Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 8 (21): 3783-3794, 2008
ISSN 1812-5654

Nonlinear Generalized Predictive Controller Based on Artificial Neural Network for Robot Control

[1]B. Durmus and [2]N. Yumusak
[1]Department of Electric-Electronic Engineering, Sakarya University, 54187 Adapazarı, Turkey
[2]Department of Computer Engineering, Sakarya University, 54187 Adapazarı, Turkey

Abstract: This study deals with the tracking control problem of a robotic manipulator with changing dynamics. A multiple-input multiple-output (MIMO) artificial neural network based generalized predictive control (NGPC) controller was designed for a six-degrees-of-freedom (6-DOF) robotic manipulator random disturbances and changing load. A three-layered neural network was used in the controller design to predict robotic manipulator inputs which track a desired trajectory. Standard back propagation (BP) algorithm was used as a learning algorithm to minimize the difference between actual trajectory and that predicted by the neural network. NGPC controller was compared the conventional GPC under different control conditions. Results show that proposed control improved the ability of GPC under uncertainties.

Key words: Robotic manipulator control, trajectory planning, generalized predictive control, artificial neural network

INTRODUCTION

Evaluation of robot actuator inputs to track a desired trajectory is one of the most basic and important problems in robotic (Sun *et al*., 2001; Behera *et al*., 1994). A robotic manipulator is highly nonlinear system because of interactions between its links. These interactions make it hard to control. Several conventional methods such as adaptive control, proportional-plus-integral-plus-derivative (PID) control, self-tuning control, self-tuning PID control and generalized predictive control have been used in robot control (Chen *et al*., 1999; Malki *et al*., 1997; Vega and Prada, 1991; Alonge *et al*., 2003; Nasisi and Carelli, 2003; Kanev and Verhaegen, 2000). However, because of high level interactions and nonlinear dynamics, industrial manipulators which use conventional linear control systems cannot be used over certain velocity limits and the productivity is limited. Furthermore, increasing performance needs also require improved manipulator techniques. Therefore, it is necessary to apply some advanced control techniques to provide high quality tracking control. One of these control techniques is generalized predictive control (GPC).

GPC is based on the use of a model which includes the prediction of the future outputs over a certain horizon (Clarke *et al*., 1987a, b). Thus, it can predict future changes of the measurement signal and base control actions on this prediction. GPC belongs to the class of model-based predictive control (MPC) techniques has become popular over the past two decades as a powerful tool for solving many problems. Today, this technique is still used widely because of their ability to systematically take into account real plant constraints in real-time (Mahfouf *et al*., 1997; Bordons and Camcho, 1998; Normey-Rico and Camcho, 2000; Zhang *et al*., 2004).

On the other hand, the conventional GPC algorithms use linear models of the process to predict the output of the process over a certain horizon and to evaluate a future sequence of control signals in order to minimize a certain cost function that represents the future output prediction errors over a reference trajectory (Clarke *et al*., 1987a, b). However, if the process is nonlinear, use of linear models becomes impractical and the identification of nonlinear models for control becomes necessary.

In recent years, the use of neural networks (NNs) for nonlinear system identification has proved to be extremely successful (Huang and Lewis, 2003; Zamarrefio and Vega, 1999). NNs have been shown to possess good function approximation capabilities and have been applied successfully by many researchers in modeling some poorly understood systems or processes. The results demonstrated the feasibility of identification and control of nonlinear dynamic systems (Tsai *et al*., 2002; Lu and Tsai, 2004). Therefore, artificial neural network (ANN) was used in the predictive controller design to improve the ability of GPC.

Corresponding Author: Burhanettin Durmus, Department of Electric-Electronic Engineering, Sakarya University, 54187 Adapazarı, Turkey Tel: (+90) 264 295 58 28 Fax: (+90) 264 295 56 08

NN based controller has been proposed by many researchers (Palos *et al.*, 2001; Laabidi *et al.*, 2008). Gupta and Sinha (2000) have presented an intelligent control system using a PD controller and ANN. Takahashi and Yamada (1994) have presented a study based on designing a NN controller, to control the tip of the angular position of a single-link flexible arm. Yildirim (2004) has proposed an adaptive robust neural controller for two-degrees-of-freedom robotic manipulator. Proposed neural controller has been shown to perform better than the conventional control schemes. Temurtas *et al.* (2005) have presented a controller ANN based on GPC for three joint robotic manipulator. Their study can be cited as closest to the work undertaken in this paper. In their study, three independent single-input single-output (SISO) controllers were used for control of joints and mechanical vibrations in the links were neglected. In practical robot applications, mechanic vibrations in links must not be neglected since they make hard to robot control. On the other hand, if a number of controllers are used in control design; many more hardware equipments will be required. Therefore, this paper focuses on a multiple-input multiple-output (MIMO) controller design. In this study, MIMO NGPC controller designed for 6-DOF robotic manipulator with random disturbances and load effect. Conventional GPC was also used for comparison. The controllers generate joint torques that will cause the robotic manipulator to follow a desired trajectory for a given trajectory. In the GPC, linear model was used to predict robot actuator inputs whereas prediction was carried out over a three-layered NN in the NGPC. The curves of trajectory and velocity belonging to joints were examined and end-effecter position errors were computed for different control status. The results obtained by using NGPC were compared with those of conventional GPC.

GENERALIZED PREDICTIVE CONTROL

Generalized Predictive Control (GPC) belongs to the class of digital control methods called Model-Based Predictive control (MBPC) and was first introduced by Clarke and his co-workers in 1987 (Clarke *et al.*, 1987a, b). MBPC techniques have been analyzed and implemented successfully in process control industries since the end of the 1970's and continue to be used because they can systematically take into account real plant constraints in real-time.

The GPC system for the robot control is given in Fig. 1. It consists of three components, the robotic manipulator, controller and parameter estimator.

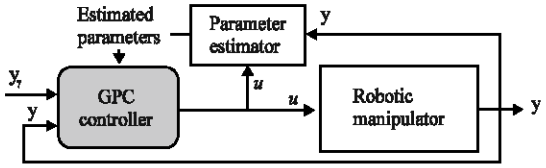

Fig. 1: Block diagram of the GPC system for robotic manipulator

In the GPC algorithm, the process is supposed to be represented by a CARIMA (Controlled Auto-Regressive Integrated Moving Average) model (Clarke *et al.*, 1987a, b):

$$A(q^{-1})\, y(t) = B(q^{-1})\, u(t-1) + \xi(t)/\Delta \tag{1}$$

where, y(t) denotes the output of the process, u(t) denotes its input and $\xi(t)$ denotes an uncorrelated random noise.

