Improved neural network-based adaptive tracking control for manipulators with uncertain dynamics

Dong-hui Wang\textsuperscript{1} and Shi-jie Zhang\textsuperscript{2}

Abstract
In this article, a robust adaptive tracking controller is developed for robot manipulators with uncertain dynamics using radial basis function neural network. The design of tracking control systems for robot manipulators is a highly challenging task due to external disturbance and the uncertainties in their dynamics. The improved radial basis function neural network is chosen to approximate the uncertain dynamics of robot manipulators and learn the upper bound of the uncertainty. The adaptive law based on the Lyapunov stability theory is used to solve the uniform final bounded problem of the radial basis function neural network weights, which guarantees the stability and the consistent bounded tracking error of the closed-loop system. Finally, the simulation results are provided to demonstrate the practicability and effectiveness of the proposed method.

Keywords
Robot manipulators, neural network, adaptive, uncertain dynamics, tracking control

Date received: 2 April 2020; accepted: 11 July 2020

Introduction
With the development of robot technologies, the tracking control of robot manipulators becomes the focus of research during recent years. The characteristic of robot manipulators is that the dynamics function are typical multivariable and nonlinear, and the system dynamics function are usually affected by many uncertain factors, such as external disturbances, friction term, unmodeled dynamics, and load fluctuation. Therefore, the accurate dynamics model equations of robot manipulators are difficult to build in practical applications. Accordingly, precise tracking control performances become difficult to implement. To overcome the tracking control problem of robot manipulators, various control strategies have been applied, as shown in the literature.\textsuperscript{1–10} The adaptive control method has been widely used because it can continuously modify its control behavior according to the complex characteristics of the controlled object, environmental interference, and modeling errors. Thereby, it is easy to achieve the satisfied control performances by using adaptive control approach. In the study of Peng and Liu,\textsuperscript{11} a stabilization tracking controller with hybrid scheme which combines an adaptive compensator with computed torque controller is proposed for robot manipulators with external disturbances and dynamic uncertainties. In the study of Mustafa et al.,\textsuperscript{12} an adaptive back-stepping sliding mode control method was used to

\textsuperscript{1} College of Mechanical and Electronical Engineering, Henan Polytechnic, Zhengzhou, China
\textsuperscript{2} College of Electrical Engineering, Henan University of Technology, Zhengzhou, China

Corresponding author:
Shi-jie Zhang, College of Electrical Engineering, Henan University of Technology, Zhengzhou 450052, China.
Email: zhangshijie@haut.edu.cn
reduce the influence of dynamic uncertainties and external disturbances for trajectory tracking of manipulators. In the study of Wang,\textsuperscript{13} two different adaptive control strategies were proposed to realize the satisfied trajectory tracking performance under the uncertain dynamics and kinematics functions. Similar research can be referred to the literature.\textsuperscript{14–17} However, there still exist many problems in practical applications when using adaptive control method, such as it is difficult to determine the initial value and estimate range of parameter estimation. In addition, it is necessary to linearize the dynamics equations in the controller design process, and a large number of calculations are needed to determine the regression matrix. Thus the application is restricted by the huge computational complexity in practice.

Radial basis function (RBF) neural network (NN) was firstly proposed by Broomhead and Lowe,\textsuperscript{18} and due to the strong nonlinear approximation capability and fast convergence rate, RBFNN-based adaptive tracking control for robotic manipulators has received considerable attention. In the study of Sun et al.,\textsuperscript{19} an NN-based adaptive controller with a linear observer was proposed for the trajectory tracking of robotic manipulators with unknown dynamics, and only joint angle position information was used in the controller design. In the study of Ge et al.,\textsuperscript{20} the RBF network-based adaptive controller was proposed to estimate coefficient matrices for the dynamics function of robot manipulators in task space. In the study of Zhang et al.,\textsuperscript{21} a neural variable structure control method with redundancy capability was proposed to guarantee the stability of the systems with suddenly changing parameters. In the study of Ngo et al.,\textsuperscript{22} a robust control scheme based on NN is proposed. The NN was used to compensate the parameters of disturbance and friction, such that higher control accuracy can be achieved.

In the previous NN-based adaptive control strategy, there are few research studies on the estimation of uncertain dynamics of the system by adaptive controller, therefore, the NN-based adaptive controller has to be retrained redundantly to improve reliability. However, this causes a great deal of computational burden, and it is not applicable for the real-time control systems. Inspired by this, an improved RBFNN was constructed firstly by modifying the clustering radius with variable step to learn the uncertain dynamics of the system online, then robust controller is designed to compensate for the uncertainty, and the stability and tracking performance are guaranteed.

