Proceedings of the 3rd
Asian Control Conference
July 4-7,2000,Shanghai

On-line Neural Network Compensator for Constrained Robot Manipulators

Shenghai Hu, Marcelo H. Ang Jr., and H. Krishnan
Dept of Mechanical and Production Engineering
National University of Singapore
Singapore 119260
mpeangh@nus.edu.sg

Abstract

In this paper, a new neural network controller for the constrained robot manipulators in task space is presented. The neural network will be used for adaptive compensation of the structured and unstructured uncertainties. The controller consisted of a model-based term and a neural network on-line adaptive compensation term. It is shown that the neural network adaptive compensation is universally able to cope with totally different classes of system uncertainties. Novel adaptive learning algorithms for tuning the weights of neural network are proposed. A suitable error filtered signal for training the neural network can be easily obtained from the controller design without using any model knowledge of the robot manipulator itself. The closed-loop system with neural network adaptation on line is guaranteed to be stable in the Lyapunov sense. Detailed simulation results are given to show the effectiveness of the proposed controller.

1. Introduction

To apply robot manipulators to a wider class of tasks, it is necessary to control not only the position of a manipulator but also the force exerted by its end-effector on an object or environment.

Force control of manipulators has been studied by many researchers [1]-[3]. Constrained motion control has been extensively studied in recent years. In constrained motion control, the robot's end-effector is assumed to be in contact with rigid frictionless surfaces [5]. As a result, kinematic constraints are imposed on the manipulator motion, which correspond to some algebraic constraints among the manipulator state variables. It is necessary to control both the motion of the robot's end effector on the constraint surfaces and the generalized constrained forces.

A general theoretical framework of constrained motion control is rigorously developed in [5]. The proposed controller is based on a modification of the computed torque method. In [4], linear descriptor system theory is applied to design control laws for constrained motion control. The controller is derived based on a linearized dynamic model of the manipulator. In [6], state feedback control and dynamic state feedback control are used to linearize the robot dynamics with respect to motion and contact force subsystems respectively.

The above methods of controller design are based on the knowledge of the exact dynamic model of constrained robot systems. From a practical point of view, in many applications, robot models have many uncertainties in the values of the parameters describing its dynamic properties, such as unknown moments of inertia. In addition, there is also the problem of unmodelled dynamics (e.g., friction). It is hard to estimate the exact form of the model and the values of the dynamic parameters, thus complicating the control design problem significantly. This has motivated the use of adaptive control, sliding mode control, robust control, etc for controller design for constrained robots.

The ability of the neural network (NN) to approximate arbitrary non-linear functions and to learn through examples lends it to many useful applications in control engineering. Many researchers have applied the NN in robot motion control with substantial success [7, 8]. Few research works (Katic and Vukobratovic [10], Yamada and Yabuta [9]) have done in implementing NN to the robot force control.

In this paper, we consider the design of NN controllers for force control in constrained robots. A nonlinear transformation is used to decouple the robot dynamics into two subsystems – motion subsystem and force subsystem respectively. A NN control law is proposed based on the decoupled dynamic equations, and a suitable online update law for the NN is derived. Simulation results illustrate the effectiveness of the proposed controller.

2. Dynamic Model of Constrained Robot

Based on the Euler-Lagrangian formulation, in the absence of friction, the motion equation of an n-link rigid, non-redundant constrained robot can be expressed in joint space as

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = f + u \tag{1}$$

where $q \in R^n$ is the joint displacement vector, $u \in R^n$ is the joint space torque; $M(q) \in R^{n*n}$ is the inertia matrix, $C(q,\dot{q})\dot{q} \in R^n$ is the vector characterizing Coriolis and Centrifugal forces, and $g(q) \in R^n$ is the gravitational force $f \in R^n$ is the vector of constraint forces in joint space.

Three simplifying properties should be noted about the dynamic structure in Equation (1) [12].

Property 1. $M(q)$ is a symmetric positive definite matrix, and is bounded both above and below, that is, there exist positive constants α_M and β_M such that,

$$\alpha_M I_n \le M(q) \le \beta_M I_n$$

Property 2. Given a proper definition of the matrix C, the matrix $\dot{M}(q)-2C(q,\dot{q})$ is skew symmetric.

Property 3. $M(q)$, $C(q,\dot{q})$, $G(q)$ are linear in terms of a suitable selected set of the robot parameters.

Let $p \in R^n$ denote the generalized position vector of the end-effector in Cartesian space. If the constraints imposed are described by a holonomic smooth manifold, then the algebraic equation for the constraints can be written as

$$\Phi(p)=0 \tag{2}$$

where the mapping $\Phi: R^n \to R^m$ is twice continuously differentiable, m is the times of the constraints.

Assuming that the vector p can be expressed in joint space by the relation

$$p=h(q) \tag{3}$$

where the mapping $h: R^n \to R^n$ is invertible and twice continuously differentiable, then the constrained equation in joint space can be written as

$$\Theta(q)=\Phi(h(q))=0 \tag{4}$$

The Jacobian matrix of the constrained equation (4) is

$$J(q)=\frac{\partial\Theta(q)}{\partial q}=\frac{\partial\Phi}{\partial p}\frac{\partial h(q)}{\partial q} \tag{5}$$

which is nonsingular due to the assumption that the robot is nonredundant, and the robot is at a nonsingual configuration.

When the end-effector is moving along the constrained surface, the constraint force in joint space is then given by

$$f=J^T(q)\lambda \tag{6}$$

where $\lambda \in R^m$ is the associated Lagrangian multiplier [5].

When motion of the robot is constrained to be on the surfaces (2), only *(n-m)* coordinates of the position vector can be specified independently. Control of all the position coordinates need to be controlled in the constrained motion of the robot. Therefore, motion control is in the $(n-m)$ mutually independent coordinates, $\Psi(p)=[\psi_1(p),\cdots,\psi_{n-m}(p)]^T$, $\Psi(p)$ are assumed to be twice continuously differentiable and independent of $\Phi(p)$ in the finite workspace Ω. Thus, once $\Psi(p)$ is regulated to the desired value $\Psi_d(t)$, combining with the constraints (2), the end-effector configuration of robot is uniquely determined.

