International Journal of Control, Automation, and Systems 21(X) (2023) 1-9
http://dx.doi.org/10.1007/s12555-022-0745-9
ISSN:1598-6446 eISSN:2005-4092
http://www.springer.com/12555

Neural Networks Based Optimal Tracking Control of a Delta Robot With Unknown Dynamics

Akram Gholami , Jian-Qiao Sun , and Reza Ehsani*

Abstract: This paper proposes a data-driven optimal tracking control scheme for unknown general nonlinear systems using neural networks. First, a new neural networks structure is established to reconstruct the unknown system dynamics of the form $\dot{x}(t) = f(x(t)) + g(x(t))u(t)$. Two networks in parallel are designed to approximate the functions $f(x)$ and $g(x)$. Then the obtained data-driven models are used to build the optimal tracking control. The developed control consists of two parts, the feed-forward control and the optimal feedback control. The optimal feedback control is developed by approximating the solution of the Hamilton-Jacobi-Bellman equation with neural networks. Unlike other studies, the Hamilton-Jacobi-Bellman solution is found by estimating the value function derivative using neural networks. Finally, the proposed control scheme is tested on a delta robot. Two trajectory tracking examples are provided to verify the effectiveness of the proposed optimal control approach.

Keywords: Data-driven control, dynamics estimation, neural networks, optimal control.

1. INTRODUCTION

Optimal tracking control is a well-known method for designing control systems. The goal of this method is to design a control in such a way that the output optimally tracks a reference trajectory by minimizing a predetermined performance function [1-3]. The optimal tracking control consists of two parts, a feedforward term to ensure the trajectory tracking and a feedback term to stabilize the tracking error dynamics, designed by solving the Hamilton-Jacobi-Bellman (HJB) equation [4-6]. The core challenge in optimal tracking control problems is solving the nonlinear HJB equations, for which a closed-form solution does not exist in case of general nonlinear dynamical systems [7,8]. Therefore, most current approaches focus on estimating the HJB equation solution numerically [9].

The universal approximation theorem states that neural networks with one hidden layer can arbitrarily estimate every continuous function [10]. Thereby, neural networks can learn the HJB equation [9]. In most studies, one network is constructed to estimate the value function, called critic, and is updated to minimize the Bellman error. Another network approximates the control policy, termed as an actor, and is updated to minimize the value function [11,12]. Although, derivative of the value function is used to establish optimal control, all of these methods are estimating the value function. For instance, Vamvoudakis and Lewis [13] proposed an online actor-critic algorithm for learning the optimal control solution of nonlinear systems with known dynamics. In their proposed algorithm, the value function is estimated by neural networks, then by taking the derivative of the neural networks, optimal control is constructed. They proved that the system states and actor-critic neural networks errors are uniformly ultimately bounded in this method.

In recent years, neural networks based optimal control has emerged as a viable technique for solving nonlinear optimal control problems. In most cases, the actor-critic neural network is constructed using polynomial activation functions defined manually [4,14-21]. As an example, Modares and Lewis [4] used neural networks with 45 activation functions containing all powers of the states up to order four. Using polynomial activation function for higher order system leads to huge and complicated neural networks. There exist only a few studies using neural networks with non-polynomial activation functions [22-24]. However, in [23,24] each neuron's activation function is defined manually.

The precise dynamics are generally unknown in many practical situations. One approach is to build a neural networks identifier to model the unknown dynamics. As an example, Bhasin *et al.* [25] developed an actor-critic-identifier to estimate the HJB equation solution using

Manuscript received August 16, 2022; revised January 23, 2023, March 2, 2023, and April 26, 2023; accepted May 8, 2023. Recommended by Associate Editor Niket Kaisare under the direction of Senior Editor PooGyeon Park. This material is based upon work supported by the National Science Foundation under Grant No. 924662.

Akram Gholami, Jian-Qiao Sun, and Reza Ehsani are with the Department of Mechanical Engineering, School of Engineering, University of California Merced, 5200 N. Lake Rd, Merced, CA 95343, USA (e-mails: {agholamipareh, jsun3, rehsani}@ucmerced.edu).
* Corresponding author.

three neural networks. It was demonstrated that the developed control scheme and the identifier guarantee that the states and actor-critic weight estimation error are uniformly ultimately bounded. There are two main methods for the formation of a neural network identifier, single neural network [7,20], and adaptive neural network identifier [26-28]. Single neural network identifier does not represent the dynamics system accurately, since dynamics system include two terms.

Based on the aforementioned discussions, this paper develops a neural network-based optimal tracking control algorithm for a general unknown nonlinear system. A new neural networks structure is proposed to reconstruct the unknown system dynamics. Then the obtained data-driven models are used to build the optimal tracking control. The designed control is made up of two parts, the feed-forward control and the optimal feedback control. The optimal feedback control is created by using neural networks to approximate the solution of the Hamilton-Jacobi-Bellman equation. Unlike the other optimal control approaches, since the gradient of the value function is the one needed for the control design, it is more efficient to construct the critic neural network to estimate the value function derivative directly. Moreover, the Adam optimizer is used to update the neural networks weight matrices.

