2015 The 5th International Workshop on Computer Science and Engineering
doi: 10.18178/wcse.2015.04.125

Neural Net Control of High Nonlinear 2-DOF Nonlinear Robotic Arm

E. Mattar

College of Engineering, University of Bahrain, P. O. Box 32308, Kingdom of Bahrain.

ebmattar@ieee.org

Keywords: UOB-Robotic Arm, ANN Learning, Arm Nonlinear Control, PD controller.

Abstract. A robust ANN system to enhance the control performance and robustness of a Two links robotic system (UoB-Arm) is presented. The UoB-Arm control architecture consists of a classical (PD) controller, independent of the high nonlinear dynamics of the two links robotic model, in parallel with a trainable ANN architecture. ANN architecture is used to identify the whole robotic arm inverse dynamics for arm control compensation. The (PD) controller is used to guarantee the stability and robustness of the whole control system, hence achieve a précised tracking accuracy.

Introduction

Artificial Neural Networks (ANN) appear to offer new promising directions toward better understanding, and perhaps even solving some of most difficult nonlinear control problems. Applications of ANN to adaptive control of nonlinear systems have been intensively conducted in recent years. This due to the fact that ANN has excellent capabilities of nonlinear mapping, they are trained from examples, they don't need any algorithmic model for the problem, they have self-learning capability, self-tuning, and prediction capability.

In this respect, Hisa [1], has applied a PD controller system for a robot arm. His approach requires ANN to identify the complete robot inverse dynamics for compensation. Lakshmi and Mashuq [2], have also introduced an adaptive Neuro-Fuzzy control method. This is for Cartesian motion control of a 4-DOF robot arm. The main controller concept was based on the use of inverse learning Adaptive Neuro-Fuzzy Inference System model only to train itself from certain given robot trajectories. In [3], both Pham and Fahmy have introduced a Neuro-Fuzzy modeling and control technique for robotics manipulators and trajectory tracking. In their research efforts, they presented a novel Neuro-fuzzy controller synthesis for robotic manipulators control. Other similar efforts are also mentioned in [4],[5], and [6]. One of the ANN applications is done by Bogdanov and Timofeev [7] where a ANN controller is used to compensate dynamics approximation errors in the model of the two link robotic system thus providing robust control. Other application for using ANN as an adaptive controller is done by Azevedo and Barreto [8], where the internal model paradigm for control (*IMC*) is implemented using neural networks. The *IMC* implementation using neural network is used to generate suitable torques to track the system at the desired angle.

Methodology: In this research our objective is to employ ANN system for the control of a home built robotics system, (UoB-Arm), as indicated to in Fig.1. The ANN system will be placed in parallel to a Proportional-Derivative PD controller. The PD controller is used to grantee the stability of the system and produce torques needed for tracking. To overcome the uncertainties in the system that was presented because of the dynamics in PD-controller, a ANN is trained with (*inputs/outputs*) set to end up with a PD-controller that compensate for arm dynamics. To accomplish this, first strategy taken was to develop a good model and controller to the system and test its performance without any ANN. The next step was to train the ANN with set of inputs and outputs provided by the first stage and apply ANN established to the system and simulate it with a PD-controller. Hence observe the improvement done due to such control methodology.

