Int. J. Appl. Math. Comput. Sci., 2005, Vol. 15, No. 3, 369–381

STABILISING SOLUTIONS TO A CLASS OF NONLINEAR OPTIMAL STATE TRACKING PROBLEMS USING RADIAL BASIS FUNCTION NETWORKS

ZAHIR AHMIDA[*], ABDELFETTAH CHAREF[**]
VICTOR M. BECERRA[***]

[*] Department of Electrical Engineering, University of Skikda
Al-Hadaik Rd., P.B.: 26, Skikda 21000, Algeria
e-mail: zahirahmida@yahoo.fr

[**] Department of Electronics, Mentouri University
Zerzara, Ain-Bey Rd., Constantine 25000, Algeria
e-mail: Afcharef@yahoo.fr

[***] Department of Cybernetics, University of Reading
Whiteknights, Reading RG6 6AY, U.K.
e-mail: v.m.becerra@reading.ac.uk

A controller architecture for nonlinear systems described by Gaussian RBF neural networks is proposed. The controller is a stabilising solution to a class of nonlinear optimal state tracking problems and consists of a combination of a state feedback stabilising regulator and a feedforward neuro-controller. The state feedback stabilising regulator is computed on-line by transforming the tracking problem into a more manageable regulation one, which is solved within the framework of a nonlinear predictive control strategy with guaranteed stability. The feedforward neuro-controller has been designed using the concept of inverse mapping. The proposed control scheme is demonstrated on a simulated single-link robotic manipulator.

Keywords: nonlinear systems, optimal control, radial basis functions, neural networks, predictive control, feedforward control

1. Introduction

The model-based nonlinear tracking control problem has attracted considerable attention from the control research community. Amongst the various choices for modelling a nonlinear system, artificial neural networks have emerged as a powerful tool capable of approximating any continuous function or any input-output mapping of a nonlinear process, to any desired degree of accuracy (Hornik *et al.*, 1989), and, since the pioneering work by Narendra and Parthasarathy (1990), to use neural networks in nonlinear control, many different control strategies have been proposed. Predictive control, also called receding horizon control, is a model-based control technique that periodically uses a model of the controlled system to calculate the control action based on optimal input-state or input-output predictions over a time horizon (Becerra *et al.*, 1998; 1999; Eaton and Rawlings, 1992; Garcia *et al.*, 1989; Morari and Lee, 1999; Richalet, 1993; Richalet *et al.*, 1978).

This paper proposes to solve a nonlinear optimal state-tracking control problem by using a hybrid controller consisting of a nonlinear stabilising state feedback controller together with a feedforward neuro-controller. This class of optimal tracking problems was introduced by Park *et al.* (1996), who proposed a combination of feedback and feedforward neuro-controllers computed off-line and applied in parallel. The feedback controller generates the transient control input to stabilise the error dynamics while minimising a cost function. In order to meet this functional requirement by the feedback controller, in this paper it is proposed to transform the optimal state-tracking control problem into a simpler regulation one which can be solved on-line within the framework of a nonlinear receding horizon regulation scheme with guaranteed asymptotic stability. The feedforward neuro-controller is computed off-line through the inversion of nonlinear input-state mapping. Thus the philosophy of this hybrid control architecture is to allow the stabilising benefits of a nonlinear neural-model based predictive regulator to be combined with a forward neural controller to provide good tracking performance and computational simplicity.

The stability analysis of nonlinear model predictive controllers is now believed to have reached a relatively mature stage (Mayne *et al*, 2000). Some algorithms are now available which ensure closed-loop stability by embedding a terminal inequality constraint in the optimisation problem and employing a stabilising local linear state-feedback controller to determine terminal penalties and terminal constraints (Chen and Allgöwer, 1998; Chen and Shaw, 1982; De Nicolao *et al.*, 1997; Keerthi and Gilbert, 1988; Magni *et al.*, 2001; Mayne and Michalska, 1990; Michalska and Mayne, 1993; Parisini and Zoppoli, 1995; Parisini *et al.*, 1998). The nominal nonlinear system in this paper is described by Gaussian radial basis function neural networks (GRBFNN). An interesting feature of the proposed GRBFNN description is that it allows for an easy transformation of the nonlinear tracking problem into an associated regulation problem. The stabilizing state feedback regulator which asymptotically steers the state-error to the origin is computed using the method of Parisini *et al.* (1998) for deriving stabilising nonlinear predictive regulators. The feedforward control signal that maintains the system state at a reference value is generated by an inverse GRBFNN model of the nonlinear system under control.

The paper is organised as follows: Section 2 presents the description of the nominal nonlinear system and the formulation of the tracking problem. Section 3 gives a description of the hybrid predictive-neural controller architecture, recalls the stabilising nonlinear predictive regulator theorem of Parisini *et al.*, and presents the training algorithm for the feedforward neuro-controller. A Gaussian radial basis function neural network model for the state-error dynamics is derived, and a summary of the control algorithm is presented in Section 4. A simulation example using a single-link robotic manipulator to illustrate the effectiveness of the proposed controller is included in Section 5, and some conclusions are drawn in Section 6.

2. System Description and Control Problem Definition

2.1. System Description

The nominal nonlinear system considered in this paper is described by the following discrete-time equations:

$$\begin{aligned} \boldsymbol{x}(t+1) &= \boldsymbol{f}\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big), \quad t = 0, 1, 2, \dots, \\ \boldsymbol{y}(t) &= \boldsymbol{h}\big(\boldsymbol{x}(t)\big), \end{aligned} \tag{1}$$

where $\boldsymbol{x}(t) \in X \subset \mathbb{R}^n$ is the state vector, $\boldsymbol{u}(t) \in U \subset \mathbb{R}^m$ represents the control vector and $y(t) \in Y \subset \mathbb{R}^r$ is the output vector. The mappings $\boldsymbol{f}(\cdot,\cdot)$ and $\boldsymbol{h}(\cdot)$ are C^1 functions of their arguments, and the regions X, U and

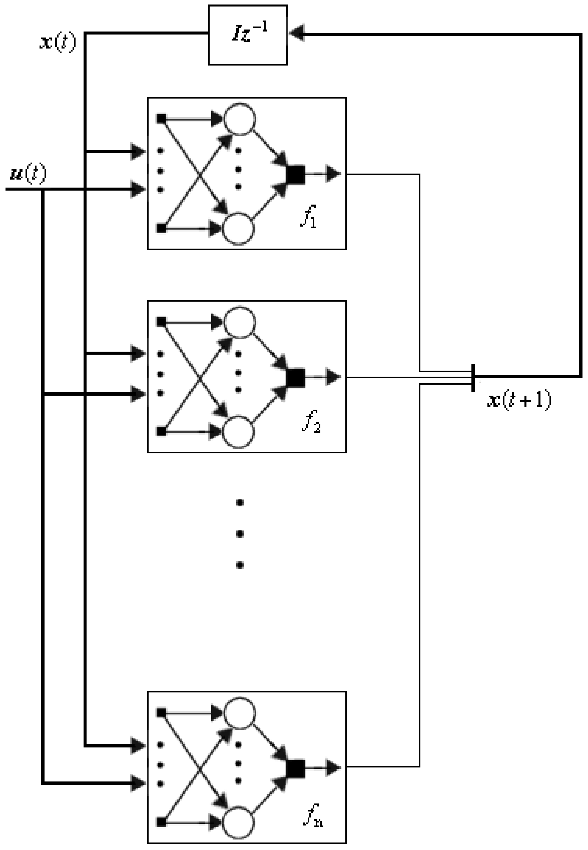


Fig. 1. Nominal GRBFNN model structure.

Y are compact subsets of $\mathbb{R}^n$, $\mathbb{R}^m$ and $\mathbb{R}^r$, respectively, each containing the origin in its interior.

Moreover, the n elements of the input-state vector mapping $\boldsymbol{f}(\cdot,\cdot)$ are assumed to be outputs of single-hidden-layer feedforward neural networks with the input vector $(\boldsymbol{x}(t), \boldsymbol{u}(t))$ as illustrated in Fig. 1. The output of the i-th network can be expressed as

$$\begin{aligned} & f_i\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) \\ & \quad = q_{i0} + \sum_{k=1}^{K_i} q_{ik} \Phi_{ik}\left(\left\|\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) - \boldsymbol{c}_{ik}\right\|, \sigma_{ik}\right), \\ & \qquad\qquad t = 0, 1, \dots \end{aligned} \tag{2}$$

The functions Φ_{ik} are Gaussian radial basis functions (GRBF) defined as

$$\Phi_{ik}\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) = \exp\left(-\frac{\left\|\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) - \boldsymbol{c}_{ik}\right\|^2}{\sigma_{ik}^2}\right), \tag{3}$$

where $\boldsymbol{c}_{ik} \in \mathbb{R}^{n+m}$ are the GRBF centres, σ_{ik} denote the widths and $\|\cdot\|$ denotes the Euclidean norm. The number of hidden units in the i-th network is K_i and the connecting weights are q_{ik}, with q_{i0} as a bias term.

