Optimal feedforward and recurrent neural net controllers, constrained by linear controller design*

Johan A.K. Suykens
Bart L.R. De Moor†
Joos Vandewalle

ESAT-SISTA report 1993-43

Katholieke Universiteit Leuven
Department of Electrical Engineering, ESAT-SISTA
Kardinaal Mercierlaan 94
B-3001 Leuven (Heverlee), Belgium
tel: 32/16/22 09 31 fax: 32/16/22 18 55
e-mail: Johan.Suykens@esat.kuleuven.ac.be

submitted to

ASIA - Pacific Engineering Journal
Special issue on Neural Control

August 20, 1993

*This research work was carried out at the ESAT laboratory of the Katholieke Universiteit Leuven, in the framework of a Concerted Action Project of the Flemish Community, entitled *Applicable Neural Networks*. The scientific responsibility is assumed by its authors.

†Research Associate of the Belgian National Fund for Scientific Research

Abstract

A method for designing optimal neural controllers, either parametrized by feedforward or recurrent neural networks is discussed. The optimal control problem considered here is to bring the system from a given initial state to a desired final state, with the additional requirement that the closed loop system at the final state has to correspond to a specific linear controller design around that point. Such type of problems occur e.g. in plants where a switching between several operating points is needed and in systems where certain startup control is currently done manually because of limitations of the classical control.

The design procedure consists basically of two steps. First a linear controller is derived around the target point resulting into a dynamic or static output or state feedback law. Depending upon this choice a recurrent or feedforward neural net controller is selected that is designed such that the linearized neural control system is identified with the linearized system under the linear feedback law. Secondly a nonlinear optimization problem is formulated in the interconnection weights, constrained by the linear controller design result.

The method is illustrated on a highly complex control task of swinging up a double inverted pendulum with local stabilization at the endpoint.

Key Words: Feedforward and recurrent neural networks, optimal control, nonlinear optimization, static and dynamic output feedback, double inverted pendulum.

1 Introduction

Since the early results on neural control systems [2], a large variety of methods have been developed in this area based on several mathematical theories such as e.g. adaptive control, optimal control, dynamic programming, feedback linearization [4][11][15][16][17][20][21] and others and it is generally expected that neural networks can offer new and better control schemes because of their ability of learning and their massively parallel and nonlinear nature. However this nonlinear nature can be considered both as an advantage and a disadvantage because e.g. stability of closed loop systems is in most cases hard to proof and many optimization schemes are converging to local optima instead of to the global optimum. Nevertheless an advantage is that neural controllers can do 'potentially more' than linear controllers.
The latter will be illustrated here for systems where a transition from a certain initial state to a final state is required with the additional specification that the closed loop system at the endpoint corresponds to a suitable linear controller design. Such problems occur e.g. in plants working around several operating points and in systems where certain startup control is currently done manually because of limitations of the classical control. A neural control approach to this problem was first introduced in [22]. The neural controller can be designed then such that it behaves as a linear controller (ranging from classical control techniques such as PID, pole placement to modern linear controller design as H_2, H_∞, μ-synthesis etc.[1][7][13]) around the target equilibrium point but with the additional capability of realizing a transition between a point outside the region where this linearization is valid to the target equilibrium point. In fact the neural controller has extra degrees of freedom which are exploited to formulate a nonlinear optimization problem in the interconnection weights that expresses the transition between two points, contrained by the linear controller design results. Hence results from well developed linear control theory can be included in the neural controller. The method holds for dynamic and static output and state feedback and is applicable in a continuous time as well as a discrete time framework. In that sense the results proposed here are more general than the optimal neural control schemes discussed in [20][24], which assume full state information of the plant. Learning in the presented context is off-line and corresponds to the specific algorithm that is chosen to solve the nonlinear optimization problem in the interconnection weights.

This paper is organized as follows: in Section 2 neural controllers are proposed that are either parametrized by feedforward or recurrent neural networks. Section 3 explains how results from linear control theory can be incorporated into the neural controller. Section 4 treats the optimal control problem, constrained by the linear feedback law. The method is illustrated on a higly complex control task for swinging up and locally stabilize a double inverted pendulum system in Section 5.

2 Static and dynamic output feedback neural controllers

2.1 Parametrization by feedforward or recurrent neural nets

Suppose a nonlinear state space model for a plant P is available

$$\begin{cases} \dot{x} &= f(x) + g(x)\,u \\ y &= h(x) \end{cases} \tag{1}$$

with state vector $x \in \mathbb{R}^n$, input $u \in \mathbb{R}$, output vector $y \in \mathbb{R}^m$ and f, g, h vector fields defined on $\mathbb{R}^n$ ($f : \mathbb{R}^n \mapsto \mathbb{R}^n$, $g : \mathbb{R}^n \mapsto \mathbb{R}^n$, $h : \mathbb{R}^n \mapsto \mathbb{R}^m$). Furthermore we suppose that $f(0) = 0$, $h(0) = 0$. If this is not the case a change of coordinates can be applied. The techniques that will be discussed in the text generalize to the multi-input and discrete time case too, but will not be treated in the text.
For the system (1) we consider both the case of parametrized static output feedback

$$u = \mu(y;\theta) \tag{2}$$

and dynamic output feedback (Fig.1)

$$\begin{cases} \dot{z} &= \nu(z,y;\theta_1) \\ u &= \rho(z,y;\theta_2) \end{cases} \tag{3}$$

where $\theta \in \mathbb{R}^p$, $\theta_1 \in \mathbb{R}^{p_1}$, $\theta_2 \in \mathbb{R}^{p_2}$ are parameter vectors, $z \in \mathbb{R}^{n_z}$ is the state of the controller and $\mu : \mathbb{R}^m \mapsto \mathbb{R}$, $\nu : \mathbb{R}^{n_z} \times \mathbb{R}^m \mapsto \mathbb{R}^{n_z}$, $\rho : \mathbb{R}^{n_z} \times \mathbb{R}^m \mapsto \mathbb{R}$ are continuous nonlinear mappings. Because any continuous nonlinear function can be approximated arbitrarily well over a compact interval by a multilayer neural network with one or more hidden layers [5][8][10] the following parametrizations by feedforward and recurrent neural networks with one hidden layer can be made for (2)(3):

$$u = \alpha \tanh(w^t \tanh(Vy)) \tag{4}$$

and

$$\begin{cases} \dot{z} &= W_1 \tanh(V_{11}z + V_{12}y) \\ u &= \alpha \tanh(w_2^t \tanh(V_{21}z + V_{22}y)) \end{cases} \tag{5}$$

with interconnection weights $w \in \mathbb{R}^{n_h}$, $V \in \mathbb{R}^{n_h \times m}$ for (4) and $W_1 \in \mathbb{R}^{n_z \times n_{h_1}}$, $V_{11} \in \mathbb{R}^{n_{h_1} \times n_z}$, $V_{12} \in \mathbb{R}^{n_{h_1} \times m}$, $V_{21} \in \mathbb{R}^{n_{h_2} \times n_z}$, $V_{22} \in \mathbb{R}^{n_{h_2} \times m}$ and vector $w_2 \in \mathbb{R}^{n_{h_2}}$ for (5) and $\alpha = |u|_{max}$[1]. n_h, n_{h_1}, n_{h_2} refer to the number of hidden neurons. The parameter vector θ in (2) and θ_1 and θ_2 in (3) are respectively a concatenation of the elements in w,V,α and W_1, V_{11}, V_{12}, α, w_2, V_{21}, V_{22}. Remark that the amplitude of the control signal is automatically limited to α by the choice of these parametrizations (4)(5). This is a potential advantage of the proposed

[1] As activation function we take the hyperbolic tangent function ($\tanh(x) = \frac{1-\exp(-2x)}{1+\exp(-2x)}$), which is applied elementwise to a vector or a matrix.

method with respect to other nonlinear control techniques such as feedback linearization and variable structure control [21]. Further notice that all neural net parametrizations are chosen with zero bias terms. The reason for this follows from linearization of the closed loop system as will be explained in Section 3.

