Preprints of the 18th IFAC World Congress
Milano (Italy) August 28 - September 2, 2011

Real-time application of a constrained predictive controller based on dynamic neural networks with feedback linearization ⋆

Jiamei Deng * **Victor M Becerra** ** **Richard Stobart** ***
Shaohua Zhong ****

* *Department of Aeronautical and Automotive Engineering, Loughborough University, Leicestershire, UK LE11 3TU (Tel: +44-1509-227268; e-mail: j.deng@lboro.ac.uk).*
** *School of Systems Engineering, University of Reading, Reading, UK RG6 6AY (e-mail: v.m.becerra@reading.ac.uk)*
*** *Department of Aeronautical and Automotive Engineering, Loughborough University, Leicestershire, UK LE11 3TU (e-mail: r.k.stobart@lboro.ac.uk)*
**** *School of Automotive Engineering, Wuhan University of Technology, Wuhan, China 430070,P.R.China (e-mail: zhongsh2004@gmail.com)*

Abstract: This paper describes an experimental application of constrained predictive control and feedback linearisation based on dynamic neural networks. It also verifies experimentally a method for handling input constraints, which are transformed by the feedback linearisation mappings. A performance comparison with a PID controller is also provided. The experimental system consists of a laboratory based single link manipulator arm, which is controlled in real time using MATLAB/SIMULINK together with data acquisition equipment.

1. INTRODUCTION

Feedback linearisation is an important nonlinear control method. The principles and applications of feedback linearization have been reported, for instance, in Kravaris and Kantor [1990a], Kravaris and Kantor [1990b], Kravaris and Arkun [1991], Henson and Seborg [1997], Nijmeijer and van der Schaft [1990], Slotine and Li [1991], Vidyasagar [1993], and Isidori [1999]. Predictive control has become an attractive control strategy, particularly for plants subject to input and output constraints Maciejowski [2002]. Different algorithms have been proposed, but they all share the essential features: an explicit internal model, the receding horizon idea, and the computation of the control signal by optimizing the predicted plant behavior. The success of predictive control is mainly due to its handling of constraints.

Some researchers have addressed the integration of feedback linearising techniques and predictive control. Ayala-Botto et al. [1999] described a constrained non-linear predictive control technique based on input-output linearisation for general discrete time affine neural network models, using an iterative method for handling the input constraints. Also, Kurtz et al. [2000] investigated the technique of combined feedback linearisation and multivariable linear predictive control to a simulated polymerisation reactor, using an approximation to deal with input constraints and based on a physical model of the process. Moreover, Nevistic and Morari [1996] compared the performance of combined predictive control and feedback linearisation with full nonlinear predictive control and with open-loop optimal control.

⋆ This work was supported by the UK Engineering and Physical Science Research Council under Grant Reference Numbers: GR/R64193 and the University of Reading Chinese Studentship Programme

The use of neural networks for dynamic system identification has received much attention over the last two decades. The researchers first paid attention to static networks such as multilayer perceptrons (MLPs) in Rumelhart and McClelland [1986] Churchland et al. [1992], and radial basis functions in Broomhead and Lowe [1988]. An alternative to MLPs is Hopfield networks described in Hopfield [1982].

This paper reports the real-time application of a constrained predictive controller based on dynamic neural networks with feedback linearization to a laboratory based single link manipulator, including a simple method for handling input constraints. A performance comparison with a PID controller is also provided. The paper is organised as follows: Section 2 presents the predictive control; Section 3 introduces the dynamic neural networks; Section 4 discusses feedback linearisation; Section 5 discusses constraint handling; and Section 6 presents real time experimental results, and finally this is followed by the conclusions in Section 7.