Define a set of coordinates as

$$r=[r_f^T, r_p^T]^T \quad r_f=[\phi_1(p),\cdots,\phi_m(p)]^T$$
$$r_p=[\psi_1(p),\cdots,\psi_{n-m}(p)]^T \tag{7}$$

Differentiate (7), we have

$$\dot{r}=J_p\dot{p}=J_q\dot{q} \tag{8}$$

where

$$J_p=\frac{\partial r(p)}{\partial p}=\begin{bmatrix}\frac{\partial\Phi^T(p)}{\partial p} & \frac{\partial\Psi^T(p)}{\partial p}\end{bmatrix}^T$$
$$J_q=\frac{\partial r(H(q))}{\partial q}=\begin{bmatrix}J^T(q) & P^T(q)\end{bmatrix}^T$$
$$P(q)=\frac{\partial\Psi}{\partial p}\frac{\partial p}{\partial q} \tag{9}$$

From its inverse, we obtain

$$\dot{q}=Q(q)\dot{r} \tag{10}$$

where

$$Q(q)=\begin{bmatrix}J(q)\\ P(q)\end{bmatrix}^{-1} \tag{11}$$

Furthermore

$$\ddot{q}=Q(q)\ddot{r}+\dot{Q}(q)\dot{r} \tag{12}$$

Substitute (6), (10), (12) into (1), we have

$$M(q)Q\ddot{r}+C_1(q,\dot{q})\dot{r}+g(q)=u+J^T(q)\lambda \tag{13}$$

where $C_1(q,\dot{q})=M(q)\dot{Q}(q)+C(q,\dot{q})Q(q)$ (14)

Multiply both sides with $Q^T(q)$, then the dynamic equation (13) can be expressed in terms of the new coordinates,

$$\overline{M}(q)\ddot{r}+\overline{C}(q,\dot{q})\dot{r}+\overline{g}(q)=Q^T(q)u+Q^T(q)J^T(q)\lambda \tag{15}$$

where

$$\overline{M}(q)=Q^T(q)M(q)Q(q) \tag{16}$$
$$\overline{C}(q,\dot{q})=Q^T(q)C_1(q,\dot{q}) \tag{17}$$
$$\overline{g}(q)=Q^T(q)g(q) \tag{18}$$

Define the following partitioning matrix

$$\begin{bmatrix}E_1\\ E_2\end{bmatrix}=\begin{bmatrix}I_m & 0\\ 0 & I_{n-m}\end{bmatrix} \tag{19}$$

Evaluate the dynamic term in equation (15), $\Phi(q)=0 \Rightarrow r_f=0,\ \dot{r}_f=\ddot{r}_f=0$. we have $r=\begin{bmatrix}r_f=0\\ r_p\end{bmatrix}=E_2^T r_p$, then the dynamic equation (15) can be expressed in two parts

$$E_1\overline{M}E_2^T\ddot{r}_p+E_1\overline{C}E_2^T\dot{r}_p+E_1\overline{g}=E_1Q^Tu+\lambda \tag{20}$$
$$E_2\overline{M}E_2^T\ddot{r}_p+E_2\overline{C}E_2^T\dot{r}_p+E_2\overline{g}=E_2Q^Tu \tag{21}$$

In (20) and (21) we have used the fact that $E_1Q^T(q)J^T(q)\lambda=\lambda$, $E_2Q^T(q)J^T(q)=0$.

By exploiting the structure of the equation (15), two properties could be obtained.

Property 4. The matrix $\overline{M}(q)$ is symmetric and positive definite. Also $E_2M(q)E_2^T$ is symmetric and positive definite. And $\overline{M}(q)$ is upper bounded and lower bounded, i.e. there exist two positive constants $\overline{\alpha}_M$ and β_M such that,

$$\overline{\alpha}_M I_n \le \overline{M}(q) \le \overline{\beta}_M I_n$$

The proof is given in the Appendix.

Property 5. The Matrix $\dot{\overline{M}}(q)-2\overline{C}(q,\dot{q})$ is skew symmetric.

The proof is given in the Appendix.

The actual position along the free motion z_1 (in task space) is

$$z_1=r_p \tag{22}$$

while satisfying Equation (2) (i.e., the robot maintains contact on the constrained surface). The actual contact forces z_2 is

$$z_2=\lambda \tag{23}$$

The manipulator is required to track a time-varying position trajectory $r_{pd}(t)$ and a time-varying force trajectory $\lambda_d(t)$. The control objective is to find a feedback control law so that the constrained manipulator's actual position and force track the desired position and force trajectory $r_{Pd}(t)$ and $\lambda_d(t)$ respectively, i.e.

$$z_1(t) - r_{Pd}(t) = r_p - r_{Pd}(t) \to 0\,, \quad \text{as } t \to \infty$$

$$z_2(t) - \lambda_d(t) = \lambda(t) - \lambda_d(t) \to 0\,, \quad \text{as } t \to \infty\,.$$

3. Controller for Constrained Robot

In this section, based on the derived dynamic equation of constrained robot (15), an on-line adaptive NN controller is developed for solving the adaptive motion and force control problem.

Defining

$$e_p = r_p - r_{pd} \tag{24}$$

$$e_f = \int_0 (\lambda - \lambda_d)dt \tag{25}$$

$$\dot{r}_{pr} = \dot{r}_{pd} - \Lambda_1 e_p - \Lambda_2 e_f \tag{26}$$

where e_p is the tracking error; e_f is the accumulated force error; r_{pr} is a "combined" reference trajectory; Λ_1 and Λ_2 are tunable matrices.