A (q^{-1}) and B (q^{-1}) are polynomials of q^{-1}, the backward shift operator:

$$A(q^{-1}) = I_m + \sum_{j=1}^{n_a} a_j\, q^{-j} \tag{2}$$

$$B(q^{-1}) = \sum_{j=0}^{n_b} b_j\, q^{-j} \tag{3}$$

$$\Delta = 1 - q^{-1}$$

In the prediction process, future outputs are predicted by using past inputs and past outputs. Predict step is selected short since using a great number of elements for prediction causes high computation overhead. In this study, the prediction was made for future three outputs. Therefore, n_a, n_b are selected as 2, 1 respectively. Initial values were used as follows; $a_1 = a_2 = 0$, $b_1 = 1$, $b_0 = 0$.

The prediction process is executed as follows:
We rewrite equation 1;

$$A(q^{-1})\Delta\, y(t) = B(q^{-1})\Delta\, u(t-1) + \xi(t) \tag{4}$$

$$\begin{aligned} y(t) &= (1-a_1)y(t-1) + (a_1-a_2)y(t-2) + a_2 y(t-3) \\ &\quad + b_0\Delta u(t-1) + b_1\Delta u(t-2) + \xi(t) \end{aligned} \tag{5}$$

The first step predicted output ($\hat{y}(t)$);

$$\begin{aligned} \hat{y}(t+1) &= (1-a_1)y(t) + (a_1-a_2)y(t-1) \\ &\quad + a_2 y(t-2) + b_0\Delta u(t) + b_1\Delta u(t-1) \end{aligned} \tag{6}$$

The second predicted output $\hat{y}(t+2)$;

$$\begin{aligned}\hat{y}(t+2) &= (1-a_1-a_2+a_1^2)y(t)+a_1(1-a_1+a_2)y(t-1)\\ &+a_2(1-a_1)y(t-2)+[b_0(1-a_1)+b_1]\Delta u(t)\\ &+b_1(1-a_1)\Delta u(t-1)\end{aligned} \quad (7)$$

and the third predicted output: $\hat{y}(t+3)$;

$$\begin{aligned}\hat{y}(t+3) &= [1-a_1-a_2+a_1^2+a_1(2a_2-a_1^2)]y(t)\\ &+[a_1(1-a_1-a_2+a_1^2)+a_2(a_2-a_1^2)]y(t-1)\\ &+a_2(1-a_1-a_2+a_1^2)y(t-2)+[b_0(1-a_1-a_2+a_1^2)\\ &+b_1(1-a_1)]\Delta u(t)+b_1(1-a_1-a_2+a_1^2)\Delta u(t-1)\end{aligned} \quad (8)$$

A (q^{-1}) and B (q^{-1}) parameters in equation (1) should be updated each step of control. For this process, a parameter estimator is needed to update. In our application, Recursive Least Square (RLS) was used to parameter estimation and detailed computational issues of the RLS are addressed in Durmus *et al.* (2008).

The GPC strategy minimizes a weighted sum of square predicted future errors and square control signal increments; the cost function for MIMO GPC is defined as follows:

$$\begin{aligned}J(N_1,N_2,N_u) &= \sum_{j=N_1}^{N_2}\left(\hat{y}(t+j)-y_r(t+j)\right)^T\left(\hat{y}(t+j)-y_r(t+j)\right)\\ &+\sum_{j=1}^{N_u}\left(\Delta u(t+j-1)\right)^T\Lambda(j)\left(\Delta u(t+j-1)\right)\end{aligned} \quad (9)$$

where, N_1 is the minimum costing horizon, N_2 is the maximum costing horizon, N_u is the control horizon, y is a predicted output, y_r is the reference output. λ is the control input weighting factor and it is selected very small (as $\lambda \approx 10^{-6}$). Detailed computational issues of the GPC are addressed in Clarke *et al.* (1987a, b).

NONLINEAR GENERALIZED PREDICTIVE CONTROL FOR ROBOTIC MANIPULATORS

The nonlinear generalized predictive control (NGPC) system for the robotic manipulator is given in Fig. 2. It consists of four components, the robotic manipulator, a tracking reference signal that specifies the desired trajectory of the manipulator, an artificial neural network for prediction and the Cost Function Minimization (CFM) algorithm that determines the input needed to produce the desired trajectory of the manipulator.

The NGPC algorithm operates in two modes, prediction and control. For realizing this aim, a double pole double throw switch is used. The CFM algorithm produces an output which is either used as an input to the robotic manipulator or the manipulator's neural network

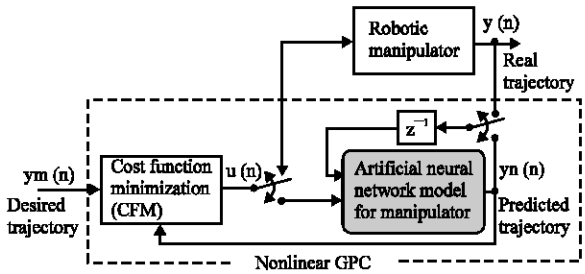

Fig. 2: Block diagram of the nonlinear GPC system

model. The switch position is set to the robotic manipulator when the CFM algorithm has solved for the best input, u(n), that will minimize a specified cost function. Between samples, the switching position is set to the manipulator's neural network model where the CFM algorithm uses this model to compute the next control input, u(n+1), from predictions of the response from the manipulator's model. Once the cost function is minimized, this input is passed to the manipulator (Durmus *et al.*, 2008).

The cost function in the NGPC is defined as following equation:

$$\begin{aligned}J(N_1,N_2,N_u) &= \sum_{j=N_1}^{N_2}\left(y_r(n+j)-yn\ (n+j)\right)^2+\sum_{j=1}^{N_u}\lambda(j)\left(\Delta u(n+j)\right)^2\\ &+\sum_{j=1}^{N_u}\left(\frac{s}{u(n+j)-u_{min}+\varepsilon}+\frac{s}{u_{max}-u(n+j)+\varepsilon}\right)\end{aligned} \quad (10)$$

Where:

j = 1,2,….#iterations
N_1 = The minimum costing horizon
N_2 = The maximum costing horizon
Nu = The control horizon
y_r (n) = A reference trajectory
y_n = The predicted output of the neural network
λ = The control input weighting factor, it is selected very small (as $\lambda \approx 10^{-6}$)
Δu (n+j) = The change in u and is defined as u (n+j)-u (n+j-1)
u_{max} and u_{min} = Maximum and minimum control inputs s is the sharpness of the corners of the constraint function, $\varepsilon = 10^{-7}$

The minimization algorithm used in the NGPC is Newton-Raphson method because it requires less iteration numbers for convergence. Detailed computational issues of the Newton-Raphson method are addressed in Durmus *et al.* (2008).

Robotic manipulator neural network model for NGPC: A multi-layer feed-forward network (MLFN) with tapped time delays was used for the robotic manipulator NN model.

