The tracking method of robot arm trajectory based on artificial neural network evolution

Fusheng ZHANG¹, a, Jiazhong XU²,b and Juntao GU³,c

¹No.109 Zhongxing Road, Nangang District, Harbin University, China
²No.52 Xuefu Road, Nangang District, Harbin, China
³No.12 Huashan Road, Nangang District, Harbin, China

a mwzfs@163.com, b xujiazhong@126.com, c 592980678@qq.com

Keywords: robot, neural network, approximation algorithm, trajectory tracking.

Abstract: Artificial neural network evolutionary method is a new machine learning method. Aiming at the control problems of high time-varying, nonlinear and strong coupling of robot arm, a nonlinear control method based on neural network evolutionary approximation is proposed. Based on the kinematics calculation method of n-joint controlled object, a RBF neural network controller is designed to track and approximate the target trajectory. The stability and feasibility of the system are demonstrated by using the integral Lyapunov method. Taking the trajectory of a two-joint manipulator as an example of MATLAB simulation, the results show that the method can effectively reduce the system modeling error, accurately track the trajectory of the manipulator, and improve the accuracy of system control.

Introduction

A robot consisting of actuating joints, sensors, controllers and actuators is a highly complex, nonlinear and strongly coupled system. The neural network is an effective way to solve the control problems of highly nonlinear and uncertain systems [1,2]. Based on the evolutionary idea of neural network, the control problem of robot manipulator is studied in this paper. The characteristics of the structure is a high-order, multi-variable, strong coupling, time-varying parameters, difficult or undetectable partial state variables. With the application of robot in special environment, the traditional control method obviously can not meet the requirements of high-precision control. Therefore, the research of advanced control methods to improve the dynamic and static performance of robot has become one of the research hotspots. An adaptive control algorithm based on neural network evolutionary approximation is proposed to enhance the autonomous target tracking and intelligence ability of the robot. The research has important practical significance for exploring the research and application of special autonomous robot with high intelligence level.

Related research

Because the neural network based on machine learning can promote the evolution of control system, and can approximate the nonlinear model with arbitrary precision, and can improve the control performance of the system by self-learning and compensation of modeling error, the intelligent control based on neural network has been widely used in the adaptive control of manipulators [3,4]. The inverse kinematics of redundant manipulator based on neural network is adopted in paper [5], which has 7 degrees of freedom and large amount of calculation. RBF neural network is used to identify the input of the whole system, and adaptive control is realized in paper [6]. In reference [7], the RBF neural network is used to self-learn the correction term on-line, and the network weight self-adaptive learning law is established based on the stability theory to ensure the convergence of tracking error and the stability of the system. His research overlooks the limitations of gravity and external disturbances, and that the stability of the system must be based on a fixed base function center and a fixed base width. In this paper, an adaptive vector neural network control strategy based on model approximation is proposed for n-joint manipulator by combining neural network
evolutionary method. The model controller is designed and its stability and robustness are analyzed. The generalization ability of the control method is enhanced. The simulation results show that the method is effective.

Establishment of neural network evolution model

For the n joint manipulator with in-plane motion, its kinematic equation is as follows [8],

\[ M(x) = C(x, \dot{\mathbf{\Theta}}) \mathbf{\dot{x}} + G(x) = \tau - \tau_w. \]  

(1)

Among them, \( M(x) \) is a positive definite inertia matrix of \( n \times n \) order. \( C(x, \dot{\mathbf{\Theta}}) \) is a \( n \times n \) order centrifugal matrix and Coriolis force matrix. \( G(x) \) is \( n \times 1 \) order gravity matrix. \( x, \dot{\mathbf{\Theta}}, \ddot{\mathbf{\Theta}} \) are the displacement, velocity and acceleration vector of joints respectively. \( \tau \) is the joint torque. \( \tau_w \) is the external disturbance of joints. In the actual control process, there is modeling error between the theoretical model and the actual model. The unknown parameter \( M(x), C(x, \dot{\mathbf{\Theta}}), G(x) \) are obtained in equation (1). The output of neural network \( M_1(x), C_1(x, \dot{\mathbf{\Theta}}), G_1(x) \) are used to approximate them. These are,

\[ M(x) = M_1(x) + E_M, \]  

