

Fuzzy Sets and Inference as an Effective Methodology in the Construction of Intelligent Controllers

Jean J. Saade

*ECE Department, FEA, American University of Beirut
P.O.Box: 11-0236, Riad El Solh 1107 2020, Beirut, Lebanon
Fax: 961.1.744 462, e-mail: jsaade@aub.edu.lb*

Abstract- Intelligent controllers are human-like thinking machines. They have the objective of controlling ill-defined, vague and complex processes in a manner similar to the control of human experts. Emphasizing fuzzy sets and inference as an effective methodology in the construction of intelligent controllers is the subject of concentration. This is based on the fact that fuzzy inference is capable of delivering machines that apply the approximate reasoning principles and other important aspects of intelligent thinking. Case studies related to non-linear function representation and robot navigation are presented to show the success of fuzzy inference in the production of intelligent machines. Data-driven and other methodologies, which are used in the construction of intelligent controllers, are emphasized and the superiority of the data-driven fuzzy learning methodology is presented.

Keywords: Intelligent controllers; Fuzzy sets and inference; Approximate reasoning; Vehicle navigation; Learning.

I. INTRODUCTION

The construction or design of intelligent controllers using fuzzy sets and inference has been an active area of research for quite a number of years. A part of this research has been concerned with the development of automatic, data-driven learning algorithms. The objective of these algorithms is to provide a fuzzy system that approximates available input-output data considered to model the human expert's control actions. Different data-driven approaches have been published. We state here, for instance, the neuro-fuzzy approaches [1-7] and approaches based on the use of clustering, genetic algorithms and combined gradient-descent-least-squares [7-19]. When the above-noted algorithms were tested and compared using non-linear functions and/or control applications, the testing and comparison has almost relied on the sole use of data approximation error as a measure of performance.

In this study, it is shown that the data approximation error cannot be considered as the only measure of performance. Rather, practical performance criteria, based on important aspects of intelligent human thinking, need to be defined and used to test, compare and determine preferences between data-driven fuzzy controller's construction algorithms. This is done in Section 2.

Section 3 reviews the literature related to the issues of noisy and incomplete training data as they relate to the performance criteria in the context of the design of fuzzy inference systems. In

this section, emphasis is also placed on the major drawbacks of the existing neuro-fuzzy approaches for fuzzy system modeling and on the fact that these drawbacks emerge from the basic structure of Takagi-Sugeno type controllers and the minimization of data approximation error.

Furthermore, Section 4 outlines the learning procedure implemented in a data-driven and purely fuzzy learning methodology for Mamdani-type fully-linguistic controllers [20] and states the advantages of this methodology compared to the neuro-fuzzy methods and the reasons for these advantages. In this section a summary of the fuzzy learning algorithm and its design aspects are also provided.

Then, a typical non-linear function, which was considered in the literature to test existing neuro-fuzzy, clustering and other design approaches, is considered in Section 5. The objective is to use the performance criteria to test the design algorithms and give comparisons and preferences. Particular emphasis is placed on the comparison of the results obtained using the fuzzy learning methodology with those given by other learning approaches.

In Section 6, use is made of the defined performance criteria to compare the fuzzy learning methodology with a powerful neuro-fuzzy approach in the area of robot navigation. Conclusive comments related to the superiority of the fuzzy algorithm over neuro-fuzzy and other approaches are offered in Section 7.

II. PERFORMANCE CRITERIA

Practical performance criteria are defined in this section based on important aspects of intelligent human thinking; such as approximate reasoning reflected as tolerance for imprecision and generalization. These criteria are then made available to test, compare and determine preferences between data-driven fuzzy controller's construction algorithms.

It is true that a fuzzy controller is a non-linear controller that should approximate a non-linear function according to which a human expert performs the control of some ill-defined, vague and complex process. This function, however, is practically unknown [8, 21-24]. Thus, expert input-output data are, in fact, measurements of the control actions taken by a human expert in response to process states while a control task is being performed. Hence, the data are practically noisy versions of the expert's actual control actions, which obey the non-linear function representing the process control.

Also, the data can be incomplete or not available in some region(s) of the input space. This could be due to missing

measurements resulting from the fact that the expert has not gone into situations where the fuzzy system designer is willing to have his system able to venture and still perform satisfactorily. After all, an intelligent system should be one that, when trained in some situation, behaves satisfactorily by generalization in a related situation whose facts were not used in training. Such an aspect does, in fact, characterize the nature of human intelligence.

Henceforth, when non-linear functions are considered to test the performance of fuzzy inference systems resulting from the use of data-driven design algorithms and establish preferences between these algorithms, the following practical performance criteria need to be adopted:

- (a) the value of some error function in the approximation of the underlying noise-free data when the training data are noisy,
- (b) the noise insensitivity,
- (c) the generalization capability,
- (d) the noise insensitivity and generalization capability,
- (e) the representation of the shape and smoothness of the non-linear function.

Assessing the performance criteria in the presence of noise (points (a) and (b)) can be done as follows: The data points extracted from a non-linear function, or some of them, are to be modified to violate the function analytical equation. The modified data are to be used in training and the extracted data are considered the noise-free ones. The smaller is the value of the error at the noise-free data and the larger the error at the noisy points, the better the performance of the fuzzy system modeling approach. We note here that both points (a) and (b) need to be considered. Since as a result of introducing noise, having the obtained fuzzy system not responding to or staying away from the noisy points does not necessarily imply that it will get closer to the noise-free data and thus to the real control curve.

As to generalization, data points in some boundary region of the input space and selected from among those extracted from a non-linear function are to be eliminated. Then, the fuzzy system obtained by training using the remaining data is to be tested based on its ability to extrapolate to the region of missing data. This can be assessed by observing whether the system obtained by training based on the remaining data is the same or close to the one obtained using the whole data set. Also, if necessary, the error value at the excluded points and at the whole set of data can be considered in the assessment of generalization.

Point (e) consists of testing the capability of the fuzzy system design approach to achieve smooth control. Point (e) can also be considered to assess generalization taken in the sense of interpolation to points within the training data.

III. REVIEW OF LITERATURE RELATED TO NOISY AND INCOMPLETE TRAINING DATA

After raising the issues related to the performance testing and comparison of data-driven fuzzy controllers design algorithms and arguing the fact that performance criteria accounting for noisy and incomplete training data need to be considered due to practical reasons (Saade, [25]) and (Saade and Al-Khatib, [26]), research reports in which these important matters have been approached started to appear. Oh and Pedrycz [27] introduced an auto-tuning

algorithm to identify T-S type fuzzy systems using a weighted performance index. The objective was to provide a balance between the approximation and generalization aspects of fuzzy models. Branco and Dente [28] pointed out ignored issues in fuzzy models design. They addressed the appearance of noise as a source of ambiguity to the fuzzy model, the fuzzy model generalization ability and the influence of the training set size on the learning performance. According to the authors, the improvement of the data approximation error by a few percentages in a new algorithm could make the new model predictions (generalization) irrelevant.

Furthermore, Shi and Mizumoto [15] used the fuzzy c-means clustering to preprocess the data, remove existing redundancies (noise) and extract typical data to be used for training in a neuro-fuzzy learning algorithm. Leski [29] recognized the intrinsic inconsistency of neuro-fuzzy modeling due to its zero tolerance to imprecision while fuzzy modeling is based on the premise that human thinking is tolerant to imprecision.

Consequently, the studies reported in [15, 27-29] can be used to provide an additional validation of the issues raised in Section 2 and in [25,26] and to the fact that the performance assessment of data-driven fuzzy systems modeling algorithms need to be done by accounting for noisy and incomplete training data. It can also be concluded from these studies that a fuzzy controller construction approach that is structured based on the minimization of data approximation error hinders the noise insensitivity and generalization capability of the resulting fuzzy model and it contradicts with Zadeh's principle of "*tolerance for imprecision*" [30].

What could be added here as well is a remark about the fact that it is the T-S fuzzy system model that has triggered the appearance of the numerous neuro-fuzzy research reports in the area of fuzzy systems modeling. In addition to having crisp values or linear combinations of the system input variables as rules consequents, an aspect that diminishes the fuzzy system linguistic representation of human knowledge, T-S models supplied with neuro-fuzzy learning techniques also have many other undesirable aspects that were brought out in [5].

Henceforth, in the next section a summary of a data-driven purely fuzzy learning methodology for Mamdani-type and fully-linguistic fuzzy controllers is provided. The learning algorithm enables the full linguistic representation of human knowledge and expertise and permits thinking tolerant to imprecision by not seeking error minimization. It rather seeks error reduction and requires that the learning stop once the error becomes less than or equal to some threshold value that can be set by the system designer. The algorithm will also be shown, as a result, to possess good noise insensitivity and generalization capability. It will in addition be compared to fuzzy clustering and partition approaches and most importantly to ANFIS [13] using a typical non-linear function and robot navigation.

IV. SUMMARY OF A DATA-DRIVEN FUZZY LEARNING ALGORITHM

The data-driven fuzzy learning algorithm for the design of intelligent controllers considers fully-linguistic fuzzy inference systems of Mamdani-type. In these systems, the fuzzy output, for a given crisp input value or vector, is obtained using the compositional rule of inference (CRI) [30] and defuzzification is applied to this fuzzy output to convert it into a crisp one.

Defuzzification in this algorithm is accomplished by applying a parameterized strategy that was established in [31].

