View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Missouri University of Science and Technology (Missouri S&T): Scholars' Mine

Missouri University of Science and Technology
Scholars' Mine

Mechanical and Aerospace Engineering Faculty Research & Creative Works | Mechanical and Aerospace Engineering

01 Jan 2001

Robust State Dependent Riccati Equation Based Robot Manipulator Control

Ming Xin

S. N. Balakrishnan
Missouri University of Science and Technology, bala@mst.edu

Zhongwu Huang

Follow this and additional works at: https://scholarsmine.mst.edu/mec_aereng_facwork

Part of the Aerospace Engineering Commons, and the Mechanical Engineering Commons

Recommended Citation

M. Xin et al., "Robust State Dependent Riccati Equation Based Robot Manipulator Control," *Proceedings of the 2001 IEEE International Conference on Control Applications, 2001. (CCA '01)*, Institute of Electrical and Electronics Engineers (IEEE), Jan 2001.
The definitive version is available at https://doi.org/10.1109/CCA.2001.973893

This Article - Conference proceedings is brought to you for free and open access by Scholars' Mine. It has been accepted for inclusion in Mechanical and Aerospace Engineering Faculty Research & Creative Works by an authorized administrator of Scholars' Mine. This work is protected by U. S. Copyright Law. Unauthorized use including reproduction for redistribution requires the permission of the copyright holder. For more information, please contact scholarsmine@mst.edu.

Proceedings of the 2001 IEEE International
Conference on Control Applications
September 5–7, 2001 • México City, México

ROBUST STATE DEPENDENT RICCATI EQUATION BASED ROBOT MANIPULATOR CONTROL

Ming Xin*, S.N. Balakrishnan**, Zhongwu Huang*
xin@umr.edu , bala@umr.edu, zhuang@umr.edu
Department of Mechanical and Aerospace Engineering and Engineering Mechanics
University of Missouri-Rolla, Rolla, MO 65401

Keyword: SDRE, Robot Control, Optimal Control, Robustness, Neural Network

Abstract

We present a new optimal control approach to robust control of robot manipulators in the framework of State Dependent Riccati Equation (SDRE) technique. To treat this highly nonlinear control system, we formulate it as a nonlinear optimal regulator problem. SDRE technique was used to synthesize an optimal controller to this class of robot control problem. We also synthesize a neural network based extra controller to achieve the robustness in the presence of the parameter uncertainties. A typical two-link robot position control problem was studied to show the effectiveness of SDRE approach and robust extra control design to robotic application.

1. Introduction:

Robot manipulators are familiar examples of trajectory-controllable mechanical systems. However , their nonlinear dynamics present a challenging control problem, since traditional linear control approaches do not easily apply. Various control methods have been developed in the literature for rigid robot motion control. Classical PD (Proportional plus Derivative) control was widely used in the robot position control [1][2]. But it is only effective for the highly geared manipulators which thereby strongly reduces the interactive dynamic effects between links. Feedback linearization (inverse dynamics)[3] is another commonly used method in the control of manipulators. Although this approach transform the nonlinear dynamics into linear one so that linear control techniques can be applied, it is difficult to implement in the sense of robustness, mainly because the coordinate transformation is a function of the system parameters and, hence, sensitive to uncertainty. Also, the large differences in magnitude among the parameters, e.g. between joint stiffness and the link inertia, may make the computation of the control ill-conditioned and the performance of the system poor.

In this paper, the state-dependent Riccati equation (SDRE) technique[4], which is an emerging systematic method for solving nonlinear regulator problems, is used to obtain an asymptotically stabilizing feedback solution of the posed nonlinear robot control problem. Section 2 will outline the basic idea of SDRE technique. Section 3 will discuss a planar two-link manipulator control problem using SDRE technique. Because of the unknown load placed on a manipulator and the other parameter uncertainties in the manipulator dynamics, it is important to design a robust control law that will guarantee the performance of the manipulator under these uncertainties. In this paper, we present a new neural network based extra control design to provide the robustness for the SDRE controller. Section 4 will illustrate the theory of the robust extra control design and give the simulation results in the presence of a parameter uncertainty. Conclusion is drawn in Section 5.

