NEURO-FUZZY MODELLING AND CONTROL OF ROBOT MANIPULATORS FOR TRAJECTORY TRACKING

D. T. Pham and A. A. Fahmy

Manufacturing Engineering Centre, Cardiff University, Cardiff CF24 0YF, U.K.

Abstract: This paper presents a new neuro-fuzzy controller for robot manipulators. First, an inductive learning technique is applied to generate the required modelling rules from input/output measurements recorded in the off-line structure learning phase. Second, a fully differentiable fuzzy neural network is developed to construct the inverse dynamics part of the controller for the on-line parameter learning phase. Finally, a fuzzy-PID-like incremental controller was employed as feedback servo-controller. The proposed control system was tested using dynamic model of a six-axis industrial robot. The control system showed good results compared to the conventional-PID individual joint controller.

Keywords: Dynamic systems, Fuzzy systems, Fuzzy-PID controllers, Neuro-fuzzy systems, Robot manipulators.

1. INTRODUCTION

Selecting the structure and initial parameters of a neural network is a tedious work in dynamic systems modelling. This paper presents an adaptive neuro-fuzzy modelling and control system for robot manipulators based on input/output measurements (Er *et al.*, 1997; Wang and Mendel, 1992). For this purpose, an inductive fuzzy learning technique introduced by Bigot (2003), was modified and used for fuzzy rule generation during the off-line structure learning phase. A fully differentiable fuzzy neural network was developed to construct the inverse dynamics part of the controller. A fuzzy-PID-like incremental controller introduced by Shankir (2001) was modified and used as feedback servo-controller. The proposed control system was tested using the virtual dynamic model of the Puma 560 robot arm.

The remainder of the paper is organized as follows. Section (2) outlines the virtual dynamic modelling process for the robot arm. Section (3) presents the overall structure of the proposed neuro-fuzzy controller and the neuro-fuzzy network. Section (4) describes the structure of the fuzzy-PID-like incremental feedback servo-controller and the on-line parameters learning phase. Section (5) compares the results of applying the proposed control system with those obtained with a conventional-PID individual joint controller. Section (6) concludes the paper.

2. VIRTUAL DYNAMIC MODEL

The parameters listed in (Armstrong and Corke, 1994), were employed to construct the virtual geometrical model as shown in Figure (1) using the engineering design software *Pro/Engineer®*.

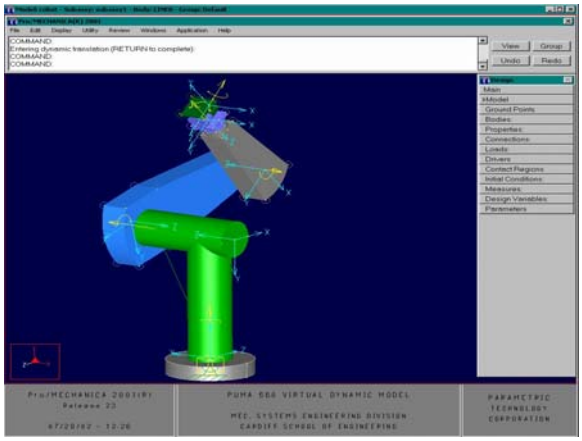

Fig.1. Dynamic model for the Puma 560 robot.

This geometrical model was then imported to the virtual dynamic modelling software *Pro/Mechanica®* where all connections between bodies, joint-reactions, weights, static and dynamic loads and frictions, etc. can be specified. *Pro/Mechanica®* also allows users to write control subroutines using C++, which can be linked to the dynamic model. Input/output data were collected by applying various trajectories to suitably tuned P-controllers, as shown in figure (2). The collected data consist of the sampled applied torque, joint angles, joint velocities, and the Cartesian position of the end effector.

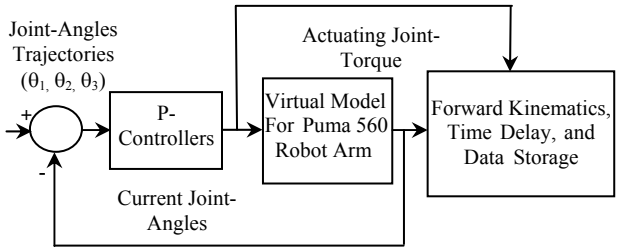

Fig.2. Data collection test for the robot arm.