Consider a collection of N if-then fuzzy inference rules for a two-input, one-output fuzzy inference system. Let the j -th rule, $1 \leq j \leq N$, be:

if x_1 is A_j and x_2 is B_j , then z is C_j .

x_1 and x_2 and z are the input and output variables of the system. Also, A_j , B_j and C_j are fuzzy sets defined over x_1 , x_2 and z respectively. The fuzzy output, corresponding to some crisp input pair (x_{1i}, x_{2i}) , can be obtained using the CRI as follows:

$$C_{0i}(z) = \max_{1 \leq j \leq N} [A_j(x_{1i}) \wedge B_j(x_{2i}) \wedge C_j(z)]. \quad (1)$$

There is a need to stress here on Zadeh's rule of composition. Consider first the two variables u and v , which assume values respectively in spaces U and V . Let A and B be two fuzzy sets defined respectively over the spaces U and V . The fuzzy conditional statement expressed as "If u is A , then v is B " can be interpreted as a fuzzy relation R defined by the Cartesian product of A and B . That is, [If u is A , then v is B] $\equiv R = A \times B$. The membership function of R , denoted $R(u, v)$ can be obtained using some operation, called "fuzzy implication operation", between $A(u)$ and $B(v)$, which are the membership functions of A and B respectively. The minimum operation has initially been suggested by Zadeh [30]. Now, the fuzzy relation R , as above, induces from a fuzzy set A' , defined over the space U , a fuzzy set B' , over the space V , such that $B' = A' \circ R$, where $\circ$ denotes relation composition. If the max-min composition is used [32], then the following is obtained:

$$B'(v) = \max_{u \in U} [\min(A'(u), R(u, v))].$$

When the fuzzy set A' is a singleton; that is, $A' = u_0 \in U$, the above equation becomes: $B'(v) = R(u_0, v)$. This is the case of interest in this study since the fuzzy controller is assumed, as is mostly the case, to be one that admits crisp inputs.

Returning now to our collection of N if-then fuzzy inference rules for a two-input, one-output fuzzy inference system, then the system rules can be represented by the following fuzzy relation:

$$\begin{aligned} R &= [(A_1 \cap B_1) \times C_1] \cup [(A_2 \cap B_2) \times C_2] \cup \dots \cup [(A_N \cap B_N) \times C_N] \\ &= \bigcup_{j=1}^N [(A_j \cap B_j) \times C_j]. \end{aligned}$$

In this relation, the symbol $\cup$ is taken as a representation of the OR operator introduced between the rules. The symbol $\cap$ represents the operator AND used in the antecedent part of the rules. The fuzzy controller output that corresponds to a crisp input pair (x_{1i}, x_{2i}) can, therefore, be obtained by $C_{0i}(z) = R(x_{1i}, x_{2i}, z)$. If the minimum operation ($\wedge$) is adopted for AND and THEN operators and the maximum (max) operation is used for OR, then Equation (1) above is obtained. Other operations, such as sum and product, can also be used. Also, Equation (1) can be generalized easily to systems with more than two input variables.

Now, the parameterized defuzzification method, developed in [31], applies to the normalized version of $C_{0i}(z)$, denoted, $C_{0in}(z)$, and obtained by dividing the membership function of $C_{0i}(z)$ by the highest membership grade, as follows:

$$F_{\delta}[C_{0in}(z)] = \int_0^1 [\delta c_1(\alpha) + (1 - \delta) c_2(\alpha)] d\alpha \quad (2)$$

$[c_1(\alpha), c_2(\alpha)]$ is the α -level set of $C_{0in}(z)$ and δ is a parameter that takes values in the interval $[0, 1]$. The defuzzification method in Equation (2) was derived by first reformulating the classical criteria; minimax, maximax and the Hurwicz criterion, which apply for ranking intervals, using the intervals characteristic functions. Then, the reformulation of the noted criteria for intervals permitted their natural extension to fuzzy sets by replacing the intervals characteristic functions by the fuzzy sets membership functions. The criteria, which were originally expressed using a defined distance measure and integration of the characteristic or membership functions along the real axis, were then expressed by an equivalent integration along the membership axis using the α -level set of a fuzzy set [33].

Furthermore, the study in [31] was one that formally related ranking fuzzy sets to the defuzzification of the outputs of fuzzy controllers. It unified the two problems to make the solution for the first applicable to the second. Also, the benefits that could be obtained if the defuzzification of the outputs of fuzzy controllers is approached from the point of view of ranking, and through the use of a parameterized defuzzification formula to satisfy design objectives (shaping the controller input-output characteristic), have been emphasized. It is in this spirit that the study in [31] unified the criteria introduced in [33] to be represented by a single parameterized ranking and defuzzification formula, which is the one expressed in Equation (2).

In the data-driven learning algorithm that is described below, and whose flow chart is shown in Figure 1, Equation (2) is used for tuning initial fuzzy controllers based on input-output data. This tuning considers the consistent modification of the parameter δ and the rules consequents to reduce the value of some error function and obtain a final fuzzy system. It is assumed that the system designer is able to specify the input and output variables of the fuzzy controller and the ranges of these variables. Then overlapping membership functions are assigned to cover the entire ranges of the variables of concern. In terms of overlap, it was observed that the smoothness of the controller input-output surface is best served whether the input membership functions, assigned over a single variable, are such that the sum of the membership grades at any crisp input is one.

Once the input membership functions are assigned, then all combinations of input fuzzy sets are considered to form the antecedent parts of the rules. All the initial rules consequents are required to be equal to the left most of the fuzzy sets assigned over the output variable. This set needs to be formed by a flat and a decreasing part or a decreasing part only. This makes the defuzzified value of any fuzzy output, obtained by Equation (1) and through the application of Equation (2) with $\delta=1$, equal to the smallest value of the output range.

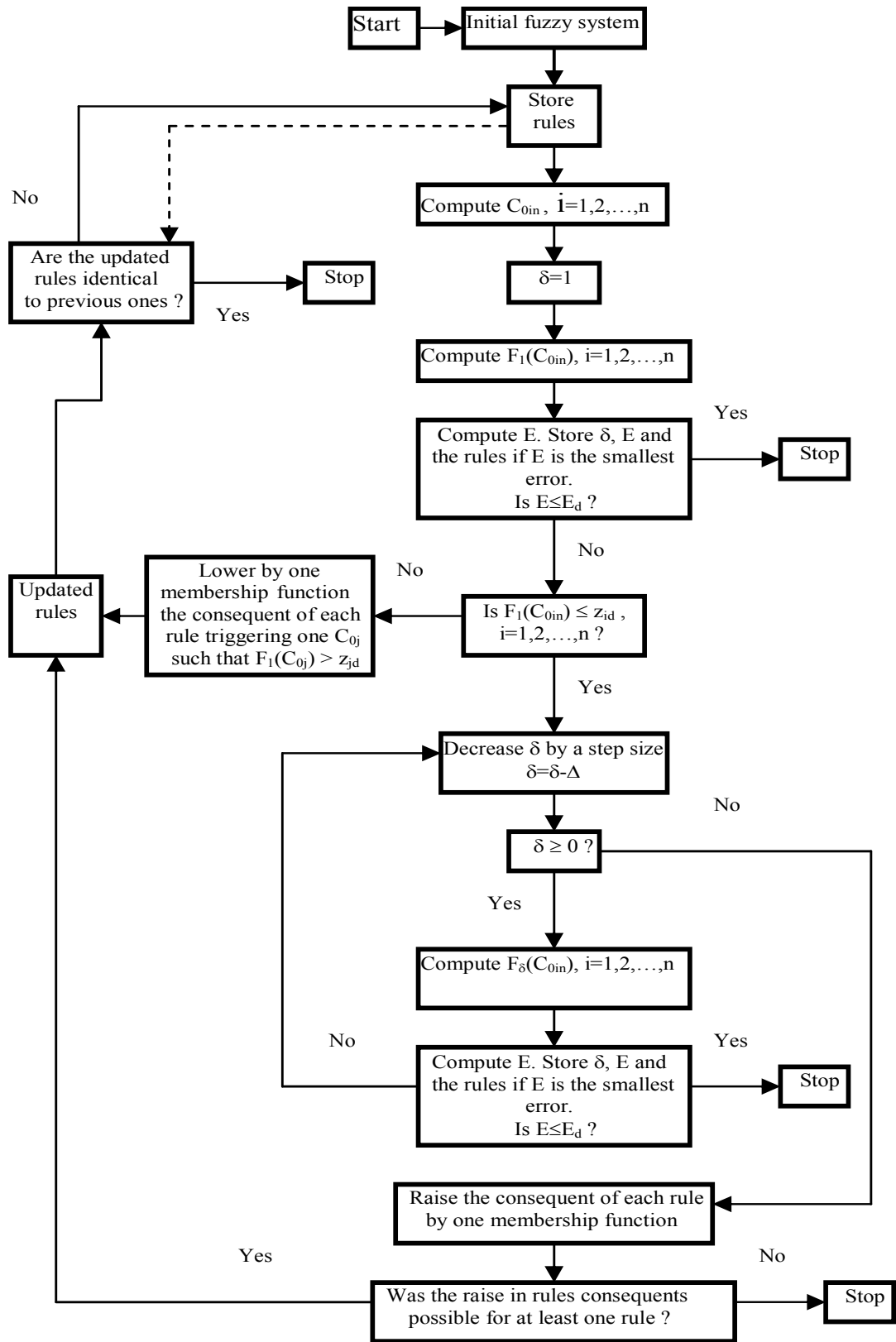


Figure 1. Flow-chart of the data-driven fuzzy controllers design algorithm described in Section 4.

Given the input-output data pairs in the form $(\underline{x}_i, z_{id})$, with $i=1, 2, 3, \dots, n$ and $\underline{x}_i = (x_{1i}, x_{2i}, x_{3i}, \dots, x_{pi})$, where p is the number of input variables, the learning process starts with an initial fuzzy system as specified above. The algorithm (see Figure 1) computes the fuzzy outputs C_{oi} for all $\underline{x}_i$, $i=1, 2, 3, \dots, n$ using the CRI (Equation (1)) and then defuzzifies their normalized versions, C_{oin} , using Equation (2) when $\delta=1$. Here, all the defuzzified values will be equal to the smallest value of the output range. Hence, given that z_{id} are all greater than or equal to this value (this should always be the case), then $F_1[C_{oin}(z)] \leq z_{id}$. For these defuzzified values, the error E is computed using some error function and compared with a desired error value E_d . If $E \leq E_d$, then the learning stops. Otherwise, δ is decreased from 1 to 0 by passing through discrete intermediate values. For each δ , the error is computed and compared with E_d . Note here that the decrease in δ results in an increase in the defuzzified values of the fuzzy outputs. These values are then made closer to the desired outputs. Whether the change in δ has led to the satisfaction of the error goal, that is, $E \leq E_d$ has been achieved for some $\delta \in [0, 1]$, then the learning stops. Otherwise, the algorithm starts again from $\delta=1$ but with new rules.