2.2 Control task and neural controller design procedure

The control task considered for the plant P is to bring the state x from $x(0)$ to a target equilibrium point x_{eq} with the additional requirement that a specific linear controller design holds around the target equilibrium point. Derivation of the neural controllers (4)(5) works then as follows

1. Linearize the plant around the target equilibrium point x_{eq} and design a suitable linear controller for local control around this endpoint. This controller may be either dynamic or static and applying output or state feedback depending on the particular system to be controlled. The nonlinear control law (4)(5) becomes then the linearized controller around x_{eq}. This linearization will be discussed in Section 3.

2. From linearization of the closed loop system under the nonlinear control law (4)(5) it will follow immediately that the linear controller design of Step 1 will impose some constraints on the choice of weights θ, θ_1 and θ_2 but a degree of freedom will remain in this choice which will be exploited to let the parameters (interconnection weights) serve to satisfy other performance criteria such as specifications on the transition from $x(0)$ to x_{eq}. This results into nonlinear optimization problems in the interconnection weights constrained by the linear controller design results of Step 1 and will be discussed in Section 4.

3 Incorporating linear controller design results

We will study here the first step in the design procedure which is the linearization of the plant model (1) under a static (4) or dynamic (5) neural output feedback law around the target equilibrium point.

3.1 Static output feedback by feedforward neural networks

Applying the static output feedback law (4) we obtain the closed loop system

$$\dot{x} = f(x) + g(x)\,\alpha\,\tanh(w^t\,\tanh(Vh(x))) \tag{6}$$

and we suppose that $x_{eq} = 0$ ($f(0) = 0$, $h(0) = 0$). In order to derive conditions for local stability around $x = 0$ the Jacobian matrix J is calculated and evaluated at $x = 0$:

$$\begin{array}{rcl} J(x) & = & f_x + g_x\alpha\tanh(w^t\,\tanh(Vh(x))) + g(x)\,\alpha\gamma_1 w^t\Gamma_2 V h_x \\ J(0) & = & f_0 + g(0)\,\alpha w^t V h_0 \end{array} \tag{7}$$

where f_x, g_x and h_x denote the matrices with partial derivatives $[\frac{\partial f_i}{\partial x_j}]$, $[\frac{\partial g_i}{\partial x_j}]$ $(i,j = 1,...,n)$ and $[\frac{\partial h_i}{\partial x_j}]$ $(i = 1,...,m,\ j = 1,...,n)$. The matrices f_0 and h_0 represent the matrices f_x and h_x evaluated at 0. The scalar γ_1 and the diagonal matrix $\Gamma_2 = diag([\gamma_{2_1},...,\gamma_{2_{n_h}}])$ are given by

$$\begin{array}{rcll} \gamma_1 & = & 1 - \tanh^2(w^t \tanh(Vh(x))) & \\ \gamma_{2_i} & = & 1 - \tanh^2(v_i^t h(x)) & i = 1,...,n_h \end{array} \tag{8}$$

where v_i^t denotes the i-th row of V. Local stability around $x = 0$ is guaranteed if the eigenvalues λ_i of $J(0)$ satisfy the condition

$$Re\{\lambda_i(f_0 + g(0)\,\alpha w^t V\, h_0)\} < 0 \qquad i = 1,...,n \tag{9}$$

On the other hand a linear static output feedback $u = -k^t y$ can be considered for the linearized system around $x = 0$

$$\left\{ \begin{array}{rcl} \dot{x} & = & f_0\, x + g(0)\, u \\ y & = & h_0\, x \end{array} \right. \tag{10}$$

leading to the closed loop system

$$\dot{x} = (f_0 - g(0)\, k^t\, h_0)\, x \tag{11}$$

which is guaranteed to be stable if

$$Re\{\lambda_i(f_0 - g(0)\, k^t\, h_0)\} < 0 \qquad i = 1,...,n \tag{12}$$

Comparing (9) and (12) we can conclude that all possible choices of w and V satisfying the relation

$$k_{stab}^t = -\alpha\, w^t V \tag{13}$$

will locally stabilize (6) around the target equilibrium point, where k_{stab} corresponds to a stabilizing controller for (10).

Remarks:

1. From relation (13) it is clear that the neural controller (4) can do potentially more than the linear controller because w and V are not completely determined by (13). This degree of freedom can be used to let the interconnection weights satisfy additional specifications such as a transition from an initial state (outside the region where the linearization is valid) to the target equilibrium point. With respect to this it is important to note that (13) only guarantees existence of a locally stabilizing controller but gives no information on the size of the region where the linearization is valid.

2. In the case of full state feedback we have $h_0 = I_n$ if $y = x$. In order to find a linear optimal and stabilizing controller one can apply e.g. an LQR [7] or H_∞ control (state

feedback case) technique [13]. In the case of LQR one seeks to minimize the cost function

$$C_{lqr} = \int_0^\infty (x^t Q x + u^t R u)\, dt \tag{14}$$

where Q and R are given positive definite symmetric matrices. The solution to this problem is given by

$$k^t_{stab} = k^t_{lqr} = R^{-1} g(0)^t P \tag{15}$$

where P is the stabilizing solution to the matrix algebraic Riccati equation

$$O = P f_0 + f_0^t P - P g(0) R^{-1} g(0)^t P + Q \tag{16}$$

3. In the case of one output neuron and no hidden layer one obtains the relation

$$k^t_{stab} = -\alpha\, w^t \tag{17}$$

which means that for a given α the weight vector w is completely determined by this relation and linear static state feedback can be placed at the same level as static state feedback consisting of one neuron.
In the general case of a neural net consisting of q hidden layers one has the condition

$$k^t_{stab} = -\alpha\, w^t V_1 V_2 ... V_q \tag{18}$$

3.2 Dynamic output feedback by recurrent neural networks

Applying the dynamic output feedback law (5) we obtain the closed loop system

$$\left\{ \begin{array}{rcl} \dot{z} &=& \phi(z,x) \\ \dot{x} &=& \psi(z,x) \end{array} \right. \tag{19}$$

where

$$\begin{array}{rcl} \phi(z,x) &=& W_1 \tanh(V_{11} z + V_{12} h(x)) \\ \psi(z,x) &=& f(x) + g(x)\, \alpha \tanh(w_2^t \tanh(V_{21} z + V_{22} h(x))) \end{array} \tag{20}$$

