RBF Neural Networks-Based Near-Optimal Tracking Control of Partially Unknown Discrete-Time Nonlinear Systems

Huang Jiashun , Xu Dengguo * , Li Yahui , Ma Yan

Posted Date: 1 March 2024

doi: 10.20944/preprints202403.0016.v1

Keywords: adaptive dynamic programming; optimal tracking control; RBF neural network; nonlinear systems

 doi:10.20944/preprints202403.0016.v1

Article

RBF Neural Networks-Based Near-Optimal Tracking Control of Partially Unknown Discrete-Time Nonlinear Systems

Jiashun Huang [1], Dengguo Xu [1,2,*], Yahui Li [1] and Yan Ma [1]

[1] School of Automation, Guangxi University of Science and Technology, Liuzhou 54500, China; 18707519318@163.com
[2] School of Physics and Electrical Engineering, Liupanshui Normal University, liupanshui 553000, China
* Correspondence: dengguoxu@163.com

Abstract: This paper proposed an optimal tracking control scheme through adaptive dynamic programming(ADP) for a class of partially unknown discrete-time nonlinear systems based on radial basis function neural network(RBF-NN). In order to acquire the unknown system dynamics, we use two RBF-NNs, the one is used to construct the identifier, and the another is used to directly approximate the steady-state control input, where a novel adaptive law is proposed to update neural network weights. While the optimal feedback control and the cost function are derived via feedforward neural networks approximating, it is proposed to regulate the tracking error, the critic network and the actor network are then trained online to obtain the solution of the associated Hamilton–Jacobi–Bellman (HJB) equation being built under the ADP framework. Simulations verify the effectiveness of the optimal tracking control technique using the neural networks.

Keywords: adaptive dynamic programming(ADP); optimal tracking control; RBF neural network(RBF-NN); nonlinear systems

1. Introduction

As is widely known, nonlinear system control is an important topic of control fields, especially for uncertainly unknown nonlinear systems, which is difficult for traditional control methods. Until 1988, radial basis function neural networks were proposed [1]. Immediately following in 1900, Narendra, K. S. and K. Parthasarathy first proposed an artificial neural network adaptive control method for nonlinear dynamical systems [2]. Since then, multilayer neural networks (MNN) and radial basis function (RBF) neural networks were successfully applied in pattern recognition and control systems [3]. Compared to multilayer feedforward networks (MFNs), the RBF neural networks attracted much attention due to their good generalization ability, simple network structure, and avoidance of unnecessary and lengthy computations. Studies on RBF-NNs have shown the ability of neural networks to approximate any nonlinear function with a compact set and arbitrary accuracy[4,5]. Many research results have been published on neural network control for nonlinear systems [6,7].

On the other hand, optimal tracking control as one of the effective methods for nonlinear systems in optimal control, received many practical engineering applications [8–10]. Therefore, exploring the optimal tracking optimal control for nonlinear systems possesses significant theoretical importance and practical value. For optimal control methods for nonlinear systems, the difficulty lies in the requirement of solving the nonlinear Hamilton-Jacobi-Belman (HJB) equation, which is usually difficult to solve analytically. Although dynamic programming is an effective method for solving optimal control problems, there is the problem of "curse of dimensionality" when facing relatively complex systems [11,12].

Faced with the difficult problem of solving nonlinear Hamilton-Jacobi-Bellman partial differential equations exactly, several methods was proposed to approximate the solutions of the Hamilton-Jacobi-Bellman equations, These include the use of reinforcement learning [8,13–19] and back-propagation through time [20]. Among these classical RL methods, combining the advantages of adaptive control and optimal control, the ADP algorithm was considered as one of the core methods for realizing optimal control strategies for the diversity of optimal control problems, and it has been successfully

 doi:10.20944/preprints202403.0016.v1

applied to both continuous-time systems [21–23] and discrete-time systems [24–28] to search for solutions of the HJB equations online. Numerous ADP and RL approaches emerged, such as robust ADP [29,30] iterative/invariant ADP [31–33], spiking/Hamiltionian-driven ADP [34,35], integral RL [36,37], and off-policy RL [38–40]. Several works have attempted to solve the discrete time nonlinear optimal regulation problem in a near optimal sense using adaptive dynamic programming through neural networks (NNs) with offline training.

In the past decades, many relevant studies was conducted on the optimal tracking control of discrete-time nonlinear system, such as generalised policy iteration adaptive dynamic programming[41], actor-critic algorithm [42], heuristic dynamic programming (HDP)[43], greedy heuristic dynamic programming iteration algorithm[44] and Q-Learning Algorithm[45]. However, in the known literatures, optimal tracking control methods using RBF-NNs applied to the ADP algorithm are barely used.

In this paper, an optimal tracking control method RBF-NNs-based for discrete-time partially unknown nonlinear systems is proposed, two RBF neural networks are used to approximate the unknown system dynamic as well as the steady-state control, and after transforming the tracking problem into a regulation problem, two feedforward neural networks are used to approximate the critic network and the actor network to obtain the error feedback control, which allows the online learning process to require only current and past system data rather than the exact system dynamics.

The contributions of article are as follows: (1) Unlike classical technique of NNs approximating [42,44–46], we propose an near-optimal tracking control scheme for a class of partially unknown discrete-time nonlinear systems based on RBF-NNs and the stability of systems is proved by the Lyapunov theory. (2) Compared with [41,44], we additionally used an RBF-NN to directly approximate the the steady-state controller of the unknown system, it can solve the requirement for the priori knowledge of the controlled system dynamics and reference system dynamics. (3) For the inverse dynamic NN to directly approximate the steady-state controller of the system, we propose an novel adaptive law to update the weight of the RBF-NN, and the convergence is completed through the selection of constants.

The organization of this paper is as follows. The problem statement is shown in Section II. The technique of the system with partially unknown nonlinear dynamic are designed in Section III, where include the RBF-NN identifier, the RBF-NN steady-state controller, near optimal feedback controller and stability analysis. Simulations and experimental results are provided in Sections IV to validate the proposed control method. Section V draws some conclusions.

2. Problem Statement

In this paper, we consider the following discrete-time nonlinear system:

$$x(k+1) = f[x(k)] + g[x(k)]u(k) \tag{1}$$

where $x(k) \in \mathbb{R}^n$ is the measurable system state and $u(k) \in \mathbb{R}^m$ is the control input. Assume that the nonlinear smooth function $f[x(k)] \in \mathbb{R}^n$ is an unknown drift function, $g[x(k)] \in \mathbb{R}^{n\times m}$ is a known function and $\|g[x(k)]\|_F \leq g_1$ where the Frobenius norm $\|\cdot\|_F$ is applied. In addition, assuming that there exists a matrix $g[x(k)]^+ \in \mathbb{R}^{m\times n}$ such that $g[x(k)]g[x(k)]^+ = I \in \mathbb{R}^{n\times n}$ where I is the identity matrix. Let $x(0)$ be the initial state.

The reference trajectory is generated by the following bounded command:

$$x_d(k+1) = \varphi(x_d(k)) \tag{2}$$

where $x_d(k) \in \mathbb{R}^n$ and $\varphi(x_d(k)) \in \mathbb{R}^n$, and $x_d(k)$ is the reference trajectory, which needs only to be stable in the sense of Lyapunov, not necessarily asymptotically stable.

 doi:10.20944/preprints202403.0016.v1

Let $u(k)$ be an arbitrary sequence of controls from k to infinity. The goal of this paper is to design a controller $u(k)$ that not only ensures the state of system (1) tracks the reference trajectory, but also minimizes the cost function

$$J(e(k), u(k)) = \sum_{k=0}^{\infty} e^T(k)Qe(k) + u^T(k)Ru(k) \tag{3}$$

where $Q \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ are symmetric positive definite; $e(k) = x(k) - x_d(k)$ is tracking error. For common solutions of tracking problems [47], the control input consists of two parts, a steady-state input u_d and a feedback input u_e. Next, we will discuss how obtain each part.

The steady-state of the control input is used to ensure perfect tracking. This perfect tracking equation is realized under the condition $x(k) = x_d(k)$. For this condition to be fulfilled, the steady-state part of the control $u_d(k)$ must exist to make $x(k)$ equivalent $x_d(k)$. By substituting $x_d(k)$ and $u_d(k)$ into system (1), the reference state is

$$x_d(k+1) = f[x_d(k)] + g[x_d(k)]u_d(k) \tag{4}$$

If the system dynamics (1) are known, $u_d(k)$ is acquired by

$$u_d(k) = g[x_d(k)]^+(x_d(k+1) - f[x_d(k)]) \tag{5}$$

where $g[x_d(k)]^+ = (g[x_d(k)]^T g[x_d(k)])^{-1} g[x_d(k)]^T$ is the generalized inverse of $g[x_d(k)]$ with $g[x_d(k)]^+ g[x_d(k)] = I$.