2. MODEL-BASED PREDICTIVE CONTROL

In this paper, a linear, discrete-time, state-space model of the plant is assumed, which makes sense as the system will be linear after the feedback linearisation. The model has the following form:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= C_y x(k) \\ z(k) &= C_z x(k), \end{aligned} \tag{1}$$

where $x(k)$ is the state vector at time k, $u(k)$ is the vector of inputs, $y(k)$ is the measured outputs, and $z(k)$ is vector of outputs which are to be controlled to satisfy some constraints, or to particular set-points, or both. In this work, a Kalman filter

was used to estimate the state vector that could be described as follows:

$$\begin{aligned}\hat{x}(k+1|k) &= A\hat{x}(k|k-1) + Bu(k) + L\hat{e}(k|k)\\ \hat{y}(k|k-1) &= C_y\hat{x}(k|k-1)\\ \hat{z}(k|k-1) &= C_z\hat{x}(k|k-1),\end{aligned} \tag{2}$$

where $\hat{x}(k+1|k)$ is the estimate of the state at future time $k+1$ based on the information available at time k, $\hat{y}(k|k-1)$ is the estimate of the plant output at time k based on information at time $k-1$, L is the Kalman filter gain matrix and $\hat{e}(k|k)$ is the estimated error: $\hat{e}(k|k) = y(k) - \hat{y}(k|k-1)$.

The formulation given in this paper is inspired in the constrained algorithm presented in Maciejowski [2002]. The cost function V minimised by the predictive controller penalises deviations of the predicted controlled outputs $\hat{z}(k+i|k)$ from a reference trajectory $r(k+i|k)$ and also it penalises changes in the future manipulated inputs $\Delta\hat{u}(k+i|k)$. Define the cost function as follows

$$\begin{aligned}V(k) = &\sum_{i=1}^{p} ||\hat{z}(k+i|k) - r(k+i|k)||^2_{Q(i)}\\ &+ \sum_{i=0}^{m-1} ||\Delta\hat{u}(k+i|k)||^2_{R(i)},\end{aligned} \tag{3}$$

where the prediction and control horizons are p and m, respectively, $Q(i)$ and $R(i)$ are output weight and input weight matrices, respectively. The cost function can also be written as follows:

$$V(k) = ||Z(k) - T(k)||^2_Q + ||\Delta U(k)||^2_R, \tag{4}$$

where

$$\begin{aligned}Z(k) &= \begin{bmatrix}\hat{z}(k+1|k)\\ \vdots\\ \hat{z}(k+p|k)\end{bmatrix}\\ T(k) &= \begin{bmatrix}\hat{r}(k+1|k)\\ \vdots\\ \hat{r}(k+p|k)\end{bmatrix}\\ \Delta U(k) &= \begin{bmatrix}\Delta\hat{u}(k|k)\\ \vdots\\ \Delta\hat{u}(k+m-1|k)\end{bmatrix},\end{aligned} \tag{5}$$

$$\begin{aligned}R &= \begin{bmatrix}R(0) & 0 & \dots & 0\\ 0 & R(1) & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \dots & R(m-1)\end{bmatrix}\\ Q &= \begin{bmatrix}Q(1) & 0 & \dots & 0\\ 0 & Q(2) & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \dots & Q(p)\end{bmatrix}.\end{aligned} \tag{6}$$

Note that

$$Z(k) = \psi x(k) + \Upsilon u(k-1) + \Theta\Delta U(k), \tag{7}$$

where

$$\psi = \begin{bmatrix}C_zA\\ \vdots\\ C_zA^m\\ C_zA^{m+1}\\ \vdots\\ C_zA^p\end{bmatrix},\ \Upsilon = \begin{bmatrix}C_zB\\ \vdots\\ \sum_{i=0}^{m-1} C_zA^iB\\ \sum_{i=0}^{m} C_zA^iB\\ \vdots\\ \sum_{i=0}^{p-1} C_zA^iB\end{bmatrix},\ \text{and}\ \Theta = \begin{bmatrix}C_zB & \dots & 0\\ AB+B & \dots & 0\\ \vdots & \ddots & \vdots\\ \sum_{i=0}^{m-1} A^iB & \dots & B\\ \sum_{i=0}^{m} A^iB & \dots & AB+B\\ \vdots & \ddots & \vdots\\ \sum_{i=0}^{p-1} A^iB & \dots & \sum_{i=0}^{p-m} C_zA^iB\end{bmatrix}.$$

