Robotica (2010) volume 28, pp. 1083–1093. © Cambridge University Press 2010
doi:10.1017/S0263574710000081

Determining maximum load-carrying capacity of robots using adaptive robust neural controller

M. H. Korayem*, A. Alamdari, R. Haghighi and A. H. Korayem

Robotic Research Laboratory, College of Mechanical Engineering, Iran University of Science and Technology, Tehran, Iran

(Received in Final Form: February 19, 2010. First published online: March 22, 2010)

SUMMARY

In this paper, the combination of neural network (NN), proportional derivative (PD), and robust controller are used for determining the maximum load-carrying capacity (MLCC) of articulated robots, subject to both actuator and end-effector deflection constraints. The proposed technique is then applied to articulated robots, and MLCC is obtained for a given trajectory. In the practical simulations, it's impossible to determine the parameters of robot model exactly, so the trajectory tracking performance of the proportional integral derivative (PID) and computed torque methods significantly decrease. The PD control of robot has major problem, it cannot guarantee zero steady state error. For this reason, the NN controller with PD and robust controller are used. The multilayer neural network is also used to compensate gravity and friction effects. By using Lyapunov Direct Method it is shown that the stability of closed loop system would be guaranteed, if the weights of multilayer had certain learning rules. Standard back propagation algorithm is used as a learning algorithm to update the connection weights of the NN controller. The simulation results of the proposed adaptive robust neural network (ARNN) controller are compred with sliding mode and feedback linearization methods for flexible joint robot, and compared with open loop controller for 3D industrial robot. The obtained results assured the robustness and improvement in MLCC in the presence of uncertainties in dynamic model of the robot arm and external disturbances. In fact, adaptive robust NN controller suppresses disturbances accurately and achieves very small errors between commanded and actual trajectories.

KEYWORDS: Robot; Neural network; Trajectory tracking; Maximum load, Capacity.

1. Introduction

The load-carrying capacity, productivity, and speed of robotic manipulators are limited by the deflection of the end-effector and the capability of their actuators. Increasing actuator size and power is largely self-defeating, because of increased cost and power consumption as well as increased inertia of the actuators. A more successful approach should maximize the load-carrying capacity of the robot manipulator, subject to the constraints imposed by actuator capacity and allowable end-effector deviation for a given trajectory. Thomas *et al.*[1] used only the actuator constraint to determine the dynamic load-carrying capacity (DLCC) for rigid links and joints manipulators. A new method for determining the DLCC for flexible joint manipulators, subject to both actuator and end-effector deflection constraints is introduced in ref. [2]. Korayem *et al.*[3] proposed a feedback linearization technique to minimize end-effector deflection. Korayem and Ghariblu[4] proposed a computational technique for obtaining the DLCC of mobile flexible joint robotic manipulators for a given trajectory by considering the actuator limitations. In ref. [5], based on sliding mode control (SMC), an algorithm proposed which identify maximum DLCC in the presence of disturbance and system uncertainties.

In recent years, multilayered neural networks (MLNNs) have been used in different aspects of robotics solutions, e.g. kinematics, dynamics, trajectory planning, sensing, and control. Neural networks (NNs) are capable of learning and reconstructing complex nonlinear mapping and have been widely used in identification and control of robotic manipulators by many researchers. Both stable NN-based and online adaptive control in continuous and discrete time for robots has been recently investigated. Lewis showed how the developments of intelligent control systems based on NNs have followed three main generations.[6] These third-generation intelligent controllers are adaptive controllers that converge to optimal feedback controllers for general nonlinear systems. Gupta *et al.*[7] used NNs in control of robotic manipulator and experimental results has been demonstrated that NNs provide model-free learning controllers for robotic systems. Nakanishi and Schaal[8] presented feedback error learning (FEL) from the viewpoint of adaptive control. They discussed the relationship between FEL and nonlinear adaptive control with adaptive feedback linearization. Yildirim[9] proposed the trajectory control of a robot using a new type of recurrent NN. Standard back propagation (BP) algorithm was used as a learning algorithm for this network to minimize the difference between the robot actual response and that predicted by the NN. In ref. [10] a single-input single-output (SISO) neural generalized predictive control (NGPC) was applied to a three-joint robotic manipulator with a cubic trajectory and random disturbances. The results showed also that the NGPC SISO reduced the influence of the load changes and disturbances.

* Corresponding author. E-mail: hkorayem@iust.ac.ir

https://doi.org/10.1017/S0263574710000081 Published online by Cambridge University Press

A new adaptive neural network (ANN) controller for robot trajectory tracking problem was developed and a novel and efficient training algorithm for the proposed controller was presented in ref. [11]. The training algorithm is based on updating the weights of the network each step by minimizing the quadrant tracking errors and their derivatives.

Sun *et al.*[12] presented a stable adaptive control approach based on dynamic neural networks (DNNs) for robot manipulators with unknown dynamics nonlinearities, which DNNs were used to approximate the whole robot dynamic systems. Akbas and Esin[13] investigated nonlinear control by NNs and fuzzy logic. Gaussian neuro-fuzzy variable structure (GNFVS), FEL and direct inverse modeling architecture (DIMA) were studied. In ref. [14] an adaptive control law has been designed for robot manipulators based on the use of reference velocities instead of the actual ones and feedback signals generated from the position errors. In ref. [15] two modifications were considered to overcome these problems, which are high-gain observer to estimate the joint velocities, and RBF neural network to compensate gravity and friction. In ref. [16] a neuro feedforward compensator and a PD control were used to approximate the whole nonlinearity of robot dynamic. The approximation errors of neural identification for the gravitational force and friction can be eliminated by a discontinuous switching control law. Hsu *et al.*,[17] proposed a robust intelligent tracking controller (RITC) for a class of unknown nonlinear systems. This controller was comprised of a neural controller and a robust controller.

A new online adaptive control scheme based on FEL was proposed in ref. [18] and applied to inverted pendulum balancing. This controller for balancing consists of a conventional feedback controller and a neural network feedforward controller (NNFC). In the NNFC, the feedback error signal is employed as input stimulator, instead of the usual reference signal. Purwar *et al.* [19], developed a new adaptive neuro controller for trajectory tracking for robot manipulators without velocity measurements, taking into account the actuator constraints. The controller is based on structural knowledge of the dynamics of the robot and measurements of joint positions only. The payload variation, unknown nonlinearities and torque disturbances is estimated by a Chebyshev neural network (CNN).

In this paper the combination of NN, PD controller, and robust controller are used for determining the MLCC, subject to both actuator and end-effector deflection constraints. It is organized as follows: Section 2 gives some details about the proposed NN and motion control of manipulator. In Section 3 maximum load-carrying capacity (MLCC) for a desired trajectory and algorithm of determination MLCC with NN controller are investigated. The effectiveness of the proposed control system is demonstrated by some simulations in Section 4 and it is shown that using the NN technique, the end-effector deflection is reduced and MLCC for a given trajectory is improved.

2. Motion Control of Robot Manipulators

Model-based robot control schemes require the inclusion of the dynamical model of the robot arm in the control law. Hence, it is necessary for robot dynamics to be modeled accurately. However, the dynamics consists of a set of second-order, nonlinear, and highly coupled partial differential equations. The dynamics are time variant and undergo changes such as variation in payloads, changes in the friction coefficients of joints, etc. These facts have always posed a major problem in the implementation of dynamic-based controllers. NN makes a viable tool in autonomous systems to deal with significant variations in the environment.

2.1. Architecture

In this paper in order to solve the difficult problems such as the stability in NNs control systems, attempts to connect NNs to adaptive control schemes. For given$X \in R^{N_1}$, the output of the two-layer NN is given by

$$y_i = \sigma\left[\sum_{l=1}^{L} W_{il}\sigma\left[\sum_{j=1}^{n} V_{lj}x_j + V_{l0}\right] + W_{i0}\right]; \quad i = 1, 2, \ldots, m, \tag{1}$$

where $\sigma(.)$ is activation function, V_{lj} is the input-to-output layer interconnection weights, and W_{il} is the first-to-second layer interconnection weights. V_{l0}, W_{i0} are threshold offsets.