*Ph.D Student, ** Professor, contact person

2. Introduction To State Dependent Riccati Equation Method:

State Dependent Riccati Equation (SDRE) method (Cloutier et al.,1996) is a recently emerging nonlinear control system design methodology for direct synthesis of nonlinear feedback controllers. By turning the equations of motion into a linear-like structure, this approach permits the designer to employ linear optimal control methods such as the LQR methodology and the H_∞ design technique for the synthesis of nonlinear control systems.

This approach assumes that the dynamic model of the system

$$\dot{x} = f(x) + g(x)u \tag{1}$$

can be placed in the State Dependent Coefficient form(SDC):

$$\dot{x} = A(x)x + B(x)u \tag{2}$$

The second ingredient of the SDRE design technique is the definition of quadratic performance index in state dependent form:

$$J = \frac{1}{2}\int_{t_0}^{\infty}[x^T Q(x)x + u^T R(x)u]dt \tag{3}$$

0-7803-6733-2/01/$10.00 © 2001 IEEE

The state dependent weighting matrices Q(x) and R(x) can be chosen to realize the desired performance objective. In order to ensure local stability, the matrix Q(x) is required to be positive semidefinite for all x and the matrix R(x) is required to be positive definite for all x.

Next, a state dependent algebraic Riccati equation:

$$A^T(x)P(x)+P(x)A(x)-P(x)B(x)R^{-1}(x)B^T(x)P(x)+Q(x)=0 \quad (4)$$

is formulated and is solved for a positive definite state dependent matrix P(x). The nonlinear state variable feedback control law is then constructed as :

$$u=-R^{-1}(x)B^T(x)P(x)x \quad (5)$$

Cloutier et al.(1996) have shown that this control law is locally stable and optimal with respect to the infinite time performance index. Moreover, Cloutier et al.(1996) have given the conditions that the SDRE control laws can be globally stable and globally optimal.

It can be observed that the crucial part of the control law derivation is the solution of the state dependent Riccati equation. In the general situation, it is difficult to get the closed-form solution. However, this equation can be numerically solved at each sample.

3. Robot Manipulator Control Problem [3]:

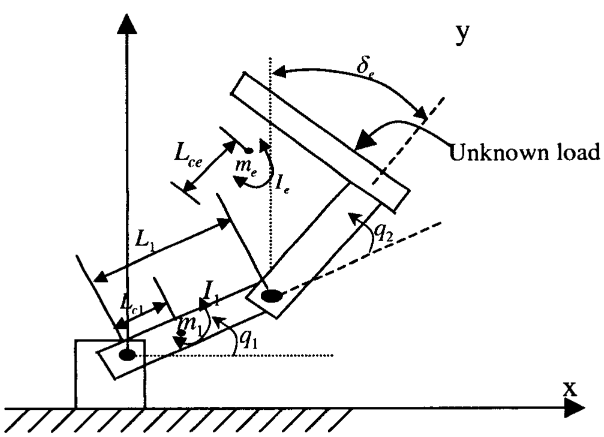

Figure-1 An articulated two-link manipulator

Consider a planar, two-link, articulated manipulator (Figure 9.1), whose position can be described by a 2-vector q of joint angles, and whose actuator inputs consist of a 2-vector τ of torques applied at the manipulator joints. The dynamics of this simple manipulator is strongly nonlinear, and can be written in the general form:

$$I(q)\ddot{q}+C(q,\dot{q})\dot{q}+g(q)=\tau \quad (6)$$

where I(q) is the 2×2 manipulator inertia matrix (which is symmetric positive definite), $C(q,\dot{q})\dot{q}$ is a 2-vector of centripetal and Coriolis torques (with $C(q,\dot{q})$ a 2×2 matrix), and g(q) is the 2-vector gravitational torques. The feedback control problem for such a system is to compute the required actuator inputs to perform desired tasks (e.g., to move it to a given final position).