The new rules are obtained by raising each rule consequent by one fuzzy set. This, however, might lead to a violation of the inequality $F_1[C_{oin}(z)] \leq z_{id}$ for some values of i . If so, the inequality can be reestablished by repeatedly lowering the consequents of the rules, which trigger one fuzzy output whose defuzzified value for $\delta=1$ is greater than its desired counterpart. Once all defuzzified values become again smaller than or equal to the desired ones, δ will be decreased from 1 to 0 and for each δ value the error is computed and compared with E_d . This process is repeated until either the error goal is satisfied or no more raise in the rules consequents is possible or when the raise and lowering of the rules consequents result in a system that has already been obtained. When the learning stops, the algorithm delivers the final fuzzy system with the least error value that can be obtained under the described procedure, the error and the final δ value.

V. PERFORMANCE TESTING USING A TYPICAL NON-LINEAR FUNCTION

In this section, we consider the typical non-linear function given in Equation (3) below to test the performance (Section 2) of the algorithm described in Section 4 and compare with other methods. This function has been considered in research studies to test existing data-driven techniques.

$$z = f(x_1, x_2) = (1 + x_1^{-2} + x_2^{-1.5})^2, 1 \leq x_1, x_2 \leq 5. \quad (3)$$

This function, whose plot is shown in Figure 2, was first considered by Sugeno and Yasukawa [16] and then considered again by Delgado et al. [10] and by Lin et al [7]. Fifty input-output data points, listed in [16], were extracted from Equation (3) and considered in the noted references. These points are also given in Table 1.

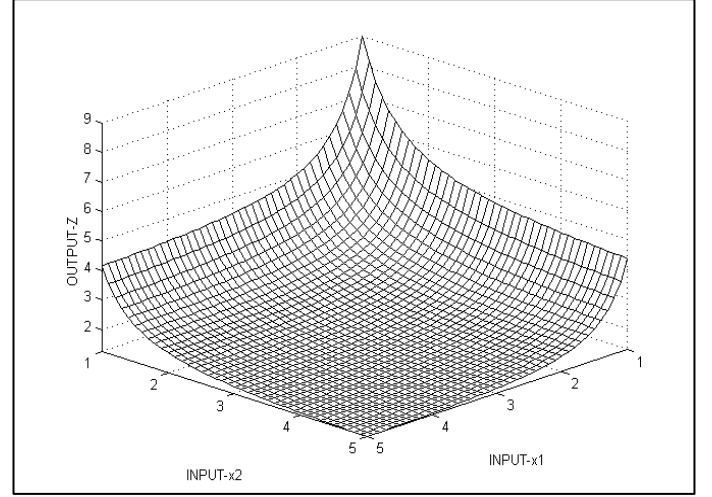


Figure 2. Input-output surface of the non-linear function given in Equation (3).

Table 1. Fifty training data points extracted from the non-linear function in Equation (3).

#	x1	x2	z	#	x1	x2	z
1	1.4	1.8	3.7	26	2	2.06	2.52
2	4.28	4.96	1.31	27	2.71	4.13	1.58
3	1.18	4.29	3.35	28	1.78	1.11	4.71
4	1.96	1.9	2.7	29	3.61	2.27	1.87
5	1.85	1.43	3.52	30	2.24	3.74	1.79
6	3.66	1.6	2.46	31	1.81	3.18	2.2
7	3.64	2.14	1.95	32	4.85	4.66	1.3
8	4.51	1.52	2.51	33	3.41	3.88	1.48
9	3.77	1.45	2.7	34	1.38	2.55	3.14
10	4.84	4.32	1.33	35	2.46	2.12	2.22
11	1.05	2.55	4.63	36	2.66	4.42	1.56
12	4.51	1.37	2.8	37	4.44	4.71	1.32
13	1.84	4.43	1.97	38	3.11	1.06	4.08
14	1.67	2.81	2.47	39	4.47	3.66	1.42
15	2.03	1.88	2.66	40	1.35	1.76	3.91
16	3.62	1.95	2.08	41	1.24	1.41	5.05
17	1.67	2.23	2.75	42	2.81	1.35	1.97
18	3.38	3.7	1.51	43	1.92	4.25	1.92
19	2.83	1.77	2.4	44	4.61	2.68	1.63
20	1.48	4.44	2.44	45	3.04	4.97	1.44
21	3.37	2.13	1.99	46	4.82	3.8	1.39
22	2.84	1.24	3.42	47	2.58	1.97	2.29
23	1.19	1.53	4.99	48	4.14	4.76	1.33
24	4.1	1.71	2.27	49	4.35	3.9	1.4
25	1.65	1.38	3.94	50	2.22	1.35	3.39

In [16], a fuzzy system was determined with 0.318 as a mean-square error (MSE) value. The error was then reduced to 0.01 using position gradient. In [10], which is also a study based on fuzzy clustering, the best-obtained MSE was 0.231. The use of fuzzy partition [7] gave a fuzzy system with 0.351 as an MSE value. This was then reduced to 0.005 by a fuzzy neural network.

As for the algorithm described in this study, the same 50 data points were considered. Three membership functions were adopted over each of the input variables (Figures 3(a) and 3(b)). The error value of 0.216 was obtained with 4 output fuzzy sets as in Figure 3(c). The final obtained rules are as listed below and the final δ value is 1. The input-output surface is shown in Figure 4.

- If x_1 is A_1 and x_2 is B_1 , then z is C_3
- If x_1 is A_1 and x_2 is B_2 , then z is C_3
- If x_1 is A_1 and x_2 is B_3 , then z is C_3
- If x_1 is A_2 and x_2 is B_1 , then z is C_3
- If x_1 is A_2 and x_2 is B_2 , then z is C_2
- If x_1 is A_2 and x_2 is B_3 , then z is C_2
- If x_1 is A_3 and x_2 is B_1 , then z is C_3
- If x_1 is A_3 and x_2 is B_2 , then z is C_2
- If x_1 is A_3 and x_2 is B_3 , then z is C_2 (4)

The non-linear function in Equation (3) is considered again to test the performance of the algorithm in complete but noisy data type. Since the methods noted in [7,10,16] considered only complete and noise-free data and the resulting error values, the comparison will be done with ANFIS. It is powerful enough to bring out the relative powers of the algorithm presented in Section 4.

ANFIS, introduced by Jang [13], is available under MATLAB and based on a combination of the least-squares and gradient-descent methods. For the comparison to be meaningful, the same number and type of the input membership functions will be considered in ANFIS. Also, due to the fact that the ANFIS results depend on the number of epochs and to a lesser degree on the initial step-size, the presented results will be the best ones we obtained after attempting different combinations of epoch-step-size values. The algorithm will also be examined for generalization and noise insensitivity plus generalization. That is, in the cases of noise-free but incomplete and noisy and incomplete data types.

Let us first consider the use of the same 50 data pairs (Table 1) in ANFIS. A fuzzy system with MSE equal to 0.0303 was obtained under 100 epochs and 0.00001 as initial step-size. The input-output surface is shown in Figure 5. Although the error value is smaller than 0.216 obtained in our approach, the comparison of Figures 4 and 5 reveals that the described algorithm has a better representation of the shape of the non-linear function.

Concerning complete and noisy data, 3 stages of modification of output values in the 50 points listed in Table 1 were performed. In each stage, 4 output values were modified to result in 4 input-output pairs, denoted as set, not satisfying the function in Equation (3). First, set 1 was used in addition to the remaining 46 noise-free data pairs. In stage 2, sets 1 and 2 were used in addition to the remaining 42 noise-free data. In stage 3, sets 1, 2 and 3 were used in addition to 38 noiseless data. The fuzzy systems obtained

by ANFIS had the surfaces shown in Figures 6(a), (b) and (c) respectively.

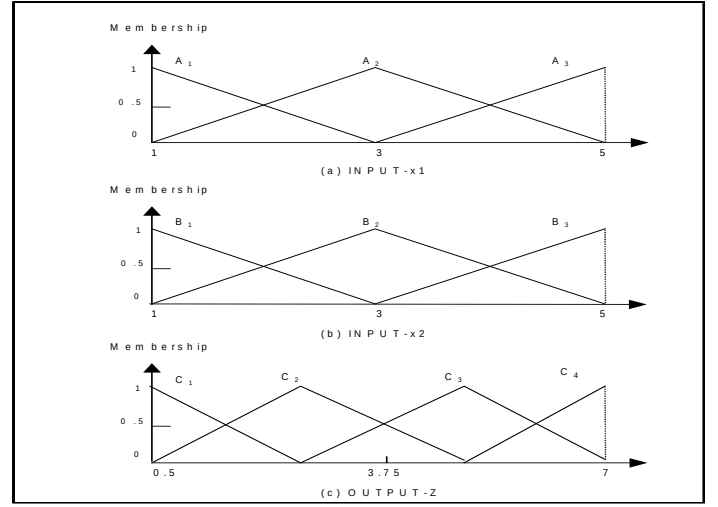


Figure 3. Membership functions assigned over the input and output variables of the non-linear function given in Equation (3).