The NN equation maybe expressed in matrix format by defining $X_{input} = [x_0\, x_1 \ldots x_n]$, $y_{output} = [y_0\, y \ldots y_n]$ and weight matrices $V = [V_{l0}]$, $W = [W_{i0}]$. Including $x_0 = 1$ as X_{input}, allows including the threshold vector then $y_{output} = W^T\sigma(V^T X_{input})$. There is linear activation function in the output layer and typical selection for $\sigma(.)$ in the hidden layer is the sigmoid function which has a nonlinear relationship as presented in the following equation:

$$\sigma(x) = \frac{2(1 - e^{-Yx})}{Y(1 + e^{-Yx})}. \tag{2}$$

Here x is the input of sigmoid function and Y is the parameter, which determines the shape of sigmoid function. The sigmoid function becomes linear function when Y becomes zero.

The block diagram of NN controller is shown in Fig. 1. A desired robot trajectory is defined in terms of joints variables $(q_d, \dot{q}_d, \ddot{q}_d)$.

2.2. Controller structure and error dynamics

The control problem is the computation of necessary torques to apply to joint actuators such that manipulator follows the demanded trajectory. This requires computation of inverse dynamics equation, which for the n-link robot arm, is given in vectorial form as

$$\tau(t) = M(q)\ddot{q} + C(q, \dot{q}) + F_r(\dot{q}) + G(q) + d(t), \tag{3}$$

where $q(t) \in R^n$ the joint variable vector, $M(q) \in R^{n\times n}$ the inertia matrix, $C(q, \dot{q}) \in R^{n\times n}$ the matrix of centrifugal and Coriolis torques, $G(q) \in R^n$ the vector resulting from gravitational torques, and $F_r(\dot{q}) \in R^n$ is a positive-definite diagonal matrix of frictional terms (coulomb friction), $d(t)$ is the vector containing the unstructured un modeled dynamics

https://doi.org/10.1017/S0263574710000081 Published online by Cambridge University Press

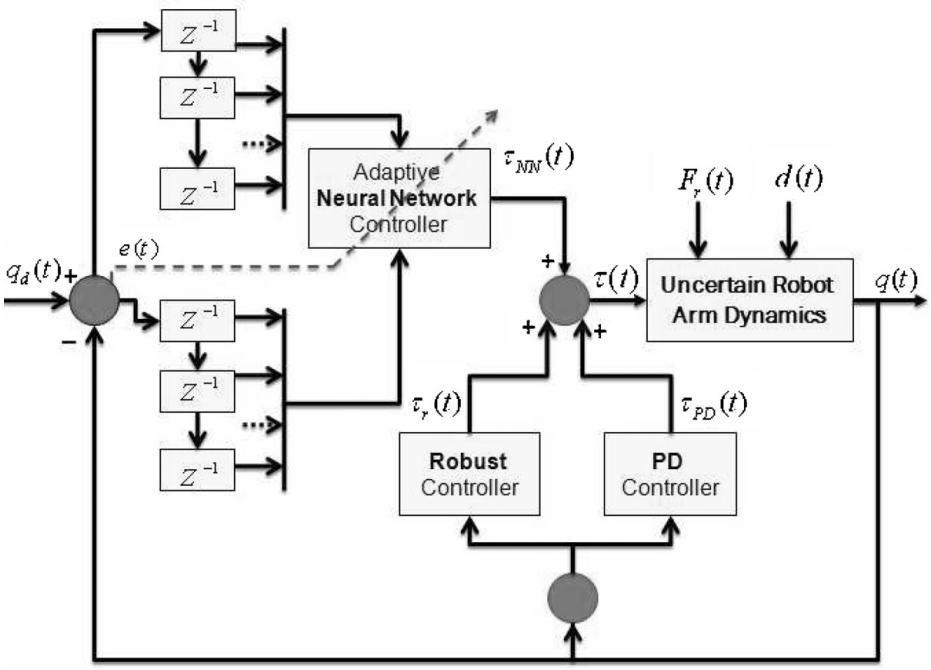


Fig. 1. Proposed robust NN controller.

and unknown external disturbances and the control input torque is $\tau(t) \in R^n$

This robot model (3) has the following structural properties that will be used on nonlinearity compensation:

Property 1: The inertia matrix is symmetric and positive definite, that is,

$$m_1 \|q\|^2 \leq q^T M(q) q \leq m_2 \|q\|^2; \quad \forall q \in R^n, \tag{4}$$

where m_1, m_2 are known positive constants, and $\|o\|$ denotes the Euclidean vector norm.

Property 2: The Coriolis and centrifugal matrix $C(q, \dot{q})$ is skew-symmetric, it satisfies the following relationship:

$$\|C(q, \dot{q})\| \leq k_c \|\dot{q}\|, \tag{5}$$

where

$$k_c = \frac{1}{2} \max \sum_{k=1}^{n} \|C_k(q)\| \quad \text{and} \quad C(q, \dot{q}) = \sum_{k=1}^{n} \|C_k(q)\|.$$

Property 3: The matrix $\dot{M}(q) - 2C(q, \dot{q})$ is skew-symmetric matrix so the matrix is satisfied with $q^T[\dot{M}(q) - 2C(q, \dot{q})]q = 0$ for $\forall q \in R^n$.

Property 4: The norm of unknown disturbance $d(t)$ has a positive upper bound b_{dis}, $\|d(t)\| \leq b_{dis}$.

Robot dynamic Eq. (3) can be written in state space form:

$$\begin{cases} \dot{x}_1 = x_2, \\ \dot{x}_2 = F(x, u), \\ y = x_1, \end{cases} \tag{6}$$

where $x_1 = q = [q_1, q_2 \ldots q_n]$ is joint position of n-link, $x_2 = \dot{q}$ is joint velocity of the link, $u = \tau$ is control input.

Let us define the tracking error as

$$\begin{cases} \tilde{x}_1 = x_1^d - x_1, \\ \tilde{x}_2 = x_2^d - x_2, \end{cases} \tag{7}$$

and filtered tracking error is

$$s = \tilde{x}_2 + \Lambda \tilde{x}_1, \tag{8}$$

where $\Lambda = \Lambda^T > 0$ is design parameter matrix usually selected diagonal. Then, the arm dynamics can be written in terms of the filtered tracking as

$$\begin{cases} M(x_1)\dot{x}_2 = \tau(t) - C(x_1, x_2) - F_r(x_2) - G(x_1) - d(t), \\ \dot{x}_2 = \dot{x}_2^d - \dot{s} + \Lambda \tilde{x}_2. \end{cases} \tag{9}$$

So,

$$M(x_1) \times \dot{s} = M(x_1)\left[\dot{x}_2^d + \Lambda \tilde{x}_2\right] + C(x_1, x_2)\left[x_2^d + \Lambda \tilde{x}_2\right] + F_r(x_2) + G(x_1) - C(x_1, x_2)s + d(t) - \tau(t), \tag{10}$$

or

$$M(x_1)\dot{s} = f - \tau(t) - C(x_1, x_2)s + d(t), \tag{11}$$

where the nonlinear robot function is

$$f(x) = M(x_1)\left[\dot{x}_2^d + \Lambda \tilde{x}_2\right] + C(x_1, x_2)\left[x_2^d + \Lambda \tilde{x}_2\right] + F_r(x_2) + G(x_1). \tag{12}$$

Multilayer NN compensation does not require structure information of uncertainties. The friction and gravity in (12) can be written in NN form, for convenience, we approximated

https://doi.org/10.1017/S0263574710000081 Published online by Cambridge University Press

the whole f instead of friction and gravity, so

$$M(x_1)\left[\dot{x}_2^d+\Lambda\tilde{x}_2\right]+C(x_1,x_2)\left[x_2^d+\Lambda\tilde{x}_2\right]+F_r(x_2)+G(x_1) = W^*\sigma(V^*x)+\varepsilon. \quad (13)$$

V^*, W^* are estimated values of ideal or optimal weights and ε is the approximated error, whose magnitude also depends on the values of V^*, W^*. ε is assumed to be quadratic bounded as

$$\varepsilon^T\Lambda_\varepsilon\varepsilon \le \bar{\varepsilon}, \quad (14)$$

where $\bar{\varepsilon}$ is positive constant. ε has very small values and their norm value is bounded by a known constant value. It is clear that, sigmoid functions, commonly used in NN satisfy Lipschitz condition, we define the weight deviations or weight estimation errors as

$$\begin{aligned}\tilde{V} &= V^* - \hat{V}\\ \tilde{W} &= W^* - \hat{W}\end{aligned} \quad (15)$$

where $\hat{V}$, $\hat{W}$ are time-varying weights of NNs. The Taylor series expansion for a given x can be written as

$$\sigma^*(V^Tx) = \sigma(\hat{V}^Tx)+\sigma'(\hat{V}^Tx)\tilde{V}^Tx+HO(\tilde{V}^Tx)^2. \quad (16)$$

