DOI: 10.1007/s12541-011-0054-3

Neural Network Based Hybrid Force/Position Control for Robot Manipulators

Naveen Kumar[1,#], Vikas Panwar[2], Nagarajan Sukavanam[3], Shri Prakash Sharma[3] and Jin-Hwan Borm[1]
1 Division of Mechanical Engineering, Ajou University, San5, Woncheon-dong, Yeongtong-gu, Suwon, South Korea, 443-749
2 Department of Applied Mathematics, Defence Institute of Advanced Technology, Pune-411025, Maharashtra, India
3 Department of Mathematics, Indian Institute of Technology Roorkee, Roorkee-247667, Uttrakhand, India
# Corresponding Author / E-mail: navindma@gmail.com, TEL: +82-31-219-2939, FAX: +82-31-219-1611

KEYWORDS: Hybrid force/position control, Decomposition, Interaction force, Feedforward neural networks, Lyapunov stability

This paper presents a neural network based adaptive control scheme for hybrid force/position control for rigid robot manipulators. Firstly the robot dynamics is decomposed into force, position and redundant joint subspaces. Based on this decomposition, a neural network based controller is proposed that achieves the stability in the sense of Lyapunov for desired interaction force between the end-effector and the environment as well as regulate robot tip position in cartesian space. A feedforward neural network is employed to learn the parametric uncertainties, existing in the dynamical model of the robot manipulator. Finally numerical simulation studies are carried out for a two link rigid robot manipulator.

Manuscript received: February 18, 2010 / Accepted: January 18, 2011

1. Introduction

In various industrial applications of robotic manipulators such as contour following, grinding, deburring edges as well as assembly related tasks, the manipulator end-effector is required to make contact with the environment. During the execution of such tasks, the motion of end-effector is constrained by the environment. For the successful execution of such type of tasks robot position as well as contact forces should be controlled accurately. Many researchers have investigated this problem in recent years. As a result, two approaches to achieving compliant motion have emerged. The first approach, usually called hybrid position/force control, is based on the observation that when the robot end-effector is in contact with the environment, the cartesian-space of the end-effector coordinates may be naturally decomposed into a position subspace and a force subspace. The objective of hybrid position/force controller is to track a position (and orientation) trajectory in the position subspace and a force (and moment) trajectory in the force subspace. The second approach to compliant motion for robots is called impedance control, and proposes that the control objective should not be the tracking of position/force trajectories, but rather should involve the regulation of the mechanical impedance of the robot end-effector which relates position and force. Raibert and Craig[1] developed a hybrid scheme which decomposes a task space into two orthogonal subspaces: position and force. Lozano and Brogliato[2] proposed an adaptive force/position control scheme, based on a particular decomposition of the robot jacobian and environment stiffness matrices. By extending the method of,[1] Yoshikawa and Sudou[3] proposed dynamic hybrid position/force control scheme, which takes into consideration the manipulator dynamics and the constraints on the end-effector specified by the given task. Kwan[4] proposed a new robust adaptive control scheme for simultaneous force/motion control of constrained rigid robots including motor dynamics. Queiroz et al.[5] presented an adaptive position/force controller for robot manipulators during the constrained motion without velocity measurements. Xiao et al.[6] utilized the force/torque and vision sensors simultaneously to propose a hybrid position/force controller such that the end-effector moved along a path on an unknown surface with the aid of a single camera assumed to be uncalibrated with respect to the robot coordinates. Kouya et al.[7] presented a general study of an adaptive force/position control using the strict-feedback backstepping technique, based on passivity and applied to a robotic system. Roy and Whitcomb[8] presented an adaptive force control algorithm with low-level position/velocity controllers for robot arms in contact with surfaces of unknown linear compliance. Cheah et al.[9] proposed a motion and force tracking controller for robots with uncertain kinematics and dynamics. Kang et al.[10] proposed a method to reduce impulsive contact force between manipulator and its environment.

Recently, neural network (NN) due to their versatile features such as learning capability, nonlinear mapping and parallel processing have gained considerable popularity among control community. The most useful property of neural network in control is their ability to approximate arbitrary linear or nonlinear mapping through learning. Due to this property neural network have been proven to be suitable tool for controlling complex nonlinear dynamical systems. The basic idea behind neural network based control is to learn unknown nonlinear dynamics and compensate for structured/unstructured uncertainties existing in the dynamic model. Many neural network based control schemes are available for control of robot manipulators.[11-13] They derived successfully control algorithms with stable tracking while approximating the unknown dynamics of the manipulators with neural network that are universally considered to be fine approximators and trained them online removing any preliminary offline training. Shenghai et al.[14] presented a new neural network controller for the constrained robot manipulators in task space. The controller consisted of a model-based term and a neural network on-line adaptive compensation term. Karayiannidis et al.[15] considered the problem of force/position tracking for a robotic manipulator in compliant contact with a surface under non-parametric uncertainties. Panwar and Sukavanam[16] designed an optimal hybrid motion and force control scheme for a constrained robotic manipulator with unknown dynamics. Zhao and Cheah[17] proposed a vision-based neural network controller for robots with uncertain kinematics, dynamics and constraint surface.

In this paper, a neural network (NN) based adaptive control scheme for hybrid force/position control of rigid robot manipulators is presented. A feedforward neural network is employed to learn the parametric uncertainties, existing in the dynamical model of the robot manipulator. The neural network controller achieves the stability in the sense of Lyapunov for desired interaction force between the end-effector and the environment as well as regulate robot tip position in cartesian space.

The paper is organized as follows. In Section 2, robot dynamical model and its decomposition is presented. A review of feedforward neural networks is presented in Section 3. In Section 4, the neural networks based controller design is presented. Numerical simulation results are included in Section 5 followed by the conclusion in Section 6.