It is further assumed that:

A1. $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a one-to-one mapping of state equations.

A2. For any equilibrium state $(\boldsymbol{x}_e, \boldsymbol{u}_e) \in X \times U$, the linear system $\boldsymbol{x}_l(t+1) = \boldsymbol{A}\boldsymbol{x}_l(t) + \boldsymbol{B}\boldsymbol{u}(t)$ obtained via the linearisation of the system (1) in the neighbourhood of $(\boldsymbol{x}_e, \boldsymbol{u}_e)$, i.e.,

$$\boldsymbol{A} = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_e\\ \boldsymbol{u}=\boldsymbol{u}_e}}, \quad \boldsymbol{B} = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_e\\ \boldsymbol{u}=\boldsymbol{u}_e}}$$

is stabilisable.

2.2. Tracking Problem

Given the system (1)–(3), a reference state trajectory $\boldsymbol{x}_d(t) \in Y$, an initial state $\boldsymbol{x}_0 = \boldsymbol{x}(t_0)$, we wish to determine a control law $\boldsymbol{u}(t) = \boldsymbol{\kappa}_{\rm tr}(t)$ such that for all $t \ge t_0$ the following conditions are satisfied:

1. The quadratic cost function

$$J = \sum_{i=t}^{t+N_c} \big(\boldsymbol{x}(i) - \boldsymbol{x}_d(i)\big)^T \boldsymbol{Q}\big(\boldsymbol{x}(i) - \boldsymbol{x}_d(i)\big) + \big(\boldsymbol{u}(i) - \boldsymbol{u}_d(i)\big)^T \boldsymbol{R}\big(\boldsymbol{u}(i) - \boldsymbol{u}_d(i)\big) \tag{4}$$

is minimised, where $\boldsymbol{Q} > \boldsymbol{0},\ \boldsymbol{R} > \boldsymbol{0}$.

2. The nominal system $\boldsymbol{x}(t+1) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t))$ is asymptotically stable.

3. $\lim_{t\to\infty} |\boldsymbol{x}(t) - \boldsymbol{x}_d(t)| = 0$.

In the formulation of the tracking problem given above, the target state $\boldsymbol{x}_d$ can undergo step-like changes between constant steady-state values, $\boldsymbol{u}_d$ is the nominal input corresponding to $\boldsymbol{x}_d$, and N_c a positive integer known as the prediction horizon.

Now, define the following vector shifts:

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_d, \tag{5a}$$

$$\tilde{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{u}_d, \tag{5b}$$

$$\tilde{\boldsymbol{y}} = \boldsymbol{y} - \boldsymbol{y}_d. \tag{5c}$$

Using these equations, the quadratic cost function in (4) can be rewritten as

$$\boldsymbol{J} = \sum_{i=t}^{t+N_c} \tilde{\boldsymbol{x}}(i)^T \boldsymbol{Q}\tilde{\boldsymbol{x}}(i) + \tilde{\boldsymbol{u}}(i)^T \boldsymbol{R}\tilde{\boldsymbol{u}}(i), \tag{6}$$

where $\tilde{\boldsymbol{x}} \in \tilde{X}$, $\tilde{\boldsymbol{u}} \in \tilde{U}$ and $\tilde{\boldsymbol{y}} \in \tilde{Y}$. The sets $\tilde{X}$, $\tilde{U}$ and $\tilde{Y}$ are compact subsets of $\mathbb{R}^n$, $\mathbb{R}^m$ and $\mathbb{R}^r$, respectively, each having the origin as an internal point.

Using (1), (5a) and (5b), the state error can be written as

$$\begin{aligned}\tilde{\boldsymbol{x}}(t+1) &= \boldsymbol{f}\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) - \boldsymbol{f}\big(\boldsymbol{x}_d(t), \boldsymbol{u}_d(t)\big)\\ &= \boldsymbol{f}\big(\tilde{\boldsymbol{x}}(t) + \boldsymbol{x}_d(t), \tilde{\boldsymbol{u}}(t) + \boldsymbol{u}_d(t)\big)\\ &\quad - \boldsymbol{f}\big(\boldsymbol{x}_d(t), \boldsymbol{u}_d(t)\big)\\ &= \tilde{\boldsymbol{f}}\big(\tilde{\boldsymbol{x}}(t), \tilde{\boldsymbol{u}}(t)\big).\end{aligned} \tag{7}$$

The linearisation of this error around the origin leads to

$$\tilde{\boldsymbol{x}}_l(t+1) = \tilde{\boldsymbol{A}}\tilde{\boldsymbol{x}}_l(t) + \tilde{\boldsymbol{B}}\tilde{\boldsymbol{u}}(t),$$

with

$$\tilde{\boldsymbol{A}} = \left.\frac{\partial \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{u}})}{\partial \tilde{\boldsymbol{x}}}\right|_{\substack{\tilde{\boldsymbol{x}}=0\\ \tilde{\boldsymbol{u}}=0}}, \quad \tilde{\boldsymbol{B}} = \left.\frac{\partial \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{u}})}{\partial \tilde{\boldsymbol{u}}}\right|_{\substack{\tilde{\boldsymbol{x}}=0\\ \tilde{\boldsymbol{u}}=0}}.$$

Letting

$$\tilde{\boldsymbol{f}}\big(\tilde{\boldsymbol{x}}(t), \tilde{\boldsymbol{u}}(t)\big) = \boldsymbol{f}\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) - \boldsymbol{f}\big(\boldsymbol{x}_d(t), \boldsymbol{u}_d(t)\big), \tag{8}$$

with $\boldsymbol{f}(\boldsymbol{x}_d(t), \boldsymbol{u}_d(t))$ being treated as an offset, the differentiation of both sides of (8) with respect to $x(t)$ and $u(t)$ at the target reference point $(\boldsymbol{x}_d, \boldsymbol{u}_d)$ yields

$$\left.\frac{\partial \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{u}})}{\partial \tilde{\boldsymbol{x}}}\right|_{\substack{\tilde{\boldsymbol{x}}=0\\ \tilde{\boldsymbol{u}}=0}} = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_d\\ \boldsymbol{u}=\boldsymbol{u}_d}}, \tag{9a}$$

$$\left.\frac{\partial \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{u}})}{\partial \tilde{\boldsymbol{u}}}\right|_{\substack{\tilde{\boldsymbol{x}}=0\\ \tilde{\boldsymbol{u}}=0}} = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_d\\ \boldsymbol{u}=\boldsymbol{u}_d}}. \tag{9b}$$

From Assumption A2 it follows that the stabilisability of the linearised system $(\boldsymbol{A}, \boldsymbol{B})$ at the reference point $(\boldsymbol{x}_d, \boldsymbol{u}_d)$ implies that of the linearised state-error system $(\tilde{\boldsymbol{A}}, \tilde{\boldsymbol{B}})$ at the origin.

3. Controller Architecture

To solve the above control problem, a hybrid control architecture consisting of a nonlinear stabilising state feedback controller together with a feedforward controller is adopted. To this end, the optimal tracking problem defined by (4) is converted into a simpler nonlinear regulation problem involving a state error described by a Gaussian radial basis function network. The nonlinear regulation problem can be solved within the framework of a nonlinear receding horizon regulation scheme with guaranteed asymptotic stability to obtain a stabilising state feedback controller. The feedforward controller is computed off-line through the inversion of the mapping $\boldsymbol{f}(\cdot,\cdot)$. Thus the philosophy of this hybrid control architecture is to allow the stabilising benefits of a nonlinear neural-model-based predictive regulator to be combined with a forward neural controller to provide good tracking performance and computational simplicity.

3.1. State-Feedback Receding Horizon Regulator

In the case of a linear system, the solution of the optimal regulator problem defined by (6) with $N_c = \infty$ is a stabilising linear state feedback control law that steers the state error $\tilde{\boldsymbol{x}}$ to zero, and is given by

$$\tilde{\boldsymbol{u}}(t) = \boldsymbol{K}\tilde{\boldsymbol{x}}(t), \tag{10}$$

where $\boldsymbol{K}$ is the feedback gain matrix obtained by solving an algebraic matrix Riccati equation (Kwakernaak and Sivan, 1972).

