Adaptive Control of a Robotic Arm Using Neural Networks Based Approach

Mahdi Vaezi m.vaezi7777@gmail.com
Faculty of Electrical & Computer Engineering
Science and Research Branch of Azad University
Tehran, Iran

Dr. Mohammad Ali Nekouie Manekoui@eetd.kntu.ac.ir
Faculty of Electrical & Computer Engineering
K. N. T. University of Technology
Tehran, Iran

Abstract

A new neural networks and time series prediction based method has been discussed to control the complex nonlinear multi variable robotic arm motion system in 3d environment without engaging the complicated and voluminous dynamic equations of robotic arms in controller design stage, the proposed method gives such compatibility to the manipulator that it could have significant changes in its dynamic properties, like getting mechanical loads, without need to change designs of the controller.

Keywords: Robotic Arm, Nonlinear Multivariable System, Neural Networks, NARMA-L2 Controller, Speed Trajectory.

1. INTRODUCTION

The modern nonlinear control theorem's failure in taming planets which suffer from problems like lack of certainty or unknown parameters in their modeling, complex or prolix regnant equations and etc are predictable sequent of this fact that all classical control theorems are based on the exact identification of systems. This has caused the domination of conventional control theories, like PID, over today's practical systems [16]. Common mechanical manipulators are vastly used systems which make an example of mentioned issue; in [16] and even in discrete form in [17] the famous sliding mode method has been utilized to control robotic manipulator with its dynamic equations which has caused to tremendous design and computational efforts, in [7] another common nonlinear control method ,feedback linearization, has been used for the purpose with similar results, more computationally voluminous adaptive control has been implied on such system in [10], even more recent approaches like neural networks has been used to adjust conventional PIDs in [10]&[3], even most recent neural networks based approaches like NARMA could be used only for single variable systems as mentioned in [1]and[11]; In this thesis after describing the relations over a three dimension robotic arm a new control strategy is introduced with aim of inflicting the reference input trajectory behavior on its wrist point movement .

2. METHODS

The applied controller is based on the NARMA (nonlinear autoregressive moving average) predictive theorem, the mathematical bases are explained in [6], the representation of dynamic discrete systems using state equations is well known as is represented in (1):

$$\begin{aligned} x(k+1) &= f[x(k), u(k)] \\ y(k) &= h[x(k)] \end{aligned} \qquad (1)$$

The problems of control related to system (1) can be stated at various levels of generality. These include the cases when we have the following:

1) f and h are known, and the state is accessible.

2) f and h are unknown, but the state is accessible.

3) f and h are unknown, and only the input and output are accessible.

Of course the case 3 is hardest to handle because system identification and control must be carried out with only observing input and output signals, if a control method can be applied on this case the two sampler cases are easy to handle.

Relative degree: if

$$\frac{\partial(h \circ f_e^k \circ f(x,u))}{\partial u} \begin{matrix} = 0 & 0 \le k \le n-2 \\ \neq 0 & k = n-1 \end{matrix}$$

Where the middle term in left side of equation above is the k time iterated composition of f, then the dynamic system is said to have a related degree of n [14].

In discrete system since by definition the state x(k+n) depends to the input sequence $U_n(k) = [u(k), u(k+1), u(k+n-1)]$, and the same state sequence so the equation (2) can be obtained:

$$x(k+n) = g[x(k), x(k+1), \cdots, x(k+n-1), u(k), u(k+1), u(k+n-1)] \qquad (2)$$

With respect to (1) the equation (3) could be obtained:

$$y(k+1) = F[y(k), y(k-1), \cdots, y(k-n+1), u(k), u(k-1), \cdots, u(k-n+1)] \qquad (3)$$

The last equation is similar with the nonlinear ARMA predictive formula [3]. If one extends (3) around it's input vector with this term that u(k) is at small amplitude compare to the systems equilibrium state then (4) could be obtained:

$$y(k+1) = f_0[y(k),\cdots,y(k-n+1),u(k-1),\cdots,u(k-n+1)] + g_0[y(k),\cdots,y(k-n+1),u(k-1),\cdots,u(k-n+1)]u(k)$$

(4)

This theorem still is not suitable to devise a control method, with a little more approximation (5) is acquirable:

$$u(k+1) = \frac{y_r(k+1) - \mathrm{f}[y(k),\ldots,y(k-n+1),u(k),\ldots,u(k-n+1)]}{\mathrm{g}[y(k),\ldots,y(k-n+1),u(k),\ldots,u(k-n+1)]} \qquad (5)$$

The last equation is known as NARMA-L2 equation and first described is [13], with authorizing a suitable training strategy for a block that contains two neural networks which must pattern f & g functions (this block will emulate the uncontrolled planet) and then rearranging this two networks to achieve (5) a neural network based adaptive controller block is acquirable; the well known multilayer perceptrone networks seems to fit perfectly for this objective since they have proved ability to map any kind of math function between their input and output.

