Comparative Study of Parametric and Non-parametric Approaches in Fault Detection and Isolation

Katebi, S.D.; Blanke, M.; Katebi, M.R.

Publication date:
1998

Document Version
Også kaldet Forlagets PDF

A NEURO-CONTROL DESIGN BASED ON FUZZY REINFORCEMENT LEARNING {PRIVATE }

REPORT

BY

S. D. KATEBI, M. BLANKE

JANUARY 1999

Department of Control Engineering
Aalbrog University
Fredrik Bajers, DK-9220,
Aalborg ø, Denmark

ABSTRACT

This paper describes a neuro-control fuzzy critic design procedure based on reinforcement learning. An important component of the proposed intelligent control configuration is the fuzzy credit assignment unit which acts as a critic, and through fuzzy implications provides adjustment mechanisms to the main controller. The main controller is the neuro-control unit consisting of a full interconnected multi-layer feed forward neural network. The neural network adjusts its weights according to the credit assigned to its output by the fuzz credit assignment unit, using back propagation algorithms. The fuzzy credit assignment unit comprises a fuzzy system with the appropriate fuzzification, knowledge base and defuzzification components. When an external reinforcement signal (a failure signal) is received, sequences of control actions are evaluated and modified by the action applier unit. The desirable ones instruct the neuro-control unit to adjust its weights and are simultaneously stored in the memory unit during the training phase. In response to the internal reinforcement signal (set point threshold deviation), the stored information is retrieved by the action applier unit and utilized for re-adjustment of the neural network during the recall phase. In order to illustrate the effectiveness of the proposed technique, the controller is tested on a cart-pole balancing problem. Results of extensive simulation studies show a very good performance in comparison with other intelligent control methods.

Keywords- Neuro, control, fuzzy, credit, reinforcement

1. INTRODUCTION

Many control theories have dealt successfully with a large class of control problems b mathematically modeling the process and solving the analytical models to generate appropriate control actions. However, the analytical models tend to become complex, especially in large, intricate systems. The nonlinear behavior of many practical systems and the unavailability of quantitative data regarding the input-output relationships make the analytical approaches even more difficult. Hence, many researchers have focused attention on neural network and fuzzy logic techniques as viable alternatives to the traditional controller design methods. Artificial Neural Networks (ANN) and Fuzzy Logic (FL) are complementary technologies in that ANN uses numerical information from the system to be trained or controlled, while FL technique uses linguistic information extracted from experts. Ideally, the two sources of information should be combined.

ANN is basically a numerical technique which utilizes experimental and trained information, while fuzzy decision making is based primarily on uncertainties, imprecision, incomplete data and approximate reasoning. Experimental data are the lowest level of information while the judgment of an expert is the secondary source of information. However, when numeric experimental data are scarce and expensive, they cannot provide a firm base for training an ANN to carry out complex relationships. Thus, the alternative sources of information are the rough and evaluative data, usually available from experts. A fuzzy expert system is particularly suitable when the training data provide incomplete information. Furthermore, an evaluative type of data may ideally be used in the framework of a reinforcement learning process.

In many practical cases there are sub-sections of a problem for which enough experimental data is available, while for other parts either no data is available or they are very limited. In these cases, the problem may be divided into appropriate sections, utilizing advantages of both ANN and Fuzzy Logic (FL) techniques. Several authors have reported, using this strategy for different applications [1,2]. Many different ANN topologies in conjunction with FL have also been reported for control applications [3]. A five layer NN in which the first or the input layer represents the linguistic input and the fifth layer those of the output variables has been studied in [4]. In that report the second and the fourth layers represent the membership functions for input and output respectively. The neurons in the third layer comprise the rule base of the controller. The connections between the third and the fourth layers form the fuzzy relations and the inference procedure. Each connection in the third layer is an antecedent and those in the fourth layer are the consequents of a rule. The back propagation algorithm is employed to train the network. Another configuration for the neuro-fuzzy control is the Approximate Reasoning based on Intelligent Control (ARIC) due to Bernji [5]. The ARIC configuration essentially consists of two main parts, one is the

Action-state Evaluation Network (AEN) and the other is the Action Selection Network (ASN). The AEN acts as a critic and guides the main controller, and the ASN is a multi-layer NN-based fuzzy controller. A reinforcement learning algorithm is used to achieve the desired performance. Lin and Lee [6,7] proposed a reinforcement structure/parameter learning neural-network-based fuzzy logic control to solve various reinforcement learning problems. Based on their previous works [7], the authors integrate two neural network fuzz logic controllers, each of which is a connectionist model with a feed forward multi-layer network. One network performs the fuzzy logic control function and the other predicts the external reinforcement signal while providing compensatory and other informative data for the controller. Both structure and parameter learning are performed automatically during the training phase. The technique works satisfactorily as demonstrated on a cart-pole system. However, the learning speed is comparatively low, and the method relatively complex in the context of an on-line implementation.

The concept of credit assignment is one of the frequently addressed research topics in the field of artificial intelligence. The basic idea is that if the performance efficiency of a process can be evaluated, then the contributing factors may be rewarded or punished accordingly. In a rule base system, this means that a reward or a punishment may be assigned to the set of rules. The idea was first used by Samuel [8] in the game of checkers and Michie and Chambers [9] used the reward/punishment strategy in their boxes system. In another paper, the state space is partitioned into non-overlapping parts (boxes), and b applying forces in the opposing directions and assigning credits to the smaller regions using two neuro-like adaptive elements, the control system is able to learn the balancing of a cart-pole system [10].

In this paper, a neuro-control fuzzy critic design based on reinforcement learning is developed. An important part of the proposed intelligent control configuration is the fuzz credit assignment unit which acts as a critic and provides adjustment mechanisms to the main controller. The main controller is the neuro-control unit consisting of a fully connected multi-layer neural network. The neural network adjusts its weights according to the credit assigned to its output by the fuzzy credit assignment unit, using back propagation algorithms. The fuzzy credit assignment unit comprises a fuzzy system with the appropriate fuzzification, data base, rule base, inference engine and defuzzification components. Initially, sequences of control actions are evaluated and, if necessary, modified by the action applier unit. The desirable ones instruct the neuro-control unit to adjust its connection weights and are simultaneously stored in the memory unit during the training phase. In response to the internal reinforcement signal, the stored information is consequently utilized by the action applier unit for re-adjustment of the neural network weights, during the recall

phase. In order to illustrate the effectiveness of the proposed technique, the controller is implemented on a cart-pole balancing problem. Simulation results obtained indicate ver good comparison with other intelligent control techniques.