Denoting $\hat{\sigma}' = \sigma'(\hat{V}^Tx)$, and the hidden layer output error for given x as

$$\begin{aligned}\tilde{\sigma} &= \sigma^* - \hat{\sigma}\\ &= \sigma(\hat{V}^Tx)+\sigma'(\hat{V}^Tx)\tilde{V}^Tx+HO(\tilde{V}^Tx)^2-\sigma(\hat{V}^Tx)\\ &= \sigma'(\hat{V}^Tx).\tilde{V}^Tx+HO(\tilde{V}^Tx)^2 = \hat{\sigma}'.\tilde{V}^Tx+\eta, \quad (17)\\ \eta &= HO(\tilde{V}^Tx)^2, \quad (18)\\ \|\eta\|^2 &= \eta^T\Lambda_\eta\eta \le \bar{\eta}, \quad (19)\end{aligned}$$

where $\bar{\eta}$ is a positive constant. In other word, for sigmoid activation function the higher order terms in Taylor series[14,15] are bounded by

$$\|\eta\| \le d_1 + d_2X_d\left\|\tilde{V}\right\| + d_3\left\|\tilde{V}\right\|\|r\|, \quad (20)$$

where d_i are computable positive constants and $X_d \in R$ is a known constant. In fact σ and derivatives are bounded by constants. In this paper σ' is $\sigma'(x) = 4\frac{e^{-YX_{input}}}{(1+e^{-YX_{input}})^2}$, and $X_{input} = [e^T\ \dot{e}^T\ q_d^T\ \dot{q}_d^T\ \ddot{q}_d^T]^T$.

Here we assume, the desired trajectory is bounded in the sense, that

$$\left\|[x_d^T\ ;\ \dot{x}_d^T\ ;\ \ddot{x}_d^T]\right\| \le X_d. \quad (21)$$

So, for each time t, X_{input} is bounded by

$$\left\|X_{input}\right\| \le C_1X_d + C_2\|s\|. \quad (22)$$

For computable positive constants C_i (C_2 decreases as Λ increases).

Control input torque is defined due to neural controller as

$$\tau_{NN} = \hat{f}(x), \quad (23)$$

where $\hat{f}(x)$ is an estimate of $f(x)$, then the NN function is estimated by:

$$\hat{f} = \hat{W}^T\sigma(\hat{V}^TX_{input}), \quad (24)$$

where $\hat{V}$, $\hat{W}$ are estimated values of ideal NN weights V^*, W^*.

Define control input torque by

$$\begin{aligned}\tau &= \tau_{NN}+\tau_{PD}+\tau_{Robust}\\ &= \hat{f}+K_D(\tilde{x}_2+\Lambda\tilde{x}_1)-K_{R1}(\tilde{x}_1)-K_{R2}.sign(\tilde{x}_2), \quad (25)\end{aligned}$$

where τ_{NN} is the output of the NN and τ_{PD} is the output of feedback (PD) controller (We assumed: $K_D\Lambda = K_P$) and τ_{Robust} is the output of robust controller which is employed to overcome large disturbances and that provides robustness in the face of higher-order terms in the Taylor series. K_{R1}, K_{R2} are proportional gain matrix and "sign" is the sign function. Note that the feedback controller plays a role in making the whole system stable. The closed loop system becomes

$$\begin{aligned}M(x_1)\dot{s} = f &-\left[\hat{f}+K_D(\tilde{x}_2+\Lambda\tilde{x}_1)+\tau_{Robust}\right]\\ &-C(x_1,x_2)s+d(t), \quad (26)\end{aligned}$$

or

$$\begin{aligned}M(x_1)\dot{s} = W^{*T}\sigma(V^{*T}X) &-[\hat{W}^T\sigma(\hat{V}^TX)+K_D(\tilde{x}_2+\Lambda\tilde{x}_1)\\ &+\tau_{Robust}]-C(x_1,x_2)s+d(t)+\varepsilon, \quad (27)\end{aligned}$$

Adding and subtracting $W^{*T}\hat{\sigma}$ yields

$$\begin{aligned}M(x_1)\dot{s} = W^{*T}\sigma(V^{*T}X)-\hat{W}^T\sigma(\hat{V}^TX)+W^{*T}\hat{\sigma}-W^{*T}\hat{\sigma}\\ -K_D(\tilde{x}_2+\Lambda\tilde{x}_1)-\tau_{Robust}-C(x_1,x_2)s+d(t), \quad (28)\end{aligned}$$

where $\hat{\sigma}$ and $\tilde{\sigma}$ are defined in (17). Adding and subtracting $\hat{W}^T\tilde{\sigma}$ yields

$$\begin{aligned}M(x_1)\dot{s} = W^{*T}\tilde{\sigma}-\tilde{W}^T\hat{\sigma}+\hat{W}^T\tilde{\sigma}-\hat{W}^T\tilde{\sigma}\\ -K_D(\tilde{x}_2+\Lambda\tilde{x}_1)-\tau_{Robust}-C(x_1,x_2)s+d(t), \quad (29)\end{aligned}$$

the key step is the use of the Taylor series approximation (16) for $\tilde{\sigma}$, the closed-loop error system is

$$\begin{aligned}M(x_1)\dot{s} = \hat{W}^T\hat{\sigma}'.\tilde{V}^Tx-\tilde{W}^T\hat{\sigma}\\ -K_D(\tilde{x}_2+\Lambda\tilde{x}_1)-\tau_{Robust}-C(x_1,x_2)s+w(t), \quad (30)\end{aligned}$$

where the disturbance terms are

$$w(t) = \tilde{W}^T\hat{\sigma}'(\tilde{V}^Tx)+\tilde{W}^T\eta+d(t)+\varepsilon. \quad (31)$$

For notational convenience define the matrix of all target weights as

$$Z = \begin{bmatrix}W^* & 0\\ 0 & V^*\end{bmatrix}. \quad (32)$$

https://doi.org/10.1017/S0263574710000081 Published online by Cambridge University Press

The ideal weights are bounded by known positive values so that $\|V^*\| \leq V_{Max}$, $\|W^*\| \leq W_{Max}$ or $\|Z\| \leq Z_{Max}$ with Z_{max} known. So disturbance is bounded according to

$$\|w(t)\| \leq C_0 + C_1 \left\|\tilde{Z}\right\| + C_2 \left\|\tilde{Z}\right\| \|s\| \tag{33}$$

with C_i known positive constants.

2.3. Stability analysis

In this section the stability analysis is done based on Lyapunov direct method. The Lyapunov function is defined as below:

$$L = \frac{1}{2}s^T Ms + \frac{1}{2}tr\left(\tilde{W}^T K_w^{-1}\tilde{W}\right) + \frac{1}{2}tr\left(\tilde{V}^T K_v^{-1}\tilde{V}\right) \tag{34}$$

and derivative of (34) is

$$\dot{L} = \left(s^T M\dot{s} + \frac{1}{2}s^T \dot{M}s\right) + tr\left(\tilde{W}^T K_w^{-1}\dot{\tilde{W}}\right) + tr\left(\tilde{V}^T K_v^{-1}\dot{\tilde{V}}\right). \tag{35}$$

Using (30),

$$\begin{aligned}\dot{L} = &-s^T K_D s + s^T[w(t) - \tau_{Robust}] + \frac{1}{2}s^T[\dot{M} \\ &- 2C(x_1, x_2)]s + tr\left(\tilde{W}^T K_w^{-1}\dot{\tilde{W}}\right) + tr\left(\tilde{V}^T K_v^{-1}\dot{\tilde{V}}\right) \\ &+ s^T[\hat{W}^T\hat{\sigma}'.\tilde{V}^T x - \tilde{W}^T\hat{\sigma}].\end{aligned} \tag{36}$$

From property 2, $q^T[\dot{M}(q) - 2C(q,\dot{q})]q = 0$ we have

$$\begin{aligned}\dot{L} = &-s^T K_D s + s^T[w(t) - \tau_{Robust}] + tr\left(\tilde{W}^T K_w^{-1}\dot{\tilde{W}}\right) \\ &+ tr\left(\tilde{V}^T K_v^{-1}\dot{\tilde{V}}\right) + s^T[\hat{W}^T\hat{\sigma}'.\tilde{V}^T x - \tilde{W}^T\hat{\sigma}].\end{aligned} \tag{37}$$