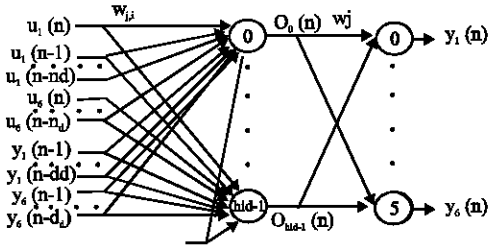

Fig. 3: Multi-layer feed-forward neural network model with a time delayed structure

The network structure is given in Fig. 3. The torque, υ, is the control input to the manipulator system and the trajectory, y is the output. The network inputs are past values of the torque and the past values of the manipulator's trajectory. The network has a single hidden layer with multiple hidden layer nodes and a single output node.

The number of neurons in the input and output layers was decided according to the structure of GPC. The past 2 inputs and past 3 outputs were used for prediction since over past data using leads to over computation. In this state, n_d, d_d and hid were selected as 2, 3 and 5 respectively. Consequently, the designed neural network has 36 inputs and 6 outputs.

This network is multilayer network (input layer, hidden layer and output layer). Whereas the hidden layer neurons use tag-sigmoid activation functions, linear activation function is used for output layer neurons. Equations used in the neural network models are given in Eq. 11-13.

Outputs of hidden layer neurons are:

$$net_j(n) = b_j + \sum_{i=0}^{n_d} w_{j,i}u(n-i) + \sum_{i=1}^{d_d} w_{j,i+n_d}y(n-i) \quad (11)$$

$$O_j(n) = f\left(net_j(n)\right) = \frac{e^{net_j(n)} - e^{-net_j(n)}}{e^{net_j(n)} + e^{-net_j(n)}} = 1 - \frac{2}{1+e^{2\cdot net_j(n)}} \quad (12)$$

Linear activation function outputs are:

$$yn(n) = b + \sum_{j=0}^{hid-1} w_j O_j(n) \quad (13)$$

where, j = 0 to hid-1and hid is the number of hidden layer nodes, net_j (n) is the activation level of the j^{th} hidden node, f (·) is the activation function for the hidden layer nodes, n_d is the number of input nodes associated with past u (n), d_d is the number of input nodes associated with past y (n), w_j is the weight connecting the j^{th} hidden node to the output node, $w_{j,i}$ is the weight connecting the i^{th} input node to the j^{th} hidden node, y_n (n-i) is the delayed output of the manipulator's joint used as an input to the network and u (n-i) is the input to the network and it's delays.

A training set was prepared by using the results of conventional GPC. The robotic manipulator was controlled for different trajectories to generate the training and test sets. To obtain the torque value at time t as a output, values of torque at time (t-1), (t-2), values of trajectory at time (t-1), (t-2), (t-3) and values of reference trajectory at time (t-1) were used in the input stage as 36 elements. These data have been generated using GPC controller for different trajectories selected uniformly. Back propagation (BP) was used for training neural network. 20, 000 input and output vector sets were generated with GPC algorithm, using the robotic manipulator simulation software. The training process was completed in approximately 2, 000, 000 iterations.

ROBOTIC MANIPULATOR MODEL

Used the robotic manipulator (6-DOF) is shown in Fig. 4 (Tarn *et al.*, 1993). The dynamic model of robot includes kinematics equations, friction effects and effect of carrying load at the end-effecter for given robot.

There are six interactive second-order nonlinear differential equations which give the dynamic behaviors of 6-DOF robotic manipulator. The fourth-order Runge-Kutta integration method was used to solve these differential equations. The motion equations of robotic manipulator were developed by Lagrange-Euler as follows (Spong and Vidyasagar, 1989).

$$D(\theta)\ddot{\theta} + H(\theta,\dot{\theta}) + G(\theta) + \tau_f(t) + \tau_l(t) = \tau(t) \quad (14)$$

where, $\ddot{\theta}$, $\dot{\theta}$ and θ are n-dimensional vectors, indicating the joint acceleration, velocity and position, respectively. D (θ) is a n-dimensional symmetric inertial matrix. $H(\theta,\dot{\theta})$ is a n-dimensional vector, represents Coriolis and centrifugal torques. G (θ) is a n-dimensional vector, represents torque due to gravity. τ_f (t) is n-dimensional vector, represents torque due to friction effects. τ_l (t) is load effect, τ (t) is, n-dimensional, the generalized torque vector.

Robot arm friction model includes static, kinetic and fluid friction used in this study (Rodrigo *et al.*, 2002). The friction model is given following:

$$\tau_{friction} = f_s\left(\frac{sgn(\dot{q})}{1+\left(\frac{\dot{q}}{x_s}\right)^2}\right) + f_k\tanh(\dot{q}) + k_{vn}(\dot{q}) \quad (15)$$

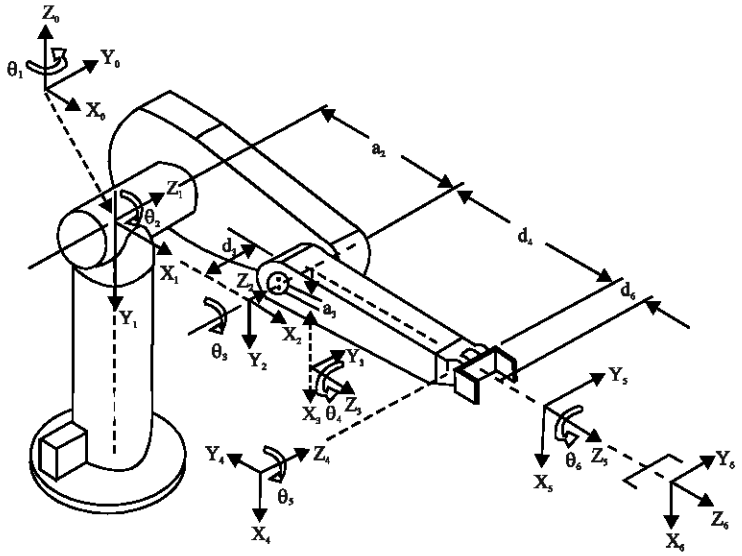

Fig. 4: Configuration of the 6-DOF robotic manipulator

where, f_x, x_s, f_k and k_{vn} are static friction, constant to correct static friction due to Stribeck Effect, kinetic friction and fluid friction, respectively (Rodrigo *et al*., 2002).

In this study, position reference and velocity reference trajectory for each joint were determined according to the sinusoidal trajectory principle. The motion for the sinusoidal trajectory is given in Fig. 5.