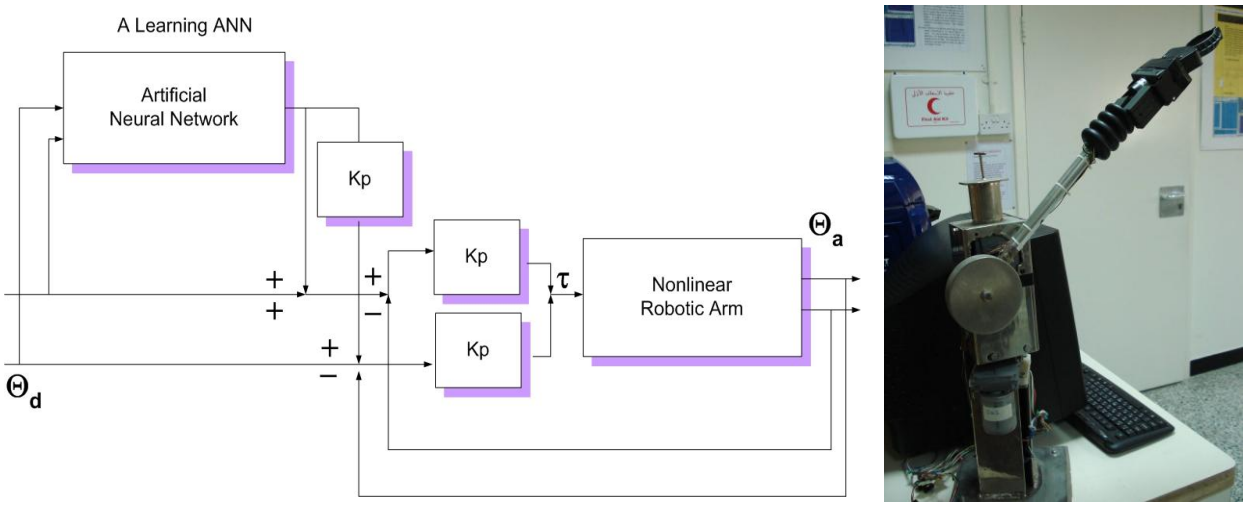

Fig. 1: (*Left*): Integrating ANN with a two links robotics arm, (UoB-Arm), refer to Mattar [9] for details. (*Right*): UoB-Arm, a two links robotic manipulator with known dynamics, Including uncertainties.

Nonlinear Robot Arm Model

In order to program and control the UoB-Robotic Arm system, an initial step is to get knowing the arm nonlinear dynamics. A robot dynamics algorithm is a method of calculating the equations of motion of a robot mechanism, and forces that cause these motions. Typical robots consists of a serial-link manipulators comprising a set of bodies called *link* in a chain, connected by *joints* and each joint has one DOF, either rotational or translational. Every joint supported by a torque driven by the DC motor and a sensor for control purposes. The two links UoB-Arm robotic system consists of two links with representations to their lengths (L_1, L_2) for *link1* and *link2* respectively, and weights of (m_1, m_2) for both links. Inverse dynamic equation of a two link robotic system is known in form of *Lagrange's Equation*, that sums total forces affects the two link robotic systems:

$$M(\theta)\ddot{\theta} + N(\theta,\dot{\theta})\dot{\theta} + C(\theta)\theta + \tau_f = \tau_{in}. \tag{1}$$

the used symbols designate the followings: (θ) *arm position* (*rad*), $(\dot{\theta})$ *trajectory velocity* (*rad/ses*), $(\ddot{\theta})$ *velocity trajectory* (*rad/sec*^{2}). (τ_{in}) *input torque vector* (*N.m*), $(M(\theta))$*inertia force matrix.* (*N.m*) $(C(\theta))$ *gravity of vector.* (*N.m*), *and* $(N(\theta))$ *centrifugal force vector* (*N.m*).

Eq. 1 was used to simulate the a robotic arm with sinusoidal inputs trajectory for both (θ_1, θ_2), hence observe its responses. The constants and inputs to the system are listed below:

Weights of link1 and link 2 are: $(m_1 = 0.4Kg, m_2 = 0.3Kg)$, *Lengths of link1 and link2 are* $(L_1 = L_2)$ *=1ft, and the inputs trajectory to the system are:*

$$\begin{aligned} \theta_d(1) &= .1\sin\left(\frac{\pi}{2}t\right) & \theta_d(2) &= .1\cos\left(\frac{\pi}{2}t\right) \\ \dot{\theta}_d(1) &= .1\times\frac{\pi}{2}\cos\left(\frac{\pi}{2}t\right) & \dot{\theta}_d(2) &= -.1\times\frac{\pi}{2}\sin\left(\frac{\pi}{2}t\right) \\ \ddot{\theta}_d(1) &= -.1\times\frac{\pi}{2}^2\sin\left(\frac{\pi}{2}t\right) & \ddot{\theta}_d(2) &= -.1\times\frac{\pi}{2}^2\cos\left(\frac{\pi}{2}t\right) \end{aligned} \tag{2}$$