This article is organized as follows. The second section introduces the tracking problem under the uncertainty dynamic of robot manipulators. The structure of improved RBFNN is described in the third section. The fourth section presents the adaptive NN controller. The fifth section provides simulation results of two-link manipulator. Finally, the sixth section concludes the article with a brief summary.

### Problem statement

The dynamics function of an $n$-link rigid robot manipulator can be expressed in the following Lagrange form

$$ M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + \tau_d = \tau $$  \hspace{1cm} (1)

where $q$ denotes the joint variable vector and $M(q)$, $C(q, \dot{q})$, and $G(q)$ are the inertia matrix, the Coriolis matrix, and the gravity vector, respectively. $\tau_d$ denotes the unknown disturbances, and the control input torque is $\tau$.

Given a desired trajectory $q_d$, the tracking error $e = q - q_d$, $\dot{e} = \dot{q} - \dot{q}_d$. If the dynamics function of the robot manipulator is known exactly, then the control law can be designed as

$$ \tau = M(q)(\ddot{q}_d - k_v\dot{e} - k_pe) + C(q, \dot{q})\dot{q} + G(q) $$  \hspace{1cm} (2)

where $k_v = k_v^T$ and $k_p = k_p^T$ are gain matrix, generally chosen diagonal.

Substituting the control law (2) to system (1) yields

$$ \ddot{e} + k_v \dot{e} + k_p e = 0 $$  \hspace{1cm} (3)

However, it is difficult to get accurate $M(q)$, $C(q, \dot{q})$, and $G(q)$ in practice, so the nominal model is usually used to design the controller. Denote the nominal model of the robot manipulators as

$$ M_0(q)\ddot{q} + C_0(q, \dot{q})\dot{q} + G_0(q) + \tau_d = \tau $$  \hspace{1cm} (4)

For the nominal model (4), the control law can be designed as

$$ \tau = M_0(q)(\ddot{q}_d - k_v\dot{e} - k_pe) + C_0(q, \dot{q})\dot{q} + G_0(q) $$  \hspace{1cm} (5)

Substituting the control law (5) to system (1) yields

$$ M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + \tau_d = M_0(q)(\ddot{q}_d - k_v\dot{e} - k_pe) + C_0(q, \dot{q})\dot{q} + G_0(q) $$  \hspace{1cm} (6)

Denote $\Delta M = M_0 - M$, $\Delta C = C_0 - C$, $\Delta G = G_0 - G$, then equation (6) can be rewritten as

$$ \ddot{e} + k_v \dot{e} + k_pe = M_0^{-1}(\Delta M\ddot{q} + \Delta C\dot{q} + \Delta G + \tau_d) $$  \hspace{1cm} (7)

It can be seen from equations (3) and (7) that the inaccurate modeling will cause a decrease in control performance, therefore, it is necessary to estimate the inaccurate part of the modeling.

Let $x = (e, \dot{e})^T$, and denote the uncertain dynamics term as

$$ f = M_0^{-1}(\Delta M\ddot{q} + \Delta C\dot{q} + \Delta G + \tau_d) $$  \hspace{1cm} (8)

Then the error state equation under the control law (5) is

$$ \dot{x} = Ax + Bf $$  \hspace{1cm} (9)

where

$$ A = \begin{bmatrix} 0 & I \\ -k_p & -k_v \end{bmatrix}, B = \begin{bmatrix} 0 \\ I \end{bmatrix} $$
Considering that the uncertain dynamics term $f$ is known, the modified control law can be designed as

$$
\tau = M_0(q)(\ddot{q} - k_c\dot{q} - k_p e) + C_0(q, \dot{q})\dot{q} + G_0(q) - M_0(q)f
$$

(10)

By substituting equation (10) into equation (1), a stable closed-loop system (3) can be obtained.

As stated previously, the tracking control problem in this article can be summarized as follows: Given dynamics function (1) and the desired trajectory $q_d$, design control law (10) to guarantee the tracking error $e \to 0$ and $\dot{e} \to 0$, in which an improved RBFNN will be used in the control law (10) to provide the estimate $f$ for the unknown robot uncertain dynamics adaptively.