Linearization of the closed loop system around the target equilibrium point $x = 0$, $z = 0$ yields the Jacobian matrix J

$$J(z,x) = \left[\begin{array}{cc} \frac{\partial \phi}{\partial z} & \frac{\partial \phi}{\partial x} \\ \frac{\partial \psi}{\partial z} & \frac{\partial \psi}{\partial x} \end{array} \right] = \left[\begin{array}{cc} W_1 \Gamma_1 V_{11} & W_1 \Gamma_1 V_{12} h_x \\ g(x) \alpha \gamma_2 w_2^t \Gamma_3 V_{21} & f_x + g_x \alpha \tanh(.) + g(x) \alpha \gamma_2 w_2^t \Gamma_3 V_{22} h_x \end{array} \right] \tag{21}$$

with diagonal matrices $\Gamma_1 = diag([\gamma_{1_1}, ..., \gamma_{1_{n_{h_1}}}])$, $\Gamma_3 = diag([\gamma_{3_1}, ..., \gamma_{3_{n_{h_2}}}])$ and γ_2 equal to

$$\begin{array}{rcll} \gamma_{1_i} &=& 1 - \tanh^2(v^t_{11_i} z + v^t_{12_i} h(x)) & i = 1, ..., n_{h_1} \\ \gamma_2 &=& 1 - \tanh^2(w_2^t \tanh(V_{21} z + V_{22} h(x)) & \\ \gamma_{3_i} &=& 1 - \tanh^2(v^t_{21_i} z + v^t_{22_i} h(x)) & i = 1, ..., n_{h_2} \end{array} \tag{22}$$

and $v_{11_i}^t$,$v_{12_i}^t$,$v_{21_i}^t$, $v_{22_i}^t$ denoting respectively the i-th row of V_{11},V_{12}, V_{21},V_{22}. The condition

$$Re\{\lambda_i(J(0,0))\} < 0 \qquad i = 1, ..., n + n_z \tag{23}$$

with

$$J(0,0) = \begin{bmatrix} W_1 V_{11} & W_1 V_{12}\, h_0 \\ g(0)\alpha\, w_2^t V_{21} & f_0 + g(0)\alpha\, w_2^t V_{22}\, h_0 \end{bmatrix} \tag{24}$$

guarantees then local stability around $x = 0$, $z = 0$.
On the other hand applying the linear dynamic output feedback law

$$\begin{cases} \dot{z} &= E\,z + F\,y \\ u &= G\,z + H\,y \end{cases} \tag{25}$$

on the linearized system around $x = 0$ (10) gives the closed loop system

$$\begin{cases} \dot{z} &= E\,z + FC\,x \\ \dot{x} &= BG\,z + (A + BHC)\,x \end{cases} \tag{26}$$

which is stable if

$$Re\{\lambda_i(\begin{bmatrix} E & FC \\ BG & A + BHC \end{bmatrix})\} < 0 \qquad i = 1, ..., n + n_z \tag{27}$$

Comparing (23) with (27) yields the following condition for local stability around the target equilibrium point

$$\begin{bmatrix} E_{stab} & F_{stab} \\ G_{stab} & H_{stab} \end{bmatrix} = \begin{bmatrix} W_1 V_{11} & W_1 V_{12} \\ \alpha\, w_2^t V_{21} & \alpha\, w_2^t V_{22} \end{bmatrix} \tag{28}$$

where E_{stab}, F_{stab}, G_{stab} and H_{stab} correspond to a linear stabilizing controller satisfying (27).

Remarks:

1. As in (13) there is again freedom in the choice of the interconnection weights which means that additional specifications can be imposed on the neural controller. This will be explained in Section 4.

2. All available linear controller design techniques may be applied in order to determine E_{stab}, F_{stab}, G_{stab} and H_{stab} including classical control techniques such as PID and pole placement and modern linear controller design like e.g. H_2, H_∞, μ-synthesis etc. [1][7][13]).

4 Additional specification of an optimal control problem

The degree of freedom in the choice of the interconnection weights in (13)(28) will be used now to impose a transition from a given initial state $x(0)$ to the target equilibrium point $x_{eq} = 0$, around which a linear controller was designed. An optimal control problem is formulated in the static and dynamic feedback case with the linear controller design as additional constraint and numerical methods are discussed to solve the problem.

4.1 Optimal control problem

In the *static* feedback case we seek to minimize the cost function C

$$\min_{\alpha,w,V} C(\alpha, w, V, f, g, h, x(0), \eta, \zeta, T) \tag{29}$$

with

$$C = \eta(x(T)) + \int_0^T \zeta(x(t))\, dt$$

under the constraints

1. Local stability at $x = 0$ (13):

$$k_{stab}^t = -\alpha\, w^t V \tag{30}$$

2. Closed loop system dynamics (6):

$$\dot{x} = f(x) + g(x)\, \alpha \tanh(w^t \tanh(Vh(x))) \tag{31}$$

In the cost function C one can consider e.g. terminal control ($\zeta = 0$) or quadratic control ($\zeta(x(t)) = x(t)^t x(t)$) [3] and $\eta(x(T)) = \|x(T)\|$ with T the time horizon.
In the *dynamic* feedback case the following cost function can be minimized with target equilibrium point $x = 0$, $z = 0$:

$$\min_{\alpha,W_1,w_2,V_{11},V_{12},V_{21},V_{22}} C(\alpha, W_1, w_2, V_{11}, V_{12}, V_{21}, V_{22}, f, g, h, x(0), z(0), \eta, \zeta, T) \tag{32}$$

with

$$C = \eta(x(T), z(T)) + \int_0^T \zeta(x(t), z(t))\, dt$$

and constraints

1. Local stability at $x = 0$, $z = 0$ (28):

$$\begin{array}{rcl} [E_{stab}\ F_{stab}] & = & W_1[V_{11}\ V_{12}] \\ [G_{stab}\ H_{stab}] & = & \alpha\, w_2^t[V_{21}\ V_{22}] \end{array} \tag{33}$$

2. Closed loop system dynamics (19)(20):

$$\begin{cases} \dot{z} &= W_1 \tanh(V_{11} z + V_{12} h(x)) \\ \dot{x} &= f(x) + g(x)\, \alpha \tanh(w_2^t \tanh(V_{21} z + V_{22} h(x)) \end{cases} \tag{34}$$

The control problems (29)-(31)(32)-(34) are suboptimal in the sense that a specific parametrization (number of hidden neurons, number of hidden layers) is chosen. The new aspect with respect to classical optimal control theory [3] and existing neural optimal control schemes [20][24] is the introduction of the local stability constraints (30)(33). Given a finite time horizon T, omitting these constraints will also result in a neural controller that brings the state from $x(0)$ (suppose outside the region where the linearization is valid) to $x_{eq} = 0$ but may result in a controller that is not locally stabilizing around 0 and hence will not stay at that point under perturbations (e.g. disturbance, modelling error, ...).
So after solving 29)-(31) or (32)-(34) the neural controller learned in fact to control the system starting from the given initial state. An important question is then how this controller can generalize with respect to another initial state. In [22] it was observed that for a specific example of swinging up an inverted pendulum system the generalization ability was much larger for a dynamic controller (using recurrent neural networks) than for a static controller. Furthermore it is also theoretically possible to define a cost function over several initial states $x(0)$.
It is shown now how the optimal control problems can be simplied. In the case of square and full rank matrices V, $[V_{11}\ V_{12}]$, $[V_{21}\ V_{22}]$, one can eliminate the variables w in (31) and W_1, w_2 in (34) because $w^t = -\frac{1}{\alpha} k_{stab}^t V^{-1}$ and $W_1 = [E_{stab}\ F_{stab}].[V_{11}\ V_{12}]^{-1}$, $w_2^t = \frac{1}{\alpha}[G_{stab}\ H_{stab}].[V_{21}\ V_{22}]^{-1}$ resulting in the *static* feedback case into the optimal control problem