2. NEURO CONTROL

Many characteristics of ANN have attracted control engineers to apply this technique to several control problems successfully [11]. The main properties of NN which make it suitable for control applications are its ability for universal mapping, availability of efficient learning algorithms, existence of special purpose hardware in this field and the similarity of NN functioning to the human brain. Neuro-control techniques may be classified according to control design strategies and objectives. Five different design strategies have been reported in the literature [11,12] which are briefly reviewed below.

In the traditional expert systems, a control strategy is evolved based on the observation of actions of an expert operator: *supervised* neuro control is based on this principle. An essential factor in the implementation of this technique is the availability of a set of reliable training data representing a clear relationship between the input variable U and the desired output X. There are many network topologies and training algorithms capable of learning the mapping of U into X. *Direct Inverse Control* (DIC) design approaches are particularl suited to the problems of trajectory following in the robotics manipulator [13,14]. If the mapping of a control variable, such as the joint angles of a robotics arm, is invertable with respect to the manipulated variable, such as the position of the gripper, then the DIC strateg can be applied by training a neural network to follow the defined trajectory in a direct manner. The basic difference between *Neural Adaptive Control* (NAC) and the well know conventional adaptive control method [15] is that a NN is used in the place of the usual linear mappings. However, the implementation of the neural adaptive control strategy is inherently more complex. One of the well known methods in the classical adaptive control is the Model Reference Adaptive Control (MRAC) in which the control system is designed to follow a desired reference model. A neural MRAC can be implemented by defining a cost function as the difference between the output of the model and that of the system, and minimizing this cost function by such method as the back propagation utility [11]. Another problem frequently addressed in the field of adaptive control is that of dealing with hidden and slowly varying modes and parameters of a dynamic system. This difficulty can be dealt with by NN using either of the following two complementary methods. The first method is based on the real-time learning in which the weights of NN are adjusted with respect to the experienced gained during the real-time operation. Another technique is the adjustment of the memory units related to the estimation of the hidden parameters. The combination of the two methods results in a more effective neuro control technique. However, the combined

method requires the Adaptive Critic (AC) [11] algorithm in the controller network, as well as the system identification component in place of the back propagation algorithm. In the conventional adaptive control, Lyapanov's function is frequently used to deal with stability. In the case of ANC this is more difficult and the critic network acts in a very similar way to Lyapanov's function. Basicall *Adaptive Critic* (AC) and *Back Propagation Utility* (BU) are the realization of the optimal control methods by NN. The main idea is to define utilit function with a performance index to be maximized and a cost function to be minimized. Implementation of either of these methods requires more than a single NN. Generally, an *Action Network* is needed to receive the current states and possibly other information as input and to produce the control action as output. The utility function can also be implemented by a NN (utility network) with fixed weights. In most cases there are a *model network* having inputs as current states, sa R(t) as well as the control action U(t) and predicts as output R(t+1) and a vector of the sensor values X(t+1). The model network can also be a *stochastic network* providing estimates rather than the prediction.

3. FUZZY LOGIC CONTROL

The idea of fuzzy sets and fuzzy logic proposed by Zadeh [16] has been applied to man control and decision-making systems. The basic configuration of a fuzzy logic syste consists of a fuzzy rule base and fuzzy inference engine. The fuzzy rule base contains a collection of IF/THEN rules and the fuzzy inference engine uses these rules to map fro fuzzy sets in the input universe of information to fuzzy sets in the output universe of information. In order to use such a fuzzy system for control applications where inputs and outputs are real-variables, a fuzzifier is added to the input and a defuzzifier is added to the output. The fuzzifier maps crisp inputs to the universe of discourse of the input and the defuzzifier maps fuzzy sets at the output to the crisp points. A fuzzy controller design, which is essentially a synthesis of both the control loop and a set of linguistic rules, has been applied to many industrial systems and processes [17-19]. The efficiency of such a fuzz control system depends largely upon the competence of the designer with regards to: a) the completeness of the fuzzy rule base; b) the subjective definition of the membership function; and c) the choice of fuzzy implication operators. Extensive research has been focused on fuzzy adaptive systems, that is, self-organizing, self-learning, etc. Wang [20] considers a fuzzy logic system as a universal approximator and proposes several training algorithms. A three layer feed forward NN in conjunction with back propagation algorithm has been utilized to design an adaptive fuzzy control system. Training algorithms based on orthogonal least square, table-lookup and nearest neighborhood clustering have also been described in [20]. Another direction of development is the application of Genetic Algorithms (GA's) in

designing and optimizing fuzzy systems [21]. Efficient use of GA's in conjunction with fuzzy logic controller design has been reported by several authors [22,23]. In [22] a three-phase framework for learning dynamic control systems has been studied and a genetic algorithm is applied to drive the control rules as decision tables. In the second phase, the rules are automatically transformed into a comprehensive form, and in the last stage the final rules are tuned via manipulation of the fuzzy relational matrix. Park *et al.* [23] show that the performance of the fuzzy control system may be improved if the fuzzy reasoning model is supplemented by a genetic-based learning mechanism. They employed a GA-based procedure to optimize the set of parameters for the fuzzy reasoning model, based either on their initial subjective selection or on a random selection. More recently , a systematic approach to design of fuzzy controller is presented Cheng [24].

4. THE PROPOSED NEURO-CONTROL FUZZY CRITICS

The basic intelligent configuration of the proposed Neuro Control Fuzzy Critic (NCFC) systems is shown in Figure 1.