Define s as

$$s = \dot{r}_p - \dot{r}_{pr} = \dot{e}_p + \Lambda_1 e_p + \Lambda_2 e_f \tag{27}$$

The proposed control law is

$$u = Q^{-T}\hat{\overline{M}}E_2^T\ddot{r}_{pr} + Q^{-T}\hat{\overline{C}}E_2^T\dot{r}_{pr} + Q^{-T}\hat{\overline{g}} - J^T(q)r_2 - QE_2^T s + v$$

$$= \hat{M}QE_2^T\ddot{r}_{pr} + \hat{C}_1 E_2^T\dot{r}_{pr} + \hat{g} - J^T r_2 - QE_2^T s + v \tag{28}$$

where $\hat{M}(q)$, $\hat{C}_1(q,\dot{q})$ and $\hat{g}(q)$ are the estimates of $M(q)$, $C_1(q,\dot{q})$ and $g(q)$. $\hat{\overline{M}}(q)$, $\hat{\overline{C}}(q,\dot{q})$ and $\hat{\overline{g}}(q)$ are estimates of $\overline{M}(q)$, $\overline{C}(q,\dot{q})$ and $\overline{g}(q)$. v is the neural network compensator signal counteracting the manipulator uncertainties.

Substitute the control law (28) into (13), we obtain

$$\begin{aligned} MQE_2^T\dot{s} &= -C_1 E_2^T s - QE_2^T s - \Delta MQE_2^T\ddot{r}_p - \Delta C_1 E_2^T\dot{r}_p \\ &\quad - \Delta g + v - J^T(r_2 - \lambda) = -C_1 E_2^T s \\ &\quad - QE_2^T s - (\eta(\dot{r}_{pr}, \ddot{r}_{pr}) - v) - J^T(r_2 - \lambda) \end{aligned} \tag{29}$$

where

$$\Delta M = M(q) - \hat{M}(q)\,,$$

$$\Delta C_1 = C_1(q,\dot{q}) - \hat{C}_1(q,\dot{q})\,,$$

$$\Delta g = g(q) - \hat{g}(q)\,, \text{ and}$$

$$\eta(\dot{r}_{pr}, \ddot{r}_{pr}) = \Delta MQE_2^T\ddot{r}_{pr} + \Delta C_1 E_2^T\dot{r}_{pr} + \Delta g \tag{30}$$

$\eta(\dot{r}_{pr}, \ddot{r}_{pr})$ represents the uncertainties in the robot dynamics.

Multiply both sides with E_2Q^T, and using the property $E_2Q^T(q)J^T(q) = 0$, the above equation becomes

$$\begin{aligned} E_2Q^t MQE_2^t\dot{s} &= -E_2Q^t C_1 E_2^t s - E_2Q^t QE_2^t s \\ &\quad - E_2Q^T(\eta(\dot{r}_{pr}, \ddot{r}_{pr}) - v) \end{aligned} \tag{31}$$

4. Neural Network Compensator Design

The two-layer feedforward neural network shown in Fig.1 is used as the compensator. It is composed of an input buffer, a non-linear hidden layer, and a linear output layer. The mathematical representation of the network is given by

$$v = W^2 f(W^1 x + b^1) + b^2 \tag{32}$$

where $W^1 \in R^{n_H \times 3(n-m)}$, $W^2 \in R^{n \times n_H}$ denote the interconnection weights for the hidden and output layers respectively, $b^1 \in R^{n_H \times 1}$, $b^2 \in R^{n \times 1}$ the bias terms to the nodes of the corresponding layer, $x \in R^{3*(n-m) \times 1}$. n_H is the number of hidden units.

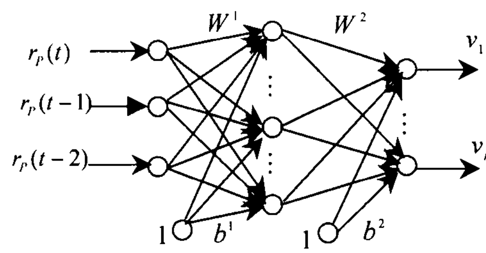

Fig. 1. NN Controller Structure

Assumed that the system uncertainties $\eta(\dot{r}_{pr}, \ddot{r}_{pr})$ in equation (30) can be approximated by an NN with an approximation error of ε_1:

$$\eta(\dot{r}_{pr}, \ddot{r}_{pr}) = v(W^*,\ b^*,\ x) + \varepsilon_1 \tag{33}$$

where $v(W^*,\ b^*,\ x)$ is the best approximation of $\eta(\dot{r}_{pr}, \ddot{r}_{pr})$ using the feedforward neural network.

The difference between the system uncertainties and the NN compensation maybe expressed as

$$\Delta v = v(W^*,\ b^*,\ x) - v(W,b,x) + \varepsilon_1 \tag{34}$$

where W, b are the current weights of the NN, W^*, b^* are the best weights of the NN.

Theorem 1

If the initial weights W and biases b in equation (34) are in the neighborhood of the desired weights W^* and biases b^*, then equation (34) becomes

$$\begin{aligned} \Delta v &= \widetilde{W}^2 f(\vartheta) + \widetilde{b}^2 + W^2 f'(\vartheta)(\widetilde{W}^1 x + \widetilde{b}^1) \\ &\quad + \varepsilon_2(\varsigma) + \varepsilon_1 \end{aligned} \tag{35}$$

where $\widetilde{W}^1 = W^{1^*} - W^1$, $\widetilde{W}^2 = W^{2^*} - W^2$, $\widetilde{b}^2 = b^{2^*} - b^2$, $\widetilde{b}^1 = b^{1^*} - b^1$ and $\varepsilon_2(\varsigma)$ denotes the vector of high-order terms of the Taylor expansion of $v(W,b,x)$ at W^* and b^*.

The proof is given in the Appendix.