The equations motions for the sinusoidal trajectory (Spong and Vidyasagar, 1989) are,

$$\theta(t) = a + bCos(wt) \tag{16}$$

$$\dot{\theta}(t) = -bwSin(wt) \tag{17}$$

$$\ddot{\theta}(t) = -bw^2 Cos(wt) \tag{18}$$

$$w = \frac{\pi}{t_f}a = \frac{\theta_f + \theta_0}{2}b = -\left(\frac{\theta_f - \theta_0}{2}\right)$$

$$\theta[i] = \left(\frac{\theta_f + \theta_0}{2}\right) - \left(\frac{\theta_f - \theta_0}{2}\right)Cos\left(\frac{\pi i}{n}\right) \quad ,i = 0\cdots n \tag{19}$$

(n = 6 for 6-DOF robotic manipulator)

$$\dot{\theta}[i] = \left(\frac{\theta_f - \theta_0}{2}\right)\left(\frac{\pi}{t_f}\right)Sin\left(\frac{\pi i}{n}\right) \tag{20}$$

$$\ddot{\theta}[i] = \left(\frac{\theta_f - \theta_0}{2}\right)\left(\frac{\pi}{t_f}\right)^2 Cos\left(\frac{\pi i}{n}\right) \tag{21}$$

where, θ_i (t) is the angular position of the joint i at time t, θ_{i0} is the initial angular position of the joint i at time t_0 and θ_{if} is the final angular position of the joint i at time t_f.

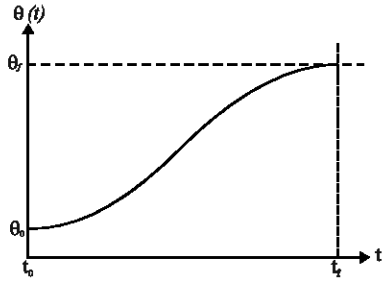

Fig. 5: The motion trajectory planning

The coordinate of end-effecter for robot arm is computed as follows:

$$x = \begin{bmatrix} d_6\left(\cos(\theta_2+\theta_3)\cos\theta_4\sin\theta_5 + \sin(\theta_3+\theta_3)\cos\theta_5\right) \\ +d_4\sin(\theta_2+\theta_3) + a_3\cos(\theta_2+\theta_3) + a_2\cos\theta_2 \end{bmatrix} \\ *\cos\theta_1 - \left(d_2 + d_6\sin\theta_4\sin\theta_5\right)\sin\theta_1 \tag{22}$$

$$y = \begin{bmatrix} d_6\left(\cos(\theta_2+\theta_3)\cos\theta_4\sin\theta_5 + \sin(\theta_3+\theta_3)\cos\theta_5\right) \\ +d_4\sin(\theta_2+\theta_3) + a_3\cos(\theta_2+\theta_3) + a_2\cos\theta_2 \end{bmatrix} \\ *\sin\theta_1 + \left(d_2 + d_6\sin\theta_4\sin\theta_5\right)\cos\theta_1 \tag{23}$$

$$z = d_6\left(\cos(\theta_2+\theta_3)\cos\theta_5 - \sin(\theta_2+\theta_3)\cos\theta_4\sin\theta_5\right) \\ + d_4\cos(\theta_2+\theta_3) - a_3\sin(\theta_2+\theta_3) - a_2\sin\theta_2 \tag{24}$$

THE DESIGN OF CONTROLLER USING NGPC FOR 6-DOF ROBOTIC MANIPULATOR CONTROL

Here, the design of the NGPC based on ANN controller is given. The designed controller has 36

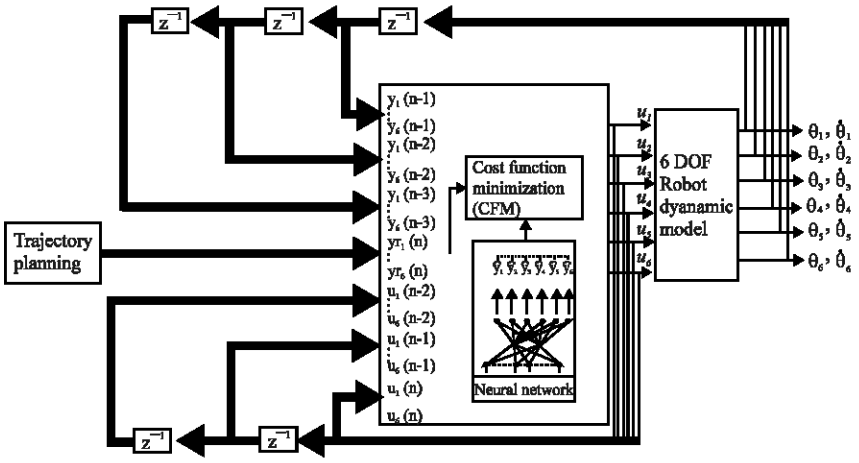

Fig. 6: Block diagram of artificial neural network implemented generalized predictive controller

inputs and 6 outputs. To obtain the torque value at time t as a output, values of torque at time (t-1), (t-2), values of trajectory at time (t-1), (t-2), (t-3) and values of reference trajectory at time (t-1) are used in the input stage as 36 elements for 6-DOF robotic manipulator. Outputs are the torques which will be applied to joints to track the desired trajectories. These data also have been used for prediction and learning of artificial neural network in the NGPC. The block diagram of the control system is shown in Fig. 6.

The NGPC starts with the input signal, θ_i (entered angle of joints), which is presented to the trajectory planning model. This model produces a tracking reference signal, $y_r(n)$, which is used as an input to the CFM block. The CFM block produces an output which is both used as an input to the plant (robotic manipulator) and to neural network for prediction. The neural network serves to predict the plant outputs from N_1 to N_2 future time steps in Eq. 10 by using past data. The predicted outputs passed to CFM. The CFM minimizes cost function that represents errors between reference signal and predicted outputs of the plant until desired minimization is achieved. Once the cost function is minimized, this input is passed to the manipulator. This process repeats for each control step.

For comparison of control algorithms, the following control states and values were used as a scenario of control.

The control states used in this study:

- There were carrying load and friction effects in the control simulation
- There were falling load and friction effects in the control simulation
- There were disturbance (between -0.5 and +0.5 Newton*meters), falling load and friction effects in the control simulation

The control values used in the simulation:

- The total simulation time is 10 second and total step number is 10000
- The end-effecter of robotic manipulator caries 5 kg load in the state of carrying and falling load. And the load is falling at the 4000th control step (at 4th second) in the state of falling load
- Random disturbances between -0.5 and +0.5 Nm were added to the torques computed at the end of each step for control states

RESULTS AND DISCUSSION

Some sample control results of the 6-DOF robotic manipulator which uses GPC algorithm at the state of A were given in Fig. 7. And some sample control results of the 6-DOF robotic manipulator which uses NGPC algorithm at the state of A were also given in Fig. 8. The same trajectory is used for both methods. The start and final angles of the joints are given in Table 1.