2. Dynamics of Rigid Robots

2.1 Robot Dynamical Model and its Properties

Based on Euler-Lagrange equations, the revolute rigid robot dynamics with environment contact can be expressed as

$$M(q)\ddot{q} + V_m(q,\dot{q}) + G(q) = \tau - \tau_e \tag{1}$$

where $M(q) \in R^{n\times n}$ represents the inertia matrix, $V_m(q,\dot{q}) \in R^{n\times n}$ represents the centripetal-coriolis matrix, $G(q) \in R^n$ represents the gravity effects, $\tau \in R^n$ represents the torque input vector and $\tau_e \in R^n$ represents the interaction torque due to contact with the environment. The robot dynamics given in Eq. (1) has the following useful properties.

Property 1: The inertia matrix is symmetric and positive definite.

Property 2: The matrix $\dot{M}(q) - 2V_m(q,\dot{q})$ is skew symmetric. The interaction force $\bar{F} \in R^m$ at the end-effector is related to the interaction torque $\tau_e \in R^n$ in the joint space through

$$\tau_e = J^T(q)\bar{F} \tag{2}$$

where $J(q) \in R^{m\times n}$ is the Jacobian matrix. The end effector position and orientation in cartesian space is related to the joint space by

$$x = f(q) \tag{3}$$

$$\dot{x} = J(q)\dot{q} \tag{4}$$

where $x \in R^m$ is the position and orientation vector and $f(q)$ is a kinematic transformation of a robot. The interaction force $\bar{F} \in R^m$ in task space is proportional to environmental deformation $(x - x_e)$ through

$$\bar{F} = \bar{K}(x - x_e) \tag{5}$$

where the stiffness matrix $\bar{K} \in R^{m\times m}$ is assumed to be constant and $x_e \in R^m$ is the coordinate of the point of contact. Practically stiffness matrix can be considered as constant for hard and homogeneous surfaces. By rearranging the terms Eq. (5) can be rewritten as

$$\bar{F} = \begin{bmatrix} F \\ F' \end{bmatrix} = \begin{bmatrix} K \\ K' \end{bmatrix}(x - x_e) \tag{6}$$

where $F \in R^p, F' \in R^{m-p}, K \in R^{p\times m} (p \le m)$ and is of full rank and $K' \in \mathrm{R}^{(m-p)\times m}$ depends linearly on the rows of K. p is the dimension of the subspace in the task space (R^m) where the forces are to be controlled. Since there are only n $(n \le 6)$ control inputs in the system, it is not possible to control force (six dimensions) and position (six dimensions) of the robot in all dimensions. The best, we can do is to control only a small subspace of the whole space. To be more specific, we can control p forces, $(m - p)$ cartesian positions and $(n - m)$ redundant joint velocities. The total dimension of the subspace is $p + (m - p) + (n - m) = n$.

The following general assumptions are made.

Assumption 1: The Jacobian $J(q)$ is of full rank.

Assumption 2: The stiffness matrix K is a known constant matrix.

2.2 Decomposition of Robot Dynamics

In this subsection, the robot dynamics is decomposed into force, position and redundant joint subspaces. The following identities are used in the decomposition process

$$I = J^+J + J^- \tag{7}$$

$$I = K^+K + K^- \tag{8}$$

where

$$J^+ = J^T[JJ^T]^{-1} \tag{9}$$

$$K^+ = K^T[KK^T]^{-1} \tag{10}$$

$$J^- = I - J^T[JJ^T]^{-1}J \tag{11}$$

$$K^- = I - K^T[KK^T]^{-1}K \tag{12}$$

where J^+ and K^+ are penrose pseudo inverse and J^- and K^- are projectors i.e., idempotent matrices.
Using Eq. (7) and Eq. (4), $\dot q$ can be decomposed as

$$\dot q = J^+J\dot q + J^-\dot q = J^+\dot x + J^-\dot q \tag{13}$$

Using Eq. (6) and Eq. (8), $\dot x$ can be decomposed as

$$\dot x = K^+K\dot x + K^-\dot x = K^+\dot F + K^-\dot x \tag{14}$$

Taking the first derivative of Eq. (4), we get

$$\ddot x = J\ddot q + \dot J\dot q \tag{15}$$

Using Eq. (7) together with Eq. (15), $\ddot q$ is decomposed as follows

$$\ddot q = J^+J\ddot q + J^-\ddot q = J^+(\ddot x - \dot J\dot q) + J^-\ddot q \tag{16}$$

Differentiation of Eq. (14), leads to

$$\ddot x = K^+K\ddot x + K^-\ddot x = K^+\ddot F + K^-\ddot x \tag{17}$$

Substituting Eq. (13), Eq. (14) and Eq. (16) into Eq. (1), we get

$$M\left(J^+\left[K^+\ddot F + K^-\ddot x - \dot J\dot q\right] + J^-\ddot q\right) + V_m\left(J^+\left[K^+\dot F + K^-\dot x\right] + J^-\dot q\right) + G = \tau - \tau_e \tag{18}$$

This is exactly equivalent to Eq. (1) and there is a clear decomposition of force (F), position (K^-x) and redundant joint velocity (J^-q).

3. Feedforward Neural Network

Mathematically, a two-layer feedforward neural network (see Fig. 1) with n input units, m output units and N units in the hidden layer, is described by Eq. (19). The output vector z is determined in terms of the input vector x by the formula

$$z_i = \sum_{j=1}^{N} [w_{ij}\sigma(\textstyle\sum_{k=1}^{n} v_{jk}y_k + \theta_{vj}) + \theta_{wi}], i = 1,2\ldots, m \tag{19}$$

where $\sigma(.)$ are the activation functions of the neurons of the hidden-layer. The inputs-to-hidden-layer interconnection weights are denoted by v_{jk} and the hidden-layer-to-outputs interconnection weights by w_{ij}. The bias weights are denoted by θ_{vj}, θ_{wi}. There are many classes of activation functions e.g. sigmoid, hyperbolic tangent and Gaussian. The sigmoid activation function used in our work, is given by