The task of position control is simply to move it to a given final position, as specified by a constant vector q_d of desired joint angles. Let's assume the desired joint angles as the reference, e.g. 0 angles. Our objective is to drive any initial manipulator position to zero. Now the states in dynamic (6) become the errors between the initial joint angles and the final desired angles. In the mean time, we want to constrain the control effort in an acceptable range. To achieve that, we can easily convert this robot position control problem to an optimal regulator problem which can be attacked using SDRE technique.

Rewrite (6) as the following form:

$$\ddot{q}=-I^{-1}(q)C(q,\dot{q})\dot{q}+I^{-1}(q)(\tau-g(q)) \quad (7)$$

If we choose q and $\dot{q}$ as the states, we can write (7) as the standard SDRE formulation:

$$\dot{x}=f(x)+g(x)u$$

Now consider a concrete example with the two-link manipulator of Figure-1, whose dynamics can be written explicitly as:

$$\begin{bmatrix} I_{11} & I_{12} \\ I_{21} & I_{22} \end{bmatrix}\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix}+\begin{bmatrix} -h\dot{q}_2 & -h(\dot{q}_1+\dot{q}_2) \\ h\dot{q}_1 & 0 \end{bmatrix}\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}=\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \quad (8)$$

corresponding to (6) (here we assume that the manipulator is in the horizontal plane (e.g. $g(q)\equiv 0$)),

where: $I_{11}=a_1+2a_3\cos q_2+2a_4\sin q_2 \quad (9)$

$$I_{12}=I_{21}=a_2+a_3\cos q_2+a_4\sin q_2 \quad (10)$$

$$I_{22}=a_2 \quad (11)$$

$$h=a_3\sin q_2-a_4\cos q_2 \quad (12)$$

with

$$a_1=I_1+m_1l_{c1}^2+I_e+m_el_{ce}^2+m_el_1^2 \quad (13)$$

$$a_2=I_e+m_el_{ce}^2 \quad (14)$$

$$a_3=m_el_1l_{ce}\cos\delta_e \quad (15)$$

$$a_4=m_el_1l_{ce}\sin\delta_e \quad (16)$$

In the simulation, we use

$m_1=1kg \quad L_1=1m \quad m_e=2kg \quad \delta_e=30^\circ$

$I_1=0.12\,kg\bullet m^2 \quad L_{c1}=0.5m$

$I_e=0.25\,kg\cdot m^2 \quad L_{ce}=0.6m$

Using the state space variables:

$$x=[q_1,\dot{q}_1,q_2,\dot{q}_2] \quad (17)$$

control variables: $u=[\tau_1,\tau_2]^T \quad (18)$

the nonlinear regulator problem can be written as

$$J=\frac{1}{2}\int_0^\infty x^TQx+u^TRudt \quad (19)$$

with respect to the state x and control u subject to the nonlinear differential constraints:

$$\dot{x} = A(x)x + B(x)u$$

where

$$A(x) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & A_{11} & 0 & A_{12} \\ 0 & 0 & 0 & 1 \\ 0 & A_{21} & & A_{22} \end{bmatrix}, B(x) = \begin{bmatrix} 0 & 0 \\ B_{11} & B_{12} \\ 0 & 0 \\ B_{21} & B_{22} \end{bmatrix} \quad (20)$$

$$-I^{-1}(x)C(x) = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, I^{-1}(x) = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \quad (21)$$

In the simulation, we choose the weighting matrix Q and R as:

$$Q=diag\{20,0,20,0\}, R = diag\{10^{-5}, 10^{-5}\} \quad (22)$$

The robot, initially at rest at ($q_1 = 0, q_2 = 0$), is commanded a step to ($q_{d1} = 60°, q_{d2} = 90°$). The corresponding transient position errors and control torques are plotted in Figure-2.