Consider the Lyapunov function candidate

$$\begin{aligned} V &= s^T E_2\overline{M}E_2^T s^T + tr(\widetilde{W}^{1^T}\Gamma_1^{-1}\widetilde{W}^1) + tr(\widetilde{W}^{2^T}\Gamma_2^{-1}\widetilde{W}^2) \\ &\quad + tr(\widetilde{b}^{1^T}\Gamma_1^{-1}\widetilde{b}^1) + tr(\widetilde{b}^{2^T}\Gamma_2^{-1}\widetilde{b}^2) \end{aligned} \tag{36}$$

where $\Gamma_1 \in R^{n_H * n_H}$, $\Gamma_2 \in R^{n*n}$ denote the diagonal positive-definite matrices.

Differentiating V with respect to time yields

$$V = \dot{s}^T E_2 M E_2^T s^T + s^T E_2 M E_2^T s + s^T E_2 M E_2^T \dot{s}$$
$$+ tr(\tilde{W}^{1^T} \Gamma_1^{-1} \tilde{W}^1) + tr(\tilde{W}^{2^T} \Gamma_2^{-1} \tilde{W}^2)$$
$$+ tr(\tilde{b}^{1^T} \Gamma_1^{-1} \tilde{b}^1) + tr(\tilde{b}^{2^T} \Gamma_2^{-1} \tilde{b}^2) \quad (37)$$

Theorem 2

Consider the system equations (31), and the rate of change in the Lyapunov function equation (37). If the adaptive weights updating laws are set to

$$\dot{W}^1 = -\dot{\tilde{W}}^1 = -\Gamma_1 f'(\vartheta) W^{2^T} Q E_2^T s x^T \quad (38)$$

$$\dot{W}^2 = -\dot{\tilde{W}}^2 = -\Gamma_2 Q E_2^T s f^T(\vartheta) \quad (39)$$

$$\dot{b}^1 = -\dot{\tilde{b}}^1 = -\Gamma_1 f'(\vartheta) W_2^T Q E_2^T s \quad (40)$$

$$\dot{b}^2 = -\dot{\tilde{b}}^2 = -\Gamma_2 Q E_2^T s \quad (41)$$

then

$$\dot{V} = -2 s^T E_2 Q^T Q E_2^T s < 0 \quad (42)$$

The proof is given in the Appendix.

From (36) and (42), it is evident that $\|s\|$ at least converges exponentially to zero, i.e., $e_m \to 0$, and $e_f \to 0$ as $t \to \infty$.

5. Simulation Results

The simulation has been performed to evaluate the effectiveness of the proposed neural network controller using the model of five-bar linkage parallelogram robot with two degrees of freedom (DOF) as shown in Fig. 2.

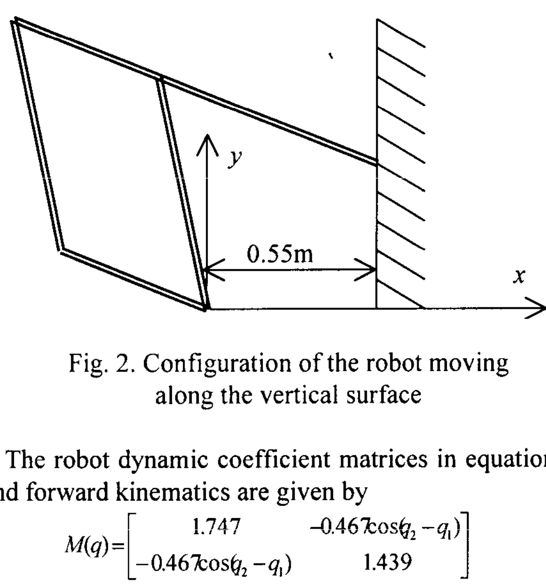

Fig. 2. Configuration of the robot moving along the vertical surface

The robot dynamic coefficient matrices in equation (1) and forward kinematics are given by

$$M(q) = \begin{bmatrix} 1.747 & -0.4767\cos(q_2 - q_1) \\ -0.4767\cos(q_2 - q_1) & 1.439 \end{bmatrix} \quad (43)$$

$$C(q_1, q_2, \dot{q}_1, \dot{q}_2) = \begin{bmatrix} 0 & 0.476\sin(q_2 - q_1)\dot{q}_2 \\ -0.476\sin(q_2 - q_1)\dot{q}_1 & 0 \end{bmatrix} \quad (44)$$

$$g(q_1, q_2) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad (45)$$

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0.4\cos(q_1) - 0.61\cos(q_2) \\ 0.4\sin(q_1) - 0.61\sin(q_2) \end{bmatrix} \quad (46)$$

The robot is in contact with a rigid frictionless surface as shown in Fig. 2. It satisfies $\Phi(p) = x - 0.55 = 0$ when expressed in task space. The constraint can also be expressed in joint variables as $0.4\cos(q_1) - 0.61\cos(q_2) - 0.55 = 0$. It then follows that $J(q) = [-0.4\sin(q_1), 0.61\sin(q_2)]$.

The nonlinear transformation is selected as

$$r = \begin{bmatrix} r_f \\ r_p \end{bmatrix} = \begin{bmatrix} 0.4\cos q_1 - 0.61\cos q_2 - 0.55 \\ 0.4\sin q_1 - 0.61\sin q_2 \end{bmatrix} \quad (47)$$

The position output is

$$y(t) = r_p = 0.4\sin q_1 - 0.61\sin q_2 \quad (48)$$

The NN controller is chosen as three inputs, $X = [r_{pr}(t-2), r_{pr}(t-1), r_{pr}(t)]^T$, six hidden neurons and two output neurons. All the weights and biases are set randomly in $[-0.5 \sim 0.5]$ initially. The weights are updated at each sampling time in on-line fashion. The controller gains are selected as $\Lambda_1 = 30$, $\Lambda_2 = 1$, $\Gamma_1 = 0.04 I_9$ and $\Gamma_2 = 0.04 I_2$. Sampling time is 0.001s. The performances of the proposed scheme are tested by tracking desired motion and force trajectories under different conditions.