$$\sigma(y) = 1/(1 + e^{-y}) \tag{20}$$

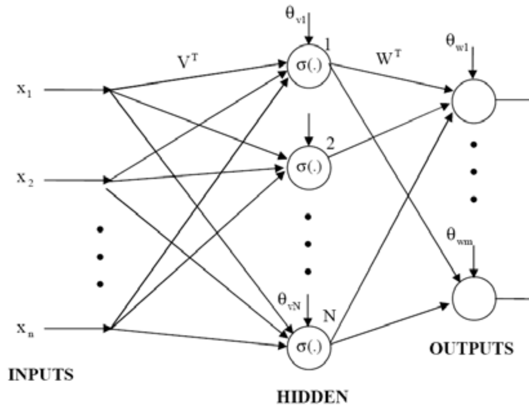

Fig. 1 Feedforward Neural Network

By collecting all the NN weights v_{jk}, w_{ij} into matrices of weights V^T, W^T, we can write the NN equation (19) in terms of vectors as

$$z = W^T\sigma(V^Ty) \tag{21}$$

with the vector of activation functions defined by $\sigma(x) = [\sigma(x_1), \sigma(x_2) \ldots \sigma(x_n)]^T$ for a vector $x \in R^n$. The bias weights are included as the first column of the weight matrices. To accommodate bias weights the vectors y and $\sigma(.)$ need to be augmented by replacing 1 as their first element e.g $y = [1, y_1, y_2 \ldots y_n]^T$.

3.1 Function Approximation Property

Let $h(y)$ be a smooth function from $R^n \to R^m$. Let $U_y \subseteq R^n$be a compact simply connected set then for $y \in U_y$, and an $\varepsilon > 0$ there exists some number of hidden layer neurons N and weights W and V such that[18]

$$h(y) = W^T\sigma(V^Ty) + \varepsilon \tag{22}$$

The value of ε is called the NN functional approximation error. In fact, for any choice of a positive number ε_N, one can find a NN such that $\varepsilon < \varepsilon_N$ in U_y. For a specified value of ε_N the ideal approximating NN weights exist. Then an estimate of $h(y)$ can be given by

$$\hat h(y) = \widehat W^T\sigma(\hat V^Ty) \tag{23}$$

where $\widehat W$ and $\hat V$ are estimates of the ideal NN weights that are provided by some on-line weight tuning algorithms.

4. Neural Network Based Controller Design

In this section, a neural network based controller is designed. Let us define filtered tracking error

$$r = J^+[K^+(\dot F - v_f) + K^-(\dot x - v_x)] + J^-\dot{\tilde q} \tag{24}$$

where

$$v_f = \dot F_d - \lambda\tilde F, v_x = \dot x_d - \lambda\tilde x \tag{25}$$

$$\tilde F = F - F_{d\,,}\ \tilde x = x - x_d, \tilde q = q - q_d \tag{26}$$

where q_d, x_d, F_d and their derivatives are the desired bounded values for q, x, F and their derivatives respectively. λ is a positive constant.

Differentiating Eq. (24) and using Eq. (25) and Eq. (26), the robot dynamics Eq. (18) can be written in terms of r as

$$M\dot r + V_m r = \tau - \tau_e - h(y) \tag{27}$$

where $h(y) = M\left[J^+\left(K^+\dot v_f + K^-\dot v_x - \dot J\dot q\right) + J^-\ddot q_d - Q\right] + V_m[J^+ (K^+v_f + K^-v_x) + J^-\dot q_d] + G$ is termed as robot nonlinear function and choose $y = [r, F, q, \dot q]$. Also $Q = (dJ^+/dt)\,[K^+(\dot F - v_f) + K^-(\dot x - v_x)] + dJ^-/dt$. To accomplish the desired goal, the following controller is proposed

$$\tau = \tau_e + \hat h(y) - K_d r \tag{28}$$

where $\hat h(y)$is an estimate of robot nonlinear function, $K_d = K_d{}^T$ is a positive definite gain matrix. With the controller in Eq. (28), the closed loop error dynamics becomes

$$M\dot r = -V_m r - \tilde h(y) - K_d r \tag{29}$$

where $\tilde h(y) = h(y) - \hat h(y)$ is the functional estimation error. The functional estimation $\hat h(y)$ with a feedforward neural network (FFNN) may be given as

$$\hat h(y) = \hat W^T \sigma(\hat V^T y) \tag{30}$$

Using this FFNN functional approximation, the closed loop error dynamics Eq. (29) becomes

$$M\dot r = -V_m r - K_d r - W^T\sigma(V^T y) - \varepsilon + \hat W^T\sigma(\hat V^T y) \tag{31}$$

For notational convenience, define the matrix of all NN weights as $Z \equiv diag\{W, V\}$ and the weight estimation errors as $\tilde W = W - \hat W, \tilde V = V - \hat V and \tilde Z = Z - \hat Z$

Define the hidden layer output error for a given y as $\tilde\sigma = \sigma - \hat\sigma = \sigma(V^T y) - \sigma(\hat V^T y)$. Adding and subtracting $W^T\sigma(\hat V^T y)$ in Eq. (31), we get

$$M\dot r = -V_m r - K_d r - W^T\left[\sigma(V^T y) - \sigma(\hat V^T y)\right] - \varepsilon - \tilde W^T\sigma(\hat V^T y) \tag{32}$$

The Taylor series expansion of $\sigma(V^T y)$about $\hat V^T y$ gives us

$$\sigma(V^T y) = \sigma(\hat V^T y) + \sigma'(\hat V^T y)\tilde V^T y + O(\tilde V^T y)^2 \tag{33}$$

with $\tilde V^T y = V^T y - \hat V^T y$, $\sigma'(\hat z) = (d\sigma(z)/dz)\,|_{z=\hat z}$, the Jacobian matrix and $O(s)^2$denoting terms of second order in s.
Denoting $\sigma'(\hat V^T y) = \hat\sigma'$, from Eq. (33) we have

$$\tilde\sigma = \hat\sigma'(\tilde V^T y) + O(\tilde V^T y)^2 \tag{34}$$

Using Eq. (34), in Eq. (32) yields

$$M\dot r = -V_m r - K_d r - \hat W^T\hat\sigma'\tilde V^T y - \tilde W^T\hat\sigma - w \tag{35}$$