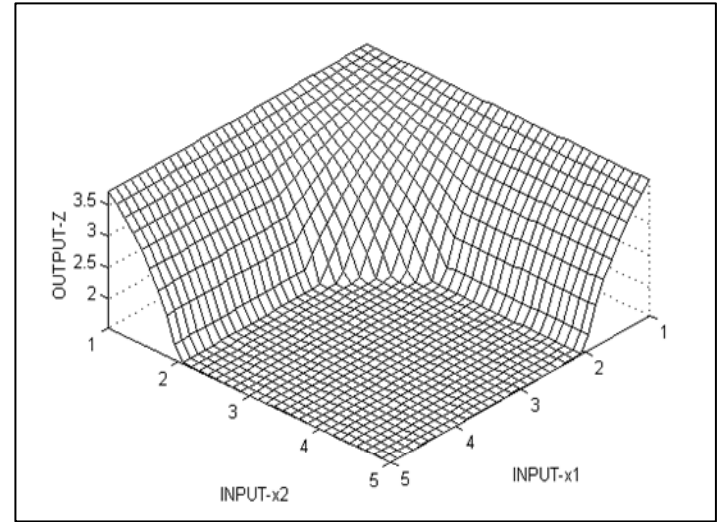


Figure 4. Input-output surface of the fuzzy system given in Equation (4) and representing the function given in Equation (3).

All the above-noted 3 cases were used in the algorithm described in Section 4. The resulting fuzzy system was always as in Equation (4) and with $\delta=1$. The input-output surface is just the one shown in Figure 4. The comparison of Figure 4 with Figures 6(a), (b) and (c) reveals that the presented approach has a better noise insensitivity than ANFIS. This result is also supported by the error values at the noisy points. In terms of the error obtained by considering the underlying 50 noise-free data (Table 1), ANFIS gave 0.2292, 0.2925 and 0.3217 respectively. Comparison of these error values with 0.216 (obtained using the presented algorithm), the noise insensitivity result and also Figure 4 with Figures 6(a), (b) and (c), shows that our approach has a performance preference over ANFIS when the learning is based on noisy data (see Section 2).

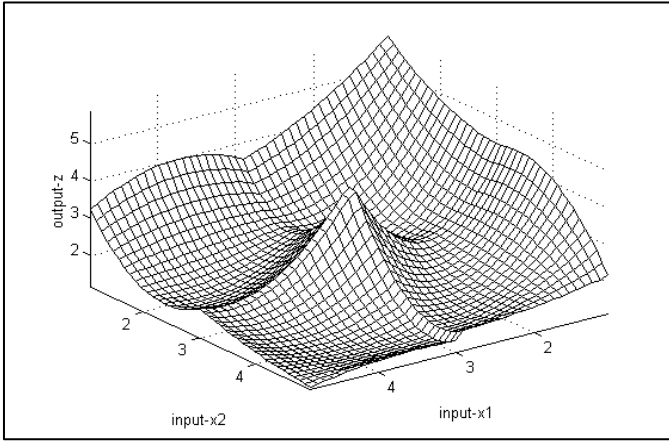


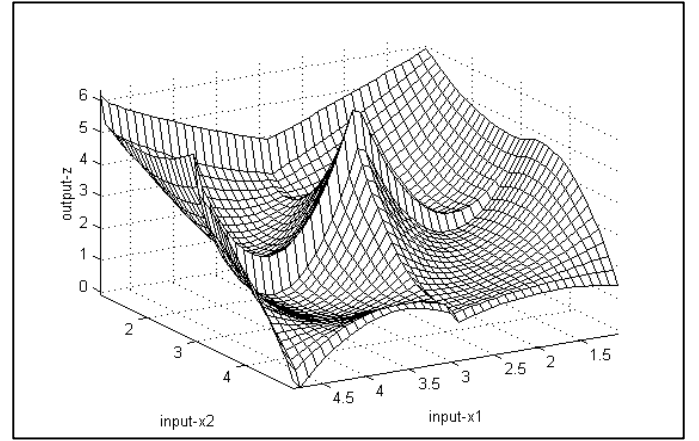
Figure 5. Input-output surface of the fuzzy system obtained from ANFIS using the 50 data pairs in Table 1.

In terms of testing the generalization capability of the presented approach, sets of data points from among those listed in Table 1 and located in specific boundary regions of the input space were excluded in succession. The remaining data were entered into the presented algorithm. First, data pairs such that $1 < x_1 < 2.5$ and $3.5 < x_2 < 5$ were eliminated. This resulted in the exclusion of 5 data points. Second, data points such that $1 < x_1 < 3$ and $3 < x_2 < 5$ (8 points) were excluded. In both cases, the final fuzzy system turned out to be as in Equation (4) and $\delta=1$. The error values were respectively 0.2355 and 0.2468. In both cases, the error value 0.216 still holds for the original 50 points in Table 1.

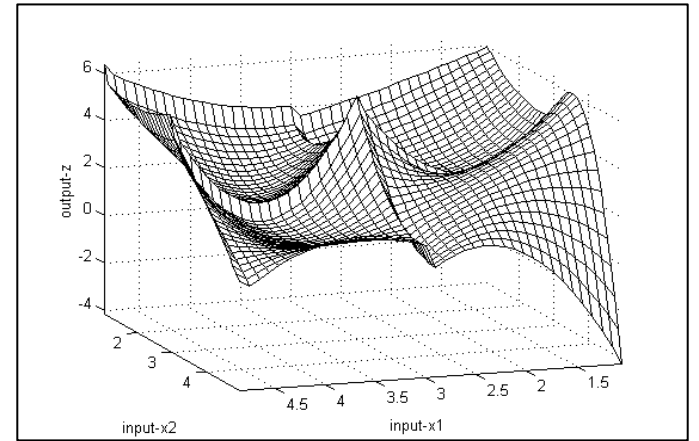
The presented algorithm was also tested for its ability to combat noise and generalize simultaneously. The data elimination process described in the preceding paragraph was again considered and noisy data from among the above-mentioned 12 points were introduced. The introduced noisy points were those which survived elimination. Hence, 9 noisy points (set 1, 1 point from set 2 and set 3) were used among the 45 data pairs, which resulted from the first data exclusion. Also, 8 noisy points (set 1 and set 3) were used among the 42 data pairs, which resulted from the second data elimination process. In both cases, the algorithm returned the final fuzzy system expressed in Equation (4) with $\delta=1$. The input-output surface is therefore as shown in Figure 4. The proposed fuzzy system modeling approach is therefore able to combat noise and generalize simultaneously.

VI. ROBOT NAVIGATION CASE

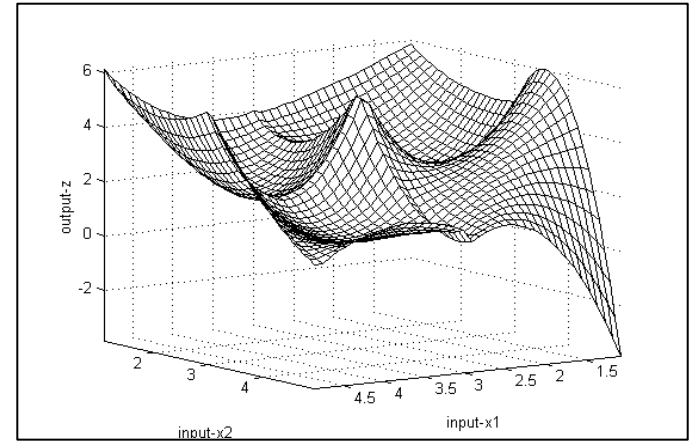
In dealing with the motion-planning problem of a mobile robot among existing obstacles, different classical approaches have been developed. Within these approaches, we state the path velocity decomposition [34,35], incremental planning [36], relative velocity paradigm [37], and potential field [38]. Soft-computing techniques, employing various learning methods, have also been used to improve the performance of conventional controllers [39,40]. Each of the above noted methods is either computationally extensive or capable of solving only a particular type of problems or both.



(a)



(b)



(c)

Figure 6. Input-output surfaces of the fuzzy systems obtained from ANFIS using: (a) 46 noise-free data pairs and 4 noisy ones (b) 42 noise-free data and 8 noisy points and (c) 38 noise-free data and 12 noisy points.

In order to reduce the computational burden and provide a more natural solution for the dynamic motion-planning (DMP) problem, fuzzy approaches, with emphasis on user-defined rules and collision-free paths, have been suggested [41-43]. Recently, a more advanced fuzzy-genetic-algorithm approach has been

devised [44]. The emphasis has been not only on obtaining collision-free paths, but also on the optimization of travel time or path between the start and target points of the robot. Genetic algorithms have, therefore, been used to come up with an optimal or near optimal fuzzy rule-base off-line by employing a number of user-defined scenarios. Although the noted fuzzy-genetic approach provided good testing results on scenarios some of which were used in training and others were not, it had its limitations. A different set of rules needed to be determined for every specific number of moving obstacles.

The approach presented in this section considers the off-line derivation of a general fuzzy rule-base; that is, a base that can be used on-line by the robot independently of the number of moving obstacles [45]. This is achieved using the data-driven learning algorithm presented in Section 4 and by devising a method for the derivation of the training data based on the general setting of the DMP problem and not on specific scenarios. Collision-free paths and reduction of travel time are still within the goals considered in the derivation of the fuzzy logic controller (FLC). Furthermore, the noise insensitivity and generalization capability of the FLC construction algorithm are emphasized again here and tested in this practical control case. Comparison of the results with those obtained by the fuzzy-genetic one and ANFIS is also done.

The robot needs to move from a start point S to a target point G located in some quadrant where the moving obstacles exist. The purpose is to find an obstacle-free path, which takes the robot from S to G with minimum time. A fuzzy controller represented by a set of fuzzy inference rules is to be constructed to achieve this objective.