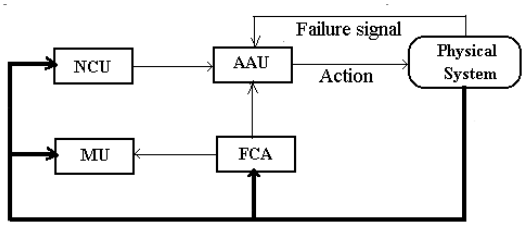


Fig. 1. NCFC configuration

The NCFC system consists of four distinct parts. An important part is the Fuzzy Credit Assignment (FCA) unit, which guides the control system toward the desired performance b evaluating the states of the system and calculating a measure of "goodness" of the syste performance. Based on the evaluated information, the FCA directs the other units. In accordance with the credit given to the states of the system by FCA, the Action Applier Unit (AAU) implements appropriate changes to the output of the Neuro-Control Unit (NCU). As the NCU tends toward a satisfactory performance (a convergent state with weight adjusted), these changes are gradually reduced to zero. The FCA continuously monitors the effects of these changes and, upon observing an improvement in the system performance, the set of state variables, the corresponding control actions and the assigned performance "goodness" values are stored in the Memory Unit (MU). The external reinforcement signal provides a supervisory information for the NCU which employs the back propagation learning

algorithm to adjust its weights until eventually all control actions are assigned " good values. At this stage, the AAU has completed its task and the
NCU acts as the final controller. The functioning of the components of the NCFC is described in more detail in the following sections.

a) Neuro control unit

This unit comprises a fully connected feed forward multi-layer NN. The number of inputs to the network equals the number of state variables, and the number of neurons in the output layer equals the number of control actions applied to the system. The back propagation training algorithm with the sigmoid activation function is employed and the weights are initialized with small random numbers. Since it is required to emphasize those actions with a better " goodness" value and de-emphasize the others with lesser credit, the learning rate is formulated as a function of the measure of " goodness " of the control action. The steepest-descent algorithm is used for modification of the weights. This algorithm adjusts the weights in the negative direction of the gradient of the mean square error function and is formulated as follows:

$$W_{k+1} = W_k + \mu(-\nabla_k) \tag{1}$$

Where the connection weights are denoted b W, and μ is the learning rate. ∇_k is the gradient of a point in the Mean Square Error (MSE) plane at the point $W = W_k$ and is expressed as:

$$\nabla_k = \frac{\partial e^2{}_k}{\partial w_k} = \begin{pmatrix} \frac{\partial e^2{}_k}{\partial w_{0k}} \\ . \\ . \\ . \\ \frac{\partial e^2{}_k}{\partial w_{nk}} \end{pmatrix} \tag{2}$$

Where e is the output error of the NCU, the weights are updated according to:

$$W_{k+1} = W_k + \mu(-\frac{\partial e^2{}_k}{\partial W_k}) \tag{3}$$

Since a sigmoid activation function is assumed, Eq. (3) can be further simplified. Denote the derivative of the square of the error for the jth neuron in the layer l as:

$$\delta^{(l)}{}_j = \frac{1}{2}\frac{\partial e^2}{\partial s^{(l)}{}_j} \tag{4}$$

Where $s^{(l)}{}_j$ is the output and $e_j{}^{(l)}$ is the error for the jth neuron in the layer (l) and is given as;

$$e^{(l)}{}_j = (\sum_{i=1}^{N^{(l)}{}_j} \delta^{(l+1)}{}_i W^{(l+1)}{}_{ij}) \tag{5}$$

Where $N^{(l)}{}_j$ is the number of neurons in the $(l+1)$ layer. With the above definition, we have;

$$\delta^{(l)}{}_j = (\sum_{i=1}^{N^{(l)}{}_j} \delta^{(l+1)}{}_i W^{(l+1)}{}_{ij}) sgm'(s^{(l)}{}_j) \tag{6}$$

Substituting for $e^{(l)}{}_j$ in Eq. (6).

$$\delta^{(l)}{}_j = e^{(l)}{}_j \; sgm'(s^{(l)}{}_j) \tag{7}$$

Therefore, the formula for the update of the weights for a sigmoid activation function is given as;

$$W_j{}^{(l)}(k+1) = W^l{}_j(k) + 2\delta^{(l)}{}_j(k) X^{(l)}{}_j(k) \tag{8}$$

and is expressed in a general form as;

$$W_{k+1} = W_{k+1} + 2\delta_k \; X_k \tag{9}$$

b) Fuzzy credit assignment

The main function of this unit is to train the NCFC with the desirable control actions. Based on the approximate reasoning and fuzzy inference, the effect of a particular control action is assigned a truth value. The fuzzy inference procedure consist of three stages: fuzzification, decision-making logic and defuzzification. In the fuzzification stage, the state variables are normalized, scaled and assigned appropriate membership functions, and an suitable technique may be used for defuzzification. The decision-making logic is performed as follows. Suppose the set of fuzzy rules has more than one antecedent and only one consequent. Let $\mathbf{v}_i$ represent a value satisfying the ith rule, for the input variables, such that;

$$v(i) = \min\{\mu_{il}(x_1),....,\mu_{il}(x_{in})\} \tag{10}$$

where $\mu_{ij}(x_j)$ is the degree of membership of the input x_i in the fuzzy set, whose label is used in the jth antecedent of rule i, and n denotes the number of antecedents in the fuzz rules. The result of applying $\mathbf{V}_i$ in the consequent of the ith rule, denoted by m(i), is obtained as;

$$m(i) = \mu_{Ci}{}^{-1}(w(i)) \tag{11}$$

where μ_{Ci} represents a monotonic membership function whose labels (w(i)) are used in the consequent of the ith rule. The assumption that μ_{Ci} is monotonic and is a one-to-one function guarantees the uniqueness of the w (i) and thereafter that of m (i). Finally, the measure of " goodness", T(E) may be evaluated by combining the consequents of the rules as follows.

$$T(E) = \frac{\sum_{i=1}^{r} m(i)v(i)}{\sum_{i=1}^{r} v(i)} \tag{12}$$

Where r denotes the number of fuzzy rules in the rule base of the FCA.