Where the disturbance terms are

$$w = \tilde W^T\hat\sigma'\tilde V^T y + W^T O(\tilde V^T y)^2 + \varepsilon \tag{36}$$

The following bound on disturbance terms $w(t)$can be found with $c_o, c_1$and c_2 as positive constants[13]

$$\|w(t)\| \le c_o + c_1\|\tilde Z\|_F + c_2\|Z\|_F\|r\| \tag{37}$$

4.1 Neural Network Weight Update Law

Let the desired trajectory be bounded and assume the disturbance term w to be zero. With positive definite design parameters F_w and G_v, the adaptive NN weight update law is given as

$$\dot{\hat W} = -F_w\hat\sigma r^T, \dot{\hat V} = -G_v y(\hat\sigma'\hat W r)^T \tag{38}$$

Proof: Consider the following Lyapunov function candidate

$$L = (1/2)\ r^T M r + \tfrac{1}{2}tr(\tilde W^T F_w^{-1}\tilde W) + \tfrac{1}{2}tr(\tilde V^T G_v^{-1}\tilde V) \tag{39}$$

The time derivative of the Lyapunov function gives

$$\dot L = (1/2)\ r^T\dot M r + r^T M\dot r + tr(\tilde W^T F_w^{-1}\dot{\tilde W}) + tr(\tilde V^T G_v^{-1}\dot{\tilde V}) \tag{40}$$

Using Eq. (35) and $w = 0$ we get

$$\begin{aligned}\dot L = (1/2)\ r^T\dot M r + r^T(-V_m r - K_d r - \hat W^T\hat\sigma'\tilde V^T y - \tilde W^T\hat\sigma)\\ +tr(\tilde W^T F_w^{-1}\dot{\tilde W}) + tr(\tilde V^T G_v^{-1}\dot{\tilde V})\end{aligned} \tag{41}$$

$$\begin{aligned}\dot L = (1/2)\ r^T(\dot M - 2V_m)r - r^T K_d r - r^T\hat W^T\hat\sigma'\tilde V^T y - r^T\tilde W^T\hat\sigma\\ +tr(\tilde W^T F_w^{-1}\dot{\tilde W}) + tr(\tilde V^T G_v^{-1}\dot{\tilde V})\end{aligned} \tag{42}$$

Now using Property 2, together with $\dot{\tilde W} = -\dot{\hat W}, \dot{\tilde V} = -\dot{\hat V}$ and adaptive learning rule Eq. (38), we have

$$\begin{aligned}\dot L = -r^T K_d r - r^T\hat W^T\hat\sigma'\tilde V^T y - r^T\tilde W^T\hat\sigma + tr(\tilde W^T\hat\sigma r^T)\\ +tr(\tilde V^T y r^T\hat W^T\hat\sigma'^T)\end{aligned} \tag{43}$$

$$\begin{aligned}\dot L = -r^T K_d r + tr(-\tilde W^T\hat\sigma r^T + \tilde W^T\hat\sigma r^T)\\ +tr(-\tilde V^T y r^T\hat W^T\hat\sigma'^T + \tilde V^T y r^T\hat W^T\hat\sigma'^T)\end{aligned} \tag{44}$$

$$\dot L = -r^T K_d r \tag{45}$$

Since $L > 0$ and $\dot L \le 0$, this shows stability in the sense of Lyapunov so that $r(t)$, $\tilde V$ and $\tilde W$ (and hence $\hat V$,, $\hat W$) are bounded. Now from Eq. (24), using Eq. (6), we get[2]

$$r = J^+[K^+(K\dot x - v_f) + K^-(\dot x - v_x)] + J^-\dot{\tilde q} \tag{46}$$

Using Eq. (8), we get

$$r = J^+[\dot x - K^+v_f - K^-v_x] + J^-\dot{\tilde q} \tag{47}$$

$$r = J^+\dot x - J^+(K^+v_f + K^-v_x) + J^-\dot q - J^-\dot q_d \tag{48}$$

Now using Eq. (13), we get

$$r = \dot q - J^+(K^+v_f + K^-v_x) - J^-\dot q_d \tag{49}$$

Due to physical constraints. (Since we consider revolute joints only) x is bounded. Therefore from Eq. (25) F, v_f and $v_x$are bounded too. Also q_d is bounded. Thus we see that $\dot q$ is bounded. Differentiating r in Eq. (49) and using Property 1, it can be similarly argued that, for a revolute joint robot, $\dot r$ is also bounded. Hence $r \to 0$ as $t \to \infty$. Now K and J are bounded and are of full rank. Multiplying Eq. (24) by KJ, we get

$$KJr = \dot F - v_f \to 0 \text{ since } r \to 0 \tag{50}$$

Now multiplying Eq. (24) by J, we get

$$Jr = K^-(\dot x - v_x) \to 0 \tag{51}$$

Now using Eq. (24), Eq. (49) and Eq. (50), we get

$$J^-\dot{\tilde q} \to 0 \tag{52}$$

Now from Eq. (50) and Eq. (51) using Eq. (25) and Eq. (26), we get $\dot{\tilde F} + \lambda\tilde F \to 0$ and $K^-\dot{\tilde x} + \lambda K^-\tilde x \to 0$. Now $\tilde F$ and $K^-\tilde x$ can be considered as the outputs of filters of the form $1/(s+\lambda)$ with inputs of $\dot{\dot{\tilde F}} + \lambda\tilde F \to 0$ and $K^-\dot{\dot{\tilde x}} + \lambda K^-\tilde x \to 0$ respectively. Hence we see that the force error $\tilde F \to 0$, position error, $K^-\tilde x \to 0$ and redundant joint velocity error $J^-\dot{\tilde q} \to 0$.