By using (1) and (4), the tracking error dynamics $e(k)$ are given by

$$\begin{aligned} e(k+1) &= f[x(k)] + g[x(k)]u(k) - x_d(k+1) \\ &= f_e(k) + g_e(k)u_e(k) \end{aligned} \tag{6}$$

where $f_e(k) = g(e(k) + x_d(k))g(x_d(k))^+(\varphi(x_d(k)) - f(x_d(k))) + f(e(k) + x_d(k)) - \varphi(x_d(k))$, $u_e(k) = u(k) - u_d(k)$ and $g_e(k) = g[x_d(k) + e(k)]$. $u_e(k) \in \mathbb{R}^m$ is the feedback control input. By minimizing the cost function, it is designed to stabilize the tracking error dynamics. For $e(k)$ under the control sequence, the cost function is defined as

$$\begin{aligned} J_e(e(k), u_e(k)) &= \sum_{k=0}^{\infty} e^T(k)Qe(k) + u_e^T(k)Ru_e(k) \\ &= e^T(k)Qe(k) + u_e^T(k)Ru_e(k) + J_e(e(k+1), u_e(k+1)) \\ &= r(k) + J_e(e(k+1), u_e(k+1)) \end{aligned} \tag{7}$$

where $r(k) = e^T(k)Qe(k) + u_e^T(k)Ru_e(k)$, and $J_e(e(k), u_e(k)) > 0$ for $\forall e(k), u_e(k) \neq 0$. $Q \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ are symmetric positive definite, $x_d(k)$ and $x_d(k+1)$ are bounded to be tracked by the reference trajectory. The tracking error $e(k)$ is used in this study of the cost function of the optimal tracking control problem. This feedback control $u_e(k)$ is found by minimizing (7) to solve the extremum condition in the optimal control framework[8]. This result is

$$u_e^*(k) = -\frac{1}{2}R^{-1}g_e(k)\frac{\partial J(e(k+1))}{\partial e(k+1)} \tag{8}$$

Then the standard control input is obtained

$$u^*(k) = u_d(k) + u_e^*(k) \tag{9}$$

where $u_d(k)$ is obtained from (5), $u_e^*(k)$ is obtained from (8).

 doi:10.20944/preprints202403.0016.v1

Remark 1. *In oreder to acquire the unknown dynamics information in system (1), we used an RBF neural network to reconstruct system dynamics. Therefore, we can use (5) to obtain the steady-state control $u_d(k)$.*

The main results of this paper are based on the following definitions and assumptions.

Assumption 1. *System (1) is controllable, and the system state $x(k) = 0$ is in equilibrium under control $u(k) = 0$. Input control $u(k) = u(x(k))$ satisfies $u(x(k)) = 0$ for $x(k) = 0$, and cost function is a positive definite function for any $x(k)$ and $u(k)$.*

Definition 1. *A control law u_e is admissible with respect to (7) on the set Ω , if u_e is continuous on a compact set $\Omega_u \in \mathbb{R}$ for $\forall e(k) \in \Omega$, $u_e(0) = 0$, and $J(e(0), u_e(\cdot))$is finite.*

Lemma 1. *For the tracking error system (6), assume that $u_e(k)$ be an admissible control and the internal dynamics $f_e(k)$ is bounded, and*

$$\begin{aligned} \|f_e(k)\|^2 \leq \Gamma\lambda_{\min}(Q)\|e(k)\|^2/2 \\ + (\Gamma\lambda_{\min}(R) - 2g_1^2)\|u_e(k)\|^2/2, \end{aligned} \tag{10}$$

where $\lambda_{\min}(R)$ is the minimum eigenvalue of R, $\lambda_{\min}(Q)$ is the minimum eigenvalue of Q, and $\Gamma > 2g_1^2/\lambda_{\min}(R)$ is a known positive constant. Then, the tracking error system (6) be asymptotically stable.

Proof. Considering the following positive definite Lyapunov function,

$$V(k) = e^T(k)e(k) + \Gamma J_e(k) \tag{11}$$

where $J_e(k) = J_e(e(k), u_e(k))$ is defined in (7). Differencing the Lyapunov function yields

$$\Delta V(k) = e^T(k+1)e(k+1) - e^T(k)e(k) + \Gamma(J_e(k+1) - J_e(k)) \tag{12}$$

Using (6) and (7), we can obtain

$$\begin{aligned} \Delta V(k) =& (f_e(k) + g_e(k)u_e(k))^T(f_e(k) + g_e(k)u_e(k)) \\ & - e^T(k)e(k) - \Gamma(e^T(k)Qe(k) + u_e^T(k)Ru_e(k)) \end{aligned} \tag{13}$$

Applying the Cauchy–Schwarz inequality yields

$$\begin{aligned} \Delta V(k) \leq 2\|f_e(k)\|^2 - (\Gamma\lambda_{\min}(R) - 2g_1^2)\|u_e(k)\|^2 \\ -\Gamma\lambda_{\min}(Q)\|e(k)\|^2 - \|e(k)\|^2 \end{aligned} \tag{14}$$

Considering the goal of the tracking error system (6) being asymptotically stable, i.e.,$\Delta V(k) < 0$, we require

$$\begin{aligned} 2\|f_e(k)\|^2 \leq \Gamma\lambda_{\min}(Q)\|e(k)\|^2 \\ + (\Gamma\lambda_{\min}(R) - 2g_1^2)\|u_e(k)\|^2 \end{aligned} \tag{15}$$

Therefore, if the bound in (10) is satisfied, we can get $\Delta V(k) < 0$ and the asymptotic stability of the tracking error system (6) is proved. □

Remark 2. *Lemma 1 shows that under the condition that the internal dynamics $f_e(k)$ is bounded is bounded to satisfy (10), then, for the nonlinear system (6), there exists an admissible control $u_e(k)$ not only stabilizes the system (6) on Ω but also guarantees that $J_e(k)$ is finite.*

 doi:10.20944/preprints202403.0016.v1

3. Optimal Tracking Controller Design with Partially Unknown Dynamic

In this section, firstly, we use an RBF-NN to approximate the unknown system dynamics $f[x(k)]$, and use another RBF-NN to approximate the steady-state controller $u_d(k)$. Secondly, two feedback neural networks are introduced to approximate the cost function and the optimal feedback control $u_e(k)$. Finally, the system stability is proved by selecting an appropriate Lyapunov function.

3.1. RBF-NN Identifier Design

In this subsection, in order to capture the unknown dynamics of the system (1), an RBF-NN-based identifier is proposed. Without losses of generality, this unknown dynamics is assumed to be a smooth function within a compact set. Then this unknown dynamics (1) can be approximated by the RBF-NN as

$$\hat{f}(x(k)) = \hat{w_f}(k)^{\mathrm{T}} h[x(k)] + \triangle_f(x) \tag{16}$$

where $\hat{w_f}(k)$ is the matrix of ideal output weights of the neural network and $h[x(k)]$ is the vector of radial basis functions, $\triangle_f(x)$ is the bounded approximation error, $||\Delta_f(x)|| < \varepsilon_f$, where ε_f is a positive constant.

For any non-zero approximation error $\triangle_f(x)$, there exists optimal weight matrix ${w_f}^*$ such that

$$f(x(k)) = \hat{f}(x, w_f^*) - \triangle_f(x) \tag{17}$$

where w_f^* is the optimal weight of identifier, and $\hat{f}(x, w_f^*) = w_f^*(k)^{\mathrm{T}} h[x(k)]$. The output weights are updated and the hidden weights remain unchanged when training, so the neural network model identification error is

$$\begin{aligned} \tilde{f}(x(k)) &= f[x(k)] - \hat{f}[x(k)] \\ &= \hat{f}(x, {w_f}^*) - \triangle_f[x(k)] - \hat{w_f}(k)^{\mathrm{T}} h[x(k)] \\ &= -\tilde{w_f}(k)^{\mathrm{T}} h[x(k)] - \triangle_f[x(k)] \end{aligned} \tag{18}$$

where$-\tilde{w_f}(k) = {w_f}^*(k) - \hat{w_f}(k)$.

The weights are adjusted to minimize the following error

$$E(k+1) = \frac{1}{2}[\tilde{f}(x(k))]^T[\tilde{f}(x(k))] \tag{19}$$

Using gradient descent method, the weights are updated by

$$\begin{aligned} \Delta w_{f_j}(k+1) &= -\eta \frac{\partial E}{\partial w_{f_j}} \\ &= \eta(f(x(k)) - \hat{f}(x(k)))h[x(k)] \\ &= \eta(\tilde{f}(x(k)))h[x(k)] \end{aligned} \tag{20}$$

and

$$w_{f_j}(k) = w_{f_j}(k-1) + \Delta w_{f_j}(k) \tag{21}$$

where $\eta > 0$ is the learning rate of the identifier.