There are errors at the beginning of motion results from inertia of motionless robot arm (Fig. 7, 8). However, angular velocity and torque errors are bigger in the GPC than those in the NGPC. This can be because the neural network improves the predicted trajectory at the beginning of the motion. So, motion of the manipulator is more smooth and flexible in the NGPC.

There were similar situations at the load changes. In the Fig. 9 and 10, angular velocity and torque errors are seen at time falling load (at 4th second) result from load change. If the parameters of the system change in the control process, the performance of control system is

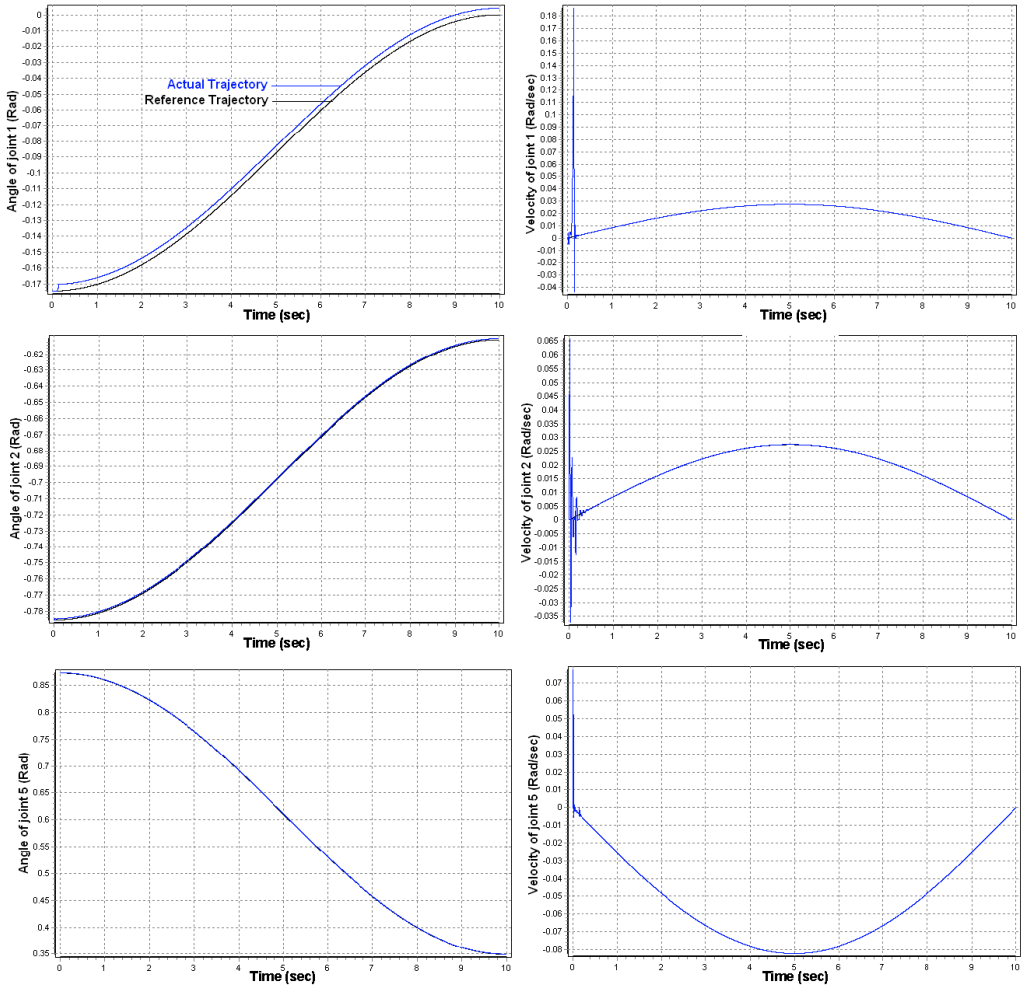

Fig. 7: Some angle and velocity graphs of joints using GPC (State A: with load and with friction)

Table 1: Start and final angles of joints

Joints	Start angle (rad)	Final angle (rad)
1	-0.174532925199 (-10°)	0.000000000000 (0°)
2	-0.785398163397 (-45°)	-0.610865238198 (-35°)
3	0.698131700798 (40°)	1.047197551197 (60°)
4	-0.698131700798 (-40°)	0.349065850399 (20°)
5	0.872664625997 (50°)	0.349065850399 (20°)
6	0.000000000000 (0°)	1.396263401595 (80°)

hardly affected. The errors were bigger in the GPC than those of the artificial neural network implemented NGPC. It is shown that the influence of load change to the NGPC is less than that of the GPC. This means that the NGPC is much stable than GPC for the load changes. It was seen that the proposed controller has better robustness in resisting against the changes of the parameters of the control system. The neural network provided quick adaptation for NGPC whereas GPC could not adapt quickly. There were also similar situations with random disturbances (Fig. 11, 12).

Comparisons of the GPC and NGPC algorithms control results are summarized and demonstrated in Table 2 and 3 by using angle location errors and x, y, z axis errors of end-effecter for all of the control status. End-effecter coordinates of robotic manipulator were computed using Eq. 22-24.

Angle location errors and x, y, z axis errors of the end-effecter are smaller in the NGPC than those in the GPC (Table 2, 3). The difference between control results for the carrying load, with random disturbances and the falling load states can be easily shown. These values for the