The calculated value of T(E) is used as the supervisory information to train the NCU. The AAU inspects the value of T(E) and where necessary modifies the control action accordingly. If it has a greater membership grade than a prescribed fuzzy threshold value, the current control action with associated values of input variables and the measure of "goodness" of the control action are stored in the MU.

c) Memory unit

The main task of the MU is to store those sets of state variables with the associated control action and the corresponding values of the performance "goodness" received fro the FCA unit. The capacity of the MU may be determined arbitrarily, as a free design parameter. When all the memory places are occupied, the desirable and improved new data may initially replace the old ones with smaller "goodness" value. When all the places are associated with the same "goodness" value, a simple technique for replacement is to use a rotational procedure, or the method of random replacement may be alternatively used. Another essential function of the MU is to provide a number of appropriate sets of the stored data for on-line re-adjustment of the weights of NCU.

d) Action applier unit

AAU produces and applies the control action to the actual plant and enables the FCA unit to retrieve "good" control actions from the MU. As the NCU learns to control the plant

satisfactorily, the role of the AAU will be reduced and eventually the control actions generated by the NCU will be directly applied to the plant. Different procedures may be adapted for the evaluation of the measure of efficiency of the NCU. A simple technique is to allow the NCU to produce sequences of control actions over a period of time. Efficienc may be calculated as the average taken over the consecutive values of the control actions produced by the FCA for the given sequence. An alternative criterion is the rate of the number of "good" control actions to the "bad" ones produced by the NCU. The AAU monitors and modifies the control actions; in this mode, the efficiency of the performance of the NCU, denoted by P, is calculated as follows.

$$P_k = \frac{\sum_{i=1}^{k} T_i}{k} \tag{13}$$

Where T_i is the "goodness" value calculated by the FCA in the i^{th} step and the following value of P is used for modification of the NCU in the action correction mode.

$$P_{k+1} = \frac{\sum_{i=1}^{k+1} T_i}{k+1} = \frac{T_{k+1} + kP_k}{k+1} \tag{14}$$

The output of the NCU (control action), denoted by U(t), is modified by a random function having its mean dependent on both U(t) and P, and its variance as a function of P only

$$\tilde{U}(t) = Z \Rightarrow f(Z \mid g(U(t), P, h(p)) \tag{15}$$

Where f is the probability distribution function of the random variable z which may be assumed as uniform.

$$\tilde{U}(t) = Z \Rightarrow Q((1-p)U(t), \frac{p}{12}) \tag{16}$$

Where $Q(.,.)$ is a uniform probability distribution function. Equation (15) is expressed in an expanded form as;

$$\tilde{U}(t) = (1-p)U(t) + P\Gamma \tag{17}$$

where Γ is a random variable with zero mean and unity variance, giving $Q(0,1)$.

5. IMPLEMENTATION OF THE NCFC

In order to demonstrate the proposed NCFC techniques, it is applied to a simulated cart-pole problem. Fig.(2) shows the schematic diagram of the system, and the highly nonlinear mathematical model is given as;

$$\ddot{\theta} = \frac{g \sin\theta + \cos\theta[-f - m/\dot{\theta}^2 \sin\theta + \mu_c \operatorname{sgn}(x)]/(m_c + m) - \mu_p \dot{\theta}/ml}{l[\frac{4}{3} - (m\cos^2\theta)/(m_c + m)]} \tag{18}$$

$$\ddot{x} = \frac{f + m/[\dot{\theta}^2 \sin\theta - \ddot{\theta}\cos\theta] - \mu_c \operatorname{sgn}(\dot{x})}{m_c} \tag{19}$$

The state variables are chosen as the position and the velocity of the cart, x and $\dot{x}$ respectively, as well as θ and $\dot{\theta}$ angular position and velocity of the pole with respect to vertical axis. The input to the system is the control force U(t) and the objective is to keep the pole balanced while the cart is constrained to move in a prescribed range. The constants in Eqs. (18) and (19) are; g the acceleration due to gravity, m_c and m the masses of cart and pole respectively, l is the length of the pole, μ_c is the constant of friction between the cart and the track and μ_p is that of the pole at the hinge. It is assumed that for $|x| > 2.4\ m$ *and/or* $|\theta| > 12°$, a failure has occurred and a scalar external reinforcement signal is generated. It is also assumed that the mathematical model is not known to the control system configuration, but a one dimensional vector represents the states of the system and is available at the required instances.

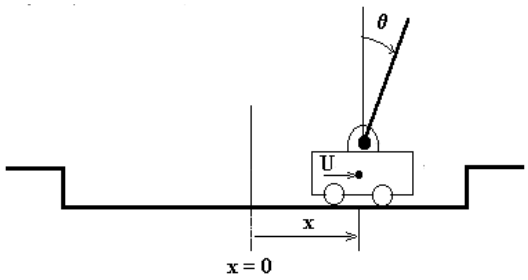


Fig. 2. Schematic diagram of the cart-pole syste

I) Design procedures

a) NCU: A five layer feed forward NN is implemented as the main controller. The number of neurons in the input layer is four (state variables) and there are two hidden layers. It is decided that the neurons in the first hidden larger be eight and those in the second hidden layer be four. The number of hidden layers and the number of neurons in each of these layers are arbitrarily chosen and depend on the particular application. There is onl

one neuron in the output layer as one control action is produced. The network is initialized with very small random numbers and the steepest decent learning algorithm is used. The NCU for this example is shown in Figure 3.

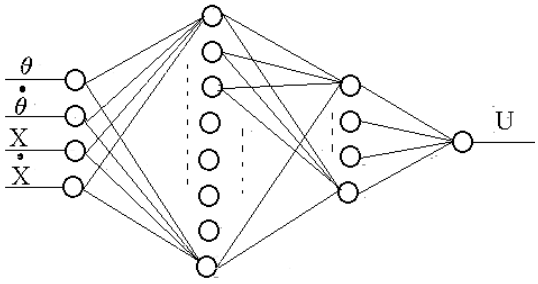


Fig. 3. The neural network for the NCU

b): FCA: For this application, the rule base consists of eighteen rules; twelve of which are related to crediting the control actions for balancing the pole: these are given in Fig.4. The other six are related to the positioning of the cart in the prescribed range and are shown in Fig. 5. The corresponding Fuzzy Associative Memory (FAM) is also shown for each case. FCA assigns a fuzzy truth value to each control action and in the case of no action (no rule is fired) an arbitrary label is assigned.

If θ is NL and $\dot{\theta}$ is NM then Action is VB
If θ is NS and $\dot{\theta}$ is NM then Action is BD
If θ is PL and $\dot{\theta}$ is NM then Action is GD
If θ is NL and $\dot{\theta}$ is NS then Action is BD
If θ is PS and $\dot{\theta}$ is NS then Action is VG
If θ is PL and $\dot{\theta}$ is NS then Action is GD
If θ is NL and θ is PS then Action is GD
If θ is NS and $\dot{\theta}$ is PS then Action is VG
If θ is PL and $\dot{\theta}$ is PS then Action is BD
If θ is NL and $\dot{\theta}$ is PM then Action is GD
If θ is PS and $\dot{\theta}$ is PM then Action is BD
If θ is PL and $\dot{\theta}$ is PM then Action is VB

The corresponding Fuzzy Associate Memory (FAM) is shown in Fig. 4.