Define

$$\varepsilon(k) = T(k) - \psi x(k) - \Upsilon u(k-1). \tag{8}$$

Now the following can be obtained:

$$\begin{aligned}V(k) &= ||\Theta\Delta U(k) - \varepsilon(k)||^2_Q + ||\Delta U(k)||^2_R\\ &= \varepsilon(k)^TQ\varepsilon(k) - 2\Delta U(k)^T\Theta^TQ\varepsilon(k)\\ &+ \Delta U(k)^T[\Theta^TQ\Theta + R]\Delta U(k).\end{aligned} \tag{9}$$

Equation (9) has the form

$$V(k) = \text{const} - \Delta U(k)^T\vartheta + \Delta U(K)^TH\Delta U(k), \tag{10}$$

where

$$\vartheta = 2\Theta^TQ\varepsilon(k), \tag{11}$$

and

$$H = \Theta^TQ\Theta + R. \tag{12}$$

Then, to minimise V as given by equation (3), the following constrained optimization problem can be solved:

$$\min_{\Delta U(k)} \Delta U(k)^TH\Delta U(k) - \vartheta^T\Delta U(k), \tag{13}$$

subject to the inequality constraints:

$$\begin{aligned}u(k) &\geq u_{min}(k)\\ u(k) &\leq u_{max}(k)\\ |\Delta u(k)| &\leq \Delta u_{max}(k)\\ z(k) &\geq z_{min}(k)\\ z(k) &\leq z_{max}(k).\end{aligned} \tag{14}$$

This optimisation problem can be expressed as follows:

$$\min_{\theta} \theta^T\phi\theta + \varphi^T\theta, \tag{15}$$

subject to

$$\Omega\theta \leq \omega, \tag{16}$$

where $\theta = \Delta U(k)$, $\phi = H$, $\varphi = -\vartheta$. Expressions for matrix Ω and vector ω are obtained from Equation (14).

Recall the state space model (1) and define

$$\begin{aligned} \Delta u(k) &= \quad u(k) - u(k-1) \\ \Delta x(k) &= \quad x(k) - x(k-1). \end{aligned} \tag{17}$$

A convenient augmented model can be obtained as follows:

$$\begin{aligned} x_a(k+1) &= A_a x_a(k) + B_a \Delta u(k) \\ y(k) &= C_a x_a(k) + Du(k), \end{aligned} \tag{18}$$

where

$$\begin{aligned} x_a(k) &= \begin{bmatrix} \Delta x(k) \\ y(k) \end{bmatrix} \\ A_a &= \begin{bmatrix} A & 0 \\ CA & I \end{bmatrix} \\ B_a &= \begin{bmatrix} B \\ CB \end{bmatrix}. \end{aligned} \tag{19}$$

This augmented model was used by the real-time predictive controller employed in this work. If the system is at steady state, then $\Delta x(0) = 0$. As $y(0)$ is the initial position of the system, $x_a(0)$ is known. This augmented model is therefore easily initialized, which is an advantage over the general state space model (1). Predictive control algorithm has been implemented in the C programming language and interfaced to SIMULINK. The algorithm uses linear state space models of the form given in Equation (18). The implemented algorithm follows the formulation given in this paper. A public domain Quadratic Programming algorithm by K. Schittkowski was employed Schittkowski [1992]. The predictive control algorithm was interfaced to SIMULINK through an S-Function. The algorithm is compatible with the Real Time Windows Target product. The controller tuning parameters can be set through the S-function interface. The implemented algorithm is described in Deng and Becerra [2004].