The simulation results illustrate that SDRE control gives a satisfactory performance in the sense of the settling time and the overshoot. The maximum control effort is about $2200 N \cdot m$ which can be reduced by increase the weights on the control. Figure-3 shows the result for $R = diag\{10^{-4}, 10^{-4}\}$. Now the maximum control efforts is about $702 N \cdot m$ which is much smaller. But we can see the settling time becomes longer.

4. Robust Design Under the Parameter Uncertainties:

In practice, a robot manipulator is usually controlled to move an unknown object. To control the manipulator, some potential uncertainties must be dealt with such as the weight of the object, the amount of friction and values of other parameters in the manipulator dynamics. Our goal is to design a robust controller that can handle these uncertainties.

In this paper, a neural network based extra control is synthesized with SDRE optimal control to provide robust characteristics in the presence of parameter uncertainties. The robust controller is obtained by

1) Synthesizing an optimal controller (SDRE) for a nominal system with the reference parameters.
2) Generating an extra control as the output of a neural network whose inputs are the error in states between the actual dynamics with parameter uncertainties and the nominal system.

Development of equations to compute the extra control is presented in the following section.

4.1 Problem Reformulation:

Consider a nominal nonlinear system (with optimal control u_{opt} obtained by SDRE techniques)

$$\dot{x}_{1d} = f_1(x_{1d}) + g_1(x_{1d})x_{2d} \quad (23)$$

$$\dot{x}_{2d} = f_2(x_{1d}, x_{2d}) + g_2(x_{1d}, x_{2d})u_{opt} \quad (24)$$

where $x_{1d} \in R^{n_1}$, $x_{2d} \in R^{n_2}$, $u_{opt} \in R^{n_2}$ and $g_1 \in R^{n_1 \times n_2}$, $g_2 \in R^{n_2 \times n_2}$, g_2^{-1} exists.

with unmodeled input uncertainties:

$$\dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 \quad (25)$$

$$\dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)(I + \Delta(x_1, x_2))u_{opt} + d_{21}(x_1, x_2) + d_{22} \quad (26)$$

where d_{22} and $d_{21}(x_1, x_2)$ are uncertainties with d_{22} bounded and $\|d_{22}\| < d_{2N}$, $\Delta(x_1, x_2)$ is bounded and $\|\Delta(x_1, x_2)\| \le 1 - \varepsilon_g$ with $0 < \varepsilon_g \le 1$.

In order to deal with the uncertainty and make the perturbed system behave like Eqs. (23)-(24), extra-control (u_e) is added to Eq. (26):

$$\dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)(I + \Delta(x_1, x_2))(u_{opt} + u_e) + d_{21}(x_1, x_2) + d_{22} \quad (27)$$

This extra control is mainly composed of an online tuned neural network (NN) which will be discussed later. The main property of neural network concerned for control and estimation purposes is the function approximation. Let $f(x)$ be a smooth function from $\Re^n \to \Re^m$. It can be shown that for some sufficient large number of neurons, there exist weights (W) and activation function ($\varphi(x)$) such that

$$f(x) = W^T \varphi(x) + \varepsilon(x) \quad (28)$$

where $\varepsilon(x)$ is the neural network functional approximation error. In fact, for some positive number ε_N, one can find a neural network such that $\|\varepsilon(x)\| \le \varepsilon_N$. For good approximations, $\varphi(x)$ should be a basis such as gaussian, log sigmoid and so on.

4.2 Extra Control design:

The goal is to find an extra control that can handle the uncertainties. To be specific, make x_1 and x_2 bounded around the desired trajectories. Here an online tuned neural network is used for this purpose.