Assumption 2. *The error of the neural network approximation is assumed to have an upper bound, namely*

$$\triangle_f(x)^{\mathrm{T}} \triangle_f(x) \leq \tilde{w_f}(k)^{\mathrm{T}} \tilde{w_f}(k) h[x(k)]^{\mathrm{T}} h[x(k)] \tag{22}$$

3.2. RBF-NN Steady-State Controller Design

We use the RBF-NN to approximate the steady-state control $u_d(k)$ directly, the inverse dynamic NN is established to approximate[48,49].

 doi:10.20944/preprints202403.0016.v1

We design the steady-state control $u_d(k)$ through the approximation of the RBF-NN

$$u_d(k) = \hat{w_d}^T(k)h[x_d(k)] \tag{23}$$

where $\hat{w_d}$ is the actual neural network weights; $h[x_d(k)]$ is the output of the hidden layers; $u_d(k)$ is the output of the RBF-NN.

Let the ideal steady-state control $u_d^*(k)$ be

$$u_d^*(k) = w_d^{*T}h[x_d(k)] + \varepsilon_u \tag{24}$$

where w_d^* is the optimal neural network weights and ε_u is the error vector. Assuming that $x_d(k+1)$ is the desired output of the system at the point $k+1$, without considering external disturbances, the control input $u_d^*(k)$ satisfies

$$L[x_d(k), u_d^*(k)] - x_d(k+1) = 0 \tag{25}$$

where $L[x_d(k), u_d^*(k)] = f[x_d(k)] + g[x_d(k)]u_d^*(k)$.

Thus, we can define the error $e_m(k)$ of the approximating state as

$$e_m(k+1) = L[x_d(k), u_d(k)] - x_d(k+1) \tag{26}$$

where $L[x_d(k), u_d(k)] = f[x_d(k)] + g[x_d(k)]u_d(k)$.

(24) subtracted from (23) yields

$$\begin{aligned} u_d(k) - u_d^*(k) &= \hat{w_d}^\top(k)h[x_d(k)] - w_d^{*\top}(k)h[x_d(k)] - \varepsilon_u \\ &= \tilde{w_d}^T(k)h[x_d(k)] - \varepsilon_u \end{aligned} \tag{27}$$

where $\tilde{w_d}(k) = \hat{w_d}(k) - w_d^*(k)$ is weight approximation error.

The weights are updated by the following update law of the weights

$$\hat{w_d}(k+1) = \hat{w_d}(k) - \gamma[h(z)e_m(k+1) + \sigma\hat{w_d}(k)] \tag{28}$$

where $\gamma > 0$ and $\sigma > 0$ are positive constant .

Assumption 3. *Within the set Ω_ε, the ideal neural network weights w^* and the approximation error are bounded*

$$\| w_d^* \| \leqslant w_m, ||\varepsilon_u|| \leqslant \varepsilon_l \tag{29}$$

3.3. Near Optimal Feedback Controller Design

In this subsection, we present an adaptive dynamic programming algorithm (ADP) based on the Bellman optimality. The objective is to find the feedback control policy that minimizes the approximated cost function.

First, the initial cost function $V^0(e(k)) = 0$ which is not necessarily the optimal value function, and then a single control vector $u_e^0(k) = 0$ can be solved by

$$V^0(e(k)) = \arg\min_{u_e(k)}\{e^T(k)Qe(k) + u_e^T(k)Ru_e(k) + V^0(e(k+1))\} \tag{30}$$

After that, we update the control law,

$$\begin{aligned} u_e^1(k) &= \arg\min_{u_e(k)}\{e^T(k)Qe(k) + u_e^T(k)Ru_e(k) + V^0(e(k+1))\} \\ &= e^T(k)Qe(k) + (u_e^0(k))^T Ru_e^0(k) \end{aligned} \tag{31}$$

 doi:10.20944/preprints202403.0016.v1

hence, for $i = 1, 2, \ldots$,the adaptive dynamic programming algorithm can be realized in a continuous iterative process in

$$\begin{aligned} V^i(e(k)) &= \min_{u_e(k)} \Big\{ e^T(k) Q e(k) + u_e^T(k) R u_e(k) + V^i(e(k+1)) \Big\} \\ &= -\frac{1}{2} R^{-1} g_e^T(k) \frac{\partial V^i(e(k+1))}{\partial e(k+1)} \end{aligned} \tag{32}$$

and

$$\begin{aligned} u_e^{i+1}(k) &= \arg\min_{u_e(k)} \{ e^T(k) Q e(k) + u_e^T(k) R u_e(k) + V^i(e(k+1)) \} \\ &= e^T(k) Q e(k) + (u_e^i(k))^T R u_e^i(k) + V^i(e(k+1)) \end{aligned} \tag{33}$$

where index i represents the number of iterations of the control law and the cost function, while index k represents time index of system state trajectory. Moreover, it is worth noting in the iterative process of adaptive dynamic programming that the number of iterations of the cost function and the control law increases from zero to infinity.

To begin the development of the feedback control policy, we use neural networks to construct the critic network and the actor network.

The critic network is used for approximating the cost function $V_i(e(k))$. The output of the critic network is denoted as

$$\widehat{V}^i(e(k)) = w_{ci}^T z(v_{ci}^T e(k)) + \varepsilon_c(k) \tag{34}$$

where $z(v_{ci}^T e(k))$ is the hidden layer function, w_{ci} is the hidden layer weight of the critic network, v_{ci} is the input layer weight of the critic network, $\varepsilon_c(k)$ is the approximation error.

So we define the prediction error of the critic network as

$$e_{ci}(k) = \widehat{V}^i(e(k)) - V^i(e(k)) \tag{35}$$

The objective function to be minimized for the critic network is

$$E_{ci}(k) = \frac{1}{2} e_{ci}^T(k) e_{ci}(k). \tag{36}$$

The weights of critic network are updated using the gradient descent method through

$$w_{ci}(k+1) = w_{ci}(k) - \alpha_c [\frac{\partial E_{ci}(k)}{\partial w_{ci}(k)}] \tag{37}$$

where $\alpha_c > 0$ is the learning rate of the critic network, and i is the update count of internal neuron to update the weight parameters.

The inputs of the actor network is the system error $e(k)$, and the outputs of the actor network is the optimal feedback control $u_e(k)$. The output can be formulated as

$$\hat{u}_e^i(k) = w_{ai}^T z(v_{ai}^T e(k)) + \varepsilon_a(k), \tag{38}$$

where $z(v_{ai}^T e(k))$ is the hidden layer function, w_{ai} is the hidden layer weight of the actor network, v_{ai} is the input layer weight of the actor network, $\varepsilon_a(k)$ is the approximation error.

Therefore, we define the prediction error of the action network as

$$e_{ai}(k) = \hat{u}_e^i(k) - u_e^i(k) \tag{39}$$

where $\hat{u}_e^i(k)$ is approximating optimal feedback control ,$u_e^i(k)$ is the optimal feedback control at the iterative number i.

 doi:10.20944/preprints202403.0016.v1

The objective function to be minimized for the action network is

$$E_{ai}(k) = \frac{1}{2} e_{ai}^T(k) e_{ai}(k) \tag{40}$$

The weights of the actor network are also updated in the same way as the critic network , we use the gradient descent method

$$w_{ai}(k+1) = w_{ai}(k) - \beta_a [\frac{\partial E_{ai}(k)}{\partial w_{ai}(k)}], \tag{41}$$

where $\beta_a > 0$ is the learning rate of the actor network and i is the update count of internal neuron to update the weight parameters.

3.4. Stability Analysis

In this subsection, the stability proof of the system is obtained by Lyapunov stability theory.

Assumption 4. *Radial basis function* $h(t) = \exp\left(-\frac{\|x(t)-c(t)\|^2}{2b^2}\right)$ *of the maximum value is* $h_{max} = 1$, *where* $c(t)$ *is the center point and* b *is the width of Radial basis function. Assuming the numbers of neurons is* $l \in [l_f, l_d]$ *for any radial basis function* $h \in [h[x(k)], h[x_d(k)]]$, *then*

$$\begin{aligned} &|h_i| \leqslant 1, \|h\| \leqslant \sqrt{l} \leqslant l, \\ &h^T h = \|h\|^2 \leqslant l \end{aligned} \tag{42}$$

we can know the maximum value $\|h\|^2$ *of the hidden layer with* l *neurons is* $l \in [l_f, l_d]$, *then we assume the maximum value* $\|h[x(k)]\|^2$ *of the hidden layer for the identifier* $\hat{f}(x(k))$ *is* l_f, *and the maximum value* $\|h_d[x(k)]\|^2$ *of the hidden layer for the steady-state controller* $u_d(k)$ *is* l_d.