3. THEORY

This thesis contains part of the research that have been accomplished through author's master of science level final project about to designing a "3d hand movement trajectory system by a robotic arm" which focused on controlling the robotic arm, first the specified robotic arm's properties must be explained; A three rotational joints robotic arm is considered with length of 40 cm for the first & 20 cm for the second and third links, this establishment of robotic arms is very common and is known as RRR or humanoid robotic arm (fig,1 & 2) and fits best to emulate human movements.

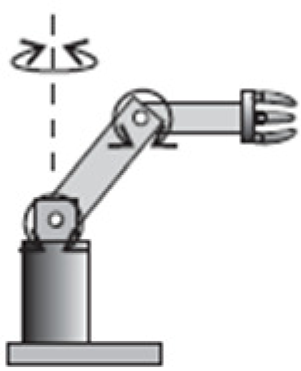

FIGURE 1: Common RRR Arm Topology.

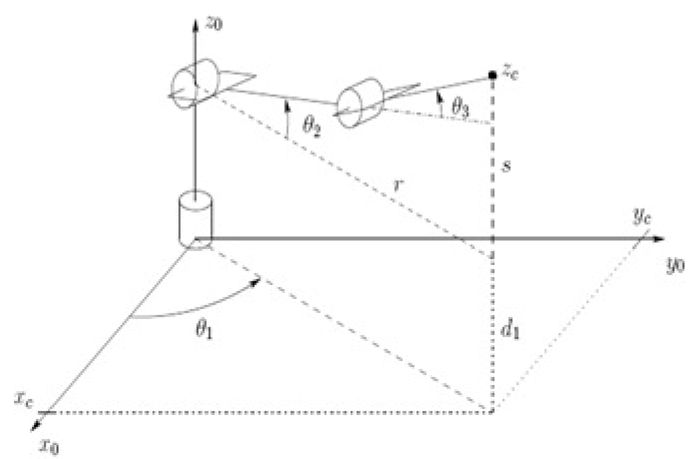

FIGURE 2: Detailed pattern.

using the standard Denavit-Hartenberg method [16] to analyze this robotic arm, the Jacobean matrix can be obtained from vector derivation of the direct kinematic matrix(6) , this kind of Jacobean matrix (7) gives the robot 's wrist point speed in three Cartesian directions in respect to its joints rotational speed(8), and the matrix's elements are three angelic functions of the joints angels. Also the invert Jacobean matrix does the inverse relation (9).

$$\begin{bmatrix} xe \\ ye \\ ze \end{bmatrix} = \begin{bmatrix} 0.2 * (cos\,\theta 1 \;* cos(\theta 2 + \theta 3) + cos\,\theta 1 * cos\,\theta 2) \\ 0.2 * (sin\,\theta 1 \;* cos(\theta 2 + \theta 3) + sin\,\theta 1 * cos\,\theta 2) \\ 0.2 * (\,sin(\theta 2 + \theta 3) + sin\,\theta 2 + 0.4 \end{bmatrix}$$

(6)

$$J = \begin{bmatrix} -\sin\theta 1 * \cos(\theta 2 + \theta 3) - \sin\theta 1 * \cos\theta 2 & -\cos\theta 1 * \sin(\theta 2 + \theta 3) - \cos\theta 1 * \sin\theta 2 & -\cos\theta 1 * \sin(\theta 2 + \theta 3) \\ \cos\theta 1 * \cos(\theta 2 + \theta 3) + \cos\theta 1 * \cos\theta 2 & -\sin\theta 1 * \sin(\theta 2 + \theta 3) - \sin\theta 1 * \sin\theta 2 & -\sin\theta 1 * \sin(\theta 2 + \theta 3) \\ 0 & \cos(\theta 2 + \theta 3) + \cos\theta 2 & \cos(\theta 2 + \theta 3) \end{bmatrix}$$

(7)

$$\mathrm{J} = \frac{\partial \mathrm{X}}{\partial \theta} => \frac{\mathrm{dX}}{\mathrm{dt}} = \mathrm{J}\frac{\mathrm{d}\theta}{\mathrm{dt}} \qquad (8)$$

$$\frac{\mathrm{d}\theta}{\mathrm{dt}} = \mathrm{J}^{-1}\frac{\mathrm{dX}}{\mathrm{dt}} \qquad (9)$$

These are the kinematic relations but there is more to be considered about robotic arms, equations (10) shows the general form of robotic arms dynamic equation.

$$D(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) \;\; = \;\; \tau \qquad (10)$$

In detailed form it is a extremely voluminous matrix equation with relative factors which explains the relation between actuators torque and respected joint's angel, rotational speed and acceleration through gravity, curiolis and torque terms. This equation is not utilized directly in this work so we postpone further explaining. The primary necessity for gaining propitious trajectory behavior is to design a suitable control diagram, with accent to preserve good precision in regulating an object's movement it seems adjusting its speed may bring better results than dealing directly with its position, it is guaranteed that the object will be in right position in right time if the ration of displacement with respect to time is accurate, so the control diagram in figure (3) is presented.