### Improved RBFNNs

A basic RBFNN structure can be determined after off-line learning, such as subtractive clustering method proposed by Chiu. In this section, in order to realize online identification of the network and obtain good network identification performance for nonlinear real-time systems, an improved algorithm is presented by using nearest neighbor clustering algorithm during online learning, and the clustering radius is modified with variable step. The algorithm of RBFNNs is modified with variable step. The algorithm during online learning, and the clustering radius algorithm is presented by using nearest neighbor clustering algorithm during online learning, and the clustering radius is modified with variable step. The algorithm of RBFNNs online learning is as follows.

**Step 1:** Calculate the Euclidean distance between input sample data

$$d[i,j] = ||X_i - X_j||$$

(11)

Then, take the minimum cluster radius $r_{\text{min}} = d[i,j]_{\text{min}}$ and the maximum clustering radius $r_{\text{max}} = d[i,j]_{\text{max}}$. Select a suitable cluster radius $r \in (r_{\text{min}}, r_{\text{max}})$. Denote $A(l)$ as set of the output vector and $C(l)$ as a counter to count the number of samples belonging to different clusters.

**Step 2:** Suppose that $M$ hidden layer node has been determined, that is, the RBFNN has $M$ cluster center, which is recorded as $c_1, c_2, \ldots, c_m$. For sample data $X_k, k = m, \ldots, n$ collected during online operation, calculate the Euclidean distance between $X_k$ and $c_i, \forall i = 1, 2, \ldots, m$, and $k = m, m + 1, \ldots, n$, then find the minimum distance

$$d_{\text{min}} = ||X_k - c_i||$$

(12)

If $d_{\text{min}} < r$, let

$$A(j) = A(j) + y^k$$
$$C(j) = C(j) + 1$$

$A(i)$ and $C(i)$ remain unchanged, and the weights from hidden layer to output layer as

Figure 1. Comparison of the improved RBFNN and the basic RBFNN. RBFNN: Radial basis function neural network.

$$W(i) = \frac{A(i)}{C(i)}, \forall i = 1, 2, \ldots, M$$

If $d_{\text{min}} > r$, a new cluster center is added as $c_{m+1} = X_k, M = M + 1, A(i)$ and $C(i)$ remain unchanged, $A(M) = y_k, C(M) = 1$, and the weight from hidden layer to output layer as

$$W(M) = \frac{A(M)}{C(M)}$$

**Step 3:** Using Gauss function as basis function, the output of the RBFNN constructed according to the above rules is

$$f(X_k) = \frac{\sum_{i=1}^{M} W_i \exp \left( -\frac{||X_k - c_i||^2}{r^2} \right)}{\sum_{i=1}^{M} \exp \left( -\frac{||X_k - c_i||^2}{r^2} \right)}$$

(13)

**Step 4:** Utilizing equation (14) to calculate the error performance index of the whole NN

$$E = \frac{1}{2} \sum_{i=1}^{N} (y_i - f(X_i))^2$$

(14)

where $X_i$ and $y_i$ are input signals and output signals, respectively. If $E < E_0$, end the learning, otherwise let $r = r - h$, and turn to Step 2. $E_0$ is the predefined threshold, and $h$ is variable step according to the error.

In order to verify the effectiveness of the improved RBFNN, a curve fitting example is used to compare with the basic RBFNN. The blue circles in Figure 1 represent the data.
to be fitted. It can be seen that the improved algorithm and the traditional algorithm can meet the same requirements in fitting accuracy, but the improved algorithm only needs 0.654 s to complete fitting, while the traditional algorithm takes 3.681 s to complete the calculation which is shown in Figure 2.

Remark 1. In the conventional nearest neighbor clustering algorithm, if the error performance index is not satisfied, the clustering radius \( r \) is decreasing with a fixed step \( h \). When the identification error is large but the step is very small, the learning speed of the NN will be slower. In order to solve this problem, we can adjust the cluster radius by variable step according to the error, when the error is large, increasing the step size to speed up the learning speed, when the error is close to the performance index, take a small step, to avoid overlearning.

Based on the above algorithm, an improved RBFNN can be constructed to provide the estimate of the robot uncertain dynamics function (8).

**NN-based adaptive tracking controller design**

In this section, an NN-based adaptive trajectory tracking control algorithm is proposed for a rigid robot with uncertain dynamics. The computational torque method is used to control the nominal model, and robust control is used to compensate for the uncertainty. The RBFNN is used to learn and approximate the upper bound of the uncertainty of the robust controller.