$$\min_{\alpha, V} C(\alpha, V, f, g, h, x(0), \eta, \zeta, T) \tag{35}$$

with constraint

$$\dot{x} = f(x) + g(x)\, \alpha \tanh(-\frac{1}{\alpha} k_{stab}^t V^{-1} \tanh(V h(x))) \tag{36}$$

and in the *dynamic* feedback case

$$\min_{\alpha, V_{11}, V_{12}, V_{21}, V_{22}} C(\alpha, V_{11}, V_{12}, V_{21}, V_{22}, f, g, h, x(0), z(0), \eta, \zeta, T) \tag{37}$$

with constraint

$$\begin{cases} \dot{z} &= [E_{stab}\ F_{stab}]\,[V_{11}\ V_{12}]^{-1} \tanh(V_{11} z + V_{12} h(x)) \\ \dot{x} &= f(x) + g(x)\, \alpha \tanh(\frac{1}{\alpha}[G_{stab}\ H_{stab}]\,[V_{21}\ V_{22}]^{-1} \tanh(V_{21} z + V_{22} h(x))) \end{cases} \tag{38}$$

For an elimination of variables for non square matrices V, $[V_{11}\ V_{12}]$, $[V_{21}\ V_{22}]$ we refer to [22]

4.2 Numerical solution

A numerical solution to the problems (35)-(36) (37)-(38) can be obtained by simulation of the closed loop system (36)(38) over a given time horizon T, applying an appropriate numerical integration rule and calculation of the cost function C from this simulation result. A nonlinear optimization problem can then be formulated in the interconnection weights: in the *static* feedback case

$$\min_{V} \; C(V, f, g, h, x(0), \eta, \zeta, T) \qquad \|V\| < \beta \tag{39}$$

and in the *dynamic* feedback case

$$\min_{V_{11}, V_{12}, V_{21}, V_{22}} \; C(V_{11}, V_{12}, V_{21}, V_{22}, f, g, h, x(0), z(0), \eta, \zeta, T) \tag{40}$$

with constraints

$$\begin{array}{rcl} \|[V_{11} \; V_{12}]\| & < & \beta_1 \\ \|[V_{21} \; V_{22}]\| & < & \beta_2 \end{array} \tag{41}$$

where $\alpha = |u|_{max}$ is chosen a priori and β, β_1, β_2 are upperbounds to ensure a sufficiently large region for which the linearization at the endpoint is valid.
These nonlinear optimization problems may have many local optima in general. One can either use local optimization techniques (such as Sequential Quadratic Programming [9][14][18]) by trying several starting points or global optimization techniques as e.g. genetic algorithms [23]. The choice of the number of hidden neurons n_h, n_{h_1}, n_{h_2} is basically a trial and error procedure, but it will will be illustrated in Section 5 that relatively small scale neural networks are capable of performing higly complex control tasks such as e.g. swinging up a double inverted pendulum system.

5 Example: swinging up a double inverted pendulum

The presented theory will be illustrated here on an example of swinging up and locally stabilize a *double* inverted pendulum system by a feedforward neural network. For an example of swinging up an inverted pendulum system either using dynamic output or static state feedback we refer to [22].
The standard state space form for a double inverted pendulum in robotic terminology is (see [12] and Fig.2):

$$M(x_p, \theta_1, \theta_2).\ddot{d} + V(x_p, \theta_1, \theta_2, \dot{x_p}, \dot{\theta_1}, \dot{\theta_2}) + G(x_p, \theta_1, \theta_2) = F_e \tag{42}$$

with

$$M = \begin{bmatrix} m_c + m_1 + m_2 & (\frac{m_1}{2} + m_2) l_1 cos\theta_1 & \frac{m_2}{2} l_2 cos\theta_2 \\ (\frac{m_1}{2} + m_2) l_1 cos\theta_1 & (\frac{m_1}{3} + m_2) l_1^2 & \frac{m_2}{2} l_1 l_2 cos(\theta_2 - \theta_1) \\ \frac{m_2}{2} l_2 cos\theta_2 & \frac{m_2}{2} l_1 l_2 cos(\theta_2 - \theta_1) & \frac{m_2}{3} l_2^2 \end{bmatrix} \tag{43}$$

and

$$V = \begin{bmatrix} (\frac{m_1}{2} + m_2) l_1 sin\theta_1 . \dot{\theta_1}^2 \\ \frac{m_2}{2} l_1 l_2 sin(\theta_2 - \theta_1) . \theta_2 \\ -\frac{m_2}{2} l_1 l_2 sin(\theta_2 - \theta_1) . \dot{\theta_1}^2 \end{bmatrix}, G = \begin{bmatrix} 0 \\ (\frac{m_1}{2} + m_2) l_1 g sin\theta_1 \\ \frac{m_2}{2} l_2 g sin\theta_2 \end{bmatrix}$$

$$F_e = \begin{bmatrix} F \\ 0 \\ 0 \end{bmatrix}, d = \begin{bmatrix} \ddot{x} \\ \ddot{\theta_1} \\ \ddot{\theta_2} \end{bmatrix}, x^t = [x_p \ \dot{x_p} \ \theta_1 \ \dot{\theta_1} \ \theta_2 \ \dot{\theta_2}]$$

the system consists of two ideal links (uniform and negligible cross section) with length l_1, l_2 and masses m_1, m_2, each joint a frictionless revolute hinge. The mass of the cart is m_c. The variables x_p, $\theta_i (i = 1, 2)$ are respectively equal to the distance of the cart from some fixed reference point and the angle of the i-th link against a fixed vertical, measured in a clockwise direction. The state vector of the system related to a model in the form (1) is $x \in \mathbb{R}^n$ with $n = 6$. The force applied to the cart is equal to F. The positive definite matrix M is known as the mass or inertia matrix, the vector V contains centrifugal and Coriolis force terms, G holds gravitational terms and F_e is an external force vector.
The control task that we consider here is the transition from the initial state $x(0) = [0\,0\,\pi\,0\,\pi\,0]^t$ to $x_{eq} = [0\,0\,0\,0\,0\,0]^t$ (swinging up problem) with local stabilization at that endpoint. We take a full static state feedback controller parametrized by a feedforward neural network consisting of 6 hidden neurons. The control signal u, which is the force F applied to the cart is then

$$u = \alpha . tanh(w^t . tanh(V.x)) \tag{44}$$

with $\alpha = |u|_{max}$ the maximal force applied to the cart.
The neural controller design consists of two steps

1. **Linear controller design:**
   First a linear controller design is done around the target equilibrium point using LQR with $Q = I_6$ and $R = 1$ in (14) for the system parameters $m_c = 1$, $m_1 = m_2 = 0.1$, $l_1 = l_2 = 0.5$ and for the controller $\alpha = 40$. This gives

   $$k^t_{stab} = k^t_{lqr} = [1 \ 2.55 \ -134.5 \ -5 \ 149.8 \ 18.6]$$

   Hence all weights w, V satisfying (13) will locally stabilize the double inverted system around the origin. However up till now nothing is guaranteed about the size of the region where the linearization is valid.