3. PROPOSED NEURO-FUZZY CONTROLLER

The structure of the proposed neuro-fuzzy control system as shown in Figure (3) resembles the additive feedforward control system presented in Craig (1996). The system consists of a forward path controller in addition to a feedback path controller.

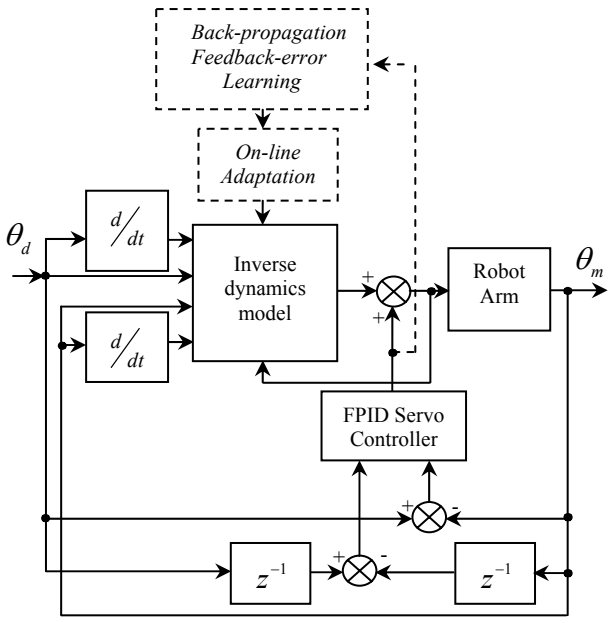

Fig.3. Proposed controller structure.

The forward path controller is a neuro-fuzzy inverse dynamics model for the robot arm. The first step is to generate this model off-line from input/output measurements using a fuzzy inductive learning algorithm introduced by Bigot (2003). This algorithm is designed to extract fuzzy IF-THEN rules from a collection of examples (training set). Initially, a manual step is performed to divide the output variable domain into target classes (fuzzy output membership functions, C_E). Here, each output variable is divided into eleven equal, 50% overlapping Gaussian membership functions as shown in figure (4). Each example (E) (input record) is described in terms of a fixed set of m (no. of inputs) attributes ($A^1, A^2, \ldots, A^m$) (equivalent to linguistic variables) and by a class (output) value (C_E). The number of output membership functions can be regarded as the degree of precision of the model. The higher this number is, the larger the number of rules and the higher the precision.

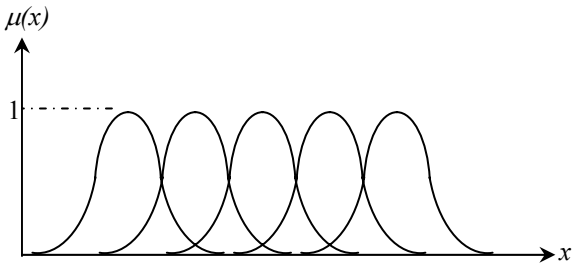

Fig.4. Selected output membership function.

A range of values (V^i_{min}, V^i_{max}), equivalent to linguistic membership functions, is assigned to the i^{th} attribute. Each created rule is composed of a number of conditions on each (or some of the) attribute(s) (Cdt_i) and by its class value (C_{rule}). Each rule can be represented as $Cdt_1 \wedge Cdt_2 \wedge \ldots \wedge Cdt_m \Rightarrow C_{rule}$. Each condition takes the form ($V^i_{min} \leq A^i \leq V^i_{max}$). In order to create a rule set, the algorithm incrementally employs a specific rule forming process until all examples are covered. Three particular steps of this process are of interest to the development of the fuzzy inductive learning algorithm. The first step in this process is to select a seed example (*SE*), which is the first example in the list not covered by previously created rules where the degree of belonging of its output value to the rule output fuzzy set (F_{SE} (a, b, c)) is a maximum and ≥ 0.5. Then, the output class value (output fuzzy set) of the *SE* is used as the target class for the rule to be created. For instance, for a Gaussian membership function, the output membership function will be the fuzzy set (F_{SE} (a, b, c)) in which the membership degree will be maximum. In the particular case of 50% overlapping membership functions, where the membership degree is equal to 0.5 for two adjacent membership functions, only one of them is considered. The second step employs a specific search process to create a consistent and general rule covering the *SE*. The main feature of this search is that the conditions (membership functions) for inputs are created automatically during the rule forming process. The result is a rule where all conditions will take the form ($V^i_{min} < A^i < V^i_{max}$). The third and final step employs a post-processing technique that reduces the coverage of some attribute to the training data range only.