It becomes

$$\begin{aligned}\dot{L} = &-s^T K_D s + s^T[w(t) - \tau_{Robust}] + tr\left[\left(K_w^{-1}\dot{\tilde{W}}\right.\right. \\ &\left.\left. -\hat{\sigma}s^T\right)\tilde{W}^T\right] + tr\left[\left(K_v^{-1}\dot{\tilde{V}} + x\hat{W}^T\hat{\sigma}'\right)\tilde{V}^T\right].\end{aligned} \tag{38}$$

Since it is desired to have $\dot{L}$ at least negative semi definite, let us have the following updating laws for the weights of NN:

$$\begin{cases}\dot{\tilde{W}} = K_w\hat{\sigma}(Vx)s^T, \\ \dot{\tilde{V}} = -K_v x\hat{W}^T\hat{\sigma}'(Vx)s^T.\end{cases} \tag{39}$$

Substituting (39) into (38), we get

$$\dot{L} = -K_D\|s\|^2 + \|w(t) - \tau_{Robust}\|\|s\|. \tag{40}$$

If the parameter "s" is selected to satisfy the following inequality:

$$-K_D\|s\|^2 + \|w(t) - \tau_{Robust}\|\|s\| \leq 0, \tag{41}$$

$$\|s\| \geq \frac{\|w(t) - \tau_{Robust}\|}{K_D}, \tag{42}$$

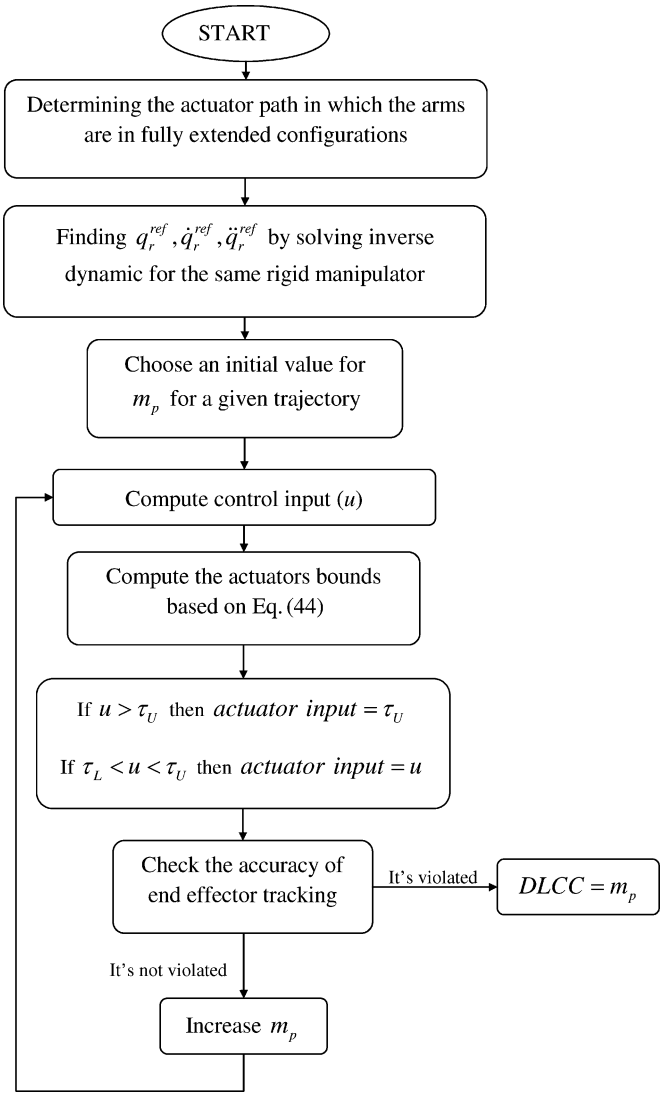


Fig. 2. Algorithm for determining MLCC with ARNN.

then we have

$$\dot{L} \leq 0 \tag{43}$$

From (43) and (34), the control system is stable based on the Lyapunov stability.

3. Maximum Load-Carrying Capacity for a Desired Trajectory

The actuator torque constraint is formulated on the basis of typical torque-speed characteristics of DC motors. The upper and lower bounds of the allowable torque are as follows:

$$\tau_U = k_1 - k_2\dot{q}, \quad \tau_L = -k_1 - k_2\dot{q}, \tag{44}$$

where $k_1 = \tau_s$, $k_2 = \frac{\tau_s}{\omega_0}$, τ_s is the stall torque, and ω_0 is maximum speed of the motor without any load. The algorithm used for finding MLCC in closed-loop case is shown in Fig. 2. In closed-loop case the actuator constraint is the major parameter in determining the MLCC while in open loop case the end-effector accuracy is the major parameter which determines the MLCC. As long as the actual mass

https://doi.org/10.1017/S0263574710000081 Published online by Cambridge University Press

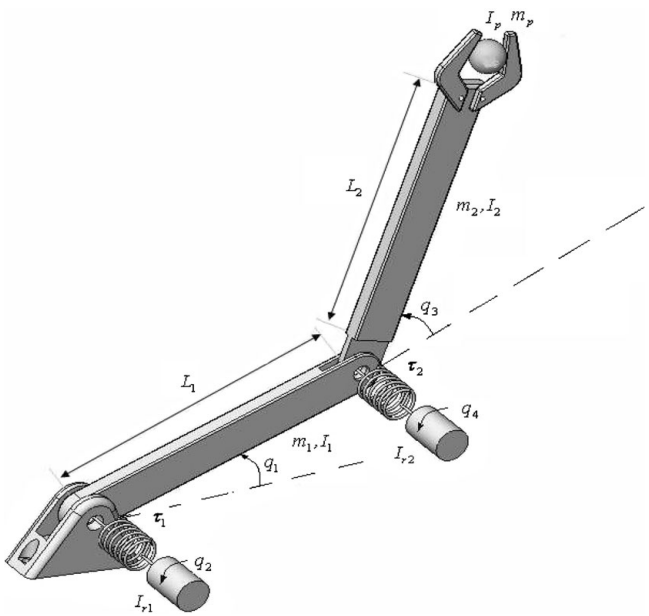


Fig. 3. Two-Link Flexible Joint Robot Manipulator.

is not greater than the maximum load for that trajectory, the actuators are adequate to execute trajectory within the allowable end-effector deflection.

4. Simulation Study and Discussion

In order to verify and demonstrate the effectiveness of the proposed adaptive robust NN controller, the simulation examples of two-link flexible joints robot and industrial robot are studied. In two-link flexible joints robot the payload is considered as a point mass m_p which is located at the end of the link 2 with mass m_2 and length l_2 (Fig. 3). Driving torques τ_1 and τ_2 are the inputs to the robot and joint positions are represented by q_1 and q_2. Robot motion is simulated by solving dynamic equation of the robot (Figs. 7–8).

A two-layered NN has been used. The network consisted of 5 inputs, 20 nodes in the first hidden layer, and 2 nodes in the output layer. The link parameters and inertia properties of the manipulator are given in Table I.

We compared the result of simulation with sliding mode controller in ref. [5] and feedback linearization method in ref. [3] so that indicate using this controller better result for MLCC can be obtained than those methods. An initial simulation condition for robot motion is

$$q_1(0) = 0.5\,\text{rad}, \quad \dot{q}_1(0) = -0.5\,\text{rad/s},$$
$$q_2(0) = 0.4\,\text{rad}, \quad \dot{q}_2(0) = -2\,\text{rad/s}.$$

The gain values of PD controller were adopted as

$$K_p = diag[200, 200],\ K_D = diag[10, 10].$$

For robust controller, K_{R1} is proportional gain matrix and empirically set to $K_{R1} = diag[10, 8]$ and $K_{R2} = [K_{R21}, K_{R22}]^T$, K_{R21}, K_{R22} are constants also empirically set to 0.4 and 0.5. The results of simulation are included angular position and velocity tracking of first and second link that are shown in Figs. 4 and 5.

The resulting path tracking in the flexible case is shown in Fig. 6 and compared with the desired path. The maximum allowable error bound at each point of the desired path is 2 cm that tracking error is complied this accuracy.

In this case the joint flexibility constraint is more dominant with respect to the joint actuator constraints and as a result the load-carrying capacity in the flexible case is less than that in the rigid case for a desired trajectory. The maximum allowable load in this desired path is 2.72 kg. It can be seen that the best results were obtained with the adaptive robust neural controller than feedback linearization[3] and sliding mode[5] controller.