The robot moves incrementally from one point to another in accordance with time steps, each of duration ΔT , and at the end of each step it needs to decide on the movement direction. Due to the problem objective, once the robot is at some point it needs to consider moving in a straight line towards the target point unless the information collected about the moving obstacles tells otherwise due to a possible collision. Hence, the information that needs to be obtained has to relate, in principle, to the position of each obstacle and its velocity relative to the robot position; i.e., the obstacle velocity vector. But, since the robot knows the position of each obstacle at every time step, an alternative to the use of the relative velocity can be the present and predicted positions of each obstacle. The predicted position can be computed based on the obstacle present and previous positions. $P_{\text{predicted}}$ is assumed the linearly extrapolated position of each obstacle from its present position P_{present} along the line formed by joining P_{present} and P_{previous} . Thus,

$$P_{\text{predicted}} = P_{\text{present}} + (P_{\text{present}} - P_{\text{previous}})$$

But, to process all this information by the robot controller is difficult. The procedure that can be applied here, and which leads to a simplification of the controller structure, consists of using the collected information to determine the “nearest obstacle forward” (NOF) to the robot [44]. Then, only the information related to this obstacle is used by the FLC to provide decisions. The NOF is the obstacle located in front of the robot and with velocity vector pointing towards the line joining the robot position to the target point. In this way it needs to constitute the most possible collision danger relative to other obstacles whether the robot chooses to move straight to the target (Figure 7). The NOF can equivalently

be identified using the present and predicted positions of each obstacle.

Therefore, what need to be used are the present and predicted positions of the NOF. The position has two components; angle and distance. The angle is the one between the line joining the target G to the robot position, denoted by R in Figure 7, and the line between the robot and the NOF. The distance is the one between the robot and the NOF. The FLC output is the deviation angle between the target-robot line and the new direction of robot movement, denoted by line RD (see also Figure 7). Based on the noted information the robot will be able to know whether the NOF will get close to or cross the line segment joining the present position of the robot and the point it reaches after ΔT time if it moves straight to the target. This knowledge is in fact necessary for the determination of the angle of deviation.

But, to include all these variables in the conditions of the FLC will complicate its structure. It will also make the derivation of the input-output data points needed for the construction of the inference rules a difficult task. To make things simpler while maintaining the practicality of the problem, a constraint (constraint #4 below), which is not too restrictive, is considered in addition to other ones implied by the aforementioned problem description and adopted in [44].

1. The robot is considered to be a single point.
2. Each obstacle is represented by its bounding circle.
3. The speed of each obstacle is constant with a fixed direction between its previous, present and predicted positions.
4. The distance traveled by the NOF in ΔT time is comparable to its diameter.

Of course, constraint 3 presupposes that the obstacles do not collide while moving. Also, constraint 4, with the problem configuration as depicted in Figure 8 and its use in the determination of the input-output data (see below), will reduce the number of FLC input variables to 2; predicted angle and distance. The present position of the NOF is still accounted for but not used explicitly in the controller conditions.

Figure 8 considers a quadrant filled by side-to-side obstacles each of which may constitute the predicted position of the NOF. Suppose that the robot is in position R (present position) and the NOF predicted position is in (A_{22}, B_{21}) . The robot initial intension is to move straight to G (no deviation) if collision is deemed impossible. Otherwise, an angle of deviation needs to be determined. Due to constraint 4, the present position of the NOF could be any of the neighboring obstacles such that the distance between the center of each of these obstacles and the center of (A_{22}, B_{21}) is approximately equal to the obstacle diameter. For the purpose of explaining how the deviation angle is to be determined for every possible pair of predicted angle and distance of the NOF, a rough representation of 8 critical neighboring obstacles is considered. These are: (A_{21}, B_{20}) , (A_{22}, B_{20}) , (A_{23}, B_{20}) , (A_{21}, B_{21}) , (A_{23}, B_{21}) , (A_{21}, B_{22}) , (A_{22}, B_{22}) and (A_{23}, B_{22}) . If the segment between the present position of the robot and the point it reaches after ΔT time, if it moves straight to G, penetrates the square formed by the outer tangent lines to the noted 8 obstacles, a deviation from the straight line between the robot and target point is required. Otherwise, no deviation is necessary. The amount of deviation is to be specified based on having the robot move in a direction that is just sufficient to avoid hitting not only the predicted obstacle position, but also any possible present position

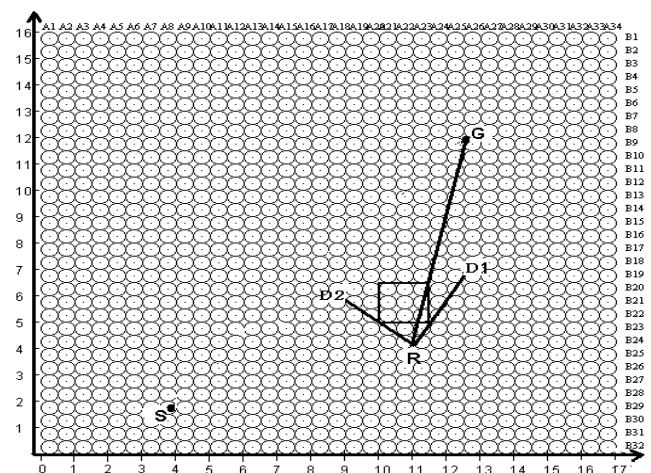


Figure 8. A general configuration of the DMP problem used in the data derivation.

Ang	Dist	Dev	Ang	Dist	Dev	Ang	Dist	Dev	Ang	Dist	Dev
-90	0.3	33	90	1.8	0	45	1.2	0	22	1	-33
-90	0.5	15	90	2	0	45	1.4	0	22	1.2	-25
-90	0.7	0	90	4	0	45	1.8	0	22	1.4	-13
-90	1	0	90	5	0	45	2	0	22	1.8	0
-90	1.2	0	90	15	0	45	2.4	0	22	2	0
-90	1.4	0	-45	0.3	90	45	20	0	22	3	0
-90	1.5	0	-45	0.5	90	-22	0.3	90	22	10	0
-90	1.8	0	-45	0.7	62	-22	0.5	90	22	20	0
-90	2	0	-45	1	15	-22	0.7	90	0	0.3	90
-90	4	0	-45	1.2	0	-22	1	33	0	0.5	90
-90	5	0	-45	1.4	0	-22	1.2	25	0	0.7	90
-90	15	0	-45	1.8	0	-22	1.4	13	0	1	57
-90	24	0	-45	2	0	-22	1.8	0	0	1.2	45
90	0.3	-33	-45	2.4	0	-22	3	0	0	1.4	37
90	0.5	-15	-45	5	0	-22	10	0	0	1.8	25
90	0.7	0	-45	20	0	-22	20	0	0	2.4	15
90	1	0	45	0.3	-90	-22	24	0	0	3	0
90	1.2	0	45	0.5	-90	22	0.3	-90	0	4	0
90	1.4	0	45	0.7	-62	22	0.5	-90	0	10	0
90	1.5	0	45	1	-15	22	0.7	-90	0	20	0

Table 2. Input-output data pairs obtained using the method described in Section 6.

A n g l e D i s t a n c e	A 1	A 2	A 3	A 4	A 5	A 6	A 7
D 1	V 7	V 9	V 9	V 9	V 1	V 1	V 4
D 2	V 6	V 9	V 9	V 9	V 1	V 1	V 4
D 3	V 5	V 8	V 9	V 9	V 1	V 2	V 5
D 4	V 5	V 6	V 7	V 8	V 4	V 4	V 5
D 5	V 5	V 5	V 6	V 7	V 4	V 5	V 5
D 6	V 5	V 5	V 6	V 7	V 4	V 5	V 5
D 7	V 5	V 5	V 5	V 6	V 5	V 5	V 5
D 8	V 5	V 5	V 5	V 5	V 5	V 5	V 5

Table 3. Final fuzzy system obtained by learning.

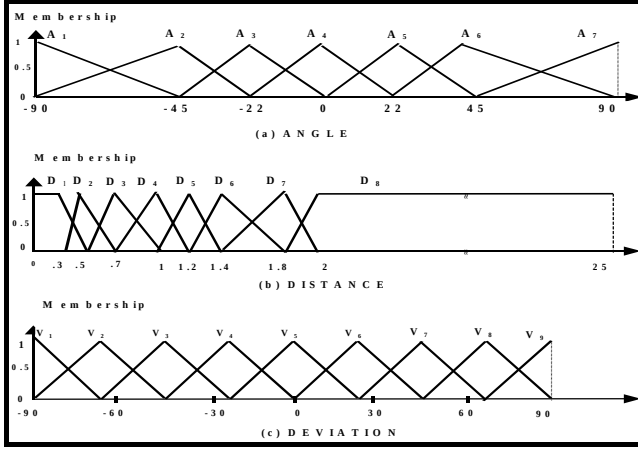


Figure 9. Input and output membership functions used in learning.

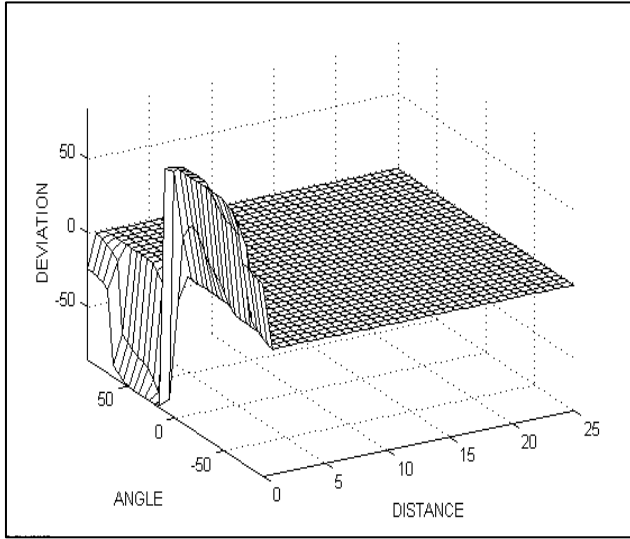


Figure 10. Control surface of the FLC constructed using the algorithm in Section 4 and the data in Table 2. The angle and deviation are in degrees and the distance is in meters.

of the NOF; i.e., any of the above-described neighboring obstacles, which are guaranteed to reside inside the previously-noted square. Among the two movement directions RD1 and RD2, which lead to the avoidance of the obstacles positions, the one with the smallest deviation angle; i.e., RD1 is chosen. This serves the travel path reduction objective.