(2)

\[ C(x, \dot{\mathbf{\Theta}}) = C_1(x, \dot{\mathbf{\Theta}}) + E_C. \]  

(3)

\[ G(x) = G_1(x) + E_G. \]  

(4)

Among them, \( E_M, E_C \) and \( E_G \) are the approximation errors of \( M(x), C(x, \dot{\mathbf{\Theta}}) \) and \( G(x) \) respectively. The formula (2), (3) and (4) are brought in (1). That is,

\[ M(x) = M_1(x) + E_M + C(x, \dot{\mathbf{\Theta}}) + G(x) + E_C + E_G = [W_M^T] \cdot [H_M(x)] + [W_C^T] \cdot [H_C(z)] + [W_G^T] \cdot [H_G(x)] + E. \]  

(5)

In formula (5), \( W_M, W_C \) and \( W_G \) are the ideal weights of neural networks. \( H_M(x), H_C(x) \) and \( H_G(x) \) are hidden layer outputs, and \( E = E_M + E_C + E_G \).

The estimation function of \( M_1(x), C_1(x, \dot{\mathbf{\Theta}}), G_1(x) \) are represented by RBF neural network.

\[ \hat{M}_1(x) = [\hat{W}_M^T] \cdot [H_M(x)]. \]  

(6)

\[ \hat{C}_1(x, \dot{\mathbf{\Theta}}) = [\hat{W}_C^T] \cdot [H_C(z)]. \]  

(7)

\[ \hat{G}_1(x) = [\hat{W}_G^T] \cdot [H_G(x)]. \]  

(8)

Among them, \( \hat{W}_M, \hat{W}_C \) and \( \hat{W}_G \) are estimated values of \( W_M, W_C \) and \( W_G \) respectively, and \( z = [x^T, \dot{\mathbf{\Theta}}]^T \).

Adaptive controller design

The traditional adaptive control uses Jacobian inverse matrix to design the manipulator operation. However, the inverse solution of Jacobian matrix is complex and time-consuming. This paper uses error compensation method to track the target trajectory, which can expand the workspace of the manipulator, avoid falling into chaos, and does not need complicated calculation. The details are as follows:

Definition 1:

\[ e(t) = x_g(t) - x(t). \]  

(9)

\[ \dot{\mathbf{\Theta}} = r(t) + \dot{\mathbf{\Theta}}. \]  

(10)

\[ \dot{\mathbf{\Theta}} = \dot{\mathbf{\Theta}} + \dot{\mathbf{\Theta}}. \]  

(11)
\[ x_d(t) \] is the ideal location. \( x(t) \) is the actual location. \( e(t) \) is the location error.

Definition 2:
\[ r(t) = \xi(t) + \lambda e. \] (12)

There are:
\[ \xi = \xi + \lambda e, \quad \eta = \eta + \lambda \xi \] and \( \lambda > 0. \)

The formula (10) and (11) are substituted (1). Calculated and sorted out:
\[
\begin{align*}
\tau &= M(x) \xi + C(x, \xi, \xi) + G(x) + \tau_w \\
&= M(x) \xi + C(x, \xi, \xi) + G(x) - M(x) \xi - C(x, \xi) \\
&= \left[ \begin{bmatrix} W_M \end{bmatrix} \cdot \begin{bmatrix} H_M \end{bmatrix} \right] \xi + \left[ \begin{bmatrix} W_C \end{bmatrix} \cdot \begin{bmatrix} H_C \end{bmatrix} \right] \xi \\
&\quad + \left[ \begin{bmatrix} W_G \end{bmatrix} \cdot \begin{bmatrix} H_G \end{bmatrix} \right] - M(x) \xi - C(x, \xi) + E + \tau_w
\end{align*}
\] (13)

