World Scientific Research Journal Volume 7 Issue 3, 2021
ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0031

Manipulator Tracking Control based on DHP Algorithm

Huimin You[1, a]

[1]Software Engineering, Shanghai Maritime University, Shanghai, China

[a]maurayou@163.com

Abstract

The manipulator system is a multi-input, multi-output nonlinear system, its pose output accuracy is unstable, and there is a certain tracking error with the given signal. To improve the tracking accuracy of the manipulator system, by analyzing the BP neural network, combining the nonlinear factors of the manipulator system and the real-time requirements of its control, a neural network control method based on the DHP algorithm is proposed and applied to the control of the manipulator system Designing. Simulation experiments show that the proposed controller can track the given trajectory well, and is suitable for real-time control of the robotic arm system.

Keywords

DHP; Neural network; Robotic arm control; Trajectory tracking.

1. INTRODUCTION

The multi-joint manipulator has developed into the most widely used core equipment in the industrial control field. Designing an effective controller to realize the end of the manipulator can track the desired reference trajectory quickly and stably is one of the key issues for the control of the manipulator. In engineering practice, the modeling errors of the robotic arm system and the uncertainty caused by external interference and other factors have brought challenges to the unbiased trajectory tracking of the robotic arm.

In recent years, people have paid more and more attention to the tracking control problem of the manipulator [1]. The goal to be achieved is to reduce the deviation between the actual movement trajectory of the manipulator and the set trajectory and achieve the desired accuracy. However, the robot system is a nonlinear, strongly coupled, high-order, and multivariable system, and its mathematical model is a nonlinear model with a time-varying structure. Because robot actions can often be completed in different ways and paths, and the arm action solution is not unique, it is necessary to consider optimization decision-making and control issues under certain constraints, but the controller based on traditional algorithms can no longer meet the control requirements. With the development of robotics technology, its related control algorithms are gradually enriched, such as neural network control, adaptive control, PID control, robust control, and iterative learning control [2-6]. Because neural networks require relatively little system dynamics information. Therefore, neural network algorithms are widely used in the control of robot systems.

However, the neural network cannot achieve an accurate approximation of the nonlinear function, and even for the same control task that has been experienced many times, the neural network still needs to perform redundant and cumbersome training every time it is executed [7-10]. Therefore, the learning ability of neural networks is quite limited. Compared with the existing neural network theory, the emerging deterministic learning theory has more in-depth research on the learning ability of the neural networks.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0031

This paper proposes a controller combined with adaptive dynamic programming based on neural network control. The adaptive dynamic programming algorithm obtains the approximate optimal solution of the system through the function approximate structure, which is very suitable for solving linear or nonlinear optimal control problems.

The structure of this article is as follows. In the second part, some basic knowledge and preparations of the robotic arm are introduced. In the third part, the design ideas and detailed information of the controller are introduced. In the fourth part, simulation experiments are given. Finally, the fifth section summarizes the conclusions and prospects.

2. QUESTIONS

Consider an n-joint robot whose dynamic performance can be described by a second-order nonlinear differential equation [11]:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G\left(q\right) + F(\dot{q}) + \tau_d = \tau \tag{1}$$

Among them, $q \in R^n$ is the joint angular displacement, $M(\mathrm{q}) \in R^{n\times n}$ is the inertia matrix of the robot, $C(\mathrm{q},\dot{\mathrm{q}}) \in R^n$ is the centrifugal force and Coriolis force, $G(\mathrm{q}) \in R^n$ is the gravity term, $F(\dot{\mathrm{q}}) \in R^n$ is the friction torque, $\tau \in R^n$ is the control torque, and $\tau_d \in R^n$ is the external disturbance.