For a nonlinear system, however, the following finite-horizon optimal control problem is defined: Minimise the cost function

$$\begin{aligned} &J\big(\tilde{\boldsymbol{x}}(t), t, \tilde{\boldsymbol{u}}_{t,t+N_c+1}, N_c\big) \\ &\quad = \sum_{i=t}^{t+N_c-1} \Big\{\tilde{\boldsymbol{x}}(i)^T\boldsymbol{Q}\tilde{\boldsymbol{x}}(i) + \tilde{\boldsymbol{u}}(i)^T\boldsymbol{R}\tilde{\boldsymbol{u}}(i)\Big\} \\ &\qquad + V_f\big(\tilde{\boldsymbol{x}}(t+N_c)\big) \end{aligned} \tag{11}$$

with respect to $\tilde{\boldsymbol{u}}_{t,t+N-1} = \{\tilde{\boldsymbol{u}}(t), \tilde{\boldsymbol{u}}(t+1), \dots, \tilde{\boldsymbol{u}}(t+N-1)\}$, subject to (1). Notice that (1) is assumed to be a GRBF network. The terminal cost function $V_f(\tilde{\boldsymbol{x}}(t+N_c))$ bounds the infinite horizon cost of the nonlinear system beyond the prediction horizon N_c (Chen and Allgöwer, 1998; Parisini and Zoppoli, 1995):

$$\begin{aligned} &V_f\big(\tilde{\boldsymbol{x}}(t+N_c)\big) \\ &\quad \geq \sum_{i=t+N_c}^{\infty} \tilde{\boldsymbol{x}}(i)^T\boldsymbol{Q}\tilde{\boldsymbol{x}}(i) + \tilde{\boldsymbol{u}}(i)^T\boldsymbol{R}\tilde{\boldsymbol{u}}(i). \end{aligned} \tag{12}$$

The receding horizon control law $\boldsymbol{\kappa}_{fb}(\tilde{\boldsymbol{x}}(t)) = \tilde{\boldsymbol{u}}(t)$ derived by solving the finite-horizon optimal control problem (11) is required to drive the state error $\tilde{\boldsymbol{x}}(t)$ to the origin asymptotically, thus ensuring the stability of (1).

The stability analysis of this class of optimal regulators has been carried out by a number of authors and the closed-loop stability conditions have been established. In this paper, stability is achieved by ensuring the satisfaction of the assumptions and conditions required by Parisini and Zoppoli (1995) and Parisini *et al.* (1998).

Let the terminal cost function be $V_f(\tilde{\boldsymbol{x}}(t+N_c)) = a\|\tilde{\boldsymbol{x}}(t+N_c)\|^2_{\boldsymbol{P}}$, where $a \in \mathbb{R}$ is a positive scalar and $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ signifies a *positive-definite symmetric matrix*. The objective function in (11) becomes

$$\begin{aligned} &J(\tilde{\boldsymbol{x}}(t), t, \tilde{\boldsymbol{u}}_{t,t+N_c+1}, N_c) \\ &\quad = \sum_{i=t}^{t+N_c-1} \Big\{\tilde{\boldsymbol{x}}(i)^T Q\tilde{\boldsymbol{x}}(i) + \tilde{\boldsymbol{u}}(i)^T\boldsymbol{R}\tilde{\boldsymbol{u}}(i)\Big\} \\ &\qquad + a\,\|\tilde{\boldsymbol{x}}(t+N_c)\|^2_{\boldsymbol{P}}\,. \end{aligned} \tag{13}$$

Assume that the following condition is satisfied:

A3. There is a compact set $\tilde{X}_0 \subset \tilde{X}$ containing the origin in its interior, with the property that there exists an integer $M_c \geq 1$ such that there is a sequence of admissible control vectors $\{\tilde{u}(i) \in \tilde{U},\ i = t, \dots, t+M_c-1\}$ that yields an admissible state trajectory $\{\tilde{\boldsymbol{x}}(i) \in \tilde{X},\ i = t, \dots, t+M_c\}$ ending in the origin of the state space (i.e., $\tilde{\boldsymbol{x}}(t+M_c) = 0$) for any initial state error $\tilde{\boldsymbol{x}}(t) \in \tilde{X}_0$.

Denote by

$$\begin{aligned} &J^0\big(\tilde{\boldsymbol{x}}(t), N_c, a, \boldsymbol{P}\big) \\ &\quad = J\big(\tilde{\boldsymbol{x}}(t), \tilde{\boldsymbol{u}}^0_{t,t+N_c-1}, N_c, a, \boldsymbol{P}\big) \\ &\quad = \sum_{i=t}^{t+N_c-1} \Big\{\tilde{\boldsymbol{x}}^0(i)^T\boldsymbol{Q}\tilde{\boldsymbol{x}}^0(i) + \tilde{\boldsymbol{u}}^0(i)^T\boldsymbol{R}\tilde{\boldsymbol{u}}^0(i)\Big\} \\ &\qquad + a\,\big\|\tilde{\boldsymbol{x}}^0(t+N_c)\big\|^2_{\boldsymbol{P}} \end{aligned}$$

the cost corresponding to the optimal N-stage trajectory starting from $\tilde{\boldsymbol{x}}(t)$, with $\tilde{\boldsymbol{u}}^0_{t,t+N_c-1}$ being the optimal solution sequence of the receding horizon regulator problem characterised by the control horizon N_c.

Theorem 1. (Parisini and Zoppoli, 1998) *If Assumptions A2 and A3 are satisfied, then there exist a positive scalar a^* and a positive-definite symmetric matrix $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ such that, for any $N_c \geq M_c$ and any $a \in \mathbb{R}$, $a \geq a^*$, the following properties hold:*

(a) *The predictive control law stabilises asymptotically the origin, which is an equilibrium point of the resulting closed-loop system.*

(b) *There exists a positive scalar β such that the compact set*

$$\tilde{W}(N_c, a, \boldsymbol{P}) = \{\tilde{\boldsymbol{x}} \in \tilde{X} : J^0(\tilde{\boldsymbol{x}}, N_c, a, \boldsymbol{P}) \leq \beta\}$$

is an invariant subset of $\tilde{X}_0$ and a domain of attraction for the origin, i.e., for any $\tilde{\boldsymbol{x}}(t) \in \tilde{W}(N_c, a, \boldsymbol{P})$, the state trajectory generated by the predictive regulator remains entirely contained in $\tilde{W}(N_c, a, \boldsymbol{P})$ and converges to the origin.

3.2. Feedforward Neuro-Controller

In designing a nonlinear tracking controller, it is also required that the system state be maintained at a reference state value by producing the corresponding reference input. For this purpose, assuming that $\boldsymbol{f}(\cdot,\cdot)$ is a one-to-one mapping (c.f. Assumption A1), a neural network

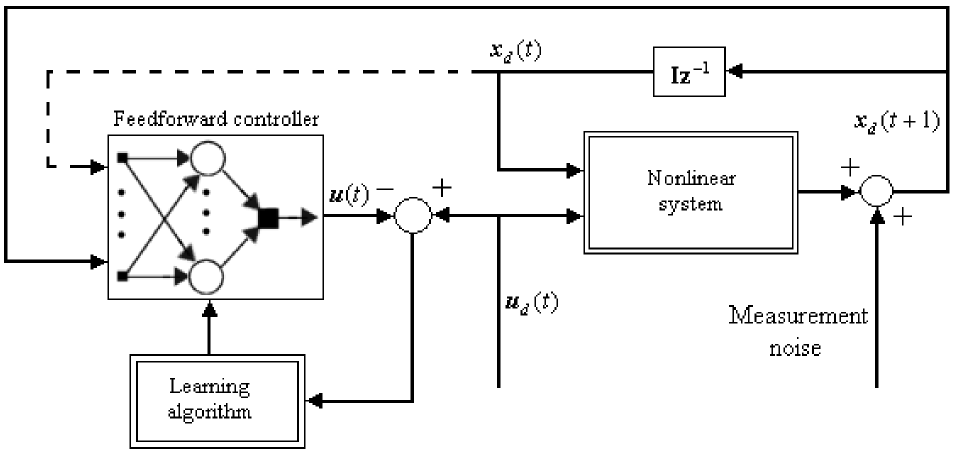


Fig. 2. Feedforward controller training structure—a direct method.

is trained to represent the inverse dynamics of the system (1).

When the system (1) reaches its steady state (i.e., $\boldsymbol{x}(t) = \boldsymbol{x}_d(t)$), the receding horizon component of the controller vanishes as the state error reaches the origin, and thus (1) can be written as

$$\boldsymbol{x}_d(t+1) = \boldsymbol{f}\big(\boldsymbol{x}_d(t), \boldsymbol{u}_d(t)\big). \tag{14}$$

Starting from (14), which relates a reference state to the corresponding reference input, the nonlinear relation describing the system's inverse can be written as

$$\boldsymbol{u}_d(t) = \boldsymbol{f}^{-1}\big(\boldsymbol{x}_d(t), \boldsymbol{x}_d(t+1)\big). \tag{15}$$

Here, the inverse function $\boldsymbol{f}^{-1}$ requires knowledge of the current and next reference state vectors to generate the current reference input.