Now, based on the problem configuration in Figure 8 and the described general approach for the determination of the necessary deviation for every possible pair of predicted distance and angle of the NOF, various locations of the NOF within the noted quadrant were considered and accordingly input-output data were derived. The derived data points are shown in Table 2. These are obtained based on obstacle diameter equal to 0.5 meters and robot traveled distance in ΔT time equal to 2 meters.

The data points in Table 2 were used in the fuzzy learning algorithm in Section 4 and a set of inference rules (Table 3) was obtained using the input and output membership functions (MF's) shown in Figure 9. The ranges of the distance, angle and deviation as considered are from 0 to 25 meters, -90 to 90 degrees and -90

to 90 degrees respectively. These ranges are considered to account for all possible values. The control surface of the FLC, whose rules are shown in Table 3, is given in Figure 10, the root mean square error is 4.679 and the δ parameter value is 0.5.

The obtained FLC is tested on various scenarios containing different numbers of obstacles. The cases of 3, 5, and 2 cases of 8 obstacles are considered and the simulation results are presented in Figure 11. In all the cases, the robot travels from point S to point G without hitting any of the obstacles. Also, the traveled paths are optimal in the sense that the deviations which took place at the end of every time step are in most cases just as necessary in order for the robot to remain as close as possible to the robot-destination direct path while not colliding with the obstacles. Moreover, two of these scenarios (Figures 11(a) and 11(d)) were presented in [44] and had obstacles with distinct diameters. Some had diameters close to the one considered in this study, and others with larger diameters. Despite this, the robot path chosen by the constructed FLC does not hit any of the obstacles. This shows that the constructed FLC can work properly for obstacles whose diameter values differ from the one used in the data derivation. Of course, a significant increase in the diameters would make the chances of having the robot hitting the obstacles higher.

Table 4 shows the distance ratio (traveled distance/direct distance) using the presented data-driven fuzzy approach and the fuzzy-genetic one. The ratio in the fuzzy methodology for the case of 3 obstacles is a bit higher than that obtained in [44]. Thus, given that the robot speed in moving from one point to another in accordance with the previously-noted incremental time steps is the same in both approaches, then slightly higher time duration is required in our approach for the robot to reach destination. This result, however, is quite acceptable given the fact that the presented methodology is general in the sense that it can be applied independently of the number of moving obstacles.

The data points in Table 2 are used to construct a Takagi-Sugeno type fuzzy system by applying ANFIS [13] and compare the results with those obtained above using the algorithm presented in Section 4. Although the data in Table 2 were determined using measuring instruments, and they are thus, noisy, care has been taken to make these data points as accurate as possible. Due to this and the fact that the actual values of the data are unknown, we consider the data in Table 2 as approximately noise-free. To be surer of the existence of noise in the data, arbitrary small modifications will be introduced later to some angles of deviations to make the corresponding data pairs noisy. Under these circumstances, a comparison between the results of the algorithm in Section 4 and ANFIS will again be done. Testing the algorithm for generalization and noise insensitivity plus generalization will also be investigated.

The ANFIS obtained FLC using the data in Table 2 and 8 triangular membership functions over each input variable has the input-output surface shown in Figure 12 and RMSE equal to 5.438. The surface in Figure 10 is much more consistent than Figure 12 with our expectations of the control surface as configured using the data in Table 2. This is especially true in the range of small distances where most of robot deviations are necessary.

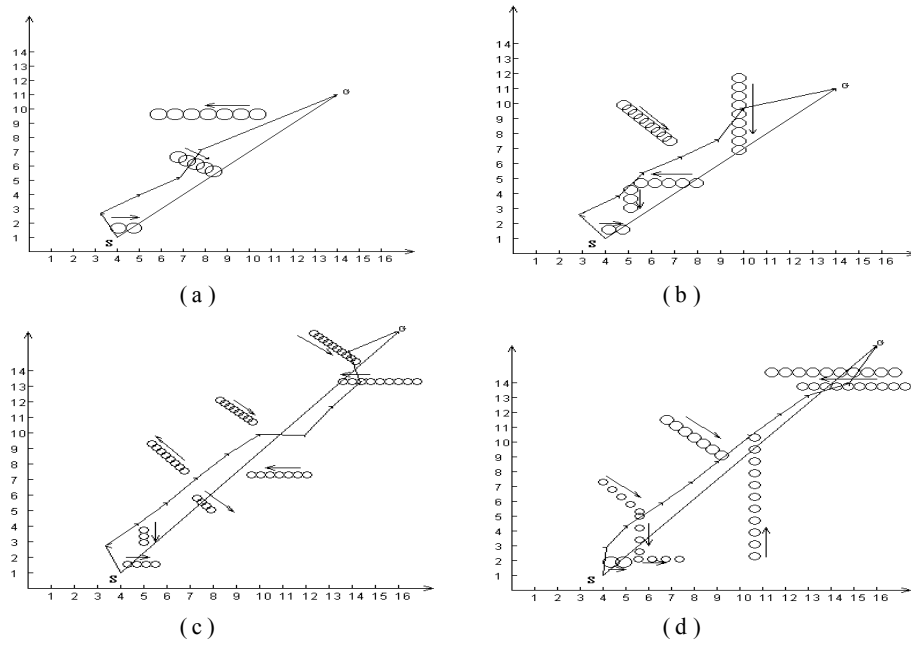


Figure 11. Paths traveled by the robot in 4 scenarios using the FLC in Table 3: (a) 3 obstacles, (b) 5 obstacles and (c) and (d) have 8 obstacles each.

OBSTACLES	3	5	8	8
DIRECT DISTANCE (m)	14	14	20	20
TRAVELED DISTANCE (m)	15	15.8	23	20.9
RATIO (OUR APPROACH)	1.07	1.286	1.15	1.045
RATIO (GENETIC)	1.046			1.05

Table 4. Traveled distance and ratios for the presented approach and the fuzzy-genetic one.

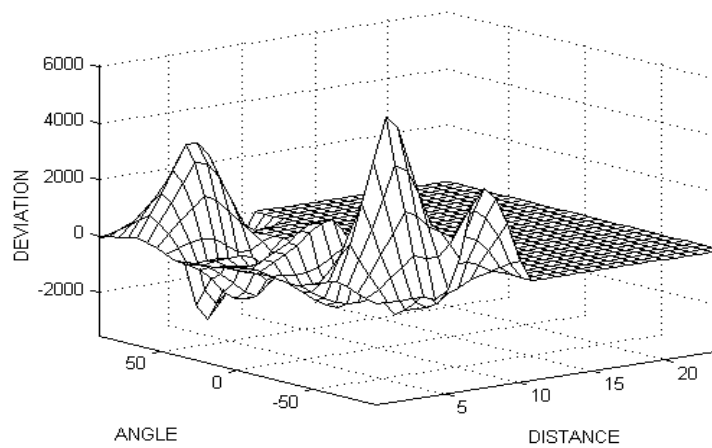


Figure 12. Control Surface of the FLC constructed by ANFIS using the data in Table 2. The angle and deviation are in degrees and the distance is in meters.

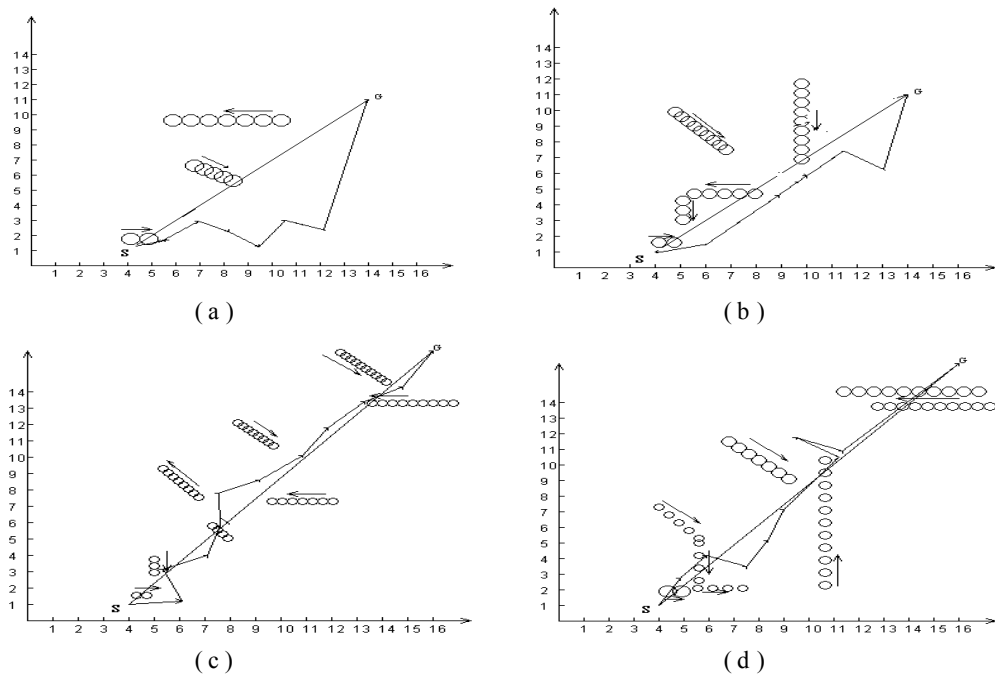


Figure 13. Paths traveled by the robot in 4 scenarios using the FLC obtained by ANFIS:
(a) 3 obstacles, (b) 5 obstacles and (c) and (d) have 8 obstacles each.