Case 1:

To simulate uncertainties exist in the robot, here we choose the nominal dynamic parameter of the robot as

$$\hat{M}(q) = \begin{bmatrix} 1 & -0.2\cos(q_2 - q_1) \\ -0.2\cos(q_2 - q_1) & 0.5 \end{bmatrix} \quad (49)$$

$$C(q_1, q_2, \dot{q}_1, \dot{q}_2) = \begin{bmatrix} 0 & 0.2\sin(q_2 - q_1)\dot{q}_2 \\ -0.2\sin(q_2 - q_1)\dot{q}_1 & 0 \end{bmatrix} \quad (50)$$

$$\hat{g}(q_1, q_2) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad (51)$$

The robot is required to move along the constraint surface tracking a time varying position with trajectory $r_1(t) = 0.35 - 0.25 * \sin(0.5 * pi * t)$, while exerting $10N$ force on the constraints.

The simulation results are shown in Figs. 3-5. The time responses of the position tracking error with and without NN are shown in Fig. 3. Fig. 4 shows the force tracking error of the robot with and without NN. The NN outputs are plotted in Fig. 5.

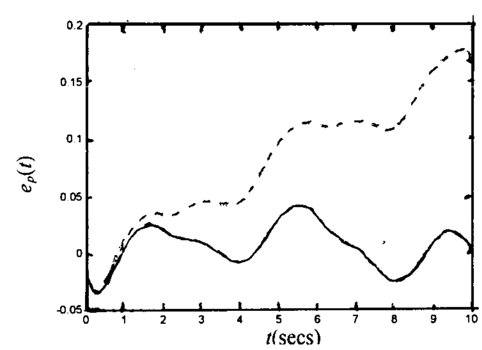

Fig. 3. Plots of position error
(With NN: solid line, without NN: dotted line)

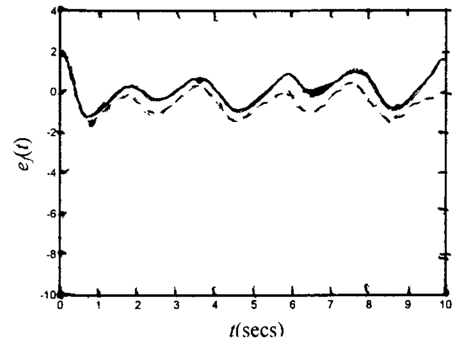

Fig. 4. Plots of force error
(With NN: solid line, without NN: dotted line)

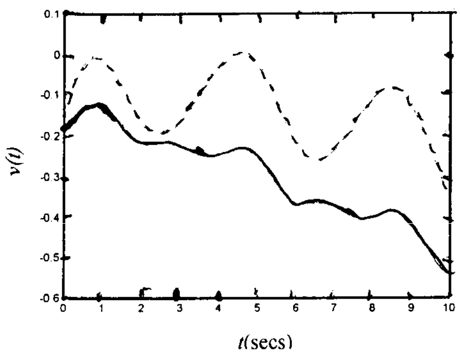

Fig. 5. Outputs of the neural network
(v_1: solid line, v_2: dotted line)

Case 2:

In this case, we choose the nominal dynamic parameter of the robot as

$$\hat{M}(q) = \begin{bmatrix} 1 & 0 \\ 0 & 0.5 \end{bmatrix} \tag{52}$$

$$\hat{C}(q_1, q_2, \dot{q}_1, \dot{q}_2) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{53}$$

$$\hat{g}(q_1, q_2) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{54}$$

The desired position and force trajectory are the same as case 1.

The simulation results are shown in Figs. 6-8. The position and force tracking errors of the constrained robot with and without the NN are plotted in Fig.6 and Fig.7. The outputs of NN are plotted in Fig. 8.

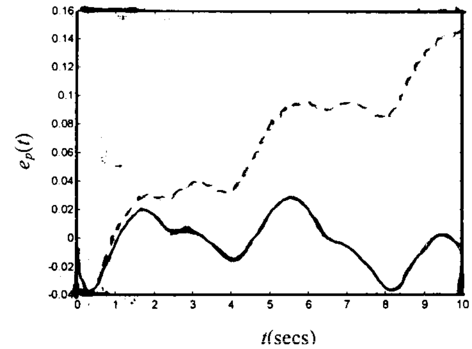

Fig. 6. Plots of position error
(with NN: solid line, without NN: dotted line)

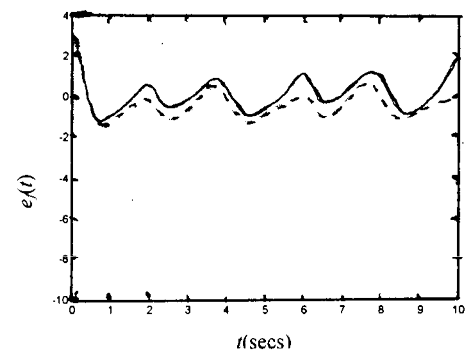

Fig. 7. Plots of force error
(with NN: solid line, without NN: dotted line)

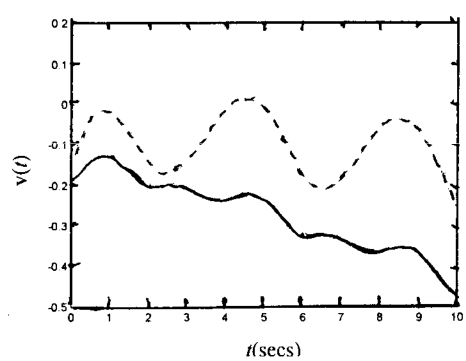

Fig. 8. Outputs of the NN
(v_1: solid line, v_2: dotted line)

From the simulation results above, we can find that in presence of the uncertainties, the suggested on-line adaptive NN controller can efficiently compensate for the parameter uncertainties. The NN controller has good tracking capability, especially in position tracking.