The optimal tracking of higher-order systems is rarely investigated in the literature because of the difficulties coming from control design and performance analysis. In this work, the developed optimal tracking control scheme is applied on a delta robot, a parallel robot with three degrees of freedom. Our specific contributions include the following:

1) A new neural networks structure is established to reconstruct the unknown system dynamics of the form $\dot{x}(t) = f(x(t)) + g(x(t))u(t)$. Two networks in parallel are designed to approximate the functions $f(x)$ and $g(x)$, which allows for a more accurate representation of the system dynamics compared to existing methods. $f(x)$ describes the generic system dynamics with or without control, while $g(x)$ is determined by hardware design which dictates how controls influence the dynamics of the system. Hence it is often called the control influence matrix. These two functions represent very different physics, it is much better to estimate them separately.
2) The optimal feedback control is constructed to approximate the solution of the Hamilton-Jacobi-Bellman equation by directly estimating the value function derivative using neural networks. The neural networks activation functions are selected to be hyperbolic tangent functions. Selecting hyperbolic tangent functions as the activation functions eliminates the manually defining activation functions done in previous studies.
3) The simulation results are presented for a complex system, a delta robot, to illustrate the effectiveness of the proposed technique.

The rest of this paper organizes as follows: The formulation and preliminaries of the optimal tracking control system are given in Section 2. An effective neural networks model is established to reconstruct the dynamics of nonlinear systems in Section 3. Section 4 presents the derivation of the optimal tracking problem solution using neural networks. Section 5 discusses the stability of the developed control. In Section 6, we present simulation results to demonstrate the effectiveness of the proposed optimal tracking control scheme on a delta robot. Finally, the conclusions are drawn in Section 7.

2. PROBLEM STATEMENT AND PRELIMINARIES

Consider a nonlinear time-invariant dynamical system of the form

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t), \tag{1}$$

and the initial condition

$$x(0) = x_0, \tag{2}$$

where $x(t) \in R^n$ is the measurable system state vector, $u(t) \in R^m$ is the control input, $f(x(t)) \in R^n$ is an unknown vector of nonlinear functions, and $g(x(t)) \in R^{n\times m}$ is an unknown matrix of nonlinear functions. It is assumed that $f(0) = 0$ and $f(x) + g(x)u$ is Lipschitz continuous on $\Omega \subseteq R^n$ containing the origin, and that the system is stabilizable on Ω. The system is stabilizable in the sense that there exists a continuous control $u \in U$ that asymptotically stabilizes the system on Ω.

The optimal tracking control objective is to find a control $u^\star(t)$, that will allow the dynamical system to track a desired trajectory, $x_d(t)$ while minimizing a predefined performance function.

Define the tracking error as

$$e(t) \triangleq x(t) - x_d(t). \tag{3}$$

The optimal control consists of the feed-forward part, $u_d(t)$, and the optimal feedback part, $u_e(t)$.

$$u^\star(t) = u_d(t) + u_e(t). \tag{4}$$

Assume that the desired trajectory satisfies

$$\dot{x}_d(t) = f(x_d(t)) + g(x_d(t))u_d(t). \tag{5}$$

The feed-forward control, $u_d(t)$, can be calculated with the knowledge of the system dynamics $f(x)$, $g(x)$ and the existence of a pseudo inverse $g^{-1}(x)$ as

$$u_d(t) = g^{-1}(x_d(t))\Big(\dot{x}_d(t) - f(x_d)\Big). \tag{6}$$

Remark 1: The desired trajectory, $x_d(t)$, must be piece-wise continuously differentiable on $\Omega \subseteq R^n$.

The tracking error dynamics can be obtained by

$$\begin{aligned}\dot{e}(t) &= \dot{x}(t) - \dot{x_d}(t) \\ &= f(x(t)) + g(x(t))u(t) - \dot{x_d}(t).\end{aligned} \tag{7}$$

The optimal feedback part $u_e(t)$ is designed to minimize the following performance function

$$V(e(t)) = \int_t^{\infty} r(e(\tau), u_e(\tau))\, d\tau, \tag{8}$$

where $r(e(\tau), u_e(\tau)) = e^T(\tau)Qe(\tau) + u_e^T(\tau)Ru_e(\tau)$. $Q \in R^{n\times n}$ and $R \in R^{m\times m}$ are symmetric positive definite matrices. This quadratic cost function makes the system state follow the reference trajectory and causes the system control input to be close to the desired control, in order to keep the states to their reference values. Thus, for the optimal control problem, the state feedback control law $u_e(t)$ must not only stabilize the system (7) on Ω but also ensure that the value function in (8) is finite, i.e., the control law must be admissible.

Definition 1 (Admissible control)**:** A control $u_e(t)$ for the given dynamical system (7) is defined to be admissible with respect to value function in (8) on Ω, denoted by $u_e(t) \in \Psi(\Omega)$, if $u_e(t)$ is continuous on Ω, $u_e(0) = 0$, $u_e(e)$ stabilizes the dynamical system on Ω, and $V(e(0))$ is finite $\forall e_0 \in \Omega$.

For any admissible control $u_e(t)$, if the associated value function is continuously differentiable; then, the infinitesimal version of the value function known as nonlinear Lyapunov equation can be written as

$$\begin{aligned}0 &= r\big(e(\tau), u(\tau)\big) \\ &+ V_e^T\Big(f(x(t)) + g(x(t))u_e(t) - \dot{x_d}(t)\Big),\end{aligned} \tag{9}$$

where V_e is the partial derivative of the value function with respect to e. Note that the value function does not depend explicitly on time.