5. Simulation Studies

The simulation has been performed for a two link rigid planar robotic manipulator as shown in the Fig. 2. The mathematical model of the manipulator is expressed as

$$\begin{bmatrix}M_{11} & M_{12}\\ M_{21} & M_{22}\end{bmatrix}\begin{bmatrix}\ddot q_1\\ \ddot q_2\end{bmatrix} + \begin{bmatrix}V_{m11} & V_{m12}\\ V_{m21} & V_{m22}\end{bmatrix}\begin{bmatrix}\dot q_1\\ \dot q_2\end{bmatrix} + \begin{bmatrix}G_1\\ G_2\end{bmatrix} = \begin{bmatrix}\tau_1\\ \tau_1\end{bmatrix} - J^T(q)\bar F$$

where the mass matrix and gravity terms are given as follows

$$M_{11} = {l_2}^2 m_2 + {l_1}^2(m_1 + m_2) + 2l_1 l_2 m_2 C_2$$
$$M_{12} = {l_2}^2 m_2 + l_1 l_2 m_2 C_2$$

$$M_{21} = l_2{}^2 m_2 + l_1 l_2 m_2 C_2$$
$$M_{22} = l_2{}^2 m_2$$
$$V_{m11} = -l_1 l_2 m_2 \dot{q}_2 S_2$$
$$V_{m12} = -l_1 l_2 m_2 (\dot{q}_1 + \dot{q}_2) S_2$$
$$V_{m21} = l_1 l_2 m_2 \dot{q}_1 S_2$$
$$V_{m22} = 0$$
$$G_1 = (m_1 + m_2) l_1 g_0 C_1 + m_2 l_2 g C_2$$
$$G_2 = m_2 l_2 g_0 C_{12}$$

where $C_1 = \cos q_1$, $C_2 = \cos q_2$, $C_{12} = \cos(q_1 + q_2)$ and $S_1 = \sin q_1$, $S_2 = \sin q_2, S_{12} = \sin(q_1 + q_2)$.

The parameter values for the manipulator model are set to be $m_1 = 15.61$ kg, $m_2 = 11.36$ kg, $l_1 = 0.432$ m, $l_2 = 0.432$ m, $g_0 = 9.8\ \mathrm{m/s^2}$.

The contact force only occurs in the x_1 direction and is given by $F = \kappa(x - x_{1e})$ where $\kappa = 10^4$ and $x_{1e} = 0.61$ m.

Also, $K^+ = 1/\kappa$, $K^- = [0\ 0\ ; 0\ 1]$, $J^+ = J^{-1}$, and $J^- = 0$.

The desired force $F_d = 20$ N. The desired trajectory x_{2d} in x_2 direction is $0.5[1 - exp(-t)]$.

The controller parameters are $\lambda = 100$, $K_d = 200 I_2$.

The architecture of the FFNN is composed of 7 input units and 1 bias unit, 5 hidden sigmoidal units and 1 bias unit and 2 output units. The NN weights may be simply initialized to zero and errors may be kept arbitrarily small. The learning rate in the weight tuning algorithm is $F_w = 100 I_6, G_v = 100 I_8$.