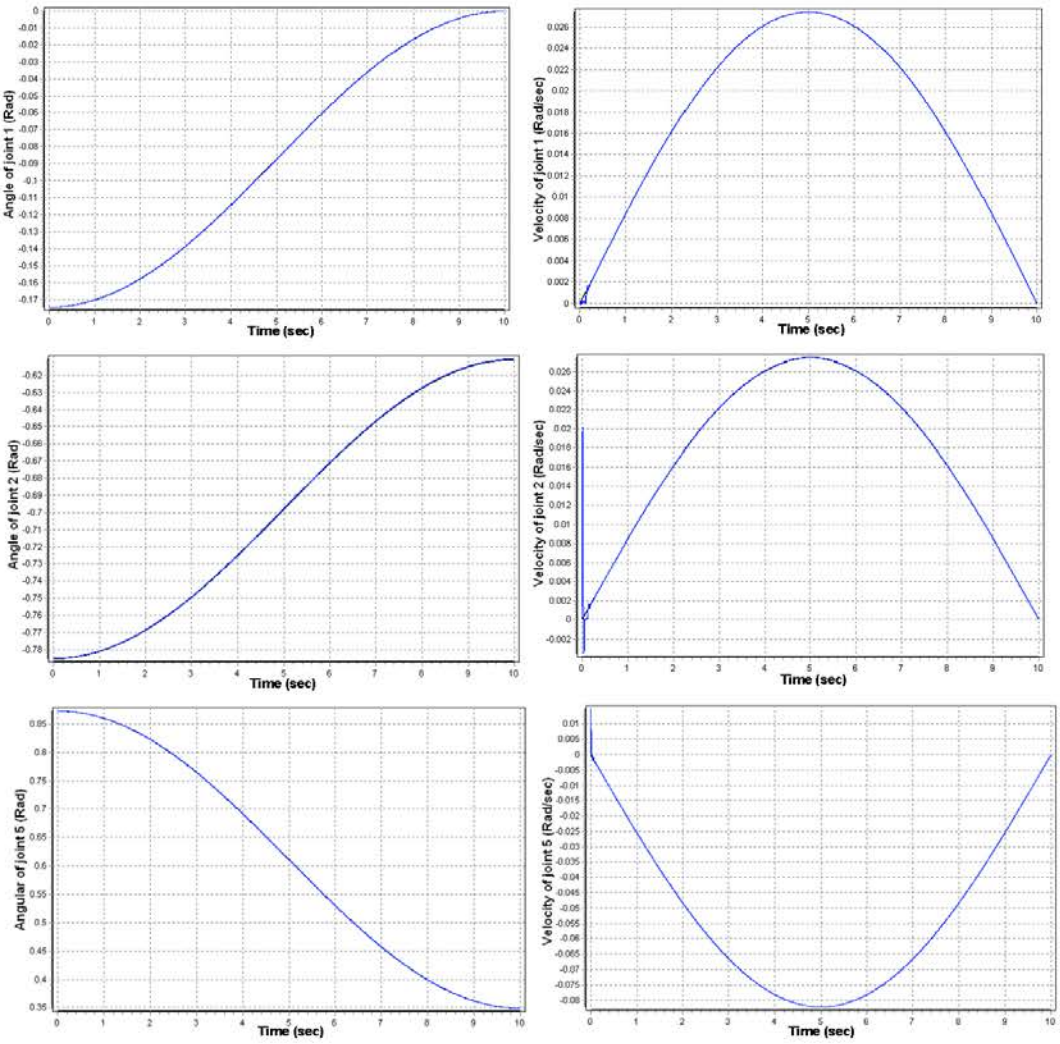

Fig. 8: Some angle and velocity graphs of joints using neural network implemented NGPC (State A: with load and with friction)

Table 2: Angle location errors of control algorithms for control status

		Control status (rad)		
Joint	Controller	A	B	C
1	GPC	0.004173337500	0.004173351978	0.000173516747
	NGPC	-0.000062396165	-0.000062553161	0.000129773015
2	GPC	0.001014069086	0.000761165639	0.006195076845
	NGPC	0.000404175469	0.000208797211	0.000588918097
3	GPC	-0.000371099469	-0.000375978325	-0.000380519649
	NGPC	0.000162112323	0.000018390058	0.000036596777
4	GPC	-0.000242959708	-0.000244147328	0.000402917819
	NGPC	-0.000024398726	-0.000016631267	-0.000476464764
5	GPC	0.000392873002	0.000409310388	0.000189147815
	NGPC	0.000131973926	0.000017202783	0.000720890061
6	GPC	-0.000013101317	-0.000013101317	0.000055395398
	NGPC	-0.000013209944	-0.000013209944	-0.001354191916

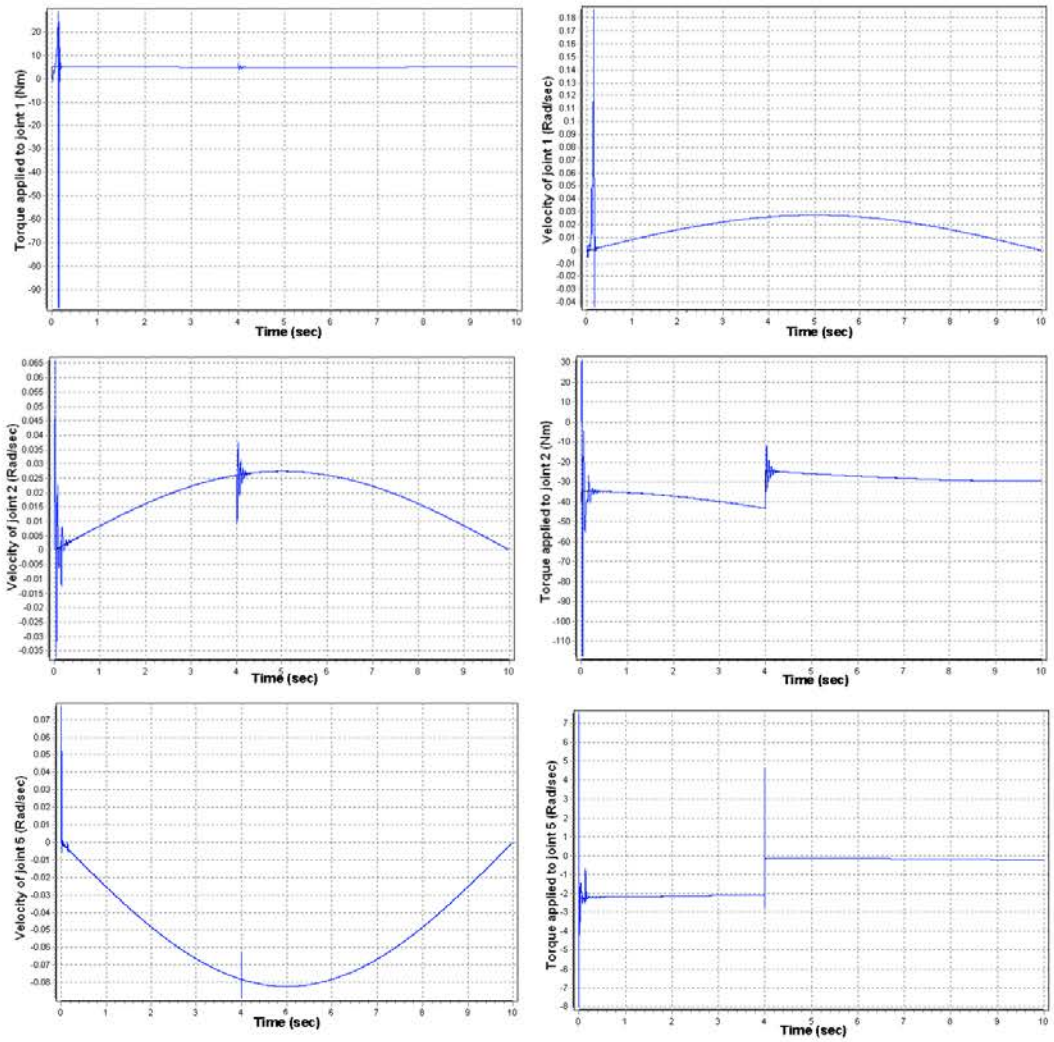

Fig. 9: Some torque and velocity graphs of joints using GPC (State B: with friction and falling load)

Table 3: The end-effecter axis errors for different control status

		End-effecter coordinate errors (mm)		
Control state	Controllers	x	y	z
A	GPC	-0.103896501760	2.328357064059	-0.505426604190
	NGPC	0.364692902864	-0.032835056385	-0.263337821119
B	GPC	-0.279583222659	2.327907356465	-0.364074534378
	NGPC	0.162438537309	-0.034844165537	-0.120771968326
C	GPC	4.067840656838	0.108070719422	-3.391902406906
	NGPC	0.431026191143	0.076745334388	-0.364679197600

NGPC are less than that of the GPC. NGPC reached at the targeted point with less error.