Using Gaussian RBF neural networks to design the feedforward controller, the prediction of the j-th element of the reference input vector can then be written as follows:

$$\begin{aligned} u_{dj}(t) &= f_j^{-1}\big(\boldsymbol{x}_d(t+1), \boldsymbol{x}_d(t)\big) = w_{j0} \\ &\quad + \sum_{k=1}^{K_j} w_{jk}\Phi_{jk}\Big(\left\|(\boldsymbol{x}_d(t+1), \boldsymbol{x}_d(t)) - \boldsymbol{\mu}_{jk}\right\|, \delta_{jk}\Big), \\ &\quad j = 1, 2, \dots, m, \end{aligned} \tag{16}$$

with the Gaussian radial basis functions defined as

$$\begin{aligned} &\Phi_{jk}\big(\boldsymbol{x}_d(t+1), \boldsymbol{x}_d(t)\big) \\ &\quad = \exp\left(-\frac{\left\|\big(\boldsymbol{x}_d(t+1), \boldsymbol{x}_d(t)\big) - \boldsymbol{\mu}_{jk}\right\|^2}{\delta_{jk}^2}\right), \end{aligned} \tag{17}$$

where $\boldsymbol{\mu}_{jk} \in \mathbb{R}^{2n}$ are the GRBF centres and $\delta_{jk} \in \mathbb{R}$ stand for the widths. The number of hidden units in the j-th network is K_j and the connecting weights are w_{jk}, with w_{j0} as a bias term.

The training of the inverse model can be done using the direct inverse modelling approach (Hunt *et al.*, 1992), where a synthetic input signal is applied to the nonlinear system. Then the corresponding states are measured and used as training data (Fig. 2).

3.3. Training Algorithm

The training algorithm for the feedforward controller, which can be used to choose the number and distribution of the GRBFs centres and to adjust the weights and widths, is one of the numerous algorithms available. In this paper an algorithm similar to the resource allocating network with extended Kalman filtering (RANEKF) is used (Kadirkamanathan and Niranjan, 1993; Yingwei *et al.*, 1997).

The algorithm uses a set of N state-input measurements collected from the response of the nominal system for various reference inputs. At the k-th training iteration, the algorithm receives the vector $(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k))$ as an input and the corresponding reference input $\boldsymbol{u}_d(k)$ as a target. The algorithm starts with a blank network with no Gaussian units. When training the j-th input $u_j(\cdot)$, the bias term is initialised to the first target value $u_{d_j}(1)$, and the first centre is selected randomly from the training data. As more data vectors are received, new centres are allocated based on the novelty of the data. The following two conditions must be satisfied before the input vector $(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k))$ is taken as the new centre:

$$\left\|\big(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k)\big) - \boldsymbol{\mu}_{j,nr}\right\| > \varepsilon_j(k), \tag{18}$$

$$e_{jr}(k) = \left|\frac{u_{dj}(k) - f_j^{-1}\big(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k)\big)}{u_{dj}(k)}\right| > \eta_j, \tag{19}$$

where $\boldsymbol{\mu}_{j,nr}$ is the nearest centre to $(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k))$ in the input space, e_{jr} is the relative error in the network output, and $\varepsilon_j(k)$ and η_j are appropriately chosen positive thresholds. The value of $\varepsilon_j(k)$ is iteratively reduced from a maximum value $\varepsilon_{\max}$ until it reaches a minimum allowed value $\varepsilon_{\min}$:

$$\varepsilon_j(k) = \max\{\varepsilon_{\max}\gamma^k, \varepsilon_{\min}\}, \quad 0 < \gamma < 1. \tag{20}$$

When a new centre is selected, the parameters of the associated Gaussian unit are assigned as follows:

$$w_{j,K_j+1} = u_{dj}(k) - f_j^{-1}\big(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k)\big), \tag{21}$$

$$\boldsymbol{\mu}_{j,K_j+1} = \big(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k)\big), \tag{22}$$

$$\sigma_{j,K_j+1} = \rho\,\big\|\big(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k)\big) - \boldsymbol{\mu}_{j,nr}\big\|, \tag{23}$$

where $\rho > 0$ is an overlap factor.

When the observation $\{(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k)), u_{dj}(k)\}$ does not satisfy the novelty criteria, the extended Kalman filter is used to update the weights and widths of the network described by the parameter vector

$$\boldsymbol{\omega}_j = [w_{j0}, w_{j1}, \dots, w_{jK_j}, \delta_{j1}, \delta_{j2}, \dots, \delta_{jK_j}]^T$$

as follows:

$$\begin{aligned}\boldsymbol{\omega}_j(k) &= \boldsymbol{\omega}_j(k-1)\\ &\quad + \boldsymbol{g}_j(k)\Big(u_{dj}(k) - f_j^{-1}\big(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k)\big)\Big),\end{aligned} \tag{24}$$

where $\boldsymbol{g}_j(k)$ is the Kalman gain vector given by

$$\boldsymbol{g}_j(k) = \boldsymbol{S}_j(k\text{–}1)\boldsymbol{a}_j(k)\big[\boldsymbol{R}_j(k) + \boldsymbol{a}_j^T(k)\boldsymbol{S}_j(k\text{–}1)\boldsymbol{a}_j(k)\big]^{-1}, \tag{25}$$

$\boldsymbol{a}_j(k)$ being the gradient vector of the following form:

$$\boldsymbol{a}_j(k) = \nabla_{\omega_j} f_j^{-1}\big(\big(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k)\big)\big). \tag{26}$$

Here $\boldsymbol{R}_j(k)$ is the variance of the measurement noise and $\boldsymbol{S}_j(k)$ is the error covariance matrix which is updated in accordance with

$$\boldsymbol{S}_j(k) = \big[\boldsymbol{I} - \boldsymbol{g}_j(k)\boldsymbol{a}_j^T(k)\big]\boldsymbol{S}_j(k-1) + Q_0\boldsymbol{I}. \tag{27}$$

The error covariance matrix $\boldsymbol{S}_j(k)$ is an $s \times s$ positive definite symmetric matrix, where s is the number of the parameters being updated. Q_0 is a scalar that determines the allowed random step in the direction of the gradient vector. Whenever a new Gaussian unit is allocated, the matrix $\boldsymbol{S}_j(k)$ is augmented as follows:

$$\boldsymbol{S}_j(k) = \begin{pmatrix} \boldsymbol{S}_j(k-1) & 0 \\ 0 & \boldsymbol{S}_0 I \end{pmatrix}. \tag{28}$$

This training algorithm can be summarised as follows:

```
Step 0.
  0.1 Set $k = 1, \eta_j, \varepsilon_{\max}, \varepsilon_{\min}, \gamma, \rho, N, \boldsymbol{R}_0, \boldsymbol{S}_0, Q_0$.
  0.2 Obtain the first observation $\{(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k)), u_{dj}(k)\}$ and set $w_{j0} = u_{dj}(1)$.
  0.3 Select a random integer $k_0$, $1 \le k_0 \le N$, and set $\boldsymbol{\mu}_{j,1} = (\boldsymbol{x}_d(k_0+1), \boldsymbol{x}_d(k_0))$.
  0.4 Set $K_j = 1$.

Step 1.
  1.1 Obtain a new observation $\{(\boldsymbol{x}_d(k+1), \boldsymbol{x}_d(k)), u_{dj}(k)\}$.
  1.2 Compute $\varepsilon_j(k)$ using (20).

Step 2.
    2.1 If (18) and (19) do
        2.1.1 Allocate a new Gaussian unit as in (21)–(23).
        2.1.2 Augment $\boldsymbol{S}_j(k)$ as in (28).
        2.1.3 Set $K_j = K_j + 1$.
    Else do
        2.1.4 Compute vector $\boldsymbol{a}_j(k)$ using (26).
        2.1.5 Compute the Kalman gain $\boldsymbol{g}_j(k)$ as in (25).
        2.1.6 Update the parameter vector $\boldsymbol{\omega}_j(k)$ using (24).
        2.1.7 Update $\boldsymbol{S}_j(k)$ using (27).

Step 3.
    Set $k = k + 1$.
    If $k \le N$ do
        go to Step 1
    else stop iteration
```

4. Controller Implementation

This section shows how a state-error expression can be explicitly derived in the form of a GRBF neural network, thus allowing the nonlinear tracking problem (4) to be converted into a nonlinear regulation problem which can be treated within the stabilising framework of Theorem 1. Secondly, an implementation of this hybrid predictive-neural tracking controller is presented.