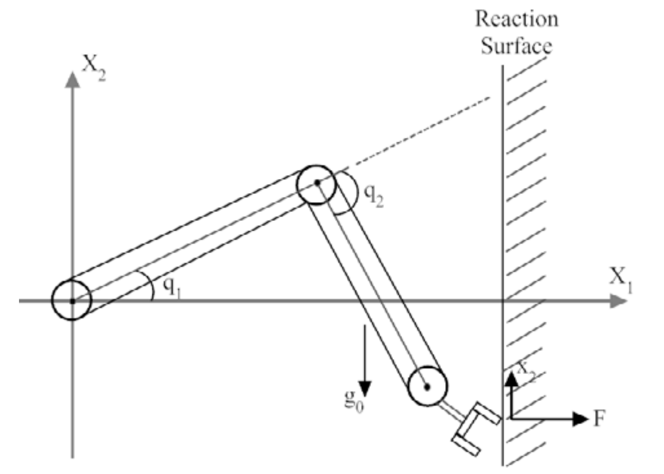

Fig. 2 Two Link Robot Manipulator Contacting a Surface

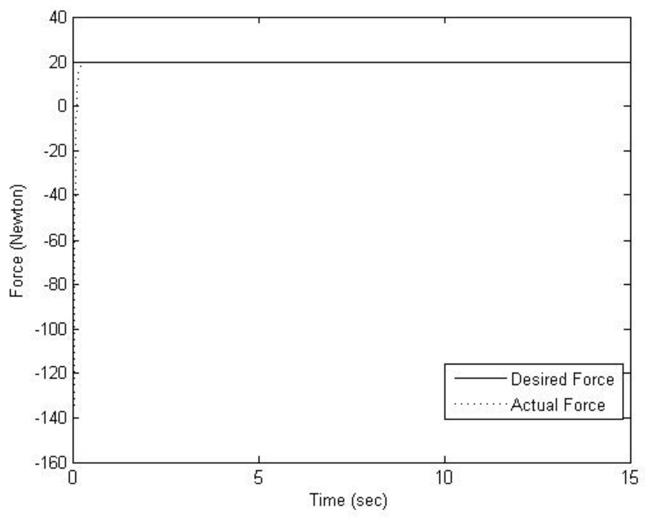

Fig. 3 Force Tracking with Known Dynamics

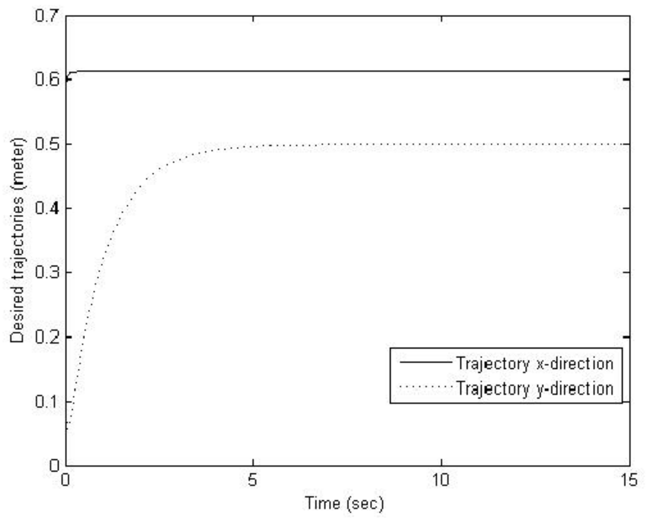

Fig. 4 End-effector Trajectory Tracking with Known Dynamics

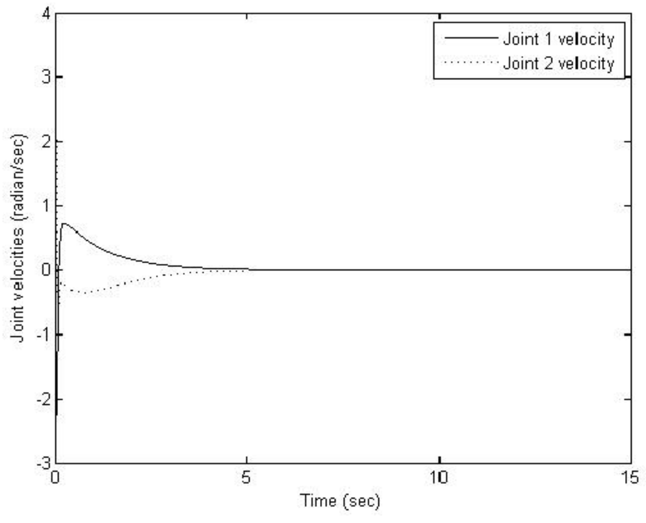

Fig. 5 Joint Velocities with Known Dynamics

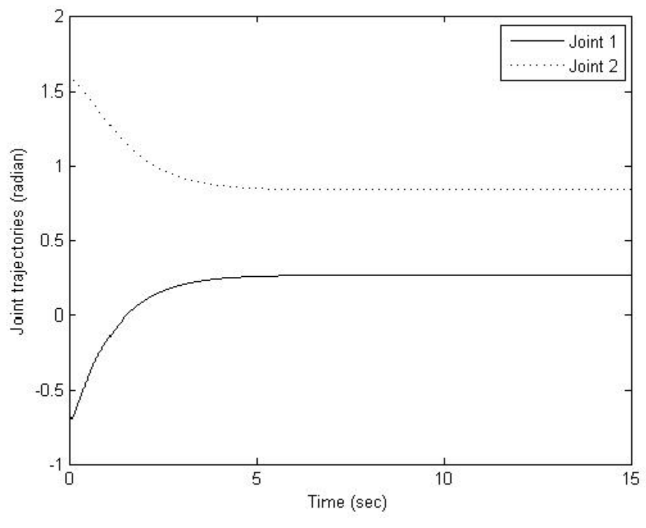

Fig. 6 Joint Trajectories with Known Dynamics

The simulation of the whole system is shown for 15 seconds. The performance of the controller with known dynamics is evident from Fig. 3-6. The position trajectory and force tracking errors are quickly convergent to zero and the tracking is stable. However, the incorrect knowledge of the system parameters causes steady state error to increase. This is evident from Fig. 7-8. The parameter