For the \( n \) joint manipulator system, the design controller can be expressed as [9]:
\[
\tau = r_m + K_p r + K_i \int_0^t \tau d t + \tau_r
\]
\[
= M_i(x) \xi + C_i(x, \xi, \xi) + G_i(x) + K_p r + K_i \int_0^t \tau d t + \tau_r
\]
\[
= \left[ \begin{bmatrix} \hat{W}_M \end{bmatrix} \cdot \begin{bmatrix} H_M \end{bmatrix} \right] \xi + \left[ \begin{bmatrix} \hat{W}_C \end{bmatrix} \cdot \begin{bmatrix} H_C \end{bmatrix} \right] \xi + \left[ \begin{bmatrix} \hat{W}_G \end{bmatrix} \cdot \begin{bmatrix} H_G \end{bmatrix} \right]
\]
\[
+ K_p r + K_i \int_0^t \tau d t + \tau_r \quad (K_p > 0, \quad K_i > 0)
\] (14)

Definition 3:
The nominal model control law is:
\[
\tau_m = M_i(x) \xi + C_i(x, \xi, \xi) + G_i(x).
\] (15)

Definition 4:
Robust term is:
\[ \tau_r = K \cdot \text{sgn}(r). \] (16)

The formula (13) and (14) can be obtained as follows:
\[
M(x) \xi + C(x, \xi, \xi) + K_p r + K_i \int_0^t \tau d t + \tau_r
\]
\[
= \left[ \begin{bmatrix} \hat{V}_M \end{bmatrix} \cdot \begin{bmatrix} H_M \end{bmatrix} \right] \xi + \left[ \begin{bmatrix} \hat{V}_C \end{bmatrix} \cdot \begin{bmatrix} H_C \end{bmatrix} \right] \xi
\]
\[
+ \left[ \begin{bmatrix} \hat{V}_G \end{bmatrix} \cdot \begin{bmatrix} H_G \end{bmatrix} \right] + E.
\] (17)

Among them, \( \hat{V}_M = W_M - \hat{W}_M, \hat{V}_C = W_C - \hat{W}_C \) and \( \hat{V}_G = W_G - \hat{W}_G \).

Formula (17) can also be written as:
\[
M(x) \xi + C(x, \xi, \xi) + K_i \int_0^t \tau d t + \tau_r
\]
\[
= -K_p - K_i \cdot \text{sgn}(r) + \left[ \begin{bmatrix} \hat{V}_M \end{bmatrix} \cdot \begin{bmatrix} H_M \end{bmatrix} \right] \xi
\]
\[
+ \left[ \begin{bmatrix} \hat{V}_C \end{bmatrix} \cdot \begin{bmatrix} H_C \end{bmatrix} \right] \xi + \left[ \begin{bmatrix} \hat{V}_G \end{bmatrix} \cdot \begin{bmatrix} H_G \end{bmatrix} \right] + E.
\] (18)

The adaptive law is used to in the paper [8].
\[
\hat{W}_{mk} = \Gamma_M \left[ \xi_{mk}(x) \right] \xi_{kr}
\] (19)
\[
\hat{W}_{ck} = \Gamma_C \left[ \xi_{ck}(z) \right] \xi_{kr}
\] (20)
\[
\hat{W}_{gk} = \Gamma_G \left[ \xi_{gk}(x) \right] r_k \quad (k=1,2, \ldots, n)
\] (21)
Stability analysis

According to the formula of control law (14), an integral Lyapunov function is proposed by reference [8].

\[
V = \frac{1}{2} r^T M_r + \frac{1}{2} \int_0^t r d\tau + \frac{1}{2} \sum_{k=1}^{n} (\hat{\theta} \Gamma \hat{\theta})
\]

\[
+ \frac{1}{2} \left( \sum_{k=1}^{n} (\hat{\theta} \Gamma \hat{\theta}) + \frac{1}{2} \sum_{k=1}^{n} (\hat{\theta} \Gamma \hat{\theta}) \right)
\]

The derivation of formula (22) can prove the convergence of the system. That is, the control method is feasible.