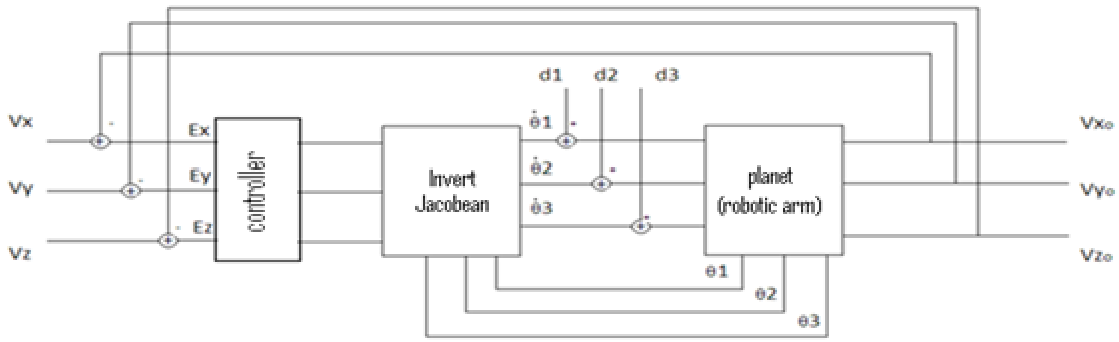

FIGURE 3: control diagram for the described system.

With the aim of problem statement the system state equations must be obtained, if $x=\theta$ is the 3×1 vector as the state variable and with neglecting controller block (k=1) the equation (11) can be obtained for the X axis, two others are similar.

$$\mathbf{Vx} = (\boldsymbol{U}\mathbf{x} + (\boldsymbol{J}_{12}\boldsymbol{D}_2 + \boldsymbol{J}_{13}\boldsymbol{D}_3))/2 \qquad (11)$$

This equation shows that in each axis the output is a direct function of input in addition of long and voluminous terms those are respectful of state variable and its derivatives. But where the disturbance input comes from? It has been shown in equation 11 that the robotic arm's dynamic relations cause a load torque in actuator, since every kind of actuators has a load-velocity profile with negative slope then the load torque (which is generated by dynamic effects) has the role of disturbance when the aim is to regulate the robot actuators velocity, for example the typical load-velocity profile of dc electrical actuator is shown in figure (4) [2]. The exact relations between input and output are acquirable but the state equations are very massive and this blocks the way to design any classical controller. It is also obvious that any control theorem that might be applied must be based on multi variable approach that doubles the required designing efforts and unguaranteed results; it seems the only remaining option is attending to implement the statistical approaches like neural networks based ones.

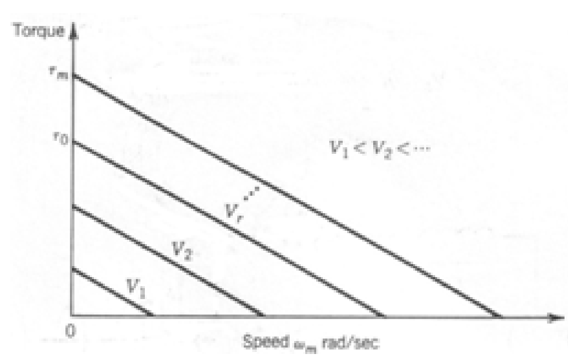

FIGURE 4: typical torque-velocity profile of dc servo actuators

Many researchers and authors have described methods like computing conventional PID compensator's factors using neural networks or genetic algorithms and it would be pointless to focus on such methods. The intended method is based upon the paraphrased NARMA-L2 predictive method, to begin let's take another look to equation (5) as equation (12), this is also interesting that (12) is very similar to (11).

$$u(k+1) = \frac{y_r(k+1) - \mathrm{f}[y(k), \dots, y(k-n+1), u(k), \dots, u(k-n+1)]}{\mathrm{g}[y(k), \dots, y(k-n+1), u(k), \dots, u(k-n+1)]} \quad (12)$$

It is obvious that a propitious tracking behavior could be expected neutralizing this theorem when this is about a SISO system, but the current problem is with a multi input-output extremely nonlinear a voluminous system then how described NARMA-L2 approach might be useful here? The answer is acquirable with noting to two separate mathematical theorems:

Kolmogorov's existence theorem: "Every continuous N variable (input) function could be approximated using linear additions of nonlinear functions which are only relative with one of the variables". [8].

A multi layer percepteron neural network can emulate behavior of any type of mathematical function with unlimited discontinuous points [5].

So it could be said that a NARMA-L2 controller can emulate & control the described system's behavior if it contains two expedient networks and a suitable training strategy is applied, then for each input axis an individual controller block is needed; in order to train the networks perfectly for example networks related to controller X are connected to the planet model's adequate input and output while two other inputs have random pulse wave inputs in order to motivate all system modes (fig. 5).