The Hamiltonian function is defined as

$$\begin{aligned}H(e, u_e, V_e) &= r\big(e(\tau), u(\tau)\big) \\ &+ V_e^T\Big(f(x(t)) + g(x(t))u_e(t) - \dot{x_d}(t)\Big).\end{aligned} \tag{10}$$

Let $V^{\star}(e)$ be the optimal value function defined by

$$V^{\star}(e) = \min_{u\in\Psi(\Omega)}\left(\int_t^{\infty} r(e(\tau), u_e(\tau))\, d\tau\right). \tag{11}$$

The optimal value function $V^{\star}(e)$ satisfies the HJB equation

$$0 = \min_{u_e\in\Psi(\Omega)}[H(e, u_e, V_e^{\star})]. \tag{12}$$

Assume that the solution of (12) exists and is unique, the optimal function can be derived as

$$u_e^{\star}(t) = -\frac{1}{2}R^{-1}g^T(x)V_e^{\star}(e). \tag{13}$$

By substituting the optimal control in the nonlinear Lyapunov equation, the HJB equation can be written in terms of $V_e^{\star}$

$$\begin{aligned}0 &= e^TQe + V_e^{\star T}(e)\Big(f(x) - \dot{x_d}\Big) \\ &- \frac{1}{4}V_e^{\star T}(e)g(x)R^{-1}g^T(x)V_e^{\star}(e),\end{aligned} \tag{14}$$

and

$$V^{\star}(0) = 0. \tag{15}$$

In the case of a linear system with a quadratic cost functional, this HJB equation is equivalent to the Riccati equation. To determine the nonlinear dynamical system optimal control solution, the HJB equation (14) must be solved for the value function and then insert the answer in (13) to get the optimal control. The nonlinear character of the HJB equation makes finding a solution difficult or impossible.

We can use the method mentioned earlier to construct the control if the system dynamics are known. However, getting the complete, or even partial, knowledge of nonlinear system dynamics is a complex undertaking in most circumstances. A neural network identifier is constructed for unknown nonlinear systems to learn the system dynamics.

3. ESTABLISHMENT OF THE NEURAL NETWORKS IDENTIFIER AND FEED-FORWARD CONTROL

In this section, a new neural networks structure is established to reconstruct the unknown system dynamics using available input-out data as shown in Fig. 1. Two neural networks are used in parallel to estimate $f(x)$ and $g(x)$. Let each neural network has a hidden layer and an output layer, respectively, with p and m neurons. Thus, the neural network model for the system is constructed as

$$\hat{f}(x) = W_f^{(2)^T}\sigma\Big(W_f^{(1)^T}x(t) + b_f^{(1)}\Big) + b_f^{(2)}, \tag{16}$$

$$\hat{g}(x) = W_g^{(2)^T}\sigma\Big(W_g^{(1)^T}x(t) + b_g^{(1)}\Big) + b_g^{(2)}, \tag{17}$$

where $W_f^{(2)}$, $W_f^{(1)}$, $W_g^{(2)}$, $W_g^{(1)}$ are estimated weight matrices with appropriate dimensions, and $b_f^{(2)}$, $b_f^{(1)}$, $b_g^{(2)}$, $b_g^{(1)}$ are the estimated bias vectors with suitable dimensions. $\sigma(\cdot)$ is the neural networks activation function and selected to be hyperbolic tangent function. As a result, the estimated system dynamics can be written as

$$\dot{\hat{x}}(t) = \hat{f}(x) + \hat{g}(x)u(t). \tag{18}$$

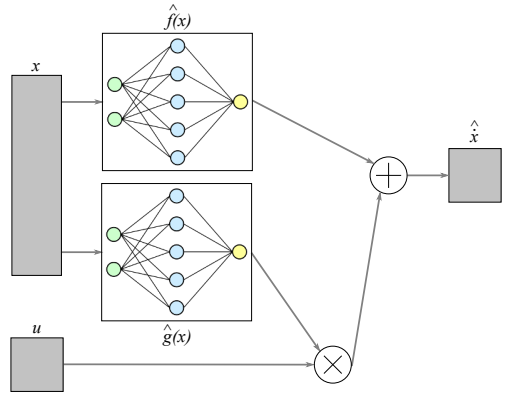

Fig. 1. Parallel neural networks identifier to reconstruct the unknown nonlinear dynamics system. Two neural networks, each with a hidden layer and an output layer, respectively, with p and m neurons, are used in parallel to estimate $f(x)$ and $g(x)$.

The neural networks weights and biases are adjusted using Levenberg–Marquardt backpropagation, a popular curve-fitting algorithm, to minimize the following error

$$J = \frac{1}{M}\sum_{j=1}^{M}\left(\dot{x}[j]-\hat{\dot{x}}[j]\right)^2, \tag{19}$$

where M is the total number of training data, $\dot{x}$ is the actual measurable data, and $\hat{\dot{x}}$ is the output of the neural network.

Then, by using the estimated functions $\hat{f}(x)$ and $\hat{g}(x)$, the feed-forward control input which guarantees perfect tracking can be computed as

$$u_d(t) = \hat{g}^{-1}(x_d(t))\left(\dot{x}_d(t)-\hat{f}(x_d)\right). \tag{20}$$

4. VALUE FUNCTION DERIVATIVE APPROXIMATION

Neural networks are adopted to solve the HJB equation. The value function derivative can be represented as using universal approximation capabilities of neural networks.

$$\frac{\partial V(e)}{\partial e} = W^{(2)^T}\phi\left(W^{(1)^T}e+b^{(1)}\right)+\varepsilon(e), \tag{21}$$

where $W^{(2)}\in R^{m\times N}$, $W^{(1)}\in R^{n\times N}$ are the ideal weight matrices, $b^{(1)}$ is the ideal bias vector, $\phi(\cdot):R^n\to R^N$ is the neural networks activation function vector, N is the number of neurons in the hidden layer, and $\varepsilon(e)$ is the neural networks approximation error. As the number of hidden layer neurons $N\to\infty$, the approximation error $\varepsilon(e)\to 0$ uniformly.

Remark 2: Based on HJB equation (14), $V^\star(0)=0$. Therefore, the neural networks activation functions should be selected to satisfy the condition $\phi(0)=0$. Hence, the neural networks activation functions are selected to be hyperbolic tangent function.

