

Article

High-Order Sliding Mode Control for Three-Joint Rigid Manipulators Based on an Improved Particle Swarm Optimization Neural Network

Jin Zhang ^{1,2}, Wenjun Meng ^{1,3,*} , Yufeng Yin ¹, Zhengnan Li ^{1,4}, Lidong Ma ^{1,4} and Weiqiang Liang ^{1,2}¹ School of Mechanical Engineering, Taiyuan University of Science and Technology, Taiyuan 030024, China² School of Intelligent Manufacturing, Shanxi Vocational University of Engineering Science and Technology, Jinzhong 030619, China³ Shanxi Institute of Energy, Taiyuan 030600, China⁴ Shanxi Provincial Engineering Research Center for Intelligent Heavy Load Equipment & Robot System, Taiyuan 030024, China

* Correspondence: b20180013@stu.tyust.edu.cn

Abstract: This paper presents a control method for the problem of trajectory jitter and poor tracking performance of the end of a three-joint rigid manipulator. The control is based on a high-order particle swarm optimization algorithm with an improved sliding mode control neural network. Although the sliding mode variable structure control has a certain degree of robustness, because of its own switching characteristics, chattering can occur in the later stage of the trajectory tracking of the manipulator end. Hence, on the basis of the high-order sliding mode control, the homogeneous continuous control law and super-twisting adaptive algorithm were added to further improve the robustness of the system. The radial basis function neural network was used to compensate the errors in the modeling process, and an adaptive law was designed to update the weights of the middle layer of the neural network. Furthermore, an improved particle swarm optimization algorithm was established and applied to optimize the parameters of the neural network, which improved the trajectory tracking of the manipulator end. Finally, MATLAB simulation results indicated the validity and superiority of the proposed control method compared with other sliding mode control algorithms.

Keywords: neural network; particle swarm optimization algorithm; sliding mode control; super-twisting adaptive algorithm; manipulators control

MSC: 93d20; 68t073

Citation: Zhang, J.; Meng, W.; Yin, Y.; Li, Z.; Ma, L.; Liang, W. High-Order Sliding Mode Control for Three-Joint Rigid Manipulators Based on an Improved Particle Swarm

Optimization Neural Network.

Mathematics **2022**, *10*, 3418. <https://doi.org/10.3390/math10193418>

Academic Editors: Nikolaos Samaras and Asier Ibeas

Received: 21 July 2022

Accepted: 13 September 2022

Published: 20 September 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

With the current development of technology, the application of robots has reached an unprecedented range, including aerospace, medical treatment, and welding. Robot application can not only reduce the labor intensity of workers but also improve the production efficiency. In addition, robots can be applied to tasks that need to be processed in inaccessible areas for humans. Therefore, manipulators play a crucial role in modern industry. Manipulators are based on the functions of human hands and arms and can be used for grasping, moving, and other complex operations similar to those carried out by human arms. Originally, a manipulator is operated by sequence control. With the optimization and development of computer intelligence, the control methods of manipulators as well as their application fields have diversified. This increment of robots also comes with a tremendous challenge, since each robot has a different structure and therefore each one requires a different algorithm to achieve its tasks [1]. The process of robot research mainly involves two aspects: trajectory planning and controller design. Complete trajectory planning includes path planning, inverse kinematics solution, and trajectory optimization [2].

For the trajectory tracking of robot manipulators, Carlos Lopez-Franco proposed a meta-heuristic optimization approach to minimize the end-effector position and orientation in 3D [1]. Xiao et al. proposed an improved method of disconnection and reconnection to simplify the solution process of inverse kinematics. The solution of joint variables in the process of solving inverse kinematics is not affected by singularity [3]. Li et al. proposed a compensation method based on command posture to overcome the geometric parameter error compensation of the manipulator; this method cannot be effectively applied directly to the inverse kinematics of the nominal model of the manipulator, although the verification results show that this method takes less time than the Newton–Raphson algorithm [4]. In actual production, however, the control system of a manipulator can be interfered by a number of uncertainty factors, which decreases the positioning accuracy of the end effector of the manipulator. Hence, further research on the control method of a robot has practical significance [5]. An ideal control strategy considerably affects the output of a complex non-linear system as well as the tracking of joints and ends. Researchers have been investigating perfect control strategies, such as adaptive control [6], sliding mode control (SMC) [7], and proportional–integral–derivative (PID) control [8], to achieve a faster and more efficient joint tracking. However, a single control strategy cannot meet all the requirements of trajectory tracking. Thus, multi-control strategies, for example, the sliding mode neural network control (SMNNC) [9], the fuzzy sliding mode neural network (FSMNNC) [10], the PID SMC [11], and the sliding mode neural network adaptive control (SMNNAC) [12], are combined so that they complement each other.

Owing to the chattering of SMC itself, the end trajectory cannot be tracked well. Therefore, research on the improvement of the chattering performance of SMC has gained attention. Kali et al. proposed a super-twisting algorithm (STA) with time delay estimation for realizing the high-precision trajectory tracking of robots in the presence of uncertainty and unexpected interference. The control law was designed on the basis of the STA to ensure robustness, finite time convergence, and chattering reduction [13]. For the robust tracking and modelling of the problems of uncertain time-delay systems, Pai proposed an NNSMC scheme with a dynamic output feedback radial basis function (RBF). This algorithm was based on dynamic output feedback SMC, the RBF neural network, and adaptive control. It solved the design of the sliding surface and the existence of the sliding mode. The validity of the scheme was also verified [14]. Zhang et al. designed an adaptive robust control strategy for the problems of low convergence speed and large positioning error of the conventional PID control in the trajectory tracking control of the manipulator. Compared with the conventional PID control, this strategy can more accurately control the trajectory of the manipulator end [15]. Scholars such as Liu et al. and Liao et al. used the RBF neural network to approximate the uncertainty in the modeling process of multi-joint manipulators and employed the adaptive control law to estimate the weights of the middle layer of the neural network. This reduced the tracking error of the end trajectory of the manipulator joint and improved the vibration phenomenon [16,17]. Liu et al. designed a self-tuning PSO fuzzy PID positioning controller to improve the robustness and accuracy of the controller. The simulation results showed that the transient response speed, tracking accuracy, and tracking system characteristics were significantly improved [18]. Liang et al. and Yang et al. designed an interference observer to observe and compensate the external uncertain interference for the multi-joint manipulator in the presence of external uncertain interference. Simulation verification demonstrated that the effect of the interference observer was significant, ensuring the control performance of the system [19,20].

The stable adaptive control for nonlinear systems using neural networks has received more and more attention in recent years. Sun et al. proposed a stable neural network-based adaptive controller design for integrating a neural network (NN) approach with an adaptive implementation of the variable structure control [21]. Sun proposed a stable adaptive controller of sampled-data nonlinear systems using neural networks by applying the approach to the stable neuro-adaptive control of rigid and flexible link robotic manipulators and

obtained a set of stable sampled-data adaptive control approaches of robotic manipulators using neural networks [22].

When the RBF neural network is applied to approximate modeling errors and uncertain terms, the RBF often has complex surfaces with multiple poles owing to the loss function. The use of PSO to update the weights results in the model falling into the local optimal value easily. The following studies have improved the PSO algorithm. The well-established particle swarm optimization (PSO) algorithm was proposed to avoid the problem of early convergence that highly affects the original implementation of this algorithm [23]. The improved particle swarm optimization algorithm was derived from an evolutionary algorithm. The evolutionary computation algorithms are based on biological Darwinian evolution to design a solution and include ecology-based algorithms and bio-inspired algorithms. Ecology-based algorithms are based on the ecosystems to solve the problem. This group includes the biogeography-based optimization (BBO) and invasive weed colony optimization (IWO) algorithms. The bio-inspired algorithms include the particle swarm optimization algorithm (PSO), artificial bee colony algorithm (ABC), fish swarm algorithm (FSA), firefly algorithm (FA), bacterial foraging algorithm (BF), ant colony optimization algorithm (ACO) and others [24]. Evolutionary algorithms are also widely used. For example, Victor Parque et al. proposed using differential evolution (DE) to optimize the smooth curve fitting of mobile robot trajectories [25]. Alessandro Niccolao et al. proposed the optimization of electric vehicles charging station deployment by means of evolutionary algorithms [23]. Liu et al. proposed an improved PSO algorithm to optimize the back propagation neural network (BPNN). The PSO algorithm used an improved adaptive acceleration factor and adaptive inertia weight to improve the initial weights and thresholds of the BPNN. The simulation results showed that the new algorithm can improve the convergence speed and prediction accuracy of the BPNN and reduce the prediction errors [26]. Wang et al. proposed a scheme in which the PSO algorithm was applied to optimize artificial neural networks (ANNs) and could effectively overcome the shortcomings of traditional neural networks with poor dynamic adaptability. That is, in traditional neural networks, it is usually necessary to perform iteration on the training samples for the process of weight convergence [27]. Zhang et al. proposed an improved PSO algorithm to optimize the PID neural network model. This new model was improved on the basis of the inertia weight and fitness function of the traditional PSO algorithm. The new model used the PSO algorithm to optimize the initial weight of the PID neural network, thereby shortening the study time for the optimal value of the particle swarm and reducing the chance of local minimum. The results indicated that this method has good self-learning ability, good optimization quality, high control accuracy, no overshoot, fast response, and short setting time [28]. For complex systems with high nonlinearity and strong coupling, Ye et al. used the decoupling control technology based on the PID neural network to eliminate the coupling between loops. However, the connection weights of the PID neural network can easily fall into local optimum. Hence, a hybrid PSO and differential evolution algorithm was proposed to optimize the connection weights of the PID neural network [29]. Yu et al. proposed a new evolutionary ANN algorithm. The improved PSO algorithm used methods including parameter automation strategies, speed resetting, crossover, and mutation, which significantly improved the performance of solution fine-tuning and the original PSO algorithm in global search. The improved PSO was employed to simultaneously evolve the structure and weight of the ANN through a specific individual representation and evolution scheme [30].

The aim of this paper was to design the controller for a three-joint manipulator; the modeling uncertainty in the system and external uncertainty interference were ignored. Even if the controller was designed by considering the various uncertainties in the system, its manipulator control performance was not sufficient, resulting in poor manipulator end trajectory tracking performance. In addition, there was obvious chattering in the later stage of the trajectory tracking of the manipulator end. To improve the manipulator's operating performance and adaptability to the environment, Yun et al. proposed an automatic dynamic bit hybrid control method based on bit control and introduced an improved position controller

based on the traditional fuzzy PID control strategy [31]. Therefore, in this study, a high-order SMC (HOSMC) based on an improved PSO neural network is proposed.

We used the Lagrange energy method to model the dynamics of the three-joint rigid manipulator. Subsequently, an HOSMC was designed for the dynamic model of the system, and a homogeneous continuous control law and super-twisting algorithm were added. In addition, an adaptive law was designed for the gain function of the super-twisting algorithm to enhance the robustness of the system. In view of factors such as the modeling uncertainty in the system, the RBF neural network was used to compensate and approximate the uncertainties. Moreover, the weights of the middle layer of the neural network were updated via the improved PSO algorithm. Finally, simulation experiments showed the superiority of the control method proposed in this paper.

The main contributions of this paper are as follows: (1) To address the problem of severe chattering of the system when approaching a sliding mode surface, the homogeneous continuous control term and the hyper spiral algorithm were added to the SMC, and the adaptive law was designed for the gain of the hyper spiral algorithm. (2) To deal with the model error and external uncertain interference in the process of system modeling, a neural network was used for approximation compensation. (3) To optimize the weights of the middle layer of the neural network, an improved PSO algorithm is used. (4) Tests were conducted on the improved PSO algorithm, and the results indicated that the proposed method has certain advantages in the tracking response speed and tracking error of a three-joint rigid manipulator.

The organizational structure of this paper is as follows. Section 2 presents details of related work and establishes a mathematical model of a three-joint manipulator. In Section 3, the SMC is designed, and in Section 4, high-order sliding mode control neural network adaptation is introduced. Section 5 presents the Proof of stability, and Section 6 presents the improved PSO algorithm to further optimize neural network parameters. In Section 7, simulation experiments are conducted on the algorithm. Finally, the conclusion is provided in Section 8.

2. Dynamic Modeling of a Three-Joint Rigid Manipulator

This paper considered a serial robot as an example, which satisfied the Pieper criterion of industrial robots. The first three joints of the manipulator were mainly responsible for the motion of the end position, and the latter three joints were responsible for the motion of the end posture. The dynamic parameter value of the former was larger, with obvious nonlinearity compared with that of the latter. To ensure that the simplified model maintained the generality of dynamics, the model was simplified with the first three degrees of freedom within the allowable error range, as shown in Figure 1.

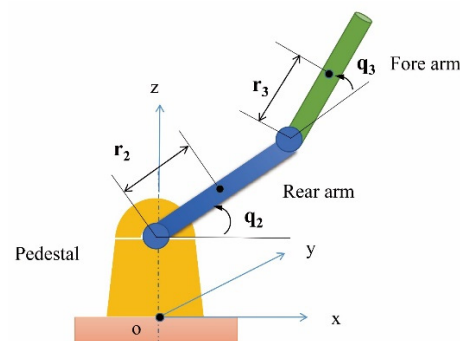


Figure 1. Simplified model of the manipulator.

For $i = 1, 2$, and 3 , m_i denotes the mass of the connecting rod i , r_i denotes the distance from joint i to the center of gravity of the connecting rod i , l_i represents the length of the connecting rod i , and q_i represents the rotating angle of the connecting rod i .

This model uses the Cartesian coordinate system. The centroid position of each joint is

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{2}l_1 \end{bmatrix} \quad (1)$$

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{2}l_2c_2c_1 \\ \frac{1}{2}l_2c_2s_1 \\ l_1 + \frac{1}{2}l_2s_1 \end{bmatrix} \quad (2)$$

$$\begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix} = \begin{bmatrix} l_2c_2c_1 + \frac{1}{2}l_3c_{23}c_1 \\ l_2c_2s_1 + \frac{1}{2}l_3c_{23}s_1 \\ l_1 + l_2s_1 + \frac{1}{2}l_3s_{23} \end{bmatrix} \quad (3)$$

The centroid velocity of each joint is

$$\begin{bmatrix} \dot{x}_1 \\ \dot{y}_1 \\ \dot{z}_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \dot{q}_1 \quad (4)$$

$$\begin{bmatrix} \dot{x}_2 \\ \dot{y}_2 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2}l_2c_2s_1 & -\frac{1}{2}l_2c_1s_2 \\ \frac{1}{2}l_2c_1c_2 & -\frac{1}{2}l_2s_1s_2 \\ 0 & \frac{1}{2}l_2c_2 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \quad (5)$$

$$\begin{bmatrix} \dot{x}_3 \\ \dot{y}_3 \\ \dot{z}_3 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2}l_3c_{23}s_1 - l_2c_2s_1 & -\frac{1}{2}l_3s_{23}c_1 - l_2c_1s_2 & -\frac{1}{2}l_3s_{23}c_1 \\ \frac{1}{2}l_3c_{23}c_1 + l_2c_1c_2 & -\frac{1}{2}l_3s_{23}s_1 - l_2s_1s_2 & -\frac{1}{2}l_3s_{23}s_1 \\ 0 & \frac{1}{2}l_3c_{23} + l_2c_2 & \frac{1}{2}l_3c_{23} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix}, \quad (6)$$

The total kinetic energy of the system is

$$\begin{aligned} K &= \sum_{i=1}^3 K_i = K_1 + K_2 + K_3 \\ &= \frac{1}{4}m_1r_1^2\dot{q}_1^2 + \frac{1}{6}m_2l_2\dot{q}_2^2 + \frac{1}{2}m_2(v_{x_2}^2 + v_{y_2}^2 + v_{z_2}^2) \\ &\quad + \frac{1}{24}m_3l_3\dot{q}_3^2 + \frac{1}{2}m_3(v_{x_3}^2 + v_{y_3}^2 + v_{z_3}^2) \end{aligned} \quad (7)$$

The total potential energy of the system is

$$\begin{aligned} P &= \sum_{i=1}^3 P_i = P_1 + P_2 + P_3 \\ &= \frac{1}{2}m_1gl_1 + m_2g(l_1 + \frac{1}{2}l_2s_1) \\ &\quad + m_2g(l_1 + l_2s_1 + \frac{1}{2}l_3s_{23}) \end{aligned} \quad (8)$$

The Lagrange function of the system is

$$\begin{aligned} L &= K - P = \sum_{i=1}^3 K_i - \sum_{i=1}^3 P_i \\ &= K_1 + K_2 + K_3 - P_1 - P_2 - P_3 \end{aligned} \quad (9)$$

Substituting Equation (9), we obtain

$$\tau_i = \frac{d}{dt} \frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i}, i = 1, 2, \dots, n \quad (10)$$

The following can be obtained:

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = \tau \quad (11)$$

$$\begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \end{bmatrix} + \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} + \begin{bmatrix} g_1 \\ g_2 \\ g_3 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} \quad (12)$$

where

$$\begin{aligned} m_{11} &= I_1 + a_1 \cos^2(q_2) + a_2^2 \cos(q_2 + q_3) \\ &\quad + a_3 \cos(q_2) \cos(q_2 + q_3) \\ m_{22} &= I_2 + a_1 + a_2 + 2a_3 \cos(q_3) \\ m_{33} &= I_3 + a_2 \\ m_{23} &= m_{32} = a_2 + a_3 \cos(q_3) \\ m_{12} &= m_{13} = m_{21} = m_{31} = 0 \\ c_{11} &= -\frac{1}{2}a_1\dot{q}_2 \sin(2q_2) - \frac{1}{2}a_2(\dot{q}_2 + \dot{q}_3) \sin(2q_2 + 2q_3) \\ &\quad - a_3\dot{q}_2 \sin(2q_2 + 2q_3) - a_3\dot{q}_3 \cos(2q_2) \sin(2q_2 + 2q_3) \\ c_{12} &= -\frac{1}{2}a_1\dot{q}_1 \sin(2q_2) - \frac{1}{2}a_2\dot{q}_1 \sin(2q_2 + q_3) \\ &\quad - a_3\dot{q}_1 \sin(2q_2 + q_3) \\ c_{13} &= -\frac{1}{2}a_1\dot{q}_1 \sin(2q_2 + q_3) \\ &\quad - a_3\dot{q}_1 \cos(2q_2) \sin(q_2 + q_3) \\ c_{22} &= -a_3\dot{q}_3 \sin(q_3) \\ c_{23} &= -a_3(\dot{q}_2 + \dot{q}_3) \sin(q_3) \\ c_{32} &= -a_3\dot{q}_2 \sin(q_3) \\ c_{21} &= -c_{12} \\ c_{31} &= -c_{13} \\ c_{33} &= 0 \\ g_1 &= 0 \\ g_2 &= b_1 \cos(q_2) + b_2 \cos(q_2 + q_3) \\ g_3 &= b_2 \cos(q_2 + q_3) \end{aligned}$$

where

$$\begin{aligned} a_1 &= m_2 r_2^2 + m_3 l_2^2 \\ a_2 &= m_3 r_3^2 \\ a_3 &= m_3 r_3 l_2 \\ b_1 &= (m_2 r_2 + m_3 r_3 l_2)g \\ b_2 &= m_3 r_3 g \end{aligned}$$

I_1 , I_2 , and I_3 are pseudo-inertial matrices.

The external uncertainty interference in the system is considered. Then, the dynamic equation of the system is

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + \tau_d = \tau \quad (13)$$

where

$$\tau_d = \begin{bmatrix} \tau_{d1} \\ \tau_{d2} \\ \tau_{d3} \end{bmatrix} \quad (14)$$

3. Sliding Mode Variable Structure Control

Let the state of the system be $x_1 = q$ and $x_2 = \dot{q}$. Then, the state equation of the system is

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = f + g\tau + d \end{cases} \quad (15)$$

$f = -M^{-1}[C(q, \dot{q})\dot{q} + G(q)]$ and $g = M^{-1}$, $d = -M^{-1}\tau_d$. Then, the control target is x_1 to track x_{1d} , where τ is the control input matrix, and τ_d represents uncertain external factors.

The error of the system is defined as

$$e = x_1 - x_{1d} \quad (16)$$

Then, the derivative of the error is

$$\dot{e} = x_2 - \dot{x}_{1d} \quad (17)$$

The sliding mode function is defined as

$$s = ce + \dot{e} \quad (18)$$

where c is a positive integer greater than zero.

The derivation of the above equation is performed to obtain

$$\dot{s} = c\dot{e} + f + g\tau + d - \ddot{x}_{1d} \quad (19)$$

The constant reaching law was used to reduce the chattering in the system and improve the robustness of the system. The constant reaching law is as follows:

$$\dot{s} = -\varepsilon \text{sign}(s) \quad (20)$$

where $\varepsilon > 0$.

Then, the control law of the system is

$$\tau = g^{-1}(-c\dot{e} - f - d + \ddot{x}_{1d} - \varepsilon \text{sign}(s)) \quad (21)$$

The discrete switching characteristic of the traditional sliding mode variable structure control (SMC) leads to the oscillation of the system near the sliding mode surface, which leads to the chattering of the system at the later stage of trajectory tracking. Even with the addition of the constant velocity reaching law, the chattering of the system cannot be avoided; this leads to the degradation of the tracking performance.

4. High-Order Sliding Mode Neural Network Adaptive Control (HOSMC-NNA)

The sliding mode switch surface function is defined as

$$S = x_1 - x_{1d} \quad (22)$$

The derivation of the above equation is performed to obtain

$$\dot{S} = x_2 - \dot{x}_{1d} \quad (23)$$

Since the control signal τ of the manipulator does not appear in the first derivative of the sliding mode switch surface function, the derivative of the above formula is performed to obtain

$$\ddot{S} = \dot{x}_2 - \ddot{x}_{1d} = f + g\tau + d - \ddot{x}_{1d} \quad (24)$$

If the finite time tracking of the system is considered, let $S = z_1$ and $\dot{S} = z_2$. Then,

$$\begin{cases} \dot{z}_1 = z_2 \\ \dot{z}_2 = f + g\tau + d - \ddot{x}_{1d} \end{cases} \quad (25)$$

The feedback control of the system is considered to obtain

$$\tau = \frac{1}{g}(u - f + \ddot{x}_{1d} - d) \quad (26)$$

Here, $u = u_{nom} + u_{vst}$, where u_{nom} is a homogeneous continuous control term, with $u_{nom} = -a_1 \text{sign}(s)|s|^{k_1} - a_2 \text{sign}(\dot{s})|\dot{s}|^{k_2}$.

In this equation, $a_1 > 0$, $a_2 > 0$, $0 < k_1 < 1$, and $k_2 = \frac{2k_1}{1+k_1}$. The continuous control law term u_{nom} makes the nominal system (that is, in the absence of external interference) converge to zero in a finite time.

In the presence of an external disturbance of the system, $d \neq 0$, the stability of the system cannot be guaranteed for a finite time when the auxiliary control item only considers the homogeneous continuous control law u_{nom} . Hence, the second-order sliding mode super spiral algorithm [32] is employed to achieve the robustness of the system.

$$u_{vst} = -\alpha|S|^{\frac{1}{2}}\text{sign}(S) + u_{ast} \quad (27)$$

$$\dot{u}_{ast} = -\beta\text{sign}(S) \quad (28)$$

where u_{ast} is a super-twisting function, and α and β are the super-twisting function control gains. The adaptive law can be designed as

$$\begin{cases} \dot{\hat{a}} = a|S|^{\frac{1}{2}} + c\text{sign}(S)\dot{u}_{ast} \\ \dot{\hat{\beta}} = 2c|S|^{\frac{1}{2}} + 2b\text{sign}(S)\dot{u}_{ast} \end{cases} \quad (29)$$

where a , b , and c are constants. In addition, $\begin{bmatrix} a & c \\ c & b \end{bmatrix}$ should be ensured to be a positive definite matrix.

The uncertainty in the modeling process of the system can reduce the trajectory tracking performance of the system. Although the homogeneous continuous control term and the super-twisting adaptive algorithm are added to the control law, the control law cannot compensate the modeling uncertainty in the system.

In the expression $f = -M^{-1}[C(q, \dot{q})\dot{q} + G(q)]$, f contains all the model uncertainty information. The goal is to use the RBF neural network to approximate f .

The RBF is a Gaussian function, expressed as $h_i = \exp \frac{\|x - c_i\|^2}{b_i^2}$ with $i = 1, 2 \dots m$, where x is the input of the RBF neural network, c_i is the center value of the input data, and b_i is the expansion constant or base width of the basis function. In

$$f(x) = W^T h + \varepsilon \quad (30)$$

W is the weight in the middle layer of the RBF neural network, and ε is a small positive real number, representing the approximation error of the neural network to the nonlinear uncertainty function. In

$$\hat{f}(x) = \hat{W}^T h \quad (31)$$

$\hat{f}(x)$ is the approximation of the neural network to function $f(x)$, and $\hat{W}^T$ is the ideal weight.

The above two equations are combined to obtain the following equation.

$$f(x) - \hat{f}(x) = W^T h + \varepsilon - \hat{W}^T h = \tilde{W}^T h + \varepsilon \quad (32)$$

where $\tilde{W} = W - \hat{W}$.

The RBF neural network weight adaptive law is designed as

$$\dot{\hat{W}} = -\gamma E^T P B h \quad (33)$$

where γ is a positive constant greater than zero, $E = [S \ dS]^T$, P is a symmetric positive definite matrix, $B = [0 \ 1]^T$, h is the neural network function.

This method uses a high-order sliding mode control, a homogeneous continuous control, and a super-spiral adaptive algorithm and thus can effectively reduce the chattering

of the system. At the same time, the neural network adaptive control algorithm is used to compensate the system modeling errors. Thus, this method not only speeds up the trajectory tracking performance but also enhances the robustness of the system. The weights of the middle layer of the neural network are adjusted by the designed adaptive law. However, the method has a low convergence speed and it can easily fall into the local optimum.

5. Proof of Stability

A stable adaptive controller design for manipulators using neural networks has been proposed, and the stability analysis of an RBF neural network control scheme has been performed for a robot manipulator [33].

Define the matrix $A = \begin{bmatrix} -\frac{\alpha}{2} & \frac{1}{2} \\ -\beta & 0 \end{bmatrix}$; when $\alpha > 0$ and $\beta > 0$, A is a Hurwitz matrix. Therefore, for any positive definite matrix Q , there must be a positive definite symmetric matrix P that satisfies the Lyapunov equation:

$$A^T P + P A = -Q \quad (34)$$

Define the matrix $\hat{A} = \begin{bmatrix} -\frac{\hat{\alpha}}{2} & \frac{1}{2} \\ -\hat{\beta} & 0 \end{bmatrix}$, where α and β are taken as the upper bounds of $\hat{\alpha}$ and $\hat{\beta}$, respectively. Next, define the errors $\tilde{\alpha} = \alpha - \hat{\alpha}$ and $\tilde{\beta} = \beta - \hat{\beta}$; then $\tilde{A} = \begin{bmatrix} -\frac{\tilde{\alpha}}{2} & 0 \\ -\tilde{\beta} & 0 \end{bmatrix}$.

Define the Lyapunov function as

$$V_0(\xi) = \xi^T P \xi \quad (35)$$

where $V_0(\xi)$ is a continuous positive definite function and is radially unbounded. Therefore, it can be derived by the chain rule.

Assume $x_1 = S$, $x_2 = -\beta \int_0^t \text{sign}(x_1) dt$

Consider

$$\xi^T = [\xi_1 \ \xi_2] = [\text{sign}(x_1)|x_1|^{\frac{1}{2}} \ x_2] \quad (36)$$

$$B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$C = [1, \ 0]$$

From the derivation of Equation (36), we obtain

$$\dot{\xi} = \begin{bmatrix} \dot{\xi}_1 \\ \dot{\xi}_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{2|x_1|^{\frac{1}{2}}} [-\hat{\alpha}|x_1|^{\frac{1}{2}} \text{sign}(x_1) + x_2] \\ -\hat{\beta} \text{sign}(x_1) \end{bmatrix} = \frac{1}{|x_1|^{\frac{1}{2}}} (\hat{A} \xi + B) \quad (37)$$

$$\begin{aligned} \dot{\xi}^T &= [\dot{\xi}_1 \ \dot{\xi}_2] = \left[\frac{1}{2|x_1|^{\frac{1}{2}}} [-\hat{\alpha}|x_1|^{\frac{1}{2}} \text{sign}(x_1) + x_2] \ -\hat{\beta} \text{sign}(x_1) \right] \\ &= \frac{1}{|x_1|^{\frac{1}{2}}} (\xi^T \hat{A}^T + B^T) \end{aligned} \quad (38)$$

From the derivation of Equation (35), we obtain

$$\begin{aligned} \dot{V}_0(\xi) &= \dot{\xi}^T P \xi + \xi^T P \dot{\xi} = \frac{1}{|x_1|^{\frac{1}{2}}} (\xi^T \hat{A}^T + B^T) P \xi + \xi^T P \frac{1}{|x_1|^{\frac{1}{2}}} (\hat{A} \xi + B) \\ &= \frac{1}{|x_1|^{\frac{1}{2}}} (\xi^T \hat{A}^T P \xi + \xi^T P \hat{A} \xi) + \frac{1}{|x_1|^{\frac{1}{2}}} (B^T P \xi + \xi^T P B) \\ &= \frac{1}{|x_1|^{\frac{1}{2}}} (\xi^T A^T P \xi + \xi^T P A \xi) - \frac{1}{|x_1|^{\frac{1}{2}}} (\xi^T \tilde{A}^T P \xi + \xi^T P \tilde{A} \xi) + \frac{1}{|x_1|^{\frac{1}{2}}} (B^T P \xi + \xi^T P B) \end{aligned} \quad (39)$$

Given that

$$\dot{V}_1(\xi) = \frac{1}{|x_1|^{\frac{1}{2}}} (\xi^T A^T P \xi + \xi^T P A \xi) + \frac{1}{|x_1|^{\frac{1}{2}}} (B^T P \xi + \xi^T P B) \quad (40)$$

$$\dot{V}_2(\xi) = \frac{1}{|x_1|^{\frac{1}{2}}} (\xi^T \tilde{A}^T P \xi + \xi^T P \tilde{A} \xi) \quad (41)$$

we have

$$\begin{aligned} \dot{V}_2(\xi) &= \frac{1}{|x_1|^{\frac{1}{2}}} (\xi^T \tilde{A}^T P \xi + \xi^T P \tilde{A} \xi) \\ &= \frac{1}{|x_1|^{\frac{1}{2}}} \xi^T (\tilde{A}^T P + P \tilde{A}) \xi \\ \text{Given} &= \frac{1}{|x_1|^{\frac{1}{2}}} \begin{bmatrix} \text{sign}(x_1) |x_1|^{\frac{1}{2}} & x_2 \end{bmatrix} \begin{bmatrix} -\tilde{\alpha}a - 2\tilde{\alpha}c - \frac{\tilde{\alpha}}{2}c - \tilde{\alpha}b & \\ -\frac{\tilde{\alpha}}{2}c - \tilde{\alpha}b & 0 \end{bmatrix} \begin{bmatrix} \text{sign}(x_1) |x_1|^{\frac{1}{2}} \\ x_2 \end{bmatrix} \\ &= \frac{1}{|x_1|^{\frac{1}{2}}} \left[-\tilde{\alpha}a |x_1| - 2\tilde{\beta}c |x_1| - 2\left(\frac{\tilde{\alpha}}{2}c + \tilde{\beta}b\right) \text{sign}(x_1) |x_1| x_2 \right] \\ &= -\tilde{\alpha}a |x_1|^{\frac{1}{2}} - 2\tilde{\beta}c |x_1|^{\frac{1}{2}} - 2\left(\frac{\tilde{\alpha}}{2}c + \tilde{\beta}b\right) \text{sign}(x_1) x_2 \\ &= -\tilde{\alpha} \left[a |x_1|^{\frac{1}{2}} + c \text{sign}(x_1) x_2 \right] - 2\tilde{\beta} \left[c |x_1|^{\frac{1}{2}} + b \text{sign}(x_1) x_2 \right] \end{aligned} \quad (42)$$

$$\begin{aligned} \dot{V}_1(\xi) &= \frac{1}{|x_1|^{\frac{1}{2}}} (\xi^T A^T P \xi + \xi^T P A \xi) + \frac{1}{|x_1|^{\frac{1}{2}}} (B^T P \xi + \xi^T P B) \\ &= \frac{1}{|x_1|^{\frac{1}{2}}} (\xi^T A^T P \xi + \xi^T P A \xi + B^T P \xi + \xi^T P B) \\ &\leq \frac{1}{|x_1|^{\frac{1}{2}}} (\xi^T A^T P \xi + \xi^T P A \xi + B^T P \xi + \xi^T P B) \\ &\leq \frac{1}{|x_1|^{\frac{1}{2}}} (\xi^T A^T P \xi + \xi^T P A \xi + \xi^T P B B^T P \xi) \\ &= \frac{1}{|x_1|^{\frac{1}{2}}} \xi^T (A^T P + P A + P B B^T P) \xi \end{aligned} \quad (43)$$

$$\begin{aligned} Q &= -(A^T P + P A + P B B^T P) \\ &= -\left\{ \begin{bmatrix} -\frac{\alpha}{2} & -\beta \\ \frac{\alpha}{2} & 0 \end{bmatrix} \begin{bmatrix} a & c \\ c & b \end{bmatrix} + \begin{bmatrix} a & c \\ c & b \end{bmatrix} \begin{bmatrix} -\frac{\alpha}{2} & \frac{1}{2} \\ -\beta & 0 \end{bmatrix} + \begin{bmatrix} a & c \\ c & b \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} [0 \ 1] \begin{bmatrix} a & c \\ c & b \end{bmatrix} \right\} \\ &= \begin{bmatrix} \alpha a + 2\beta c - c^2 & \frac{\alpha}{2}c + \beta b - \frac{1}{2}a - bc \\ \frac{\alpha}{2}c + \beta b - \frac{1}{2}a - bc & -b^2 - c \end{bmatrix} \end{aligned}$$

We have

$$\dot{V}_0(\xi) \leq -\frac{1}{|x_1|^{\frac{1}{2}}} \xi^T Q \xi + \tilde{\alpha} [a |x_1|^{\frac{1}{2}} + c \text{sign}(x_1) x_2] + 2\tilde{\beta} [c |x_1|^{\frac{1}{2}} + b \text{sign}(x_1) x_2] \quad (44)$$

where

$$-\frac{1}{|x_1|^{\frac{1}{2}}} \xi^T Q \xi \leq -\frac{1}{|x_1|^{\frac{1}{2}}} \xi^T \xi = -\frac{1}{|x_1|^{\frac{1}{2}}} \|\xi\|^2 \leq -\|\xi\|_2 \quad (45)$$

For $V_0(\xi)$, we obtain

$$\lambda_{\min}(P) \|\xi\|_2^2 \leq V_0(\xi) = \xi^T P \xi \leq \lambda_{\max}(P) \|\xi\|_2^2 \quad (46)$$

Then,

$$\left[\frac{V_0(\xi)}{\lambda_{\max}(P)} \right]^{\frac{1}{2}} \leq \|\xi\|_2 \Rightarrow \frac{V_0^{\frac{1}{2}}(\xi)}{\lambda_{\max}^{\frac{1}{2}}(P)} \leq \|\xi\|_2 \quad (47)$$

If $r = \frac{1}{\lambda_{\max}^{\frac{1}{2}}(P)}$, then

$$r V_0^{\frac{1}{2}}(\xi) \leq \|\xi\|_2 \Rightarrow \dot{V}_0(\xi) \leq -r V_0^{\frac{1}{2}}(\xi) \quad (48)$$

Consequently,

$$\dot{V}_0(\xi) \leq -rV_0^{\frac{1}{2}}(\xi) + \tilde{\alpha}[a|x_1|^{\frac{1}{2}} + c\text{sign}(x_1)x_2] + 2\tilde{\beta}[c|x_1|^{\frac{1}{2}} + b\text{sign}(x_1)x_2] \quad (49)$$

The approximation error of the neural network model is defined as:

$$\omega = \hat{f}(x|W^*) - f(x)$$

where W^* is the optimal weight.

The overall Lyapunov function is considered as follows:

$$V = V_0(\xi) + \frac{1}{2}\tilde{\alpha}^2 + \frac{1}{2}\tilde{\beta}^2 + \frac{1}{2}E^T XE + \frac{1}{2\gamma}(\hat{W} - W^*)^T(\hat{W} - W^*) \quad (50)$$

From the derivation of Equation (50), we get

$$\begin{aligned} \dot{V} &= \dot{V}_0(\xi) + \tilde{\alpha}\dot{\tilde{\alpha}} + \tilde{\beta}\dot{\tilde{\beta}} + \frac{1}{2}\dot{E}^T XE + \frac{1}{2}E^T X\dot{E} + \frac{1}{\gamma}(\hat{W} - W^*)^T \dot{\hat{W}} \\ &= \dot{V}_0(\xi) - \tilde{\alpha}\dot{\tilde{\alpha}} - \tilde{\beta}\dot{\tilde{\beta}} + \frac{1}{2}\dot{E}^T XE + \frac{1}{2}E^T X\dot{E} + \frac{1}{\gamma}(\hat{W} - W^*)^T \dot{\hat{W}} \\ &\leq -rV_0^{\frac{1}{2}}(\xi) + \tilde{\alpha}[a|x_1|^{\frac{1}{2}} + c\text{sign}(x_1)x_2] + 2\tilde{\beta}[c|x_1|^{\frac{1}{2}} + b\text{sign}(x_1)x_2] - \tilde{\alpha}\dot{\tilde{\alpha}} - \tilde{\beta}\dot{\tilde{\beta}} \\ &\quad - \frac{1}{2}E^T HE + (\hat{W} - W^*)^T E^T X B h(x) + E^T X B \omega + \frac{1}{\gamma}(\hat{W} - W^*)^T \dot{\hat{W}} \\ &= -rV_0^{\frac{1}{2}}(\xi) + \tilde{\alpha}[a|x_1|^{\frac{1}{2}} + c\text{sign}(x_1)x_2 - \dot{\tilde{\alpha}}] + 2\tilde{\beta}[c|x_1|^{\frac{1}{2}} + b\text{sign}(x_1)x_2 - \frac{1}{2}\dot{\tilde{\beta}}] \\ &\quad - \frac{1}{2}E^T HE + (\hat{W} - W^*)^T E^T X B h(x) + E^T X B \omega + \frac{1}{\gamma}(\hat{W} - W^*)^T [\dot{\hat{W}} + \gamma E^T X B h(x)] \end{aligned} \quad (51)$$

By substituting the adaptive law (29) into Equation (51), we obtain

$$\dot{V} \leq -rV_0^{\frac{1}{2}}(\xi) - \frac{1}{2}E^T HE + E^T P B \omega \quad (52)$$

where ω is the model approximation error. The model approximation error is small enough so that $\dot{V} \leq 0$.

According to the Lyapunov stability theory, the system is stable in a finite time.

6. Improved Particle Swarm Optimization Algorithm (IPSO)

In the basic PSO algorithm, each particle corresponds to a feasible solution of an optimization problem. The quality of the particles is determined by the fitness function set in advance.

In the traditional PSO algorithm, the convergence accuracy of the algorithm is low. Furthermore, in the later stage of the algorithm convergence, the reduction in particle swarm diversity causes the algorithm to fall into the local optimum, that is, the algorithm becomes "premature". Moreover, particles are easily trapped in the poor search area and cannot easily escape. This can lead to search stagnation, resulting in a reduction in the algorithm convergence efficiency. To effectively avoid the above-mentioned problems, this paper proposes an improved PSO algorithm.

The particle-update formula of the traditional PSO algorithm includes three parts: the velocity value of the previous-generation particle, the learning part of a single particle, and the mutual learning part between particle swarms. The first part is controlled by the weighting factor ω , the second part is controlled by the acceleration factor c_1 , and the third part is controlled by the acceleration factor c_2 . In order to improve the global search ability of the PSO in the initial stage, the chaos theory was used to initialize the initial particle swarm. It is expressed as

$$\lambda_{n+1} = a\lambda_n(1 - \lambda_n) \quad (53)$$

where λ_0 represents the initial value, which is a random number in the interval of (0,1), a represents the logistic parameter, and $a \in [0, 4]$. When a is close to 4, the sequence generated

by the logistic model has the characteristics of aperiodicity and non-convergence. Hence, $a = 4$ was adopted.

Therefore, the above formula became

$$\lambda_{n+1} = 4\lambda_n(1 - \lambda_n) \quad (54)$$

The chaotic motion is traversed by various states in its chaotic attraction domain without losing the randomness requirement of particle swarm initialization. Therefore, the sequence generated by chaos has the characteristics of divergence and randomness such that it can improve the group search ability of the particle swarm.

In the PSO algorithm, the variance of the group fitness, that is, the entropy δ^2 between particles, is used to evaluate the degree of convergence or dispersion of all particles in the swarm. The larger the value of δ^2 , the larger can be the entropy value between the particles in a swarm, which means that the more dispersed the particles are, whereas, in the opposite situation, the more the particles converge. The formula for calculating the entropy δ^2 between particles in a swarm is as follows:

$$\delta^2 = \frac{1}{n} \sum_{i=1}^n \left(\frac{f(t_i) - \bar{f}}{\bar{f}} \right)^2 \quad (55)$$

where $f(t_i)$ represents the fitness value of the i -th particle, and $\bar{f}$ represents the mean value of the particle fitness values in the current particle swarm:

$$\bar{f} = \begin{cases} \max\{|f_i - \bar{f}|\} & \max\{|f_i - \bar{f}|\} \geq 1 \\ 1 & \text{other} \end{cases} \quad (56)$$

The improved formula of inertia weight is

$$\omega = \omega_{\min} + \bar{\omega} * \frac{f(t_i) - \bar{f}}{\bar{f}} + \xi r(\omega_{\max} - \omega_{\min}) \quad (57)$$

where

$$\bar{\omega} = \frac{(\omega_{\max} - \omega_{\min})}{2} \quad (58)$$

$$\xi = \frac{T - t}{T} \quad (59)$$

here, t is the current iteration number, and T is total number of iterations. The improved acceleration constants are

$$c_1 = c_{1s} - \frac{1.4(T - t)}{T} \quad (60)$$

$$c_2 = c_{2s} + \frac{1.6(T - t)}{T} \quad (61)$$

where c_{1s} and c_{2s} are the initial values of c_1 and c_2 , respectively.

When $|\delta^2| \leq \varepsilon$, the diversity of the particles in the swarm population is poor, and the algorithm is more likely to fall into the local optimal solution. Therefore, the following formula is used to update the position of the particles, which lets the particles overcome the local optimal solution.

$$x_{id}(t+1) = x_{id}(t) + r_2(p_{gd}(t) - x_{id}(t)) \quad (62)$$

To verify the validity of the IPSO algorithm, in this paper, five standard test functions were randomly selected for function optimization. The basic mathematical properties of the selected five standard test functions are shown in Table 1, where U represents a single peak, M represents multiple peaks, N represents inseparable peaks, and S represents separable peaks. To ensure the validity of the data obtained in the experiment, 50 statistical

experiments were performed for each standard test function. The optimal value, average optimal value (Mean), and standard deviation (Std) according to the statistical results were calculated. In addition, the performance of the IPSO algorithm was evaluated and compared with that of the traditional PSO algorithm. For both algorithms, the same parameters were set, with the number of populations being 50, and the number of iterations being 200. The experimental results are shown in Table 2.

Table 1. Standard test functions.

Function	Function Name	Dimension	Range of the Independent Variable	Function Formula	Function Feature	Theoretical Values
f_1	Sphere	D	$[-100,100]$	$f(X) = \sum_{i=1}^D x_i^2$	US	$f_{\min} = 0$
f_2	Rosenbrock	D	$[-4,4]$	$f(X) = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$	US	$f_{\min} = 0$
f_3	Ackley	D	$[-32,32]$	$f(X) = -20 \exp(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2}) - \exp(\frac{1}{D} \sum_{i=1}^D \cos(2\pi x_i)) + 20 + e$	MN	$f_{\min} = 0$
f_4	Rastrigin	D	$[-5.21,5.21]$	$f = \sum_{i=1}^D [x_i^2 - 10 \cos(2\pi x_i) + 10]$	MN	$f_{\min} = 0$
f_5	Noncontinuous Rastrigin	D	$[-5.21,5.21]$	$f = \sum_i [x_i^2 - 10 \cos(2\pi y_i) + 10]$ $y_i = \begin{cases} x_i, & x_i < 0.5 \\ \frac{\text{round}(2x_i)}{2}, & \text{else} \end{cases}$	MN	$f_{\min} = 0$

Table 2. Function optimization results.

Function	Function Dimension		Traditional Particle Swarm Optimization Algorithm	Improved Particle Swarm Optimization Algorithm
f_1	10	Optimal value	2.869000×10^{-12}	2.931071×10^{-22}
		Mean	6.149297×10^{-4}	1.103658×10^{-17}
		Std	2.296402×10^{-3}	3.203834×10^{-17}
f_2	10	Optimal value	2.136403	9.210587×10^{-2}
		Mean	6.277601	3.972730
		Std	2.077086	2.757420
f_3	10	Optimal value	3.253477×10^{-9}	2.269699×10^{-11}
		Mean	0.3731386	1.493499×10^{-9}
		Std	0.6930378	1.896008×10^{-9}
f_4	10	Optimal value	2.1520	1.0473
		Mean	2.69	1.13
		Std	8.43	1.73
f_5	10	Optimal value	2.82	1.102
		Mean	3.81	1.39
		Std	1.00	1.78

For different types of functions, f_1, f_2, f_3, f_4, f_5 , with different dimensions, we compared the traditional particle swarm optimization algorithm with the improved particle swarm optimization algorithm. The functions f_1 and f_2 are unimodal functions and were mainly used to test the convergence speed and optimization accuracy of the algorithm degree. The functions f_3, f_4 , and f_5 are multi-modal functions, which were mainly used to test the ability of the algorithm to jump out of the local optimum and avoid “premature” convergence. It can be seen from Tables 1 and 2 that the improved particle swarm

optimization algorithm could obtain the optimal value on both unimodal and multi-modal functions, but it was significantly better than the traditional particle swarm optimization algorithm in terms of accuracy of the convergence speed.

From the mean and standard deviation, it can be seen that the stability of the IPSO algorithm was better than that of the traditional PSO algorithm. Therefore, this paper used the IPSO algorithm to optimize the weights of the middle layer of the radial basis function neural network, so as to improve the approximation and compensation performance of the radial basis function neural network for the modeling uncertainty information in the manipulator system. In summary, the IPSO algorithm could achieve a relatively high convergence accuracy and was better than the traditional PSO algorithm. Figure 2 shows the flow chart of the particle swarm optimization neural network.

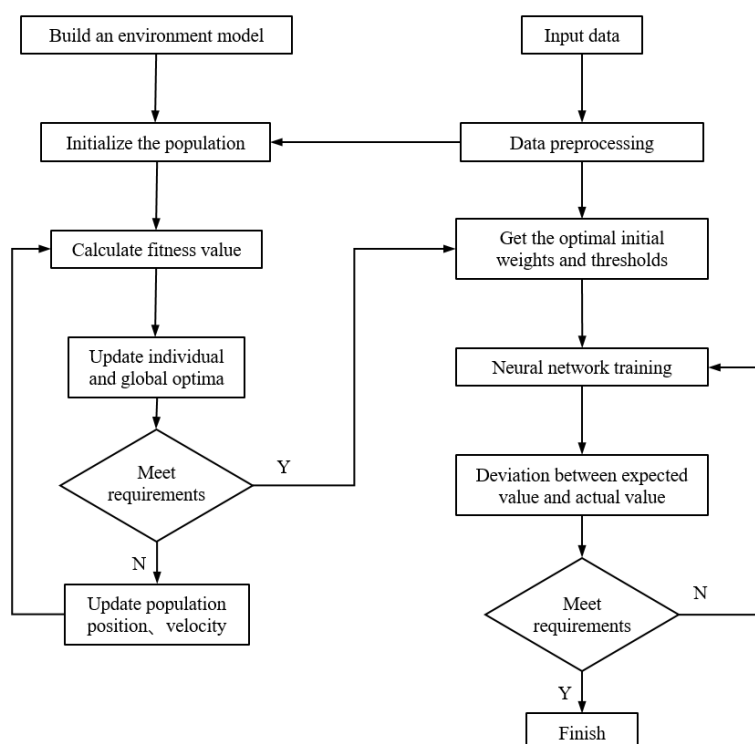


Figure 2. Flow chart of the particle swarm optimization neural network.

As shown in Figure 2, the first step was to build an environment model. The second step of the proposed model was to initialize the population with a random method in which the preprocessing data were input. The third step consisted in calculating the fitness value for calculating the entropy between the particles in a swarm using Equation (56). The fourth step involved updating the individual and globally optimizing population position and velocity. These steps were repeated until the requirements were satisfied. We then performed the fifth step and obtained the optimal initial weights and threshold the number of iterations or the error rate. If this was not achieved, the calculated fitness value was rejected until the conditions were met. The sixth step consisted in performing the neural network training. The seventh step is allowed obtaining the deviation between the expected value and the actual value. These steps were repeated until the stop condition was satisfied. Figure 3 shows the proposed control strategy.

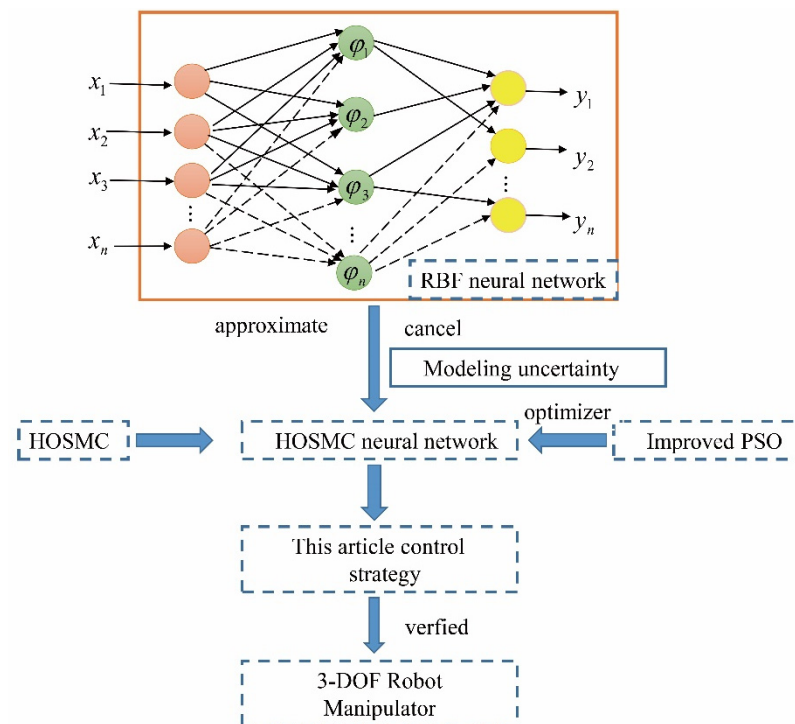


Figure 3. Control strategy block diagram.

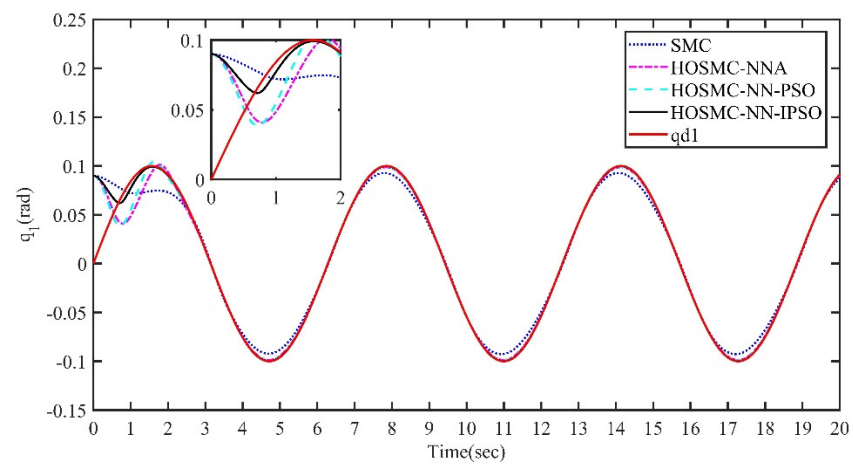
As shown in Figure 3, this study control strategy included the high-order sliding mode control (HOSMC), aiming at the uncertainty of system modeling. A neural network (NN) was adopted, and the improved particle swarm optimization algorithm (IPSO) was used to optimize the neural network. Finally, this method (HOSMC-NN-IPSO) was verified in a 3-DOF robot manipulator.

7. Simulation Analysis

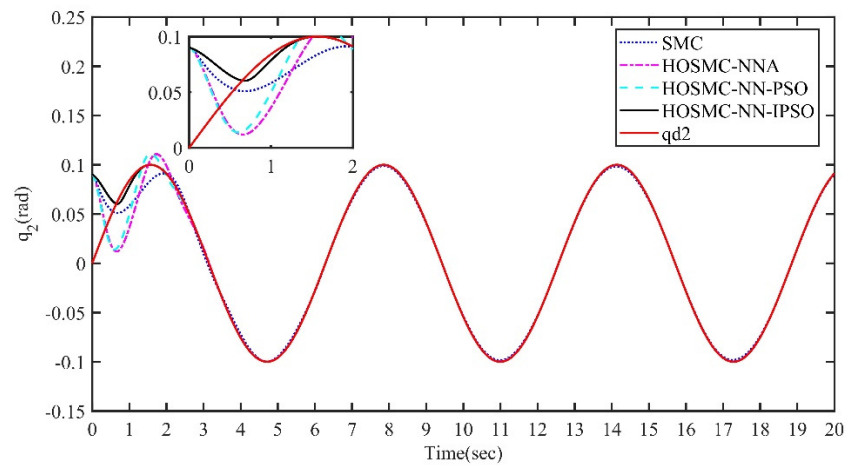
A three-joint rigid manipulator was considered in the simulation analysis. The system parameters were as follows: $m_1 = 5.022$, $m_2 = 10.764$, $m_3 = 10.386$, $l_2 = 0.29$, $l_3 = 0.121$, $r_2 = 0.145$, $r_3 = 0.0605$, $g = 9.8$, $I_1 = 5.04$, $I_2 = 10.99$, and $I_3 = 10.54$, where 1, 2, and 3, correspond to joints 1–3, respectively. In the design of the RBF neural network, the network structure was set to 2-7-1; the input of the neural network was $x = [e^T, \dot{e}^T]$; the ideal position tracking curves in the system were obtained as $q_{1d} = 0.1 \sin t$, $q_{2d} = 0.1 \sin t$, and $q_{3d} = 0.1 \sin t$; the initial state of the system was set to $(0.09, 0, 0.09, 0, 0.09, 0)$; the external disturbance in the system was $\tau_d = [0.5 \sin t, 0.5 \sin t, 0.5 \sin t]$. In the IPSO algorithm, $\omega_{\min} = 0.4$, $\omega_{\max} = 0.9$, $c_1 = c_2 = 2$, $c_{1s} = 3$, $c_{2s} = 1.5$; the number of populations was 50; the number of iterations was 200. The simulation was performed on MATLAB/Simulink, and the results are shown in Figures 4–7.

The figures show the curves of position tracking (Figure 4), tracking error (Figure 5), angular velocity tracking (Figure 6), and control input (Figure 7) of joints 1–3 under different controllers, namely, SMC, HOSMC–NNA, high–order sliding mode neural network control based on the PSO algorithm (HOSMC–NN–PSO), and high-order sliding mode neural network control based on the improved PSO algorithm (HOSMC–NN–IPSO).

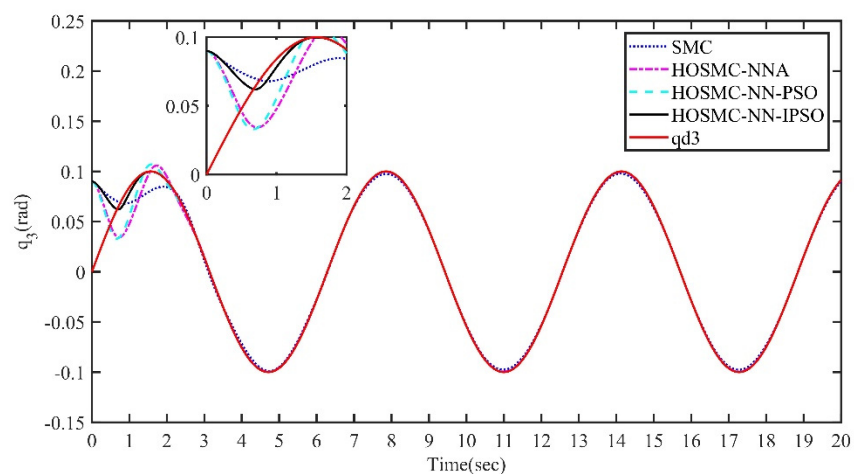
For joint 1, as shown in Figure 4a, compared with other methods, HOSMC–NN–IPSO showed a faster and more effective position tracking performance. As shown in Figure 5a, HOSMC–NN–IPSO had the smallest tracking error of q_1 within a finite time and converged to zero faster. Figures 4a and 5a show that HOSMC–NN–IPSO reduced the maximum overshoot.



(a)

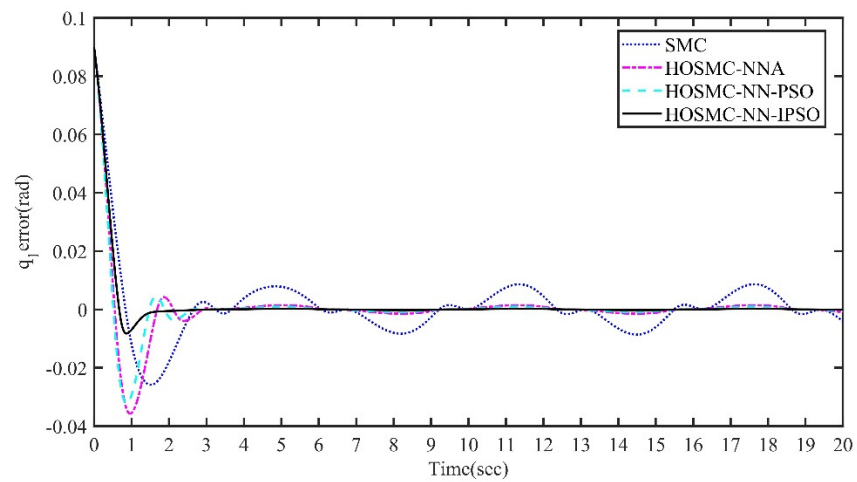


(b)

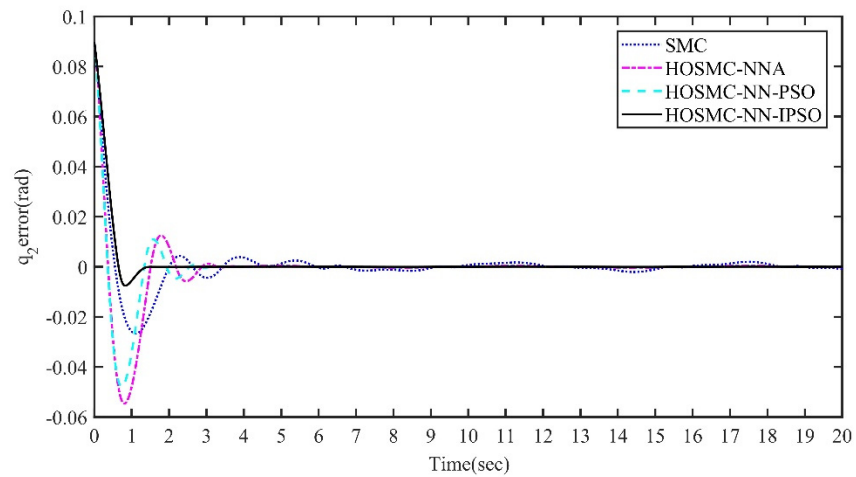


(c)

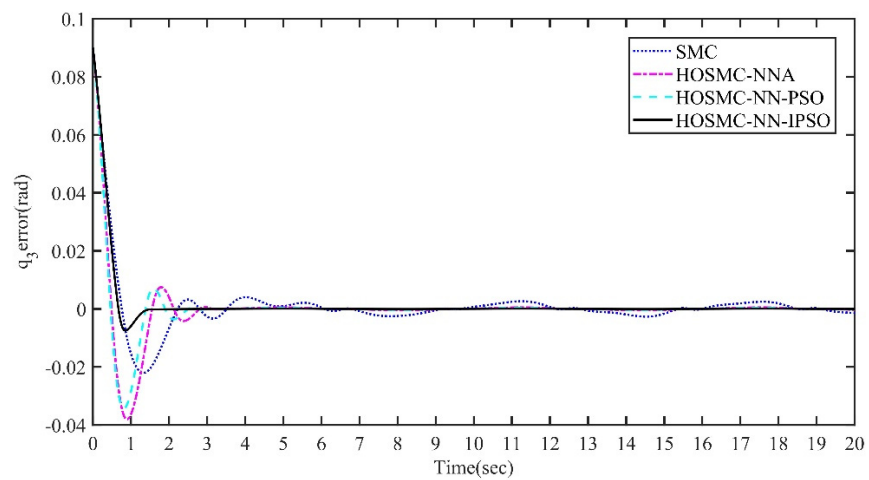
Figure 4. Position tracking of the joints under the SMC, HOSMC–NNA, HOSMC–NN–PSO and HOSMC–NN–IPSO controllers. (a) Position tracking of the joint 1. (b) Position tracking of the joint 2. (c) Position tracking of the joint 3.



(a)

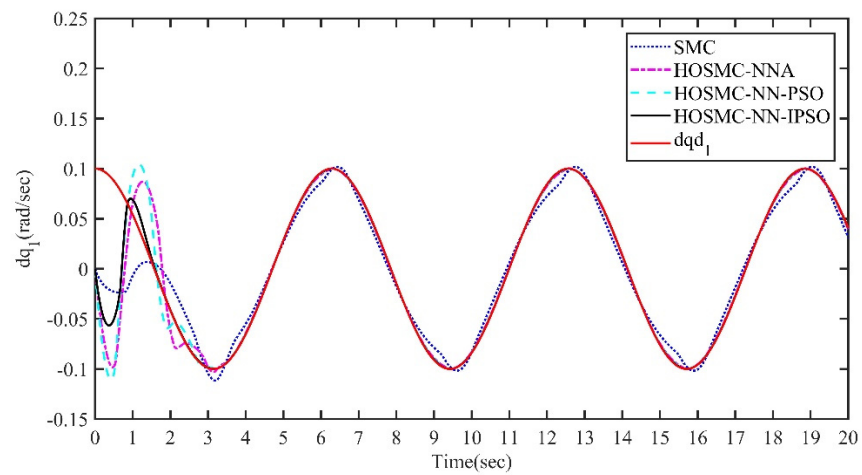


(b)

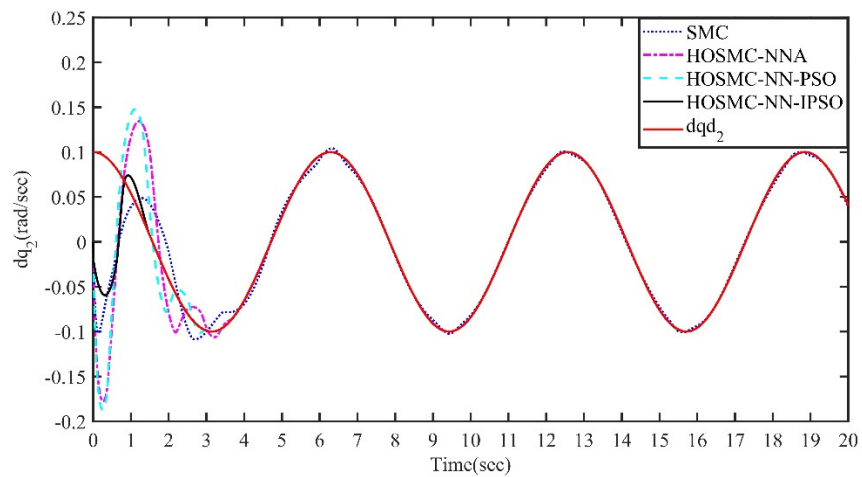


(c)

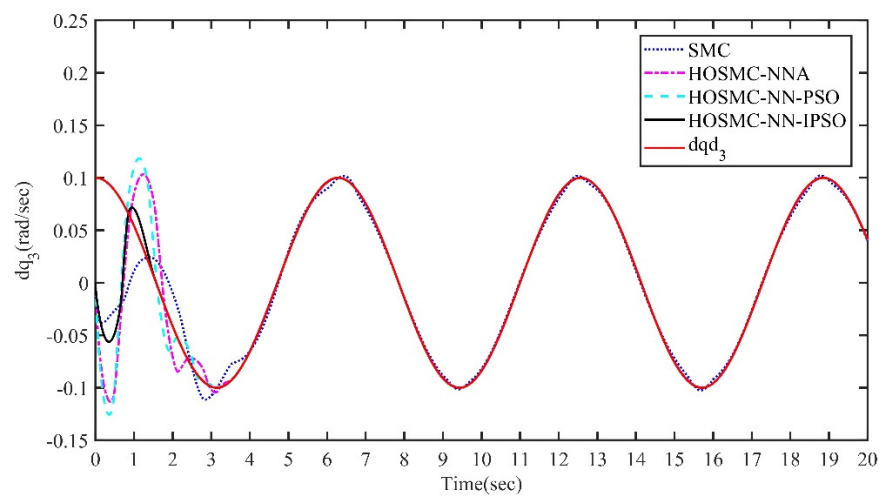
Figure 5. Position tracking error of the joints under the SMC, HOSMC–NNA, HOSMC–NN–PSO and HOSMC–NN–IPSO controllers. (a) Position tracking error of the joint 1. (b) Position tracking error of the joint 2. (c) Position tracking error of the joint 3.



(a)

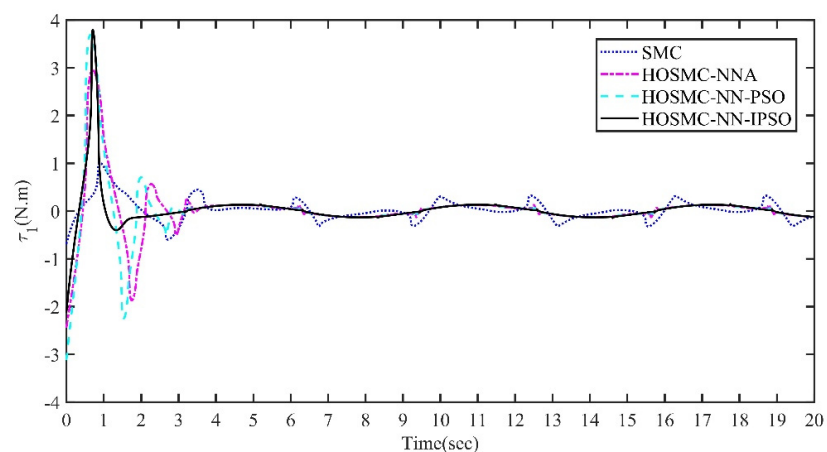


(b)

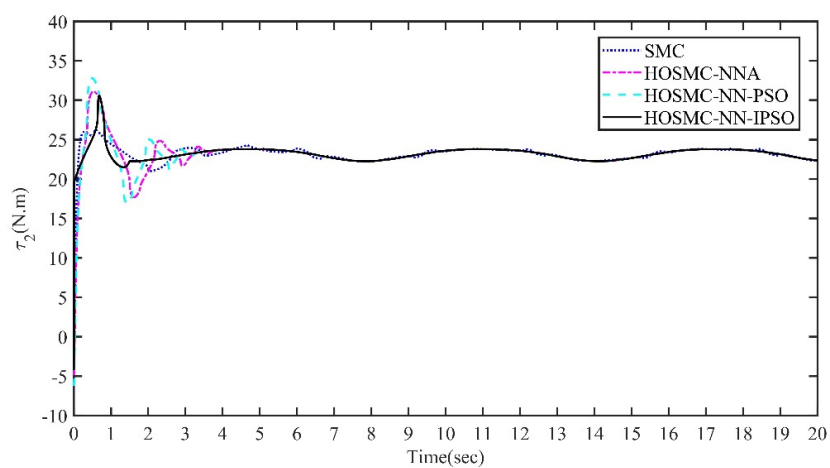


(c)

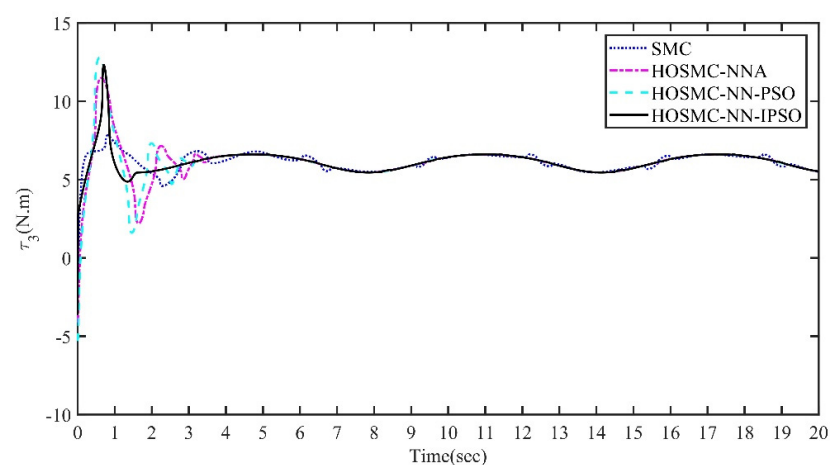
Figure 6. Angular velocity tracking of joints under SMC, HOSMC–NNA, HOSMC–NN–PSO and HOSMC–NN–IPSO controllers. (a) Angular velocity tracking of joint 1. (b) Angular velocity tracking of joint 2. (c) Angular velocity tracking of joint 3.



(a)



(b)



(c)

Figure 7. Control input of the joints under the SMC, HOSMC–NNA, HOSMC–NN–PSO and HOSMC–NN–IPSO controllers. (a) Control input of the joint 1. (b) Control input of the joint 2. (c) Control input of the joint 3.

Figure 4b shows that HOSMC–NN–PSO also completed position tracking within 2 s, while Figure 5b shows that its tracking error was significantly larger (by 0.45) than that of HOSMC–NN–IPSO. As shown in Figure 5a–c, especially in Figure 5a, the position tracking error of joints 1, 2, and 3 was significantly smaller with the HOSMC–NNA, HOSMC–NN–PSO, and HOSMC–NN–IPSO controllers than with the SMC, which illustrates the effect of the RBF neural network control in stability control.

Figure 6b shows that HOSMC–NN–PSO and HOSMC–NN–IPSO could complete the velocity tracking of dq_2 . However, the chattering of HOSMC–NN–IPSO was significantly reduced compared with that of other methods.

Figure 4c shows the position tracking of HOSMC–NN–IPSO for joint 3. HOSMC–NN–IPSO not only converged to the tracking curve faster but also considerably eliminated the chattering in the later stage. Figure 6c also shows that HOSMC–NN–IPSO had obvious advantages over other methods in minimizing the chattering in the later stage.

Figure 7a–c show that the control input of the HOSMC–NN–IPSO method converged faster than that of other methods and tended to stabilize within 2 s; thus, the chattering caused by SMC was eliminated in the later stage.

8. Conclusions

In this paper, a high-order sliding mode control method based on an improved PSO neural network is proposed for three-joint rigid manipulators. Among the various controllers studied in this paper, HOSMC–NN–IPSO was the fastest in tracking the trajectory of the manipulator. Moreover, in the later stage of the tracking, its tracking error was the smallest and could converge to 0 faster compared with other controllers. The control input of each joint manipulator could be smoothed after 2 s with small chattering in the later stage. Therefore, the proposed HOSMC–NN–IPSO controller showed a good trajectory tracking performance. It appears that it cannot only track the ideal trajectory quickly but also guarantee the performance of the system in the later stage of trajectory tracking. Furthermore, the system is more robust than other controllers.