After exciting the two links robotic system with such sinusoidal input patterns, Eq.2 the robotic arm resulted in the following arm joint-space motions. This is indicated to in Fig. 2.

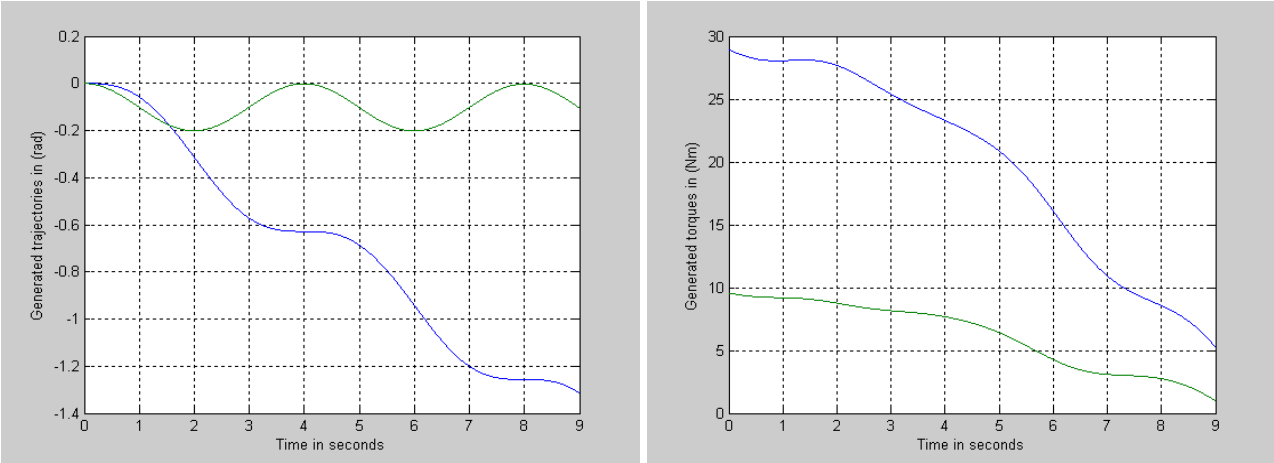

Fig. 2. Open-loop UoB-Robotic arm responses with no control.
(*Left*): Open-loop arm joints motion. (*Right*): Open-loop torques.

Fig.2. shows arm output displacement and torques responses. After such run, we notice that the outputs preformed are not highly related to the inputs imposed. This is because of the dynamic equation's non-linearity, time varying and complexity. The arm was subjected to structured and unstructured uncertainties in all industrial applications. Structured uncertainty is defined as the case of a correct dynamical model with parameter uncertainty due to tolerance variations in manipulator link properties, unknown loads, inaccuracies in the system constants (such as the torque constants of the actuators), etc. Unstructured uncertainty corresponds to the case of un-modeled dynamics.

ANN and Model-Based Arm Control.

As was noticed previously, robot arm dynamic model is a highly non-linear, and complex to model. However, a controller to the robot is frequently needed to stabilize the system or give better performance in the closed loop and also to provide torques needed to the system for the desired trajectories. This is carried out with a standard PD Controller. The PD controller for robot manipulators has been of great interest because of its simplicity and stability, as in Fig. 3.