Lemma 2. *The relationship between (25) and weight approximation error (27) satisfies the following equation.*

$$\tilde{w_d}^T(k) h[x_d(k)] = \frac{e_m(k+1)}{L_u} + \varepsilon_u \tag{43}$$

where $e_m(k)$ *is the error of the approximating state* $x_d(k)$, $L_u = \left.\frac{\partial L}{\partial u}\right|_{u=\xi}$, $\xi \in [u_d^*(k), u_d(k)]$, $g_1 \geqslant \left|\frac{\partial L}{\partial u}\right| > \epsilon > 0$, g_1 *and* ϵ *are positive constants.*

Proof. Subtracting w_d^* from both sides of (28) , we get

$$\tilde{w_d}(k+1) = \tilde{w_d}(k) - \gamma [h[x_d(k)] e_m(k+1) + \sigma \hat{w_d}(k)] \tag{44}$$

Combining(25) and (27) with the mean value theorem, we can obtain

$$\begin{aligned} L[x_d(k), u_d(k)] &= L[x_d(k)], u_d^*(k) + \tilde{w_d}^T(k) h[x_d(k)] - \varepsilon_u \Big] \\ &= L[x_d(k), u_d^*(k)] + \Big[\tilde{w_d}^T(k) h[x_d(k)] - \varepsilon_u\Big] L_u \\ &= x_d(k+1) + \Big[\tilde{w_d}^T(k) h[x_d(k)] - \varepsilon_u\Big] L_u \end{aligned} \tag{45}$$

 doi:10.20944/preprints202403.0016.v1

Further combining (45) with (26), we can obtain

$$\begin{aligned} e_m(k+1) &= L[x_d(k), u_d(k)] - x_d(k+1) \\ &= [\tilde{w_d}^T(k)h[x_d(k)] - \varepsilon_u]L_u \end{aligned} \tag{46}$$

After sorting out, we can obtain

$$\tilde{w_d}^T(k)h[x_d(k)] = \frac{e_m(k+1)}{L_u} + \varepsilon_u \tag{47}$$

the proof is completed. □

Lemma 3. *For analysis simplicity, ε_u and $e_m(k+1)$ have an inequality relation though using young's inequality.*

$$-2\varepsilon_u e_m(k+1) \leqslant k_0\varepsilon_l^2 + \frac{1}{k_0}e_m^2(k+1) \tag{48}$$

where k_0 is a positive constant.

From Figure 1, it can be seen that with $e(k)$, $x_d(k)$ and $u_e^i(k)$, the estimated error $e(k+1)$ can be obtained with the aid of the RBF-NN identifier and the steady-state controller. Using the steady-state controller, we can obtain the reference trajectory $x_d(k)$ corresponding to the steady-state controller $u_d(k)$. Using the ADP algorithm, we can obtain optimal feedback controller $u_e^i(k)$. Then, the actual controller $u(k) = u_e^i(k) + u_d(k)$ and system dynamic $x(k+1)$ can be obtained. Furthermore, with $x_d(k)$ and $x(k)$ we can get the estimated tracking error $e(k)$,further obtained $e(k+1)$. Finally, we can reconstruct the system dynamic to track the reference trajectory.

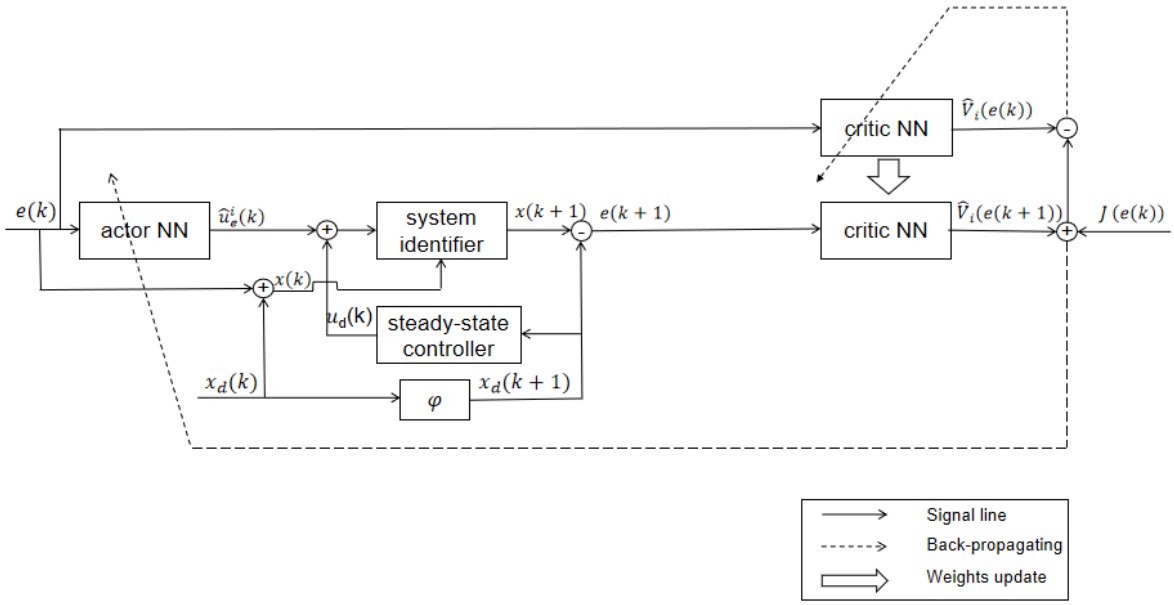

Figure 1. The structure schematic of the proposed technique.

Theorem 1. *For the optimal tracking problem (1)-(3), the RBF-NN identifier (16) is used to approximate $f(x(k))$, the steady-state controller $u_d(k)$ is approximated by the RBF-NN (23), and the feedforward networks*

 doi:10.20944/preprints202403.0016.v1

(34),(38) *is used to approximate the cost function* $J(e(k),u(k))$ *and the feedback controller* $u_e(k)$*, respectively. Assume that the parameters satisfy the following inequality,*

$$\begin{aligned}
&(a)0 < \eta \leqslant \frac{1}{l_f} \\
&(b)0 < g_1 \leqslant k_0 \\
&(c)0 < (1+\sigma) l_d \gamma \leqslant \frac{1}{g_1} - \frac{1}{k_0} \\
&(d)0 < (l_d + \sigma)\gamma \leqslant 1 \\
&(e)a_c \leq 2/\|z(v_{ci}^T e(k))\|^2 \\
&(f)\beta_a \leq 2/\|z(\nu_{ai}^T e(k))\|^2
\end{aligned} \tag{49}$$

where η *is the learning rate of the RBF-NN identifier,* σ *and* γ *are the update parameters of the steady-state controller approximating network weights,* a_c *is the learning rate of the actor network,* β_a *is the learning rate of the critic network,* $z(v_{ci}^T e(k))$ *and* $z(\nu_{ai}^T e(k))$ *are hidden layer function of the actor network and the critic network. Then, the closed loop system* (6) *of approximating error is asymptotically stable when the parameter estimation errors are bounded.*

Proof. Considering the following positive definite Lyapunov function candidate

$$\begin{aligned}
J(k) &= J_1(k) + J_2(k) + J_3(k) + J_4(k) \\
&= \frac{1}{\eta}\tilde{w_f}(k)^T\tilde{w_f}(k) + \frac{1}{g_1}e_m^2(k) + \frac{1}{\gamma}\tilde{w_d}(k)^T\tilde{w_d}(k) + w_{ci}(k)^T w_{ci}(k) + w_{ai}(k)^T w_{ai}(k)
\end{aligned} \tag{50}$$

where $J_1(k) = \frac{1}{\eta}\tilde{w_f}(k)^T\tilde{w_f}(k)$, $J_2(k) = \frac{1}{g_1}e_m^2(k) + \frac{1}{\gamma}\tilde{w_d}(k)^T\tilde{w_d}(k)$, $J_3(k) = w_{ci}(k)^T w_{ci}(k)$, $J_4(k) = w_{ai}(k)^T w_{ai}(k)$.