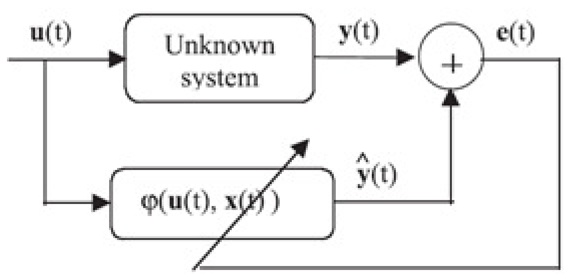

FIGURE 5: training strategy

In general three separated networks are used and each one has several delayed inputs and two out puts (parallel networks) to train and emulate f and g functions, in this case we don't know the exact state equations of the plant so the control method is called NARMA-L2 , if we knew the state equations this method would have been called neural based feedback linearization method, however figure 6 shows the networks general form in the simplest shape.

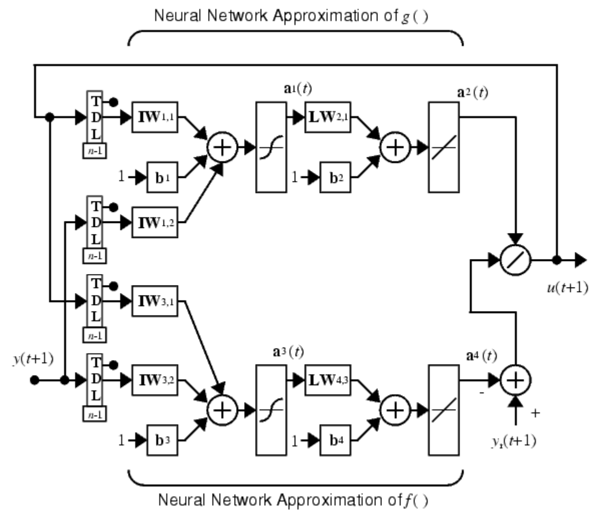

FIGURE 6: general formation of NARMA-L2 control made in MATLAB SIMULINK

There is only one neuron in hidden layer in fig. 6 but in practice more neurons are needed, usually between 10-13 hidden layer's neurons could handle the job , also number of delayed input is an important matter, since we don't know the degree of the plant's model (NARMA-L2 case) then number of delayed inputs must be obtained by examining in simulation, as described commonly robotic arms have very complicated and extensive ruling relations, in this works not less than 5 delayed inputs showed the suitable behavior. After getting the trained NARMA-L2 controller it must be connected to the plant, the general connection shape is shown in figure 7.

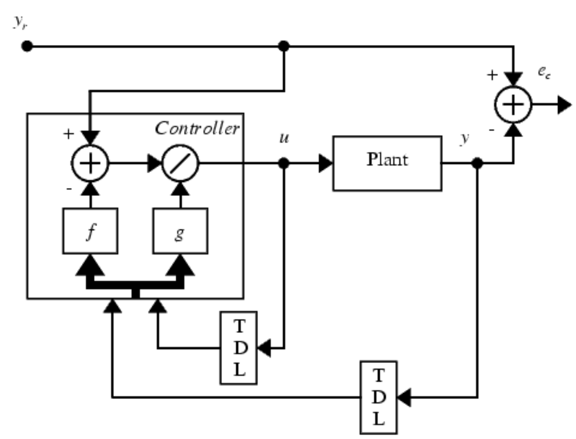

FIGURE 7: general shape for connecting the NARMA-L2 controller, error output could be useful for further experimentations.

And at last figure 8 shows the created formation of three NARMA-L2 controller in this work, note that in figure 3 there is feedback lines those are needed in classic control methods but in NARMA-L2 the conventional feedback is not used and the similar signal lines are needed to provide delayed inputs for the inside networks. The training performed using levenberg – marquardt method (trainlm), this presents a nonlinear algorithm for training multi layer perceptrones based on the Newton's method for minimizing functions which are consisted of linear sum of nonlinear functions squared form, this method has better performance and results that many others but is profable only until several hundred neuron establishment of a network because of its high memory capacity requirement, the learning rule is shown in (13):

$$W(n+1) = W(n) - [J^T(n)J(n) + \mu I]^{-1} J^T(n)e(n) \tag{13}$$

Where μ is a factor which in small amount learning rule is Newton's formula with a hessian matrix and in larger amounts of μ this is gradient descent with small step size, since first form has faster and more accurate performance near a performance function's minimum point training's goal must be to reduce μ in every step' thus μ is always reduced and only in case of errors in crescent it steps up, and J is Jacobean matrix consisted of error's derivations with respect to weight factors variation shown in (14).

$$J(n) = \begin{bmatrix} \frac{\partial e_1(n)}{\partial w_1} & \frac{\partial e_1(n)}{\partial w_2} & \cdots & \frac{\partial e_1(n)}{\partial w_n} \\ \frac{\partial e_2(n)}{\partial w_1} & \frac{\partial e_2(n)}{\partial w_2} & \cdots & \frac{\partial e_2(n)}{\partial w_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_N(n)}{\partial w_1} & \frac{\partial e_N(n)}{\partial w_2} & \cdots & \frac{\partial e_N(n)}{\partial w_n} \end{bmatrix} \tag{14}$$