The search mechanism looks for rules that cover as many examples as possible from the target class and at the same time exclude examples belonging to other classes. The rule formation starts with a condition excluding the closest example not belonging to the target class. To find the closest example, a measure is used to assess the distance between any two examples for the i^{th} continuous attribute as:

$$D_{1\&2}=\sqrt{\sum_{c}\left(\frac{Attr\_Value\_Ex.1 - Attr\_Value\_Ex.2}{Max\_Attr\_Value - Min\_Attr\_Value}\right)^2} \quad (1)$$

where $\sum_{c}$ is the sum over all attributes in the examples, *Attr_Value_Ex1* and *Attr_Value_Ex2* are the values of i^{th} attribute in these two examples, and *Max_Attr_Value* and *Min_Attr_Value* are the maximum and minimum values for the i^{th} attribute.

By applying this procedure, the algorithm handles continuous measurements collected from the robot arm. At the end of the rule formation process, each condition takes the form ($V_1^i < A^i < V_2^i$), where V_1^i and V_2^i are continuous values included in the i^{th} attribute range (V_{min}^i, V_{max}^i). After the rule forming process, each continuous condition is transformed into a fuzzy condition using the following method in order to obtain the final fuzzy rule. Consider the condition ($V_1^i < A^i < V_2^i$). This can be transformed into a membership function *F* (a, b, c), where a, b, and c can be defined as follows:

- If V_1^i and V_2^i exist and are real numbers, then a= V_1^i , b= V_2^i , and c = ($V_1^i + V_2^i$)/2.
- If $V_1^i = -\infty$, then a = -∞, b = V_2^i , and c = V_{min}^i , which is the minimum known value of the attribute.
- If $V_2^i = +\infty$, then a = V_1^i , b = +∞, and c = V_{max}^i , which is the maximum known value of the attribute.

In this way, these values can be then employed to generate equivalent Gaussian and sigmoidal membership functions to be used in the Mamdani-type neuro-fuzzy network as shown in figure (5).

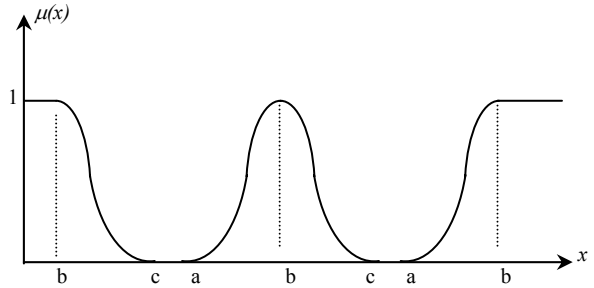

Fig.5. Types of generated input membership functions.

Inverse Dynamics Network

To model the inverse dynamics of the robot arm in a Mamdani-type neuro-fuzzy network using the data recorded, a fuzzy rule-base that represents the inverse dynamics of the robot arm is generated first using the aforementioned inductive learning algorithm. Equation (2) expresses an approximate relation between the desired joint angles trajectories, the required joint torques, and the current joint variables of the robot,

$$T_i^{k+1} \cong f(T_1^k,..., T_n^k, \theta_1^{k+1},..., \theta_n^{k+1}, \theta_1^k,..., \theta_n^k, v_1^{k+1}, ..., v_n^{k+1}, v_1^k,..., v_n^k) \quad (2)$$

where *k* is the sampling interval, *i* = (*1*,*2*,..., *n*), *n* is the number of links, *T* is the joint torque, *v* is the joint velocity, and θ is the joint angle.

Using equation (2), three sets of fuzzy rules can be generated representing the robot inverse dynamics. Each set expresses a joint torque trajectory required to achieve the joint angle trajectories as a function of these trajectories and previous recorded values of these trajectories forming a 12-input single-output relationship. The entire training set is composed of 39,821 examples. All outputs have been decomposed into 11 Gaussian membership functions. The resulting model is composed of 85 rules for the prediction of T_1, 92 rules for the prediction of T_2 and 51 rules for the prediction of T_3, totaling 228 rules performing the prediction of all outputs compared to more than fourteen hundred rules using the method proposed by Wang and Mendel, (1992).