The dynamic characteristics of the robotic arm are as follows:

(1) Feature 1, $M(\mathrm{q}) - 2C(\mathrm{q},\dot{\mathrm{q}})$ is an oblique symmetric matrix, $x^T(M(\mathrm{q}) - 2C(\mathrm{q},\dot{\mathrm{q}}))x = 0$;

(2) Feature 2, the inertial matrix $M(\mathrm{q})$ is a symmetric positive definite matrix, there are positive numbers m_1 and m_2, which satisfy the following inequalities:

$$m_1 \|x\|^2 \le x^T M(\mathrm{q})x \le m_2 \|x\|^2 \tag{2}$$

(3) Feature 3: There is a parameter vector that depends on the parameters of the manipulator, so that A, B, C, and D satisfy the linear relationship:

$$M(\mathrm{q})\vartheta + \mathrm{C}(\mathrm{q},\dot{\mathrm{q}})\rho + G(\mathrm{q}) + F(\dot{\mathrm{q}}) = \Phi(\mathrm{q},\dot{\mathrm{q}},\rho,\vartheta)\mathrm{P} \tag{3}$$

Where, $\Phi(\mathrm{q},\dot{\mathrm{q}},\vartheta,\rho) \in R^{n\times m}$ is the regression matrix of the known joint variable functions, which is the known function matrix of the robot's generalized coordinates and its derivatives, and $P \in R^n$ is the unknown steady parameter vector describing the quality characteristics of the robot.

This article mainly uses a typical double-joint rigid manipulator for research. The schematic diagram of the manipulator is shown in Figure 1.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0031

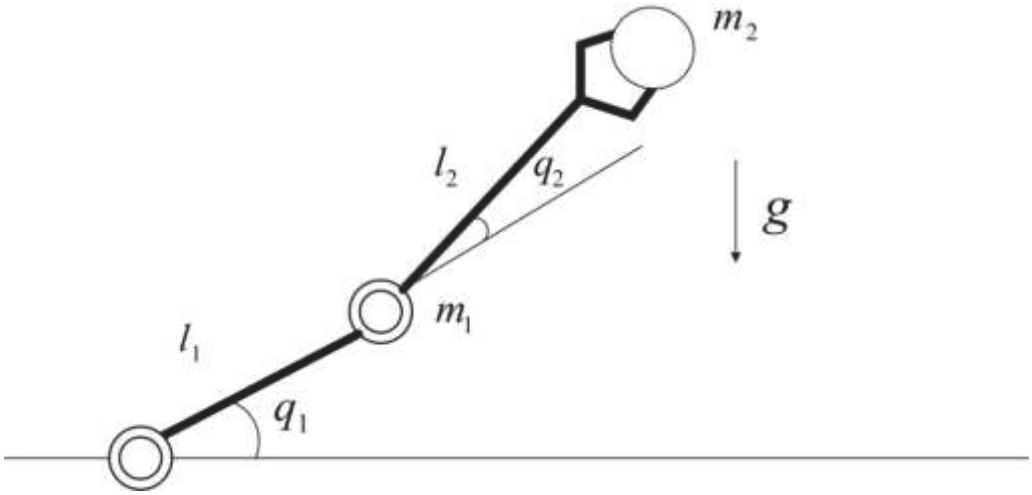

Figure 1. Schematic diagram of the robotic arm

Where m_i and l_i are the mass and length of the ith bracket; q_1 and q_2 are the position angles of joint 1 and joint 2.

The control goal of this paper is to drive the dual-joint manipulator to track the reference position trajectory at the reference speed and to minimize the position tracking error and speed tracking error.

3. CONTROLLER DESIGN BASED ON DHP ALGORITHM

This section mainly introduces the main design ideas of the controller and the model design of the robot arm. Then the design of the neural network is introduced in detail.

3.1. Main Design Ideas

The robotic arm is a complex system with multiple inputs, multiple outputs, and strong coupling. It is difficult to achieve precise control. For this reason, the dynamic control of the robotic arm is not considered first, and only the motion control is performed to enable it to accurately track the given trajectory curve. The basic control structure is shown in the following figure.

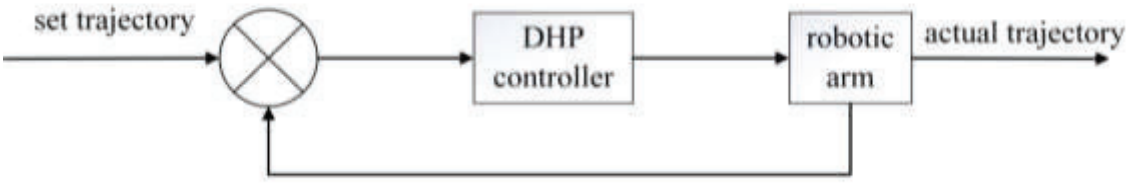

Figure 2. The motion control diagram of the robotic arm

3.2. DHP Controller

The controller used in this paper is the DHP structure in the adaptive dynamic algorithm. Its structure is shown in Figure 4. The structure of DHP includes a criticl network, action network, and model network. The functions and roles of the control network and model network are the same as those of HDP. But for DHP, the critical network will approximate the reciprocal of the performance index function J to the state x instead of the performance index function J itself, where $\frac{\partial J(x(k))}{\partial x(k)}$ is also called the costate. For this, we need to know the derivative