Firstly, differencing it according to the Lyapunov function of $J_1(k) = \frac{1}{\eta}\tilde{w_f}(k)^T\tilde{w_f}(k)$ yields

$$\begin{aligned}
\Delta J_1(k) &= J_1(k+1) - J_1(k) \\
&= \frac{1}{\eta}\tilde{w_f}(k+1)^T\tilde{w_f}(k+1) - \frac{1}{\eta}\tilde{w_f}(k)^T\tilde{w_f}(k) \\
&= \frac{1}{\eta}[\tilde{w_f}(k) + \eta\tilde{f}(x(k))h[x(k)]]^T[\tilde{w_f}(k) + \eta\tilde{f}(x(k))h[x(k)]] - \frac{1}{\eta}\tilde{w_f}(k)^T\tilde{w_f}(k) \\
&= \frac{1}{\eta}[\tilde{w_f}(k)^T\tilde{w_f}(k) - \frac{1}{\eta}\tilde{w_f}(k)^T\tilde{w_f}(k) + \eta^2[\tilde{f}(x(k))^T\tilde{f}(x(k))h[x(k)]^T h[x(k)] \\
&\quad + 2\eta\tilde{f}(x(k))\tilde{w_f}(k)^T h[x(k)] \\
&= \eta[[\tilde{w_f}(k)^T h[x(k)] + \triangle_f[x]]^T[\tilde{w_f}(k)^T h[x(k)] + \triangle_f[x]]h[x(k)]^T h[x(k)]] \\
&\quad + 2[\tilde{w_f}(k)^T h[x(k)] + \triangle_f[x]]\tilde{w_f}(k)^T h[x(k)] \\
&= \eta[\tilde{w_f}(k)^T\tilde{w_f}(k)h[x(k)]^T h[x(k)] + 2\tilde{w_f}(k)^T h[x(k)]\triangle_f[x] - 2\tilde{w_f}(k)^T h[x(k)] \\
&\quad - 2\tilde{w_f}(k)^T\tilde{w_f}(k)h[x(k)]^T h[x(k)] + \triangle_f[x]^T\triangle_f[x]h[x(k)]^T h[x(k)]
\end{aligned} \tag{51}$$

According to the Assumption 2, Assumption 4 and (42), (51) can be done to get

$$\begin{aligned}
\Delta J_1(k) &\leqslant \eta l_f^2 \parallel \tilde{w_f}(k) \parallel^2 - 2l_f \parallel \tilde{w_f}(k) \parallel^2 + \eta l_f^2 \parallel \tilde{w_f}(k) \parallel^2 \\
&\quad + 2\eta l_f \tilde{w_f}(k)^T h[x(k)]\triangle_f[x] - 2\tilde{w_f}(k)^T h[x(k)]\triangle_f[x] \\
&\leqslant \parallel \tilde{w_f}(k) \parallel^2 (2\eta l_f^2 - 2l_f) + (2l_f\eta - 2)\tilde{w_f}(k)^T h[x(k)]\triangle_f[x] \\
&\leqslant \parallel \tilde{w_f}(k) \parallel^2 (4l_f^2\eta - 4l_f)
\end{aligned} \tag{52}$$

 doi:10.20944/preprints202403.0016.v1

After that, differencing it according to the Lyapunov function of $J_2(k) = \frac{1}{g_1}e_m^2(k) + \frac{1}{\gamma}\tilde{w}_d(k)^T\tilde{w}_d(k)$ yields

$$\begin{aligned}
\Delta J_2(k) &= J_2(k+1) - J_2(k) \\
&= \frac{1}{g_1}\left[e_m^2(k+1) - e_m^2(k)\right] - \frac{1}{\gamma}\tilde{w}_d(k)^T\tilde{w}_d(k) + \frac{1}{\gamma}\tilde{w}_d(k+1)^T\tilde{w}_d(k+1) \\
&= \frac{1}{\gamma}\langle\tilde{w}_d(k) - \gamma[h[x_d(k)]e_m(k+1) + \sigma\hat{w}_d(k)]\rangle^T\langle\tilde{w}_d(k) - \gamma[h[x_d(k)]e_m(k+1) + \sigma\hat{w}_d(k)]\rangle \\
&\quad - \frac{1}{\gamma}\tilde{w}_d(k)^T\tilde{w}_d(k) + \frac{1}{g_1}[e_m^2(k+1) - e_m^2(k)] \\
&= \frac{1}{g_1}\left[e_m^2(k+1) - e_m^2(k)\right] - 2\tilde{w}_d(k)^T h[x_d(k)]e_m(k+1) + \gamma\sigma^2\hat{w}_d(k)^T\hat{w}_d(k) \\
&\quad - 2\sigma\tilde{w}_d(k)^T\hat{w}_d(k) + \gamma h^T[x_d(k)]h[x_d(k)]e_m^2(k+1) + 2\gamma\sigma\hat{w}_d{}^T(k)h[x_d(k)]e_m(k+1)
\end{aligned} \tag{53}$$

where

$$\begin{aligned}
2\sigma\tilde{w}_d(k)^T\hat{w}_d(k) &= \sigma\tilde{w}_d(k)^\top[\bar{w}_d(k) + w_d^*] + \sigma[\hat{w}_d(k) - \omega_d^*]^T\hat{w}_d(k) \\
&= \sigma \parallel \tilde{w}_d(k) \parallel^2 + \parallel \hat{w}_d(k) \parallel^2 + \tilde{w}_d(k)^T w_d^* - w_d^*\hat{w}_d(k)^T \\
&= \sigma[\parallel \tilde{w}_d(k) \parallel^2 + \parallel \hat{w}_d(k) \parallel^2 - \parallel w_d^* \parallel^2],
\end{aligned}$$

$$\gamma h^T[x_d(k)]h[x_d(k)]e_m^2(k+1) \leqslant \gamma l_d e_m^2(k+1), \tag{54}$$

$$2\gamma\sigma\hat{w}_d{}^T(k)h[x_d(k)]e_m(k+1) \leqslant \gamma\sigma l_d[\parallel \hat{w}_d(k) \parallel^2 + e_m^2(k+1)],$$

$$\gamma\sigma^2\hat{w}_d{}^T(k)\hat{w}_d(k) = \gamma\sigma^2\|\hat{w}_d(k)\|^2$$

Substituting (54) into (53) yields

$$\begin{aligned}
\Delta J_2(k) &\leqslant \frac{1}{g_1}\left[e_m^2(k+1) - e_m^2(k)\right] - 2\left[\frac{e_m(k+1)}{L_u} + \varepsilon_u\right]e_m(k+1) \\
&\quad - \sigma\left[\parallel \tilde{w}_d(k) \parallel^2 + \gamma\sigma^2 \parallel \hat{w}_d(k) \parallel^2 + \parallel \hat{w}_d(k) \parallel^2 - \parallel w_d^* \parallel^2\right] + \gamma l_d e_m^2(k+1) \\
&\quad + \gamma\sigma l_d[\parallel \hat{w}_d(k) \parallel^2 + e_m^2(k+1)] \\
&= \left[\frac{1}{g_1} - \frac{2}{L_u} + \gamma(1+\sigma)l_d\right]e_m^2(k+1) - \frac{1}{g_1}e_m^2(k) - 2\varepsilon_u e_m(k+1) - \sigma \parallel \tilde{w}_d(k) \parallel^2 \\
&\quad + \sigma \parallel w_d^* \parallel^2 + \sigma(-1 + \gamma l_d + \gamma\sigma) \parallel \hat{w}_d(k) \parallel^2
\end{aligned} \tag{55}$$

Considering (26) and $g_1 \geqslant \left|\frac{\partial L}{\partial u}\right| > \epsilon > 0$, we can deduce

$$\frac{1}{g_1} - \frac{2}{L_u} \leqslant \frac{1}{g_1} - \frac{2}{g_1} = -\frac{1}{g_1} < 0 \tag{56}$$

 doi:10.20944/preprints202403.0016.v1

With Lemma 2 and Lemma 3, we can further deduce

$$\begin{aligned}\Delta J_2(k) \leqslant & \left[-\frac{1}{g_1}+\gamma(1+\sigma)l_d+\frac{1}{k_0}\right]e_m^2(k+1)+\sigma(\gamma l_d+\gamma\sigma-1)\parallel \hat{w_d}(k)\parallel^2 \\ & -\frac{1}{g_1}e_m^2(k)-\sigma\parallel \bar{w_d}(k)\parallel^2+\sigma\omega_m^2+k_0\varepsilon_l^2 \\ = & -\left[\frac{1}{g_1}-(1+\sigma)l_d\gamma-\frac{1}{k_0}\right]e_m^2(k+1)+\sigma[(l_d+\sigma)\gamma-1]\parallel \hat{w_d}(k)\parallel^2 \\ & -\frac{1}{g_1}\left[e_m^2(k)-\beta\right]-\sigma\parallel \tilde{w_d}(k)\parallel^2\end{aligned} \tag{57}$$

where $\beta = g_1(\sigma w_m^2 + k_0\varepsilon_l^2)$ is a positive constant.

Next, we consider the following Lyapunov function

$$J_3(k)+J_4(k) = w_{ci}(k)^T w_{ci}(k)+w_{ai}(k)^T w_{ai}(k). \tag{58}$$

Then, differencing it according to the Lyapunov function of (58) yields

$$\begin{aligned}\Delta J_3(k)+\Delta J_4(k) = & \{w_{ci}(k+1)^T w_{ci}(k+1)+w_{ai}(k+1)^T w_{ai}(k+1)\} \\ & -\{w_{ci}(k)^T w_{ci}(k)+w_{ai}(k)^T w_{ai}(k)\} \\ = & a_c\|e_{ci}(k)\|^2(-2+a_c\|z(v_{ci}^T e(k))\|^2) \\ & +\beta_a\parallel e_{ai}(k)\parallel^2(-2+\beta_a\parallel z(v_{ai}^T e(k))\parallel^2).\end{aligned} \tag{59}$$

Finally, $\Delta J(k)$ is derived from (52), (57) and (59)

$$\begin{aligned}\Delta J(k) = & \Delta J_1(k)+\Delta J_2(k)+\Delta J_3(k)+\Delta J_4(k) \\ \leqslant & 4\parallel \tilde{w_f}(k)\parallel^2(l_f^2\eta-l_f)-\sigma\parallel \tilde{w_d}(k)\parallel^2-\left[\frac{1}{g_1}-(1+\sigma)l_d\gamma-\frac{1}{k_0}\right]e_m^2(k+1) \\ & +\sigma[(l_d+\sigma)\gamma-1]\parallel \hat{w_d}(k)\parallel^2-\frac{1}{g_1}\left[e_m^2(k)-\beta\right]+a_c\|e_{ci}(k)\|^2(-2+a_c\|z(v_{ci}^T e(k))\|^2) \\ & +\beta_a\parallel e_{ai}(k)\parallel^2(-2+\beta_a\parallel z(v_{ai}^T e(k))\parallel^2).\end{aligned} \tag{60}$$

Based on the above analysis, when the parameters are selected to fulfill the following condition with $e_m^2(k) \geqslant \beta$,

$$\begin{aligned}& 0<\eta\leqslant\frac{1}{l_f} \\ & 0<g_1\leqslant k_0 \\ & 0<(1+\sigma)l_d\gamma\leqslant\frac{1}{g_1}-\frac{1}{k_0} \\ & 0<(l_d+\sigma)\gamma\leqslant 1 \\ & a_c\leq 2/\|z(v_{ci}^T e(k))\|^2 \\ & \beta_a\leq 2/\|z(v_{ai}^T e(k))\|^2\end{aligned} \tag{61}$$

we can obtain $\Delta J(k) \leqslant 0$. This proof is completed. □

4. Simulation

In this section, in order to demonstrate the effectiveness of the proposed tracking control method, a discrete-time nonlinear system is introduced.The case is derived from [47]. We assume that the

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 March 2024 doi:10.20944/preprints202403.0016.v1

nonlinear smooth function $f \in \mathbb{R}^n$ is an unknown nonlinear drift function and $g \in \mathbb{R}^{n\times m}$ is a known function. The corresponding $f[x(k)]$ and $g(k)$ are given as

$$f[x(k)] = \begin{bmatrix} -\sin(0.5x_2(k))x_1^2(k) \\ -\cos(1.4x_2(k))\sin(0.9x_1(k)) \end{bmatrix}$$
$$g(k) = \begin{bmatrix} (x_1(k))^2 + 1.5 & 0.1 \\ 0 & 0.2((x_1(k) + x_2(k))^2 + 1) \end{bmatrix} \tag{62}$$

The reference trajectory $x_d(k)$ for the above system is defined as

$$x_d(k) = \begin{bmatrix} 0.25sin(10^{-3}k) \\ 0.25\cos(10^{-3}k) \end{bmatrix} \tag{63}$$

where $time(s)$ of y-axis is chosen as k(1,...,10000) multiplied by $ts = 0.001$ in the simulation.

The RBF networks have a three-layer structure with 2 input neurons, hidden layers have 9 neurons, and output layer have 2 neurons, the parameters c_i and b_i of the radial basis functions are chosen as c_i(i=1,2,...,9) = $\begin{bmatrix} -2 & -1.5 & -1.0 & -0.5 & 0 & 0.5 & 1.0 & 1.5 & 2 \\ -2 & -1.5 & -1.0 & -0.5 & 0 & 0.5 & 1.0 & 1.5 & 2 \end{bmatrix}$ and $b_j = [2, 2]$, the initial weights w_0 were chosen to be random numbers between (0,1), where the inputs to the RBF-NN identifier are chosen to be $x(k)$ and the inputs to the RBF-NN steady-state control u_d are chosen to be $x_d(k)$. Update of weights $\hat{w_d}$, $\hat{w_f}$ is used in (21) and (28). Because $g_1 \geqslant \frac{\partial L}{\partial u} = 1$, we can select $g_1 = 5$. According to $0 < g_1 \leqslant k_0$ of Theorem 1, we can select $k_0 = 10$. For the control parameters η, because hidden layers have 9 neurons, $l = 9$, $0 < \eta \leqslant \frac{1}{l} \leqslant \frac{1}{9}$, we select $\eta = 0.1$. While control parameters γ, σ, we can know $0 < (1+\sigma)9\gamma \leqslant \frac{1}{5} - \frac{1}{10} = \frac{1}{10} = 0.10$ and $0 < (9+\sigma)\gamma \leqslant 1$ from Theorem 1, so selecting $\gamma = 0.01, \sigma = 0.001$. The initial state is set as $x(0) = 0$. We trained the RBF networks with 10,000 steps of acquired data , and Figures 2 and 3 shows the RBF-NN identifier to approximate the tracking curves of the unknown dynamics $\tilde{f}$.

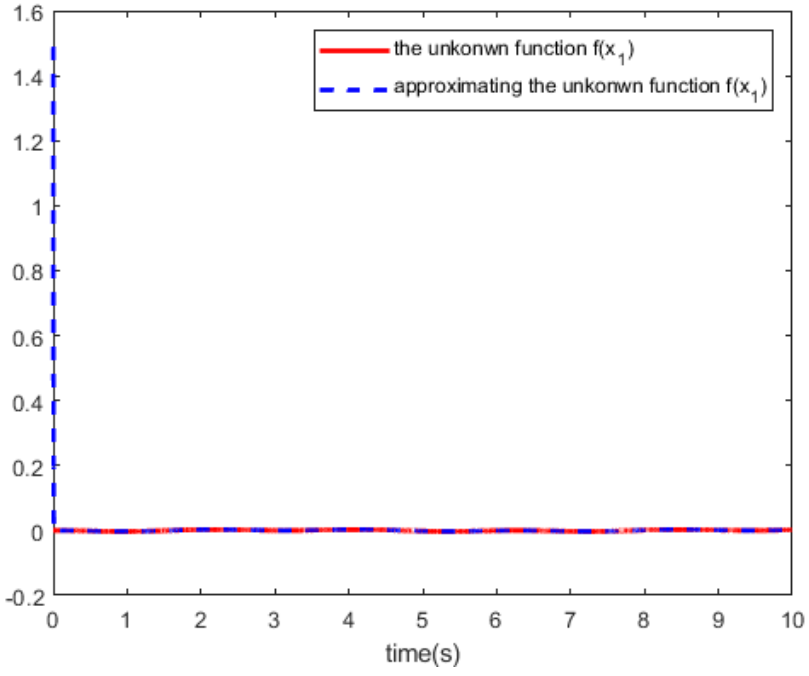

Figure 2. The unkonwn function $f(x_1)$ and approximating the unkonwn function $\tilde{f}(x_1)$.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 March 2024 doi:10.20944/preprints202403.0016.v1

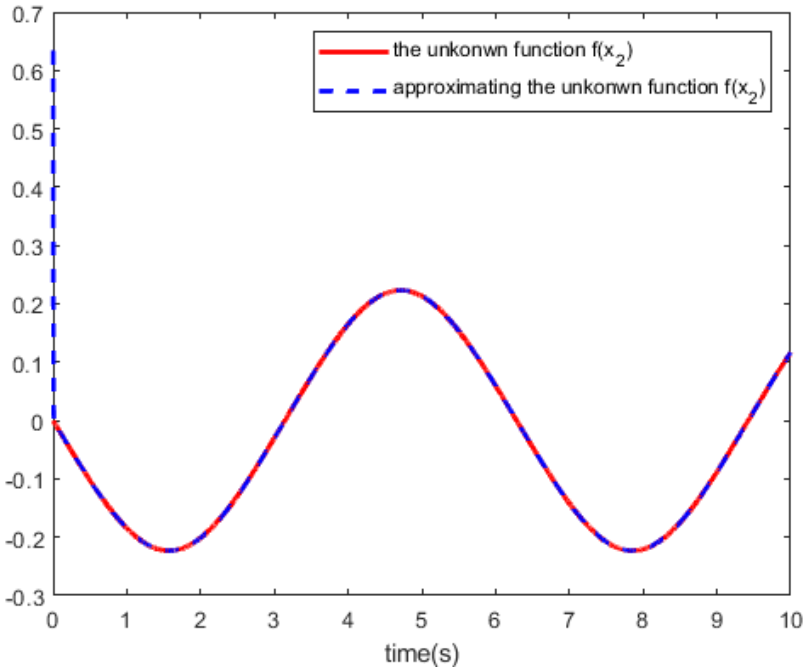

Figure 3. The unkonwn function $f(x_2)$ and approximating the unkonwn function $\tilde{f}(x_2)$.

The performance index is select as $Q = I$ and $R = I$ that I is the identity matrix with appropriate dimension. For the actor network and critic network, we used the same parameter settings. The initial weights of the critic network and the actor network are chosen as random numbers between $(-10, 10)$. The input layer have 2 neurons, the hidden layer have 15 neurons, the output layer have 2 neurons, the learning rate is 0.1. The hidden layer uses the function *tansig* and the function *purelin*, the output layer uses the function *trainlm* .Though parameter settings, we train the actor network and the critic network with 5000 training steps to reach the given accuracy 1e-9. Figure 4 shows the curves of the system control u. In Figures 5 and 6, we can see the curves of the state trajector x and the reference trajector x_d.

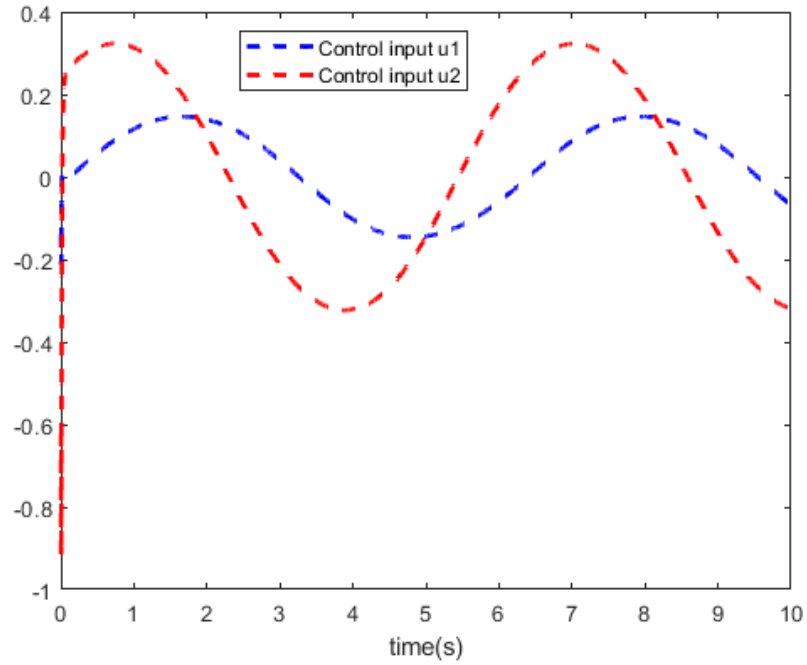

Figure 4. Control input u.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 March 2024 doi:10.20944/preprints202403.0016.v1

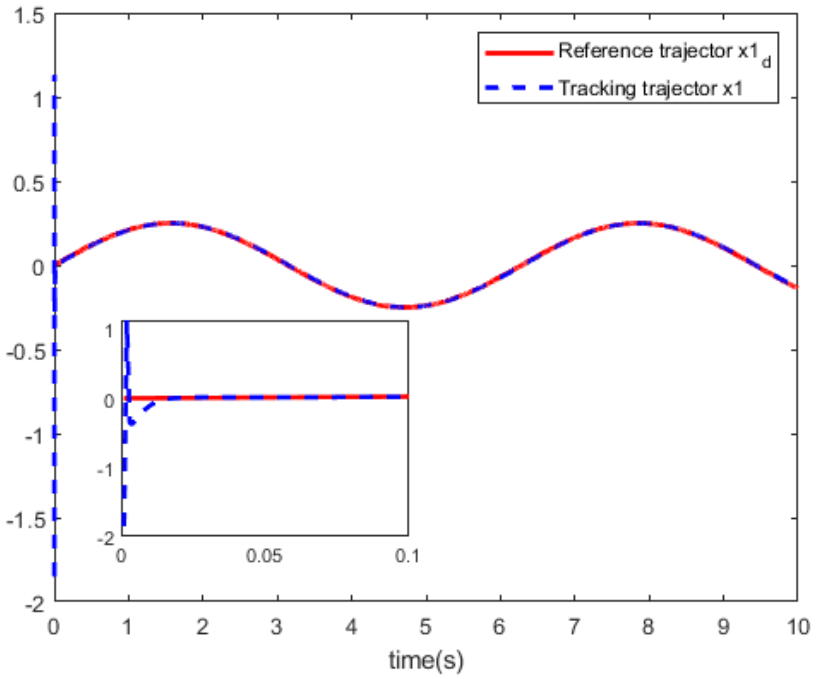

Figure 5. The state trajector $x1$ and the reference trajector $x1_d$.

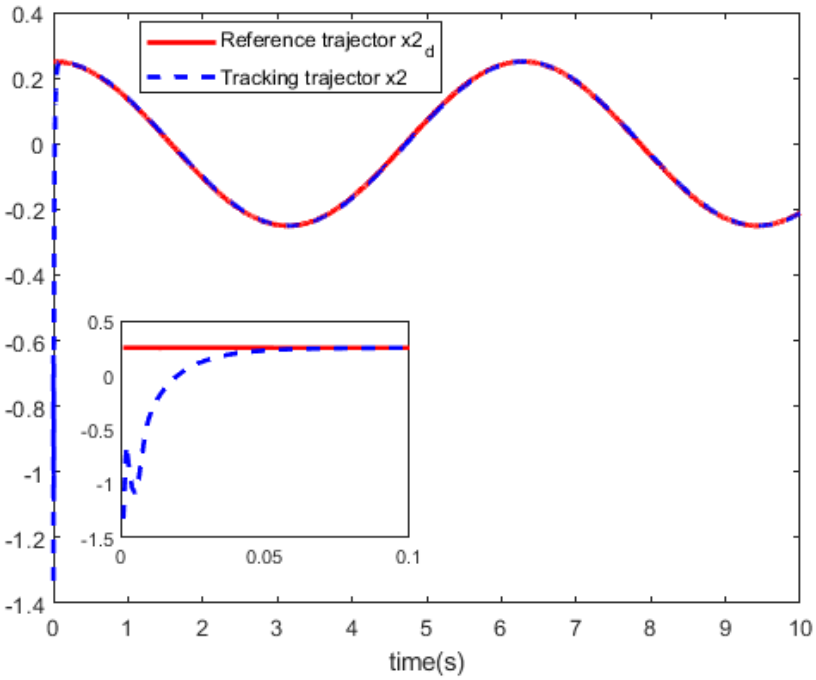

Figure 6. The state trajector $x2$ and the reference trajector $x2_d$.

Based on above the results, the simulation results show that this tracking technique obtains a relatively satisfactory tracking performance for partially unknown discrete-time nonlinear systems.

5. Conclusion

This paper proposed an optimal tracking control scheme through approximate dynamic programming for a class of partially unknown discrete-time nonlinear systems based on RBF-NNs . In dealing with unknown variables, two RBF-NNs are used to approximate the unknown function and the steady-state controller, respectively. Moreover, ADP algorithm are introduced to get the optimal feedback

 doi:10.20944/preprints202403.0016.v1

control for tracking the error dynamics, two feedforward neural networks are utilized as structures to approximate the cost function and feedback control inputs severally. Finally, simulation results show a relatively satisfactory tracking performance, which verify the effectiveness of the optimal tracking control technique. In future works, we will consider event-triggered control as well as completely unknown dynamics.

Author Contributions: All the authors contributed equally to the development of the research. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grant No. 61463002, the Guizhou Province Natural Science Foundation of China under Grant No. Qiankehe Fundamentals-ZK[2021] General 322 and the Doctoral Foundation of Guangxi University of Science and Technology Grant No. Xiaokebo 22z04 .

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is contained within the article.