If θ is NVS and $\dot{\theta}$ is ZE and C$\dot{\theta}$ is PS and x is NL
then action is GD
If θ is NVS and $\dot{\theta}$ is ZE and C$\dot{\theta}$ is PS and x is NS

then action is VG

If θ is NVS and θ is ZE and $C\dot{\theta}$ is PM and x is NL

then action is VG

If θ is PVS and $\dot{\theta}$ is ZE and $C\dot{\theta}$ is NM and x is PL

then action is VG

If θ is PVS and $\dot{\theta}$ is ZE and $C\dot{\theta}$ is NS and x is PS

then action is VG

If θ is PVS and $\dot{\theta}$ is ZE and $C\dot{\theta}$ is NS and x is PL

then action is GD

Where $C\dot{\theta}$ denotes the change in $\dot{\theta}$ and is calculated as;

$$C\dot{\theta}_t = \dot{\theta}_t - \dot{\theta}_{t-1}$$

$\dot{\theta}$ \ θ	NL	NS	NVS	PVS	PS	PL
NM	VB	BD				GD
NS	BD				VG	GD
ZE						
PS	GD	VG				BD
PM	GD				BD	VB

Fig. 4. Fuzzy rules and the corresponding FAM for controlling the pole

The corresponding FAM's are also given in Fig. 5.

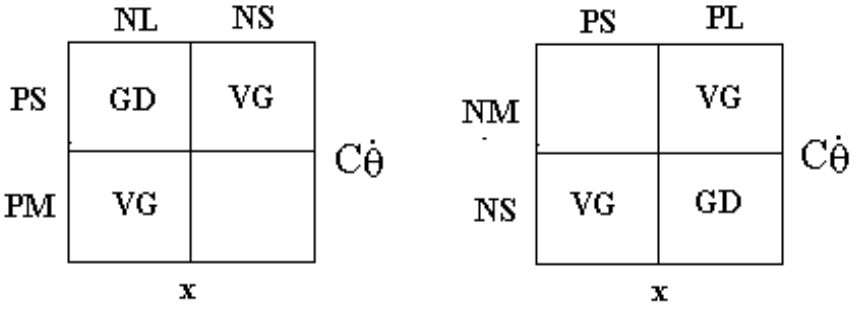


Fig. 5. Fuzzy rules and the corresponding FAM's for controlling the cart

The rules are derived based on the triangular membership functions for the state variables as well as the value of " goodness" of the control action. In order to obtain a

satisfactory resolution and prevent possible oscillation, nine labels are considered for the state variables as shown in Fig. 6a.

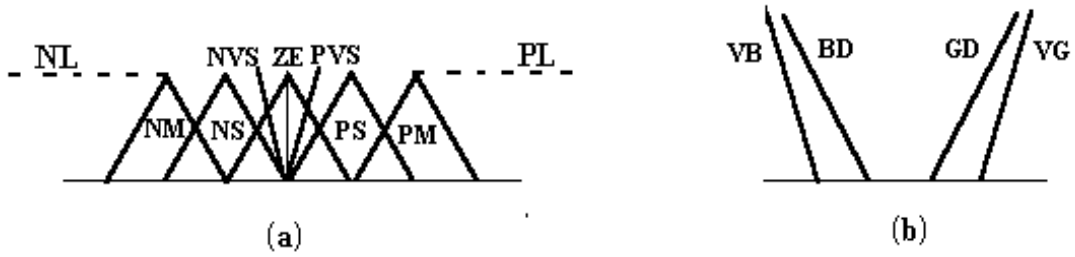


Fig. 6. Membership functions: (a)- state variables, (b)- the "goodness value

The fuzzy labels for the state variable are; NL (Negative Large), NM (Negative Medium), NS (Negative Small), NVS (Negative Very Small), Zero (ZE), PSV (Positive Very Small), PS (Positive Small), PM (Positive Medium), PL (Positive Large). In the case of the "goodness" value, four labels VB (Very Bad), BD (BaD), GD (GooD), and VG (Ver Good), are used, as shown in Fig. 6b.

c): AAU: Initially, AAU allows the NCU to apply the control action directly until a failure signal is received (external reinforcement). This unit then calculates the mean of the syste performance in a recursive manner as follows:

SET " NUMBER OF ACTION " TO ONE

WHILE NOT ACCEPT FAILURE SIGNAL DO

GET " ACTION GOODNESS

System performance = [(number of action-1)*system performance + action goodness]/Number action

INCREMENT "NUMBER OF ACTION

The AAU modifies the control action produced by the NCU as follows;

Applied action = system performance* NCU's action + (1-system performance) e

Where e is a small random number with an arbitrary distribution function, and for this example a uniform distribution is assumed. The modified control action will be applied to the system until another failure signal is received. The procedure is repeated until the control action produces no failure signal and the learning phase is completed.

d): MU: For this application, the MU provides fifty places for storing the actions with higher values than a preset threshold. Each place in the MU contains the values of the state variables, associated control action and corresponding values of the performance "goodness". An internal reinforcement signal is generated when θ and/or x deviates from

the set point by a pre-defined fuzzy threshold value. In the recall phase, a set of ten memor contents is randomly selected for re-adjustment of the NCU's weights.