For comparison the maximum load obtained in this work, the method proposed in ref. [3] is applied for given trajectory introduced herein. As a result, maximum load using this controller 2.21 kg is computed. In ref. [5] sliding mode control for determining maximum load of elastic joint robots is used and maximum load using this controller 2.53 kg obtained for the same trajectory. Robot and actuator parameters are listed in Table I.

For another case, we applied this controller for industrial robot by considering the effect of gravity, so it shows the ability of this controller for compensation of gravity and friction. In this case the payload is considered as a point mass m_p which is located at the end of the link 3 with mass m_3 and length l_3. Driving torques τ_1, τ_2 and τ_3 are the inputs

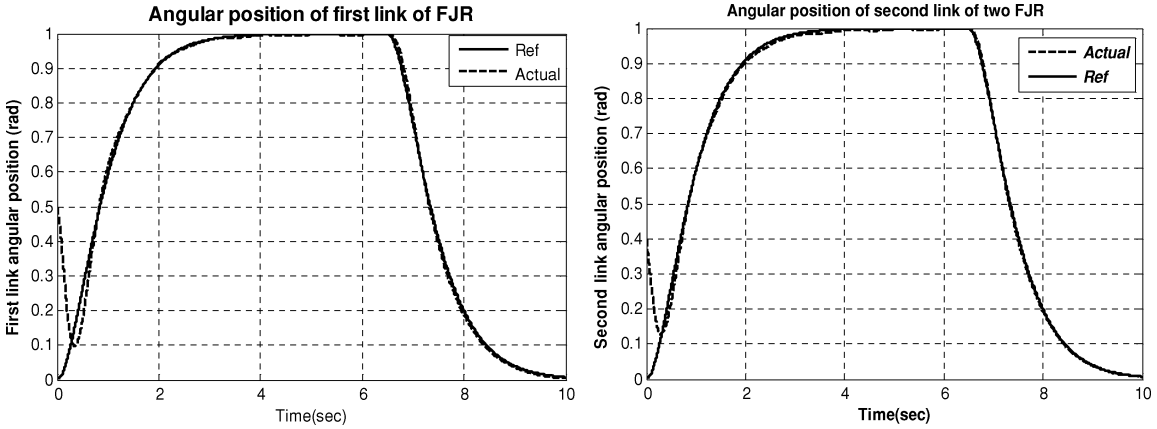


Fig. 4. Angular position of the first link and second link.

https://doi.org/10.1017/S0263574710000081 Published online by Cambridge University Press

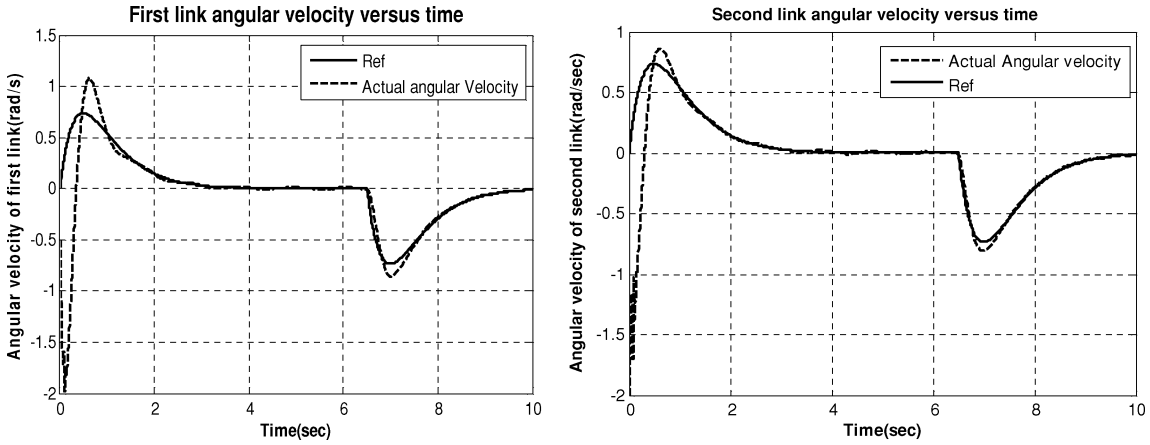


Fig. 5. Angular velocity of the first link and second link.

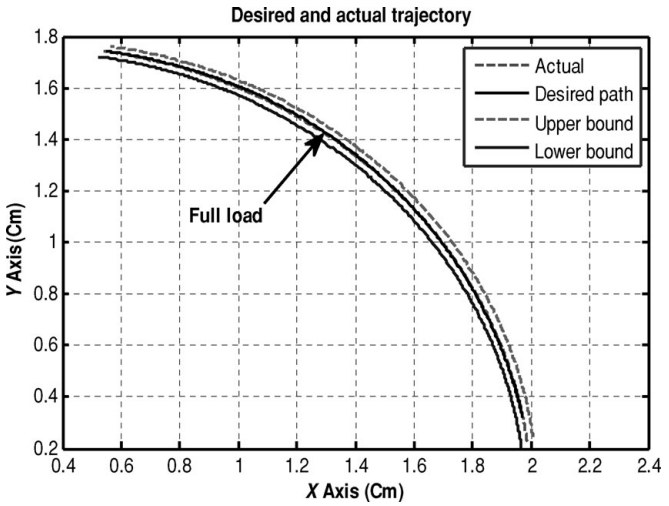


Fig. 6. Comparison of flexible joint robot path with respect to desired path.

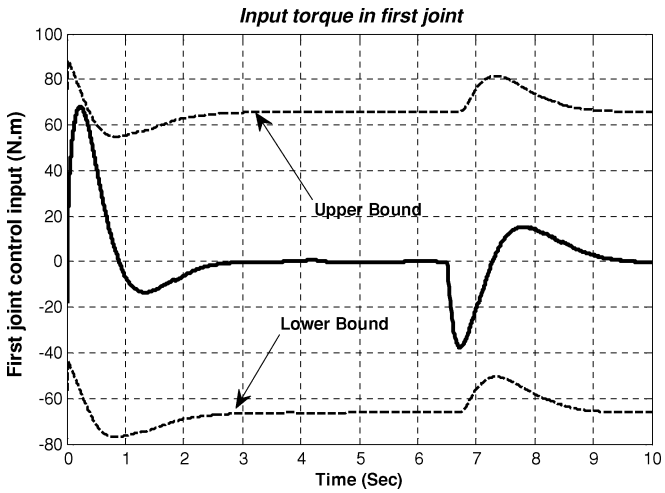


Fig. 7. FJR case, actuator torque at first joint.

to the robot and joint positions are represented by q_1, q_2 and q_3.

As it can be seen in Fig. 9 the tip position will be

$$\begin{aligned} x &= \cos(q_1)\left[l_2\sin(q_2) + l_2\sin(q_1+q_3)\right], \\ y &= \sin(q_1)\left[l_2\cos(q_2) + l_2\cos(q_1+q_3)\right], \\ z &= l_1 + l_2\sin(q_2) + l_2\sin(q_2+q_3). \end{aligned} \tag{45}$$

Inertia matrix $M(q)$ in Eq. (3) is symmetric. After deriving the dynamic equations of manipulator, the inertia matrix elements can be written as below:

$$\begin{cases} \mathrm{M}(1,1) = M_1 + X_1 C_{2q_2} + X_2 C_{q_3} + X_2 C_{2q_2+q_3} \\ \qquad\qquad + X_3 C_{2q_2+2q_3}, \\ \mathrm{M}(2,1) = 0, \\ \mathrm{M}(3,1) = 0, \\ \mathrm{M}(2,2) = M_2 + 2X_2 C_{q_3}, \\ \mathrm{M}(3,2) = M_3 + X_2 C_{q_3}, \\ \{\mathrm{M}(3,3) = M_4, \end{cases} \tag{46}$$

Table I. Link parameters and inertia properties of articulated robot.