4.1. Tracking–Error Model

By using (2) and the definition of the Gaussian RBF in (3), the i-th element of the state-error vector (7) can be written down as follows:

$$\begin{aligned}\tilde{x}_i(t+1) &= f_i\big(\boldsymbol{x}(t),\boldsymbol{u}(t)\big) - f_i\big(\boldsymbol{x}_d(t),\boldsymbol{u}_d(t)\big)\\ &= \sum_{k=1}^{K_i} q_{ik}\Big[\exp\Big(-\frac{\|\big(\boldsymbol{x}(t),\boldsymbol{u}(t)\big)-\boldsymbol{c}_{ik}\|^2}{\sigma_{ik}^2}\Big)\\ &\quad - \exp\Big(-\frac{\|\big(\boldsymbol{x}_d(t),\boldsymbol{u}_d(t)\big)-\boldsymbol{c}_{ik}\|^2}{\sigma_{ik}^2}\Big)\Big] \qquad (29)\end{aligned}$$

by manipulating the term $\|(\boldsymbol{x}(t),\boldsymbol{u}(t))-\boldsymbol{c}_{ik}\|^2$ in (29). Adding and subtracting the column vector $(\boldsymbol{x}_d(t),\boldsymbol{u}_d(t))$ yield

$$\begin{aligned}&\big\|\big(\boldsymbol{x}(t),\boldsymbol{u}(t)\big)-\boldsymbol{c}_{ik}\big\|^2\\ &\quad= \big\|\big(\boldsymbol{x}(t),\boldsymbol{u}(t)\big)-\big(\boldsymbol{x}_d(t),\boldsymbol{u}_d(t)\big)\\ &\qquad + \big(\boldsymbol{x}_d(t),\boldsymbol{u}_d(t)\big)-\boldsymbol{c}_{ik}\big\|^2\\ &\quad= \big\|\big(\tilde{\boldsymbol{x}}(t),\tilde{\boldsymbol{u}}(t)\big)-\big(\boldsymbol{c}_{ik}-\big(\boldsymbol{x}_d(t),\boldsymbol{u}_d(t)\big)\big)\big\|^2. \qquad (30)\end{aligned}$$

Let $\tilde{\boldsymbol{c}}_{ik}(t)$ be the shifted centre defined by

$$\tilde{c}_{ik}(t) = \boldsymbol{c}_{ik} - \big(\boldsymbol{x}_d(t),\boldsymbol{u}_d(t)\big). \qquad (31)$$

Then the dynamics of the i-th element of the state-error vector can be written as

$$\begin{aligned}\tilde{x}_i(t+1) &= \tilde{f}_i\big(\tilde{\boldsymbol{x}}(t),\tilde{\boldsymbol{u}}(t)\big)\\ &= \sum_{k=1}^{K_i} q_{ik}\Big[\exp\Big(-\frac{\|\big(\tilde{\boldsymbol{x}}(t),\tilde{\boldsymbol{u}}(t)\big)-\tilde{\boldsymbol{c}}_{ik}(t)\|^2}{\sigma_{ik}^2}\Big)\\ &\quad - \exp\Big(-\frac{\|\tilde{\boldsymbol{c}}_{ik}(t)\|^2}{\sigma_{ik}^2}\Big)\Big] \qquad (32)\end{aligned}$$

for $i=1,2,\dots,n$. Thus for every reference state value $\boldsymbol{x}_d(t)$, a nonlinear state-error model represented by the nonlinear vector mapping $\tilde{\boldsymbol{f}} = [\tilde{f}_1,\tilde{f}_2,\dots,\tilde{f}_n]^T$ is explicitly expressed based on the parameters of the GRBF neural network nominal system. From (32), it is possible to conclude that the state-error model is a GRBF neural network with centres $\tilde{\boldsymbol{c}}_{ik}(t)$, widths σ_{ik} and bias terms

$$\tilde{q}_{i0} = -\sum_{k=1}^{K_i} q_{ik}\exp\left(-\frac{\|\tilde{\boldsymbol{c}}_{ik}(t)\|^2}{\sigma_{ik}^2}\right). \qquad (33)$$

4.2. Architecture of the GRBF-Based Predictive Tracking Controller

The general architecture of the stabilising hybrid predictive-neural tracking controller based on GRBF neural networks is illustrated in Fig. 3. The state-error model generator takes the current values of the state reference vector $\boldsymbol{x}_d(t)$ and the output $\boldsymbol{u}_d(t)$ of the feedforward controller as its input to produce the associated state-error model parameters, namely the shifted centres $\tilde{\boldsymbol{c}}_{ik}$. Then the error model is used by the optimisation function to generate the stabilising optimal predictive control component $\tilde{\boldsymbol{u}}(t)$ which ensures closed-loop stability.

The minimisation of the objective function $J(\tilde{\boldsymbol{x}}(t),\tilde{\boldsymbol{u}}_{t,t+N_c-1},N_c,a,\boldsymbol{P})$ at any sampling time t, involves the use of a predicted state-tracking-error sequence

$$\tilde{\boldsymbol{x}}_{t,t+N_c} = \big\{\tilde{\boldsymbol{x}}(t|t),\tilde{\boldsymbol{x}}(t+1|t),\dots,\tilde{\boldsymbol{x}}(t+N_c|t)\big\}$$

over the prediction horizon N_c to generate the optimal predicted control sequence

$$\tilde{\boldsymbol{u}}_{t,t+N_c-1} = \big\{\tilde{\boldsymbol{u}}(t|t),\tilde{\boldsymbol{u}}(t+1|t),\dots,\tilde{\boldsymbol{u}}(t+N_c-1|t)\big\}.$$

The vector $\tilde{\boldsymbol{x}}(t|t)$ denotes the current state error and is given by

$$\tilde{\boldsymbol{x}}(t|t) = \boldsymbol{x}(t) - \boldsymbol{x}_d(t). \qquad (34)$$

Using the recursive technique and the *one*-step-ahead predictor expression in (32), the d-step-ahead recursive predictor of the state-tracking-error at time t is

$$\begin{aligned}\tilde{x}_i(t+d) &= \sum_{k=1}^{K_i} q_{ik}\Bigg\{\exp\\ &\Big(-\frac{\big\|\big(\tilde{\boldsymbol{x}}(t+d-1|t),\tilde{\boldsymbol{u}}(t+d-1|t)\big)-\tilde{\boldsymbol{c}}_{ik}(t+d-1|t)\big\|^2}{\sigma_{ik}^2}\Big)\\ &\quad - \exp\Big(-\frac{\big\|\tilde{\boldsymbol{c}}_{ik}(t+d-1|t)\big\|^2}{\sigma_{ik}^2}\Big)\Bigg\} \qquad (35)\end{aligned}$$

for $d=1,2,\dots,N_c$ and $i=1,2,\dots,n$. The set of future shifted centres is obtained by transforming (31) into

$$\begin{aligned}&\tilde{c}_{ik}(t+d-1|t)\\ &\quad= \boldsymbol{c}_{ik} - \big(\boldsymbol{x}_d(t+d-1|t),\boldsymbol{u}_d(t+d-1|t)\big). \qquad (36)\end{aligned}$$

Clearly, the future shifted centres depend on the future reference state values and future reference inputs. The future reference inputs in (36) are generated, at each sample time, by the feedforward controller. From (35) and (36) it is noted that information about the present and future values of the reference state trajectory is embedded into the state-error predictors through the shift of the GRBF centres in the input-state space (Fig. 3).

Unlike other nonlinear model based predictive controllers where the model of the controlled process is directly used to compute the control profile based on optimal input-state predictions over a prediction horizon,

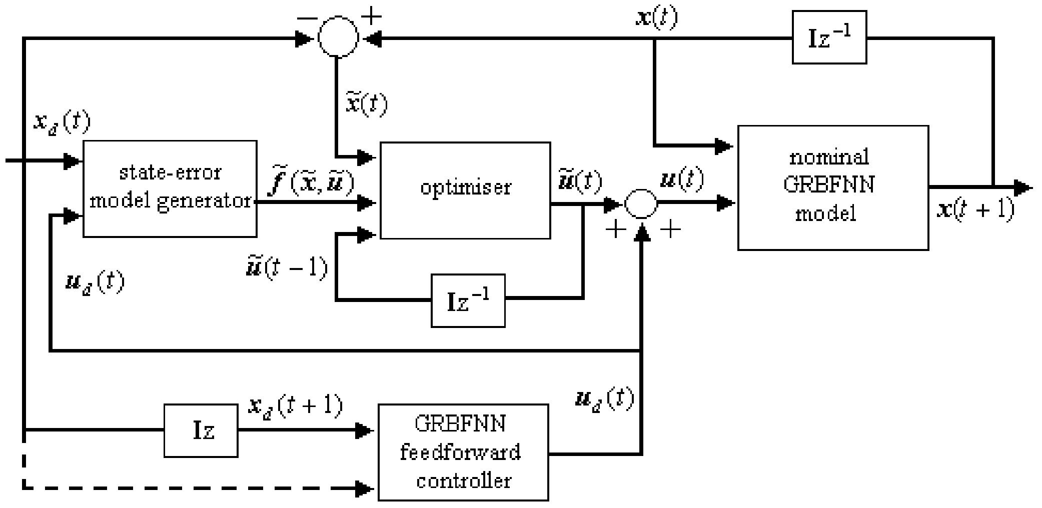


Fig. 3. Block diagram for the nonlinear hybrid predictive-neural tracking controller.

here the nominal process model is used to generate another model that describes the dynamics of the tracking error for a particular reference state or set-point. It is this tracking-error model that is used to predict future tracking errors and minimise a performance objective function to yield the predictive control action that steers the state-tracking-error to zero. The implementation of the hybrid predictive-neural tracking control algorithm can be summarised as follows:

Step 0.