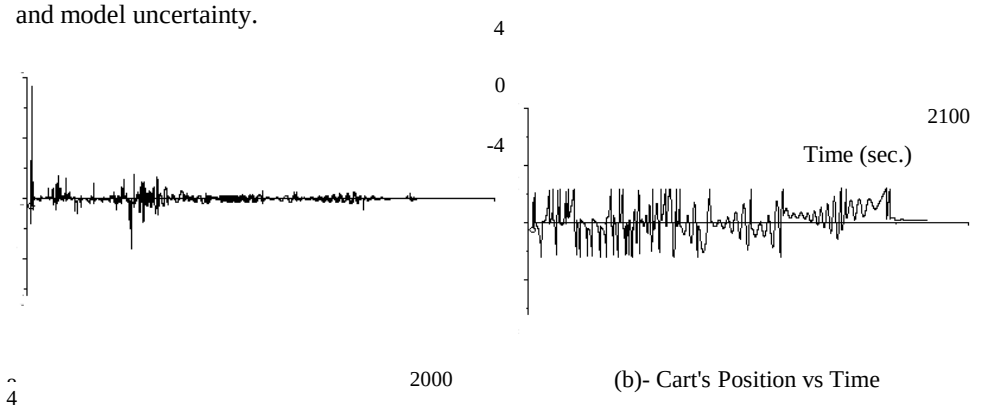
II) Simulation results

In order to illustrate the efficiency of the proposed approach and in particular the learning capabilities of the NCFC, simulation results for a wide range of the cart-pole physical parameters and characteristics are presented. In each set of graphs, the time behavior of the pole's angle, cart's position, control signal and failure signal is presented. The value of the control force is limited to a range of $\pm$150 Newton. When the pole's angle falls outside the range [-12,12] degrees, a failure signal is generated and a new value of this state variable is randomly chosen (within the given range). The cart's position is allowed to vary within [-2.4,2.4] meters and if it exceeds these limits, it is positioned in the middle of the track and the process continues. In all cases the time unit of 20 ms has been used in the simulation, but the results are stored for the plots at every 10 integration steps (200 ms). Each trial begins at the instant of receiving a failure signal and ends with either receipt of another failure signal or the situation of cart-pole systems reaching a steady state (being balanced), and it stays in that situation for a long time (approximately 3 hours). For each simulation result, four sets of graphs are presented.

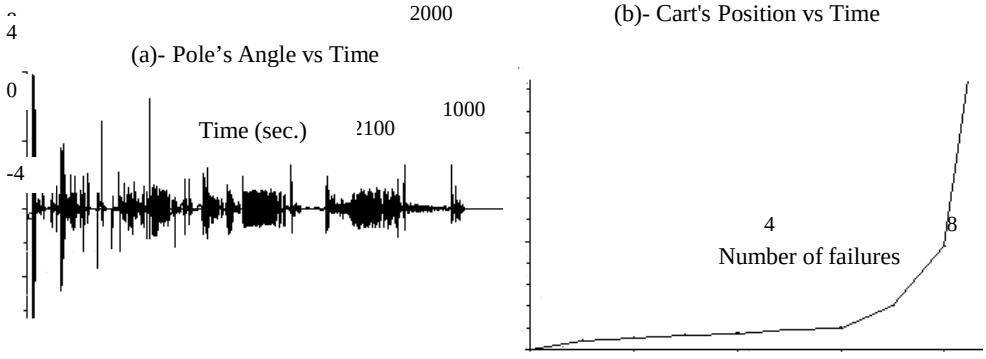
Figures (7a-7d) show the first set of time responses for the length of the pole ; l=0.5 meter, weight of the pole, *m*=0.1 kg and weight of the cart *m*1=2.0 kg. An undesirable system behavior is observed at the beginning, but the system quickly learns to improve and attains desirable behavior. It is seen from the graph for the control force that an improperl large control action has not been required and just enough force is applied to keep the system balanced. This is very near to the behavior of an optimal controller. From the learning curve, it is observed that the number of trials taken for the completion of learning is 10.

In Figs. (8a-8d), only the length of the pole has been decreased to l=0.2 meter and all other parameters are unchanged. It is seen that the system response has not been significantly affected and the learning process for the NCFC has been properly performed. Similar results are shown in Figs. (9a-9d) for increased length of the pole (l=1.0) and in Fig. (10a-10d) both the length of the pole and the weight of the cart have been increased (l=1.0 *m*, *m*=0.2 and *m*1=2.0 kg). It is seen that the intelligent NCFC is robust and has the learning

capabilities of balancing the cart-pole system under a wide range of parameter variations and model uncertainty.

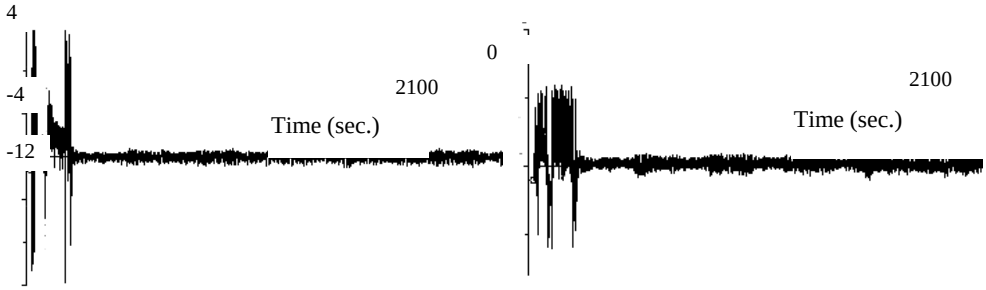


(a)- Pole's Angle vs Time (b)- Cart's Position vs Time

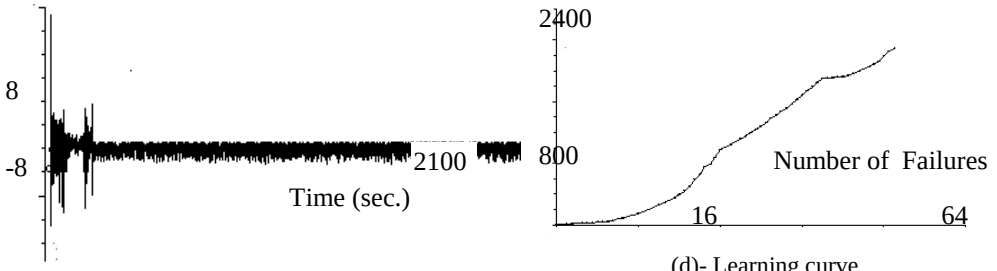


(c)- Control Force vs Time (d)- Learning Curve

Fig. 7. Simulation results for l=0.5, *m*=0.1, *m*1=2.0



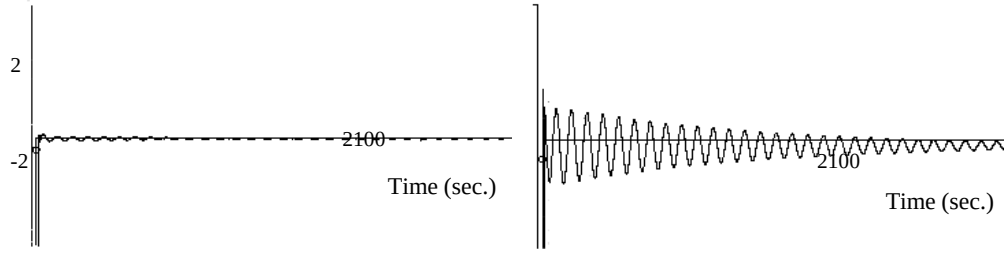
(a)- Pole's Angle vs Time (b)- Cart's Position vs Time