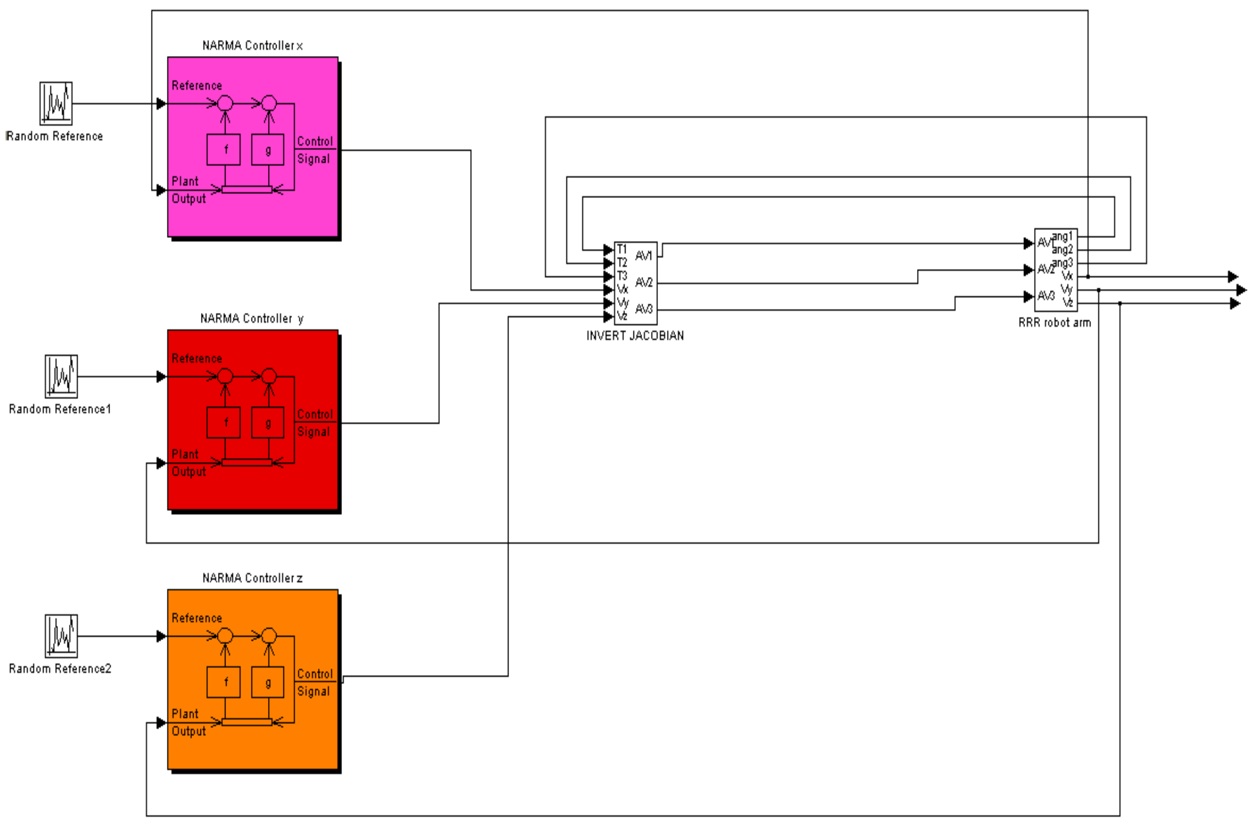

FIGURE 8: experimental formation of NARMA-L2 control of described three input-output robotic arm plant.

3. SIMULATION EXPERIMENTAL RESULTS

The whole system could be simulated in MATLAB SIMULINK, after making the planets model the three NARMA-L2 controllers trained executing the paraphrased strategy, the following figures are the three stage of networks training (training, testing and validating sets), figures 9 for NARMA-L2 controller X, 10 for Y & 11 for Z.