0.1 Load the nominal system and its inverse.

0.2 Set the desired state trajectory.

0.3 Set N_c and $\boldsymbol{x}_0$.

0.4 Set the initial guess of the predicted control sequence to zero.

Step 1.

1.1 Obtain the future reference state vector sequence

$$\boldsymbol{x}_{d|_{t,t+N_c}} = \big\{\boldsymbol{x}_d(t|t), \boldsymbol{x}_d(t+1|t), \dots, \\ \boldsymbol{x}_d(t+N_c|t)\big\}.$$

1.2 Use the inverse model (16) to compute the future reference input sequence

$$\boldsymbol{u}_{d|_{t,t+N_c}} = \big\{\boldsymbol{u}_d(t|t), \boldsymbol{u}_d(t+1|t), \dots, \\ \boldsymbol{u}_d(t+N_c-1|t)\big\}.$$

1.3 Set $d=1$, use (36) to compute the shifted centre $\tilde{\boldsymbol{c}}_{ik}(t|t)$, and generate the *one*-step-ahead predictor model of the state-error as given by (35).

Step 2.

2.1 Linearise the state-error model around the origin.

2.2 Use Parisini and Zoppoli's procedure to determine the matrix $\boldsymbol{P}$ and the scalar a^* in the objective function (13).

Step 3.

3.1 Compute the current state shift

$$\tilde{\boldsymbol{x}}(t|t) = \boldsymbol{x}(t) - \boldsymbol{x}_d(t).$$

3.2 Use (36) to compute the sequence of centres

$$\big\{\tilde{\boldsymbol{c}}_{ik}(t+1|t), \tilde{\boldsymbol{c}}_{ik}(t+2|t), \dots, \\ \tilde{\boldsymbol{c}}_{ik}(t+N_c-1|t)\big\},$$

and then use (35) to recursively determine the state-error predictions $\tilde{\boldsymbol{x}}(t+j|t)$ for $1 \le j \ge Nc$.

Step 4.

Form the objective function (13)

Step 5.

5.1 Use the optimisation algorithm to generate the control sequence

$$\{\tilde{\boldsymbol{u}}(t|t), \tilde{\boldsymbol{u}}(t+1|t), \dots, \tilde{\boldsymbol{u}}(t+N_c-1|t)\}.$$

5.2 Apply the hybrid control input $\boldsymbol{u}(t) = \tilde{\boldsymbol{u}}(t|t) + \boldsymbol{u}_d(t)$ to the nominal system, and measure the current state $\boldsymbol{x}(t)$.

Step 6.

6.1 Set the next initial guess of the predicted control sequence to

$$\{\tilde{\boldsymbol{u}}(t|t), \tilde{\boldsymbol{u}}(t+1|t), \dots, \tilde{\boldsymbol{u}}(t+N_c-1|t)\}.$$

6.2 Set $t=t+1$ and go to Step 1.

5. Case Study

This section illustrates how the proposed approach to design a stabilising hybrid predictive-neural tracking controller is implemented for a nonlinear system. The nominal system considered in this simulation is an identified GRBF neural netwok model for a single link robotic manipulator (SLM) (Garces *et al.*, 2003; Kambhampati *et al.*, 1997; Zhihong *et al.*, 1998) described by the following second-order nonlinear differential equation:

$$ml^2\ddot{\theta}(t) + \nu\dot{\theta}(t) + mgl\sin\theta(t) = u(t), \tag{37}$$

where the mass $m = 1.0\,\text{kg}$, the length of the manipulator $l = 1.0\,\text{m}$, the coefficient of the viscous friction $\nu = 1.0\,\text{kg m}^2/\text{s}$, the acceleration due to gravity $g = 9.8\,\text{m/s}^2$, $\theta(t)$ is the angle (in radians) the manipulator makes with the vertical line, and $u(t)$ is the applied torque in Nm.

Using the second-order Taylor expansion to discretise the continuous state-space representation of (37) yields the following discrete-time model:

$$\begin{aligned} x_1(t+1) &= x_1(t) + \left[T - \frac{T^2}{2}\right] x_2(t) \\ &\quad - 9.8\frac{T^2}{2}\sin x_1(t) + \frac{T^2}{2}u(t), \\ x_2(t+1) &= \left[1 - T - \frac{T^2}{2} - 9.8\frac{T^2}{2}\cos x_1(t)\right] x_2(t) \\ &\quad - 9.8\left[T - \frac{T^2}{2}\right]\sin x_1(t) \\ &\quad + \left[T - \frac{T^2}{2}\right] u(t), \end{aligned} \tag{38}$$

where the state variables are $x_1(t) = \theta(t)$, $x_2(t) = \dot{\theta}(t)$, i.e., the angular position and angular velocity, respectively, and $T = 0.01\,\text{s}$ is the sampling period.

Two GRBF neural networks (each having 78 Gaussian units) are identified to represent the states $x_1(t)$ and $x_2(t)$ over the operation domains $X \subset \mathbb{R}^2$ and $U \subset \mathbb{R}^1$ with $X = \{\boldsymbol{x} = [x_1, x_2]^T : |x_1| \le 0.7\,\text{rad}, |x_2| \le 2.5\,\text{rad/s}\}$ and $U = \{u : |u| \le 7\,\text{Nm}\}$. The input vector to each network is

$$\text{col}\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) = \big[x_1(t), x_2(t), u(t)\big]^T. \tag{39}$$

Note that this GRBF model was used for computing the predictions associated with the receding horizon regulator. The nominal system has to fulfil the stabilisability requirement set in Assumption A2. Thus, the linearization of the GRBF neural network in the neighbourhood of the origin gives

$$\boldsymbol{A} = \begin{bmatrix} 0.99597 & 0.01108 \\ -0.09821 & 0.98783 \end{bmatrix},$$

$$B = \begin{bmatrix} -0.00055 \\ 0.01074 \end{bmatrix}.$$

The eigenvalues of $\boldsymbol{A}$ are

$$\xi_{\boldsymbol{A}} = \begin{bmatrix} 0.9919 + 0.03274i \\ 0.9919 - 0.03274i \end{bmatrix}.$$

Therefore, the linearization of the nominal system is stabilisable at the origin with the eigenvalues of the $\boldsymbol{A}$ matrix satisfying $|\xi_{\boldsymbol{A}}(j)| < 1$. The stabilisability of the linearization of the nominal system at equilibrium points far from the origin is also indicated in Table 1, and it can be seen that all eigenvalues are inside the unit circle.

Table 1. Eigenvalues of the $\boldsymbol{A}$ matrix for the linearised nominal system at different equilibrium points.

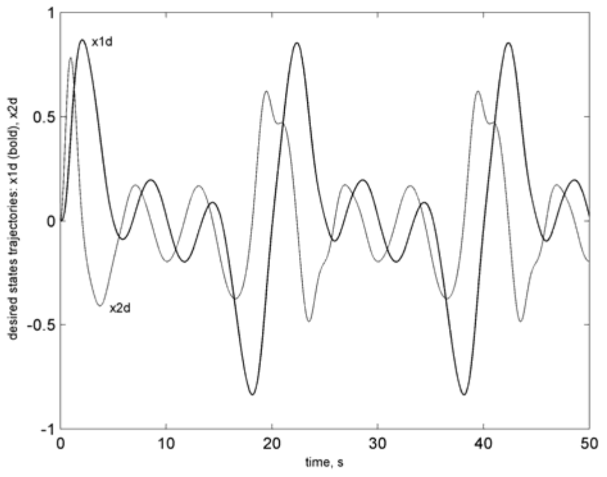
Set point $x_d = [x_1, x_2]^T$	Eigenvalues of matrix A	Radius $\lvert\xi_A(j)\rvert,\ j = 1, 2$
$[0, 0]^T$	$0.9919 \pm 0.032747i$	0.9924
$[0.2, 0]^T$	$0.99537 \pm 0.022741i$	0.9956
$[0.4, 0]^T$	$0.99589 \pm 0.025258i$	0.9962
$[0.6, 0]^T$	$0.99359 \pm 0.037889i$	0.9943
$[-0.2, 0]^T$	$0.98903 \pm 0.04378i$	0.9899
$[-0.4, 0]^T$	$0.98145 \pm 0.022426i$	0.9817
$[-0.6, 0]^T$	$0.98872 \pm 0.029596i$	0.9891

Assuming that the nonlinear system described above is a one-one mapping of its arguments, the feedforward controller can be designed as described in Subsections 3.2 and 3.3. The training data for the inverse model are generated by applying different sine inputs, and collecting the states x_1 and x_2. The measurement noise is assumed to be a Gaussian white process with zero mean and 0.01 variance. All inputs are bounded to be in the operation region of $\pm 7\,\text{Nm}$. The parameters of the training algorithm are selected as follows:

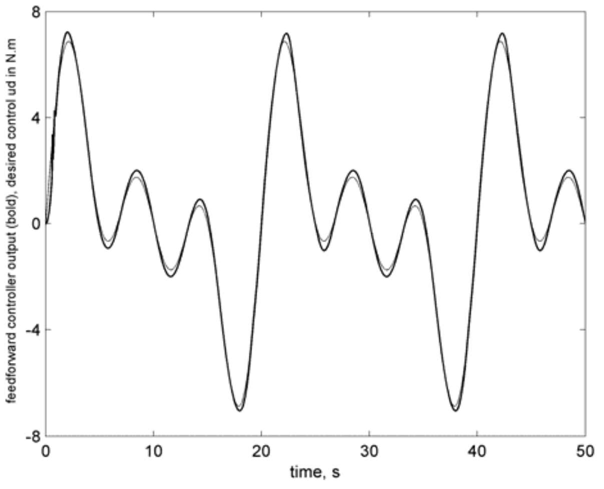
$$\varepsilon_{\min} = \varepsilon_{\max} = 0.175, \quad \gamma = 1, \quad \eta = 0.07, \quad \rho = 0.85,$$

$$\boldsymbol{Q}_0 = 0.025, \quad \boldsymbol{R}_0 = \boldsymbol{S}_0 = 1.$$

When training the inverse, it appeared that a good performance can be achieved with the input vector reduced to

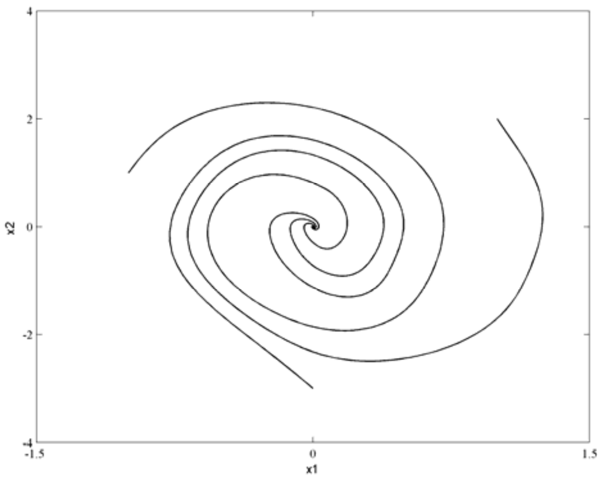


(a)

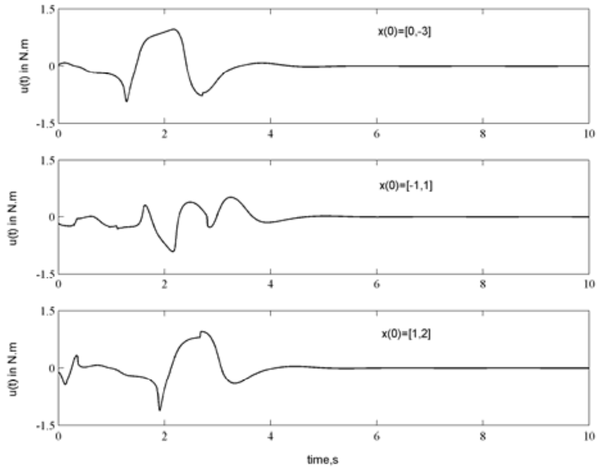


(b)

Fig. 4. Training results for feedforward GRBFNN controller: (a) desired state trajectories, (b) computed control using feedforward network (bold) versus the desired control.



(a)



(b)

Fig. 5. Closed-loop results for tracking a zero reference state: (a) state trajectories, (b) control signals.

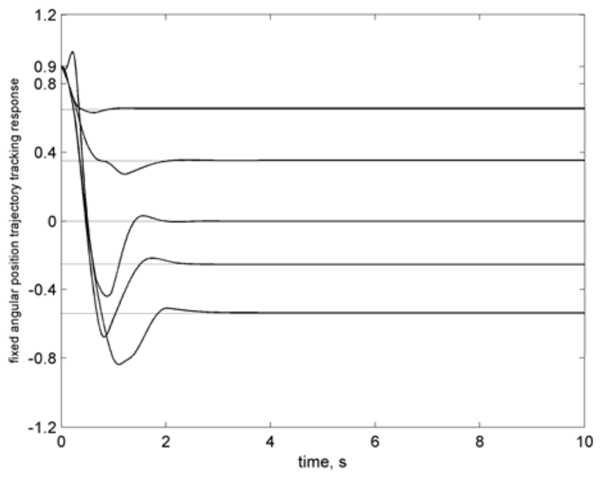
the next state $[x_1(t+1), x_2(t+1)]^T$ only. The performance of the inverse GRBFNN, consisting of 46 Gaussian functions, is illustrated in Fig. 4.

To show the asymptotic closed-loop stability of the hybrid predictive-neural controller, a number of simulations were carried out using the function *fmincon* for constrained optimisation from the MATLAB Optimisation Toolbox. Given (5b) and the input operation domain U defined above, the shifted input $\tilde{u}(t)$ is constrained to vary inside the set $\tilde{U} = \{\tilde{u} = u - u_d : |\tilde{u}| \leq 14\,\mathrm{Nm}\}$. In the first simulation, the goal was to use the nominal GRBFNN model in a classic receding horizon regulation scheme with initial states outside the training range. Here $\boldsymbol{x}_d(t) = \boldsymbol{u}_d(t) = 0$ and the GRBFNN model was directly used to compute the control signal. The matrices in the cost function (13) were selected as $\boldsymbol{Q} = \alpha_1 \boldsymbol{I}$ and $\boldsymbol{R} = \alpha_2 \boldsymbol{I}$ with $\alpha_1 = 0.001$ and $\alpha 2 = 0.5$. The control horizon Nc was set to 5, and the matrix $\boldsymbol{P}$ and the scalar a^* were computed using the procedure from (Parisini *et al.*, 1998), which led to

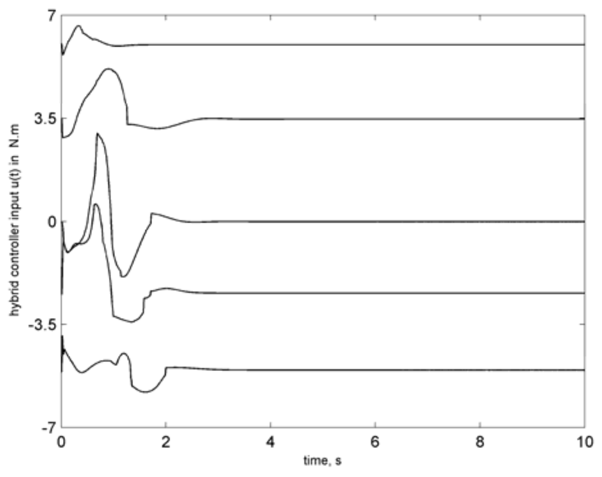
$$\boldsymbol{P} = 10^3 \begin{bmatrix} 141.98 & 3.9460 \\ 3.9460 & 9.6054 \end{bmatrix}$$

and $a^* = 0.008982$. The state trajectories and the control profiles for different initial states are shown in Figs. 5(a) and (b).

The second simulation illustrates the performance of the proposed controller architecture (Fig. 3) while tracking both fixed and changing reference states. The predictive regulator signal $\tilde{u}(t)$ of the tracking controller was computed by successive minimisation of the objective function (13) with the weight parameters α_1 and α_2



(a)



(b)

Fig. 6. Tracking response for fixed reference trajectories: (a) angle trajectories, (b) control signals.

selected as 0.001 and 0.025, respectively. The tracking response of the controller for several set points is illustrated in Figs. 6(a) and 6(b), with the starting point outside the training area. The control horizon is set to $N_c = 9$ and the computed values for the scalar a^* and the matrix $\boldsymbol{P}$ are presented in Table 2 for all set points.

The case of tracking a changing reference trajectory is shown in Fig. 7(a), and the control profiles of the feedforward and predictive (bold) components of the controller are depicted in Fig. 7(b). The control horizon for this latter simulation is set to $N_c = 9$, and having *a-priori* knowledge of the changing set point, the values of the scalar a^* and the matrix $\boldsymbol{P}$ are computed off-line using the same method as indicated previously.

Table 2. Cost function matrix $\boldsymbol{P}$ and scalar a^* for different set points.