3. DYNAMIC NEURAL NETWORKS

The structure of the DNN used in this paper is analogous to a general affine system of the form

$$\begin{aligned} \dot{X} &= f(X) + g(X)u \\ y &= h(X) \end{aligned} \tag{20}$$

where $f(x), g(x) = [g_1, g_2, ..., g_m]$ are smooth vector fields, and $h(x)$ smooth function, defined on an open set of R^n.Any finite time trajectory of a given dynamic system can be approximated by the internal state of the output units of a continuous time DNN with N dynamic units, m inputs and n outputs Garces et al. [2003]. These networks are neural unit arrays which have connections both ways between a pair of units, and from a unit to itself. The model is defined by a one-dimensional array of N neurons or units, in which each neuron of the DNN can be described as follows:

$$\dot{X}_i = -\beta_i X_i + \sum_{j=1}^{N} \omega_{ij} \sigma(X_j) + \sum_{j=1}^{m} \gamma_{ij} u_j, \tag{21}$$

where β_i, ω_{ij} and γ_{ij} are adjustable weights, with $1/\beta_i$ as positive time constant and $n \leq N$, x_i the activation state of unit i, and $u_1, ..., u_m$ the input signals. The states of the first n neurons are taken as the output of the network, leaving $N - n$ units as hidden neurons. The network is defined by the vectorised expression of (21) as follows:

$$\dot{X} = -\beta X + \omega\sigma(X) + \gamma u \tag{22}$$

$$y = CX, \tag{23}$$

where X are the coordinates on $\Re^N$, $\beta \in \Re^{N \times N}$, $\gamma \in \Re^{N \times m}$, $u \in \Re^m$, $\sigma(X) = [\sigma(X_1), ..., \sigma(X_N)]^T$, $\sigma(\cdot)$ is a sigmoidal function, such as the hyperbolic tangent, $C = [I_{n \times n}\, 0_{n \times (N-n)}]$ and $\beta = \mathrm{diag}(\beta_1, ..., \beta_N)$ is a diagonal matrix. This paper used a DNN described by Equation (22)-(23). The state vector X of the DNN of Equation (22) can be partitioned into the output state X^o and the hidden states X^h:

$$X = \begin{bmatrix} X^o \\ X^h \end{bmatrix}. \tag{24}$$

A dynamic neural network training problem can be cast as a nonlinear unconstrained optimization problem:

$$\min_{\theta} \quad F_M(\theta, Z_M) = \frac{1}{2M} \sum_{k=1}^{M} ||y(t_k) - \hat{y}(t_k|\theta)||^2 \tag{25}$$

where $Z_M = [y(t_k), u(t_k)]_{k=1,M}$ is a training data set, $y(t_k)$ represents the measured output, $\hat{y}(t_k|\theta)$ is the DNN output, and θ is a parameter vector.

The optimization problem associated with training usually exhibits local minima. Hence, training DNN's is typically performed using unconstrained local optimization with multiple random starts, global optimization methods, or hybrid methods.

In this work the dynamic networks are trained using a gradient optimization method. The training algorithm will find the parameters of the network in Equation (22) for which the error function F_M is minimized. The weight vector θ used by the algorithm is the aggregate of the neurons feedback weight matrix $\omega_{N \times N}$, the input weight matrix $\gamma_{N \times m}$, the state feedback weight vector $\beta_{N \times 1} = [\beta_1, \ldots, \beta_N]^T$ and the initial values of the hidden states of the DNN, $x^h(t_0)$. That is:

$$\theta = \begin{bmatrix} \beta_{N \times 1} \\ \mathrm{vec}(\gamma_{N \times m}) \\ \mathrm{vec}(\omega_{N \times N}) \\ x^h(t_0)_{(N-n) \times 1} \end{bmatrix}. \tag{26}$$

4. FEEDBACK LINEARISATION

Many nonlinear control methods are based on state space models where the time derivative of the states depends nonlinearly on the states and linearly on the control inputs as described in Isidori [1995], which are known as control affine systems, and are described by Equation (20).

The purpose of input-output linearisation is to introduce a new input variable v and a nonlinear transformation that uses state feedback to compute the original input u, so that a system described by Equation (20) behaves linearly from the new input v to the output y.

It is of considerable importance to assess which systems can be input-output linearized. The necessary and sufficient conditions for the existence of a feedback law are presented in Isidori [1995].