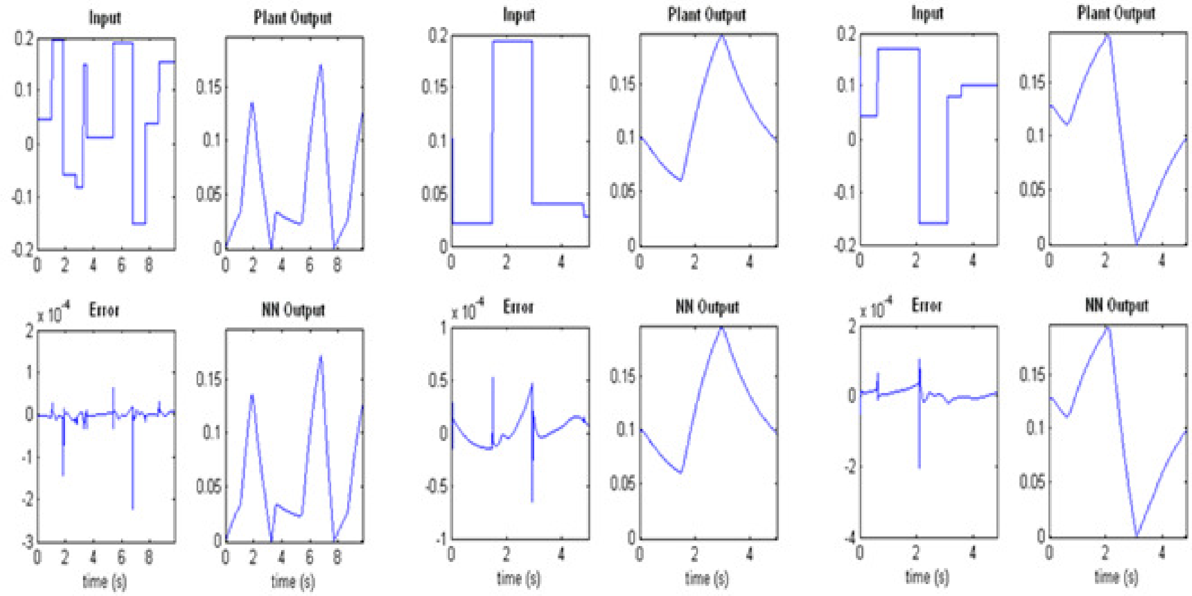

FIGURE 9: training, testing and validation sets for the NARMA-L2 controller in axis X, this figures also show the untamed plant's respond to puls inputs.

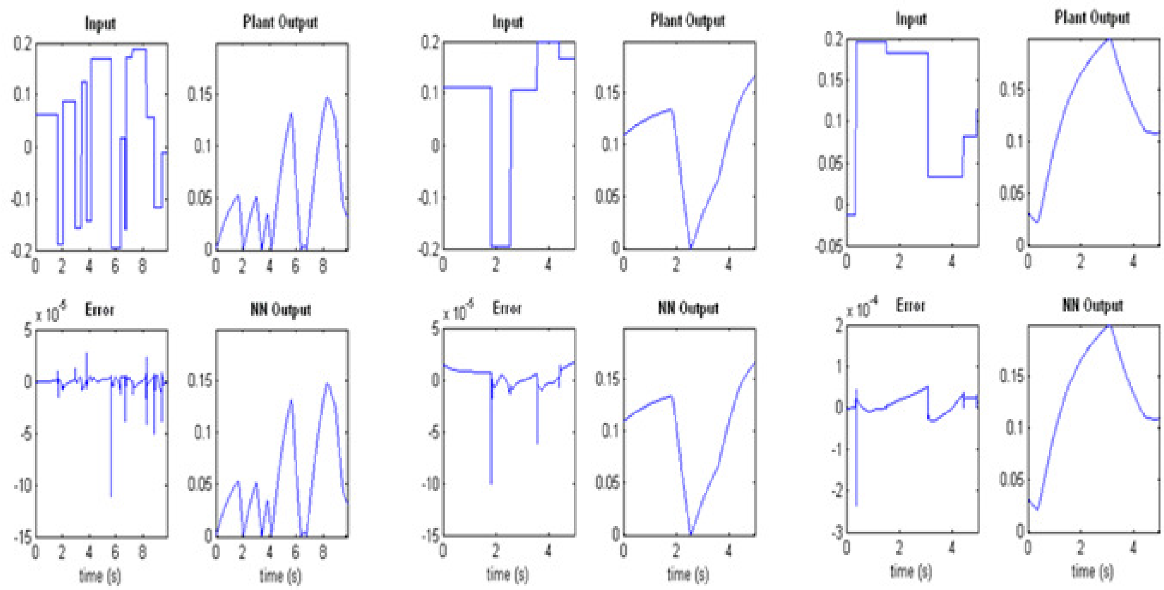

FIGURE 10: training, testing and validation sets for the NARMA-L2 controller in axis Y, this figures also show the untamed plant's respond to puls inputs.