2. **Nonlinear optimization:**
   The nonlinear optimization problem (39) in V is then solved for the swinging up problem for $x(0) = [0\,0\,\pi\,0\,\pi\,0]^t$ to $x_{eq} = [0\,0\,0\,0\,0\,0]^t$ with $\eta = \|x(T)\|$, $\zeta = 0$ (terminal control), $T = 2$ and $\beta = 5$ in order to create a sufficiently large region for which the linearization of Step 1 is valid. The choice of the time horizon T was high enough to enforce the state entering the linearized region around x_{eq}. The closed loop system was simulated using a trapezoidal integration rule in the time interval $[0, 2]$

with constant step length equal to 0.002. The nonlinear optimization problem was solved using a Sequential Quadratic Programming technique of Matlab's optimization toolbox [14] (function *constr*) with numerical calculation of the gradients. In order to obtain fast simulation of the closed loop system, the *mex* facility was used with fast simulations performed in C. Trying several starting points for V (randomly chosen according to a normal distribution $N(0, 0.1)$), the best solution obtained was equal to

$$V = \begin{bmatrix} 0.0010 & 0.1026 & -0.0734 & 0.2104 & 0.4061 & 0.1893 \\ 0.1792 & 0.1322 & 0.7934 & -0.1090 & -0.6426 & -0.0825 \\ 0.0397 & 0.0144 & -0.0076 & -0.0116 & 0.0258 & 0.0322 \\ 0.1041 & 0.0758 & 0.0228 & -0.0355 & 0.1201 & 0.1425 \\ -0.0627 & -0.1485 & -0.0579 & 0.0039 & -0.4937 & -0.1307 \\ -0.0105 & -0.1615 & -0.3100 & -0.1008 & -0.2082 & 0.0102 \end{bmatrix}$$

The numerical solution to the optimization problem is shown in Fig.3. After solving this optimization problem the neural controller learned the swinging up problem for the given initial state $x(0)$. Some generalization properties with respect to other initial states are shown for $x(0) = [x_p(0)\, 0\, \pi\, 0\, \pi\, 0]^t$ with $x_p(0) \in [-0.2; 0.2]$ (see Fig.4). Small variations on $\dot{x}_p(0)$, $\dot{\theta}_1(0)$, $\dot{\theta}_2(0)$ are also allowed, but variations on $\theta_1(0)$ and $\theta_2(0)$ are critical. Simulations indicate that small perturbations on the system parameters m_c, m_1, m_2, l_1, l_2 (order of magnitude $< 1\%$) are allowable in order to remain perform the control task well. Close to the target equilibrium point one can expect similar robustness properties as for the corresponding LQR controller.

6 Conclusion

In this paper a method is proposed for realizing transitions between working points with local stabilization at the endpoint by neural controllers, either parametrized by feedforward or recurrent neural networks and applying dynamic or static output feedback. Both results from linear control theory and optimal control theory can be included and combined in one single neural controller. The neural controller is derived from a nonlinear optimization problem in the interconnection weights, constrained by the linear controller design around the endpoint. This has the advantage that a priori knowledge on linear control, which is often already available and well understood, can be translated into the neural controller. By the example of swinging up a double inverted pendulum it was illustrated that relatively small scale neural networks are capable of performing highly complex control task, using a straightforward design procedure.

References

[1] Åström K.J., B. Wittenmark, *Computer-controlled systems, theory and design*, Englewood Cliffs NJ: Prentice-Hall, 1984.

[2] Barto A.G., R.S. Sutton, C.W. Anderson, Neuronlike adaptive elements that can solve difficult learning control problems, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-13, No.5, pp. 834-846, 1983.

[3] Bryson A.E., Y.C. Ho, *Applied optimal control*, Waltham, MA: Blaisdel, 1969.

[4] Chen F-C, H.K. Khalil, Adaptive control of nonlinear systems using neural networks, *Int. J. Control*, Vol.55, No.6, pp.1299-1317, 1992.

[5] Cybenko G., Approximations by superpositions of a sigmoidal function, *Mathematics of Control, Signals and Systems*, 2, pp.183-192, 1989.

[6] Fletcher R., *Practical methods of optimization*, second edition, Chichester and New York: John Wiley and Sons, 1987.

[7] Franklin G.F., J.D. Powell, M.L. Workman, *Digital control of dynamic systems*, Reading MA: Addison-Wesley, 1990.

[8] Funahashi K-I., On the Approximate Realization of continuous Mappings by Neural Networks, *Neural Networks*, Vol.2, pp 183-192, 1989.

[9] Gill P.E., W. Murray, M.H. Wright, *Practical Optimization*, London: Academic Press, 1981.

[10] Hornik K., M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, *Neural Networks*, Vol.2, pp.359-366, 1989.

[11] Hunt K.J. , D. Sbarbaro, R. Zbikowski, P.J. Gawthrop, Neural networks for control systems - a survey, *Automatica*, Vol. 28., No. 6, pp. 1083-1112, 1992.

[12] Larcombe P., On the control of a two dimensional multi-link inverted pendulum: coordinate system suitable for dynamic formulation, *Proceedings of the 30th Conference on Decision and Control*, pp. 136-141, Brighton, England, Dec. 1991.

[13] Maciejowski J.M., *Multivariable feedback design*, Addison-Wesley, 1989.

[14] Matlab User's Guide, *Optimization Toolbox User's Guide*, Version 3.5f, The MathWorks, Inc., 1992.

[15] Miller W.T., R.S. Sutton, P.J. Werbos, *Neural networks for control*, Cambridge, MA: M.I.T. Press, 1990.

[16] Narendra K.S., Parthasarathy K., Identification and control of dynamical systems using neural networks, *IEEE Trans. on Neural Networks*, Vol.1, No.1, pp. 4-27, March 1990.

[17] Nguyen D., Widrow B., Neural networks for self-learning control systems, *IEEE Control Systems Magazine*, 10(3), pp.18-23, 1990.

[18] Powell M.J.D., Variable metric methods for constrained optimization, In: A. Bachem, M. Grötschel and B. Korte (Eds.), *Mathematical programming: the state of the art*, pp.288-311, Berlin and New York: Springer-Verlag, 1983.

[19] Rice J.R., *Numerical methods, software and analysis*, Mc Graw-Hill, 1983.

[20] Saerens M., J.-M. Renders, H. Bersini, Neural controllers based on backpropagation algorithm. In *IEEE-Press Book on Intelligent Control: Theory and Practice*, M. M. Gupta, N. K. Sinha (Eds.), IEEE Press, 1993.

[21] Slotine J.-J. E., R.M. Sanner, Neural networks for adaptive control and recursive identification: a theoretical framework. in *Essays on control: Perspectives in the theory and its applications*, H.L. Trentelman, J.C. Willems (Eds.), Birkhäuser, 1993.

[22] Suykens J., B. De Moor, J. Vandewalle, Static and dynamic stabilizing neural controllers, applicable to transition between equilibrium points, *ESAT-SISTA report 1993-03*, Department of Electrical Engineering, Leuven, Belgium, submitted for publication.

[23] Thierens D., J. Suykens, J. Vandewalle, B. De Moor, Genetic weight optimization of a feedforward neural network controller, *Artificial neural nets and genetic algorithms*, Proceedings of the international conference in Innsbruck, Austria 1993, (eds. R.F. Albrecht, C.R. Reeves, N.C. Steele), Springer-Verlag Wien New York.