In Eq. (25), $\dot{x}_{1d}$ is subtracted on both sides:

$$\begin{aligned} \dot{e}_1 &= f_1(x_1) + g_1(x_1)x_2 - \dot{x}_{1d} \\ &= f_1 + g_1\alpha_2 - \dot{x}_{1d} + g_1(x_2 - x_{2d} - \alpha_2 + g_1^T(\partial V_1/\partial e_1)) \\ &\quad - g_1 g_1^T(\partial V_1/\partial e_1) + g_1 x_{2d} \end{aligned} \quad (29)$$

where $e_1 = x_1 - x_{1d}$ and V_1 is a Lyapunov function and α_2 is a stablizing control for

$$\dot{e}_1 = f_1(x_1) + g_1(x_1)u - \dot{x}_{1d} \quad (30)$$

For expression simplicity, x_1 is omitted in the expression of f_1 and g_1 in Eq. (29).

$$\dot{V}_1 = (\partial V_1/\partial t) + (\partial V_1/\partial e_1)^T (f_1 + g_1\alpha_2 - \dot{x}_{1d}) + (\partial V_1/\partial e_1)^T g_1 z - (\partial V_1/\partial e_1)^T g_1 g_1^T (\partial V_1/\partial e_1) + (\partial V_1/\partial e_1)^T g_1 x_{2d}$$
$$\le -\gamma_3(|e_1|) + Az - AA^T + Ax_{2d}$$
$$\le -\gamma_3(|e_1|) - (\frac{1}{2}\|A\| - \|z\|)^2 - (\frac{1}{2}\|A\| - \|x_{2d}\|)^2 - \frac{1}{2}\|A\|^2 + \|z\|^2 + \|x_{2d}\|^2 \quad (31)$$

Where $A = (\partial V_1/\partial e_1)^T g_1$, $z = x_2 - x_{2d} - \alpha_2 + g_1^T (\partial V_1/\partial e_1)$ and $(\partial V_1/\partial t) + (\partial V_1/\partial e_1)^T (f_1 + g_1\alpha_2 - \dot{x}_{1d}) \le -\gamma_3(|e_1|)$. From Eq. (31), if z is bounded, so are V_1 and e_1. Consider the derivative of z

$$\dot{z} = \dot{x}_2 - \dot{x}_{2d} - \dot{\alpha}_2 + (g_1^T (\partial V_1/\partial e_1))_t = \dot{x}_2 - \dot{x}_{2d} - \dot{\alpha}_2 + G(x_1, x_{1d}) \quad (32)$$

where

$G(x_1, x_{1d}) = (g_1^T (\partial V_1/\partial e_1))_t = d(g_1^T (\partial V_1/\partial e_1))/dt$

Insert Eq. (27) into Eq. (32) to get

$$\dot{z} = f_2 + g_2(I + \Delta)(u_{opt} + u_e) + d_{21} + d_{22} - \dot{x}_{2d} - \dot{\alpha}_2 + G \quad (33)$$

By choosing

$$u_e = -g_2^{-1}(K_z e_2 + \hat{f}) \quad (34)$$

where $e_2 = x_2 - x_{2d}$ and $\hat{f}$ is the output of a NN with x_1, x_2, x_{1d}, x_{2d}, e_1 and e_2 as inputs. The part of $-K_z e_2$ is a stablizing part that helps the initial convergence.

Insert Eq. (34) into Eq. (33) to get

$$\dot{z} = -g_2(I+\Delta)g_2^{-1}K_z z + f_2 + g_2(I+\Delta)u_{opt} + d_{21}(x_1,x_2) + d_{22} + G - \dot{\alpha}_2 - \dot{x}_{2d} + g_2(I+\Delta)g_2^{-1}K_z(A^T - \alpha_2) - g_2(I+\Delta)g_2^{-1}\hat{f} \quad (35)$$

Assume there exists ideal weights, such that

$$f_2 + g_2(I+\Delta)u_{opt} + d_{21}(x_1,x_2) + g_2(I+\Delta)g_2^{-1}K_z(A^T - \alpha_2) + AA^T z/\|z\|^2 + \alpha_2^T\alpha_2 z/\|z\|^2 + G - \dot{\alpha}_2 - \dot{x}_{2d} = W^T\varphi(net) + \varepsilon(x_1,x_2) \quad (36)$$

with $\|\varepsilon(x_1, x_2)\| < \varepsilon_N$ and $\|W\|_F < W_N$. Where $\|\cdot\|_F$ is Frobenius norm and $\|A\|^2{}_F = tr(A^T A)$. One of its properties is $tr(A^T B) \le \|A\|_F \|B\|_F$. For vectors, Frobenius norm is the same as 2-norm.