The proposed neuro-fuzzy network is a representation of the Mamdani-model-based feedforward fuzzy neural network (FFNN). The neural network employs time-delayed feedback from the output layer to the input. The selected Gaussian and sigmoidal membership functions are differentiable and their parameters (a, b, and c) can be tuned on-line. Furthermore in order to achieve effective application of the back-propagation learning method, the network employs differentiable alternatives for the *logic-min* and *logic-max* functions in its decision-making mechanism termed *softmin* and *softmax* (Shankir, 2001).

$$softmin(a_i, i = 1, 2, ..., n) = \frac{\sum_{i=1}^{n} a_i e^{-\gamma a_i}}{\sum_{i=1}^{n} e^{-\gamma a_i}} \quad (3)$$

$$softmax(a_i, i = 1, 2, ..., n) = 1 - \frac{\sum_{i=1}^{n} \bar{a}_i e^{-\gamma \bar{a}_i}}{\sum_{i=1}^{n} e^{-\gamma \bar{a}_i}} \quad (4)$$

where a_i is the i^{th} argument, $a_i = \mu_{A_i}$, $\bar{a}_i = 1 - a_i$, and the parameter γ controls the softness of the *softmin* function. As $\gamma \Rightarrow \infty$, *softmin* $\Rightarrow$ *logic-min and softmax* $\Rightarrow$ *logic-max*.

Figure (6) presents the structure of the proposed network consisting of six-layers. The first four layers have the same structure as the first four layers in (Lin and Lee, 1991). The defuzzification function is represented using the last two layers. The *softmin and softmax* functions are used as layer (3) and layer (4) activation functions respectively.

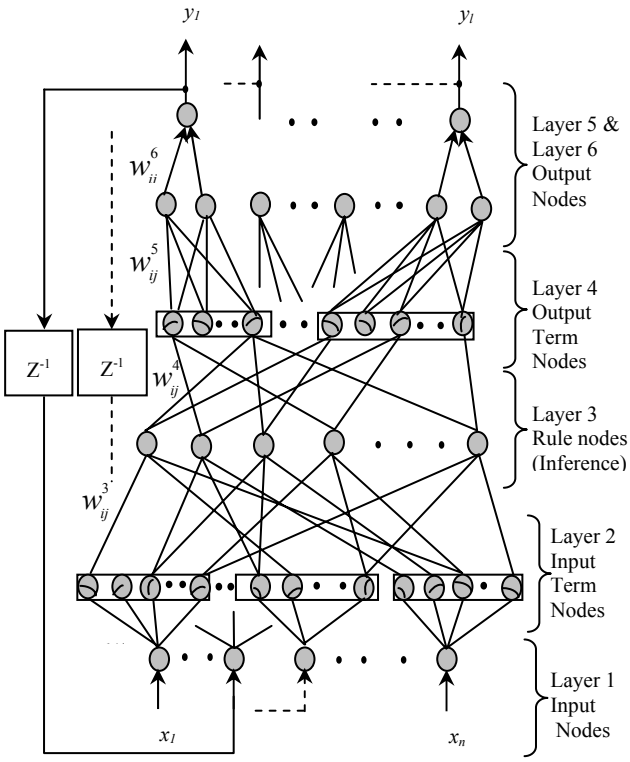

Fig.6. Structure of the proposed neuro-fuzzy network.

4. FUZZY-PID-LIKE SERVO-CONTROLLER

This controller employs two inputs, present and previous errors, and three outputs, P, I, and D. Each output element can approximate the corresponding (functional) control action with independent non-linear gain. The input and output universe of discourses are partitioned using five triangular fuzzy membership functions with 50% overlap as shown in figure (7) and figure (8).

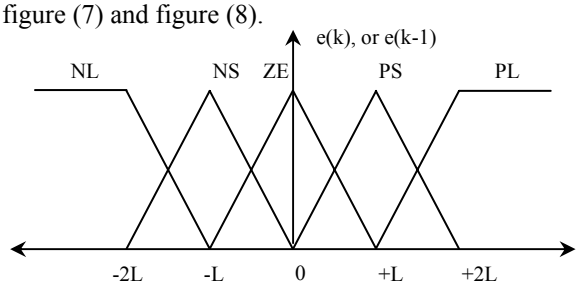

Fig.7. Input membership functions of fuzzy controller.