In this study, GPC and NGPC algorithms were designed and applied to 6-DOF robotic manipulator for joint control. Nonlinear prediction model was proposed instead of linear model to improve the ability of GPC. Conventional GPC and NGPC algorithms were compared according to different scenarios of control. When the parameters of the system change in the control process, the performance of control system is hardly affected. It was seen that the proposed controller has better robustness in resisting against the changes of the parameters of the control system and its trajectory tracking performance was observed higher than

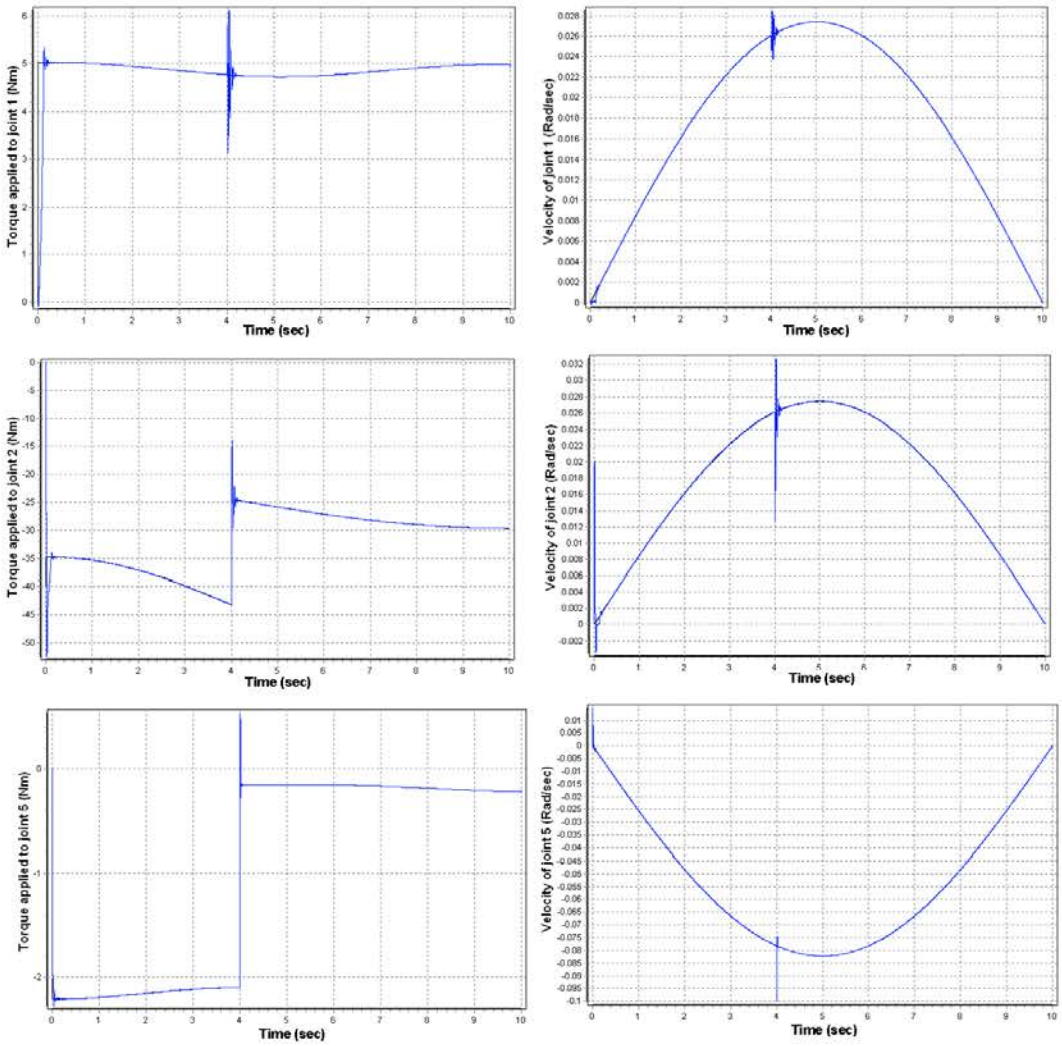

Fig. 10: Some torque and velocity graphs of joints using neural network implemented NGPC (State B: with friction and falling load)

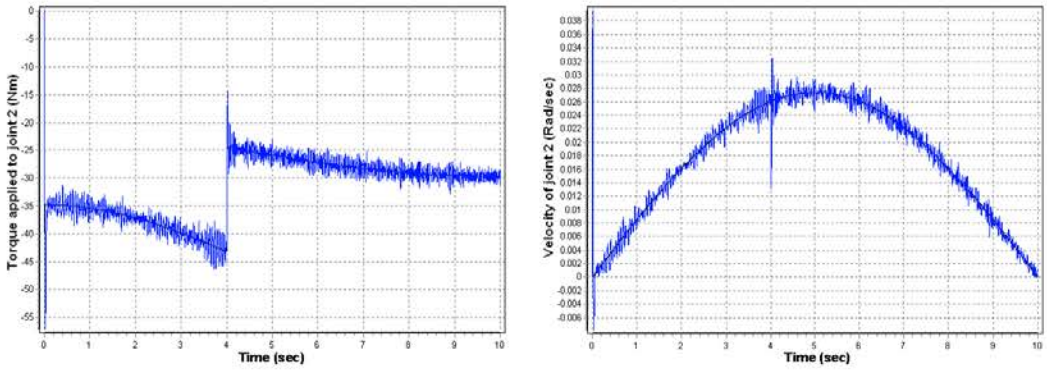

Fig. 11: Torque and velocity graphs of joint 2 using GPC (State C: with falling load and random disturbances)