By choosing a proper weight-update rule of NN, the u_e in Eq. (34) can make z bounded. Then e_1 and e_2 are bounded. It is called practical stability. The problem is how to find such a weight-update rule. We pick the structure of the neural network for u_e with three layers. The $\hat{f}$ in (35) can be written as a general form:

$$\hat{f} = \hat{W}_3^T \varphi_2(\hat{W}_2^T \varphi_1(\hat{W}_1^T p)) \quad (37)$$

The weighting functions in each layer are updated according to the following rules:

$$\dot{\hat{W}}_1 = -\gamma_1 p[\hat{W}_1^T p + B_1 K_z e_2]^T \quad (38)$$

$$\dot{\hat{W}}_2 = -\gamma_2 [\hat{W}_2^T \hat{\varphi}_1 + B_2 K_z e_2]^T \quad (39)$$

$$\dot{\hat{W}}_3 = -\gamma_3 \hat{\varphi}_2 e_2^T - \gamma_4 \hat{W}_3 \quad (40)$$

where B_1 and B_2 are two constant coefficient matrices. γ_1, γ_2, γ_3, and γ_4 are learning rate;

$\hat{\varphi}_1 = \varphi_1(\hat{W}_1^T p)$ and $\hat{\varphi}_2 = \varphi_2(\hat{W}_2^T \hat{\varphi}_1)$

Here we omit the proof just for brevity of the paper.

4.3. ROBUSTNESS RESULTS:

To illustrate the effectiveness of robust neural extra controller, we assume the mass of unknown load on the second link is the uncertainty parameter, e.g. m_e. In the design of neural network, we adopt three layers and N(12-5-5-2) structure with 12 inputs and 2 outputs. The inputs include 4 reference states $x_r = [q_1, \dot{q}_1, q_2, \dot{q}_2]_r$ driven by SDRE controller, 4 real states $x_c = [q_1, \dot{q}_1, q_2, \dot{q}_2]_c$ with parameter uncertainties driven by SDRE controller without extra control and 4 errors between x_r and x_c. The tangent sigmoid function was chosen as the activation function in each layer. From the equations (34), (37)-(40) we can note that this extra controller design does not need complicated training process of the neural network. The update of the weights follows a fixed dynamic equation. These formulations are based on the Lyapunov function analysis and guarantee the stability of the neural network. This is a big advantage of this design. The parameters we need to adjust are mainly K_z which helps the initial convergence, γ_1 - γ_4 which can be tuned to adjust the learning rates, and B_1, B_2 which combined with K_z can adjust the gain magnitude of the extra control. Here we have to mention that SDRE controller as such possess the robustness to some extent. Extra control plays a bigger role when there exist large unknown load variations. Figure-4 demonstrates SDRE robustness under 50% load uncertainty. We assume the normal value of m_e is 2 as before. • (u) stands for the responses under the parameter uncertainty; • (r) stands for the reference response. We can see SDRE still performs well in terms of the error between the reference and uncertainty system. Figure-5 compares the states and control response with and without extra control under

150% load uncertainty (not only mass but moment of inertia). Note that this type of variations would occur in practice if the same robot is used to handle variable loads. In the plots, and $\bullet(e)$ stands for the response after adding the extra control. Note the total control τ is the sum of SDRE control $\tau(c)$ and extra control $\tau(e)$. The dashed line represents the reference trajectory obtained from the normal parameters. The initial position is the same as before, ($q_{d1} = 60°, q_{d2} = 90°$).

The simulation results illustrate that the extra controls drive the state trajectory very closely to the reference trajectory in terms of both the overshoot and the settling time. They become very effective in the presence of large mass uncertainty. Also we can find from the control plots that the extra control efforts are not so big which is acceptable for implementation.