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0031

$\frac{\partial l(x(k),u(k))}{\partial x(k)}$ of the utility function to the state and the derivative $\frac{\partial f(x(k),u(k))}{\partial x(k)}$ of the system function to the state. $U(x(k),u(k))$ is the utility function. The dotted line in the figure indicates the direction of error propagation of the weight vector update.

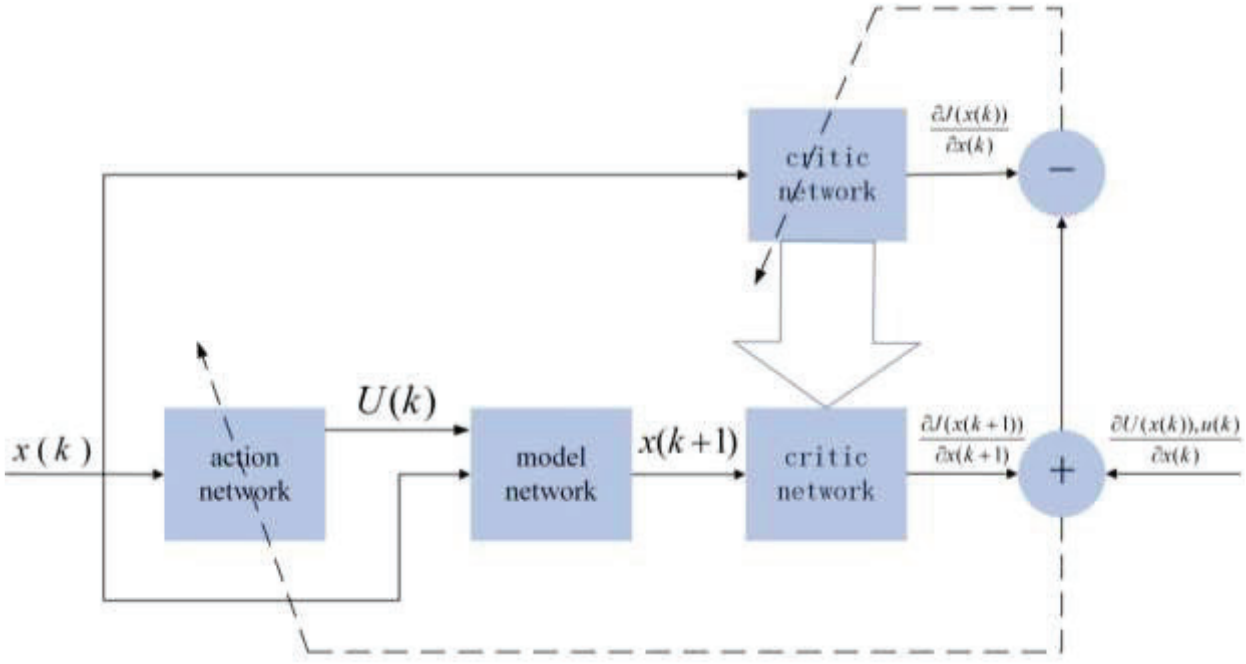

Figure 3. DHP structure diagram

The DHP algorithm iterates the derivative of the state according to the performance index function and the utility function, as shown in the following formula

$$\frac{\partial l(x(k))}{\partial x(k)} = \frac{\partial l(x(k),u(x(k)))}{\partial x(k)} + \frac{\partial l(x(k+1))}{\partial x(k)} \tag{4}$$

Among them, $u(x(k))$ is the feedback control variable, and the co-states $\frac{\partial J(x(k))}{\partial x(k)}$ and $\frac{\partial J(x(k+1))}{\partial x(k)}$ are the output of the evaluation network. If the weight of the critical network is set to w, we make the right formula of (4)

$$e(x(k),w) = \frac{\partial l(x(k),u(x(k)))}{\partial x(k)} + \frac{\partial l(x(k+1),w)}{\partial x(k)} \tag{5}$$

By adjusting the evaluation network weight w, the following mean square error function is minimized

$$w^* = \arg\inf_w \left\{ \left| \frac{\partial J(x(k),w)}{\partial x(k)} - e(x(k),w) \right|^2 \right\} \tag{6}$$

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0031

Obtain the optimal agreement state. According to the principle of optimality, optimal control should meet the necessary conditions of first-order differential, namely

$$\begin{aligned}\frac{\partial J^*(x(k))}{\partial u(k)} &= \frac{\partial l(x(k),u(x(k)))}{\partial u(k)} + \frac{\partial J^*(x(k+1))}{\partial u(k)} \\ &= \frac{\partial l(x(k),u(x(k)))}{\partial u(k)} + \frac{\partial J^*(x(k+1))}{\partial x(k+1)} \frac{\partial f(x(k),u(k))}{\partial u(k)}\end{aligned} \tag{7}$$

So get optimal control

$$u^* = \arg\inf_u \left(\left| \frac{\partial J(x(k))}{\partial u(k)} - \frac{\partial l(x(k),u(k))}{\partial u(k)} - \frac{\partial J^*(x(k+1))}{\partial x(k+1)} \frac{\partial f(x(k),u(k))}{\partial u(k)} \right| \right) \tag{8}$$

Among them, $\frac{\partial J^*(x(k+1))}{\partial x(k+1)}$ is the optimal co-state satisfying formula (6).

The following will introduce the main components in the DHP controller, namely the critical network, and explain in detail the algorithm and formula derivation of the module.

3.3. Critical Network

The TD algorithm updates the value function online, it will get the state value of the current state, and the state value only needs to wait until it jumps to the next state. However, there is an error between the actual value and the estimated value. Since this is an update process, the purpose is to minimize the error between the lowest predicted value and the actual value. Therefore, the learning goal of the critical network in the controller is to minimize the error between the predicted value and the actual value while optimizing the future cumulative "reward". Therefore, based on the above idea, the prediction error of the critical element is defined as for formula (9).

$$E_j(k) = \lambda_j(k) - \frac{\partial U(k)}{\partial x_j(k)} - \gamma \frac{\partial J(k+1)}{\partial x_j(k)} \tag{9}$$

And the objective function to be minimized in the critical network is

$$E_c(k) = \frac{1}{2} \sum_{j=1}^{n} E_j^2(k) \tag{10}$$

In this article, the critical network is constructed by a BP neural network with hidden layers. The network structure is shown in Figure 4.

The training of the critical network consists of forwarding calculation and reverse error propagation process. The forward calculation process of the critical network is

$$c_{h1j}(k) = \sum_{i=1}^{n} x_i(k) \bullet W_{c1ij}(k), \quad j = 1, \cdots, kj \tag{11}$$

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0031

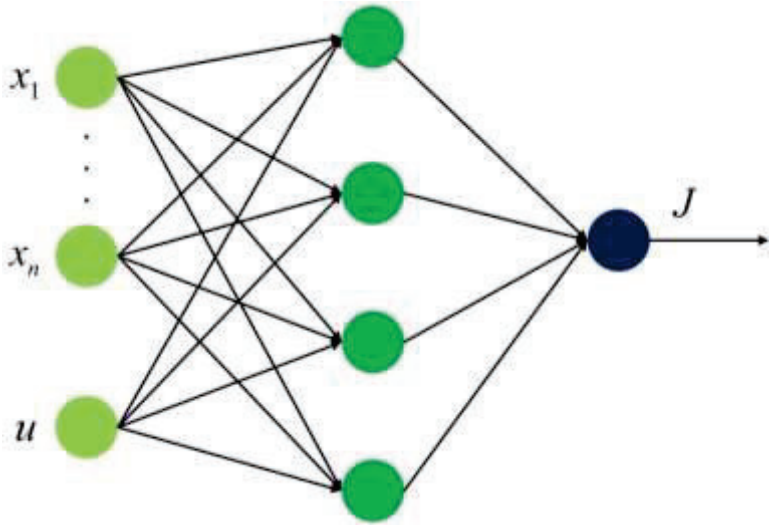

Figure 4. Neural network structure diagram

$$c_{h2j}(k)=\frac{1-e^{-c_{h1j}(k)}}{1+e^{-c_{h1i}(k)}},\quad j=1,\cdots,kj \tag{12}$$

$$\lambda_i(\mathrm{k})=\sum_{j=1}^{kj}c_{jh2}(k)\bullet W_{c2ji}(k),\ \ i=1,\cdots,n \tag{13}$$

- $c_{h1j}(k)$ is the input of the jth node of the evaluation network,
- $c_{h2j}(k)$ is the corresponding output of the jth hidden node,

According to error backpropagation and chain rules, the update process of the critical network is as follows:

(1) W_{c2} (Hidden to the output layer)

$$\Delta W_{c2ij}(k)=l_c(k)\left[-\frac{\partial E_c(k)}{\partial W_{c2ij}(k)}\right] \tag{14}$$

$$\frac{\partial E_c(k)}{\partial W_{c2ij}(k)}=\frac{\partial E_c(k)}{\partial E_j(k)}\frac{\partial E_j(k)}{\partial \lambda_j(k)}\frac{\partial \lambda_j(k)}{\partial W_{c2ij}(k)}=E_j(k)c_{h2i}(k) \tag{15}$$

(2) W_{c2} (input to hidden layer)

$$\Delta W_{c1ij}(k)=l_c(k)\left[-\frac{\partial E_c(k)}{\partial W_{c1ij}(k)}\right] \tag{16}$$

$$\frac{\partial E_c(k)}{\partial W_{c1ij}(k)}=\left[\sum_{l=1}^{n}\frac{\partial E_c(t)}{\partial E_l(k)}\frac{\partial E_l(k)}{\partial \lambda_l(k)}\frac{\partial \lambda_l(k)}{\partial c_{h2j}(k)}\right]\frac{\partial c_{h2j}(k)}{\partial c_{h1j}(k)}\frac{\partial c_{h1j}(k)}{\partial W_{c1ij}(k)} \tag{17}$$

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0031

4. SIMULATION

Choose a two-joint robotic arm system, and its dynamic model is

$$M(q)\ddot{q}+V(q,\dot{q})\dot{q}+G(q)+F(\dot{q})+\tau_d=\tau$$

Where

$$M(q)=\begin{bmatrix} p_1+p_2+2p_3\cos q_2 & p_2+p_3\cos q_2 \\ p_2+p_3\cos q_2 & p_2 \end{bmatrix}$$

$$V(\mathrm{q},\dot{\mathrm{q}})=\begin{bmatrix} -p_3\dot{q}_2\sin q_2 & -p_3(\dot{q}_1+\dot{q}_2)\sin q_2 \\ p_3\dot{q}_1\sin q_2 & 0 \end{bmatrix}$$

$$G(q)=\begin{bmatrix} p_4 g\cos q_1+p_5 g\cos(q_1+q_2) \\ p_5 g\cos(q_1+q_2) \end{bmatrix}$$

$$F(\dot{q})=0.2\,\mathrm{sgn}(\dot{q})$$

$$\tau_d=\begin{bmatrix} 0.1\sin(t) \\ 0.1\sin(t) \end{bmatrix}$$

Take $p=[p_1,p_2,p_3,p_4,p_5]=[2.9\ 0.76\ 0.87\ 3.04\ 0.87]$. The parameters of the robotic arm system are: $g=9.8m/s^2$. Set the initial state of the system as $[0.09\ 0\ -0.09\ 0]$ and the angle commands of the two joints as $q_{1d}=0.1\sin t$ and $q_{2d}=0.1\sin t$ respectively.

The neural network is simulated, its network input is $z=[e\ \dot{e}\ q_d\ \dot{q}_d\ \ddot{q}_d]$, the number of hidden layers is 1, and the number of neurons $n=20$ in the initial weight of the network is zero.

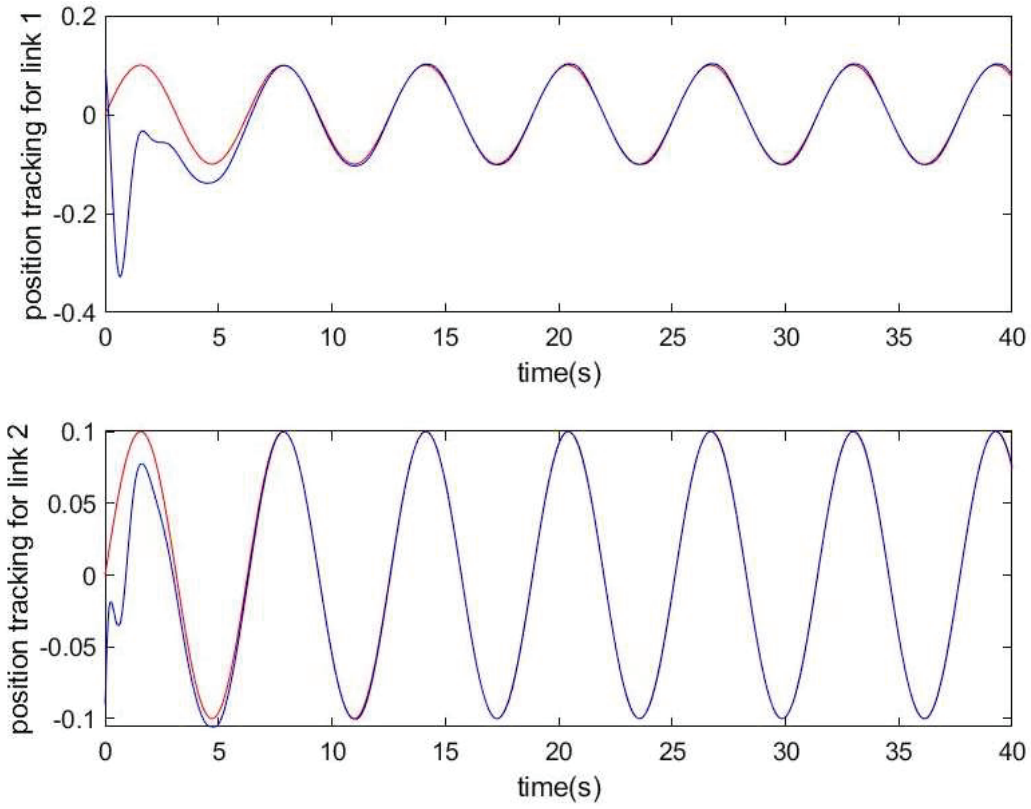

Figure 5. Angle tracking of joint 1 and joint 2

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0031

Figures 6 and 7 are the simulation results of the dual-joint robotic arm in Matlab. Figures 6 and 7 are the angle tracking and angular velocity tracking of joint 1 and joint 2, respectively. It can be seen from the figure that the red line represents the expected trajectory, and the blue line represents the actual trajectory.

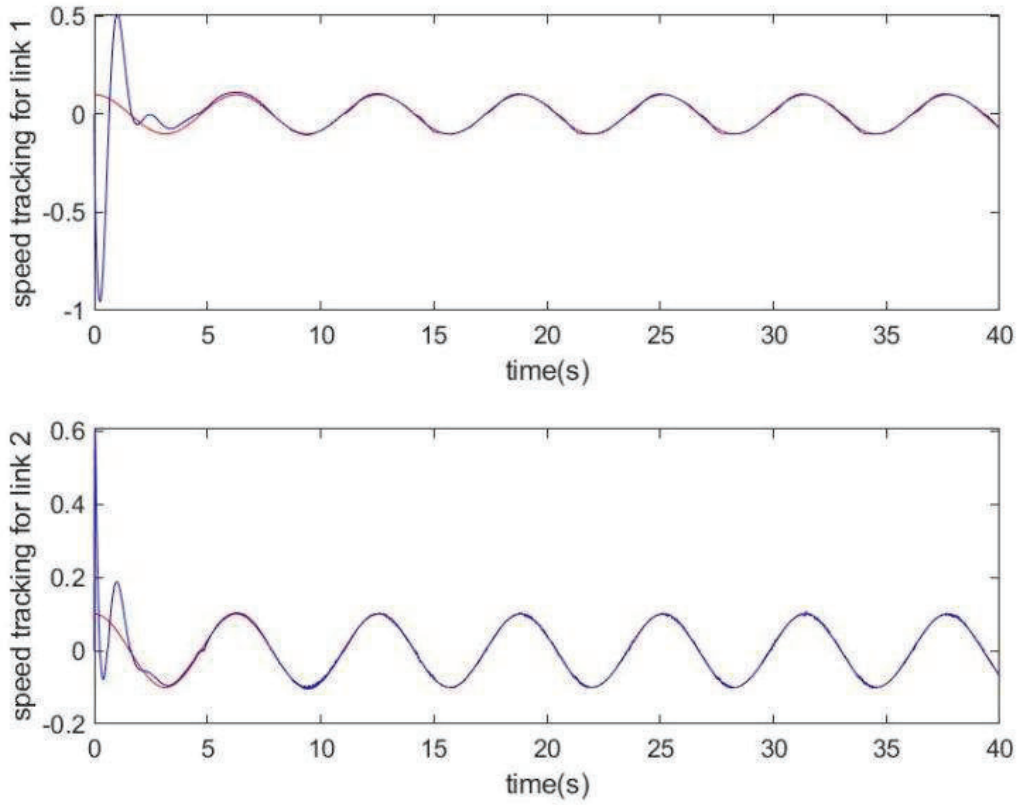

Figure 6. Angular velocity tracking of joint 1 and joint 2

It can be seen from the simulation results that all the manipulator joints of the robot system can successfully track the set trajectory through the DHP algorithm, and the error can converge and approach zero.

5. CONCLUSION

Aiming at the tracking control problem of the manipulator, this paper proposes a controller based on the DHP algorithm, which uses a neural network to approximate the system to minimize position tracking error and speed tracking error.

Through the simulation of the dual-joint manipulator system, the experimental results further verify the feasibility of the tracking control of this scheme. The running trajectory is close to the set trajectory, and the speed, accuracy, and stability of smooth tracking are improved. The proposed controller based on the DHP algorithm can successfully drive the manipulator to track the reference position trajectory at the reference speed and has a good dynamic performance.

At the same time, machine learning is developing rapidly, and new learning algorithms are emerging endlessly. In the research process, to further reduce the error, it is necessary to try different algorithms to improve the design of the controller. According to the advantages of the algorithm, the algorithm is combined with practical applications to design a better controller to

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202103_7(3).0031

effectively solve the problem. At the same time, combining theory with the practice must be realized in the future.

REFERENCES

[1] Wang L , Liu Z , Chen C L P , et al. Energy-efficient SVM learning control system for biped walking robots.[J]. IEEE Trans Neural Netw Learn Syst, 2013, 24(5):831-837.

[2] Tao Zhao, Jiahao Liu, Songyi Dian, et al. Sliding-Mode-Control-Theory-Based Adaptive General Type-2 Fuzzy Neural Network Control for Power-line Inspection Robots. 2020, 401:281-294.

[3] Lili Zhang, Bing Chen, Chong Lin. Adaptive neural consensus tracking control for a class of 2-order multi-agent systems with nonlinear dynamics. 2020, 404:84-92.

[4] Pradhan S K , Subudhi B . Position control of a flexible manipulator using a new nonlinear self tuning PID controller[J]. IEEE/CAA Journal of Automatica Sinica, 2018:1-14.

[5] Yuan Zhou, Hesuan Hu, Yang Liu, et al. A distributed approach to robust control of multi-robot systems. 2018, 98:1-13.

[6] Shengda Liu, JinRong Wang, Dong Shen, et al. Iterative learning control for nonlinear differential inclusion systems. 2020, 30(7):2937-2952.

[7] Yen V T , Nan W Y , Cuong P V . Recurrent fuzzy wavelet neural networks based on robust adaptive sliding mode control for industrial robot manipulators[J]. Neural Computing and Applications, 2019, 31(11):6945-6958.

[8] Zerari N , Chemachema M , Essounbouli N . Neural Network Based Adaptive Tracking Control for a Class of Pure Feedback Nonlinear Systems With Input Saturation[J]. IEEE/CAA Journal of Automatica Sinica, 2019.

[9] Wu L , Yan Q , Cai J . Neural Network-Based Adaptive Learning Control for Robot Manipulators With Arbitrary Initial Errors[J]. IEEE Access, 2019, 7:180194-180204.

[10] Liu A , Zhao H , Song T , et al. Adaptive control of manipulator based on neural network[J]. Neural Computing and Applications, 2020:1-9.

[11] Lewis F L , Yegildirek A . Multilayer neural-net robot controller with guaranteed tracking performance[J]. IEEE Transactions on Neural Networks, 1996, 7(2):388-99.