

Two-layer networked learning control using self-learning fuzzy control algorithms *

Du Dajun¹, Fei Minrui¹, Hu Huosheng², Li Lixiong¹

(1 School of Mechatronical Engineering and Automation, Shanghai University, Shanghai 200072, China;

2 Department of Computing & Electronic Systems, University of Essex, Colchester CO4 3SQ, U. K.)

Abstract: Since the existing single-layer networked control systems have some inherent limitations and cannot effectively handle the problems associated with unreliable networks, a novel two-layer networked learning control system (NLCS) is proposed in this paper. Its lower layer has a number of local controllers that are operated independently, and its upper layer has a learning agent that communicates with the independent local controllers in the lower layer. To implement such a system, a packet-discard strategy is firstly developed to deal with network-induced delay and data packet loss. A cubic spline interpolator is then employed to compensate the lost data. Finally, the output of the learning agent based on a novel radial basis function neural network (RBFNN) is used to update the parameters of fuzzy controllers. A nonlinear heating, ventilation and air-conditioning (HVAC) system is used to demonstrate the feasibility and effectiveness of the proposed system.

Key words: networked learning control system (NLCS); radial basis function neural network (RBFNN); cubic spline interpolator; HVAC system

1 Introduction

Up to now, networked control systems (NCS) have been widely used in car suspension systems^[1], large pressurized heavy water reactors^[2] and mobile robots^[3] due to their low cost installation, ease of maintenance and great flexibility. Most of the existing systems communicate with sensors and actuators over single-layer communication networks and the controlled plant is linear. Recently, researches on nonlinear NCS have attracted significant amount of interests from both industry and academics. To improve NCS performance, some learning control algorithms have been proposed. For instance, a previous cycle based learning (PCL) method was incorporated into networked control for a nonlinear system, and the convergence in the iteration domain could be guaranteed^[4]. With the delays estimated by a delay window (DW), a novel fuzzy logic method was proposed to calculate LQR parameters online^[5], which could not only save the

power of control node but also improve system performance.

For a nonlinear NCS, conventional modelling and control methods may be difficult to apply as the adverse effects of network-induced delay and data packet loss make the control of a nonlinear plant more complicated. Although learning control is able to improve control performance by online mining of valuable knowledge and potential laws, accumulating experience and adapting to environments, it is difficult to design and implement such control strategies using embedded controllers or program logic controllers (PLC) due to high computational complexity in a typical NCS. However, two-layer networked control systems^[6-7] can provide a solution for the implementation of learning strategies due to strong computational ability of the upper (second) layer controller. Motivated from the above observations, a novel two-layer NLCS is investigated in this paper. Comparing with a typical NCS, the proposed system is characterized by two-layer network, local controller, learning agent, and complex plant.

The rest of this paper is organized as follows. Section 2

Received Date:2007-09 收稿日期:2007-09

* Foundation item: Supported by National Natural Science Foundation of China (60774059), Project of Science & Technology Commission of Shanghai Municipality (061107031, 06DZ22011, 06ZR14131), the Sunlight Plan Following Project of Shanghai Municipal Education Commission and Shanghai Educational Development Foundation (06GG10), and Shanghai Leading Academic Disciplines (T0103)

describes the two-layer networked learning control system architecture, and a discard-packet strategy is proposed and cubic spline interpolation is used to compensate data packet loss. Self-learning fuzzy control algorithm based on a novel RBF network is introduced in Section 3. In Section 4, simulation results are given to show the feasibility of the proposed system. Finally, a brief conclusion and future work are given in Section 5.

2 Two-layer networked learning control system

2.1 Introduction of two-layer networked learning control system architecture

Fig. 1 shows the proposed two-layer networked learning control system architecture, which aims to achieve better control performance, better interference rejection and better adaptability to a dynamic environment.

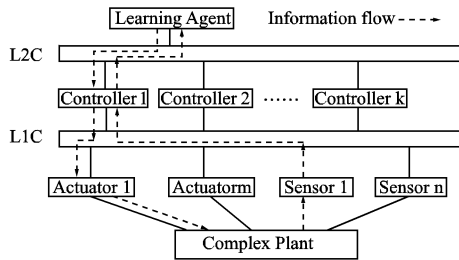


Fig. 1 Two-layer networked learning control system

In this design, local controllers communicate with the sensors and actuators that are attached to the plant through the first layer communication network called L1C, typically some kind of field bus dedicated to real-time control. Local controller also communicates with computer system, which typically functions as a learning agent through the second layer communication network called L2C. This network is typically some kind of existing local area network, wide area network (WAN), or possibly the Internet. Control signal traffic at L2C shares the available network bandwidth with other data communications. This general architecture can be used in many industrial applications for distributed plants. For example, an industrial process control systems such as SIMATIC PCS 7 process control system where a number of embedded controllers or PLC through L1C control a number of sub-processes or units, and supervisory computers through L2C co-ordinate various low-level control actions carried out by local controllers.

2.2 Characteristics of two-layer networked learning control architecture

Since network-induced delay and data packet loss are

unavoidably introduced into L2C, strategies were deployed to deal with both short network-induced delay [6] and long network-induced delay [8]. As shown in fig. 1, L1C uses typically some kinds of field bus so that network-induced delay and data packet loss can be minimized. A local area network is used as L2C. The sensor node was sampled periodically, and the sampling period was $h = 15$ ms. The characteristic of the network-induced delay is illustrated in fig. 2.