6. Conclusion

An on-line adaptive neural network controller, which uses a two-layer feed-forward neural network, is proposed for the constrained robot in the task space. The NN weights updating law using is derived. The simulation was carried out based on the dynamic model of the 2-dof robot. The simulation results show that the proposed neural network with the on-line updating law can compensate the uncertainties efficiently.

References:

[1] K. J. Salisbury, "Active Stiffness Control of a Manipulator in Cartesian Coordinates," *Proceedings of the 19th IEEE Conference on Decision and Control,* Albuquerque, New Mexico, pp. 95-100, 1980.

[2] M. H. Raibert and J. J. Craig, "Hybrid Position/Force Control of Manipulators," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 102, No. 2, pp.126-133, June 1981.

[3] N. Hogan, "Impedance Control: An Approach to Manipulation: Part I, II, III," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 107, No.1, pp. 1-24, March 1985.

[4] J. K. Mills and A. A. Goldenberg, "Force and Position Control of Manipulators During Constrained Motion Tasks," *IEEE Journal of Robotics and Automation*, Vol.5, No.1, pp. 30-46. February 1989.

[5] N. H. McClamroch and D. W. Wang, "Feedback Stabilization and Tracking of Constrained Robots," *IEEE Transactions on Automation and Control*, Vol. 33, No. 5, pp. 419-426, May 1988.

[6] X. Yun, "Dynamic State Feedback Control of Constrained Robot Manipulator," *Proceedings of the 27th IEEE Conference on Decision and Control*, Austin, Texas, pp. 622-626, Dec. 1988.

[7] Y. H. Kim and F. L. Lewis, "Neural Network Output Feedback Control of Robot Manipulators," *IEEE Transactions on Robotics and Automation*, Vol. 15, No. 2, pp. 301-309, April 1999.

[8] S. S. Ge, C. C. Hang and L.C. Woon, "Adaptive Neural Network Control of Robot Manipulators in Task Space," *IEEE Transactions on Industrial Electronics*, Vol. 44, No.6, pp. 746-752, December 1997.

[9] T. Yabuta and T. Yamada, "Force Control Using Neural Networks," *Advanced Robotics*, Vol. 7, No. 4, pp. 395-408, 1993.

[10] D. Katic and M. Vukobratovic, "Robot Compliance Control Algorithm Based on Neural Network Classification and Learning of Robot-Environment Dynamic Models," *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, Albuquerque, New Mexico, pp. 2632-2637, April 1997.

[11] K. Kiguchi and T. Fukuda, "Robot Manipulator Hybrid Control for an Unknown Environment Using Visco-Elastic Neural Networks," *Proceedings of the 1998 IEEE International Conference on Robotics & Automation,* Leuven, Belgium, pp. 1447-1452, May 1998.

[12] J. J. Craig, "Introduction to Robotics, Mechanics and Control," II Ed. Addsion Wesley, second edition, 1989.

Appendix

Proof of Property 4

$M(q)$ is symmetric and positive definite.
Thus

$$\overline{M}^T(q) = (Q^T(q)M(q)Q(q))^T = Q^T(q)M(q)Q(q) \tag{55}$$

Therefore $\overline{M}(q)$ is symmetric, and positive definite. Similarly it can be shown that $E_2\overline{M}(q)E_2^T$ is symmetric and positive definite.

$$\overline{M}(q) = Q^T(q)M(q)Q(q) \geq \alpha_M \lambda_{\min}(Q^T(q)Q(q))I_n$$
$$= \overline{\alpha}_M I_n \tag{56}$$

where $\lambda_{\min}(Q^T(q)Q(q))$ is the minimum eigenvalue of matrix $Q^T(q)Q(q)$ and

$$\overline{\alpha}_M = \alpha_M \lambda_{\min}(Q^T(q)Q(q)) \tag{57}$$

$$\overline{M}(q) = Q^T(q)M(q)Q(q) \leq \beta_M \lambda_{\max}(Q^T(q)Q(q))I_n \tag{58}$$

where $\lambda_{\max}(Q^T(q)Q(q))$ is the maximum eigenvalue of matrix $Q^T(q)Q(q)$ and

$$\overline{\beta}_M = \beta_M \lambda_{\max}(Q^T(q)Q(q)) \tag{59}$$

Thus $\overline{M}(q)$ is bounded above and below.

Proof of Property 5

$$\dot{\overline{M}} - 2\overline{C}(q,\dot{q})$$
$$= \dot{Q}^T(q)M(q)Q(q) + Q^T(q)\dot{M}(q)Q(q) + Q^T(q)M(q)\dot{Q}(q)$$
$$-2Q^T(q)M(q)\dot{Q}(q) - 2Q^T C(q,\dot{q})Q(q)$$
$$= Q^T(q)(\dot{M}(q) - 2C(q,\dot{q}))Q(q) + [\dot{Q}^T(q)M(q)Q(q)$$
$$- Q^T M(q)\dot{Q}(q)] \tag{60}$$

Since

$$[\dot{Q}^T(q)M(q)Q(q) - Q^T(q)M(q)\dot{Q}(q)]^T$$
$$= -\dot{Q}^T(q)M(q)Q(q) + Q^T(q)M(q)\dot{Q}(q) \tag{61}$$

which is a skew symmetric matrix

From property 1, $\dot{M}(q) - 2C(q,\dot{q})$ is skew symmetric, so the matrix $\dot{\overline{M}} - 2\overline{C}(q,\dot{q})$ is also skew symmetric.