Ang	Dist	Dev	Ang	Dist	Dev	Ang	Dist	Dev	Ang	Dist	Dev
-90	0.3	33	90	1.8	0	45	1.2	0	22	1	-31
-90	0.5	15	90	2	0	45	1.4	2	22	1.2	-22
-90	0.7	0	90	4	0	45	1.8	0	22	1.4	-13
-90	1	0	90	5	0	45	2	3	22	1.8	0
-90	1.2	0	90	15	0	45	2.4	0	22	2	1
-90	1.4	0	-45	0.3	86	45	20	2	22	3	0
-90	1.5	0	-45	0.5	87	-22	0.3	90	22	10	2
-90	1.8	0	-45	0.7	58	-22	0.5	87	22	20	0
-90	2	0	-45	1	13	-22	0.7	86	0	0.3	87
-90	4	0	-45	1.2	-2	-22	1	33	0	0.5	90
-90	5	0	-45	1.4	0	-22	1.2	21	0	0.7	88
-90	15	0	-45	1.8	-2	-22	1.4	13	0	1	59
-90	24	0	-45	2	0	-22	1.8	0	0	1.2	45
90	0.3	-33	-45	2.4	-3	-22	3	-3	0	1.4	35
90	0.5	-15	-45	5	0	-22	10	0	0	1.8	24
90	0.7	0	-45	20	0	-22	20	-2	0	2.4	14
90	1	0	45	0.3	-86	-22	24	0	0	3	0
90	1.2	0	45	0.5	-86	22	0.3	-87	0	4	2
90	1.4	0	45	0.7	-58	22	0.5	-90	0	10	-1
90	1.5	0	45	1	-13	22	0.7	-87	0	20	0

Table 5. Input-output data pairs obtained by modifying the deviation in 33 data pairs in Table 3.

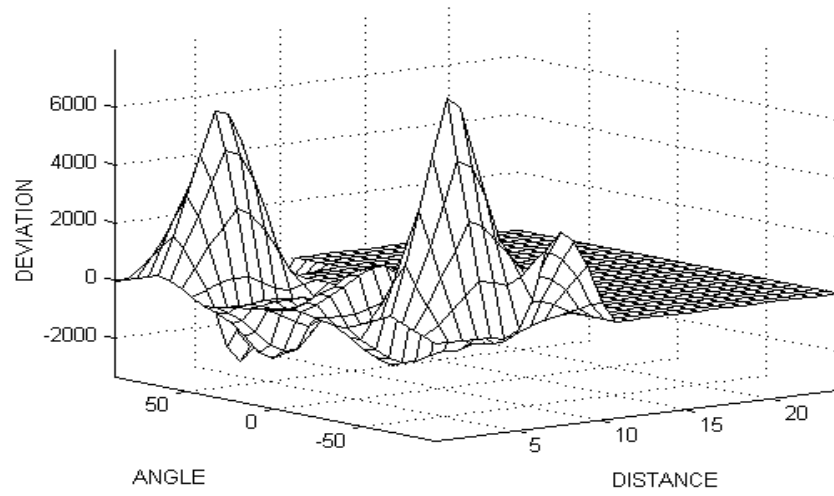


Figure 14. Control surface of the FLC constructed by ANFIS using the data in Table 5. The angle and deviation are in degrees and the distance is in meters.

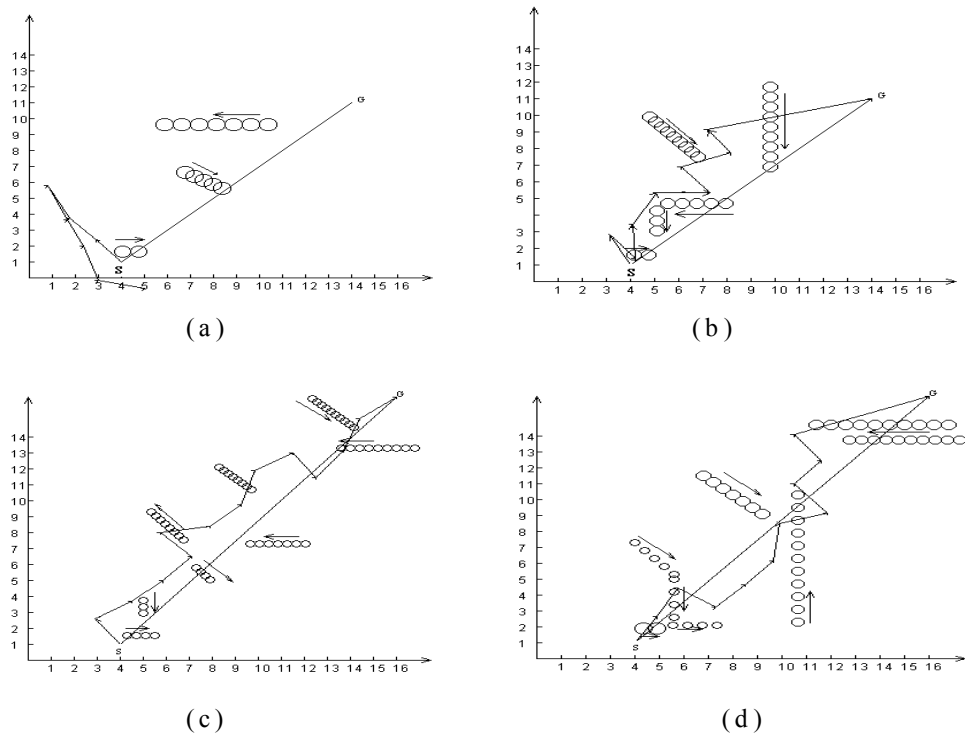


Figure 15. Paths traveled by the robot in 4 scenarios using the FLC obtained by ANFIS with the data in Table 5: (a) 3 obstacles, (b) 5 obstacles and (c) and (d) have 8 obstacles each.

The testing of the ANFIS obtained fuzzy system on the 4 scenarios in Figure 11 gave the robot trajectories shown in Figure 13. The ANFIS obtained trajectories lengths were respectively 20.2, 17.3, 23.4 and 25.9 meters for cases (a), (b), (c) and (d) in Figure 13. These are therefore larger than the paths lengths in Figure 11, which were obtained using the fuzzy algorithm (Table 4). Also, ANFIS gave two collision cases; one in (c) and another in (d).

Now, we consider the data points in Table 5. These were obtained from Table 2 by randomly introducing between 1 to 4 degrees modifications to 33 of the deviation angles and in such a manner to adequately cover the input space. The use of the data in Table 5 in the algorithm described in Section 4 gave the same FLC as the one in Table 3 with the same parameter value. Thus, the control surface is the one shown in Figure 10. Hence, the robot paths shown in Figure 11 remain unchanged. The ANFIS obtained system, however, turned out to be different from the one obtained with the use of the original data (Table 2). It had the surface shown in Figure 14. Figure 15 shows the testing of the ANFIS given system on the 4 scenarios in Figure 11. Regarding the robot paths shown in Figure 15, they have the following respective lengths for cases (b), (c) and (d): 23, 26.1, and 26 meters. In case (a), the robot is not able to reach destination and in case (d) one hit occurs. ANFIS, therefore gave here longer trajectories than those in Figure 13 and in Figure 11.

Furthermore, the fuzzy controller construction algorithm was tested for generalization by eliminating 8 data points from those listed in Table 2. The eliminated points were those with input pairs located in the input region for $0^\circ < \text{Angle} \leq 90^\circ$ and 2 meters $< \text{Distance} \leq 25$ meters. The learning was therefore based on the remaining 72 data. The fuzzy system whose rules are given in Table 3 was again obtained with the same parameter value. The control surface is, thus, as given in Figure 10. In addition, the same 8 points were eliminated from the data listed in Table 5. The learning based on the remaining 72 data points also gave the fuzzy system in Table 3 and the same parameter value.

The presented results validate the fact that the algorithm is capable of combating noise, which is usually present in the data, generalizing to regions of missing data and combating noise and generalizing simultaneously. The comparison of Figures 12 and 14 and also Figures 13 and 15 shows that the fuzzy systems constructed by ANFIS are noise sensitive. They are also sensitive to data exclusion and do not present a good generalization capability. This was concluded by looking at the ANFIS obtained surfaces using the above-noted 72 data points remaining after the exclusion of the mentioned 8 points from Tables 2 and 5. These surfaces, in fact, turned out to be different and no better than the ones presented in Figures 12 and 14 and the testing on the 4 scenarios in Figure 11 did not lead to any improvement over those obtained in Figures 13 and 15.

VII. SUMMARY AND CONCLUSIONS

In this study, performance criteria, which need to be considered to test and compare data-driven fuzzy system modeling algorithms using non-linear control functions, have been defined based on a practical perspective and on important aspects of intelligent human thinking represented by approximate reasoning and generalization. Hence, priority has been given to the criteria accounting for noisy and incomplete training data. Despite the extreme importance of these performance criteria,

they have not been spelled out, discussed, validated nor tested as is done in this study. The following summarized results and conclusions are to be noted.

When the data pairs are noise-free and complete, the proposed algorithm provided lower error values than the methods based on clustering and fuzzy partitions. Further, the fuzzy partition neural-network method, clustering-position-gradient and ANFIS gave lower error values than the presented algorithm. In the example given in Section 5 and the robot navigation case, however, ANFIS gave an inferior function-shape representation.

Furthermore, when the data are noisy, which is the practical case (Section 2), it has been shown in the non-linear function case and robot navigation that the proposed fuzzy learning algorithm (Section 4) has a performance preference over ANFIS. The proposed algorithm is also capable of generalization. It extrapolates very reliably to regions of missing data. It is also able to combat noise and generalize simultaneously. In addition, the algorithm provides readable fuzzy controllers. That is, controllers which can be interpreted linguistically in a simple manner. The T-S type fuzzy models, as was explained in Section 3, cannot furnish this linguistic aspect.

The fuzzy learning algorithm in Section 4 has also been shown capable of providing fuzzy if-then rules that can be represented in the form of a rule-table, it has only one parameter to be tuned and the setting of initial rules can easily be done. The algorithm also permits thinking tolerant to imprecision. This can be concluded from the learning procedure, which is based on error reduction rather than error minimization and customized setting of the error threshold to reflect the level of precision needed in a particular situation. It can also be concluded from the resulting noise insensitivity and generalization capability of the algorithm. It is worth noting here as well that the learning procedure in the introduced algorithm is independent of the form of the error function and also of the shape of the fuzzy system membership functions.