Set point $x_d = [x_1, x_2]^T$	Matrix $\boldsymbol{P}$	Scalar a^*
$[0.65, 0]^T$	$10^7 \begin{bmatrix} 1.4671 & 0.0561 \\ 0.0561 & 0.1554 \end{bmatrix}$	5.4414×10^{-6}
$[0.35, 0]^T$	$10^7 \begin{bmatrix} 2.0465 & 0.0909 \\ 0.0909 & 0.0988 \end{bmatrix}$	9.2380×10^{-6}
$[0, 0]^T$	$10^9 \begin{bmatrix} 1.8984 & -0.0334 \\ -0.0334 & 0.2407 \end{bmatrix}$	4.2685×10^{-8}
$[-0.25, 0]^T$	$10^6 \begin{bmatrix} 5.1442 & 0.0083 \\ 0.0083 & 1.7891 \end{bmatrix}$	1.3407×10^{-5}
$[-0.55, 0]^T$	$10^7 \begin{bmatrix} 1.6600 & -0.0710 \\ -0.0710 & 0.2404 \end{bmatrix}$	2.1231×10^{-6}

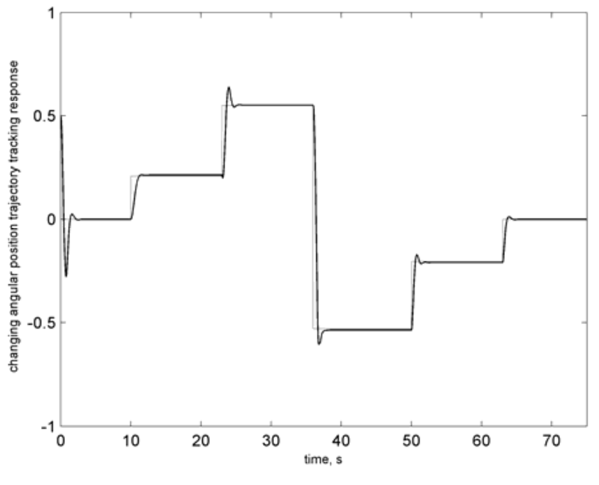
6. Conclusions

A controller architecture for nonlinear systems described by Gaussian radial basis function neural networks has been proposed. The controller solves a nonlinear optimal state-tracking control problem by a combination of a stabilising nonlinear state feedback regulator and a feedforward neuro-controller. The state feedback regulator was derived by converting the nonlinear tracking problem into a nonlinear regulation problem for state-error dynamics. The regulation problem was solved using a nonlinear predictive control approach with guaranteed asymptotic stability. The feedforward neuro-controller was designed using the concept of inverse mapping. The proposed control scheme was applied to a single-link robotic manipulator. Simulation results demonstrated good tracking performances and the stability of the hybrid control scheme.

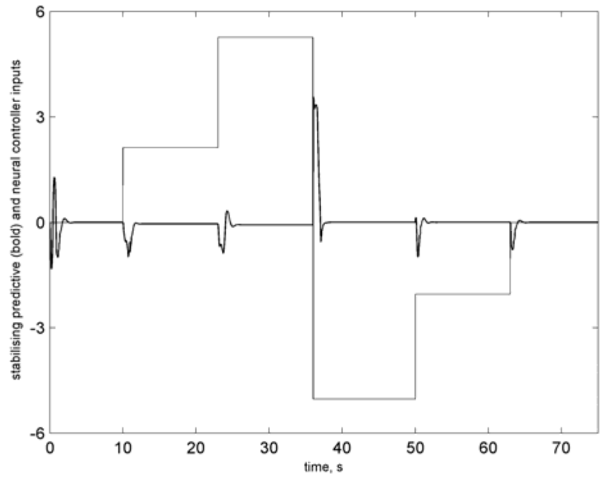
References

Becerra V.M., Roberts P.D. and Griffiths G.W. (1998): *Novel developments in process optimisation using predictive control.* — J. Process Contr., Vol. 8, No. 2, pp. 117–138.

Becerra V.M., Abu-el-zeet Z.H. and Roberts P.D. (1999): *Integrating predictive control and economic optimisation.* — Comput. Contr. Eng. J., Vol. 10, No. 5, pp. 198–208.



(a)



(b)

Fig. 7. Tracking response for a changing reference trajectory: (a) angle trajectory (bold), (b) stabilising predictive regulator input (bold), and a feedforward controller input.

Chen C.C. and Shaw L. (1982): *On receding horizon feedback control*. — Automatica, Vol. 18, No. 3, pp. 349–352.

Chen H. and Allgöwer F. (1998): *A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability*. — Automatica, Vol. 34, No. 10, pp. 1205–1217.

De Nicolao G., Magni L. and Scattolini R. (1997): *Stabilizing receding-horizon control of nonlinear time-varying systems*. — IEEE Trans. Automat. Contr., Vol. 43, No. 7, pp. 1030–1036.

Eaton J.W. and Rawlings J.B. (1992): *Model predictive control of chemical processes*. — Chem. Eng. Sci., Vol. 47, No. 4, pp. 705–720.

Garces F., Becerra V.M., Kambhampati C. and Warwick K. (2003): *Strategies for Feedback Linearisation: A Dynamic Neural Network Approach*. — London: Springer.

Garcia C.E., Prett D.M. and Morari M. (1989): *Model predictive control: Theory and practice—A survey*. — Automatica, Vol. 25, No. 3, pp. 335–347.

Hornik K., Stinchcombe M. and White H. (1989): *Multilayer feedforward networks are universal approximators*. — Neural Networks, Vol. 2, No. 5, pp. 359–366.

Hunt K.J., Sbarbaro D., Zbikowski R. and Gawthrop P.J. (1992): *Neural networks for control systems: A survey*. — Automatica, Vol. 28, No. 6, pp. 1083–1112.

Kadirkamanathan V. and Niranjan M. (1993): *A function estimation approach to sequential learning with neural networks*. — Neural Comput., Vol. 5, No. 6, pp. 954–975.

Kambhampati C., Delgado A., Mason J.D. and Warwick K. (1997): *Stable receding horizon control based on recurrent networks*. — IEE Proc. Contr. Theory Applic., Vol. 144, No. 3, pp. 249–254.

Keerthi S.S. and Gilbert E.G. (1988): *Optimal, infinite-horizon feedback laws for a general class of constrained discrete-time systems*. — J. Optim. Theory Applic., Vol. 57, No. 2, pp. 265–293.

Kwakernaak H. and Sivan R. (1972): *Linear Optimal Control Systems*. — New York: Wiley.

Magni L., De Nicolao G., Magnani L. and Scattolini R. (2001): *A stabilizing model-based predictive control algorithm for nonlinear systems*. — Automatica, Vol. 37, No. 9, pp. 1351–1362.

Mayne D.Q. and Michalska H. (1990): *Receding horizon control of nonlinear systems*. — IEEE Trans. Automat. Contr., Vol. 35, No. 7, pp. 814–824.

Mayne D.Q., Rawlings J.B., Rao C.V. and Scokaert P.O.M. (2000): *Constrained model predictive control: Stability and optimality*. — Automatica, Vol. 36, No. 6, pp. 789–814.

Michalska H. and Mayne D.Q. (1993): *Robust receding horizon control of constrained nonlinear systems*. — IEEE Trans. Automat. Contr., Vol. 38, No. 11, pp. 1623–1633.

Morari M. and Lee J.H. (1999): *Model predictive control: Past, present and future*. — Comput. Chem. Eng., Vol. 23, No. 4, pp. 667–682.

Narendra K.S. and Parthasarathy K. (1990): *Identification and control of dynamical using neural networks*. — IEEE Trans. Neural Netw., Vol. 1, No. 1, pp. 4–27.

Parisini T. and Zoppoli R. (1995): *A receding-horizon regulator for nonlinear systems and a neural approximation*. — Automatica, Vol. 31, No. 10, pp. 1443–1451.

Parisini T., Sanguinetti M. and Zoppoli R. (1998): *Nonlinear stabilization by receding-horizon neural regulators*. — Int. J. Contr., Vol. 70, No. 3, pp. 341–362.

Park Y.M., Choi M.S. and Lee K.W. (1996): *An optimal tracking neuro-controller for nonlinear dynamic systems*. — IEEE Trans. Neural Netw., Vol. 7, No. 5, pp. 1099–1110.

Richalet J. (1993): *Industrial applications of model based predictive control*. — Automatica, Vol. 29, No. 5, pp. 1251–1274.

Richalet J., Rault A., Testud J.L. and Papon J. (1978): *Model predictive heuristic control: Application to industrial processes*. — Automatica, Vol. 14, No. 2, pp. 413–428.

Yingwei L., Sundararajan N. and Saratchandran P. (1997): *Identification of time-varying nonlinear systems using minimal radial basis function neural networks*. — IEE Proc. Contr. Theory Appl., Vol. 144, No. 2, pp. 202–208.

Zhihong M., Wu H.R. and Palaniswami M. (1998): *An adaptive tracking controller using neural networks for a class of nonlinear systems*. — IEEE Trans. Neural Netw., Vol. 9, No. 5, pp. 947–954.

Received: 6 October 2004
Revised: 22 April 2005