For system (1), using the RBFNN, the controller can be designed as follows

\[
\tau = M_0(q)(\ddot{q}_d - k_r \dot{e} - k_pe) + C_0(q, \dot{q})\dot{q} + G_0(q) - M_0(q)\hat{f}(x, W)
\]

(15)

where \( \hat{f}(x, W) = \hat{W}^T \varphi(x) \), \( \varphi(x) \) are the output of Gauss function, \( \hat{W} \) is the estimated value of \( W \).

Substituting the control law (15) to system (1) yields

\[
M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + \tau_d = M_0(q)(\ddot{q}_d - k_r \dot{e} - k_pe) + C_0(q, \dot{q})\dot{q} + G_0(q) - M_0(q)\hat{f}(x, W)
\]

(16)

Utilizing \( M_0(q)\ddot{q} + C_0(q, \dot{q})\dot{q} + G_0(q) \) to the left and right sides of equation (16), we have

\[
\Delta M(q)\ddot{q} + \Delta C(q, \dot{q})\dot{q} + \Delta G(q) + \tau_d = M_0(q)\ddot{q} - M_0(q)(\ddot{q}_d - k_r \dot{e} - k_pe) + M_0(q)\hat{f}(x, W)
\]

(17)

that is

\[
\ddot{e} + k_v \dot{e} + k_pe + \hat{f}(x, W) = f(x)
\]

(18)

Let \( x = (e, \dot{e})^T \), and use equation (8), equation (18) can be written as follows

\[
\dot{x} = Ax + B\left(f(x) - \hat{f}(x, W)\right)
\]

(19)

Considering that

\[
\hat{f}(x) - \hat{f}(x, W)
\]

\[
= f(x) - f(x, W^*) + \hat{f}(x, W^*) - \hat{f}(x, W)
\]

\[
= \zeta + \hat{W}^T \varphi(x) - \hat{W}^T \varphi(x)
\]

\[
= \zeta - \hat{W}^T \varphi(x)
\]

where \( \hat{W} = \hat{W} - W^* \). Then we get

\[
\dot{x} = Ax + B\left(\zeta - \hat{W}^T \varphi(x)\right)
\]

(20)

Denote the Lyapunov function as

\[
V = \frac{1}{2}x^TPx + \frac{1}{2}\kappa||\hat{W}||^2
\]

(21)

where \( \kappa > 0 \), \( P \) is symmetric positive definite matrix, and satisfy the Lyapunov equation

\[
PA + A^TP + Q = 0
\]

(22)

where \( Q \geq 0 \).

By deriving the Lyapunov function

---

**Figure 2.** Calculation time of the improved RBFNN and the basic RBFNN. RBFNN: Radial basis function neural network.
\[
\dot{V} = \frac{1}{2}(x^TPx + \dot{x}^TPx) + \frac{1}{\kappa} \text{tr}\left(\dot{W}^T \dot{W}\right)
\]

\[
= \frac{1}{2}(x^T(Ax + B(-\ddot{W}^T \varphi(x) + \zeta)) + (x^T A^T + (-\ddot{W}^T \varphi(x) + \zeta)^T B^T) Px) + \frac{1}{\kappa} \text{tr}\left(\dot{W}^T \dot{W}\right)
\]

\[
= \frac{1}{2}\left(x^T validations\right) + \dot{x}^TPx + \frac{1}{\kappa} \text{tr}\left(\dot{W}^T \dot{W}\right)
\]

Due to \(\dot{\varphi}^T(x) B^TPx = \text{tr}(B^TPx \varphi^T(x) \ddot{W})\), then we have

\[
\dot{V} = -\frac{1}{2} x^T Q x + \frac{1}{\kappa} \text{tr}\left(-\kappa B^TPx \dot{\varphi}^T(x) \ddot{W} + \dot{W}^T \dot{W}\right) + \zeta B^TPx
\]

The adaptive law in this article is designed as follows

\[
\dot{W} = \kappa \varphi^T PB + \alpha \kappa |x| \dot{W}
\]

Substituting the adaptive law (24) to (23) yields

\[
\dot{V} = -\frac{1}{2} x^T Q x + \frac{1}{\kappa} \text{tr}\left(\alpha \kappa |x| \dot{W}^T \dot{W}\right) + \zeta B^TPx
\]

In order to ensure \(\dot{V} \leq 0\), the following conditions need to be satisfied

\[
\frac{1}{2} \lambda_{\min}(Q) |x|^2 \geq \frac{\alpha}{4} W^2_{\max} + \|\zeta\| \lambda_{\max}(P)
\]

which means the convergence conditions are

\[
|x| \geq \frac{2}{\lambda_{\min}(Q)} \left(\frac{\alpha}{4} W^2_{\max} + \|\zeta\| \lambda_{\max}(P)\right)
\]

As stated previously, the following theorem which gives the solution to the tracking control problem in the second section can be easy to get.