[24] Werbos P., Backpropagation through time: what it does and how to do it, *Proceedings of the IEEE*, 78 (10), pp.1150-1560, 1990.

List of captions:

Figure 1: *Stabilizing neural controllers, parametrized by feedforward or recurrent neural networks.*

Figure 2: *Double inverted pendulum system with state vector* $x^t = [x_p\ \dot{x}_p\ \theta_1\ \dot{\theta}_1\ \theta_2\ \dot{\theta}_2]$ *and input* u *equal to the force* F *applied to the cart.*

Figure 3: *Numerical solution to the swinging up problem for a double inverted pendulum using a feedforward neural net controller, consisting of one hidden layer with 6 hidden neurons and applying state feedback. An LQR design was done around the target equilibrium point. Initial state is* $x(0) = [0\,0\,\pi\,0\,\pi\,0]^t$ *and desired final state* $x_{eq} = 0$. *Fig.a/* $x_1(t)$, $x_3(t)$, $x_5(t)$, *Fig.b/* $x_2(t)$, $x_4(t)$, $x_6(t)$, *Fig.c/ control signal* $u(t)$*: first a high force is needed in order to swing up the double inverted pendulum and after approximately* $1.5s$ *the neural controller acts as a classical LQR controller, Fig.d/ simulation result, visualized by the position of the two links through time.*

Figure 4: *Generalization ability of the neural controller after learning the swinging up problem for* $x(0) = [0\,0\,\pi\,0\,\pi\,0]^t$ *for other initial positions of the cart* $x_p(0) \in [-0.2; 0.2]$ *with* $x(0) = [x_p(0)\,0\,\pi\,0\,\pi\,0]^t$.

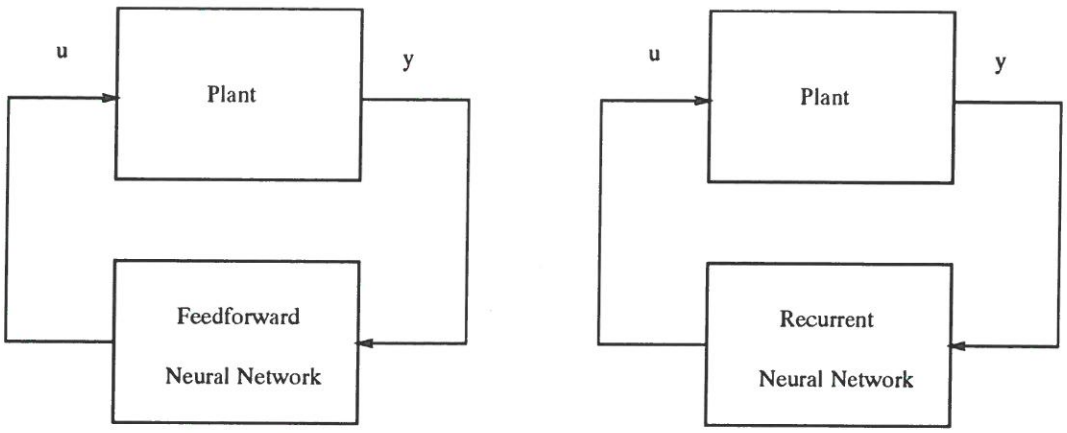

u
Plant
y
Feedforward
Neural Network
u
Plant
y
Recurrent
Neural Network

Figure 1:

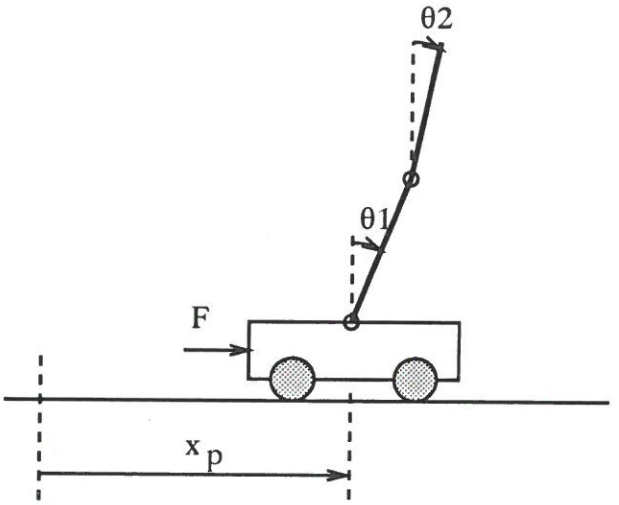
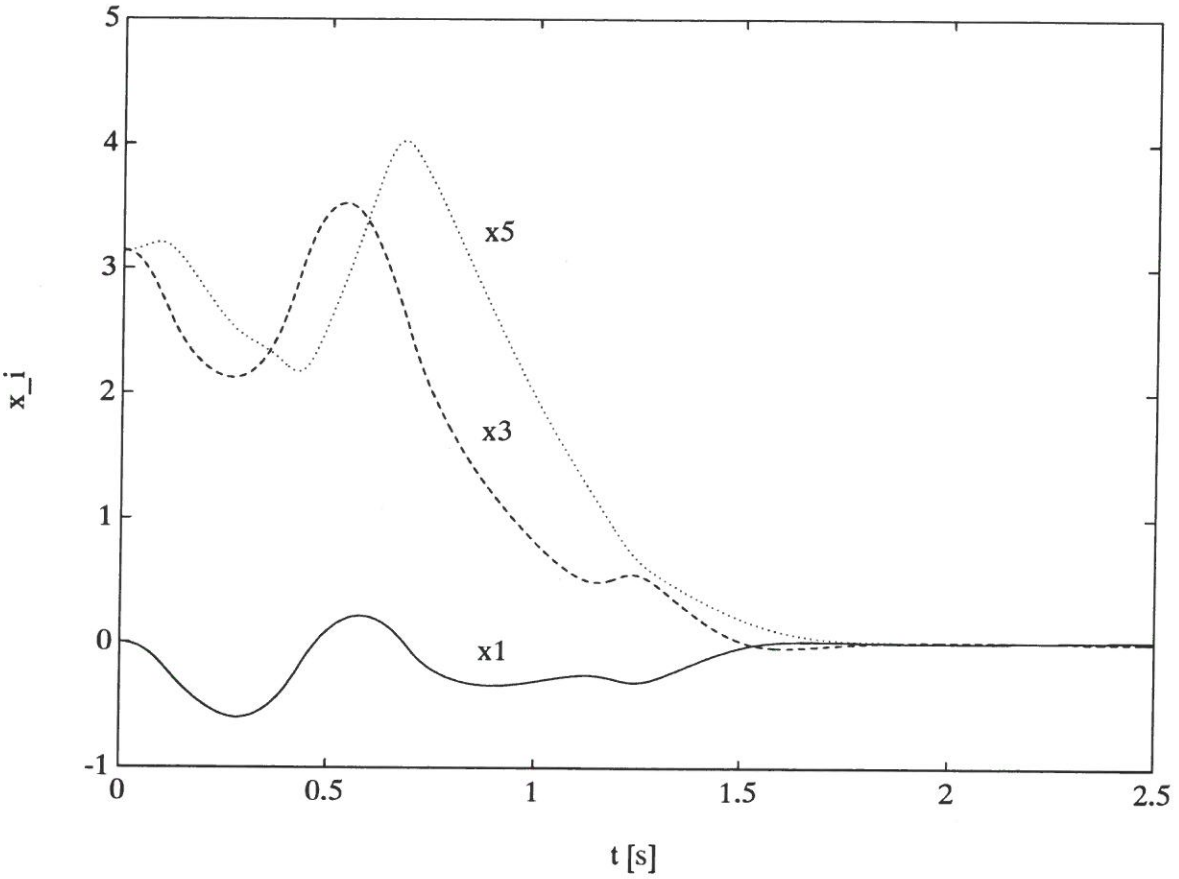
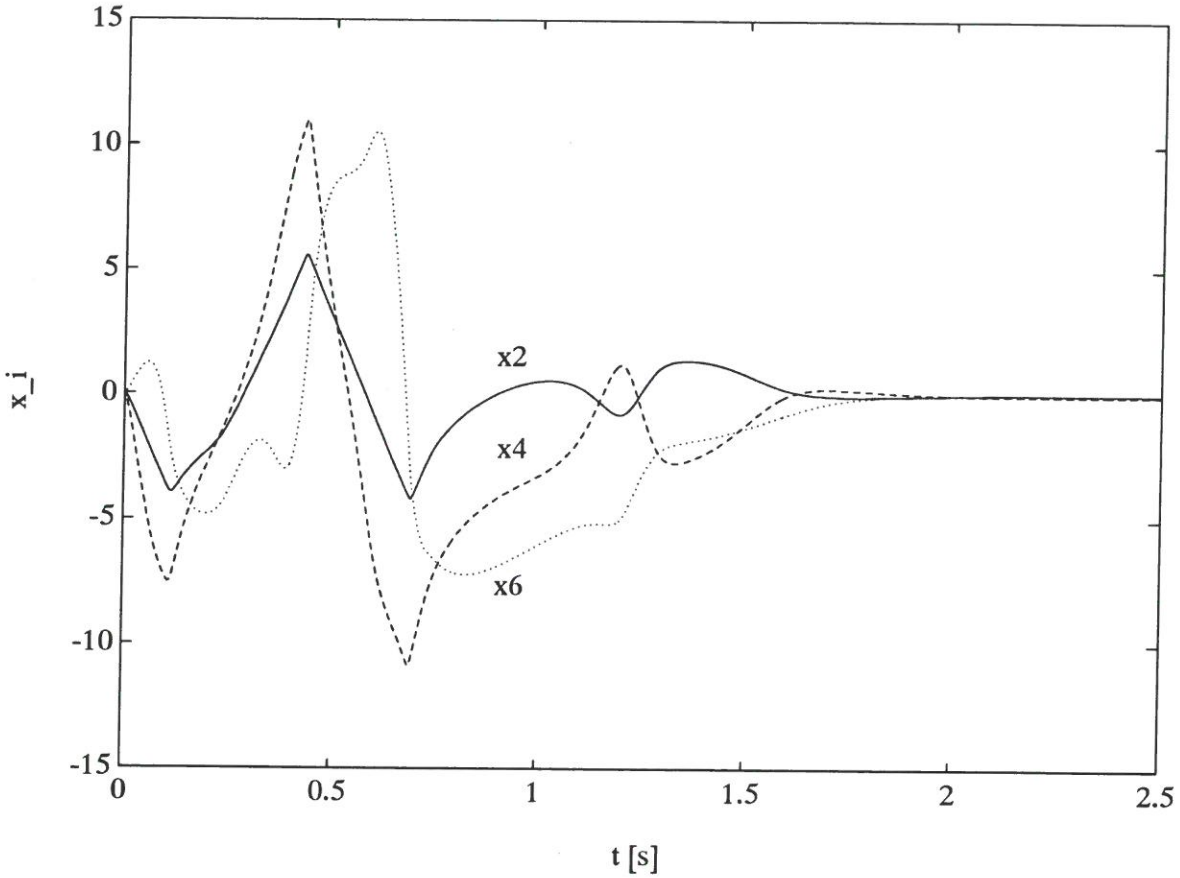
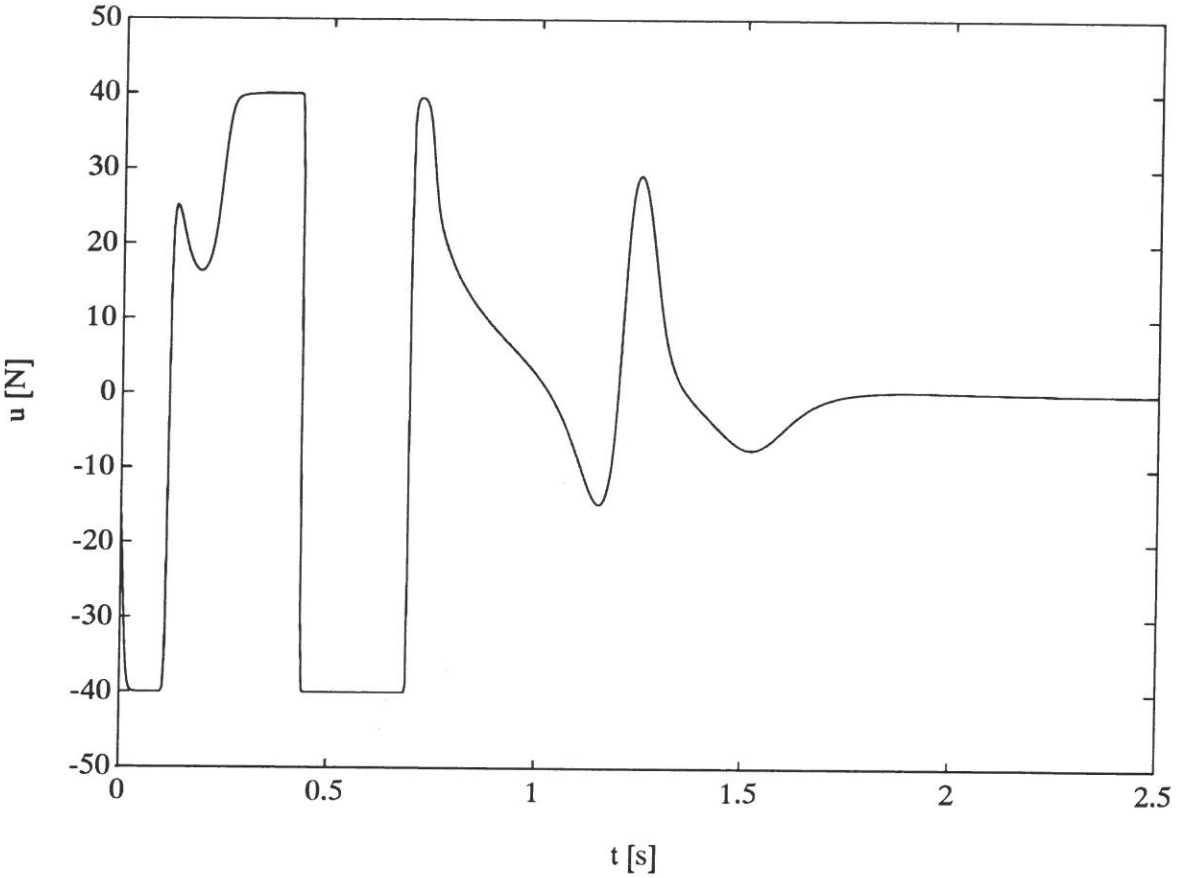