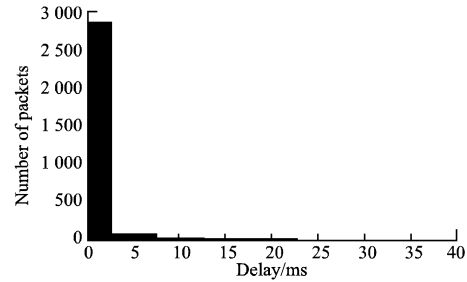


Fig. 2 Network-induced delay from a local controller to the learning agent

It is well known that the network transmission may be unreliable and data packets can suffer from network-induced delays, out-of-order, in the worst situation, they can be lost during the transmission pathway. Those cases are illustrated in fig. 3. The time instants at which the data packets reach the learning agent are random due to the random nature of network-induced delay. Let T_i^l denote those instants relative to initial time, where the subscript 'i' is the serial number of the data packet received by the learning agent and the superscript 'l' means the learning agent. Note that we have $T_{i+1}^l > T_i^l$. Therefore, corresponding to the arrival data packets by the learning agent, there exists a serial pairs $\{y(i), T_i^l\}$, where 'y(i)' is a subset of $\{y(k)\}$. Several methods have been proposed to deal with packet loss. A zero order holding (ZOH) at the receiving side of a communication medium was used [9], i. e., when an actuator or sensor fails to access the medium, the value stored in a ZOH is fed into the plant or controller ($y(i) = y(i - 1)$). A value of zero was fed into the plant or controller when an actuator or sensor fails to access the medium ($y(i) = 0$), which however was inappropriate due to the real system output unequal to zero [10].

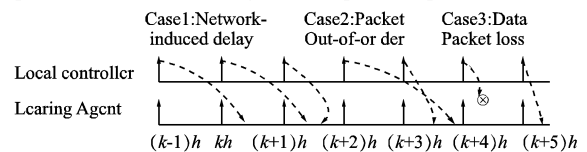


Fig. 3 Timing diagram

2.3 Discard-packet strategy

Unlike the above methods that deal with network-induced delays and data packet loss separately, we take into account network-induced delays, data packet out-of-order and data packet loss within one framework. It is well known that in many control strategies, only the ‘newest’ data points are used to achieve best control performance and guarantee the system stabilization. Therefore, it is fairly reasonable that a long delayed data packet and out-of-order data packet, even when it finally arrives, should also be ignored. Then, a data packet processing strategy, namely discard-packet scheme, is used in the paper. Here, if a data packet is unable to arrive at the learning agent within the corresponding sampling period, it will be discarded. This strategy treats long delayed data packets and out-of-order data packets as packet loss such that the ‘newest’ data are used. Furthermore, to compensate the data packet ($y(i)$) loss and improve the control performance, a cubic spline interpolator, $y(i) = \hat{y}(i)$, is designed.

2.4 Cubic spline interpolator

A cubic spline^[11] is a spline consisting of piecewise cubic polynomials $g(x)$ that interpolates a real-valued function $f(x)$ at the points $x_0 < x_1 < \dots < x_N$ where the values of $f(x)$ are known. So, we construct $g(x)$ on each interval $[x_i, x_{i+1}]$, $i=0, \dots, N-1$, as a cubic polynomial:

$$V_i(x) = c_{0,i} + c_{1,i}(x-x_i) + c_{2,i}(x-x_i)^2 + c_{3,i}(x-x_i)^3 \quad (1)$$

Once we find the coefficients $c_{j,i}$, $j=0, \dots, 3$; $i=0, \dots, N-1$, we can evaluate $g(x)$ for any point x in $[x_0, x_N]$.

To guarantee $g(x)$ to be continuous on $[x_0, x_N]$, it is assumed that:

$$V_{i-1}(x_i) = V_i(x) = f_i(x_i), \quad i=1, \dots, N \quad (2)$$

Each $V_i(x)$ is interpolated at only two points. Since a cubic polynomial can interpolate a function at four points, we have freedom remaining in choosing $V_i(x)$. To make use of the freedom, further constraints are imposed such that $V_i(x)$ must agree with $V_{i-1}(x_i)$ in both slope and curvature; that is:

$$V'_i(x_i) = V'_{i-1}(x_i), \quad i=1, \dots, N-1 \quad (3)$$

$$V''_i(x_i) = V''_{i-1}(x_i), \quad i=1, \dots, N-1 \quad (4)$$

We assume that curvatures k_i at x_i and k_{i+1} at x_{i+1} be parameters of $V_i(x)$. Since $V_i(x)$ is a cubic polynomial, $V''_i(x)$ is a linear polynomial constrained such that $V''_i(x_i) = k_i$ and $V''_i(x_{i+1}) = k_{i+1}$, $i=0, \dots, N-1$. That is:

$$V''_i(x) = k_i \frac{x_{i+1} - x}{x_{i+1} - x_i} + k_{i+1} \frac{x - x_i}{x_{i+1} - x_i} \quad (5)$$

Integrating (5), we have:

$$V'_i(x) = -\frac{k_i}{2} \frac{x_{i+1} - x}{x_{i+1} - x_i} + \frac{k_{i+1}}{2} \frac{(x - x_i)^2}{x_{i+1} - x_i} + a_i \quad (6)$$

Integrating (6) and then using (2), we have:

$$V_i(x) = -\frac{k_i}{6} \frac{(x_{i+1} - x)^3}{x_{i+1} - x_i} + \frac{k_{i+1}}{6} \frac{(x - x_i)^3}{x_{i+1} - x_i} + A_i(x_{i+1} - x) + B_i(x - x_i), \quad i=0, \dots, N-1 \quad (7)$$

where $A_i = \frac{f(x_i)}{x_{i+1} - x_i} - \frac{k_i}{6}(x_{i+1} - x_i)$, $i=0, \dots, N-1$;

$$B_i = \frac{f(x_{i+1})}{x_{i+1} - x_i} - \frac{k_{i+1}}{6}(x_{i+1} - x_i), \quad i=0, \dots, N-1.$$

Applying the constraint given by (3) yielding:

$$k_{i-1}(x_i - x_{i-1}) + 2k_i(x_{i+1} - x_{i-1}) + k_{i+1}(x_{i+1} - x_i) = 6 \left[\frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} - \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} \right], \quad i=1, \dots, N-1 \quad (8)$$

Equation (8) represents a system of $N-1$ linear equations with $N+1$ unknowns k_i , $i=0, \dots, N$. Obviously, if the boundary conditions at the endpoints x_0 and x_N are given, (8) can be solved, thus the coefficients $c_{j,i}$ of $V_i(x)$ in (1) can be found, which in turn construct our interpolator $g(x)$. If cubic spline interpolation has no boundary conditions, then the not-a-knot end conditions are commonly used. This is equivalent to assuming that the first two cubic polynomials equal in the cubic derivative, in the mean time the last two cubic polynomials also equal in the cubic derivative.

3 Self-learning fuzzy control algorithm based on a novel RBF network

In the proposed NLCS, local controllers use fuzzy control strategy while RBFNN based FRA is employed in the learning agent. The algorithms of each component are explained as follows respectively.

3.1 The fuzzy controller

The fuzzy controller is described in the form of an analytical formula^[12] instead of common fuzzy rule tables, that is:

$$u(t) = a(t)b(t)E(t) + [1 - a(t)]b(t)EC(t) + [1 - b(t)]ER(t) \quad (9)$$

where $E(t)$, $EC(t)$ and $ER(t)$ denote the fuzzy variables corresponding to their crisp variables, control error $e(t)$, change in error $ec(t) = e(t) - e(t-1)$ and acceleration error $er(t) = ec(t) - ec(t-1)$; $a(t)$ and $b(t) \in [0, 1]$ denote the tuning parameters between $E(t)$, $EC(t)$ and $ER(t)$. The fuzzy variables and their corresponding crisp variables differ in the transformation factors relating to their universes of discourses. Contrary to common approach, the

all universes of discourse are considered continuously in this study.

3.2 RBFNN based on FRA

A schematic of the RBF network with n inputs and a scalar output is depicted in fig. 4. Such a network implements a nonlinear mapping according to:

$$f_r(x) = w_0 + \sum_{i=1}^{n_r} w_i u_i \quad (10)$$

where w_0 is the biasing term; u_i is the i -th hidden unit output; $w_i, 0 \leq i \leq n_r$, are the weights between the hidden layer and the output layer; $c_i, 0 \leq i \leq n_r$, are known as the RBF centres; and n_r is the number of centres.

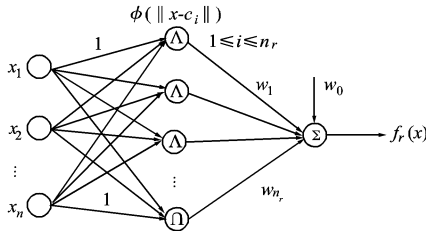


Fig. 4 Radial basis function network structure

The choice of nonlinearity $\phi(\cdot)$ is not crucial to the performance of the RBF network. The thin-plate-spline function and Gaussian function are two typical choices. In this paper, $\phi(\cdot)$ is chosen to be the Gaussian function, that is:

$$u_i = \phi(\|x - c_i\|) = \exp\left[-\frac{(x - c_i)^T(x - c_i)}{2\sigma_i^2}\right] \quad (11)$$

where $\phi(\cdot)$ is a Gaussian function from \mathbf{R}^+ to \mathbf{R}^- , $\|\cdot\|$ denotes the Euclidean norm, $x \in \mathbf{R}^n$ is the input vector, $c_i = (c_{i1}, c_{i2}, \dots, c_{in})^T$ is the centre of the i -th hidden node, σ_i is the i -th hidden unit spread factor.

The RBFNN with scalar output (10) can be viewed as a special case of the linear regression model:

$$y(t) = \sum_{i=1}^M p_i(t) \theta_i = e(t) \quad (12)$$

where $y(t)$ is the desired output; θ_i are the parameters; $p_i(t)$ are known as regressors, which are some fixed functions of $x(t)$. It is apparent that a hidden unit output u_i corresponds to a regressor $p_i(t)$. The error signal $\varepsilon(t)$ is assumed to be uncorrelated with the regressor $p_i(t)$. For $t = 1$ to N , (12) is arranged in the matrix form:

$$\mathbf{y} = \mathbf{p}\boldsymbol{\theta} + \mathbf{E} \quad (13)$$

where $\mathbf{y} = [y(1), \dots, y(N)]^T$, $\mathbf{p} = [p_1, \dots, p_M]$, $\mathbf{p}_i = [p_i(1), \dots, p_i(N)]^T$, $1 \leq i \leq M$; $\boldsymbol{\theta} = [\theta_1, \dots, \theta_M]^T$; $\mathbf{E} = [e(1), \dots, e(N)]^T$.

The performance of an RBF network critically depends upon the chosen centres. Thus the problem of how to select a

suitable set of RBF centres from the data set can be regarded as an example of how to select a subset of significant regressors $p_i(t)$ from a given candidate set \mathbf{P} . In general, the orthogonal least squares (OLS) [13] algorithm was used to select a suitable set of centres (regressors) from a large set of candidates, however a fast recursive algorithm will be employed to select the centres in this paper. Comparing with OLS, FRA is characterized by its better numerical stability and less computation time [14-15].

Firstly, we define a matrix $\boldsymbol{\phi}_k \in \mathbf{R}^{N \times k}$, which contains the first k columns of the regression matrix \mathbf{P} in (13), resulting in the model of the form:

$$\mathbf{y} = \boldsymbol{\phi}_k \boldsymbol{\theta}_k + \mathbf{e} \quad (14)$$

where $\mathbf{y} = [y(1), \dots, y(N)]^T$; $\boldsymbol{\phi}_k = [p_1, \dots, p_k]$; $\boldsymbol{\theta}_k = [\theta_1, \dots, \theta_k]^T$; $\mathbf{e} = [e(1), \dots, e(N)]^T$.

A cost function can be defined as:

$$\mathbf{J}_k = \sum_{i=1}^N \left(y - \sum_{i=1}^k p_i \theta_i \right)^2 = (\mathbf{y} - \boldsymbol{\phi}_k \boldsymbol{\theta}_k)^T (\mathbf{y} - \boldsymbol{\phi}_k \boldsymbol{\theta}_k) \quad (15)$$

If the matrix $\boldsymbol{\phi}_k$ is of full-column rank, the least-squares estimation of the regression coefficients in (14) is given by:

$$\hat{\boldsymbol{\theta}}_k = (\boldsymbol{\phi}_k^T \boldsymbol{\phi}_k)^{-1} \boldsymbol{\phi}_k^T \mathbf{y} \quad (16)$$

To achieve a fast selection of model terms, we define a recursive matrix $\mathbf{M}_k \stackrel{\Delta}{=} \boldsymbol{\phi}_k^T \boldsymbol{\phi}_k$. Then (16) can be rewritten as follows:

$$\hat{\boldsymbol{\theta}}_k = (\mathbf{M}_k)^{-1} \boldsymbol{\phi}_k^T \mathbf{y} \quad (17)$$

From (15) and (17), the cost function becomes:

$$\mathbf{J}_k = \mathbf{y}^T \mathbf{y} - \hat{\boldsymbol{\theta}}_k^T \boldsymbol{\phi}_k^T \mathbf{y} \quad (18)$$

Next, a recursive matrix $\mathbf{R}_k \in \mathbf{R}^{N \times N}$, $k = 1, \dots, M$.

$$\mathbf{R}_k \stackrel{\Delta}{=} \mathbf{I} - \boldsymbol{\phi}_k \mathbf{M}_k^{-1} \boldsymbol{\phi}_k^T \quad (19)$$

where $\boldsymbol{\phi}_k, k = 1, \dots, M$ is of full-column rank, and $\mathbf{R}_0 \stackrel{\Delta}{=} \mathbf{I}$.

The matrix \mathbf{R}_k is a residue matrix. Suppose $\{P_i, i = 1, \dots, M\}$ in \mathbf{P} are mutually linearly independent, whose properties are given as follows:

$$\mathbf{R}_{k+1} = \mathbf{R}_k - \frac{\mathbf{R}_k \mathbf{p}_{k+1} \mathbf{p}_{k+1}^T \mathbf{R}_k^T}{\mathbf{p}_{k+1}^T \mathbf{R}_k \mathbf{p}_{k+1}}; k = 0, \dots, M-1 \quad (20)$$

$$\mathbf{R}_k^T = \mathbf{R}_k, (\mathbf{R}_k)^2 = \mathbf{R}_k \quad (21)$$

$$\mathbf{R}_k \mathbf{R}_j = \mathbf{R}_j \mathbf{R}_k = \mathbf{R}_k, \quad k \geq j \quad (22)$$

$$\mathbf{R}_k \mathbf{p}_i = 0, \quad \forall i \in \{1, \dots, k\} \quad (23)$$

Substituting (19) into (18), the result is:

$$\mathbf{J}_k = \mathbf{y}^T \mathbf{y} - \hat{\boldsymbol{\theta}}_k^T \boldsymbol{\phi}_k^T \mathbf{y} = \mathbf{y}^T \mathbf{R}_k \mathbf{y} \quad (24)$$

Then, using (20) and (24), we have:

$$\begin{cases} \mathbf{J}_{k+1} = \mathbf{y}^T \mathbf{R}_{k+1} \mathbf{y} = \mathbf{J}_k - \frac{\mathbf{y}^T \mathbf{R}_k \mathbf{p}_{k+1} \mathbf{p}_{k+1}^T \mathbf{R}_k^T \mathbf{y}}{\mathbf{p}_{k+1}^T \mathbf{R}_k \mathbf{p}_{k+1}} \\ \mathbf{J}_0 = \mathbf{y}^T \mathbf{y} \end{cases} \quad (25)$$

Furthermore, we define:

$$\mathbf{p}_i^{(k)} \triangleq \mathbf{R}_k \mathbf{p}_i; \mathbf{p}_i^{(0)} \triangleq \mathbf{R}_0 \mathbf{p}_i = \mathbf{p}_i; i = 1, \dots, M; k = 0, 1, \dots, M \quad (26)$$

Consider (21) and (26), (25) can be rewritten as follows:

$$\delta \mathbf{J}_{k+1} = -(\mathbf{y}^T \mathbf{p}_{k+1}^{(k)})^2 / ((\mathbf{p}_{k+1}^{(k)})^T \mathbf{p}_{k+1}^{(k)}); k = 0, \dots, n-1 \quad (27)$$

Equation (27) expresses the net contribution of term \mathbf{p}_{k+1} to the cost function when it is included in the model. To further simplify the computational complexity, two new quantities are defined:

$$\begin{cases} a_{k,i} \triangleq (\mathbf{p}_k^{(k-1)})^T \mathbf{p}_i^{(k-1)}; a_{1,i} \triangleq \mathbf{p}_1^T \mathbf{p}_i; \\ a_{k,y} \triangleq (\mathbf{p}_k^{(k-1)})^T \mathbf{y}; a_{1,y} \triangleq \mathbf{p}_1^T \mathbf{y}; \\ i = k, \dots, M; k = 1 \dots, M \end{cases} \quad (28)$$

Using properties of the matrix \mathbf{R}_k and the definition of $\mathbf{p}_i^{(k)}$ in (26), (28) can be simplified as follows:

$$\begin{cases} a_{k,i} = \mathbf{p}_k^T \mathbf{p}_i - \sum_{j=1}^{k-1} (a_{j,k} a_{j,i}) / a_{j,j} \\ a_{k,y} = \mathbf{p}_k^T \mathbf{y} - \sum_{j=1}^{k-1} (a_{j,k} a_{j,y}) / a_{j,j} \end{cases} \quad (29)$$

Using (29), it follows that:

$$\begin{cases} \mathbf{y}^T \mathbf{p}_i^{(k)} = \mathbf{y}^T \mathbf{p}_i - \sum_{j=1}^k (a_{j,y} a_{j,i}) / a_{j,j} \\ (\mathbf{p}_i^{(k)})^T \mathbf{p}_i^{(k)} = (\mathbf{p}_i)^T \mathbf{p}_i - \sum_{j=1}^k (a_{j,i}^2) / a_{j,j} \end{cases} \quad (30)$$

Finally, substituting (30) into (27), the term $\mathbf{p}_{k+1}, k=0, \dots, M-1$ to the cost function can be explicitly expressed as:

$$\delta \mathbf{J}_{k+1} = - \frac{(\mathbf{y}^T \mathbf{p}_{k+1} - \sum_{j=1}^k (a_{j,y} a_{j,k+1} / a_{j,j}))^2}{(\mathbf{p}_{k+1})^T \mathbf{p}_{k+1} - \sum_{j=1}^k (a_{j,k+1}^2 / a_{j,j})} \quad (31)$$

Equation (31) computes the net contribution of every candidate. Then the centres are selected one by one with the cost function being maximally reduced each time. The procedure is terminated at the M_s step when:

$$1 - \sum_{j=1}^{M_s} [err]_j < \rho \quad (32)$$

where $0 < \rho < 1$ is a chosen tolerance. This gives rise to a subset containing M_s centres (significant regressors).

After the centres of RBFNN were determined, the weights between the hidden layer and the output layer could be computed. First, both sides of (17) are multiplied by ϕ_k :

$$\phi_k \hat{\Theta}_k = \phi_k (\mathbf{M}_k)^{-1} \phi_k^T \mathbf{y} = (\mathbf{I} - \mathbf{R}_k) \mathbf{y} = \mathbf{y} - \mathbf{R}_k \mathbf{y} \quad (33)$$

Next, multiplying both sides of (33) by $(\mathbf{P}_j^{j-1})^T$ and using (33) to produce:

$$\sum_{i=j}^k \hat{\theta}_i (\mathbf{p}_j^{(j-1)})^T \mathbf{p}_i = (\mathbf{p}_j^{(j-1)})^T \mathbf{y}; j = 1, \dots, k \quad (34)$$

Finally, using (29) and (34), it follows from (34) that:

$$\hat{\theta}_j = (a_{j,y} - \sum_{i=j+1}^k \hat{\theta}_i a_{j,i}) / a_{j,j}; j = k, k-1, \dots, 1 \quad (35)$$

Using (35), the weights between the hidden layer and the output layer could be then derived.

3.3 Self-learning algorithm for fuzzy controller

The index of control errors is considered as follows:

$$J_e = \sum_{t=1}^N [r - y(t+1)]^2 / 2 \quad (36)$$

where r and $y(t+1)$ denote the desired regulated value and the output of the plant respectively.

Assuming that the RBFNN parameters are known variables that have been obtained off-line or by the last one-step learning result. The following equations are used to modify the tuning parameters $a(t)$ and $b(t)$.

$$a(t+1) = a(t) + \Delta a(t) \quad (37)$$

$$b(t+1) = b(t) + \Delta b(t) \quad (38)$$

$$\Delta a(t) = -h_a (\partial J_e / \partial a(t)) \quad (39)$$

$$\Delta b(t) = -h_b (\partial J_e / \partial b(t)) \quad (40)$$

learning factors are $h_a, h_b \in (0, 1)$.

$$\begin{aligned} \partial J_e / \partial a(t) &= [r - y(t+1)] [\partial y(t+1) / \partial a(t)] \approx \\ &[r - \varepsilon - \bar{y}(t+1)] [\partial \bar{y}(t+1) / \partial a(t)] = [r - y(t+1)] \cdot \\ &[\partial \bar{y}(t+1) / \partial a(t)] \end{aligned} \quad (41)$$

$$\partial \bar{y}(t+1) / \partial a(t) = [\partial f_r / \partial u(t)] [\partial u(t) / \partial a(t)] \quad (42)$$

$$\partial u(t) / \partial a(t) = -b(t) [E(t) - EC(t)] \quad (43)$$

where $\partial \varepsilon / \partial a(t)$ is neglected; $\varepsilon = y(t+1) - \bar{y}(t+1)$.

Similarly:

$$\begin{aligned} \partial J_e / \partial b(t) &= [r - y(t+1)] [\partial y(t+1) / \partial b(t)] \approx \\ &[r - \varepsilon - \bar{y}(t+1)] [\partial \bar{y}(t+1) / \partial b(t)] = [r - y(t+1)] \\ &[\partial \bar{y}(t+1) / \partial b(t)] \end{aligned} \quad (44)$$

$$\partial \bar{y}(t+1) / \partial b(t) = [\partial f_r / \partial u(t)] [\partial u(t) / \partial b(t)] \quad (45)$$

$$\partial u(t) / \partial b(t) = -[a(t) E(t) + (1 - a(t)) EC(t) - ER(t)] \quad (46)$$

where $\partial \varepsilon / \partial b(t)$ is neglected. And then:

$$\begin{aligned} \frac{\partial f_r}{\partial u(t)} &= \frac{\partial f_r}{\partial x_1} = \frac{\partial f_r}{\partial u_i} \cdot \frac{\partial u_i}{\partial x_1} = \sum_{i=1}^{n_i} w_i \frac{\partial u_i}{\partial x_1} = \\ &= \sum_{i=1}^{n_i} \frac{w_i u_i (x_1 - c_{i1})}{\sigma_1^2} \end{aligned} \quad (47)$$

Equations (37) ~ (47) form one-step self-learning algorithm for tuning parameters $a(t), b(t)$ of the fuzzy controller

in a controlling period, essentially, which implies to regulate the fuzzy control rules as a human operator does in real-time.

4 Simulation results

In order to investigate feasibility of the proposed scheme, a HVAC system^[15] is considered, which is shown in fig. 5. The states of the system are input and output temperatures of air and water, and the air and water flow rates, $T_{ai}, T_{ao}, T_{wi}, T_{wo}, f_a, f_w$. The control signal, C , affects the water flow rate. The variables T_{ai}, T_{wi} , and f_a were modified by random walks to model disturbances and changing conditions that would occur in actual heating and air conditioning systems. The bounds on the random walks were $5\text{ }^\circ\text{C} \leq T_{ai} \leq 6\text{ }^\circ\text{C}, 7\text{ }^\circ\text{C} \leq T_{wi} \leq 78\text{ }^\circ\text{C}$ and $0.7\text{ kg/s} \leq f_a \leq 0.9\text{ kg/s}$.

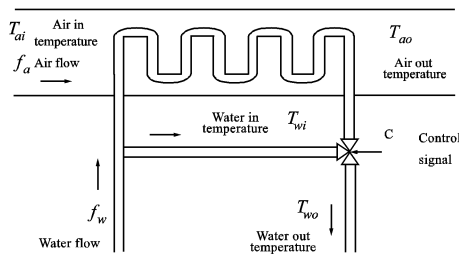


Fig. 5 Schematic diagram of heating, ventilation and air-conditioning systems

Based on the control scheme shown in fig. 1, the experiments on controlling the output temperature of air have been performed. More detailed implementation can be briefly outlined here.

(1) Shanghai University campus-wide network was used as the second layer network in the experiment. TCP/IP sockets are used for communication between nodes. The two loss rates of data packet, 3.64% and 6.36%, were separately considered, and the discard-packet strategy had been employed in the experiment.

(2) Five cases were considered in our experiments;

Only local controller (fuzzy controller) was used to control the HVAC system;

The HVAC system was controlled by a two-layer networked learning control system with a cubic spline interpolator under 3.64% loss probability;

The HVAC system was controlled by a two-layer networked learning control system with a cubic spline interpolator under 6.36% loss probability;

The HVAC system was controlled by a two-layer networked learning control system with ZOH under 3.64% loss probability;

The HVAC system was controlled by two-layer networked

learning control system with ZOH under 6.36% loss probability.

(3) Inputs to RBF neural networks were $C(t), T_{ao}, f_a, T_{ai}, T_{wi}, T_{wo}$. Every step learning results of RBF neural network were used to tune the parameters $a(t), b(t)$ by using Equations (37) ~ (47).

(4) The output $u(t)$ of fuzzy controllers was a normalized control value ranged from 0 to 1 to specify flow rates from minimum to maximum of the allowed values. This normalized control signal was converted to the control signal for the model $C(t) = 1400 - 730 u(t)$. The control signal $C(t)$ ranged from 670 (for the maximal opening position) to 1400 (for the maximal closing position).

(5) The initial value of the parameters $a(t), b(t)$ were 0.7 and 0.6 respectively. The learning factors h_a and h_b were given at 0.01 and 0.01 respectively. During the centre selection in RBFNN an appropriate tolerance p was given a value of 0.01.

The disturbance trajectories for T_{ai}, T_{wi}, f_a were illustrated in fig. 6. With real-time learning of RBFNN for HVAC systems, the parameters $a(t)$ and $b(t)$ of the fuzzy controller were adaptively modified as shown in fig. 7 and fig. 8, respectively. Then, control signals with tuning parameters $a(t)$ and $b(t)$ were demonstrated in fig. 9. Finally, the output temperatures of air were shown in fig. 10.

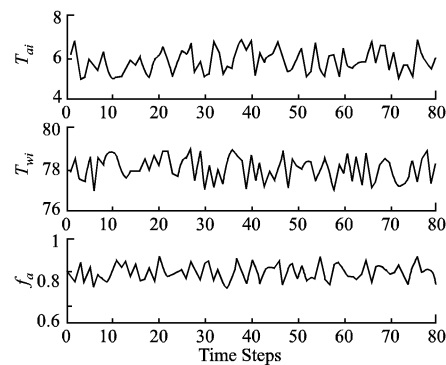


Fig. 6 Disturbances

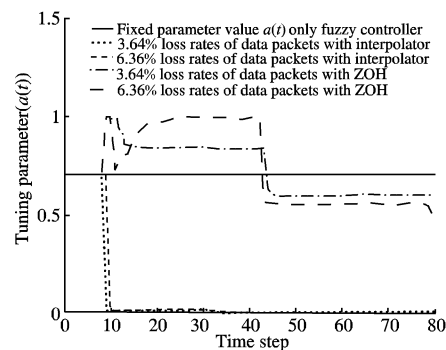
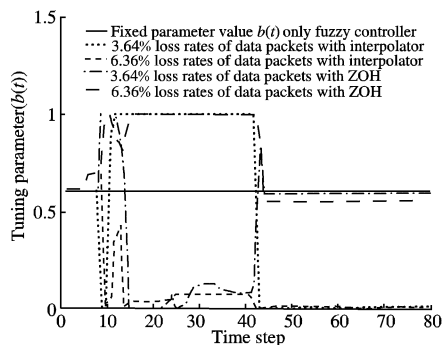
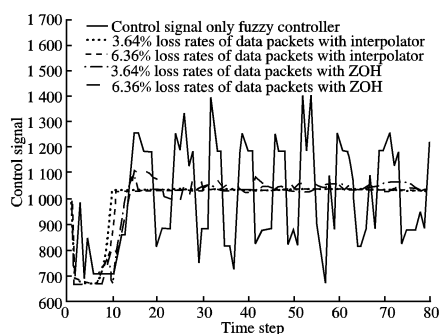
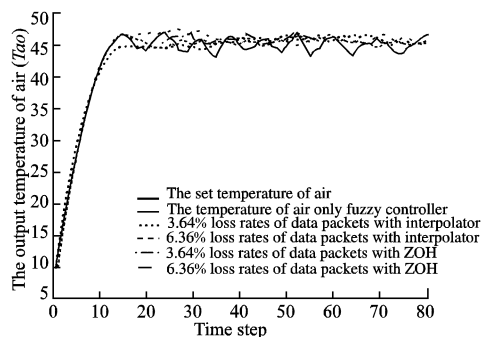


Fig. 7 Tuning parameters $a(t)$

Fig. 8 Tuning parameters $b(t)$ Fig. 9 Control signal $(C(t))$ Fig. 10 The output temperature of air (T_{ao})

From fig. 6 to fig. 10, the following results could be observed.

Comparing with the fixed parameter values, the self-learning algorithm for fuzzy controllers successfully tuned the parameters $a(t)$ and $b(t)$ every step, based on learning results of RBF neural networks in fig. 7 and fig. 8.

According to fig. 9, it is found that the control signal achieves the maximal opening position at initial stage, and then the stable opening position and slightly tunes to reject the disturbance in the stable stage. However, after achieving the set value of the output temperature of air, the control signal fluctuates strongly to reject the disturbances only by fuzzy controllers, which cannot fulfil the needs for higher quality

control performance shown in fig. 10.

Comparing the output temperatures of the air under five cases in fig. 10, the control performance by the proposed two-layer networked learning control system were better than the control performance by fuzzy controllers alone. The control performance by using a cubic spline interpolator was better than the control performance by ZOH.

It should be noticed that the proposed scheme has not only produced superior control performance by cubic spline interpolator compensation, but also achieved better interference rejection for the disturbances T_{ai}, T_{wi}, f_a .

5 Conclusions

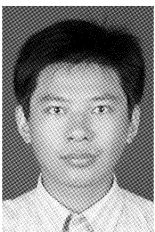
This paper presents a two-layer networked learning control system to deal with the problems associated with network transmission, such as unreliable data packets, network-induced delays, and in the worst situation, data lost. A novel RBFNN was employed in the upper layer for a learning agent to dynamically tune the parameters of fuzzy controllers in the lower layer. The effectiveness and online learning capabilities of the proposed system were demonstrated with experimental studies of a HVAC system. In the next stage of our research, the proposed system will be implemented in a real system to further verify its robustness and performance.

References

- [1] GAID M E M B, CELA A, HAMAM Y. Optimal integrated control and scheduling of networked control systems with communication constraints: Application to a car suspension system[J]. IEEE Transaction on Control Systems Technology, 2006, 14(4): 776-787.
- [2] MONOTOSH D, GHOSH R, GOSWAMI B, et al. Network control system applied to a large pressurized heavy water reactor[J]. IEEE Transactions on Nuclear Science, 2006, 53(5): 2948-2956.
- [3] WEI W B, PAN Y D, FURUTA K. Internet-based tele-control system for wheeled mobile robot[C]. Proceedings of the IEEE International Conference on Mechatronics & Automation, Niagara Falls, 2005: 1151-1156.
- [4] PAN Y J, MARQUEZ H J, CHEN T. Sampled-data iterative learning control for a class of nonlinear networked control systems[C]. Proc dions of the 2006 American Control Conference Minneapolis, 2006: 3494-3499.
- [5] LI Z, FANG H J. A novel controller design and evaluation for networked control systems with time-variant delays [J]. Journal of the Franklin Institute, 2006, 343(2): 161-167.

- [6] YANG T C. Networked control system: A brief survey [C]. IEE Proc. Control Theory Appl., 2006, 153(4): 403-412.
- [7] YANG T C, YU H, FEI M R, et al. Networked control systems: a historical review and current research topics [J]. Measurement + Control, 2005, 38(1): 11-17.
- [8] LI Y Q, FANG H J. Control methodologies of large delays in networked control systems[C]. Proceeding of the 5th International Conference on Control and Automation, ICCA05, 2005: 1225-1230.
- [9] NILSSON J. Real-time control systems with delays[D]. Sweden: Lund Institute of Technology, 1998.
- [10] LEI Z, DIMITRIOS H V. Communication and control co-design for networked control systems[J]. Automatica, 2006, 42(6): 953-958.
- [11] DYER S A, HE X. Cubic-spline interpolation: Part2 [J]. IEEE Instrumentation and Measurement Magazine, 2001, 4(2): 34-36.
- [12] ABDELNOUR G M, CHANG C H, HUANG F H, et al. Design of a fuzzy controller using input and output mapping factors[J]. IEEE Transactions on Systems, Man, and Cybernetics, 1991, 21(5): 952-960.
- [13] CHEN S. Orthogonal least square learning algorithm for radial basis function networks[J]. IEEE Transactions on Neural Networks, 1992, 2(2): 302-309.
- [14] LI K, PENG J X, IRWIN G W. A fast nonlinear model identification method [J]. IEEE Transactions on Automatic Control, 2005, 50(8): 1211-1216.
- [15] LI K, PENG J X, BAI E W. A two-stage algorithm for identification of nonlinear dynamic systems[J]. Automatica, 2006, 42(7): 1189-1197.
- [16] TASHTOUSH B, MOLHIM M, ROUSAN M A. Dynamic model of a HVAC system for control analysis[J]. Energy, 2005, 30(10): 1729-1745.

Biographies



Du Dajun, male, was born in June 1978, obtained BSc degree in 2002 and MSc degree in 2005, both from Zhengzhou University, China. He is a doctoral student in Shanghai University now, and his main research interests are networked learning control and intelligent control.

E-mail: ddj559@163.com



Fei Minrui received his PhD degree in control theory and control engineering from Shanghai University in 1997. Since 1998, he has been a professor and doctoral supervisor in Shanghai University. His current research interests are in the areas of intelligent control, networked learning control systems, wireless sensor networks, etc. He is a vice-chairman of Chinese Association for System Simulation, a director of China Instrument & Control Society, and a director of Chinese Artificial Intelligence Association.

E-mail: mrfei@staff.shu.edu.cn



Hu Huosheng received his PhD degree from Oxford University, U. K. He is a professor in computer science in University of Essex, United Kingdom, leading the Human Centred Robotics Group. He is also a Zhiqiang professor in Shanghai University. His research interests include biologically-inspired robotics, service robots, human-robot interaction, evolutionary robotics, data fusion, artificial life, networked systems, pervasive computing, and RoboCup. Also, he is a Chartered Engineer, a senior member of IEEE, and a member of IET, AAAI, IAS, IASTED and ACM.

E-mail: hhu@essex.ac.uk



Li Lixiong received his PhD degree in control theory and engineering from Shanghai University in 2005. He is now a lecturer in Shanghai University. His research interests include networked control systems, adaptive control systems and intelligent control strategies.

E-mail: lxli@staff.shu.edu.cn