Simulation example

Based on the MATLAB simulation software, the kinematic equations of the rational dual joint mechanical arm system are chosen as follows[8]:

\[
M(x) \dot{\mathbf{q}} + C(x, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) + G(x) = \tau,
\]

\[
M(x) = \begin{bmatrix} p_1 + p_2 + 2 p_1 \cos x_2 & p_2 + p_3 \cos x_2 \\ p_2 + p_3 \cos x_2 & p_2 \end{bmatrix},
\]

\[
C(x, \dot{\mathbf{q}}) = \begin{bmatrix} -p_3 \sin x_2 & -p_3 (\dot{\mathbf{q}} + \dot{\mathbf{q}}) \sin x_2 \\ p_3 \dot{\mathbf{q}} \sin x_2 & 0 \end{bmatrix},
\]

\[
G(x) = \begin{bmatrix} p_4 g \cos x_1 + p_5 g \cos(x_1 + x_2) \\ p_5 g \cos(x_1 + x_2) \end{bmatrix},
\]

\[
p = [3.00 \ 0.75 \ 0.85 \ 3.05 \ 0.85]^T, \ x_0 = [0.02 \ 0.02]^T \text{ and } \dot{\mathbf{q}} = [0.0 \ 0.0]^T.
\]

Ideal tracking instruction is: \( x_{d1} = \frac{1}{2} \sin(\pi t) \) and \( x_{d2} = \sin(\pi t) \). The model control law adopts formula (12) ~ (16). The adaptive law adopts formula (19) ~ (21). We take the values as follows:

\[
K_p = \begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix}, \ K_i = \begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix}, \ K_r = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix} \text{ and } \lambda = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}.
\]

The gain of the adaptive (19) ~ (21) is as follows: \( \Gamma_{M_k} = 5, \Gamma_{C_k} = 10 \) and \( \Gamma_{G_k} = 5 \).

The structure of the neural network is 2-5-1. For the approximation of \( M \) and \( G \), the input of RBF neural network is \( z = [x_1 \ x_2 \ \dot{\mathbf{q}} \ \ddot{\mathbf{q}}] \). For the approximation of \( C \), the input of RBF neural network is \( z = [x_1 \ x_2 \ \dot{\mathbf{q}} \ \ddot{\mathbf{q}}] \).

Parameters of Gauss function: \( c_i = [-1.5 \ -1.0 \ -0.5 \ 0 \ 0.5 \ 1.0 \ 1.5] \) and \( b_i = 10 \). The initial weight of the neural network is 0. The simulation results are shown in Fig. 1~Fig. 4.
Fig. 1 shows the position tracking trajectories of joint 1 and joint 2 based on neural network approximation. Fig. 2 shows the speed tracking trajectories of joint 1 and joint 2 based on neural network approximation. This proves that there is almost no error in tracking accurately based on neural network.

Fig. 3 is the input control curve of the adaptive controller in joints 1 and joints 2. Fig. 4 is the tracking curve of the nominal control model approximating the actual model. However, there is a lack of prediction of the nominal model of the manipulator, which leads to $C$ divergence. In future research, we can optimize the system model by improving the activation function.

Conclusions

Thinking based on human learning perception and thinking evolution are trying to transplant to robots in the paper, an artificial neural network based perception method is proposed to improve the intelligent tracking of robot end-effector to target trajectory [9,10]. However, in practical applications, there are modeling errors between the nominal model and the real model due to measurement errors, external disturbances and other factors, which make the tracking error of the target trajectory can not converge to zero. Based on the multi-layer neural network model, this paper uses RBF neural network to study and identify the correction term, and uses Lyapunov stability theory for reference to prove the feasibility of RBF neural network control method. By introducing the robust term to compensate the modeling error, the trajectory tracking of the manipulator is realized accurately, and the global stability of the system is improved. Compared with the method of inverse of Jacobian matrix, the weight learning law of a target trajectory corrected method is simpler, which reduces the calculation of inverse of time-varying inertial matrix and reduces the computational complexity. Successful
experimental results demonstrate that the proposed approaches can be applied in various types of manipulation missions [11,12].

Acknowledgements
This work was financially supported by the Harbin special Foundation for scientific and technological innovation talents (RC2016JQ009005).

References