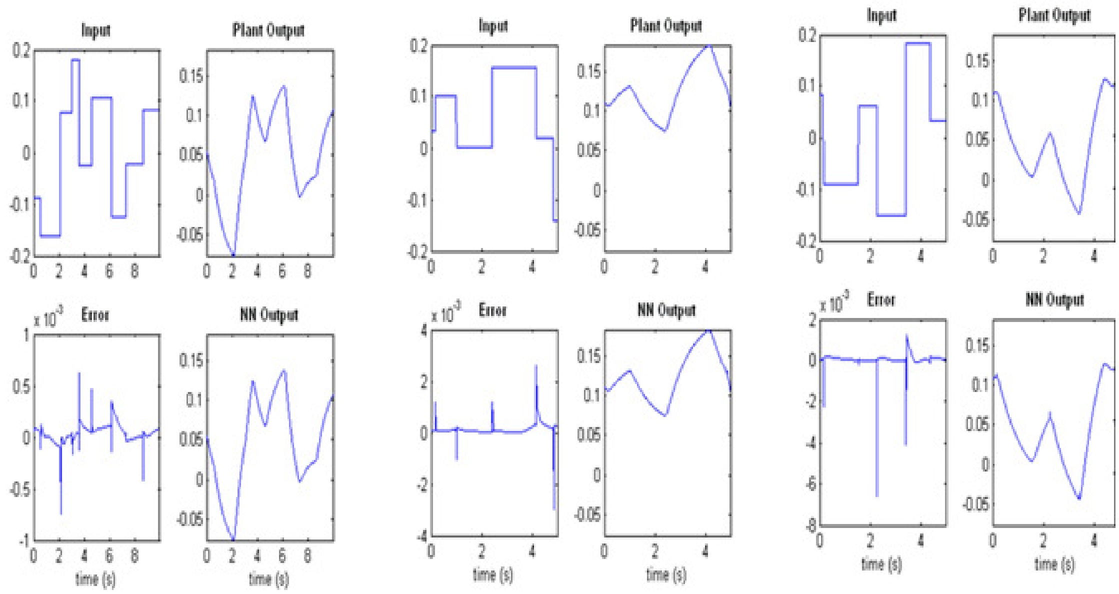

FIGURE 11: training, testing and validation sets for the NARMA-L2 controller in axis Z, this figures also show the untamed plant's respond to puls inputs.

After some retryings in training procedure a suitable reffrence tracking behaviore could be obtained, it seems that NARMA-L2 controller does the best about this system with 14 hidden layers and 5 delaied input-outputs; figue 12 shows the trajectory profile in X axis and the two other for Y and Z.

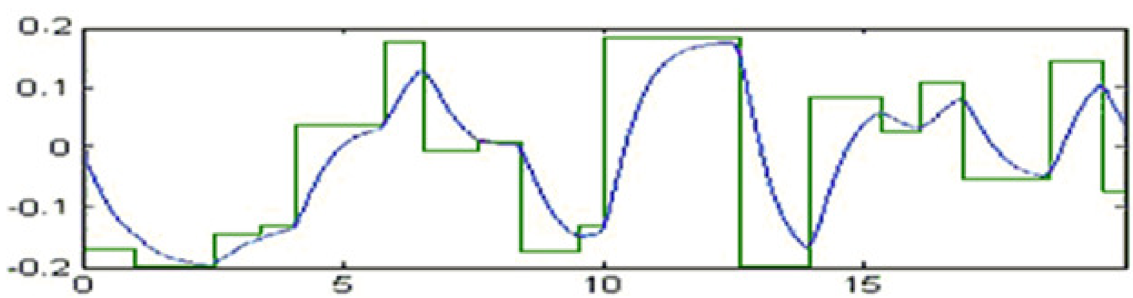

FIGURE 12: controled plant's refrence tracking profile in axis X neutralizing NARMA-L2 control method.

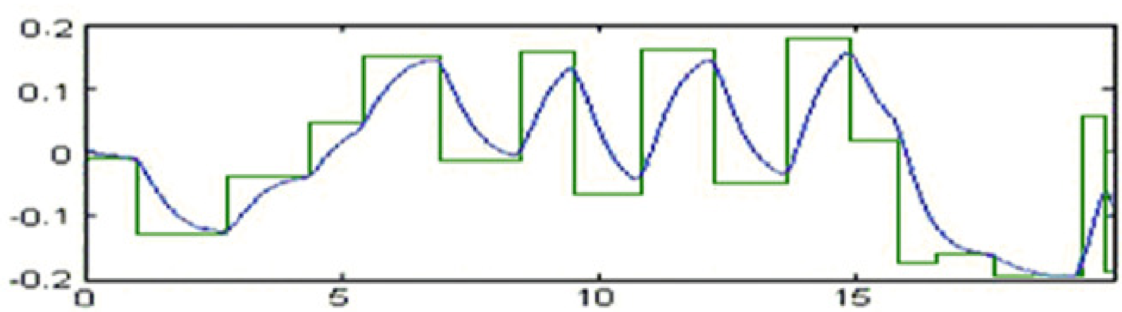

FIGURE 13: controled plant's refrence tracking profile in axis Y neutralizing NARMA-L2 control method.

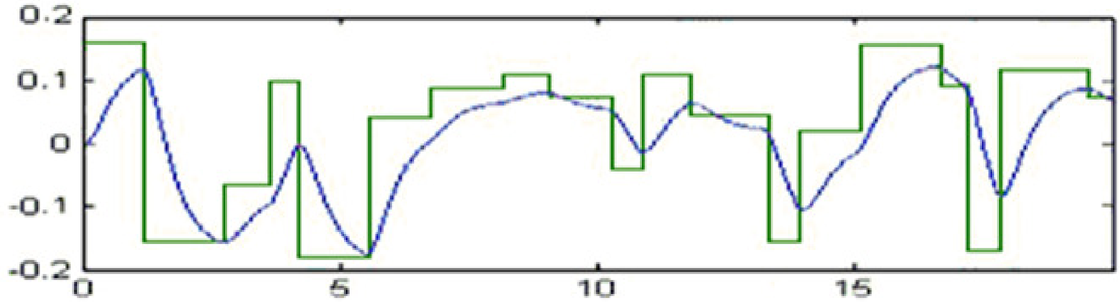

FIGURE 14: controled plant's refrence tracking profile in axis Z neutralizing NARMA-L2 control method.