Figure 2:


5
4
3
2
1
0
-1
0
0.5
1
1.5
2
2.5
x_i
t [s]
x5
x3
x1


15
10
5
0
-5
-10
-15
0
0.5
1
1.5
2
2.5
x_i
t [s]
x2
x4
x6


50
40
30
20
10
0
-10
-20
-30
-40
-50
u [N]
0
0.5
1
1.5
2
2.5
t [s]

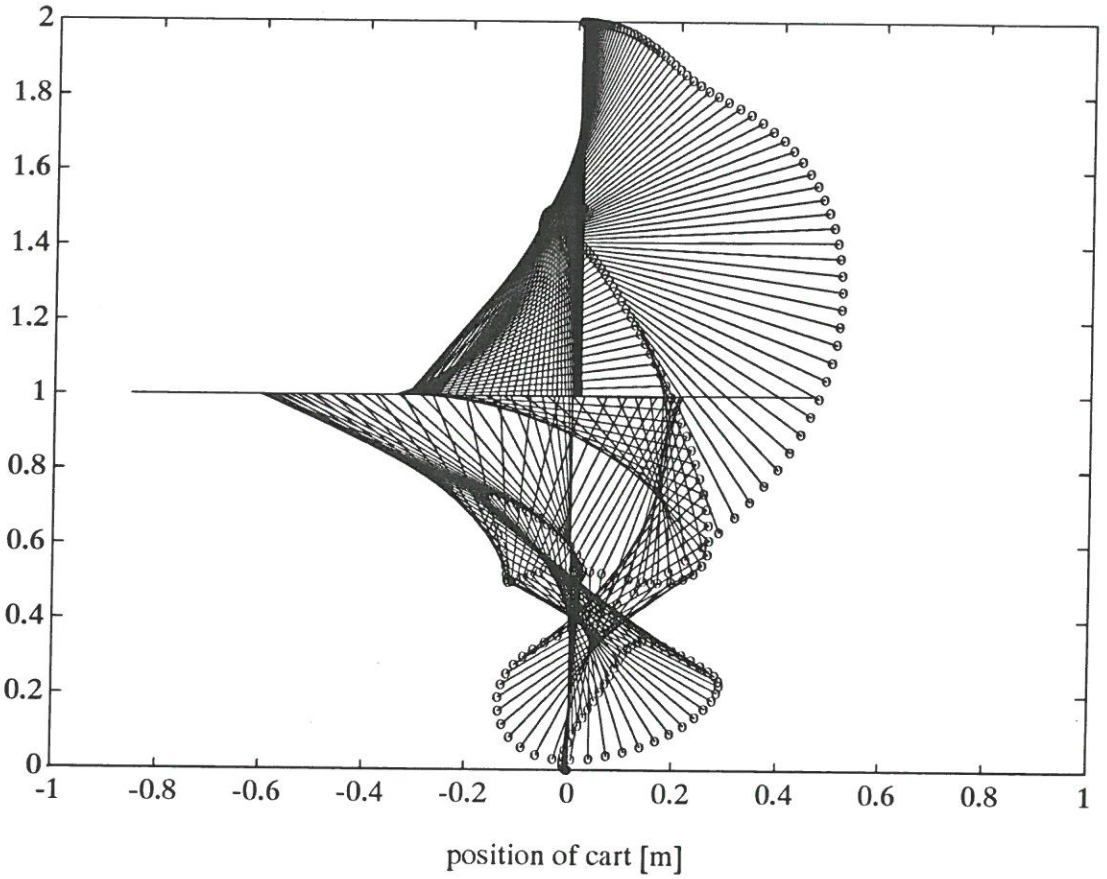

2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
position of cart [m]

Figure 3:

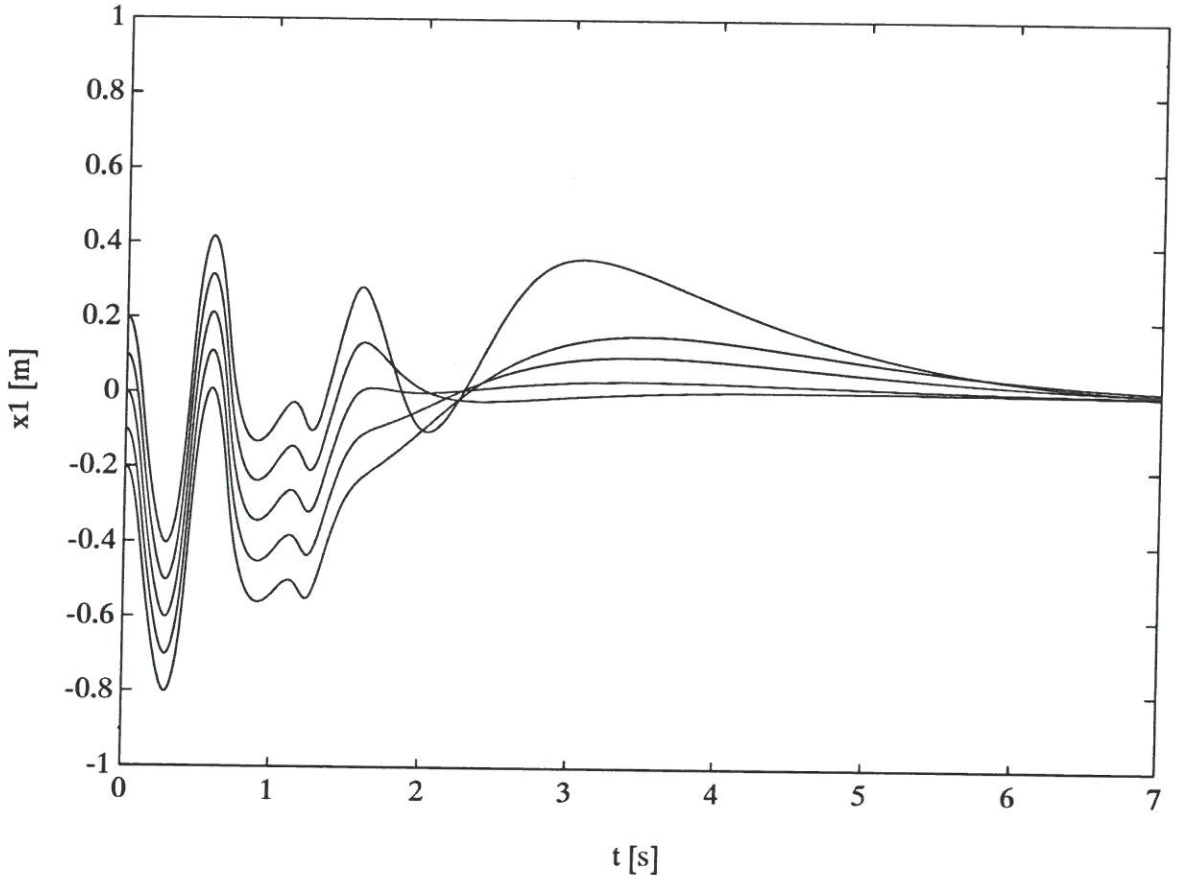


Figure 4: