Posted on 24 Feb 2023 — The copyright holder is the author/funder. All rights reserved. No reuse without permission. — https://doi.org/10.22541/au.167722567.71561433/v1 — This a preprint and has not been peer reviewed. Data may be preliminary.

Trajectory tracking based on neural network sliding mode controller

JieYun Yu[1]

[1]Jinan University

February 24, 2023

Abstract

In this paper, we aim to improve the tracking performance of the manipulator joint system under the presence of the uncertainties, such as modelling error, friction, and external disturbance. Firstly, the nonsingular fast terminal sliding mode control is developed to guarantees a finite-time convergence and to solve the singular issue of the terminal sliding mode control. Secondly, in view of the established system model, an adaptive sliding mode controller (SMC) based on radial basis function neural network (RBFNN) and sliding mode variable structure control theory is designed for the tracking of the bi-joint manipulator and six-degree of freedom parallel robot. Finally, the results show that our method improves the robustness of the adaptive RBFNN controller further, weakens the chattering phenomenon, reduces error, and has an excellent control performance.

Trajectory tracking based on neural network sliding mode controller

Jieyun Yu

Department of Mathematics,

Jinan University,

Guangzhou,510000, China.

Email: 15919575526@163.com

Abstract: In this paper, we aim to improve the tracking performance of the manipulator joint system under the presence of the uncertainties, such as modelling error, friction, and external disturbance. Firstly, the nonsingular fast terminal sliding mode control is developed to guarantees a finite-time convergence and to solve the singular issue of the terminal sliding mode control. Secondly, in view of the established system model, an adaptive sliding mode controller (SMC) based on radial basis function neural network (RBFNN) and sliding mode variable structure control theory is designed for the tracking of the bi-joint manipulator and six-degree of freedom parallel robot. Finally, the results show that our method improves the robustness of the adaptive RBFNN controller further, weakens the chattering phenomenon, reduces error, and has an excellent control performance.

Keywords: neural network, sliding mode control, adaptive control law, trajectory tracking

1 Introduction

With the strategy of "Industry 4.0 Robotics Challenge", the manufacturing industry is developing rapidly in the direction of intelligence. In view of the manoeuvrability of the manipulator systems, it has been widely used in the agricultural, manufacturing industry, medical treatment, and other hazardous fields [1]. As a multi-input multi-output system with complicate structure and strong coupling, the robotic arm is often affected by unmodeled uncertainties, sensory detecting noise, and time-varying external disturbance in practice. However, the techniques of traditional robot control have many limitations and are difficult to simultaneously meet the accuracy and speed requirements of the actual system [2]. The methods applied to robotic systems have mostly been based on state feedback control schemes such as sliding mode control (SMC), adaptive control, PID control, and neural network control. Among them, the principle of SMC forces the state of the control system to move according to the predefined sliding mode state trajectory, which improves the tracking accuracy [3], and due to its insensitivity to parameter changes, SMC theory has broad applications. Reference [4] proposed an adaptive robust control based on an integral SMC with parameter variations and disturbance observer. The RBFNN-based control scheme is often used to identify the uncertain items in the dynamic system of multi-degree of freedom manipulators. Sun et al. [5] propose an adaptive controller based on a neural network for a class of nonlinear systems with input quantization, unknown function, and time delays. Traditional RBFNN controllers need the availability of a high-precision model to avoid high control chattering, but the poor initial state of the iteration will cause a low control accuracy of the system, in which the initial values of parameters determine whether the system converges, the number of nodes in the hidden layer pre-setting affect the convergent speed, and too many hidden neurons will inevitably lead to a significant reduction in the training process time efficiency.

The servos used in the manipulator typically cannot provide enough information for calculating the dynamic coupling. Work in [6] introduced disturbance observers to compensate the dynamic coupling. An approaching law and switching function were proposed in [7], but the method is time-consuming and complex for its adjusts parameters one at a time, and it is sometimes difficult to adjust the optimal parameters. Noteworthily, the robustness of RBF control is poor, i.e., the robotic arm is poorly resistant to disturbances, and SMC has good robustness. In [8], a neural network-based robust sliding-mode controller for a five-dof robot arm was proposed and RBFNN is used to compensate for the influence of the system uncertainty and the changes in the sliding-mode control parameter. Reference [9] respectively designed non-singular fast terminal SMC, output feedback, and adaptive SMC strategy to optimize the traditional SMC strategy, however, the chattering phenomenon still exists. Nojavanzadeh [10]. developed the SMC with adaptive fractional-order term based on an integrated adaptive robust gain law to estimate the boundary of the uncertainties.

Inspired by the above works of literature, in order to ensure that the end effector does not fail due to jitter during the execution of the prescribed task, and further improve the convergence performance of the closed-loop system within finite time, this paper adopts sliding mode control as an auxiliary control to robustly compensate the radial basis neural network system in real-time with bounded error perturbation. Then we prove the stability and robustness of the system to parameter variation by using the Lyapunov method. The results show that the adaptive sliding mode control method of RBFNNs can weaken the jitter and has good potential for engineering applications.

The main contributions of this paper are concluded as follows:

• A novel SMC scheme with a nonlinear fractional order sliding mode surface is designed for the first time, which guarantees a fast convergence rate and overcomes the chattering problem.

• Different from the other existing methods, the proposed adaptive law is designed to handle the parameter variation, and a weighted square synchronized updating law is designed for RBF-neurons to reduce the impact of improper parameter selection.

• The good transparency performance with both position tracking and force feedback is obtained. The experiments are carried out to demonstrate the theoretical analysis correctness and superiority of our proposed control law.

The rest of this paper is organized as follows. In section 2, the proposed controller of a bi-joint manipulator is designed and its stability is analysed. In section 3, the kinematics model of the six-axis parallel mechanical platform is introduced and then the adaptive controller is discussed. Subsequently, experiments to verify the proposed scheme compared with the classical neural network controller are described in Section 4. At last, we conclude the whole paper.

2 RBFNN-based adaptive sliding mode controller of bi-joint manipulator

2.1 Dynamic model

The simplified plane diagram structure of the two-link manipulator is shown in Figure 1. Considering a general n-link robot arm, its dynamics are governed by:

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

where $q, \ \dot{q}, \ \ddot{q}\in R^n$ are the position, angular velocity, and angular acceleration vectors of each joint, respectively. $M(q)\in R^{n\times n}$ denotes the inertia matrices, $C(q,\dot{q})\in R^{n\times n}$ is the centripetal Coriolis matrix, $G(q)\in R^n$ is the gravity term, $F(\dot{q})\in R^n$ is the friction effect, $\tau_d\in R^n$ stands for the disturbance term, $\tau\in R^n$ represents torque input.

In this work, the well-known dynamic properties of the robot arms in (1) are satisfied [1]:

Property 1: Inertial matrix M(q) is a symmetric and positive definite matrix with a bounded norm.

Property 2: $\dot{M}(q)-2C(q,\dot{q})$ is a skew-symmetric matrix. i.e.,

$$z^T\left(\dot{M}(q)-2C(q,\dot{q})\right)z=0,\forall z\epsilon R^n \tag{2}$$

let $e=q_d-q$ is the joint tracking error and q_d expressed the desired trajectory, the error function is defined as:

$$r=\dot{e}+\Lambda e \tag{3}$$

where $\Lambda=\Lambda^T>0$. then equation (1) can be rewritten as:

$$M\dot{r}=M(\ddot{q}_d+\Lambda\dot{e})+C\dot{q}+G+F+\tau_d-\tau=-Cr-\tau+f+\tau_d \tag{4}$$

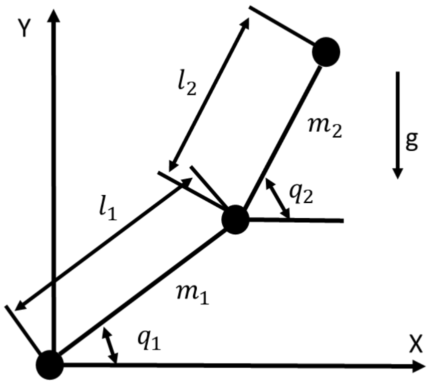


Figure 1 Structure sketch of the bi-joint manipulator

2.2 Description of RBFNN

As a kind of feed-forward network, RBFNN is capability of approximation to any single-valued continuous function with arbitrary precision. Figure 2 describes a typical RBFNN, which is a three-layer feed-forward network with a single hidden layer, which includes an input layer, a hidden layer, and an output layer. The first input layer consists of the signal source nodes; the neurons in the second implicit layer are transformed by radial basis functions, which are non-negative linear functions with radial symmetry and decay to the centroid; the third output layer responds to the input signal, and the implied layer is a linearly transformed function to the output. The signal in the layer is linearly weighted to the output value in the output layer.

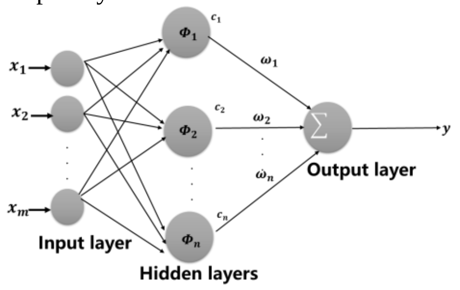


Figure 2 Structure of the RBFNN

The training process of the RBFNN determines by parameters: the clustering center, the width of the hidden neuron, and the weights of the hidden layer. The k-means method is used to determine clustering centre and the width of the hidden neuron, and we use the gradient method to train the network parameters b_j, c_j and ω:

$$c_k(t+1) = c_k(t) + \beta \frac{\partial J}{\partial c_k}$$
$$b_k(t+1) = b_k(t) + \beta \frac{\partial J}{\partial b_k}$$
$$\omega_k(t+1) = \omega_k(t) + \beta \frac{\partial J}{\partial \omega_k} \tag{5}$$

where J is the cost function given by $J = \frac{1}{t_N}\sum_{i=1}^{t_N} \|q_d(i) - q(i)\| + \|\dot{q}_d(i) - \dot{q}(i)\|$.

The hidden layer derives the input values with the radius basis function, we select the Gaussian function $\varphi_j(x) = \exp\left(-\|x - c_j\|^2 / b_j^2\right)$, j=1,2,,N as the basic function, where $x = [x_1, \cdots, x_n]^T$is the input variables, n is the input dimension, N is the number of neurons in the hidden layer, $\varphi_j(x)$ is the output of the hidden layer, c_j is the central of the hidden nodes, b_j is the width of the Gaussian function, and the output of the RBFNN is the linear superposition of the hidden layer node.

Lemma 1: Defining a continuous function $f(x)$ on a compact set Ω, there exists a weight vector W and a positive constant ε_N such that f can be approximated by a RBFNN:

$$\hat{f} = \widehat{W}\varphi \tag{6}$$

let $\widetilde{W} = W - \widehat{W}$, then we have:

$$\xi + \tilde{W}^T\varphi = f - \hat{W}^T\varphi \tag{7}$$

where ξ is the approximation error of the neural network, satisfying $\|\xi\| \le \varepsilon_N$, $W, \widehat{W}, \widetilde{W}$ represent the actual, approximate and the error value weight vectors, respectively.

Substitute equation (7) into equation (4), we get:

$$M\dot{s} = -(K_v + C)s + \tilde{W}^T\varphi + (\xi + \tau_d) + v \tag{8}$$

Assuming that the neural network system has ε and τ_d , the robust terms are expressed as follows:

$$v = -(\varepsilon_N + b_d)\operatorname{sgn}(s) \tag{9}$$

where sgn is the signum function given by:

$$sgn(s) = \begin{cases} 1, s > 0 \\ 0, s = 0 \\ -1, s < 0 \end{cases} \tag{10}$$

Define the RBFNN control law as:

$$u_{nn} = \widehat{W}^T\varphi + K_v r - v \tag{11}$$

the expected weight of RBFNNs can be acquired based on the optimization method as follows:

$$W_i^* := arg\min_{W_i}\{\sup|f(x) - W_i^T\Phi_i(x)|\} \tag{12}$$

where W_i is the estimated weight of W. from equation (4), it can be obtained:

$$W_i^{*T}\Phi_i(x) + \epsilon(x) = M\ddot{q} + C\dot{q} + G + F + \tau_d \tag{13}$$

among them, M, C, F, and G need to be measured and calculated to obtain their nominal parameter matrices.

2.3 Design of RBFNN-based sliding mode controller

The global slip surface [9] is chosen as equation (14) such that the controlled system achieves a fast response and finite-time convergence without the singular problem:

$$s(t) = Ae(t) + e^{\alpha}(t) + B\int_0^t e(\tau)d\tau + \dot{e}^{p/q}(t) - \theta(t) \tag{14}$$

where p and q are positive odd numbers satisfying the relation 1<p/q<2 and α>p/q, $\theta(t)$ is the global sliding mode factor of the sliding mode switching function, which satisfying the conditions as follows:

$$\theta(t) = \theta(0)e^{-\delta t}, \theta(0) = Ae(0) + \dot{e}(0) \tag{15}$$

Using the hyperbolic tangent function instead of symbolic functions can weaken the jitter, we use the switching control law as:

$$\Delta\tau = \gamma_1 s + \gamma_2\tanh(s) \tag{16}$$

The time derivative of $s(t)$ is:

$$\dot{s} = A\dot{e} + \alpha|e|^{\alpha-1}\dot{e} + Be - \dot{\theta} + \frac{p}{q}|\dot{e}|^{\frac{p}{q}-1}(\ddot{q}_d - \ddot{q}) + \ddot{q}_d - \ddot{q} \tag{17}$$

let $\dot{s}(t) = 0$, we get the equivalent control as:

$$\tau_{eq} = C_0\dot{q} + G_0 + M_0\ddot{q}_d + M_0(A\dot{e} + Be - \dot{\theta}) \tag{18}$$

in equation (18), we take:$A \triangleq {}^1/_{\Delta\tau}$,$B \triangleq {}^1/_{\lambda\Delta\tau}$, $\dot{\theta} \triangleq \lambda\dot{q}_d - \frac{q_d}{\Delta\tau}$, respectively.

The global sliding mode control law can be expressed as:

$$\tau = \tau_{eq} + \Delta\tau \tag{19}$$

when system is away the system, it moves to the sliding surface under the effect of equivalent term and switch term.

According to equation (13), in order to enhance the adaptive learning capability of the system and improve the robustness, the desired trajectory information is introduced into the neural network and combined with the sliding mode control as follows:

$$\overline{W}_i^{*T}\Phi_i + \hat{\epsilon} + |(\ddot{q}_d + \dot{e}) + \lambda(\dot{q}_d + e) + q_d|\Delta\tau = M\ddot{q} + C\dot{q} + G + F + \tau_d \tag{20}$$

in equation (20), assuming that $\overline{W}$ is the new weight corresponding to the neural network, it is easy to know that $\overline{W}_i < \widehat{W}_i$ which is guaranteed to be globally bounded by equation (13).

Let $I_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, $I_2 = (1 \quad 0)$, combining with equation (10) and equation (20), we get the control law of neural network adaptive sliding mode controller:

$$u = u_{nn} + I_1 \times (\gamma_1 s + \gamma_2\tanh(s)) + |(\ddot{q}_d + \dot{e}) + \lambda(\dot{q}_d + e) + q_d| \times I_2 \times (\gamma_1 s + \gamma_2\tanh(s)) \tag{21}$$

The system control structure flow chart is shown in Figure 3.

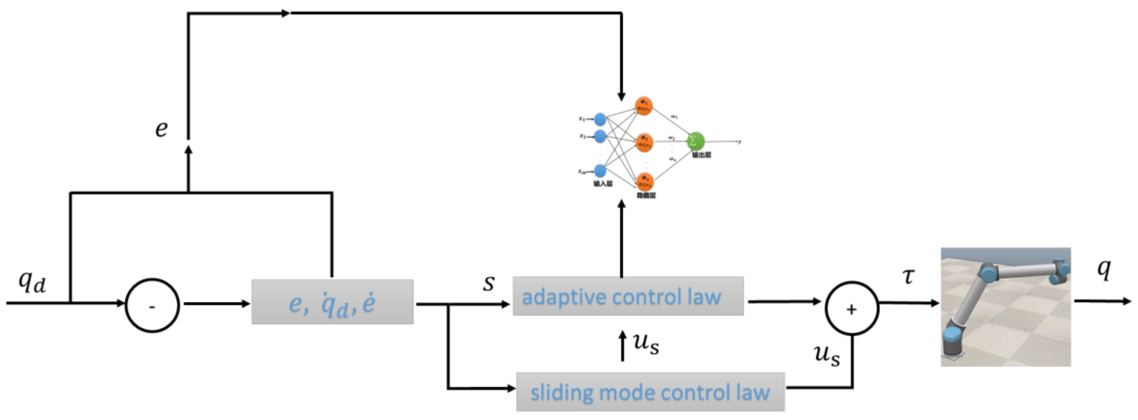


Figure 3 Block diagram of RBFNN-based adaptive scheme

2.4 Stability analysis

According to Lyapunov stability analysis, if the designed Lyapunov function is positive and its derivative is semi-negative, the control system is stable.

Theorem 1: When we consider the uncertainties in the manipulator described in equation (4), the sliding mode surface presented in equation (14), and the controller defined in equation (21), the trajectory tracking error will converge fast to zero within a finite time and singularity issue will be isolated.

Proof:

step 1: The first Lyapunov function V_0 is selected as follows:

$$V_0 = \frac{1}{2}s^T Ms + \frac{1}{2}tr(\tilde{W}^T F^{-1}\tilde{W})$$

taking the derivative of V_0 with respect to time, we have:

$$\dot{V}_0 = s^T M\dot{s} + \frac{1}{2}s^T \dot{M}s + tr(\tilde{W}^T F^{-1}\dot{\tilde{W}})$$

combining with equation (4)-(8), it can be found that:

$$\dot{V}_0 = -s^T K_v s + \frac{1}{2}s^T(\dot{M} - 2C)s + tr\tilde{W}^T\left(F^{-1}\dot{\hat{W}} + \varphi s^T\right) + s^T(\varepsilon + \tau_d + v)$$

by simplifying, we have:

$\dot{V}_0 = -s^T K_v s + s^T(\varepsilon + \tau_d + v)$

due to $s^T(\varepsilon + \tau_d + v) = s^T(\varepsilon + \tau_d) - \|s\|(\varepsilon_N + b_d) \le 0$, thus, $\dot{V}_0 \le 0$

Step 2: The second Lyapunov function can be further defined as follows:

$$V = \frac{1}{2}s^T Ms + V_0$$

then it can be formulated that:

$$\dot{V} = s^T[M(\ddot{q}_d + \Lambda e) + C(\dot{q}_d + \Lambda e) + Kq - \tau_d - \tau] + \dot{V}_0$$

let:

$M\ddot{q}_d - |\ddot{q}_d I_2|\left(\gamma_1 s + \gamma_2 \tanh(s)\right) = \Delta M\ddot{q}_d$, $C\dot{q}_d - |\dot{q}_d I_2|\left(\gamma_1 s + \gamma_2 \tanh(s)\right) = \Delta C\dot{q}_d$,
$Kq_d - |\lambda q_d I_2|\left(\gamma_1 s + \gamma_2 \tanh(s)\right) = \Delta K q_d$, $\Delta\varepsilon = I_1 + \dot{e}I_2 + eI_2$,
when $\gamma_1 > (|\Delta M|_{max}|\ddot{q}_d + \Lambda e| + |\Delta C|_{max}|\dot{q}_d + \Lambda\dot{e}| + |\Delta K|_{max}|q| + |\tau_d|_{max}) + \gamma_2 \tanh(s)$ holds, $\dot{V} \le 0$.

3 Adaptive sliding mode control for six-axis parallel platform based on RBFNN

3.1 Six-axis parallel platform kinematics analysis

In Section 2, a neural network adaptive sliding mode control method is designed for a common n-joint robotic arm, however, it is difficult to decouple the kinematics for a multi-branch, multi-joint, and strongly coupled nonlinear system, such as a parallel robot, so a kinematic analysis is needed first, and then an appropriate control law is constructed according to equation (21). Compared with the general tandem robots, six-axis parallel robots are characterized by compact complex structure, high stiffness, high load capacity, no cumulative error, high accuracy, low component wear, and long service life [13]. However, as most of parallel mechanisms are open loop and have many links and joints, calibration is crucial to ensure the motion accuracy. For instance, to improve the pose accuracy of parallel mechanisms, Zubizarreta et al. [14] applied neural networks to the forward solution of parallel mechanisms, the hypothesis is that the parameters of the mechanism are known and the calibration of the actual parameters is not concerned. Vision-based measurement is a new method which can simplify the measurement process. Dehghani, et al. [15] used a visual inspection method to calibrate a Hexa parallel robot. In previous work [16], a continuous nonsingular fast terminal sliding mode (CNFTSM) was developed for control of Stewart platforms in combination with perturbation estimation to obtain a chattering-free robust controller in the presence of time-varying external disturbances. However, many parameters need to be identified such as initial link lengths of Stewart and initial coordinates of hinge points, which increase the difficulty of calibration.

Based on the analysis on the bi-joint in Section 3, an adaptive SMC-based strategy with RBFNN is employed to solve the six-DOF. Figure 4 presents the kinematics parameters of the system. As shown in Figure 4, the six-axis parallel mechanism is mainly composed of six 3d-parametric moving platform hinges, six three-dimensional parametric fixed platform hinges, six drive lengths and the position pose of the moving platform, where j denotes the command number.

In the center plane of the hinge of the upper and lower platform, we take any point coordinate O_2 and O_1 as the origin, the axis Z_2 and Z_1 are perpendicular to the upper and lower platform respectively to establish the moving coordinate $O_2X_2Y_2Z_2$ system and the fixed coordinate system connected with the upper and lower platform. Then we find the rotation transformation matrix and the translation vector from moving coordinate system $O_2X_2Y_2Z_2$ to the fixed coordinate system $O_1X_1Y_1Z_1$ to determine the spatial position.

Suppose the coordinates A_i in the fixed coordinate system $O_1X_1Y_1Z_1$ is $(a_{xi}, a_{yi}, 0)$, and the coordinates of the origin O_2 in the fixed coordinate system $O_1X_1Y_1Z_1$ denotes $P(x, y, z)$, the coordinates B_i in the moving coordinate system denotes $(b_{xi}, b_{yi}, 0)$, the length A_iB_i denotes L_i, and the rotation matrix of the moving coordinate system $O_2X_2Y_2Z_2$ to the fixed coordinate system $O_1X_1Y_1Z_1$ denotes $R = \begin{pmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{pmatrix}$ From the constraint of rod length, we have:

$$L_i^2 = (RQ_{ib} + p - Q_{ia})^T (RQ_{ib} + P - Q_{ia}) \tag{22}$$

$i = 1, 2, \cdots, 6$ substitute the coordinates into equation (22), we have:

$$(b_{xi}r_1 + b_{yi}r_4 + x - a_{xi})^2 + (b_{xi}r_2 + b_{yi}r_5 + y - a_{yi})^2 + (b_{xi}r_3 + b_{yi}r_6 + z)^2 - L_i^2 = 0 \tag{23}$$

The following system of equations is obtained based on the property that the rotation transformation matrix R is a unitary orthogonal array:

$$\begin{cases} r_1^2 + r_2^2 + r_3^2 - 1 = 0 \\ r_4^2 + r_5^2 + r_6^2 - 1 = 0 \\ r_1r_2 + r_2r_5 + r_3r_6 = 0 \\ r_4r_8 - r_5r_7 - r_3 = 0 \\ r_2r_7 - r_1r_8 - r_6 = 0 \\ r_1r_5 - r_2r_4 - r_9 = 0 \end{cases} \tag{24}$$

From the above analysis, the corresponding relationship between the six servo motor angles and the six degrees of freedom of the moving stage are obtained, so that the input position can be derived from the given output position.

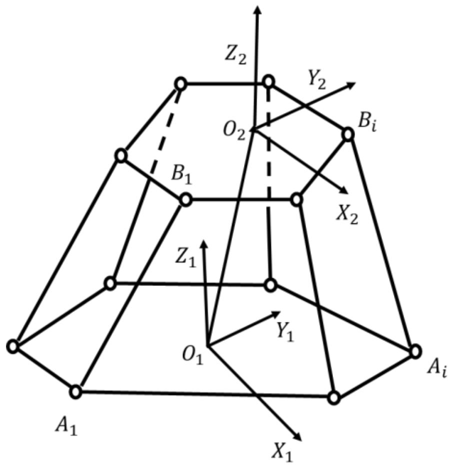


Figure 4 Simplified model of six-axis Platform

3.2 Design of the proposed controller for 6-dof parallel platform

Since each of the six active arms is linked to the moving platform by a rod, all the linked rods can achieve independent motion, but there is a strong coupling between the motion outputs of each degree of freedom of the system, which needs to be decoupled to obtain good performance, and the sliding mode control has a "self-decoupling" function, combining RBFNN with the sliding mode controller, the upper platform attitude is decoupled by inverse solution to make it a second-order nonlinear system as follows [17]:

$$\ddot{x} = f(x,t) + g(x,t)u + d(t) \tag{25}$$

where x denotes the state variable, f, g denote the unknown nonlinear functions of the system respectively, u denotes the control quantity, and $|d(t)| \le D$ is the bounded disturbance.

The desired trajectory x_d of each branch is given by solving the system of equation (24).

We take the sliding surface as follows:

$$s = \lambda e + \dot{e} \tag{26}$$

If $g(x,t) \ne 0$,we choose the equivalent sliding mode control theory is as [18]:

$$\bar{u} = \frac{v - f(x,t)}{g(x,t)} \tag{27}$$

where $v = \ddot{x}_d - \lambda\dot{e} - \eta\,\mathrm{sgn}(s), \eta > D$

Define Lyapunov function as:

$$L = \frac{1}{2}s^2 \tag{28}$$

then it can be formulated that:

$$\dot{L} = s\dot{s} = s(\ddot{x} - \ddot{x}_d + \lambda\dot{e}) = s(f + gu + d - \ddot{x}_d + \lambda\dot{e}) \tag{29}$$

by simplifying, we have:

$$\dot{L} = s(\ddot{x}_d - \lambda\dot{e} - \eta sgn(s) + d - \ddot{x}_d + \lambda\dot{e}) = s(-\eta sgn(s) + d) = -\eta|s| + ds \le 0$$

according to equation (20), the sliding mode control law of six-axis parallel platform is taken as follows:

$$\tau = u + \bar{u} \tag{30}$$

The control algorithm satisfies Lyapunov stability.

3. 3 RBF neural network learning

Dataset acquisition: Assuming that the geometrical characteristics of the robot are known, the kinematic analysis of the robot arm is based on the kinematic analysis of its linkage coordinate system to obtain the D-H parameters, and the end-effector position and attitude are obtained by solving the kinematic equation, and the joint angle of each linkage and Cartesian coordinates of the end-effector are sampled to create a data set containing about 500 randomly generated entries. If the inverse mapping of end-effector positions to joint angle vectors has only a unique solution, the inverse kinematic model is considered to be fit; if the mapping has multiple solutions, i.e., there are multiple sets of angle vectors for the desired end-effector positions, or no solutions, the inverse kinematic model is considered to be not fit [19]. In order to solve these problems that can lead to inefficient training of the model, the dataset was preprocessed to remove invalid data samples, resulting in a trainable dataset of about 200 different entries, and then the dataset was divided into a training set and a test set in the ratio of 70% and 30%.

Training of the dataset: With the help of the neural network toolbox in MATLAB, the improved RBF neural network was trained using the mean square error (MSE) as the radial basis neural network loss function.

3.4 Framework

Our framework mainly contains three parts: trajectory planning, system control and trajectory optimization, which is shown in Figure 5.

(1) Trajectory planning.

Firstly, the host computer will send the pose commands for the key points of the robot operator. The kinematic model is established based on the D-H method, where the positive kinematics is used to find the x, y and z values of the end effector and the inverse kinematics is used to find the joint angle values. The ideal trajectory is obtained by using the fifth polynomial interpolation method [22,23].

(2) System control stage.

In this stage, the dynamics model is established based on Lagrange's theorem. the torque of each joint under the ideal trajectory is solved by the dynamic method, and then the torque information is transmitted to the attitude controller. The posture controller drives the control motor rotation to realize the movement.

(3) Trajectory optimization.

The designed RBF neural network adaptive sliding mode controller is used to optimize the tracking trajectory, and the new control commands are transmitted to the attitude controller, and the trajectory that achieves the desired goal is used as the actual trajectory of the robot arm [24,25].

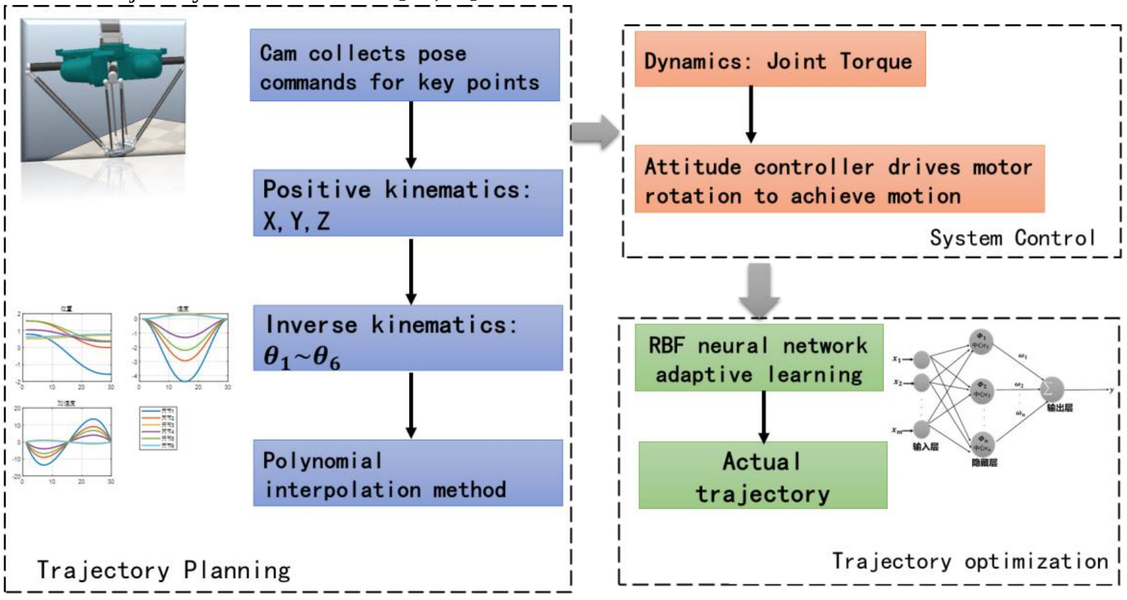

Cam collects pose commands for key points
Positive kinematics: X, Y, Z
Inverse kinematics: $\theta_1 \sim \theta_6$
Polynomial interpolation method
Trajectory Planning
Dynamics: Joint Torque
Attitude controller drives motor rotation to achieve motion
System Control
RBF neural network adaptive learning
Actual trajectory
输入层
隐藏层
输出层
Trajectory optimization

Figure 5 Trajectory planning framework

4 Experimental Results

4.1 two-link manipulator trajectory tracking

In order to investigate the effect of the improved algorithm on the number of initial hidden nodes and parameters of the neural network on the convergence speed of the system, the control law proposed in equation (21) is used to verify the effectiveness of the proposed control method by using a double-joint robot arm as the object of simulation analysis, the matrix of each coefficient in equation (1) is:

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

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

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

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

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

The parameter c is taken strictly as a range of network input values and the parameters of the system are listed in Table 1.

The initial weights of the RBF neural networks are 0, the desired trajectories of the manipulator are set as $q_{1d} = 0.1\sin(t)$, $q_{2d} = 0.1\sin(t)$.

Table 1 Controller parameter

parameter	**value**
$\boldsymbol{p}$	$[2.9, 0.76, 0.87, 3.04, 0.87]$
$\boldsymbol{x}$	$[0.9, 0, -0.9, 0]$
$\boldsymbol{K_v}$	$diag\{20,20\}$
$\boldsymbol{\Lambda}$	$diag\{5,5\}$
$\boldsymbol{F}$	$diag\{15,15\}$
$\boldsymbol{\varepsilon_N}$	0.20
$\boldsymbol{b_d}$	0.10
$\boldsymbol{\gamma_1}$	50
$\boldsymbol{\gamma_2}$	0.1
$\boldsymbol{\lambda}$	5.0

For comparison, we simulate two sets of parameters.

set 1: the initial hidden layer width $b_j = 0.5$, the initial hidden layer center c:

$$c = 0.1\begin{bmatrix} -1.5 & -1 & -0.5 & 0 & 0.5 & 1 & 1.5 \\ -1.5 & -1 & -0.5 & 0 & 0.5 & 1 & 1.5 \\ -1.5 & -1 & -0.5 & 0 & 0.5 & 1 & 1.5 \\ -1.5 & -1 & -0.5 & 0 & 0.5 & 1 & 1.5 \\ -1.5 & -1 & -0.5 & 0 & 0.5 & 1 & 1.5 \end{bmatrix}$$

The simulation results of set 1 are shown in Figure.6-Figure.10 and Table 2.

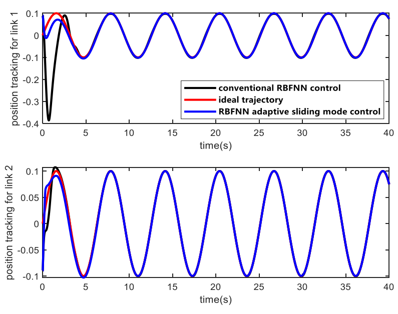


Figure 6 Position tracking trajectories

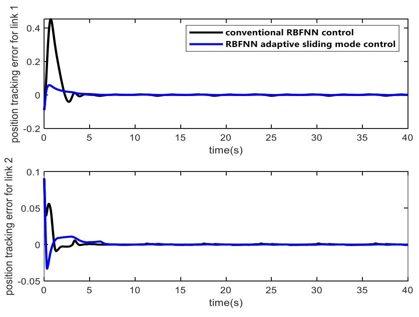


Figure 7 Position-tracking error

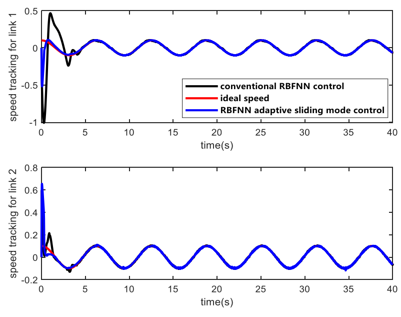


Figure 8 Angular velocity

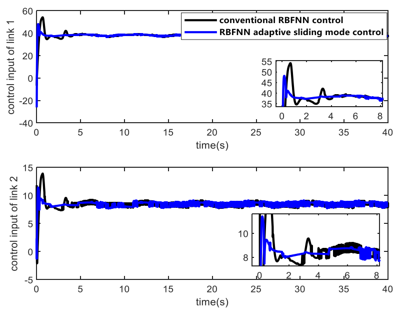


Figure 9 control inputs

Table 2 Result of node=7, b=0.5

		Maximum errors	Steady-state errors	Convergence time
Traditional RBFNN	1	0.450	1.9×10^{-4}	5.8s
	2	0.09008	-4.225×10^{-4}	5.8s
adaptive RBF-SMC	1	0.060	-3.077×10^{-4}	2.0s
	2	0.09002	-1.781×10^{-4}	2.0s

set 2: To verify the robustness of the proposed control algorithm to changes of RBF neural network, the number of hidden layers are increased to 15, $b_j = 10$.

The results of set 2 are shown in Figure10-Figure 13 and Table 3.

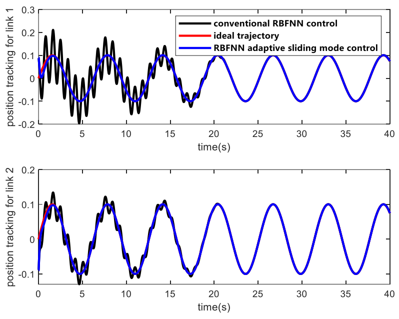


Figure 10 Position tracking

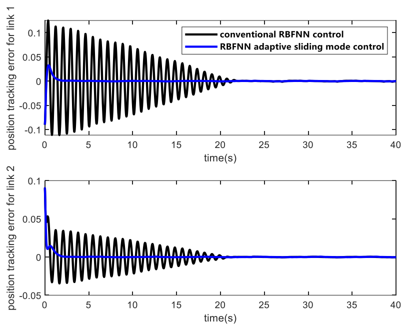


Figure 11 Position tracking error

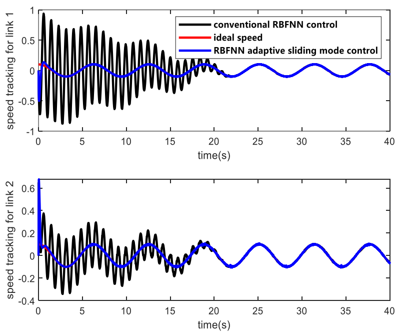


Figure 12 Angular velocity

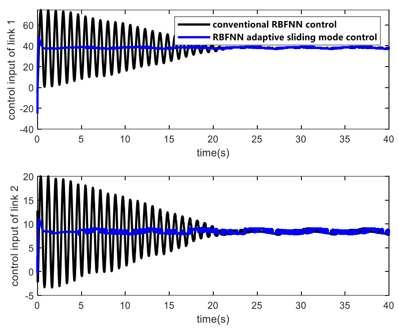


Figure 13 Control inputs

Table 3 Result of node=15, b=10

		Maximum errors	Steady-state errors	Convergence time
Traditional RBFNN	1	0.125	-1.194×10^{-4}	21.6s
	2	0.09007	-2.595×10^{-4}	21.6s
adaptive RBF-SMC	1	0.034	-1.734×10^{-4}	2.0s
	2	0.09002	3.1×10^{-4}	2.0s

The trajectory position tracking and its errors are shown in Figure6, Figure 7, Figure10, and Figure 11. According to Table 2, the tracking steady-state errors of two algorithms reach the accuracy requirement of 10^{-4} level, but the maximum error is reach up to 0.450 under the traditional RBFNN control, while the maximum error is less than 0.1 after adding the adaptive sliding mode controller. The system convergence is accelerated by 65.52%, which clearly indicates that the improved neural network adaptive sliding mode controller has better adjustment ability under the condition of large initial position error of the robot arm.

As shown in Table 3, when we increase the number of nodes in the hidden layer, the system still starts to track the reference model at about 1s, and it tracks the reference model well at about 2s. The maximum error is 0.125 before improvement, while the maximum error is less than 0.1 under the proposed RBFNN-based adaptive SMC strategy, which indicates that the neural network adaptive control algorithm is insensitive to the system parameter changes and has good robustness. Our method can avoid the influence of parameter settings on the system to a certain extent, and it can realize the end-of-execution which can achieve high precision trajectory tracking at the end of execution.

From Figure 9 and Figure 13, it can be seen that the improved control torque output is smoother and more de-jittering.

4.2 Six-axis parallel platform trajectory tracking

To further verify the effectiveness of the trajectory tracking optimization method proposed in equation (30), the trajectory tracking of the 6-Dof parallel platform Adept Quattro650HS was carried out using a joint simulation of v-rep platform and MATLAB. This experiment requires the robot moves from (14.27mm, -18.68mm, -86.65mm) to (13.46mm, -106.6mm, -99.05mm) within 30s. Some of the data shown in Table 4 were obtained according to the kinematic equation (24) in Section 3. The geometric parameters of the parallel platform are given in Table 5. Since the time variation of $f(x,t)$ and $\hat{g}(x,t)$ in equation (25) are unknown, the output of RBFNN are used to approximate, the trajectory tracking performed by the conventional RBF neural network for the robotic arm end-effector is compared with the adaptive radial-based neural network discriminative control law proposed in Section 3.

Table 4 Example of partial training set data

J1	**J2**	**J3**	**J4**	**J5**	**J6**	**x**	**y**	**z**
1.653	0.561	0	1.850	0.565	0.714	0.135	-1.052	-0.981
0.175	1.124	0.339	0.105	1.143	0.698	0.141	-0.526	-0.887
1.858	0.509	0.798	1.558	0	0.655	0.136	-0.806	-0.905
0.652	5.587	3.704	0.033	4.112	1.704	0.137	-0.753	-0.891
2.281	0.486	4.143	0	1.193	0	0.139	-1.019	-0.953

Table 5 The geometric parameters

platform parameters	**values**
Radius of the outer circle of the moving platform	0.20m
Moving platform short side distance	0.10m
Radius of external circle of static platform	0.40m
Static platform short side distance	0.20m
Upper linkage length	0.30m
Lower linkage length	0.50m
Moving platform quality	30kg
Upper linkage mass	5kg
Lower linkage mass	10kg
Moving platform inertia matrix /(kg•m^2)	$diag\{1, 1, 1.5\}$
Upper linkage inertia matrix /(kg•m^2)	$diag\{0.5, 0.5, 0.1\}$
Lower linkage inertia matrix/(kg•m^2)	$diag\{0.8, 0.8, 0.2\}$

Since each axis of the parallel robot has the same structure, a decentralized control strategy is used for trajectory tracking on a single axis, and the mathematical model of the single channel can be expressed as a transfer function:

$$G(s) = \frac{K_t \times L}{s(Js + B)}$$

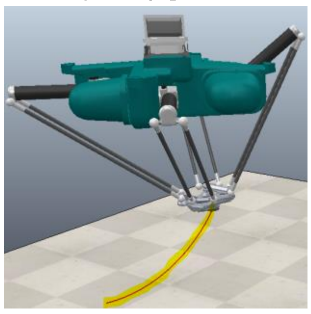
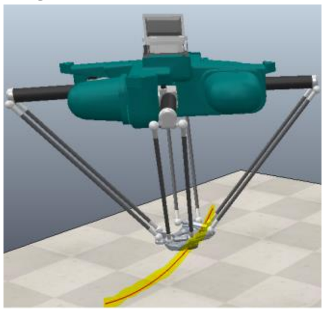
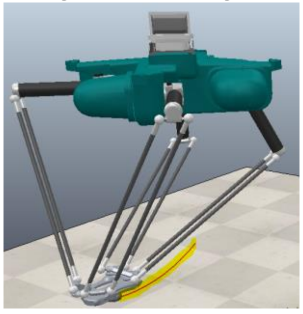
where L denotes the lead of the screw, $J = J_1 + J_2$ is the AC servo motor torque constant, J_1 is the motor rotational inertia, J_2 is the screw rotational inertia, the specific values are shown in Table 6.

Taking $f(x,t) = -1.0\times10^3 x_1 - 5.0\times10^1 x_2$, $g(x,t) = 2$, and the simulation experiments do not consider the effect of friction on the system. The number of nodes in hidden layer of the RBFNN is 5, with Gaussian basis function parameters: b=3.0, $c = 0.5\begin{bmatrix} -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \end{bmatrix}$

Table 6 Controller parameters

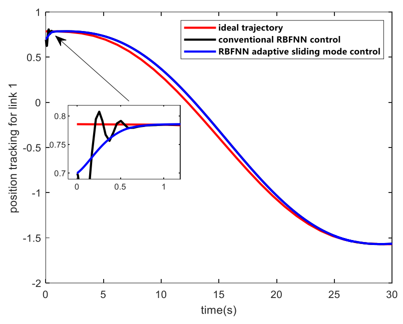
Parameter	**value**
L	0.20m
K_t	0.90m
J_1	6.0×10^{-4}kg/m2
J_2	9.0×10^{-4}kg/m2
B	5.0×10^{-3}N·m·s/rad
K_v	$diag\{10, 10\}$
λ	5.0

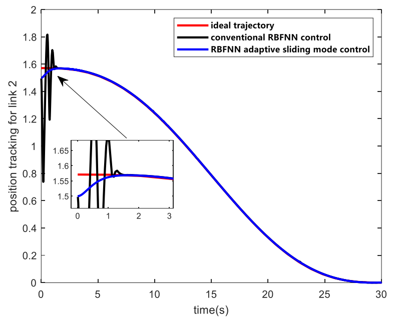
The trajectory position tracking and its errors are shown in Figure 14 and Figure 15.

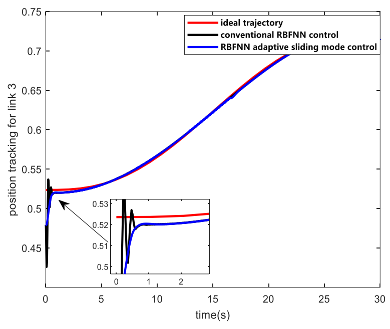




(a) (b) (c)

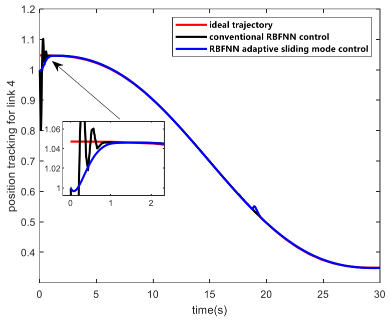
Figure 14 motion process ((a)Starting position; (b) Tracking Process; (c)arriving endpoint)

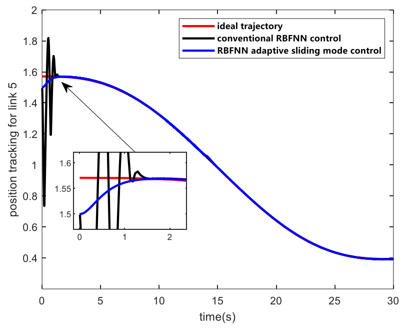


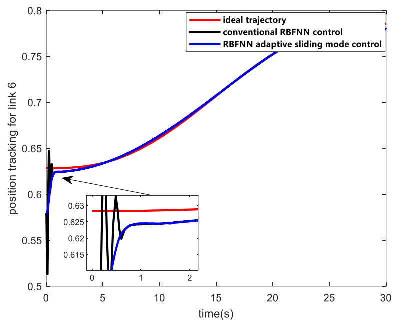




(a) (b) (c)







(d) (e) (f)

Figure 15 Trajectory tracking for each joint ((a)-(f) link 1-link6)

From the six branch trajectory tracking in figure 15, it can be seen that the two algorithms above can achieve infinitesimal steady-state tracking error, but the whole tracking process of the robot arm under the RBF neural network sliding mode adaptive control is smoother, the velocity curve is smooth without abrupt changes and inflection points, and the motion of the robot arm end-effector is smoother, which improves the response time of the control system and further solves the problems faced by the robot arm in practical applications, such as severe hand jitter during start and stop and the lack of accuracy of the robot arm end reaching the specified position.

5 Conclusion

In this paper, an adaptive control method has adopted by combining the RBF neural network and sliding mode control for the two-link manipulator and six-axis parallel mechanical platform with uncertainty, respectively. the RBFNN is used to approximate the equivalent part of the sliding mode control, which greatly reduces the influence of the initial number of hidden nodes and parameters of the RBFNN on the convergence speed of the system. The RBFNN is combined with the sliding mode control to overcome the shortcomings such as the dependency of sliding mode control on system model, and a lack of methods of systematically determining the system control parameters, and the model obtained by using this algorithm is insensitive to the initial parameter conditions, which approximates the real system well and ensures the stability and robustness of the system under the conditions of disturbances, quality changes, and model errors. Additionally, the control laws is derived by the Lyapunov approach to guarantee the globally stability. At last, based on the analysis on the dynamic model, the simulation is performed by MATLAB. Its results are compared with conventional RBFNN controller to exhibit the efficiency of the proposed approach with the uncertainties, and the robustness. The algorithm is also effective when applied to other nonlinear systems. There are two main areas of future work:

(1) To study and construct S-functions for complex control laws to facilitate debugging of the system and reduce the simulation effort.
(2) To combine PID control with fuzzy control and RBF neural network to realize the adjustment of RBF neural network by using the good convergence of fuzzy control and the advantage of operation on fuzzy quantities and the self-learning and self-adaptive property of neural network.

References

[1]. GANG Mingyi, CHEN Lixin. (2021) ‘Trajectory tracking control of manipulator based on RBF neural network inverse system’, *Journal of Anhui Science and Technology University*, Vol.35, pp.68-74. (in Chinese).

[2]. BAEK J, JIN M, HAN S. (2016) ’A new adaptive sliding-mode control scheme for application to robot manipulators’, *IEEE Transactions on Industrial Electronics*, Vol.63, pp.3628-3637.

[3]. Han H. (2018) ‘An adaptive-PSO-based self-organizing RBF neural network’, *IEEE Trans Neural Netw Learn Syst*, Vol.29, pp.104-117.

[4]. Li, M., Li, Y., Ge, S. S., & Lee, T. H. (2017). ‘Adaptive control of robotic manipulators with unified motion constraints’, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Vol.47, pp.184–194.

[5]. H. Sun, L. Hou, G. Zong, and X. Yu. (2020) ‘Adaptive decentralized neural network tracking control for uncertain interconnected nonlinear systems with input quantization and time delay’, *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 4, pp. 1401–1409.

[6]. F. Ruggiero, J. Cacace, H. Sadeghian, V. Lippiello. (2014) ‘Impedance control of VToL. UAVs with a momentum-based external generalized forces estimator’, *in: 2014 IEEE International Conference on Robotics and Automation (ICRA), IEEE*, pp. 2093.

[7]. ATTARAN S M, YUSOF R, SELAMAT H. (2016) ‘A novel optimization algorithm based on epsilon constraint-RBF neural network for tuning PID controller in recoupled HVAC System’, *Applied Thermal Engineering*, Vol.99, pp.613-624.

[8]. GANG Mingy. (2021) ‘Trajectory tracking control of manipulator based on RBF neural network inverse system’, *Journal of Anhui Science and Technology University*, Vol.35, pp.68-74.

[9]. Cheng S, Fan J, Dey A. (2018) ‘Smooth gaze: A framework for recovering tasks across devices using eye tracking’, *Personal and Ubiquitous Computing*, Vol.22, pp.489–501.

[10]. Nojavanzadeh, D. Badamchizadeh, M. (2016) ’Adaptive fractional-order non-singular fast terminal sliding mode control for robot manipulators’, *IET Control Theory & Applications*, Vol.10, pp.1565–1572.

[11]. W. Qi, G. Zong, and H. R. Karimi. (2020) ‘Sliding mode control for nonlinear stochastic singular semi-Markov jump systems’, *IEEE Trans. Autom. Control*, vol. 65, no. 1, pp. 361–368.

[12]. Y. Li and Q. Xu. (2010) ‘Adaptive sliding mode control with perturbation estimation and PID sliding surface for motion tracking of a Piezo-driven micromanipulator’, *IEEE Trans. Control Syst. Technol.*, vol. 18, no. 4, pp. 798–810.

[13]. Ahmad, I.; Liaquat, M.; Malik, F.M.; Ullah, H.; Ali, U. (2020) ‘Variants of the Sliding Mode Control in Presence of External Disturbance for Quadrotor’, *IEEE Access*, Vol.8, pp.227810–227824.

[14]. Zhu, H. and Ni, Q. (2018), “A simple alternating direction method for the conic trust region subproblem”, *Mathematical Problems in Engineering*, Vol. 2018.

[15]. Dehghani, M., Eghtesad, M., Ahmadi, M., Khayatian, A. and Yazdi, M. (2014), ‘Vision-based calibration of a hexa parallel robot’, *Industrial Robot: An International Journal*, Vol. 41 No. 3, pp. 296-310.

[16]. S. Kang, H. Wu, X. Yang, Y. Li, and Y. Wang. (2020) ‘Model-free robust finite time force tracking control for piezoelectric actuators using time-delay estimation with adaptive fuzzy compensator’, *Trans. Inst. Meas. Control*, vol. 42, pp. 351–364.

[17]. Andrievskiy, B.R.; Arseniev, D.G.; Zegzhda, S.A.; Kazunin, D.V.; Kuznetsov, N.V.; Leonov, G.A.; Tovstik, P.E.; Tovstik, T.P.; Yushkov, M.P. (2017) ‘Dynamics of a Stewart platform’, *Vestn.St. Petersburg Univ. Math.*, Vol.50, pp.297–309.

[18]. Busjahn T, Bednarik R, Begel A, Crosby M, Paterson J H, Schulte C, Sharif B, Tamm S. (2015) ‘Eye movements in code reading: Relaxing the linear order’, *In: Proceedings of the 2015 IEEE 23rd International Conference on Program Comprehension*, pp:255−265.

[19]. Y. Wu and X.-J. Xie. (2021) ‘Robust adaptive control for state-constrained nonlinear systems with input saturation and unknown control direction’, *IEEE Trans. Syst., Man, Cybern., Syst*, Vol. 51, pp.1192–1202.

[20]. S. A. Khader, H. Yin, P. Falco. (2020) ’Stability-guaranteed reinforcement learning for contact-rich manipulation’, *IEEE Robotics and Automation Letters*, Vol. 6, pp.1–8.

[21]. X. Yang et al. (2019) ‘Dynamic modelling and decoupled control of a flexible Stewart platform for vibration isolation’, *J. Sound Vib*, Vol.439, pp.398–412.

[22]. LIU Zhigao. (2018) ‘Design of a Modified Tracking Differentiator’, *Navigation and Control*, Vol.17, pp.61-65.

[23]. R. Mart´ın-Mart´ın, M. A. Lee, R. Gardner, S. Savarese, J. Bohg, and A. Garg. (2019) ‘Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks’, *in 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp.1010–1017.

[24]. HOU T, GUO Y Y, NIU H X. (2019) ‘Research on speed control of high-speed train based on multi-point model’, *Archives of Transport*, Vol. 50, pp.35-46.

[25]. HAN Jiang, WANG Pen. (2021) ‘Robust Servo Constrained Control of Parallel Robots Based on the Udwadia-Kalaba Metho’, *Applied Mathematics and Mechanics*, Vol. 42, pp.264-272.