Proof of Theorem1

In the function v given in equation (32), the weights and biases W^1, W^2, b^1 and b^2 are variables of the function v and vary with time. Hence, taking the partial derivative of v with respect to time yields

$$\frac{\partial v}{\partial t} = \frac{\partial W^2}{\partial t} f(s_1) + W^2 \frac{\partial f(s_1)}{\partial t} + \frac{\partial b^2}{\partial t} \tag{62}$$

$$\frac{\partial f(s_1)}{\partial t} = \frac{\partial f(\vartheta)}{\partial \vartheta}\frac{\partial \vartheta}{\partial t} \tag{63}$$

$$\frac{\partial \vartheta}{\partial t} = \frac{\partial W^1}{\partial t} x + \frac{\partial b^1}{\partial t} \tag{64}$$

$$\frac{\partial f(\vartheta)}{\partial \vartheta} = f'(\vartheta)$$

$$
=\begin{bmatrix}
\dfrac{\partial f_1(\vartheta_{1(1)})}{\partial \vartheta_{1(1)}} & \dfrac{\partial f_1(\vartheta_{1(1)})}{\partial \vartheta_{1(2)}} & \cdots & \dfrac{\partial f_1(\vartheta_{1(1)})}{\partial \vartheta_{1(n_H)}} \\
\dfrac{\partial f_2(\vartheta_{1(2)})}{\partial \vartheta_{1(1)}} & \dfrac{\partial f_2(\vartheta_{1(2)})}{\partial \vartheta_{1(2)}} & \cdots & \dfrac{\partial f_2(\vartheta_{1(2)})}{\partial \vartheta_{1(n_H)}} \\
\vdots & \vdots & \ddots & \vdots \\
\dfrac{\partial f_{n_H}(\vartheta_{1(n_H)})}{\partial \vartheta_{1(1)}} & \dfrac{\partial f_{n_H}(\vartheta_{2(n_H)})}{\partial \vartheta_{1(2)}} & \cdots & \dfrac{\partial f_{n_H}(\vartheta_{1(n_H)})}{\partial \vartheta_{1(n_H)}}
\end{bmatrix}
$$

$$
=\begin{bmatrix}
f_1'(\vartheta_{1(1)}) & 0 & \cdots & 0 \\
0 & f_2'(\vartheta_{2(2)}) & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & f_{n_H}'(\vartheta_{n_H(n_H)})
\end{bmatrix} \tag{65}
$$

Substituting equation (63)-(65) into equation (62) yields

$$
\frac{\partial v}{\partial t}=\frac{\partial W^2}{\partial t}f(\vartheta)+\frac{\partial b^2}{\partial t}+W^2 f'(\vartheta)\left(\frac{\partial W^1}{\partial t}x+\frac{\partial b^1}{\partial t}\right) \tag{66}
$$

Let

$$
\partial W^2=\widetilde{W}^2 \qquad \partial b^2=\widetilde{b}^2
$$

$$
\partial W^1=\widetilde{W}^1 \qquad \partial b^1=\widetilde{b}^1
$$

$$
\partial v+\varepsilon_2=v(W^*,b^*,x)-v(W,b,x) \tag{67}
$$

Multiplying both sides of equation (67) by ∂t and expanding the terms yield

$$
\Delta v=\widetilde{W}^2 f(\vartheta)+\widetilde{b}^2+W^2 f'(\vartheta)(\widetilde{W}^1 x+\widetilde{b}^1)
$$

$$
+\varepsilon_2(\vartheta)+\varepsilon_1 \tag{68}
$$

Proof of Theorem 2

The following relationships are found useful for the proposed development:

1. Given a matrix $A=[a_{ij}]$, $A\in R^{p\times p}$, where $tr(\bullet)$ denotes a trace function which sums the diagonal elements of a $p\times p$ matrix, there exists $tr(A^T A)=\sum_{i,j}a_{ij}^2$.
2. For any two $p\times 1$ matrices $A=\lfloor a_1 \quad a_2 \quad \cdots \quad a_p\rfloor^T$ and $B=\lfloor b_1 \quad b_2 \quad \cdots \quad b_p\rfloor^T$, there exists $A^T B=tr(A^T B)=tr(BA^T)$

Substituting equation (31) into equation (37) yields

$$
\dot{V}=\dot{s}^T E_2\overline{M}E_2^T s+s^T E_2\dot{\overline{M}}E_2^T s+s^T E_2\overline{M}E_2^T\dot{s}
$$

$$
+2tr(\dot{\widetilde{W}}^{1^T}\Gamma_1^{-1}\widetilde{W}^1)+2tr(\dot{\widetilde{W}}^{2^T}\Gamma_2^{-1}\widetilde{W}^2)
$$

$$
+2tr(\dot{\widetilde{b}}^{1^T}\Gamma_1^{-1}\widetilde{b}^1)+2tr(\dot{\widetilde{b}}^{2^T}\Gamma_2^{-1}\widetilde{b}^2) \tag{69}
$$

Using the property 5. $\dot{\overline{M}}-2\overline{C}$ is skew-symmetric. Equation (69) can be written as

$$
\dot{V}=2s^T E_2\overline{C}E_2^T s+2s^T E_2\overline{M}E_2^T\dot{s}+2tr(\dot{\widetilde{W}}^{1^T}\Gamma_1^{-1}\widetilde{W}^1)
$$

$$
+2tr(\dot{\widetilde{W}}^{2^T}\Gamma_2^{-1}\widetilde{W}^2)+2tr(\dot{\widetilde{b}}^{1^T}\Gamma_1^{-1}\widetilde{b}^1)+2tr(\dot{\widetilde{b}}^{2^T}\Gamma_2^{-1}\widetilde{b}^2)
$$

$$
=2s^T E_2\overline{C}E_2^T s+2s^T(-E_2Q^T C_1E_2^T s-E_2Q^T QE_2^T s)
$$

$$
+2tr(\dot{\widetilde{W}}^{2^T}\Gamma_2^{-1}\widetilde{W}^2)+2tr(\dot{\widetilde{b}}^{1^T}\Gamma_1^{-1}\widetilde{b}^1)+2tr(\dot{\widetilde{b}}^{2^T}\Gamma_2^{-1}\widetilde{b}^2)
$$