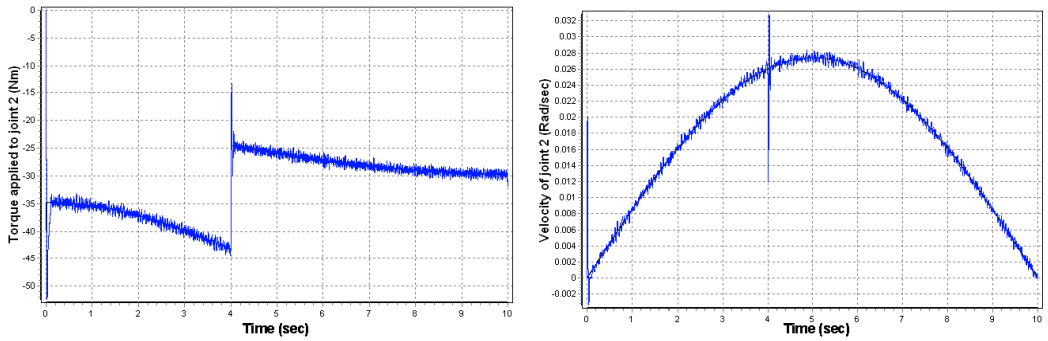

Fig. 12: Torque and velocity graphs of joint 2 using neural network implemented NGPC (State C: with falling load and random disturbances)

conventional GPC's under uncertainties. Artificial neural network improved the accuracy of predicted trajectory at the beginning of motion, load changes and with random disturbances and provided quick adaptation.

REFERENCES

Alonge, F., F. D'lppolito and F.M. Raimondi, 2003. An adaptive control law for robot manipulators without velocity feedback. Control Eng. Practice, 11: 999-1005.

Behera, L., M. Gopal and S. Chandhury, 1994. Trajectory tracking of robot manipulator using Gaussian networks. Robotics Autonomous Syst., 13: 107-115.

Bordons, C. and E.F. Camacho, 1998. A generalized predictive controller for a wide class of industrial process. IEEE Trans. Control Syst. Technol., 6: 372-387.

Chen, W.H., D.J. Balance, P.J. Gawthrop, J.J. Gribble and J. O'Reilly, 1999. Nonlinear PID predictive controller. IEE Proc. Control Theory Appl., 146: 603-611.

Clarke, D.W., C. Mohtadi and P.S. Tuffs, 1987a. Generalized predictive control-part 1: The basic algorithm. Automatica, 23: 137-148.

Clarke, D.W., C. Mohtadi and P.S. Tuffs, 1987b. Generalized predictive control-part 2: The basic algorithm. Automatica, 23: 149-163.

Durmus, B., H. Temurtas, N. Yumusak, F. Temurtas and R. Kazan, 2008. The cost function minimization for predictive control by Newton-Raphson method. Proceedings of the International MultiConference of Engineers and Computer Scientists 2008 (IMECS, 2008), March, 19-21 Hong Kong, pp: 1347-1352.

Gupta, P. and N.K. Sinha, 2000. Intelligent control of robotic manipulators: Experimental study using neural networks. Mechatronics, 10: 289-305.

Huang, J.Q. and F.L. Lewis, 2003. Neural-network predictive control for nonlinear dynamic systems with time-delay. IEEE Trans. Neural Networks, 14: 377-389.

Kanev, S. and M. Verhaegen, 2000. Controller reconfiguration for non-linear systems. Control Eng. Practice, 8: 1223-1235.

Laabidi, K., F. Bouani and M. Ksouri, 2008. Multi-criteria optimization in nonlinear predictive control. Math. Comput. Simulation, 76: 363-374.

Lu, C.H. and C.C. Tsai, 2004. Adaptive neural predictive control for industrial multivariable processes. J. Syst. Control Eng., 218: 557-567.

Mahfouf, M., D.A. Linkens and A.J. Asbury, 1997. Generalised predictive control (GPC): A powerful control tool in medicine. IEE Proc. Control Theory Appl., 144: 8-14.

Malki, H.A., D. Misir, D. Feigenspan and G. Chen, 1997. Fuzzy PID control of a flexible-joint robot arm with uncertainties from time-varying loads. IEEE Trans. Control Syst. Technol., 5: 371-378.

Nasisi, O. and R. Carelli, 2003. Adaptive servo visual robot control. Robotics Autonomous Syst., 43: 51-78.

Normey-Rico, J.E. and E.F. Camacho, 2000. Robust design of gpc for process with time delay. Int. J. Robust Nonlinear Control, 10: 1105-1127.

Palos, A.G., S. Parthasarathy and A.F. Atiya, 2001. Neural-predictive process control using on-line controller adaptation. IEEE Trans. Control Syst. Technol., 9: 741-755.

Rodrigo, J., M.H. Ang, D. Oetomo, O. Khatib, T.M. Lim and S.Y. Lim, 2002. The operational space formulation implementation to aircraft canopy polishing using a mobile manipulator. Proceedings of the International Conference on Robotics and Automation (ICRA, 2002). May 11-15, Washington DC., USA, pp: 400-405.

Spong, M.W. and M. Vidyasagar, 1989. Robot Dynamics and Control. 1st Edn. John Wiley and Sons, New York, USA, pp: 10-130. ISBN: 978-0-471-61243-8.

Sun, F.C., Z.Q. Sun and P.Y. Woo, 2001. Neural network-based adaptive controller design of robotic manipulators with an observer. IEEE Trans. Neural Networks, 12: 54-67.

Takahashi, K. and I. Yamada, 1994. Neural-network based learning control of flexible mechanism with application to a single link flexible robot arm. J. Dynam. Syst. Measur. Control, (ASME Digital Library J DSMC), 116: 792-795.

Tarn, T.J., A.K. Bejczy, G.T. Marth and A.K. Ramadorai, 1993. Performance comparison of four manipulator servo schemes. IEEE Control Syst. Magazine, 13: 22-29.

Temurtas, F., H. Temurtas and N. Yumusak, 2005. Application of neural generalized predictive control to robotic manipulators with a cubic trajectory and random disturbances. Robotics Autonomous Syst., 54: 74-83.

Tsai, P.F., J.Z. Chu, S.S. Jang and S.S. Shieh, 2002. Developing a robust model predictive control architecture through regional knowledge analysis of artificial neural networks. J. Process Control, 13: 423-435.

Vega, P. and C. Prada, 1991. Self-tunning predictive PID controller. IEE Proc. Control Theory Appl., 138: 303-311.

Yildirim, S., 2004. Adaptive robust neural controller for robots. Robotics Autonomous Syst., 46: 175-184.

Zamarrefio, J.M. and P. Vega, 1999. Neural predictive control, application to a highly non-linear system. Eng. Appl. Artif. Intel., 12: 149-158.

Zhang, J., J. Wei and T. Zhong, 2004. A generalized predictive controller to a thermal process for batch dyeing process. Proceedings of International Conference on Information Acquisition, June 21-25 China, pp: 433-435.