Input-output linearisation and decoupling is a particular case of input-output linearisation. An appropriate selection of the design parameters leads to a feedback linearised system where the ith output depends on the ith external input only. The basic linearising-decoupling technique is described in Isidori [1995]. In this paper, we apply the linearising-decoupling technique to the dynamic neural network model, which is a control affine system, as formulated by Garces et al. [2003]. The control affine

system mappings related to Equation (20) can be related to the DNN parameters as follows:

$$\begin{aligned} f(X) &= -\beta X + \omega\sigma(X) \\ g(X) &= \gamma \\ h(X) &= CX \end{aligned} \tag{27}$$

Consider the dynamic neural network described by Equation (22) and (23), and suppose that the DNN has a vector relative degree given by $r = [r_1, r_2, ..., r_n]$ at a point x_0. Assume that DNN has the same number of inputs and output ($n = m$). For arbitrary values $\hat{\lambda}_{ik}$ $(i = 1, \ldots, n$ and $k = 0, \ldots, r_i)$ a state feedback law with

$$u = P(X) + Q(X)v \tag{28}$$

where

$$\begin{aligned} P(X) &= -A(X)^{-1}B(X) \\ Q(X) &= A(X)^{-1} \end{aligned} \tag{29}$$

with

$$A(X) = \begin{bmatrix} \hat{\lambda}_{1r_1} L_{g_1} L_f^{r_1-1} X_1 & \cdots & \hat{\lambda}_{1r_1} L_{g_n} L_f^{r_1-1} X_1 \\ \vdots & \ddots & \vdots \\ \hat{\lambda}_{pr_p} L_{g_1} L_f^{r_n-1} X_n & \cdots & \hat{\lambda}_{nr_n} L_{g_n} L_f^{r_p-1} X_n \end{bmatrix}_{n\times n} \tag{30}$$

and

$$B(X) = \begin{bmatrix} \sum_{k=0}^{r_1} \hat{\lambda}_{1k} L_f^k X_1 \\ \vdots \\ \sum_{k=0}^{r_n} \hat{\lambda}_{nk} L_f^k X_n \end{bmatrix}_{n\times 1} \tag{31}$$

where the notation $L_f^k h$ denotes a Lie derivative of order k of a scalar function $h(x)$ along vector field f, $\hat{\lambda}_{ik}$s are scalar design parameters, and r_i is the relative degree of the ith output $\hat{y}_i$, produces when applied to a DNN described by Equations (22) and (23), a linearised-decoupled system that obeys

$$\sum_{k=0}^{r_i} \hat{\lambda}_{ik} \frac{d^k \hat{y}_i}{dt^k} = v_i, \quad i = 1 \ldots n \tag{32}$$

if the relative vector is well defined, $A(X)$ is invertible and

$$\det\left[\operatorname{diag}\left(\hat{\lambda}_{1r_1}, \hat{\lambda}_{2r_2}, \ldots, \hat{\lambda}_{nr_n}\right)\right] \neq 0 \tag{33}$$

Suppose that the DNN described by Equations (22) and (23) is trained to approximate trajectories from a nonlinear plant described by Equation (20). If the above linearising-decoupling law described by Equations (28)–(31), which based on a dynamic neural network model, is applied to nonlinear plant (20), which the neural network approximates over a region of interest, then the dynamics of the plant are approximately input-output linearised and decoupled, and obey the following equations described in Garces et al. [2003]:

$$\sum_{k=0}^{r_i} \hat{\lambda}_{ik} \frac{d^k y_i}{dt^k} = v_i + \sum_{k=0}^{r_i} \hat{\lambda}_{ik} \frac{d^k e_i}{dt^k}, \quad i = 1, \ldots, n \tag{34}$$

where e_i is the model error corresponding to the ith output:

$$e_i = y_i - \hat{y}_i \tag{35}$$

It is not difficult to see that the model error acts as an output disturbance on the linearised and decoupled system. It is possible to ensure that the model error e_i remains bounded by appropriate training, so that its effect on the linearising and decoupling action remains small within a region of the input–output space.

It should be noted that the approximate feedback linearisation and decoupling laws described above require state information. Since the model employed in this work to generate the linearising-decoupling laws is a DNN, the same model can also be used as a closed loop observer to provide state information to the feedback linearizing laws.