$$
+2tr(\dot{\widetilde{W}}^{1^T}\Gamma_1^{-1}\widetilde{W}^1)+2s^T(-E_2Q^T(\widetilde{W}^2 f(\vartheta)
$$

$$
+\widetilde{b}^2+W^2 f'(\vartheta)(\widetilde{W}^1 x+\widetilde{b}^1)+\varepsilon_2(\varsigma)+\varepsilon))
$$

$$
=-2s^T E_2Q^T QE_2^T s+2tr(\dot{\widetilde{W}}^{1^T}\Gamma_1^{-1}\widetilde{W}^1)+2tr(\dot{\widetilde{W}}^{2^T}\Gamma_2^{-1}\widetilde{W}^2)
$$

$$
+2tr(\dot{\widetilde{b}}^{1^T}\Gamma_1^{-1}\widetilde{b}^1)+2tr(\dot{\widetilde{b}}^{2^T}\Gamma_2^{-1}\widetilde{b}^2)-2s^T E_2Q^T\widetilde{W}^2 f(\vartheta)
$$

$$
-2s^T E_2Q^T W^2 f'(\vartheta)\widetilde{W}^1 x-2s^T E_2Q^T W^2 f'(\vartheta)\widetilde{b}^1
$$

$$
-2s^T E_2Q^T\widetilde{b}^2-2s^T E_2Q^T\varepsilon_2(\varsigma)-2s^T E_2Q^T\varepsilon \tag{70}
$$

Through the relationships 1 and 2 mentioned above, we have the following equalities

$$
s^T E_2Q^T\widetilde{W}^2 f(\vartheta)=tr[f(\vartheta)s^T E_2Q^T\widetilde{W}^2] \tag{71}
$$

$$
s^T E_2Q^T W^2 f'(\vartheta)\widetilde{W}^1 x=tr[xs^T E_2Q^T W^2 f'(\vartheta)\widetilde{W}^1] \tag{72}
$$

$$
s^T E_2Q^T\widetilde{b}^2=tr[s^T E_2Q^T\widetilde{b}^2] \tag{73}
$$

$$
s^T E_2Q^T W^2 f'(\vartheta)\widetilde{b}^1=tr[s^T E_2Q^T W^2 f'(\vartheta)\widetilde{b}^1] \tag{74}
$$

Substitute equation (71)-(74) into (70), we get

$$
\dot{V}=-2s_1^T E_2Q^T QE_2^T s_1+2tr(\dot{\widetilde{W}}^{1^T}\Gamma_1^{-1}\widetilde{W}^1)
$$

$$
+2tr(\dot{\widetilde{W}}^{2^T}\Gamma_2^{-1}\widetilde{W}^2)+2tr(\dot{\widetilde{b}}^{1^T}\Gamma_1^{-1}\widetilde{b}^1)
$$

$$
+2tr(\dot{\widetilde{b}}^{2^T}\Gamma_2^{-1}\widetilde{b}^2)-2tr[f(\vartheta)s^T E_2Q^T\widetilde{W}^2]
$$

$$
-2tr[s^T E_2Q^T\widetilde{b}^2]-2tr[xs^T E_2Q^T W^2 f'(\vartheta)\widetilde{W}^1]
$$

$$
-2tr[s^T E_2Q^T W^2 f'(\vartheta)\widetilde{b}^1]
$$

$$
-2s^T E_2Q^T\varepsilon_2(\varsigma)-2s^T E_2Q^T\varepsilon \tag{75}
$$

If

$$
tr(\dot{\widetilde{W}}^{1^T}\Gamma_1^{-1}\widetilde{W}^1)-tr[xs^T E_2Q^T W^2 f'(\vartheta)\widetilde{W}^1]=0 \tag{76}
$$

$$
tr(\dot{\widetilde{W}}^{2^T}\Gamma_2^{-1}\widetilde{W}^2)-tr[f(\vartheta)s^T E_2Q^T\widetilde{W}^2]=0 \tag{77}
$$

$$
tr(\dot{\widetilde{b}}^{1^T}\Gamma_1^{-1}b^1)-tr[s^T E_2Q^T W^2 f'(\vartheta)\widetilde{b}^1]=0 \tag{78}
$$

$$
tr(\dot{\widetilde{b}}^{2^T}\Gamma_2^{-1}b^2)-tr[s^T E_2Q^T\widetilde{b}^2]=0 \tag{79}
$$

and equivalently, with the adaptive control law equation (76) to (79) applied, there results

$$
\dot{V}=-2s^T E_2Q^T QE_2^T s-2s^T E_2Q^T\varepsilon_2(\varsigma)-2s^T E_2Q^T\varepsilon
$$

$$
\le -2s^T E_2Q^T QE_2^T s<0 \tag{80}
$$

From (36) and (80), it is evident that $\|s\|$ at least converges exponentially to zero, i.e., $e_m\to 0$, and $e_f\to 0$ as $t\to\infty$.

Equating the terms in equations (76) to (80) yields the following adaptation laws for the weights and biases of the NN:

$$
\dot{W}^1=-\dot{\widetilde{W}}^1=-\Gamma_1 f'(\vartheta)W^{2^T}QE_2^T sx^T \tag{81}
$$

$$
\dot{W}^2=-\dot{\widetilde{W}}^2=-\Gamma_2 QE_2^T sf^T(\vartheta) \tag{82}
$$

$$
\dot{b}^1=-\dot{\widetilde{b}}^1=-\Gamma_1 f'(\vartheta)W_2^T QE_2^T s \tag{83}
$$

$$
\dot{b}^2=-\dot{\widetilde{b}}^2=-\Gamma_2 QE_2^T s \tag{84}
$$

where $\widetilde{W}^1=W^{1^*}-W^1$, $\widetilde{W}^2=W^{2^*}-W^2$, $\widetilde{b}^2=b^{2^*}-b^2$, $\widetilde{b}^1=b^{1^*}-b^1$.