Parameter	Value	Unit
Link length	$l_1 = l_2 = l_3 = 1$, $l_{c1} = l_{c2} = l_{c3} = 0.5$	m
Moment of inertia around y-axis	$I_{y1} = I_{y2} = I_{y3} = 1$	kg.m^2
Moment of inertia around z-axis	$I_{Z1} = I_{Z2} = I_{Z3} = 1$	kg.m^2
Link mass	$m_1 = m_2 = m_3 = 1$	kg
No load speed	$w_0 = 3.5$	rad/s
Stall torque	$T_s = 0.63$	N-m
Link linear mass density	$\mu_1 = \mu_2 = \mu_3 = 0.405$	kg/m
viscous friction coefficients	$v_1 = v_2 = v_3 = 11$	Nm
joint Coulomb friction	$c_1 = c_2 = c_3 = 0.8$	Nm s/rad

https://doi.org/10.1017/S0263574710000081 Published online by Cambridge University Press

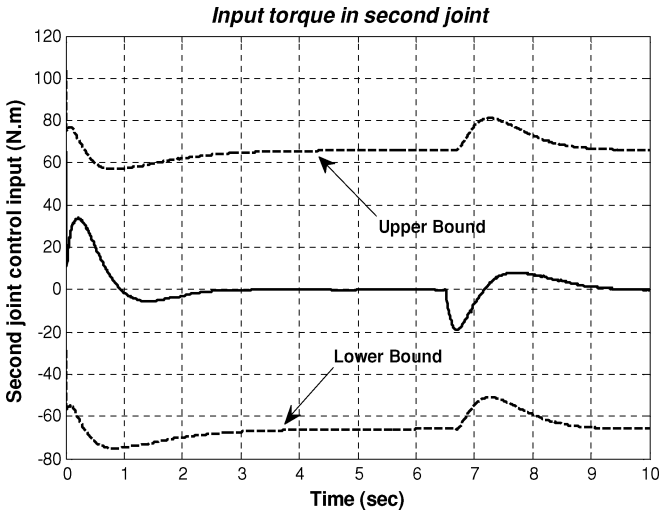


Fig. 8. FJR case, actuator torque at second joint.

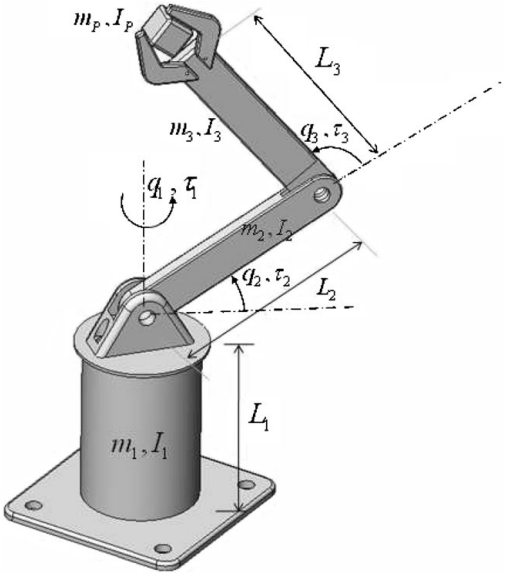


Fig. 9. Articulated robot (3R).

where $C_{aq_i+bq_j} = \cos(aq_i + bq_j)$. The parameters M_i and X_i are given below:

$$\begin{aligned} M_1 &= I_{y1} + 0.5I_{y2} + 0.5I_{y3} + 0.5m_2l_{c2}^2 + 0.5m_3l_{c3}^2 \\ &\quad + 0.5m_3L_2^2 + 0.5M_PL_2^2 + 0.5M_PL_3^2, \\ M_2 &= I_{z2} + I_{z3} + m_2l_{c2}^2 + m_3l_{c3}^2 + m_3L_2^2 + M_PL_2^2 \\ &\quad + M_PL_3^2, \\ M_3 &= I_{z3} + m_3l_{c3}^2 + M_PL_3^2, \\ M_4 &= I_{z3} + m_3l_{c3}^2 + M_PL_3^2, \end{aligned} \tag{47}$$

$$\begin{aligned} X_1 &= 0.5I_{y2} + 0.5m_2l_{c2}^2 + 0.5m_3L_2^2 + 0.5M_PL_2^2, \\ X_2 &= m_3l_{c3}L_2 + M_PL_2L_3, \\ X_3 &= 0.5I_{y3} + 0.5m_3l_{c3}^2 + 0.5M_PL_3^2. \end{aligned} \tag{48}$$

The link parameters and inertia properties of the manipulator are given in Table I. For comparison with open loop controller the initial simulation condition for articulated robot motion is zero. The gain values of PD controller were adopted as

$$K_p = diag[150, 200, 120],\ K_D = diag[20, 10, 15].$$

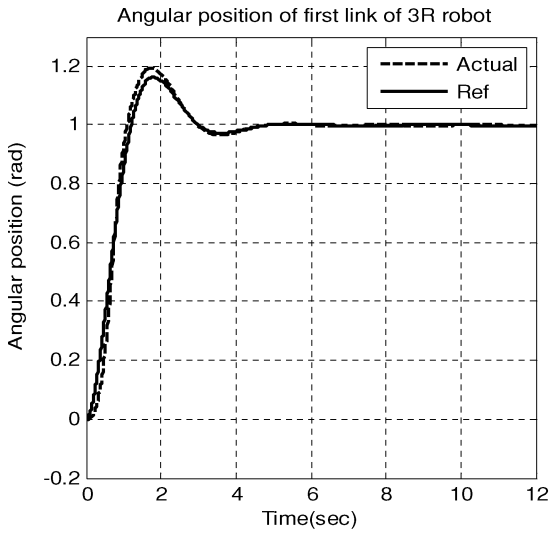
For robust controller, K_{R1} is proportional gain matrix and empirically set to $K_{R1} = diag[10, 8, 10]$ and $K_{R2} = [K_{R21}, K_{R22}, K_{R23}]^T$, $K_{R21}, K_{R22}, K_{R23}$ are constants also empirically set to 0.4, 0.5, and 0.5. The results of simulation are included angular position, error, and velocity tracking of first, second, and third link, that are shown in Figs. 10–13.

The result of path tracking in 3R robot case is shown in Fig. 14 and compared with the desired path. The maximum allowable error bound at each point of the desired path is 3 cm, in the beginning of desired path tracking the error is too high, but after time lapse NN compensate the effect of gravity and friction.

In this case the maximum allowable load 3.3 kg is obtained (Figs. 15–17).

5. Conclusions

In practice, it is very difficult for the parameters associated with the robot model to be determined exactly. Under such



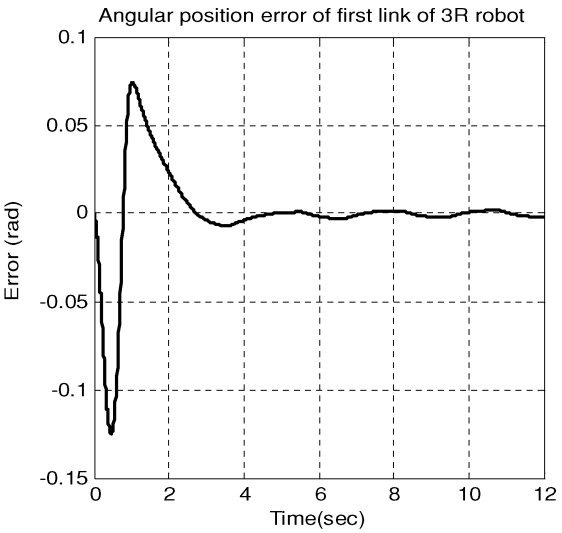


Fig. 10. 3R robot case, angular position of the first link.

https://doi.org/10.1017/S0263574710000081 Published online by Cambridge University Press

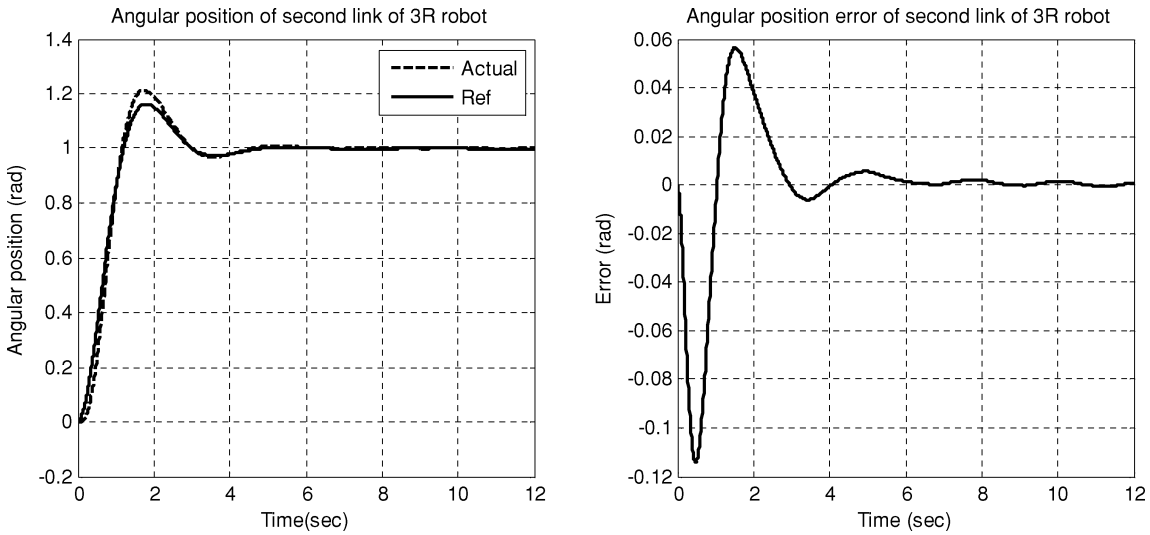


Fig. 11. 3R robot case, angular position of the second link.