4. CONCLUSION

This new approach for the robot control porpuse suffers from the extra ordiniary training properties, all neural networks based methods have a training phase but in this particular case training is a little more prolix and needs more efforts. There is also several concession point, like profable trajectory behaviore and not a very coplicated designing, then again it could be said that the proposed control strategy may eliminate the necessarily of complex driver boards for the actuator's control [12]; most importantly because all dynamic quandaries have the disturbance role in control loop but they could be included during the training procces, like manipulating a load with the robotic arm in addition of a gripper, the convinsing PWM shaped output provided by NARMA-L2 controlers for DC actuators are sufficient evidence for this claim. But after all it seems that the most important properties of this input trajectory system is that this control method has forced a similar output for step inputs in all manners, this is not guaranteed that a nonlinear system shows such behaviore (nonlinear systems respond is extremly related to their initial conditions, chaos theorem) but now with a suitable control method and training strategy it seems that the nonlinear multi input-output systems respond to step input similarily during simulation, now the way is clear to identify the whole system as three linear second degree systems and present a conventional PID controller for each input in order to further improvement in robotic arm's refrence trajectory profile.
Yet the advantages of this control methods should be counted as follow:

1- elimination of relativly expensive robot actuator's driver board's necessity.

2- the explained extremly nonlinear system has been seen showing a behaviore like simple identifiable linear system.

3- the whole control method should need less hardware equipments compare to those are necessary to establish complicated nonlineare controlers.

4- the utilized neural networks based control method has a discrete nature which is most desirable when it comes to implanting the system on digital chipsets.

5- while simulation some times errors accured when the robotic arm's configoration approach singulare configuration, but situations got handeled by reducing the simulation's time step, it seems the NARMA-L2 controlers can handle the conditions; since this control system is designed for input signal which is generated in wrist trajectory and hand operation stimulation (speed and position in 3d) so in practice such situation should never accure becouse human arm's operation space is limited and it doesn't reach singular configuration which happen in arm's far reaches.

Further research and expansion outlines should be to investigate that if further changes in mechanical properties of system (like grasping an object) could be handles by adjusting a PID controler to whole now linearized system (or the training may be resumed),as the main fallback in neural networks based control methods is the fact that with any change in system trait obliges the training procedure repeated (yet it is worse in nonlinear controlers becouse whole design must be changed) , and to investigate how much training effort is required in practice.

5. REFFRENCES

[1] D.Beldekas. *"Comparative Control of A Nonlinear First Order Velocity System By A Neural Network NARMA Method".* Issn 1392 – 1215 Elektronika Ir Elektrotechnika.2004. Nr. 6(55):1-4, 2004.

[2] R.Bishop. *"Mechatronics",* Taylor & Francis Group, PP. 129-136 (2006).

[3] C.Cox. *"Modern Adaptive Control with Neural Networks".* International Conference on Neural Networks Information Processing, Iconip 96, Hong Kong, 1996.

[4] M.Haggan. *"neural networks design",* PWS publishing company, PP.391-399 (1996).

[5] S.Hawkins. *" Neural Networks",* Prentice Hall, PP. 130-170 (1999).

[6] Hen Yu, Hen. *"Handbook of Neural Network Signal Processing",* CRC Press, PP. 34-39, 42-71, 249-267 (2002).

[7] M.Koot. *"Identification And Control Of The RRR-Robot".* M.Sc thesis, Eindhoven University Of Technology,Serbia, 2001.

[8] R.Lippmann. *" An Introduction To Computing With Neural Nets".* IEEE ASSP Magazine: 4-22, 1987.

[9] L.Guoping. *"Variable Neural Networks for Adaptive Control Of Nonlinear Systems"* IEEE Transactions On Systems, Man, And Cybernetics—Part C: Applications And Reviews: 1-3, 1999.

[10] L.Marton. *"Robust-Adaptive Control of Nonlinear Single variable Mechatronic System".* PhD thesis, Budapest University of Technology And Economics, 2006.

[11] H. Mokri. *"Real Time Implementation of NARMA Control of a Single Link Manipulator".* American Journal Of Applied Sciences 5 (12): 1642-1649, 2008.

[12] G.Moleykutty. *"Speed Control of Separately Excited Dc Motor".* PhD Thesis, Multimedia University Melaka Campus, 75450 Melaka, Malaysia, 2008.

[13] Narendra. *"Identification And Control Of Dynamical Systems Using Neural Networks".* IEEE Transaction of Neural Networks: 4-27, 1990.

[14] H.Nijmeijer. *"Nonlinear Dynamic Control Systems".* Springer, PP.130-160 (1990)

[15] Pandian, Shunmugham. *"Model-Based Sliding Mode Control of Underwater Robot Manipulators".* International Journal of Offshore and Polar Engineering (ISSN 1053-5381) Vol. 16, No. 3, September 2006: PP. 210–217, 2006.

[16] M.Spong. *"Robot Dynamic And Control",* Springer, PP: 61-83,99-153,175-193,231-300 (2004).

[17] G.Van Der Zalm. *"Discrete-Time Sliding Mode Control of the RRR-Robot"* PhD Thesis, Eindhoven University Of Technology, 2003.