5. Conclusions:

In this paper, SDRE technique was applied to the robot manipulator control problem. This class of nonlinear control system can be easily formulated in terms of a nonlinear optimal control problem and SDRE approach provides a systematic way to deal with it. When parameter uncertainties is considered, a neural network based extra control was designed to provide the robustness characteristics. A simple two-link manipulator control example was studied and the simulation results demonstrate the effectiveness of the SDRE technique and the robust extra control design. The combination of these two new techniques presents a great potential in the robot control application.

Reference:

[1]. Asda, H. and Slotine, J-J .E, "Robot Analysis and Control", John Wiley& Sons, Inc., New York, 1986

[2] Luh, J. Y. S, Conventional Controller Design for Industrial Robots: a tutorial", IEEE Trans. Syst. Man. Cybern, 13(3), 298-316, May/June 1983.

[3] J-J.E Slotine and Weiping Li, "Applied Nonlinear Control", Prentice Hall, New Jersey, 1991.

[4] Cloutier, J.R., D'Souza, C.N. and Mracek, C.P.(1996), "Nonlinear Regulation and Nonlinear H_{∞} Control Via the State-Dependent Riccati Equation Technique", *Proc. 1st Interna. Conf. On Nonlinear Problems in Aviation and Aerospace, Daytona Beach,FL, May 1996.*

[5] K.J.Hunt, "Neural Networks for Controller Systems, A Survey", Automatica,Vol.28, No.6, 1992,pp. 1083-1112