Further strengthening of the drawn conclusions should later on come up through the application of the algorithm and the other design methods to more non-linear functions and practical control cases while accounting for the criteria defined in Section 2. This can be done by programming the referenced methods, especially those that have recently accounted for the issues of noise and generalization, and testing them in different types of data.

REFERENCES

- [1] Takagi T., Sugeno M., "Fuzzy identification of systems and its application to modeling and control," *IEEE Trans. Systems, Man and Cybernetics*, SMC-15, 1, pp.116-132, 1985.
- [2] Horikawa S., Furuhashi T., Uchikawa Y., "On fuzzy modeling using fuzzy neural networks with the backpropagation algorithm," *IEEE Trans. Neural Networks*, Vol. 3, No. 5, pp.801-806, 1992.
- [3] Jang JSR., "Self-learning fuzzy controllers based on temporal back-propagation," *IEEE Trans. Neural Networks*, Vol. 3, No. 5, pp. 714-723, 1992.
- [4] Nomura H., Hayashi I., Wakami N., "A learning method of fuzzy inference rules by descent method," *IEEE Int'l Conf. on Fuzzy Systems*, San Diego, California, March 18-22, 1992, pp. 203-210.

- [5] Shi Y., Mizumoto M., "Some considerations on conventional neuro-fuzzy learning algorithms," *Fuzzy Sets and Systems*, Vol. 112, pp. 51-63, 2000.
- [6] Shi Y., Mizumoto M., "A new approach of neuro-fuzzy learning algorithm for tuning fuzzy rules," *Fuzzy Sets and Systems*, Vol. 112, pp. 99-116, 2000.
- [7] Lin Y., Cunningham III G.A., Coggeshall S.V., "Using fuzzy partitions to create fuzzy systems from input-output data and set the initial weights in a fuzzy neural network," *IEEE Trans. Fuzzy Systems*, Vol. 5, No. 4, pp. 614-621, 1997.
- [8] Klawonn F., Kruse R., "Constructing a fuzzy controller from data," *Fuzzy Sets and Systems*, Vol. 85, pp.177-193, 1997.
- [9] Tarn Y.S., Yeh Z.M., Nian C.Y., "Genetic synthesis of fuzzy logic controllers in turning," *Fuzzy Sets and Systems*, Vol. 83, pp. 301-310, 1996.
- [10] Delgado M., Skarmita A.F.G., Martin F., "A fuzzy clustering-based rapid prototyping for fuzzy rule-based modeling," *IEEE Trans. Fuzzy Systems*, Vol. 5, No. 2, pp. 223-233, 1997.
- [11] Carse B., Fogarty T.C., Munro A., "Evolving fuzzy rule-based controllers using genetic algorithms," *Fuzzy Sets and Systems*, Vol. 80, pp. 273-293, 1996.
- [12] Chen J.Q., Xi Y.G., Zhang Z.J., "A clustering algorithm for fuzzy model identification," *Fuzzy Sets and Systems*, Vol. 98, pp. 319-329, 1998.
- [13] Jang J.S.R., "ANFIS: Adaptive-network-based fuzzy inference system," *IEEE Trans. Systems, Man and Cybernetics*, Vol. 23, No. 3, pp. 665-685, 1993.
- [14] Jang J.S.R., Sun C.T., "Neuro-fuzzy modeling and control," *Proceedings of the IEEE*, Vol. 83, No. 3, pp. 378-406, 1995.
- [15] Shi Y., Mizumoto M., "An improvement of neuro-fuzzy learning algorithm for tuning fuzzy rules," *Fuzzy Sets and Systems*, Vol. 118, pp. 339-350, 2001.
- [16] Sugeno M., Yasukawa T., "A fuzzy-logic-based approach to qualitative modeling," *IEEE Trans. on Fuzzy Systems*, Vol. 1, No. 1, pp. 7-31, 1993.
- [17] Siarry P., Guely F., "A genetic algorithm for optimizing Takagi-Sugeno fuzzy rule-bases," *Fuzzy Sets and Systems*, Vol. 99, pp. 37-47, 1998.
- [18] Papadakis S.E., Theocharis J.B., "A GA-based fuzzy modeling approach for generating TSK models," *Fuzzy Sets and Systems*, vol. 131, pp. 121-152, 2002.
- [19] Lee C.W., Shin Y.C., "Construction of fuzzy systems using least-squares method and genetic algorithm," *Fuzzy Sets and Systems*, Vol. 137, pp. 297-323, 2003.
- [20] Saade J.J. "A defuzzification based new algorithm for the design of Mamdani-type fuzzy controllers," *Mathware and Soft Computing*, Vol. 7, pp. 159-173, 2000.
- [21] Buckley J.J., Hayashi Y., "Numerical relationships between neural networks, continuous functions, and fuzzy systems," *Fuzzy Sets and Systems*, Vol. 60, pp.1-8, 1993.
- [22] Klawonn F, Novak V. The relation between inference and interpolation in the framework of fuzzy systems, *Fuzzy Sets and Systems*, Vol. 81, 1996, pp. 331-354.
- [23] Koczy L.T., Zorat A., "Fuzzy systems and approximation," *Fuzzy Sets and Systems*, Vol. 85, pp. 203-222, 1997.
- [24] Lee J., Chae S., "Analysis on function duplicating capabilities of fuzzy controllers," *Fuzzy Sets and Systems*, Vol. 56, pp.127-143, 1993.
- [25] Saade J.J., "A new algorithm for the design of Mamdani-type fuzzy controllers," *Proc. Of the EUSFLAT-ESTYLF Joint Conference*, Palma de Mallorca, Spain, September 22-25, 1999, pp. 55-58.
- [26] Saade J.J, Al-Khatib M., "Efficient representation of non-linear functions by fuzzy controllers design algorithms," *7th IEEE Int'l Conference on Electronics, Circuits and Systems*, December 17-20, 2000, Beirut, Lebanon, pp. 554-557.
- [27] Oh S., Pedrycz W., "Identification of fuzzy systems by means of an auto-tuning algorithm and its application to non-linear systems," *Fuzzy Sets and Systems*, Vol. 115, pp. 205-230, 2000.
- [28] Branco P.J.C., Dente J.A., "Fuzzy systems modeling in practice," *Fuzzy Sets and Systems*, Vol. 121, pp. 73-93, 2001.
- [29] Leski J.M., "Neuro-fuzzy systems with learning tolerant to imprecision," *Fuzzy Sets and Systems*, Vol. 138, pp. 427-439, 2003.
- [30] Zadeh L. A., "Outline of a new approach to the analysis of complex systems and decision processes," *IEEE Trans. Systems, Man and Cybernetics*, Vol. SMC-3, No. 1, pp. 28-44, 1973.
- [31] Saade J.J. "A unifying approach to defuzzification and comparison of the outputs of fuzzy controllers," *IEEE Trans. Fuzzy Systems*, Vol. 4, No. 3, pp. 227-237, 1996.
- [32] Klir G., Yuan B., *Fuzzy Sets and Fuzzy Logic, Theory and Applications*, Prentice Hall PTR, New Jersey, 1995.
- [33] Saade J.J. and Schwarzlander H., "Ordering fuzzy sets over the real line: an approach based on decision making under uncertainty," *Fuzzy Sets and Systems*, Vol. 50, pp. 237-246, 1992.
- [34] Fujimura K., Samet H., "A hierarchical strategy for path planning among moving obstacles," *IEEE Trans. Robotics and Automation*, Vol. 5, pp. 61-69, 1989.
- [35] Griswold N.C., Eem J., "Control for mobile robots in the presence of moving objects," *IEEE Trans. Robotics and Automation*, Vol. 6, pp. 263-268, 1990.
- [36] Lamadrid J.G., "Avoidance of obstacles with unknown Trajectories: Locally optimal paths and periodic sensor readings," *Int.J. Robotics Research* , pp. 496-507, 1994.
- [37] Fiorini P., Shiller Z., "Motion planning in dynamic environments using the relative velocity paradigm," *Proc. IEEE Conference on Robotics and Automation*, 1993, pp. 560-565.
- [38] Barraquand J., Langlois B., Latombe J.C., "Numerical potential field techniques for robot path planning," *IEEE Trans. Systems, Man and Cybernetics*, Vol. 22, pp. 224-241, 1992.
- [39] Donnat J.Y., Meyer J.A., "Learning reactive and planning rules in a motivationally autonomous animat," *IEEE Trans. Systems, Man and Cybernetics B*, Vol. 26, pp. 381-395, 1996.
- [40] Floreano D., Mondada F., "Evolution of homing navigation in a real mobile robot," *IEEE Trans. Systems, Man and Cybernetics B*, Vol. 26, pp. 396-407, 1996.
- [41] Beaufriere B., Zegloul S., "A mobile robot navigation method using a fuzzy logic approach," *Robotica*, Vol. 13, pp. 437-448, 1995.
- [42] Matinez A., et al., "Fuzzy logic based collision avoidance for a mobile robot," *Robotica*, Vol. 12, pp. 521-527, 1994.
- [43] Pin F.G., Watanabe Y., "Navigation of mobile robots using a fuzzy behaviorist approach and custom-designed fuzzy inferencing boards," *Robotica*, Vol. 12, pp. 491-503, 1994.
- [44] Pratihari D.K., Deb K., Ghosh A., "A genetic-fuzzy approach for mobile robot navigation among moving obstacles," *Int. J. Approximate Reasoning*, Vol. 20, pp. 145-172, 1999.
- [45] Al-Khatib M., Saade J. J., "An efficient data-driven fuzzy approach to the motion planning problem of a mobile robot," *Fuzzy Sets and Systems*, Vol. 134, pp. 65-82, 2003.