In this control method, the homogeneous continuous control law and super-twisting adaptive algorithm were included to further strengthen the robustness of SMC. In addition, for the modeling of uncertainty in the system, the RBF neural network was used to approximate and compensate the modeling uncertainty in the system. The weights of the middle layer of the neural network were updated by applying an IPSO algorithm. Finally, the MATLAB simulation results comparing different algorithms validated the proposed high-order sliding mode neural network adaptive control and high-order sliding mode neural network with IPSO control and confirmed their superiority over SMC.

Author Contributions: J.Z. is responsible for algorithm writing, testing, and thesis writing. Y.Y. and Z.L. are responsible for obtaining key data. W.L. and L.M. are participants of the study. W.M. is the project leader. All authors have read and agreed to the published version of the manuscript.

Funding: This study was financially supported by the National Key R&D Program of China for Robot Project (grant number 2018YFB1308700), the Shanxi Provincial Key Core Technologies and Common Technology Special Projects (grant number 2020XXX009), and the Science and Technology Project of the Shanxi Provincial Department of Transportation (grant number 2019-01-09).

Data Availability Statement: Data are contained within the article.

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

Nomenclature

q_i	Rotation angle at manipulator joint i
K	System kinetic energy
P	System's potential
$M(q)$	Mass matrix
$G(q)$	Gravity matrix

$V(q, \dot{q})$	Coriolis force and centrifugal force matrix
τ	Control input matrix
τ_d	Uncertain external factors
S	Sliding mode surface function
u_{nom}	Homogeneous continuous control term
u_{ast}	Super-twisting function
α	Super-twisting I function control gain
β	Super-twisting function control gain
h_j	Output of the Gauss function.
W	Weights of the intermediate layer
$V_i(\xi)$	Lyapunov stability function
x_i	Position of any particle i in the swarm
v_i	Velocity of any particle i in the swarm
p_i	Best position experienced by the particles in the swarm
p_g	Best position experienced by the swarm in the swarm
ω	Inertial weight
c_1	Acceleration constant usually called the cognitive learning factor
c_2	Acceleration constant usually called the social learning factor

References

- Lopez-Franco, C.; Diaz, D.; Hernandez-Barragan, J.; Arana-Daniel, N.; Lopez-Franco, M. A Metaheuristic Optimization Approach for Trajectory Tracking of Robot Manipulators. *Mathematics* **2022**, *10*, 1051. [\[CrossRef\]](#)
- Zhang, X.; Xiao, F.; Tong, X.; Yun, J.; Liu, Y.; Sun, Y.; Tao, B.; Kong, J.; Xu, M.; Chen, B. Time Optimal Trajectory Planning Based on Improved Sparrow Search Algorithm. *Front. Bioeng. Biotechnol.* **2022**, *10*, 14. [\[CrossRef\]](#) [\[PubMed\]](#)
- Xiao, F.; Li, G.; Jiang, D.; Xie, Y.; Yun, J.; Liu, Y.; Li, H. An effective and unified method to derive the inverse kinematics formulas of general six-DOF manipulator with simple geometry. *Mech. Mach. Theory* **2021**, *14*, 104265. [\[CrossRef\]](#)
- Li, G.; Xiao, F.; Zhang, X.; Tao, B.; Jiang, G. An inverse kinematics method for robots after geometric parameters compensation. *Mech. Mach. Theory* **2022**, *174*, 104903. [\[CrossRef\]](#)
- Dong, J.; Chen, L. The nonlinear integral sliding mode of RBF neural network algorithm is used to control the motion trajectory error of the manipulator. *Chin. J. Constr. Mach.* **2018**, *16*, 106–110. [\[CrossRef\]](#)
- Khan, S.G.; Herrmann, G.; Pipe, T.; Melhuish, C.; Spiers, A. Safe adaptive compliance control of a humanoid robotic arm with anti-windup compensation and posture control. *Int. J. Soc. Robot.* **2010**, *2*, 305–319. [\[CrossRef\]](#)
- Rahmani, M.; Komijani, H.; Rahman, M.H. New sliding mode control of 2-DOF robot manipulator based on extended grey wolf optimizer. *Int. J. Control. Autom. Syst.* **2020**, *18*, 1572–1580. [\[CrossRef\]](#)
- Asl, R.M.; Pourabdollah, E.; Salmani, M. Optimal fractional order PID for a robotic manipulator using colliding bodies design. *Soft Comput.* **2018**, *22*, 4647–4659. [\[CrossRef\]](#)
- Li, Z.; Yin, Y.; Zhang, J.; Qi, C. Disturbance observer control of the multi-joint manipulator based on the backstepping sliding mode's neural network. *J. Mach. Des.* **2021**, *38*, 126–131. [\[CrossRef\]](#)
- Wang, T. Research on obstacle avoidance of wheeled mobile robot based on wavelet neural network and fuzzy sliding mode control. *Chin. J. Constr. Mach.* **2020**, *18*, 278–282. [\[CrossRef\]](#)
- Noordin, A.; Basri, M.A.M.; Mohamed, Z.; Lazim, I.M. Adaptive PID controller using sliding mode control approaches for quadrotor UAV attitude and position stabilization. *Arab. J. Sci. Eng.* **2021**, *46*, 963–981. [\[CrossRef\]](#)
- Zhang, J.; Li, Z.; Yin, Y.; Qi, C.; Wu, K. High-order sliding mode neural network adaptive control of multi-joint manipulator. *Mech. Sci. Technol. Aerosp. Eng.* **2021**, *40*, 710–715. [\[CrossRef\]](#)
- Kali, Y.; Saad, M.; Benjelloun, K.; Khairallah, C. Super-twisting algorithm with time delay estimation for uncertain robot manipulators. *Nonlinear Dyn.* **2018**, *93*, 557–569. [\[CrossRef\]](#)
- Pai, M.-C. Dynamic output feedback RBF neural network sliding mode control for robust tracking and model following. *Nonlinear Dyn.* **2015**, *79*, 1023–1033. [\[CrossRef\]](#)
- Zhang, C.; Zhang, Z. Application of adaptive robust control of the manipulator trajectory tracking. *Modul. Mach. Tool Autom. Manuf.* **2019**, 86–89+93. [\[CrossRef\]](#)
- Liu, Y.; Chen, J. Research on trajectory error of mechanical arm based on radial basis function neural network control. *Mach. Tool Hydraul.* **2018**, *46*, 105–108.
- Liao, L.; Yang, Y. Adaptive radial basis function neural network bi-quadratic functional optimal control for manipulators. *Control. Theory Appl.* **2020**, *37*, 47–58.
- Liu, Y.; Jiang, D.; Yun, J.; Sun, Y.; Li, C.; Jiang, G.; Kong, J.; Tao, B.; Fang, Z. Self-Tuning Control of Manipulator Positioning Based on Fuzzy PID and PSO Algorithm. *Front. Bioeng. Biotechnol.* **2022**, *9*, 817723. [\[CrossRef\]](#)
- Liang, H.; Mi, G. An improved reaching law of sliding mode control for robotic manipulator based on disturbance observer. *Meas. Control. Technol.* **2019**, *38*, 140–144. [\[CrossRef\]](#)

20. Yang, Y.; Wang, S. High-order sliding mode interference observer based on space flexible manipulator for trajectory tracking. *Transducer Microsyst. Technol.* **2019**, *38*, 114–117+128. [[CrossRef](#)]
21. Sun, F.; Sun, Z. Stable Adaptive Controller Design for Manipulators Using Neural Networks. *Control. Theory Appl.* **1997**, *14*, 809–816.
22. Sun, F. Stable Adaptive Control of Robot Manipulators Using Neural Networks. Ph.D. Thesis, Tsinghua University, Beijing, China, 1997.
23. Niccolai, A.; Bettini, L.; Zich, R. Optimization of electric vehicles charging station deployment by means of evolutionary algorithms. *Int. J. Intell. Syst.* **2021**, *36*, 5359–5383. [[CrossRef](#)]
24. Faculty of Sciences and Techniques of Sidi Bouzid, U.o.K., Tunisia; Dhahri, H. Biogeography-Based Optimization for Weight Optimization in Elman Neural Network Compared with Meta-Heuristics Methods. *Brain Broad Res. Artif. Intell. Neurosci.* **2020**, *11*, 82–103. [[CrossRef](#)]
25. Parque, V.; Miyashita, T. Smooth Curve Fitting of Mobile Robot Trajectories Using Differential Evolution. *IEEE Access* **2020**, *8*, 82855–82866. [[CrossRef](#)]
26. Liu, T.; Yin, S. An improved particle swarm optimization algorithm used for BP neural network and multimedia course-ware evaluation. *Multimed. Tools Appl.* **2017**, *76*, 11961–11974. [[CrossRef](#)]
27. Wang, J.; Yang, H.; Hu, X.; Wang, X. An adaline neural network-based multi-user detector improved by particle swarm optimization in CDMA systems. *Wirel. Pers. Commun.* **2011**, *59*, 191–203. [[CrossRef](#)]
28. Zhang, H.; Yuan, X. An improved particle swarm algorithm to optimize PID neural network for pressure control strategy of managed pressure drilling. *Neural Comput. Appl.* **2020**, *32*, 1581–1592. [[CrossRef](#)]
29. Ye, H.-T.; Li, Z.-Q. PID neural network decoupling control based on hybrid particle swarm optimization and differential evolution. *Int. J. Autom. Comput.* **2020**, *17*, 867–872. [[CrossRef](#)]
30. Yu, J.; Xi, L.; Wang, S. An improved particle swarm optimization for evolving feedforward artificial neural networks. *Neural Process. Lett.* **2007**, *26*, 217–231. [[CrossRef](#)]
31. Yun, J.; Sun, Y.; Li, C.; Jiang, D.; Tao, B.; Li, G.; Liu, Y.; Chen, B.; Tong, X.; Xu, M. Self-adjusting force/bit blending control based on quantitative factor-scale factor fuzzy-PID bit control. *Alex. Eng. J.* **2022**, *61*, 4389–4397. [[CrossRef](#)]
32. Rahmani, M.; Komijani, H.; Ghanbari, A.; Etefagh, M.M. Optimal novel super-twisting PID sliding mode control of a MEMS gyroscope based on multi-objective bat algorithm. *Microsyst. Technol.* **2018**, *24*, 2835–2846. [[CrossRef](#)]
33. Jung, S. Stability analysis of reference compensation technique for controlling robot manipulators by neural network. *Int. J. Control. Autom. Syst.* **2017**, *15*, 952–958. [[CrossRef](#)]