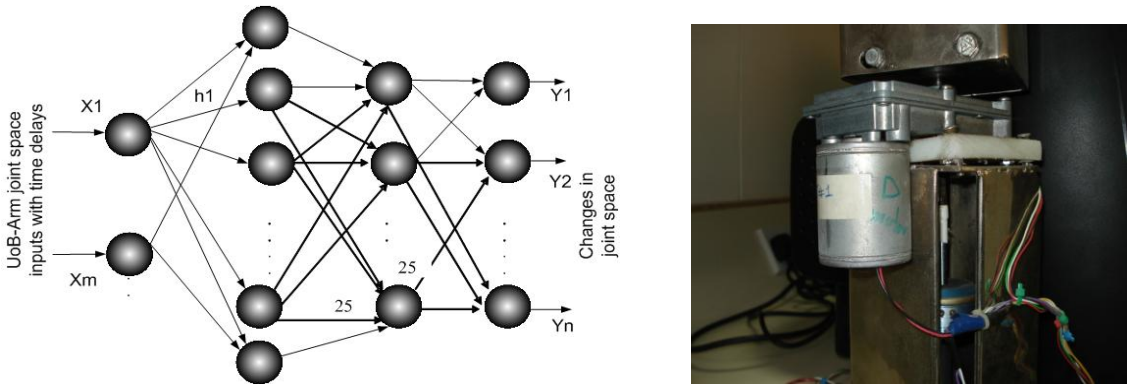

Fig. 3. (*Left*). ANN system for learning arm dynamics. (*Right*). Hardware interface.

The PD controller has the form in Eq. 3 and Eq. 4, $e=(\theta_d-\theta_a)$ represents error and equal to:

$$\tau=\left(K_d.\dot{e}+K_p.e\right). \quad (3)$$

$$e=(\theta_d-\theta_a). \quad (4)$$

θ_d is joint-space *desired trajectory* and θ_a is *actual trajectory*. The input torque to the system will be in realtion to $\hat{M}(\theta)$, $\hat{N}(\theta,\dot{\theta})$ and $\hat{C}(\theta)$, as they are the estimated arm dynamics and models.

$$\tau_{in} = \hat{M}(\theta)S + \hat{N}(\theta,\dot{\theta})\dot{\theta} + \hat{C}(\theta)\theta\,. \tag{5}$$

$$S = \ddot{\theta}_d + K_d.\dot{e} + K_p.e\,. \tag{6}$$

$$\tau_{in} = \hat{M}(\theta)(\ddot{\theta}_d + K_d.\dot{e} + K_p.e) + \hat{N}(\theta,\dot{\theta})\dot{\theta} + \hat{C}(\theta)\theta\,. \tag{7}$$

In Eq. 7, and are diagonal matrices of proportional and speed gains. Each element on the diagonals being positive and are computed by tuning. By this stage, we knew inputs torques to the robot arm, by now we use forward dynamic model to represent the system instead of inverse dynamic equation as stated previously. In such a case, for using forward dynamics, inputs to the UoB-Arm are torques provided by PD controller. It generates trajectories as outputs of the arm, as shown in the Fig. 3. ANN is used to learn the forward dynamic equation by training approach, as derived from the inverse dynamic equation, defined by Eq. 8. Therefore, Eq. 8. has also been used to generate learning patterns for training the ANN system. This means we have learned the arm dynmaics.

$$\ddot{\theta} = M^{-1}\left(-\left(N(\theta,\dot{\theta})\dot{\theta}\right) + C(\theta)\dot{\theta} + \tau_{in}\right)\,. \tag{8}$$

Controller Performace and Results.

Responses of the overall controlled arm, while incorporating ANN and PD controller are discussed here. Results are shown in Fig. 4 and Fig. 5. Here in Fig. 4, we show joint-space motion errors, and changes in errors. The amount of errors is small enough, indicating better closed loop performance. In addition, Fig. 5. shows also the arm joint-space motion (*in rad*), and related computed arm torques needed to achieve the arm motion.