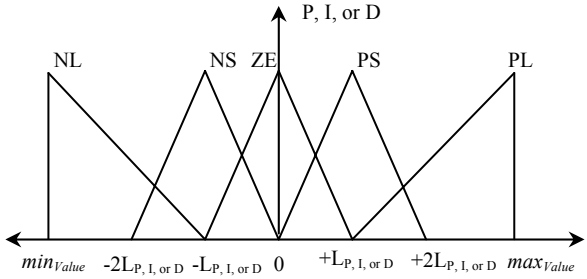

Fig.8. Output membership functions of the fuzzy controller.

The proportional, derivative and incremental part of the integral control actions of the fuzzy-PID-like incremental controller are functions of the two present and past normalized error variables, $e(KT)$ and $e(KT-T)$. Consequently,

$$U_{PID}(KT) = U_I(KT-T) + f_P(e(KT), e(KT\text{-}T)) + f_D(e(KT), e(KT\text{-}T)) + f_I(e(KT), e(KT\text{-}T)) \quad (5)$$

where the three functions f_P, f_D, and f_I are the proportional, derivative and incremental integral functions to be implemented using the fuzzy logic controller and U_I *(KT-T)* is the past output of the integral controller element. The partitions of the output universe of discourses are with different scaling factors to allow different tuning for each control element. The design values were set to L=0.3, L_P=0.2, L_D=0.15, and L_I=0.05.

The fuzzy rules of the Fuzzy Proportional Control Element (FPCE) are generated heuristically based on the intuitive concept that the proportional control action at any time step is directly proportional to the error e_1 at the same time step regardless of the value of the error at the previous time step e_2. Therefore, if the error variable e_1 is expressed linguistically as positive small, the proportional control action can be expressed linguistically positive small and so on. The fuzzy rules of the Fuzzy Derivative Control Element FDCE are generated based on the intuitive concept that the derivative control action at any time step is directly proportional to the rate of change of the error (difference) between two successive time steps. Therefore, if the error variables e_1 and e_2 are both expressed linguistically as positive, the derivative control action can be expressed linguistically as zero and so on. The integral control action is composed of two parts. The first part is the controller output history $U_I(KT\text{-}T)$, while the second part is the incremental output of the controller $f_I(e_1, e_2) = \Delta U_I(KT)$. The fuzzy rules of the Fuzzy incremental Integral Control Element (FICE) are generated based on the intuitive concept that the incremental part of the integral control action

at a time step is directly proportional to the sum of the error variables at two successive time steps. Therefore, if the error variables e_1 and e_2 are expressed linguistically as positive and negative, the incremental part of the integral control action can be expressed linguistically as zero and so on. The rule base of the three incremental FCEs (P, D, and I) can be combined together to form one rule base for the fuzzy-PID-like incremental servo controller output as listed in the table below. The total servo-controller output can be represented as:

$$U_{PID} = k_{NP}e_1 + k_{ND}(e_1 - e_2) + k_{NI}(e_1 + e_2) \quad (6)$$

where k_{NP}, k_{ND}, and k_{NI} are the equivalent non-linear gains that can be defined according to the input condition.

e_1	e_2	P-element	D-element	I-element
NL	NL	NL	ZE	NL
NS	NL	NS	PS	NL
ZE	NL	ZE	PL	NL
PS	NL	PS	PL	NS
PL	NL	PL	PL	ZE
NL	NS	NL	NS	NL
NS	NS	NS	ZE	NL
ZE	NS	ZE	PS	NS
PS	NS	PS	PL	ZE
PL	NS	PL	PL	PS
NL	ZE	NL	NL	NL
NS	ZE	NS	NS	NS
ZE	ZE	ZE	ZE	ZE
PS	ZE	PS	PS	PS
PL	ZE	PL	PL	PL
NL	PS	NL	NL	NS
NS	PS	NS	NL	ZE
ZE	PS	ZE	NS	PS
PS	PS	PS	ZE	PL
PL	PS	PL	PS	PL
NL	PL	NL	NL	ZE
NS	PL	NS	NL	PS
ZE	PL	ZE	NL	PL
PS	PL	PS	NS	PL
PL	PL	PL	ZE	PL