Acknowledgments: The authors thank to the Journal editors and the reviewers for their helpful suggestions and comments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Broomhead, D S , and D. Lowe . Radial basis functions, multi-variable functional interpolation and adaptive networks. **1988**.
2. Narendra, K. S. , and K. Parthasarathy . Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks* **1990**,*1*:.
3. Narendra, Kumpati S. , and S. Mukhopadhyay . Adaptive control of nonlinear multivariable systems using neural networks. **1994**,*737*-752.
4. Hartman, Eric J. , J. D. Keeler , and J. M. Kowalski . Layered Neural Networks with Gaussian Hidden Units as Universal Approximations.*Neural Computation*. **1990**, 210-215.
5. Park, J. Universal approximation using radial basis function networks. *Neural Comput.* **1993**.
6. Lewis, F. L. , and A. Yegildirek . Multilayer neural-net robot controller with guaranteed tracking performance. *IEEE Transactions on Neural Networks*, **1996**, 7.
7. Kobayashi, Hiroaki , and R. Ozawa . Adaptive neural network control of tendon-driven mechanisms with elastic tendons. *Automatica* **2003**,1509-1519.
8. Lewis, F.L., et al.*Optimal Control*, 3rd ed. John Wiley & Sons, Inc, New Jersey, 2012
9. Mannava, A., et al. Optimal tracking control of motion systems. *IEEE Trans. Control Syst. Technol.* **2012**, 1548–1558
10. Sharma, R., Tewari, A. Optimal nonlinear tracking of spacecraft attitude maneuvers. *IEEE Trans. Control Syst. Technol.* **2013**, 12(5), 677–682
11. R.E. Bellman, *Dynamic Programming*, Princeton University Press, Princeton, NJ, 1957.
12. F.L. Lewis, V.L. Syrmos, *Optimal Control*, Wiley, New York, 1995
13. W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. New York, NY, USA: Wiley, 2009.
14. D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*. Belmont, MA, USA: Athena Scientifific, 1996.
15. J. Si, A. G. Barto, W. B. Powell, and D. Wunch, *Handbook of Learning and Approximate Dynamic Programming*. New York, NY, USA: Wiley, 2004.
16. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction.* Cambridge, MA, USA: MIT Press, 1998.
17. F. L. Lewis and D. Vrabie, Reinforcement learning and adaptive dynamic programming for feedback control, *IEEE Circuits Syst*. **2009**,*8*,32–50
18. F. L. Lewis, D. Vrabie, and K. G. Vamvoudakis, Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers, *IEEE Control Syst.*, **2012**,*11*,76–105
19. F. L. Lewis and D. Liu, *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*. Hoboken, NJ, USA: Wiley, 2013.

 doi:10.20944/preprints202403.0016.v1

20. M. Fairbank, S. Li, X. Fu, E. Alonso, and D. Wunsch, An adaptive recurrent neural-network controller using a stabilization matrix and predictive inputs to solve a tracking problem under disturbances, *Neural Netw.*, **2014**,*1*,74–86.
21. D. Vrabie and F. Lewis, Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems, *Neural Netw.*, **2009**,*4*,237–246.
22. D. Liu, X. Yang, and H. Li, Adaptive optimal control for a class of continuous-time affifine nonlinear systems with unknown internal dynamics, *Neural Comput.Appl.* , **2013**, *11*, 1843–1850
23. S. Bhasin, R. Kamalapurkar, M. Johnson, K. Vamvoudakis, F. L. Lewis, and W. Dixon, A novel actor–critic–identififier architecture for approximate optimal control of uncertain nonlinear systems, *Automatica*, **2013**, *1*, 82–92
24. A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof, *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, **2008**,*8*, 943–949
25. D. V. Prokhorov and D. C. Wunsch, Adaptive critic designs, *IEEE Trans. Neural Netw.*, **1997**, *9*,997–1007
26. Y. Luo and H. Zhang, Approximate optimal control for a class of nonlinear discrete-time systems with saturating actuators, *Prog. Natural Sci.*, **2008**, 1023–1029
27. T. Dierks and S. Jagannathan, Online optimal control of nonlinear discrete-time systems using approximate dynamic programming, *Control Theory Appl.*, **2011**, 361–369
28. J. Si and Y.-T. Wang, Online learning control by association and reinforcement, *IEEE Trans. Neural Netw.*, **2001**, *5*, 264–276
29. L. Ren, G. Zhang, and C. Mu, Data-based H_∞ control for the constrained-input nonlinear systems and its applications in chaotic circuit systems, *IEEE Trans. Circuits Syst.*, **2020**, *8*, 2791–2802
30. F. Zhao, W. Gao, T. Liu, and Z.-P. Jiang, Event-triggered robust adaptive dynamic programming with output-feedback for large-scale systems, *IEEE Trans. Control Netw. Syst.*, **2023**,*8*,63–74
31. Q. Wei, H. Li, X. Yang, and H. He, Continuous-time distributed policy iteration for multicontroller nonlinear systems, *IEEE Trans. Cybern.*, **2021**,*5*,2372–2383
32. Y. Zhang, B. Zhao, D. Liu, and S. Zhang, Event-triggered control of discrete-time zero-sum games via deterministic policy gradient adaptive dynamic programming, *IEEE Trans. Syst., Man, Cybern., Syst.*, **2022**, *8*,4823–4835
33. Y. Zhu, D. Zhao, and H. He, Invariant adaptive dynamic programming for discrete-time optimal control, *IEEE Trans. Syst., Man, Cybern., Syst.*, **2020**,*11*,3959–3971
34. Q. Wei, L. Han, and T. Zhang, Spiking adaptive dynamic programming based on poisson process for discrete-time nonlinear systems, *IEEE Trans. Neural Netw. Learn. Syst.*, **2022**, *5*,1846–1856
35. Y. Yang, D. Wunsch, and Y. Yin, Hamiltonian-driven adaptive dynamic programming for continuous nonlinear dynamical systems, *IEEE Trans. Neural Netw. Learn. Syst.*, **2017**, *8*,1929–1940
36. M. Li, J. Qin, N. M. Freris, and D. W. Ho, Multiplayer Stackelberg– Nash game for nonlinear system via value iteration-based integral reinforcement learning, *IEEE Trans. Neural Netw. Learn. Syst.*, **2022**, *4*,1429–1440
37. X. Guo, W. Yan, and R. Cui, Integral reinforcement learning-based adaptive NN control for continuous-time nonlinear MIMO systems with unknown control directions, *IEEE Trans. Syst., Man, Cybern., Syst.*, **2020**, *11*, 4068–4077
38. W. Xue, J. Fan, V. G. Lopez, Y. Jiang, T. Chai, and F. L. Lewis, Off-policy reinforcement learning for tracking in continuous-time systems on two time scales, *IEEE Trans. Neural Netw. Learn. Syst.*, **2021**,*10*,4334–4346
39. C. Sun, X. Li, and Y. Sun, A parallel framework of adaptive dynamic programming algorithm with off-policy learning, *IEEE Trans. Neural Netw. Learn. Syst.*, **2021**, *8*,3578–3587
40. J. Duan, Y. Guan, S. E. Li, Y. Ren, Q. Sun, and B. Cheng, Distributional soft actor-critic: Off-policy reinforcement learning for addressing value estimation errors, *IEEE Trans. Neural Netw. Learn. Syst.*, **2022**, *11*,6584–6598
41. Qiao, L., et al. A novel optimal tracking control scheme for a class of discrete-time nonlinear systems using generalised policy iteration adaptive dynamic programming algorithm. *Syst. Sci.*, **2017**, 525–534
42. Kiumarsi, B., Lewis, F.L. Actor-critic-based optimal tracking for partially unknown nonlinear discrete-time systems.*IEEE Trans. Neural Networks Learn. Syst.*, **2017**, 140–151
43. Zhang, H., et al. Optimal tracking control for a class of nonlinear discrete-time systems with time delays based on heuristic dynamic programming. *IEEE Trans. Neural Networks*, **2011**,1851–1862

 doi:10.20944/preprints202403.0016.v1

44. Zhang, H , Q. Wei , and Y. Luo . A Novel Infinite-Time Optimal Tracking Control Scheme for a Class of Discrete-Time Nonlinear Systems via the Greedy HDP Iteration Algorithm.*IEEE Trans. Syst., Man, Cybern., Syst.*, **2008**,937-942.
45. S. Song, M. Zhu, X. Dai and D. Gong, Model-Free Optimal Tracking Control of Nonlinear Input-Affine Discrete-Time Systems via an Iterative Deterministic Q-Learning Algorithm, *IEEE Trans. Neural Networks Learn. Syst.*, **2024**, *1*,999-1012
46. Huang, Yuzhu , and D. Liu . Neural-network-based optimal tracking control scheme for a class of unknown discrete-time nonlinear systems using iterative ADP algorithm. *Neurocomputing*,**2014**, 46-56.
47. Dierks, Travis , and S. Jagannathan . Optimal tracking control of affine nonlinear discrete-time systems with unknown internal dynamics. IEEE Conference on Decision & Control IEEE, (2010).
48. Zhang, J. , S. S. Ge , and T. H. Lee . Direct RBF neural network control of a class of discrete-time non-affine nonlinear systems. American Control Conference, (2002).
49. Ge, S. S. , J. Zhang , and T. H. Lee . Adaptive MNN control for a class of non-affine NARMAX systems with disturbances. *Systems & Control Letters* **2004**,53,1-12.