**Theorem 1.** Consider the dynamics function of an n-link rigid robot manipulator (1), choose the control law (15) and the RBFNN adaptive law (24), then the tracking error and the convergence of the closed-loop system parameters can be achieved.

**Simulation results**

The two-link robot manipulator model shown in Figure 3 is considered in this section to illustrate the efficiency of the controller designed previously.

The dynamic equation is described as (1), in which the inertia matrix, the Coriolis matrix, and the gravity vector are
\[
M(q) = \begin{bmatrix}
(m_1 + m_2)l_1^2 + m_2l_2^2 + 2m_2l_1l_2\cos q_2 & m_2l_2^2 + m_2l_1l_2\sin q_2 \\
m_2l_2^2 + m_2l_1l_2\cos q_2 & m_2l_2^2 \\
\end{bmatrix}
\]
\[
C(q, \dot{q}) = \begin{bmatrix}
-m_2l_1\dot{q}_2\sin q_2 & -m_2l_1(\dot{q}_1 + \dot{q}_2)\sin q_2 \\
m_2l_1\dot{q}_2\sin q_2 & 0 \\
\end{bmatrix}
\]
\[
G(q) = \begin{bmatrix}
(m_1 + m_2)g\cos q_2 + m_2g(\sin q_1 + q_2) \\
m_2g\cos(q_1 + q_2) \\
\end{bmatrix}
\]

where the parameters of two-link robot manipulator are given as \(m_1 = 3\, \text{kg}, \ m_2 = 1\, \text{kg}, \ l_1 = 0.8\, \text{m}, \) and \(l_2 = 0.5\, \text{m}.

The external disturbance is expressed as \(\tau_d = [0.1\sin(2t) \ 0.5\sin(3t)]^T.\) The desired trajectory is expressed as \(q_{1d} = 1 + 0.5\sin(0.2\pi t), \ q_{2d} = 1 - 0.5\cos(0.2\pi t).\) The initial states are expressed as \([0.8 \ 0.5 \ 1 \ 0.3]^T.\) The range of \(\Delta M, \ \Delta C, \ \Delta G\) is 10\%. The parameters used in the adaptive control law are \(\kappa = 15, \ \alpha = 0.001,\)

\[
k_p = \begin{bmatrix}
16 & 0 \\
0 & 16 \\
\end{bmatrix}, \ k_v = \begin{bmatrix}
8 & 0 \\
0 & 8 \\
\end{bmatrix}, \ Q = 48I_{4 \times 4}.
\]

Simulation results of position tracking, control input of joint 1 and joint 2, and the estimations of the uncertain dynamics are shown in Figures 4 to 9. From these results, we can find that the proposed control scheme is effective.

In Figures 10 and 11, the simulation results based on proportional-derivative (PD) control are given. It can be

---

**Figure 3.** Two-link robot manipulator.

**Figure 4.** Position tracking responses of joint 1.

**Figure 5.** Position tracking responses of joint 2.

**Figure 6.** Control input responses of joint 1.
seen from these figures that the traditional PD control cannot accurately track the desired trajectory in the case of modeling uncertainty and external interference. The controller designed in this article can solve this problem well.

Conclusions

In this article, an improved RBFNN adaptive control method was proposed to eliminate the effects of uncertain dynamics. An improved RBFNN was constructed firstly by modifying the clustering radius with variable step to learn the uncertain dynamics of the system online. Then robust controller is designed to compensate for the uncertainty, and the degree of uncertainty and external disturbance of the system are not required to be estimated beforehand. The proposed adaptive control law guarantees the boundedness of weights and solves the bounded upper bound problem of the weights of NNs. Finally, simulation results show that the proposed method works well in the joint tracking control of two-link robot manipulator.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.
Funding
The author(s) received no financial support for the research, authorship, and/or publication of this article.

