system with input dependent parameters). To see this, we combine equations (5) and (6) and calculate the output of the model, $\hat{y}$

$$\hat{y} = \frac{\sum_{i=1}^{c} \beta_i^{\alpha}(x) \times (a_i x + b_i)}{\sum_{i=1}^{c} \beta_i^{\alpha}(x)} = k_i x + k_2 \quad (22)$$

$$k_1 = \frac{\sum_{i=1}^{c} \beta_i^{\alpha}(x) \times a_i x}{\sum_{i=1}^{c} \beta_i^{\alpha}}, k_2 = \frac{\sum_{i=1}^{c} \beta_i^{\alpha}(x) \times b_i}{\sum_{i=1}^{c} \beta_i^{\alpha}} \quad (23)$$

Figure 3 demonstratesθ_1 output from both fuzzy modeling system and equation (19). Obviously, fuzzy system properly approximates (with PI=.0018) the equation (19), so we can use the FMS in the case that it is not possible the inverse mapping algebraically.

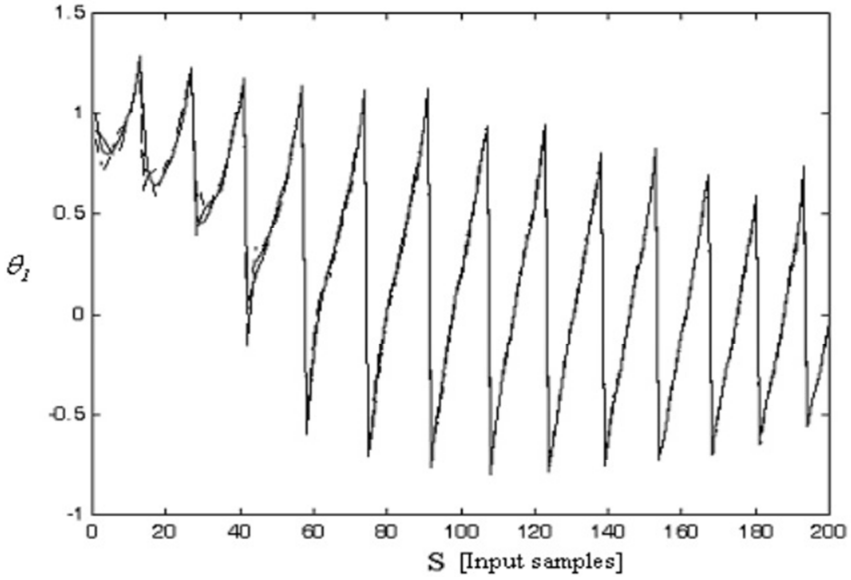


Figure 3. *Algebric output(----), Proposed fuzzy system output (- - - - -), ANFIS system output (-,-,-,-,-).*

Figure 3, also, demonstrates the result of comparing proposed FMS with ANFIS (adaptive neuro fuzzy inference system). The ANFIS has 9 rules consist of trapezoidal membership functions. The performance index (PI) for ANFIS is .0018for θ_1. So, our proposed system approximates the inverse kinematics with smaller number of rules and with the same performance index. Table 1 and 2 show the parameter of system for obtaining of θ_1 and θ_2 and figure 4 and 5 show the membership and equation of θ_1 and θ_2.

6. Conclusions

In this paper a FSM was introduced in which the inference mechanism, number of rules and order of fuzziness of model are identified from data. For this purpose, a parameterized formulation was applied by the suitable inference mechanism that is adjusted for the system based on evolutionary programming. Moreover, to take advantages of linear system properties, linear functions of the antecedent variables were implemented in the consequence of the model, thus, we could describe the system with a set of local with linear input-output relations and take advantages of linear systems properties. The modeling methodology described in this article has been successfully applied to inverse kinematics control of two-joint planar robot arm. The results show that the since system is self tuning, so we don't need to tune the system parameters again. The bright future works for this research is that the proposed method can be generalized so that all system structure and parameters can be obtained by the optimization methods such as genetic algorithms.

Table 1. *Parameters of FSM for* θ_1.

a_{i1}	a_{i2}	b_i	α	Υ	PI
-0.3256	-0.2192	1.8794			
-0.0473	0.0271	0.9012			
0.0041	0.4338	-0.5187			
-0.2966	0.7833	-2.6069	0.8609	0.5118	0.0018
0.0622	0.4518	-6.8508			
0.0517	0.0259	-0.2489			
0.0438	0.0785	-1.0941			

Table 2. *Parameters of FSM for* θ_2.

c_{i1}	c_{i2}	d_i	α	Υ	PI
1.2232	0.4456	-2.3191			
-1.4525	-1.6835	26.4234			
1.2160	0.3862	-16.5236			
-0.0865	-1.0860	10.8796	0.3139	0.1564	0.0080
-0.7162	-1.5388	16.3696			
-1.0411	0.4972	3.8106			
0.9470	1.9124	-12.0559			