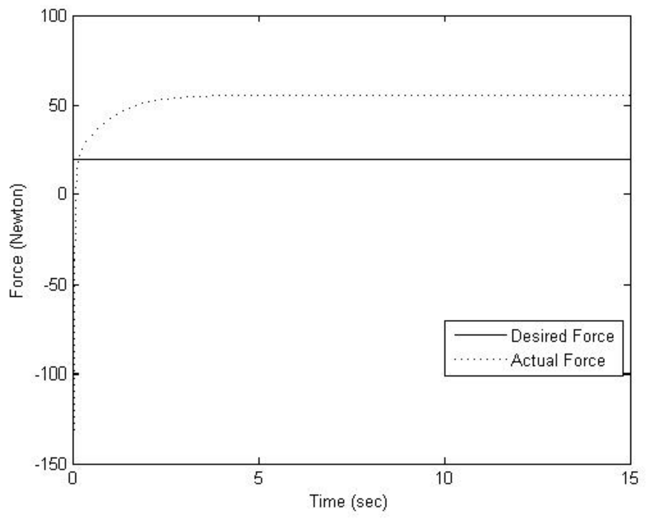

Fig. 7 Force Tracking with Incorrect Dynamics

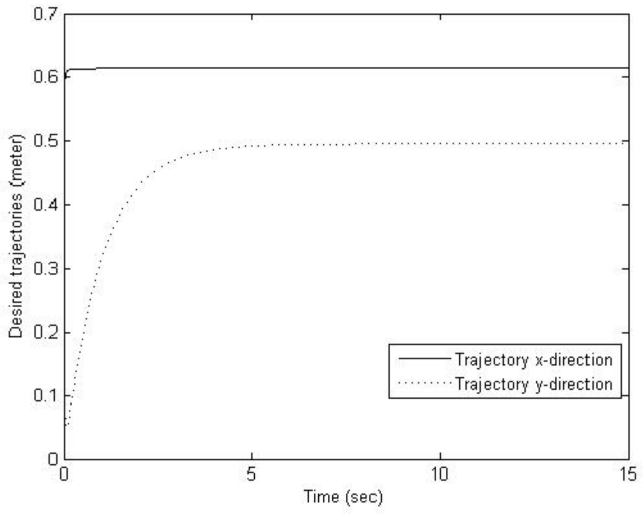

Fig. 8 End-effector Trajectory Tracking with Incorrect Dynamics

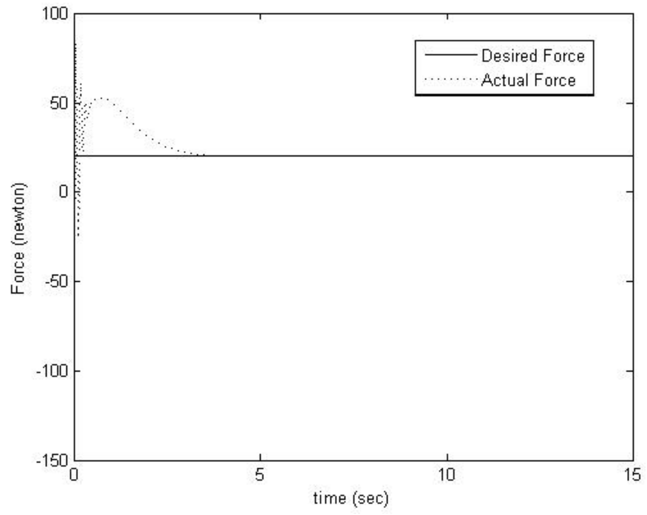

Fig. 9 Force tracking with NN Learned Dynamics

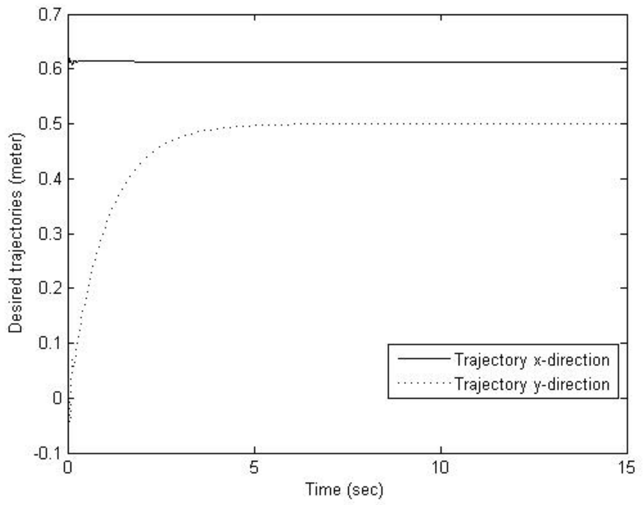

Fig. 10 End-effector Trajectory Tracking with NN Learned Dynamics

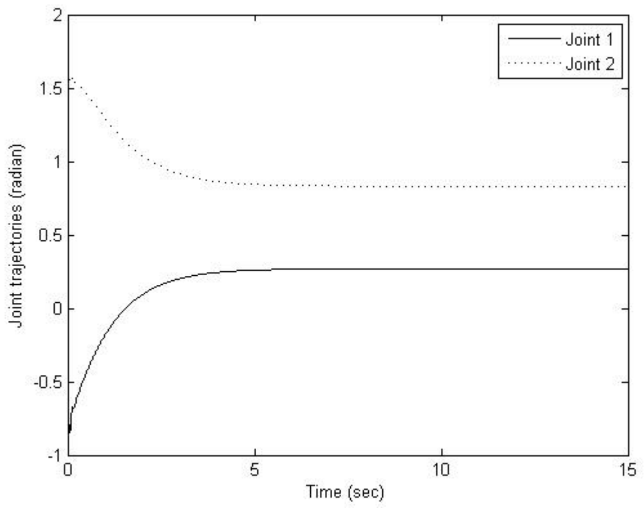

Fig. 11 Joint Trajectories with NN Learned Dynamics

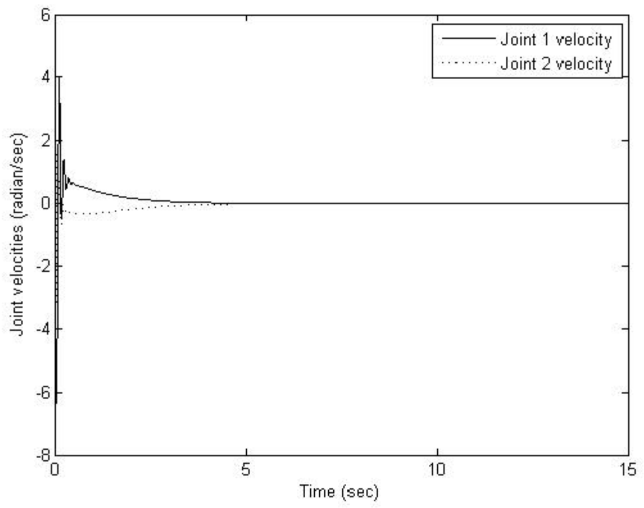

Fig. 12 Joint Velocities with NN Learned Dynamics

values for system parameters in the controller for incorrect dynamics are taken as $m_1 = 14$ kg, $m_2 = 10$ kg, $l_1 = 0.4432$ m, $l_2 = 0.4432$ m. Thus the variation of parameters from its true values adversely affects the performance of the controller The ability of the FFNN to learn the unknown dynamics is evident from Fig. 9-12. The trajectory and force tracking steady state errors are convergent to zero as compared to Fig. 7-8. Finally the tracking of a time varying force trajectory taken as $\mathrm{F_d} = 20(1 - 0.5\sin\mathrm{t})$ is shown in Fig. 13 with known dynamics and Fig. 14 with NN Learned dynamics.

6. Conclusion

In this paper, a neural network based adaptive control scheme for hybrid force/position control for rigid robot manipulators. Firstly the robot dynamics is decomposed into force, position and redundant joint subspaces. Based on this decomposition, a controller is proposed that achieves desired interaction force between the end-effetor and the environment as well as regulate robot tip position in Cartesian space. A feedforward neural network is employed to learn the existing unknown dynamics of redundant robot manipulator, which requires no preliminary learning. The stability of the system is proved using Lyapunov function, generated by weighting matrices. Finally simulation is carried out for a two link rigid robot manipulator to illustrate the control methodology. The simulation results show that the feedforward neural network with the on-line updating law can compensate the full robot dynamics effectively.