Feedback-Error Learning Scheme

Kawato *et al.*, (Kawato *et al.*, 1988) proposed a novel architecture for adaptive control called the Feedback Error Learning (FEL) control technique. The novelty of the FEL method lies in its use of feedback error as a teaching signal for learning the inverse model. This scheme ensures that on-line training will stop only when the feedback error is zero. This behaviour resembles the integration action in integral controllers so that only the incremental part of the integral control element is used in the proposed feedback servo-controller. The neuro-fuzzy forward path controller parameters are tuned on-line using the feedback controller response as the error signal. The network adjustable free parameters were selected to be centres of the output membership functions of the output term nodes in layer four as well as the link weights at layers two and six. Fukuda *et al.,* (1990) proposed a variable learning method for robotic manipulators neural network controllers called "Fuzzy Turbo", based on fuzzy set theory to avoid stagnation during learning. In this method, a linear-PID feedback controller is used with the feedforward controller. In (Arabshahi *et al.*, 1992), fuzzy control of the learning rate η is suggested. The idea behind fuzzy control of the learning rate is the implementation of the heuristics used for faster convergence in terms of fuzzy IF-THEN rules. However, there is still no general guidance for the proper selection of the learning rate and one can say it is case-dependent. In this work, the fuzzy-PID-like incremental feedback controller along with a fixed learning rate provides the general non-linear policy of the controller and learning signal as well. Weight changes are performed at the k^{th} iteration as,

$$\Delta w_k = \eta T^i_{FB} * \partial T^i_{FF}(w_k) \big/ \partial w_k \quad (7)$$

where T^i_{FF} is the feedforward torque at robot link *i,* T^i_{FB} is the feedback torque at robot link *i,* w_k is the vector of weight values after the k^{th} iteration, Δw_k is the change in these weights, and η is the learning rate. The chain rule is then applied to calculate the network output partial derivatives with respect to the variable weights at each layer.

5. RESULTS

The proposed control system was tested on the first three links of the Puma 560 robot. For comparison purposes, a conventional-PID controller, tuned to produce the most acceptable performance, was used to control the robot over a sinusoidal trajectory for each link, while the arm was carrying a fixed payload of 7.0 kg

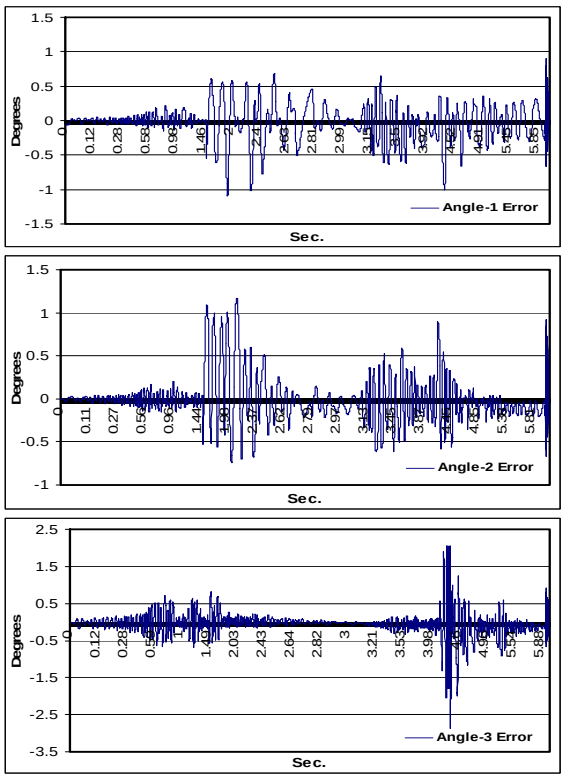

Fig.9. Neuro-fuzzy controller tracking errors.