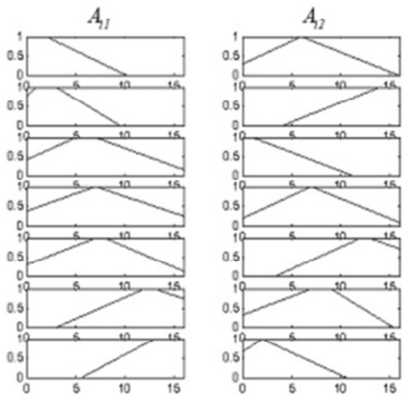
IF Part	THEN Part
$A_{i1}A_{i2}$	
IF Part 	$\theta_1 = -0.0473x + 0.0271y + 0.9012$
	$\theta_1 = 0.0041x + 0.4338y - 0.5187$
	$\theta_1 = -0.2966x + 0.7833y - 2.6069$
	$\theta_1 = 0.0622x + 0.4518y - 6.8508$
	$\theta_1 = 0.0517x + 0.0259y - 0.2489$
	$\theta_1 = 0.0438x + 0.0785y - 1.0941$

Figure 4. *Membership function and equation of FSM for* θ_1.

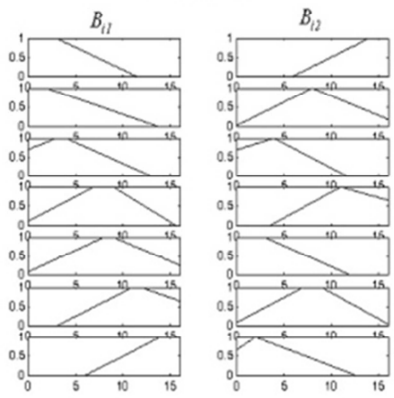
IF Part	THEN Part
$B_{i1}B_{i2}$	
IF Part 	$\theta_2 = 1.2232x + 0.4456y - 2.3191$
	$\theta_2 = -1.4525x - 1.68358y26.4234$
	$\theta_2 = 1.2160x + 0.3862y - 16.5236$
	$\theta_2 = -0.0865x - 1.0860y + 10.8796$
	$\theta_2 = -0.7162x - 1.5388y + 16.3696$
	$\theta_2 = -1.0411x + 0.4972y + 3.8106$

Figure 5. *Membership function and equation of FSM for* θ_2.

References

[1] W. Pedrycz, fuzzy Sets Engineering. Boca Raton, FL: CRC, 1995.

[2] B. Kosko, Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence. Englewood Cliffs, NJ: Prentice-Hall, 1992.

[3] J. H. Nie and T. H. Lee, "Rule-based modeling: Fast construction and optimal manipulation," IEEE Transactions on Systems, Man and Cybernetics, A, vol. 26, pp. 728-738, Nov. 1996.

[4] R. Babuska and H. B. Verbruggen, "Fuzzy set methods for local modeling and identification," in Multiple Model Approaches to Nonlinear Modeling and Control, R. Murray-Smith and T. A. Johansen, Eds. London, U.K.: Taylor& Francis, 1997, pp. 75-100.

[5] M. Sugeno and T. Yasukawa, "A fuzzy-logic-based approach to qualitative modeling," IEEE Transactions on Fuzzy Systems, 1, No. 1:7-31, 1993.

[6] M. R. Emami, I. B. Turksen, A. A. Goldenberg, "An improved fuzzy modeling algorithm, part II: System identification," NAFIPS, 1996.

[7] K. Al-Sultan. A Tabu Search Approach to Clustering problems pattern recognition, Vol. 28, pp.1443-1451, 1995.

[8] S. Chiu, "Method and software for extracting fuzzy classification rules by subtractive clustering," NAFIPS, 1996.

[9] A. Rezaee, "PSO for fuzzy goal programming," Appl. Comput. Math, 5(2):218-226, 2006.

[10] M. Setnes, R. Babuska, and H. B. Verbruggen. "Rule-based modeling: Precision and transparency." IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews, 28(1): 165-169, 1998.

[11] R. R. Yager and D. P. Filev, Essentials of Fuzzy Modeling and Control, John Wiley & Sons, Inc, 1994.

[12] D. Dubois, H. Prade, Fuzzy Sets and Systems: Theory and Applications. Academic Press: New York, 1980.

[13] X. Yao, "Evolving artificial neural networks," Proceedings of the IEEE, 87(9): 1423-1447, September 1999.

[14] J.-S. R. Jang and N. Gulley, "The Fuzzy Logic Toolbox for Use with MATLAB," The MathWorks, Inc., Natick, Massachusetts, 1995.

[15] www.mathworks.com