(c)- Control force vs Time

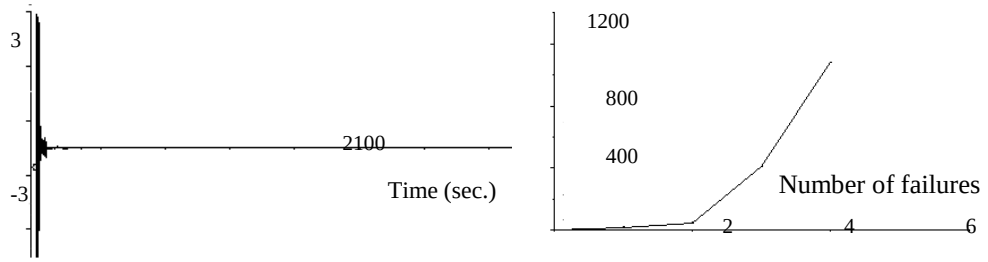
(d)- Learning curve

Fig. 8. Simulation results for l=0.2, *m*=0.1, *m*1 =2.0



(a)- Pole's Angle vs Time

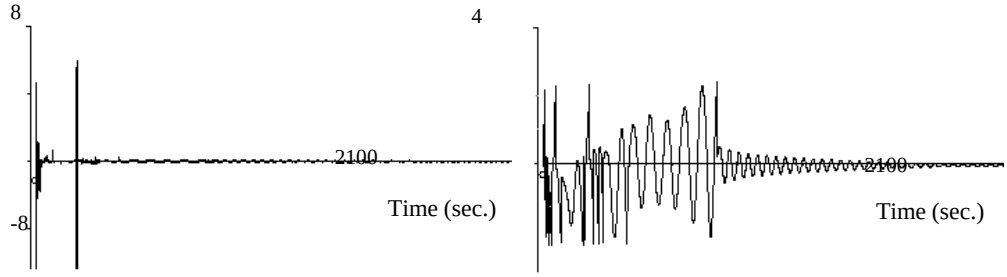
(b)- Cart's Position vs Time



(c)- Control Force vs Time

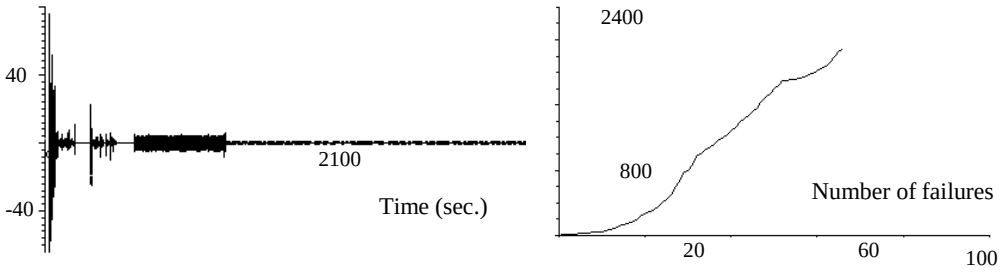
(d)- Learning curve

Fig. 9. Simulation results for l=1.0, *m*=0.2*m*, *m*1=10.0



(a)- Pole's Angle vs Time

(b)- Cart's Position vs Time

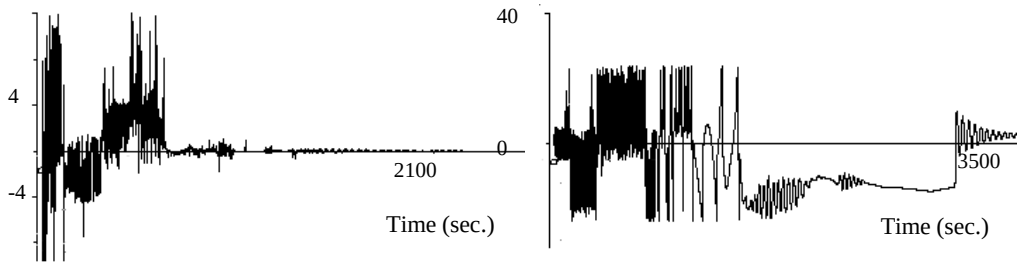


(c)- Control Force vs Time

(d)- Learning curve

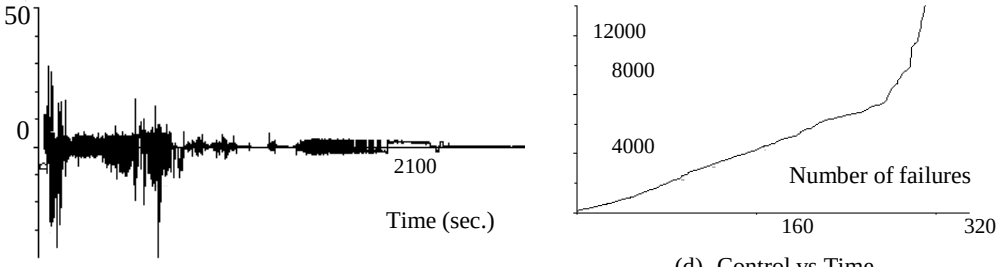
Fig. 10. Simulation results for l=1.0, *m*=0.2*m*, *m*1=2.0

In the above experiments, the parameters of the cart-pole systems were subject to variations. It was previously mentioned that some of the characteristics of the NCFC are arbitrarily chosen by the designer. One of these parameters is the number of layers in the NCU unit. In this last experiment, the number of layers of the NCU was changed fro (4,8,4,1) to (4,8,6,4,1), that is, a second hidden layer with six neurons has been added to the NCU. It is observed from Fig. (11a-11d) that the learning speed has been considerably reduced, but since the number of layers and the number of neurons is increased, the NCU is capable of storing more information, making it more reliable and robust.