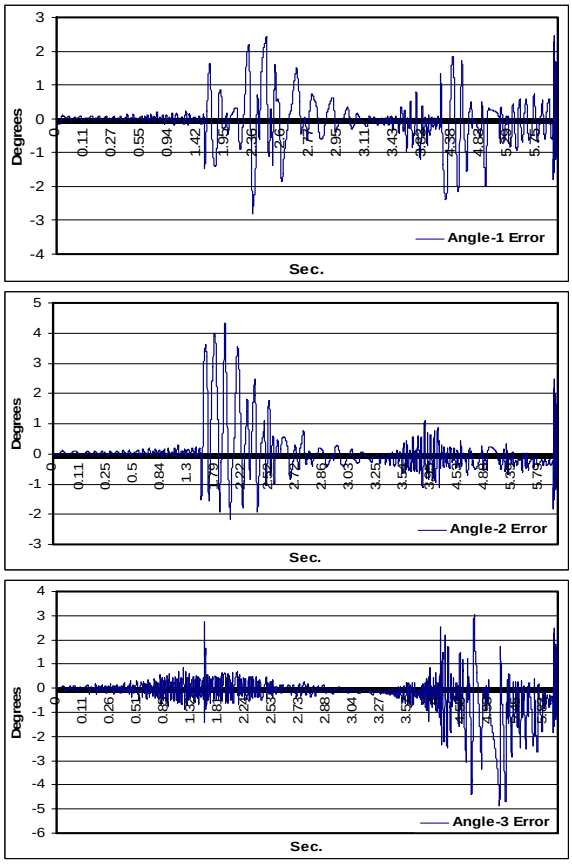

Fig.10. Conventional-PID controller tracking errors.

Figure (9) presents the position tracking errors for the suggested neuro-fuzzy controller, while figure (10) shows the tracking errors for the conventional-PID controller.

6. CONCLUSION

A new technique for modelling and control for robot manipulators was presented. An inductive learning technique is employed to construct the forward path inverse controller using input/output measurements, in the form of a new neuro-fuzzy network. A new fuzzy-PID-like incremental feedback controller was incorporated in the control system. A feedback error learning scheme was used to tune the network weights on-line. It can be observed from the obtained results that the proposed method gave better tracking accuracies non achievable with the conventional-PID controller.

ACKNOWLEDGMENTS

The research described in this paper was carried out within the EC project IST-1999-13109 “Supporting Rehabilitation of Disabled Using Industrial Robots for Upper Limb Motion Therapy”. The other partners in the project are: Budapest University of Technology and Economics, Hungary; National Institute for Medical Rehabilitation, Hungary; University of Rousse, Bulgaria and Zebris Medizintechnik GmbH, Germany.

The authors are grateful for support from the EC FP6 Innovative Production Machines and Systems (*IPROMS) Network of Excellence. Thanks are also due to colleagues in the Manufacturing Engineering Centre, Cardiff University, for their help.

REFRENCES

Arabshahi P., Choi J.J., Marks R.J., and Caudell T.P. (1992), Fuzzy Control of Back propagation, *IEEE International Conference on Fuzzy Systems*, San Diego, USA, Pages 967-972.

Armstrong B. and Corke P.I. (1994), A Search for Consensus Among Model Parameters Reported for the PUMA 560 Robot, *IEEE International Conference on Robotics and Automation*, San Diego, USA, Volume 2, Pages 1608-1613.

Bigot S. (2003), New Techniques for Continuous Values Handling in Inductive Learning, *Ph.D. Thesis*, University of Wales, UK.

Craig J.J. (1996), *Introduction to Robotics: Mechanics and Control*, Addison-Wesley Publishing Company.

Er M.J., Yap S.M., Yeaw C.W., and Luo F.L. (1997), A Review of Neural-Fuzzy Controllers for Robotic Manipulators, Thirty-Second IAS Annual Meeting, *IEEE Industry Applications Conference*, Los Anglos, USA, Volume 2, Pages 812-819.

Fukuda T., Shibata T., Tokita M., and Mitsuoka T. (1990), Adaptation and Learning for Robotic Manipulator by Neural Network, *Proceedings of The Twenty Ninth IEEE International Conference on Decision and Control*, Honolulu, USA, Volume 6, Pages 3283-3288.

Kawato M., Uno Y., Isobe M., and Suzuki R. (1988), Hierarchical Neural Network Model for Voluntary Movement with Application to Robotics, *IEEE Control Systems Magazine*, Volume 8, Issue 2, Pages 8-15.

Lin C.-T. and Lee C.S.G. (1991), Neural Network-Based Fuzzy Logic Control and Decision System, *IEEE Transactions on Computers*, Volume 40, Issue 12, Pages 1320-1336.

Shankir Y. (2001), Fuzzy Logic Systems and Fuzzy Neural Networks for Dynamic Systems Modelling and Control, *Ph.D. Thesis*, University of Wales, Cardiff School of Engineering, Cardiff University, UK.

Wang L.X. and Mendel J.M. (1992), Generating Fuzzy Rules by Learning from Examples, *IEEE Transactions on Systems, Manufacturing, and Cybernetics*, Volume 22, Issue 6, Pages 1414-1427.