The value function can be estimated as

$$\begin{aligned} V(e) &= \int_0^e \frac{\partial V(\zeta)}{\partial \zeta}d\zeta \\ &= W^{(2)^T}\int_0^e \phi\left(W^{(1)^T}\zeta+b^{(1)}\right)d\zeta \\ &\quad+\int_0^e \varepsilon(\zeta)d\zeta. \end{aligned} \tag{22}$$

Remark 3: It is assumed that for fixed number of neurons, N, the neural networks approximation error $\varepsilon(e)$ and $\int_0^e \varepsilon(\zeta)\,d\zeta$ are bounded by constants on a compact set.

Based on the neural networks value function derivative approximation in (21), the Hamiltonian equation (10) becomes

$$\begin{aligned} H(e,u_e,W^{(i)}) &= r\big(e(t),u_e(t)\big) \\ &\quad+W^{(2)^T}\phi\left(W^{(1)^T}e+b^{(1)}\right) \\ &\quad\times\left(\hat{f}(x(t))+\hat{g}(x(t))u_e(t)-\dot{x}_d(t)\right) \\ &= \varepsilon_H. \end{aligned} \tag{23}$$

The residual error, ε_H, consists of dynamics estimation and value function derivatives approximation errors. Under the Lipschitz assumption on the dynamics, this residual error is bounded on a compact set.

4.1. Tuning and convergence of critic neural networks

The weights of the critic neural networks, $W^{(1)}$ and $W^{(2)}$ which provide the best approximate solution for (23) are unknown. Thus, the approximate value function derivative is defined as

$$\hat{V}_e(e) = \hat{W}^{(2)^T}\phi\left(\hat{W}^{(1)^T}e+\hat{b}^{(1)}\right), \tag{24}$$

where $\hat{W}^{(2)}$ and $\hat{W}^{(1)}$ are the current estimated values of the ideal neural networks weights $W^{(2)}$ and $W^{(1)}$. And $\hat{b}^{(1)}$ is the current estimation value of the ideal neural networks bias $b^{(1)}$. The approximate nonlinear Lyapunov equation is then

$$\begin{aligned} H(e,\hat{V}_e(e)) &= e^TQe+\hat{V}_e^T(e)\left(\hat{f}(x)-\dot{x}_d\right) \\ &\quad-\frac{1}{4}\hat{V}_e^T(e)\hat{g}(x)R^{-1}\hat{g}^T(x)\hat{V}_e(e) \\ &= \varepsilon. \end{aligned} \tag{25}$$

Given any admissible control policy $u(t)$, it is desired to select $\hat{W}^{(2)^T}$, $\hat{W}^{(1)^T}$, and $\hat{b}^{(1)}$ to minimize the squared residual error

$$E = \frac{1}{2}\varepsilon^T\varepsilon. \tag{26}$$

Then $\hat{W}^{(2)^T}\to W^{(2)^T}$, $\hat{W}^{(1)^T}\to W^{(1)^T}$, $\hat{b}^{(1)}\to b^{(1)}$, and $\varepsilon\to\varepsilon_H$. We select the tuning law for the neural network weights and biases using adaptive moment estimation (Adam).

Algorithm 1: Overview of the Adam optimizer algorithm for tuning the weights and biases of the neural network.

input : Step size, α
Exponential decay rates for the moment estimates, β_1, β_2
input : Initial parameter vector, θ_0
Initialize 1^{st} moment vector, $m_0 \leftarrow 0$
Initialize 2^{nd} moment vector, $v_0 \leftarrow 0$
Initialize time step, $t \leftarrow 0$
output: Tuned parameter vector, θ_t

while θ_t *not converged* **do**
$t \leftarrow t+1$;
$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot \partial E/\partial \theta_t$;
$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1-\beta_2) \cdot (\partial E/\partial \theta_t)^2$;
$\hat{m}_t \leftarrow m_t/(1-\beta_1^t)$;
$\hat{v}_t \leftarrow v_t/(1-\beta_2^t)$;
$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t/(\sqrt{\hat{v}_t}+\epsilon)$;
end

Algorithm 2: Overview of the algorithm for the critic neural networks tuning.

input : Error set, $e \in \Omega_e$
Desired trajectory, $x_d(t), \dot{x}_d(t)$
Q and R
input : Initial parameters, $\hat{W}_0^{(2)^T}$, $\hat{W}_0^{(1)^T}$, $\hat{b}_0^{(1)}$
output: Tuned parameters, $\hat{W}^{(2)^T}$, $\hat{W}^{(1)^T}$, $\hat{b}^{(1)}$

while *parameters not converged* **do**
for $i \leftarrow 1$ **to** 20 **do**
Select random error vector, $e \in \Omega_e$;
$x = x_d + e$;
Compute $\hat{f}(x)$;
Compute $\hat{g}(x)$;
Compute $\hat{V}_e(e)$, (24);
Compute ε, (25);
Compute E_i, (26) ;
end
$E = \frac{1}{20}\sum_{j=1}^{20} E_j$;
Update parameters using Adam optimizer;
end

Adam optimization [29] is a variant of stochastic gradient descent, based on adaptive estimation of first-order and second-order moments (Algorithm 1). It can be used to update network weights more quickly than traditional stochastic gradient descent.

Based on the above preparations, we now summarize the critic neural networks tuning in Algorithm 2.

The block diagram of the proposed data-driven optimal tracking control system is shown in Fig. 2. The block diagram consists of a trained neural network identifier to estimate the unknown nonlinear dynamics system. A critic neural networks approximates the value function derivative. A feed-forward control ensures trajectory tracking, and optimal feedback control stabilizes the tracking error dynamics.