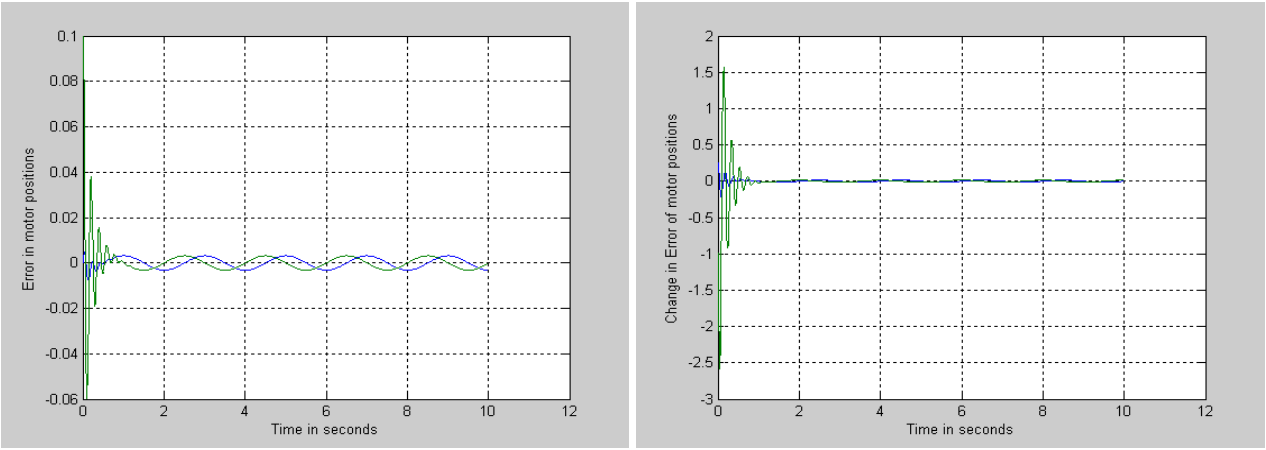

Fig. 4. Initial arm control run using PD controller. (*Left*), error in motor positions. (*Right*), change in error.

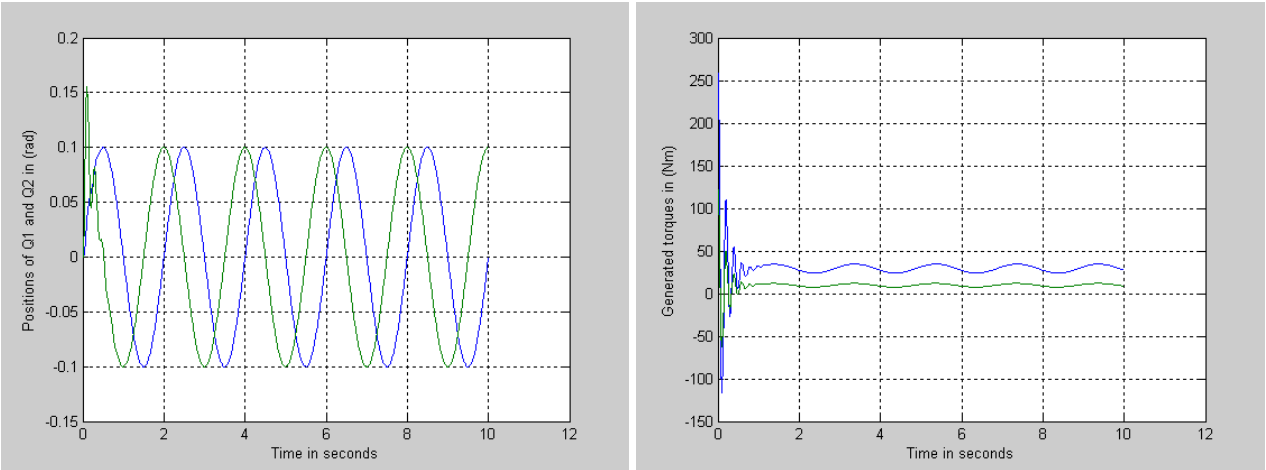

Fig. 5. Closed loop arm control performance. (*Left*). using ANN-PD control. (*Right*). Corresponding arm torques.