(a)- Pole's Angle vs Tim

(b)- Cart's Position vs Time



(c)- Control vs Time

(d)- Control vs Time

Fig. 11. Results of simulation for 2-hidden layers for NCU and (l=0.5, *m*=0.1, *m*1=2.0)

Table 1. Comparison of different intelligent control methods

Control design method	Discrete state	Initialize to random	Continuous force	No. of trials
AHC (Bartol *et al*. [10])	yes	no	no	50
Boxes (Michie, Chambers [9])	yes	no	no	150
Anderson [26]	no	yes	no	8000
Lee and Berenji [24]	no	no	yes	6
ARIC (Berenji [5,25])	no	no	yes	4
NCFC	no	yes	yes	50

In Table 1, the proposed approach is compared with some of the well known intelligent control techniques in which the cart-pole benchmark example has been used. In particular, the results were compared in more detail with those obtained by ARIC due to Lee and Berenji, and Berenji and Khedkar [25,26]. It was observed that, since the nucleus of the ARIC is a fuzzy controller, the learning speed is high, and for this reason the number of trials is small. However, for the same reason, even after the completion of learning the control force behaves in an oscillatory manner, that is, in each step an amount of force is exerted to compensate (neutralize) the previous action.

6. CONCLUSION

A neuro-fuzzy controller design procedure based on reinforcement learning has been proposed. The essential part of the intelligent control configuration is a mechanism for credit assignment to the controller output which is a multi-layer feed forward neural network. The proposed design approach has been implemented on the well known bench-mark problem of cart-pole systems, and the results of extensive simulation studies are presented. The results have shown very good performance in comparison with other intelligent control techniques for controlling such systems. The procedure is also applicable to other nonlinear and/or ill-defined complex systems and processes.

REFERENCES

1. Kosko, B., Neural network and fuzzy systems: A dynamical systems approach to machine intelligence, Englewood Cliffs, N. J. Prentice-Hall (1992).
2. Jin, Y., Jiang, J. and Zhu, J., Neural network based fuzzy identification and its application to modeling and control complex systems, *IEEE Trans. Systems Man. and Cybern* . **25**, No. 6 pp.990-997 (1995).
3. Ishibuchi, H., Fujioka, R. and Tanaka, H., Neural networks that learn from if-then rules, *IEEE Trans, on Fuzzy Systems*, **1**, No. 2, pp. 85-97 (1993).

4. Werbos, P. J., Neurocontrol and elastic fuzzy logic: capabilities, concept and applications, *IEEE Trans. on Indust. Elect.*, **40**, No. 2, pp.170-180 (1993).
5. Berenji, H. R., A reinforcement learning-based architecture for fuzzy logic control, *Int. Journal of Approx. Reasoning*, **6**, pp. 267-292 (1992).
6. Lin, C. T. and Lee, C. S. G., Neural-network-based fuzzy logic control and decision system, *IEEE Trans. Comput*. C-**40**, No. 12, pp.1320-1336 (1991).
7. Lin, C. T. and Lee, C. S. G., Reinforcement structure/parameter learning neural-network-based fuzzy logic control systems, *IEEE Trans. Fuzzy Syst*., FS-**2**, No. 1, pp. 46-63 (1994).
8. Samuel, A. L., Some studies in machine learning using the game of checkers, *J. Res. Develop*., IBM (1959).
9. Michie, D. and Chambers, R. A., Boxes: an experiment in adaptive control, *In: Machine Intelligence*, **12**, pp. 13-152 (1968).
10. Bartol, A., Sutton, R. S. and Anderson, C.W., Neuro-like adaptive elements that can solve difficult learning problems, *IEEE Trans. Syst., Man, Cybern*., **13**, pp. 834-846 (1983).
11. Werbos, P., Neurocontrol and related techniques, (Maren, A. ed.), Academic Press, New York (1990).
12. Narenda, K. and Parthasarathy, K., Identification and control of dynamical systems using neural networks, **1**, No. 1 pp. 4-27 (1990).
13. Eckmiller, R., Beckmann, J., Werntges, H. and Lades, M., Neural kinematics net for a redundant robot arm, IJCNN Proc. IEEE, New York (1989).
14. Jordan, M., Generic constraints on under specified target trajectories, IJCNN Proc. IEEE, Ne York (1989).
15. Astrom, K. J. and Wittenmark, B., Adaptive control, Addison-Wesley (1989).
16. Zadeh, L. A., The concept of linguistic variables and its applications to approximate reasoning, *Information Sciences*, **8**, pp.199-249 (1975).
17. King, P. J. and Mamdani, E. H., The application of fuzzy control systems to industrial process, *Automatica*, No.13 (1977).
18. Jamshidi, M., Vadiee, N. and Ross, T. J., Fuzzy logic and control, Prentice-Hall (1993).
19. Aliev, R., Aliev, F. and Babaev, M., Fuzzy process control and knowledge engineering in petrochemical and robotics manufacturing, Verlag TUV Rheinland (1991).
20. Li-Xin Wang, Adaptive Fuzzy Systems and Control, Prentice-Hall (1994).
21. Kaar, C., Gentry, E. J., Control of pH using genetic algorithms, *IEEE Trans. Fuzzy Systems*, **1**, No. 1, pp. 46-53 (1993).
22. Varsek, A., Urbanica, T., Filipic, B., Genetic algorithms in controller design and tuning, *IEEE Trans. SMC*, **23**, No. 5, pp. 1330-1339 (1993).
23. Park, D., Kandel, A. and Langholz, G., Genetic-based new fuzzy reasoning models with application to fuzzy control, *IEEE Trans. SMC*, **24**, No. 1, pp. 30-47 (1994).

24. Cheng-Sheng, *et al.,* An approach to systematic design of the fuzzy control system, FSS 77 151-166 (1996).
25. Lee, C. C. and Berenji, H. R., An intelligent controller based on approximate reasoning and reinforcement learning, Proc. IEEE Int. Sympos. on Intelligent Control, Albany, N. J. (1989).
26. Berenji, H. R. and Khedkar, P., Learning and tuning fuzzy logic controller through reinforcements, *IEEE Trans. Neural Net*., **3**, No. 5, pp.724-740 (1992).