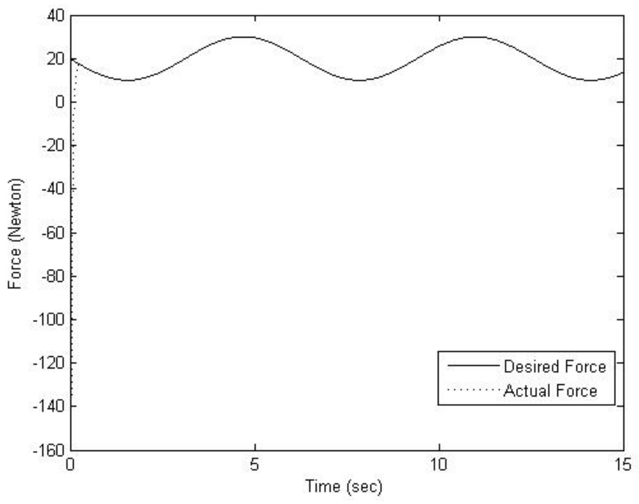

Fig. 13 Tracking of Time Varying Force Trajectory with Known Dynamics

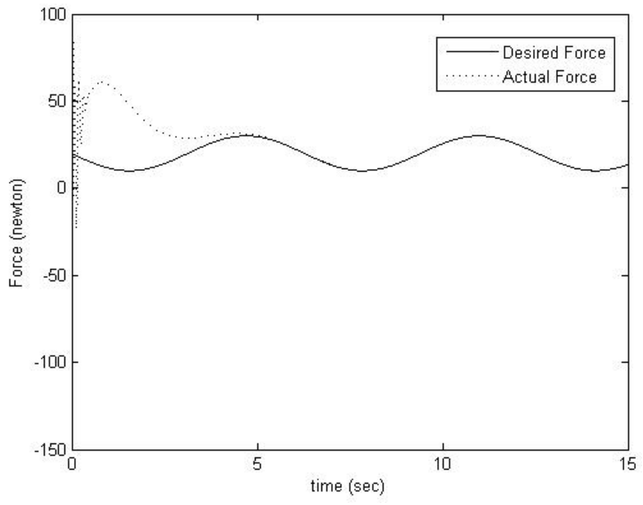

Fig. 14 Tracking of Time Varying Force Trajectory with NN Learned Dynamics

ACKNOWLEDGEMENT

This research has been supported by BK21 Project, Division of Mechanical Engineering, Ajou University, Suwon-443749, Korea.

REFERENCES

1. Raibert, M. H. and Craig, J. J., “Hybrid Position/Force Control of Manipulators,” ASME Journal of Dynamical Systems, Measurement and Control, Vol. 103, No. 2, pp. 126-133, 1981.
2. Lozano, R. and Brogliato, B., “Adaptive Hybrid Position/Force Control of Redundant Manipulators,” IEEE Transactions on Automatic Control, Vol. 37, No. 10, pp. 1501-1505, 1992.
3. Yoshikawa, T. and Sudou, A., “Dynamic Hybrid Position/Force Control of Robot Manipulators: Online-Estimation of Unknown Constraint,” IEEE Transactions on Robotics and Automation, Vol. 9, No. 2, pp. 220-226, 1993.
4. Kwan, C. M., “Robust Adaptive Force/Motion Control of Constrained Robots,” IEE Proc.-Control Theory and Applications, Vol. 143, No. 1, pp. 103-109, 1996.
5. De Queiroz, M. S., Jun, H. and Dawson, D. M., “Adaptive Position/Force Control of Robot Manipulators without Velocity Measurements: Theory and Experimentation,” IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics, Vol. 27, No. 5, pp. 796-809, 1997.
6. Xiao, D., Ghosh, B. K., Ning, X. and Tarn, T. J., “Dynamic Hybrid Position/Force Control of a Robot Manipulator in an uncalibrated environment,” IEEE Transactions on Control Systems Technology, Vol. 8, No. 4, pp. 635-645, 2000.
7. Kouya, D. N., Saad, M. and Lamarche, L., “Backstepping Adaptive Hybrid Force/Position Control for Robotic Manipulators,” Proc. of the American Control Conference, pp. 4595-4600, 2002.
8. Roy, J. and Whitcomb, L. L., “Adaptive Force Control of Position/Velocity Controlled Robots: Theory and Experiment,” IEEE Transactions on Robotics and Automation, Vol. 18, No. 2, pp. 121-137, 2002.
9. Cheah, C. C., Zhao, Y. and Slotine, J. J. E., “Adaptive Jacobian

Motion and Force Control for Constrained Robots with Uncertainties," Proc. of the International Conference on Robotics and Automation, pp. 2226-2231, 2006.

10. Kang, S., Komoriya, K., Yokoi, K., Koutoku, T., Kim, B. and Park, S., "Control of Impulsive Contact Force Between Mobile Manipulator and Environment Using Effective Mass and Damping Controls," Int. J. Precis. Eng. Manuf., Vol. 11, No. 5, pp. 697-704, 2010.

11. Lewis, F. L., Liu, K. and Yesildirek, A., "Neural Net Robot Controller with Guaranted Tracking Performance," IEEE Transactions on Neural Networks, Vol. 6, No. 3, pp. 703-715, 1995.

12. Lewis, F. L., Yesildirek, A. and Liu, K., "Multilayer Neural Net Robot Controller with Guaranted Tracking Performance," IEEE Transactions on Neural Networks, Vol. 7, No. 2, pp. 388-399, 1996.

13. Lewis, F. L., Jagannathan, S. and Yesildirek, A., "Neural Networks Control of Robot Manipulators and Nonlinear Systems," Taylor and Francis, 1999.

14. Shenghai, H., Marcelo, H. A. and Krishnan, H., "Online Neural Networks Compersator for Constrained Robot Manipulators," Proc. of the 3rd Asian Control Conference, pp. 1621-1627, 2000.

15. Karayiannidis, Y., Rovithakis, K. and Doulgeri, Z., "Force/ Position Tracking for a Robotic Manipulator in compliant contact with a surface using Neuro-adaptive Control," Automatica, Vol. 43, No. 7, pp. 1281-1288, 2007.

16. Panwar, V. and Sukavanam, N., "Design of Optimal Hybrid Position/Force Controller for a Robot Manipulator using Neural Networks," Mathematical Problems in Engineering, Vol. 23, Article ID. 65028, 2007.

17. Zhao, Y. and Cheah, C. C., "Vision Based Neural Network Control for Constrained Robots with Constraint Uncertainty," IET Control Theory and Applications, Vol. 2, No. 10, pp. 906-916, 2008.

18. Hornic, K., Stinchcombe, M. and White, H., "Multilayers Feedforward Networks are Universal Approximators," Neural Networks, Vol. 2, No. 5, pp. 359-366, 1989.