After a number of training trails, the used ANN was able to learned the arm dynamics. The ANN then used within the control loop to enhance the control performance. As observed from the above results, (Fig. 4 and Fig. 5), errors between desired arm trajectories and the actual arm displacements are small. Fig. 4 shows initial arm control run using PD controller. It shows error and change in error in motor positions. Fig. 5 also shows the closed loop arm control performance. using ANN-PD control, and the corresponding arm torques. A drawback of the computed torque method is the inverse arm model computation, as it causes error with the estimated parameters. This is due to insufficiently accurate parameters (structured inaccuracies). Error occurs also due of unstructured inaccuracies of the robot system estimation (static and Coulomb friction), and the resulting parameter changes in joints, bearings, etc. these errors also affects the sampling time of the system which causes delay in having the desired torques. Back-propagation learning approach has proven successful in training ANN. It must also calculate how error changes as each weight is increased or decreased slightly during the learning process.

Summary

The proposed ANN controller has shown an excellent solution to overcome complicated dynamics, once dynamics are involved in typical robotic arm model-based control. Initial design stages involved modeling issues, real-time simulation (*for training patterns generation*), hence designing a PD controller based ANN for arm control. The other project part was the implementation of ANN controller. The ANN based controller is considered as an advanced technique compared to classical control approaches. This is due to its capability for learning and non-linear mapping. However the training of the ANN controller was the most time consuming and complicated, and was much potentially hard compared to the other parts of the research. The control design went through many stages of trial and error, to end up with an adequate ANN for such obtained arm motion performance.

References

[1] S. J. Hsia, On an Effective Design Approach of Cartesian Space Neural Network Control for Robot Manipulator, International Journal of Robotica, vol. 15, pp. 305-312 (1997).

[2] V. Lakshmi, U. Mashuq, An adaptive Neuro-Fuzzy control approach for motion control of a robot arm, (ICIEV) International Conference on Informatics, Electronics & Vision, vol. 1, pp.832-836, 18-19 May (2012).

[3] D. T. Pham, and A. A. Fahmy. Neuro-Fuzzy Modeling and Control of Robot Manipulators for Trajectory Tracking, The 16th IFAC WORLD CONGRESS (2005).

[4] Lazarevska, E., A Neuro-fuzzy Model of the Inverse Kinematics of a 4 DOF Robotic Arm, 2012 UKSim 14th International Conference on Computer Modeling and Simulation (UKSim), vol. 1, pp. 306-311, 28-30 March (2012).

[5] H. Pham Huy Anh, K. Kwan Ahn, and Y. Jong I, Dynamic model identification of the 2-Axes PAM robot arm using Neural MIMO NARX model, Proceedings of the ICCE 2008. 2nd International Conference on Communications and Electronics, vol. 1, pp.18-23, 4-6 June (2008).

[6] W. Kelly, R. Challoo, R. Mclauchlan, S. Iqbal, Neuro-fuzzy Control of a Robotic Arm, Proceedings of the Artificial Neural Networks In Engineering Conference, St. Louis, MO, November 10-13, 1996, pp. 837-842 (1996).

[7] D. T. Pham, and A. A. Fahmy. "Neuro-Fuzzy Modelling and Control of Robot Manipulators for Trajectory Tracking," The 16th IFAC WORLD CONGRESS (2005).

[8] F. M. de Azevedo and J. M. Barreto, IMC Scheme Using Neural Networks for Robot Arm, 10 CBA 6 Cong. Latinoamericano de Controle Automático - PREPRINT.

[9] E. Mattar, A Practical Neuro-fuzzy Mapping and Control for a 2 DOF Robotic Arm System, International Journal of Computing and Digital Systems, Int. J. Com. Dig. Sys. 2, No. 3, 109-121 (2013).