Once the feedback linearization is tested and validated on the actual plant, the constrained predictive controller can be used to control the linearised system. This controller can deal with disturbances arising from modelling errors and other sources, and it can handle constraints on plant variables. While handling output constraints using the method presented in this paper is straightforward, input constraints require special treatment due to the presence of the nonlinear transformations associated with feedback linearisation, which will be addressed in the next section.

5. HANDLING OF INPUT CONSTRAINTS AFTER LINEARISATION

Consider the second order SISO dynamical neural network described by the following equations:

$$\dot{X} = -\beta X + \omega\sigma(X) + \gamma u \tag{36}$$

$$y = X_1 \tag{37}$$

where $\beta = \begin{bmatrix} \beta_1 & 0 \\ 0 & \beta_2 \end{bmatrix}$, $\gamma = \begin{bmatrix} \gamma_1 \\ \gamma_2 \end{bmatrix}$, $\omega = \begin{bmatrix} \omega_{11} & \omega_{12} \\ \omega_{21} & \omega_{22} \end{bmatrix}$, $X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$, and $\sigma(X) = \begin{bmatrix} \sigma(X_1) \\ \sigma(X_2) \end{bmatrix}$. This neural network is an example of a SISO control affine system with $f(X)$, $g(X)$, and $h(X)$ given by:

$$f(X) = \beta X + \omega\sigma(X) \tag{38}$$

$$g(X) = \gamma \tag{39}$$

$$h(X) = X_1 \tag{40}$$

Notice that the vector fields $f(X)$, $g(X)$ and the function $h(X)$ are smooth and differentiable for all $X \in \Re^2$. consider that:

$$L_f^0 h(X) = h(X) = X_1 \tag{41}$$

$$C(X) = L_g L_f^0 h(X) = [1\ 0] g(X) \tag{42}$$

Using Definition 1, it is possible to conclude that the relative degree of this system is $r = 1$, provided that $\gamma_1 \neq 0$. Suppose that λ_0 and λ_1 are the design parameters for the linearised system. Then the linearising input is given by:

$$\begin{aligned} u &= \frac{v - \sum_{k=0}^{r} \lambda_k L_f^k h(X)}{\lambda_r L_g L_f^{r-1} h(X)} = \frac{v = \lambda_0 L_f^0[X_1] - \lambda_1 L_f^1[X_1]}{\lambda_1 L_g L_f^0[X_1]} \\ &= \frac{v - \lambda_0 X_1 - \lambda_1(-\beta_1 X_1 + \omega_{11}\sigma(X_1) + \omega_{12}\sigma(X_2))}{\lambda_1\gamma_1} \end{aligned} \tag{43}$$

λ_0 and λ_1 can be computed such that the linearised system $v - y$ had the same static gain and complex poles as the Jacobian linearisation of the dynamic neural network. Recall that Equation (13) is subject to Equation (14). However, The

implementation of the feedback linearisation scheme maps the input vector of the non-linear process, u into predictive controller's output v, and so the original set of inequality constraints in u is transformed into a new set of inequality constraints in v. For the single-input and single-output system, we have Deng et al. [2009]:

$$\begin{aligned} u_{min} \leq \ u \ \leq u_{max} \\ \Delta u_{min} \leq \Delta u \leq \Delta u_{max} \end{aligned} \tag{44}$$

Recall Equation (43) and define:

$$A_0 = \lambda_1 \gamma_1 \tag{45}$$

$$B(X(k)) = \lambda_0 X_1 + \lambda_1(-\beta_1 X_1 + \omega_{11}\sigma(X_1) + \omega_{12}\sigma(X_2))$$

Therefore,

$$\begin{aligned} v(k) = A_0 u(k) + B(X(k)) \\ v(k-1) = A_0 u(k-1) + B(X(k-1)) \end{aligned} \tag{46}$$

Thus, we have:

$$v(k) - v(k-1) = A_0(u(k) - u(k-1)) + B(X(k)) - B(X(k-1)). \tag{47}$$

Therefore,

$$\Delta v(k) = A_0 \Delta u + B(X(k)) - B(X(k-1)). \tag{48}$$

The constraints for $\Delta v(k)$ are:

$$\begin{aligned} \Delta u_{min} A_0 + B(X(k)) - B(X(k-1)) \leq \Delta v(k) \\ \leq \Delta u_{max} A_0 + B(X(k)) - B(X(k-1)). \end{aligned} \tag{49}$$