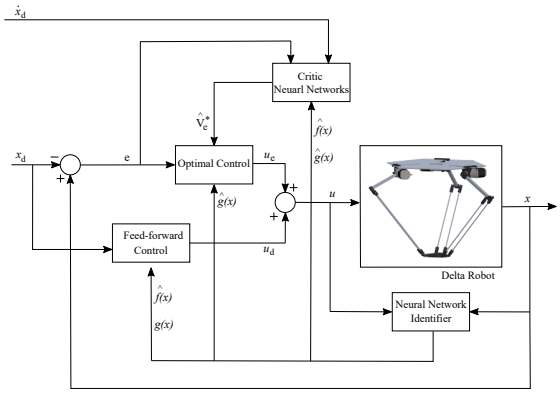

Fig. 2. Structure of the data-driven optimal tracking control system for a delta robot. The block diagram consists of a trained neural network identifier, critic neural networks, a feed-forward control, and optimal feedback control.

5. STABILITY ANALYSIS

It has been proved that the estimation error of the neural network can be arbitrarily small with bounded activation functions and a sufficiently large number of hidden layer neurons [30,31]. Considering the controlled system in (1) with unknown system dynamics. It is assumed all system states and control inputs are observable, and the approximate optimal control can be obtained. Let the critic network be given by (24) and the update law for the critic network be provided by Algorithm 1. Then, based on [30-32] it can be proved that the corresponding network weight estimation error are uniformly ultimately bounded by using Lyapunov stability analysis.

6. SIMULATION RESULTS

In this section, a simulation example is given to show the effectiveness of the proposed method. Delta robot is a three degree of freedom parallel manipulator in which the base platform is linked to a moving platform by three parallel identical kinematic chains, as shown in Fig. 3. A rotary motor actuates each active arm of the delta robot. The system state vector is $x = [\theta_1 \ \theta_2 \ \theta_3 \ \dot{\theta}_1 \ \dot{\theta}_2 \ \dot{\theta}_3]^T$ consisting of active arms angles and velocities, and the control input vector is $u = [\tau_1 \ \tau_2 \ \tau_3]^T$, the input torque to the motors.

Simulation tests are performed in MATLAB/Simulink environment by modeling the delta robot using the Simscape Multibody Toolbox. The parameters of the delta robot are summarized in Table 1. Also, a damping coefficient of 0.01 Nm/(deg/s) is added to all the universal

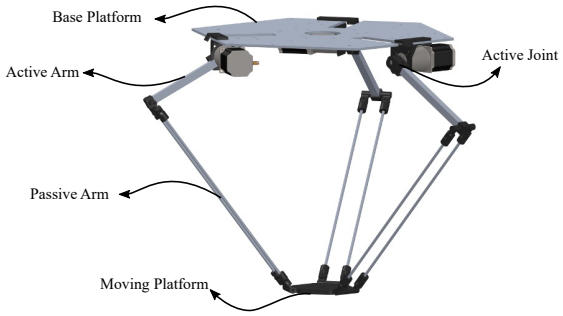

Fig. 3. Schematic view of the delta robot.

Table 1. Parameters of the delta robot.

Link	Dimension (m)	Mass (kg)
Active arm length	0.25	0.2
Passive arm length	0.52	0.6
Base platform radius	0.225	-
Moving platform radius	0.075	0.2

joints.

To begin with, the proposed neural network identifier is established by using two three-layer neural networks, which is chosen as $6-30-3$ structure with 6 input neurons, 30 hidden neurons, and 3 output neurons. The hidden layer uses the hyperbolic tangent function, and the output layer uses the linear function. The neural network identifier is trained using 2000 data samples by using MATLAB neural network toolbox. Note that by choosing 30 hidden layer neurons, the neural network estimation error is ensured to be arbitrarily small. The accuracy of the neural identifier significantly affects the feed-forward control and the feedback control. Thus, we choose a large number of hidden layer neurons to improve the accuracy of neural network identifier.

We present two trajectory tacking examples to illustrate the implementation and control system performance of the proposed data-driven optimal trajectory tracking approach. In the first example, a smooth path is defined as the desired trajectory. The second trajectory is a non-smooth but piece-wise differentiable path.

6.1. Smooth trajectory tracking

For the first reference trajectory, a spiral path is defined as follows:

$$\begin{cases} X = 0.2\cos(4\pi \times t/100), \\ Y = 0.2\sin(4\pi \times t/100), \\ Z = (-0.2 \times t)/100 - 0.4, \end{cases} \tag{27}$$

where $t \in [0 \;\; 100]$. Using the inverse kinematics of the delta robot, the desired active arm angles and then the desired angular velocities and accelerations can be computed. The value function is defined as (8), where $Q = diag[2 \;\; 5 \;\; 2 \;\; 1 \;\; 1 \;\; 1]$ and $R = 1$.

The critic neural network is chosen as three-layer neural networks with the structure of $6-10-3$ with 6 input neurons, 10 hidden neurons and 3 output neurons. The hyperbolic tangent function is used as the activation function. The initial weights of the critic network is all set to be random in $[-0.1 \;\; 0.1]$. The Adam optimizer parameters are set as $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$.

Next, the trained critic network is considered as the tracking control to make the delta robot track the reference trajectory. The results of the spiral tracking are illustrated in Fig. 4. In addition, tracking errors are depicted in Fig. 5. The feed-forward, feedback optimal control input, and the tracking control are given in Fig. 6. The trend of the tracking error to zero means that the system states successfully tracked the desired reference trajectory. The small bounded tracking errors after 20 s in Fig. 5 might be due to estimation errors of unknown dynamics and value function derivative.

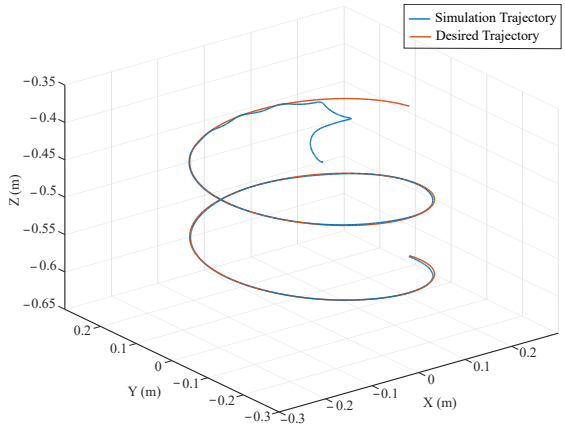

Fig. 4. Delta robot tracking a smooth desired trajectory using neural network based optimal tracking control.

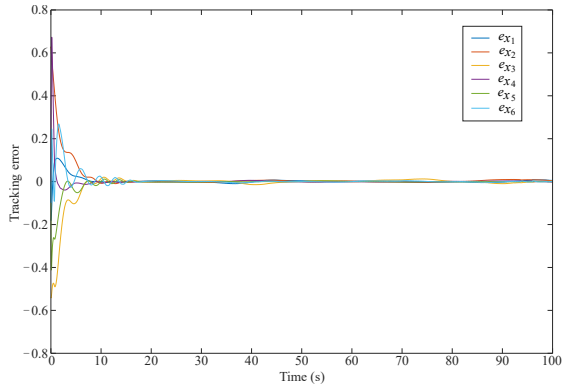

Fig. 5. Delta robot tracking error in smooth trajectory tracking using neural network based optimal tracking control. The tracking error is bounded, and it tends to go to zero.

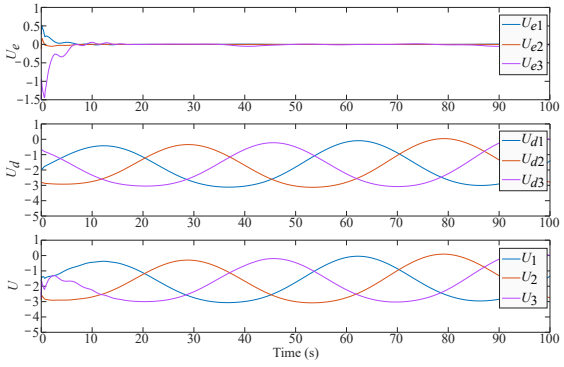

Fig. 6. Delta robot tracking control input in smooth trajectory tracking using neural network based optimal tracking control. The optimal control, $u(t)$, consists of the feed-forward part, $u_d(t)$, and the optimal feedback part, $u_e(t)$.

6.2. Non-smooth trajectory tracking

For the second reference trajectory, a non-smooth but piece-wise differentiable trajectory is defined. Using the inverse kinematics of the delta robot, the desired active arm angles and then the desired angular velocities and accelerations can be computed. The value function is defined as (8), where $Q = diag[3\ 6\ 2\ 1\ 1\ 2]$ and $R = 1$.

The critic neural network is chosen as three-layer neural networks with the structure of $6-10-3$ with 6 input neurons, 10 hidden neurons and 3 output neurons. The hyperbolic tangent function is used as the activation function. The initial weights of the critic network is all set to be random in $[-0.5\ \ 0.5]$. The Adam optimizer parameters are set as $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$.

Next, the trained critic network is considered as the tracking control to make the delta robot to track the reference trajectory. The results of the non-smooth path tracking are illustrated in Fig. 7. In addition, tracking errors are depicted in Fig. 8. The feed-forward, feedback optimal control input, and the tracking control are given in Fig. 9. The simulation results demonstrate that the approximate optimal tracking control law derived by the proposed algorithm is able to provide the good control performance. According to Fig. 8 tracking errors around the sharp edges are higher compared to other parts of the trajectory. However, as it can be seen, the tracking error is bounded, and it tends to go to zero.

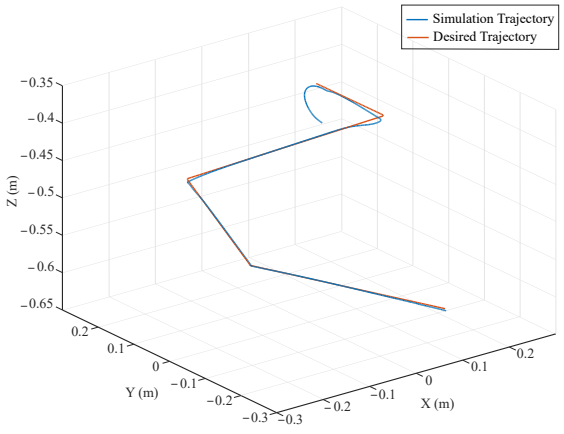

Fig. 7. Delta robot tracking a non-smooth desired trajectory using neural network based optimal tracking control.

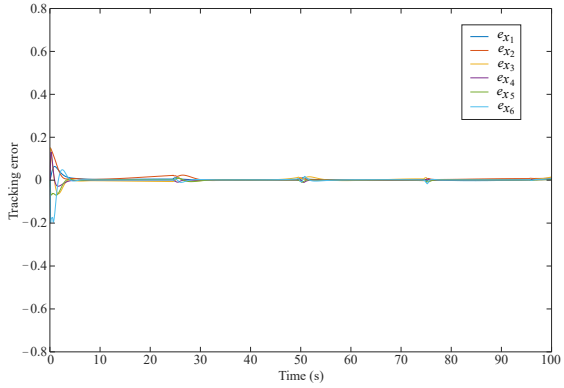

Fig. 8. Delta robot tracking error in non-smooth trajectory tracking using neural network based optimal tracking control. The tracking error is bounded, and it tends to go to zero.

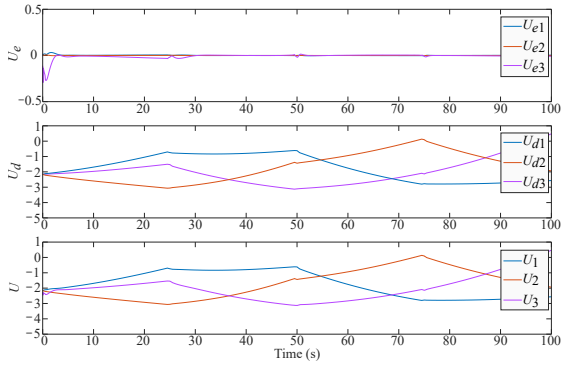

Fig. 9. Delta robot tracking control input in non-smooth trajectory tracking using neural network based optimal tracking control. The optimal control, $u(t)$, consists of the feed-forward part, $u_d(t)$, and the optimal feedback part, $u_e(t)$.

Remark 4: In this paper, the neuron sizes of the critic networks are chosen by experiments. We believe that for a specific nonlinear system, several effective structures of neural networks may exist, which can implement the developed data-driven algorithm to obtain the optimal control law.

In Table 2, the averages of the absolute tracking error in X-, Y-, and Z-directions and the Euclidean distance error

Table 2. Average of the absolute tracking error in trajectory tracking using neural network based optimal tracking control.

	Duration (s)	e_X (mm)	e_Y (mm)	e_Z (mm)	e (mm)
Smooth path	$t \in [0\ 100]$	6.24	2.16	2.39	7.28
	$t \in [20\ 100]$	1.72	0.64	0.98	2.24
Non-smooth path	$t \in [0\ 100]$	1.44	0.96	1.25	2.41
	$t \in [20\ 100]$	1.01	0.56	0.86	1.61

during the entire path are shown. Euclidean distance error can be computed as

$$e = \sqrt{e_X^2 + e_Y^2 + e_Z^2}, \tag{28}$$

where e_X, e_Y, and e_Z are the error in X-, Y-, and Z-directions, respectively.

7. CONCLUSIONS

In this paper, an effective optimal tracking control scheme using neural networks is proposed for a class of unknown nonlinear systems. First, a data-driven identifier is constructed to estimate unknown functions $f(x)$ and $g(x)$. Then, the feed-forward control is derived based on the estimated dynamics. The HJB equation is solved by estimating the value function derivative using neural networks. By using the Lyapunov technique, the stability of the proposed method is proved. Finally, two numerical simulation examples on a delta robot are presented to display the effectiveness. In this work, neural networks identifier and critic neural networks are trained offline. However, in many cases, the desired trajectory is not available beforehand. Therefore, we intend to use the proposed optimal tracking control online and in real-time for future work.

CONFLICT OF INTEREST

The authors declare no conflicts of interest.

REFERENCES

[1] F. L. Lewis, D. L. Vrabie, and V. L. Syrmos, *The Tracking Problem and Other LQR Extensions*, ch. 4, pp. 177-212, John Wiley & Sons, Ltd, 2012.

[2] D. Liberzon, *Calculus of Variations and Optimal Control Theory: A Concise Introduction*, Princeton University Press, 2012.

[3] J. Löber, *Optimal Trajectory Tracking of Nonlinear Dynamical Systems*, Springer, 01 2017.

[4] H. Modares and F. L. Lewis, "Optimal tracking control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning," *Automatica*, vol. 50, no. 7, pp. 1780-1792, 2014.

[5] D. Wang, D. Liu, and Q. Wei, "Finite-horizon neuro-optimal tracking control for a class of discrete-time nonlinear systems using adaptive dynamic programming approach," *Neurocomputing*, vol. 78, no. 1, pp. 14-22, 2012.

[6] C. Mu, C. Sun, A. Song, and H. Yu, "Iterative gdhp-based approximate optimal tracking control for a class of discrete-time nonlinear systems," *Neurocomputing*, vol. 214, pp. 775-784, 2016.

[7] Y. Huang and D. Liu, "Neural-network-based optimal tracking control scheme for a class of unknown discrete-time nonlinear systems using iterative ADP algorithm," *Neurocomputing*, vol. 125, pp. 46-56, 2014.

[8] M. H. Cohen and C. Belta, "Approximate optimal control for safety-critical systems with control barrier functions," *Proc. of 59th IEEE Conference on Decision and Control (CDC)*, pp. 2062-2067, 2020.

[9] Y. Zhang, S. Li, and L. Liao, "Near-optimal control of nonlinear dynamical systems: A brief survey," *Annual Reviews in Control*, vol. 47, pp. 71-80, 2019.

[10] B. C. Csáji *et al.*, "Approximation with artificial neural networks," *Faculty of Sciences, Etvs Lornd University, Hungary*, vol. 24, no. 48, p. 7, 2001.

[11] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 6, pp. 2042-2062, 2018.

[12] Y. Zhu and D. Zhao, "Comprehensive comparison of online adp algorithms for continuous-time optimal control," *Artificial Intelligence Review*, vol. 49, pp. 531-547,.2018.