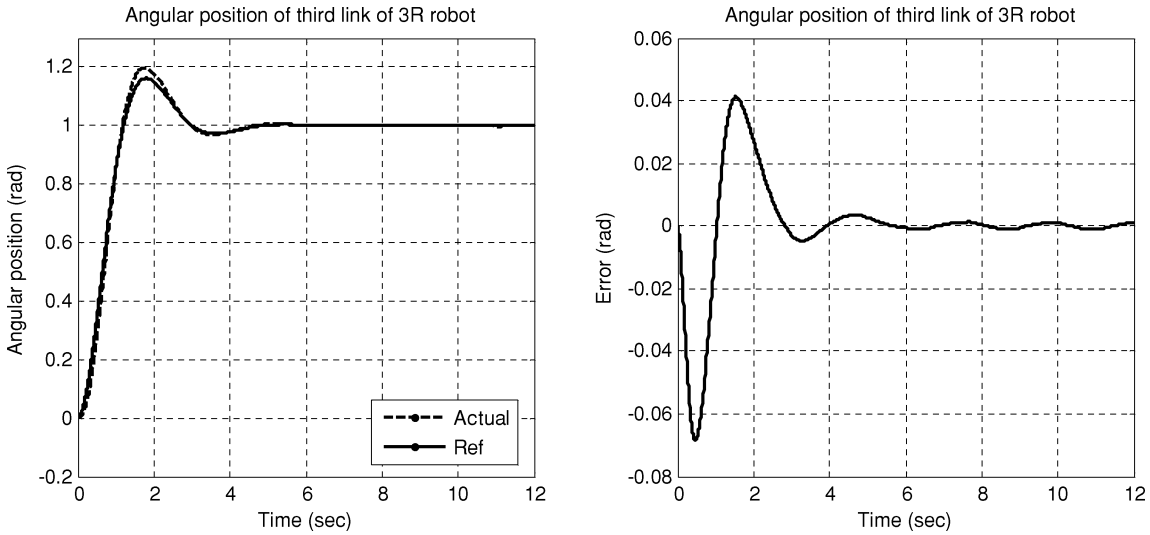


Fig. 12. 3R robot case, angular position of the third link.

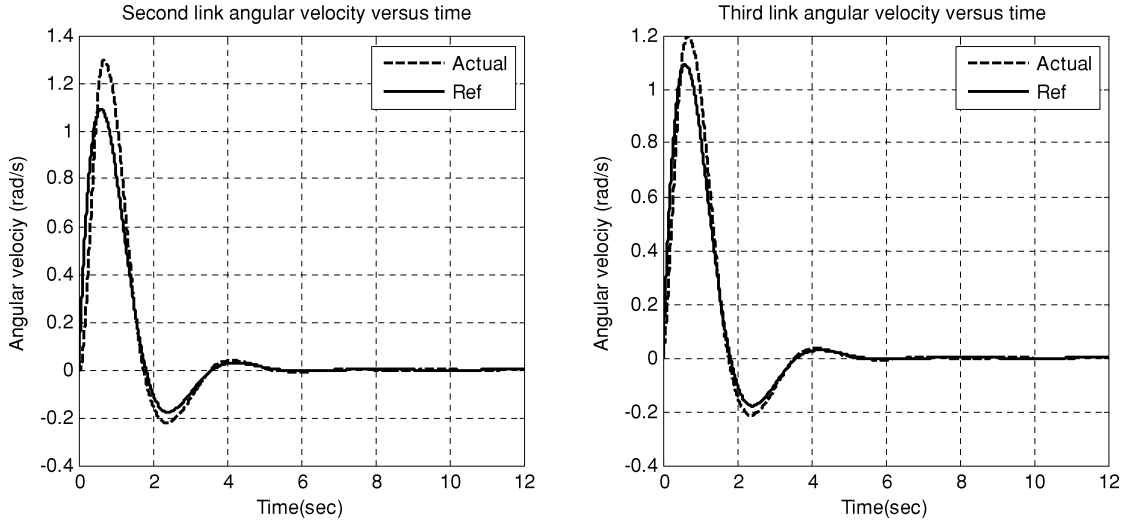


Fig. 13. 3R robot case, angular velocity of the second and third link.

https://doi.org/10.1017/S0263574710000081 Published online by Cambridge University Press

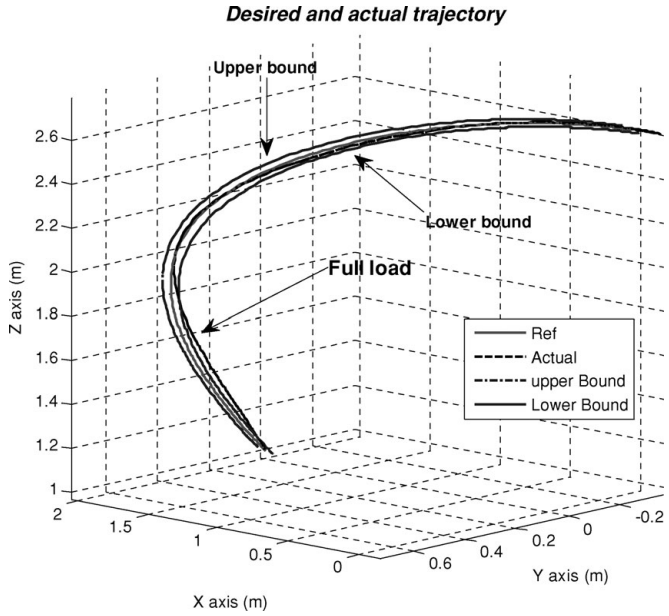


Fig. 14. Comparison of 3R robot path with respect to desired path.

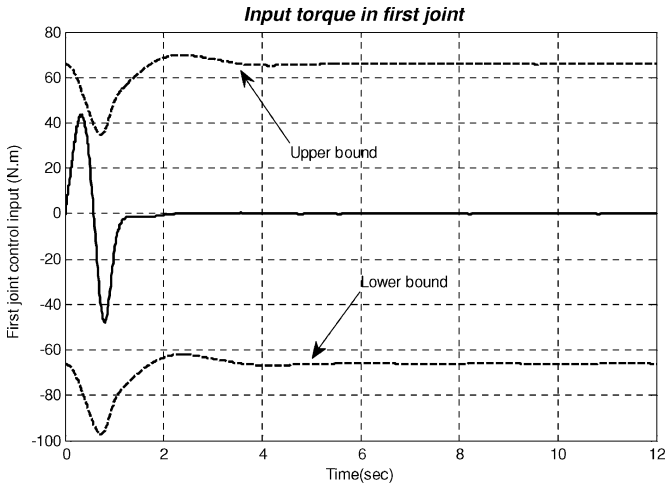


Fig. 15. 3R robot case, actuator torque at first joint.

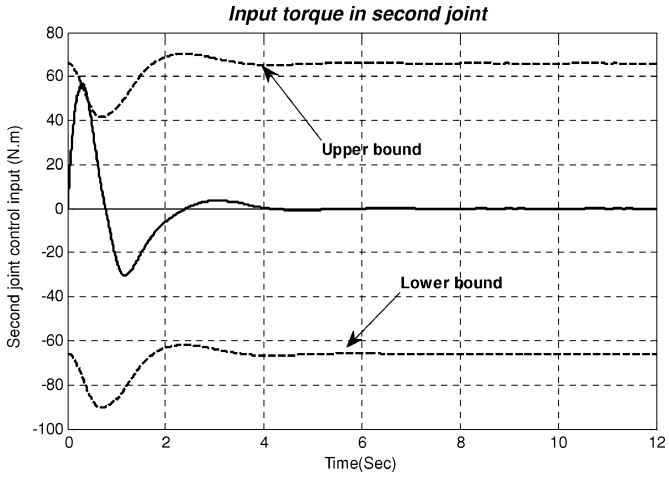


Fig. 16. 3R robot case, actuator torque at second joint.