ORCID iD
Shi-jie Zhang https://orcid.org/0000-0002-2061-2720

References
1. Zou Y, Wang Y, Liu X, et al. Neural network robust H\(_1\) tracking control strategy for robot manipulators. *Appl Math Model* 2010; 34(7): 1823–1838.
2. Alavandar S and Nigam MJ. Neuro-fuzzy based approach for inverse kinematics solution of industrial robot manipulators. *Int J Comput Commun Control* 2008; 3(3): 224–234.
3. Islam S and Liu PX. Robust sliding mode control for robot manipulator. *IEEE Trans Ind Electron* 2011; 58(6): 2444–2453.
4. Ciliz MK. Adaptive control of robot manipulators with neural network based compensation of frictional uncertainties. *Robotica* 2005; 23(2): 159–167.
5. Chen Q, Meng-Meng YU, and Wei Q. Adaptive neural network control of robotic manipulators servo system with friction and dead zone compensation. *J New Ind* 2016; 338: 391–404.
6. Corradini ML, Giantomassi A, Ippoliti G, et al. Robust control of robot arms via quasi sliding modes and neural networks. In: *Advances and applications in sliding mode control systems*. Cham: Springer International Publishing, 2015, pp. 79–105.
7. Zheng D, Na J, Ren X, et al. Adaptive control of robotic servo system with friction compensation. In: *Proceedings of IEEE conference on robotics, automation and mechatronics*, Qingdao, China, 17–19 September 2011, pp. 285–290. IEEE.
8. Pham CV and Wang YN. Robust adaptive trajectory tracking sliding mode control based on neural networks for cleaning and detecting robot manipulators. *J Intell Robot Syst* 2015; 79(1): 101–114.
9. Cuong PV and Wang YN. Adaptive trajectory tracking neural network control with robust compensator for robot manipulators. *Neural Comput Appl* 2018; 27(2): 525–536.
10. Moradi M and Malekizade H. Neural network identification based multivariable feedback linearization robust control for a two-link manipulator. *J Intell Robot Syst* 2018; 72(2): 167–178.
11. Peng J and Liu Y. Adaptive robust quadratic stabilization tracking control for robotic system with uncertainties and external disturbances. *J Control Sci Eng* 2014; 43(9): 1311–1318.
12. Mustafa A, Dhar NK, Agrawal P, et al. Adaptive backstepping sliding mode control based on nonlinear disturbance observer for trajectory tracking of robotic manipulator. In: *Proceedings of the 2nd international conference on control and robotics engineering*, Bangkok, Thailand, 1–3 April 2017, pp. 29–34. IEEE.
13. Wang HL. Adaptive control of robot manipulators with uncertain kinematics and dynamics. *IEEE Trans Autom Control* 2018; 62(2): 948–954.
14. Gao YF, Sun XM, Wen C, et al. Adaptive tracking control for a class of stochastic uncertain nonlinear systems with input saturation. *IEEE Trans Autom Control* 2019; 62(5): 2498–2504.
15. Chen Y, Wang K, Zhai L, et al. Feedforward fuzzy trajectory compensator with robust adaptive observer at input trajectory level for uncertain multi-link robot manipulators. *J Franklin Inst* 2017; 4(9): 1311–1318.
16. Jin X. Adaptive finite-time tracking control for joint position constrained robot manipulators with actuator faults. In: *Proceedings of the American control conference*, Boston, MA, USA, 6–8 July 2016, pp. 6018–6023. IEEE.
17. Zhao D, Li S, and Zhu Q. Adaptive synchronised tracking control for multiple robotic manipulators with uncertain kinematics and dynamics. *Int J Syst Sci* 2018; 47(4): 791–804.
18. Broomhead DS and Lowe D. Multivariable functional interpolation and adaptive networks. *Complex Syst* 1988; 2(3): 321–355.
19. Sun FC, Sun ZQ, and Woo PY. Neural network-based adaptive controller design of robotic manipulators with an observer. *IEEE Trans Neural Netw* 2001; 12(1): 54–67.
20. Ge SS, Hang CC, and Woon LC. Adaptive neural network control of robot manipulators in task space. *IEEE Trans Ind Electron* 1997; 44(6): 746–752.
21. Zhang WH, Qin M, Ma J, et al. Neural integrated control for a free-floating space robot with suddenly changing parameters. *Sci China Inform Sci* 2019; 54(10): 2091–2099.
22. Ngo TQ, Wang YN, Mai TL, et al. Robust adaptive neural-fuzzy network tracking control for robot manipulator. *Int J Comput Commun Control* 2017; 7(2): 341–352.
23. Chiu SL. Fuzzy model identification based on cluster estimation. *J Intell Fuzzy Syst* 1994; 2(3): 756–759.