From Equation (44) and Equation (46), the constraint for $v(k)$ is that given by the bounds $u_{max}A_0 + B(X(k))$ and $u_{min}A_0 + B(X(k))$:

$$u_{min} A_0 + B(X(k)) \leq \Delta v(k) \leq u_{max} A_0 + B(X(k)). \tag{50}$$

6. CASE STUDY

6.1 A single link manipulator

The single link manipulator is driven by a servo motor system, see Figure 1. The servo motor drives an independent output gear whose angular position is measured by an encoder. The single link manipulator arm is mounted to an output gear. The system is interfaced by means of a data acquisition card and driven by MATLAB/SIMULINK based real time software, with a sampling interval of 0.02 seconds. The control input is the voltage (in V) applied to the DC motor, while the output is the angular position (in rad) of the link. An identification experiment was carried out on this single link manipulator by applying random steps in the input u over a period of time of 100s. The histories were split into two sets of equal length intended for training and validation. Using the trained dynamic neural network, Figure 2 compares the neural network output obtained for the validation input with the validation output.

6.2 Experimental results

The predictive controller described in Section 2 and the linearising control strategy describe in Section 4 were implemented in SIMULINK and applied to the single link manipulator.

The identified dynamic neural model was used. The feedback linearising law given in (43) was employed, with the values

Fig. 1. A single link manipulator and a servo-motor.

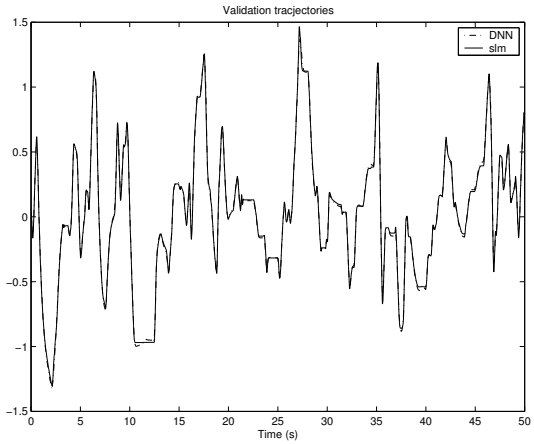

Fig. 2. Comparison between the DNN model output and the validation output

$\lambda_0 = 2.0459$ and $\lambda_1 = 0.7676$. The tuning of the predictive controller was as follows: prediction horizon $p = 300$, control horizon $m = 2$, output weight $Q = 0.1$, input weight $R = 0.1$. The control signal $u(k)$ was bounded to the interval $[-5, 5]$ V, while its increment was bounded by $|\Delta u(k)| \leq 0.01$ V. Figure 3 shows the comparison of the PID and the proposed predictive control scheme using feedback linearisation. The PID controller was manually tuned to produce fast response with little overshoot. It can be seen that the performance of the proposed predictive control scheme is better that of the PID controller as the response based on predictive control is faster and produces no overshoot.

7. CONCLUSIONS

This paper has described the SIMULINK/Real Time Windows Target implementation of a constrained predictive control algorithm using feedback linearisation based on dynamic neural

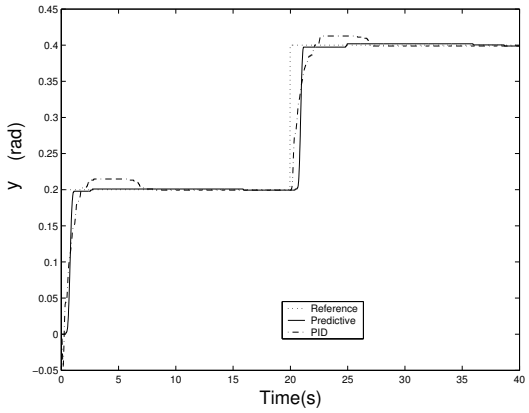

Fig. 3. Comparison of performance between the constrained predictive controller with feedback linearisation and a PID controller.

networks, and its application to a laboratory based single link manipulator. The dynamic behaviour of the nonlinear system is modelled using the dynamic neural network, which is trained based on data measured from the system. The feedback linearisation scheme employed allows the approximate cancellation of the nonlinearities of the plant, so that it can be easily controlled using a linear control scheme. A method for handling input constraints within the proposed control scheme has also been verified. The effectiveness of the methods has been experimentally demonstrated.