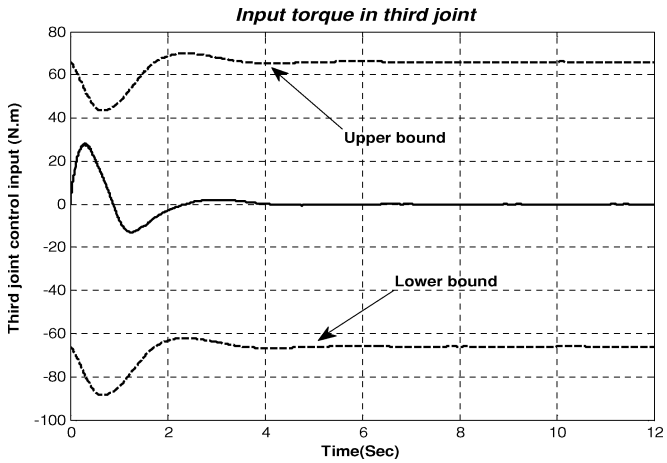


Fig. 17. 3R robot case, actuator torque at third joint.

circumstances, the trajectory tracking performance using the computed-torque control significantly degrades due to the inaccuracy of the dynamic model. For this reason, we developed a nonlinear NN controller for articulated robots. The ARNN controller offers guaranteed tracking behavior. A robust control term is also needed to overcome higher order modeling error terms. Considering both actuator torque and end-effector precision constraints, MLCC is obtained. Results show that this adaptive robust neural controller suppresses disturbances accurately and achieves very small errors between commanded and actual trajectories. The maximum allowable error bound at each point of the desired path for two-link flexible joints and 3R robot, respectively, were obtained 2 cm and 3 cm and corresponding maximum allowable load are 2.72 kg and 3.3 kg.

Appendix. Dynamic Equation of the Two-Link Flexible Robot

The dynamic equations of robot arm can be written as follows:

Kinematic equations

$$\begin{aligned}(-l_1\sin(q_1) - l_2\sin(q_1+q_3))\ddot{q}_1 - l_2\sin(q_1+q_3)\ddot{q}_2 = R_{t1},\\ (-l_1\cos(q_1) - l_2\cos(q_1+q_3))\ddot{q}_1 + l_2\cos(q_1+q_3)\ddot{q}_2 = R_{t2}.\end{aligned} \tag{A 1}$$

Dynamic equations based on Lagrangian approach

$$\tau(t) = D(q)\ddot{q} + C(q,\dot{q}) + F_r(\dot{q}) + G(q), \tag{A 2}$$

$$\begin{cases} D_{11}\ddot{q}_1 + D_{12}\ddot{q}_3 - C\dot{q}_3^2 - 2C\dot{q}_1\dot{q}_3 + k(q_1-q_2) + F_{r1} = 0,\\ D_{22}\ddot{q}_3 + D_{12}\ddot{q}_1 + C\dot{q}_1^2 + k(q_3-q_4) + F_{r2} = 0,\\ I_{r1}\ddot{q}_2 + k(q_2-q_1) = \boldsymbol{\tau}_1,\\ I_{r2}\ddot{q}_4 + k(q_4-q_3) = \boldsymbol{\tau}_2. \end{cases} \tag{A 3}$$

https://doi.org/10.1017/S0263574710000081 Published online by Cambridge University Press

For this manipulator the terms appearing in the dynamical equation are given as

$$\begin{aligned}
D_{11} &= m_1 L_{C1}^2 + I_1 + m_2[L_1^2 + L_{c2}^2 + 2L_1 L_{C2}\cos q_3] + I_2 \\
&\quad + m_p[L_1^2 + L_2^2 + 2L_1 L_{c2}^2 + 2L_1 L_2 \cos q_3] + I_p, \\
D_{12} &= m_2 L_1 L_{C2}\cos(q_3) + m_2 L_{c2}^2 + I_2 + m_p L_1 L_2 \cos(q_3) \\
&\quad + m_p L_2^2 + I_p, \\
D_{22} &= m_2 L_{C2}^2 + I_2 + m_p L, \\
C &= m_2 L_1 L_{C2}\sin q_3 + m_p L_1 L_2 \sin q_3, \\
F_{r1} &= v_1 \dot{q}_1 + c_1 \mathrm{sgn}(\dot{q}_1), \\
F_{r1} &= v_v \dot{q}_2 + c_2 \mathrm{sgn}(\dot{q}_2),
\end{aligned} \tag{A 4}$$

where v_1, v_2 are the viscous friction coefficients of the arm joints, c_1, c_2 are the joint Coulomb friction torques, q_1 and q_3 are link angles, and q_2 and q_4 are rotor angles.

References

1. M. Thomas, H. Yuan-Chou and D. Tesar, "Optimal actuator sizing for robotic manipulators based on local dynamic criteria," *ASME J. Mech. Trans. Autom. Des*. **107**, 163–169 (1985).
2. M. H. Korayem and A. Basu, "Dynamic load carrying capacity of robotic manipulators with joint elasticity imposing accuracy constraints," *J. Robot. Auton. Syst.* **13**, 219–229 (1994).
3. M. H. Korayem, F. Davarpanah and H. Ghariblu, "Load carrying capacity of flexible joint manipulators with feedback linearization," *Int. J. Adv. Manuf. Technol.* **21**, 389–397 (2006).
4. M. H. Korayem, H. Ghariblu and A. Basu, "Dynamic load-carrying capacity of mobile-base flexible joint manipulators," *J. Adv. Manuf. Technol.* **25**, 62–70 (2005).
5. M. H. Korayem and A. Pilechian, "Maximum allowable load of elastic joint robots: Sliding mode control approach," *Amirkabir J. Sci. Technol.* **17**(65), 75–82 (2007).
6. F. Lewis, "Neural network feedback control: Work at UTA's Automation and Robotics Research Institute," *J. Intell. Robot. Syst.* **48**, 513–523 (2007).
7. P. Gupta and N. K. Sinha, "Intelligent control of robotic manipulators: Experimental study using neural networks," *J. Mechatronics* **10**, 289–305 (2000).
8. J. Nakanishi and S. Schaal, "Feedback error learning and nonlinear adaptive control," *Neural Netw.* **17**, 1453–1465 (2004).
9. S. Yildirim, "Adaptive robust neural controller for robots," *J. Robot. Auton. Syst.* **46**, 175–184 (2004).
10. F. Temurtasa, H. Temurtas and N. Yumusak, "Application of neural generalized predictive control to robotic manipulators with a cubic trajectory and random disturbances," *Robot. Auton. Syst.* **54**, 74–83 (2006)
11. M. M. Abdelhameed, "Adaptive neural network based controller for robots," *J. Mechatronics* **9**, 147–162 (1999).
12. F. Sun, Z. Sun, N. Li and L. Zhang, "Stable adaptive control for robot trajectory tracking using dynamic neural networks," *J. Mach. Intell. Robot. Control* **2**, 71–78 (1999).
13. E. Akbas and E. Esin, "A simulational comparison of intelligent control algorithms on a direct drive manipulator," *J. Robot. Auton. Syst.* **49**, 235–244 (2004).
14. W. Yu and X. Li, "PD control of robot with velocity estimation and uncertainties compensation," *Int. J. Robot. Autom.* **21** (2006).
15. F. Lewis and T. Parisini, "Neural network feedback control with guaranteed stability," *Int. J. Control* **70**, 337–339 (1998).
16. W. Yu, A. S. Poznyak and E. N. Sanchez, "Neural adaptive control of two-link manipulator with sliding mode," *IEEE Int. Conf. Robot. Autom.* **4**, 3122–3127 (1999).
17. H. Chun, C. Guan-Ming and T. Lee, "Robust intelligent tracking control with PID-type learning algorithm," *Neurocomputing* **71**, 234–243, (2007)
18. R. Xiaogang, D. Mingxiao, G. Daoxiong and Q. Junfei, "On-line adaptive control for inverted pendulum balancing based on feedback-error-learning," *Neurocomputing* **70**, 770–776 (2007).
19. S. Purwar, I. Kar and A. Jha, "Adaptive output feedback tracking control of robot manipulators using position measurements only," *Expert Syst. Appl.* **34**, 2789–2798 (2008).

https://doi.org/10.1017/S0263574710000081 Published online by Cambridge University Press