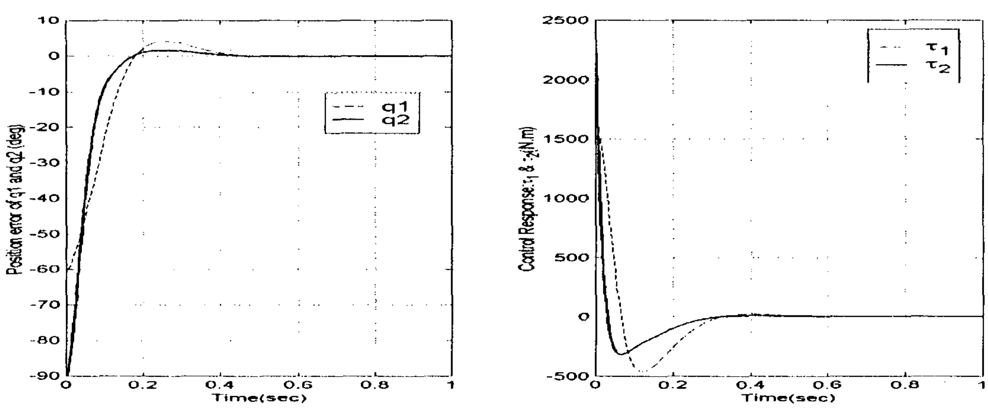

Figure-2

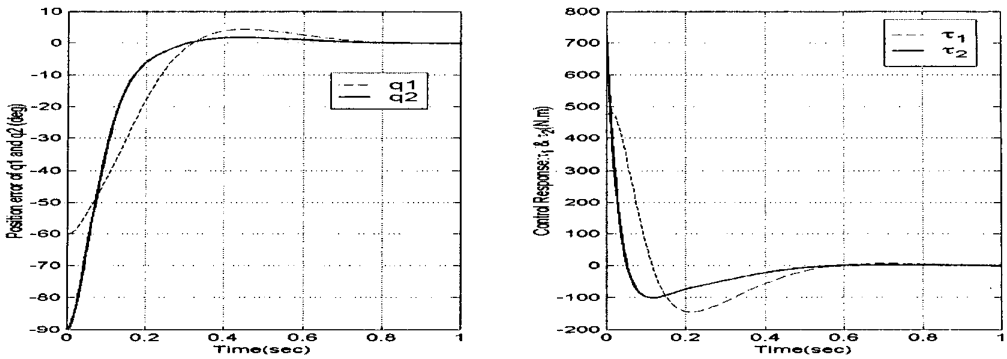

Figure-3

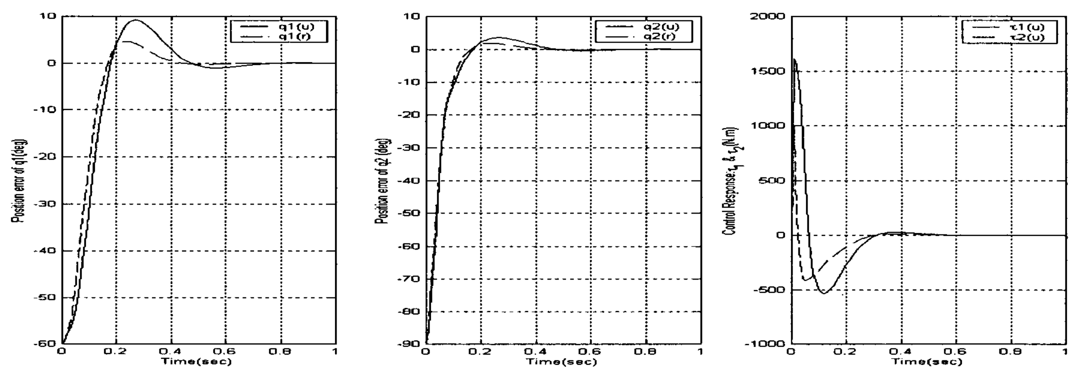

Figure-4: 50% load uncertainty

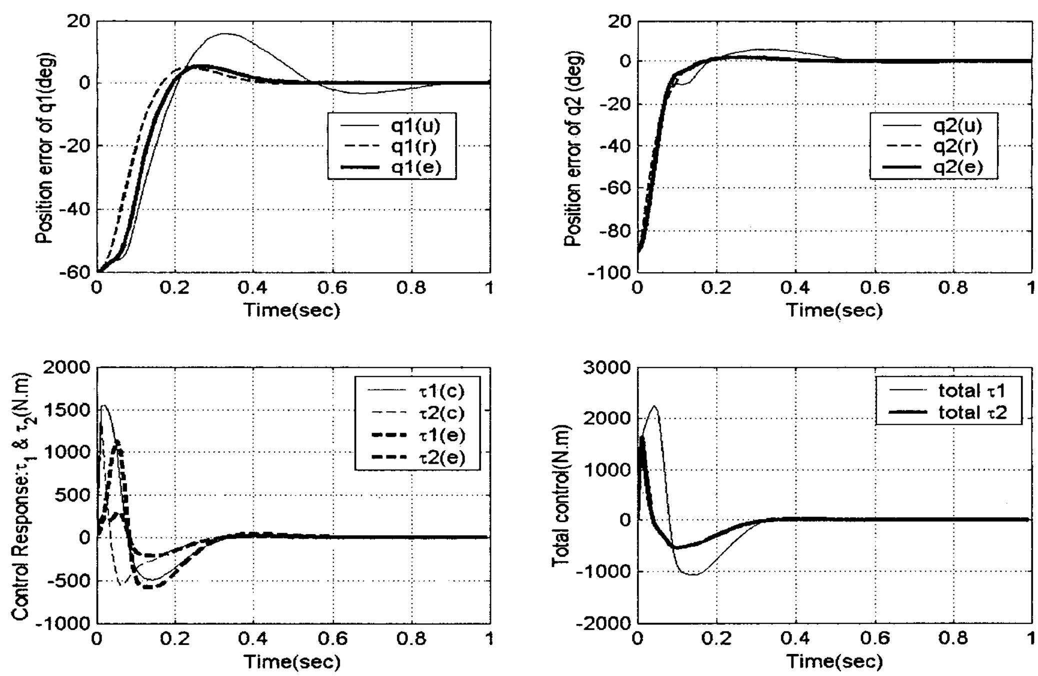

Figure-5: 150% load uncertainty