[13] K. G. Vamvoudakis and F. L. Lewis, "Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem," *Automatica*, vol. 46, no. 5, pp. 878-888, 2010.

[14] H.-N. Wu and B. Luo, "Neural network based online simultaneous policy update algorithm for solving the hji equation in nonlinear h_∞ control," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 12, pp. 1884-1895, 2012.

[15] D. Liu, H. Li, and D. Wang, "Online synchronous approximate optimal learning algorithm for multi-player non-zero-sum games with unknown dynamics," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 8, pp. 1015-1027, 2014.

[16] H. Modares, F. L. Lewis, and M.-B. Naghibi-Sistani, "Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems," *Automatica*, vol. 50, no. 1, pp. 193-202, 2014.

[17] B. Luo, H.-N. Wu, T. Huang, and D. Liu, "Data-based approximate policy iteration for affine nonlinear continuous-time optimal control design," *Automatica*, vol. 50, no. 12, pp. 3281-3290, 2014.

[18] H. Modares, F. L. Lewis, and Z.-P. Jiang, "H_∞ tracking control of completely unknown continuous-time systems via off-policy reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 10, pp. 2550-2562, 2015.

[19] Q.-Y. Fan and G.-H. Yang, "Adaptive actor–critic design-based integral sliding-mode control for partially unknown nonlinear systems with input disturbances," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 1, pp. 165-177, 2016.

[20] D. Wang, D. Liu, Q. Zhang, and D. Zhao, "Data-based adaptive critic designs for nonlinear robust optimal control with uncertain dynamics," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 11, pp. 1544-1555, 2016.

[21] H. Zhang, H. Jiang, Y. Luo, and G. Xiao, "Data-driven optimal consensus control for discrete-time multi-agent systems with unknown dynamics using reinforcement learning method," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 5, pp. 4091-4100, 2017.

[22] H. Zhang, C. Qin, B. Jiang, and Y. Luo, "Online adaptive policy learning algorithm for h_∞ state feedback control of unknown affine nonlinear discrete-time systems," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2706-2718, 2014.

[23] B. Luo, D. Liu, T. Huang, and D. Wang, "Model-free optimal tracking control via critic-only q-learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 10, pp. 2134-2144, 2016.

[24] D. Ding, Z. Wang, Q.-L. Han, and G. Wei, "Neural-network-based output-feedback control under round-robin scheduling protocols," *IEEE Transactions on Cybernetics*, vol. 49, no. 6, pp. 2372-2384, 2019.

[25] S. Bhasin, R. Kamalapurkar, M. Johnson, K. G. Vamvoudakis, F. L. Lewis, and W. E. Dixon, "A novel actor–critic–identifier architecture for approximate optimal control of uncertain nonlinear systems," *Automatica*, vol. 49, no. 1, pp. 82-92, 2013.

[26] J. Na and G. Herrmann, "Online adaptive approximate optimal tracking control with simplified dual approximation structure for continuous-time unknown nonlinear systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 1, no. 4, pp. 412-422, 2014.

[27] F. F. M. El-Sousy, M. M. Amin, and A. Al-Durra, "Adaptive optimal tracking control via actor-critic-identifier based adaptive dynamic programming for permanent-magnet synchronous motor drive system," *IEEE Transactions on Industry Applications*, vol. 57, no. 6, pp. 6577-6591, 2021.

[28] J. Na, Y. Lv, K. Zhang, and J. Zhao, "Adaptive identifier-critic-based optimal tracking control for nonlinear systems with experimental validation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 1, pp. 459-472, 2022.

[29] D. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," *Proc. of International Conference on Learning Representations*, December 2014.

[30] C. Mu, D. Wang, and H. He, "Novel iterative neural dynamic programming for data-based approximate optimal control design," *Automatica*, vol. 81, pp. 240-252, 2017.

[31] D. Wang, M. Zhao, M. Ha, and J. Ren, "Neural optimal tracking control of constrained nonaffine systems with a wastewater treatment application," *Neural Networks*, vol. 143, pp. 121-132, 2021.

[32] R. Song, Y. Xie, and Z. Zhang, "Data-driven finite-horizon optimal tracking control scheme for completely unknown discrete-time nonlinear systems," *Neurocomputing*, vol. 356, pp. 206-216, 2019.

Akram Gholami received her B.Sc. degree in mechanical engineering from K. N. Toosi University of Technology, Iran; and an M.S. degree in mechanical engineering and an M.Eng. degree in agricultural and biological engineering, concurrently, from the University of Florida. She received her Ph.D. degree in mechanical engineering from University of California, Merced in 2022. Her work is focused on modeling, simulating, and controlling robots using data-driven algorithms and machine learning.

Jian-Qiao Sun earned his B.S. degree in solid mechanics from Huazhong University of Science and Technology in 1982, and a Ph.D. degree in mechanical engineering from UC Berkeley in 1988. In 1994, Dr. Sun joined the faculty at University of Delaware until 2007 when he moved to University of California at Merced. He is currently Professor of mechanical engineering. He serves as the Editor-in-Chief of the International Journal of Dynamics and Control. His research interests include vibrations, controls, energy harvesting, and data-driven modeling and analysis of complex systems.

Reza Ehsani received his Ph.D. degree in biological and agricultural engineering from the University of California, Davis. He was a faculty member at the Ohio State University and University of Florida before joining UC Merced in 2017. Currently, he is a professor of mechanical engineering at the University of California, Merced. His areas of research include engineering systems for agriculture, automation, and intelligent machines for production and postharvest of high value crops, precision agriculture, sensors for biotic and abiotic plant stress detection, mechanical harvesting machines, and robotic harvesting systems for fruit and nut trees.