REFERENCES

M. Ayala-Botto, T. Van Den Boom, A. Krijgsman, and J. Sa da Costa. Predictive control based on neural network models with i/o feedback linearization. *International Journal of Control*, 72(17):1538–1554, 1999.

D. S. Broomhead and D. Lowe. Multi-variable functional interpolation and adaptive networks. *Complex Systems*, 2: 321–355, 1988.

P. S. Churchland, T.J. Sejnowski, and T.J. Seynowski. *The computational brain*. MIT Press, Cambridge, Mass., 1992.

J. Deng and V. M. Becerra. Real-time constrained predictive control of a 3d crane system. In *2004 IEEE Conference on Robotics, Automation and Mechatronicss*, pages 583–587, Singapore, 2004.

J. Deng, V. M. Becerra, and R. Stobart. Input constraints handling in a mpc/feedback linearisation scheme. *Journal of Applied Mathematics and Computer Science*, 19(2):219–232, 2009.

F. Garces, V.M. Becerra, C. Kambhampati, and K. Warwick. *Strategies for feedback linearisation: A dynamic neural network approach*. Springer, London, 2003.

M. A. Henson and D. E. Seborg. Industrial applications of nonlinear control. *In Chemical Process Control - V. AIChE Symposium Series*, 93:46–59, 1997.

J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America-Biological Sciences*, 79(8):2554–2558, 1982.

A. Isidori. *Nonlinear control systems*. Springer, Berlin; New York, 2rd edition, 1995.

A. Isidori. *Nonlinear control systems II*. Springer, London, 1999.

C. Kravaris and Y. Arkun. Geometric nonlinear control - an overview. *In Chemical Process Control - CPC IV - AIChE Symposium Series*, pages 477–515, 1991.

C. Kravaris and J. C. Kantor. Geometric methods for nonlinear process-control .1. background. *Industrial & Engineering Chemistry Research*, 29(12):2295–2310, 1990a.

C. Kravaris and J. C. Kantor. Geometric methods for nonlinear process-control .2. controller synthesis. *Industrial & Engineering Chemistry Research*, 29(12):2310–2323, 1990b.

M. Kurtz, G. Y. Zhu, and M. Henson. Constrained output feedback control of a multivariable polymerization reactor. *IEEE Transactions on Control Systems Technology*, 8(1):87–96, 2000.

J. M. Maciejowski. *Predictive control with constraints*. Prentice Hall, 2002.

V. Nevistic and M. Morari. Robustness of mpc-based schemes for constrained control of nonlinear system. In *the 13th IFAC World Congress*, volume M, pages 25–30, San Franciso, USA, 1996.

H. Nijmeijer and A. J. van der Schaft. *Nonlinear dynamical control systems*. Springer-Verlag, New York, 1990.

D. E. Rumelhart and J. L. McClelland. *Parallel distributed processing : explorations in the microstructure of cognition*. MIT Press, Cambridge, Mass., 1986.

K. Schittkowski. Quadratic Programming implementation - C version translated from Fortran Version 1.4 (March 1987) - C translation 1992, modified by M.J.D. Powel (University of Cambridge), A.L. Tits, J.L. Zhou and C. Lawrence (University of Maryland). Mathematisches Institut, Universitaet Bayreuth, Germany, 1992.

J. J. E. Slotine and W. Li. *Applied nonlinear control*. Prentice Hall, Englewood Cliffs, New Jersey, 1991.

M. Vidyasagar. *Nonlinear Systems Analysis*. Prentice Hall, New Jersey,, 1993.