Article

Non-Singular Fast Terminal Sliding Mode Control of High-Speed Train Network System Based on Improved Particle Swarm Optimization Algorithm

Xiangyu Kong 1 and Tong Zhang 2,*

1 College of Electrical and Information Engineering, Dalian Jiaotong University, Dalian 116028, China; kxy1430130309@163.com
2 College of Locomotive and Rolling Stock Engineering, Dalian Jiaotong University, Dalian 116028, China
* Correspondence: zhang_tong66@126.com

Received: 31 December 2019; Accepted: 15 January 2020; Published: 1 February 2020

Abstract: This paper proposes a non-singular fast terminal sliding mode control strategy based on the self-organizing radial basis function neural network (RBFNN) approximation for the train key network system to realize the safe and reliable operation of the train. In order to improve the RBFNN approximation performance and speed, an improved multi-strategy particle swarm optimization (IMPSO) algorithm, which utilizes multi-strategy evolution ways with a nonlinear deceasing inertia weight to improve the global optimizing performance of particle swarm, is proposed to optimize the structure and parameters for better mapping the highly nonlinear characteristics of train traction braking. In addition, the IMPSO is also introduced into a non-singular fast terminal sliding mode (NFTSM) controller to obtain the most appropriate tuning parameters of the controller and suppresses the chattering phenomenon from sliding mode controller. The stability characteristic of the system under the proposed NFTSM controller is studied based on the Lyapunov theory. Further combined with effective delay prediction and delay compensation methods, the NFTSM high-precision control of the train key nonlinear network system is implemented. The simulation results show that the proposed method has more efficient and robust tracking performance and real-time performance compared with other control methods, which can provide effective means for realizing the symmetrical bus control by automatic train operation (ATO) at both ends of the train, with the safe operation of the train under every complex motion condition.

Keywords: train; particle swarm optimization; RBF neural network; terminal sliding mode control; delay

1. Introduction

Modern high-speed trains are using the train communication network (TCN) to realize train control and diagnosis. However, due to the existence of a large number of subsystems, the conversion of communication protocol and the changeable operating environment of the train network, the time delay is inevitably generated in the process of information transmission. If the time delay is too large, not only the stability of the control system will be affected, but also the safety of the passengers will be threatened. Therefore, it is critical to control the network delay in the train network systems [1].

TCN works in a particularly complex environment, and the traction and braking key systems of high-speed trains have obvious nonlinear characteristics, which bring difficulties to the research of train network control system [2]. In recent years, some scholars have studied the scheduling algorithm of the train network and the delay problem of composite Ethernet [3,4]. However, there are still few reports on the delay and control aspects of TCN networks.
At present, there are many nonlinear system control schemes and approaches proposed for reference. For the neural network control [5–7], He et al. [5] adopted the adaptive neural network to handle a robotic system with full-state constraints uncertainties and disturbances and employed the Moore–Penrose inverse term to prevent the violation of the full-state constraints, which guaranteed the control performance of the system. Wang et al. [6] considered the finite-time quantized feedback control for the first time compared with the existing researches, and a novel adaptive neural output-feedback control strategy was designed by proposing a new finite-time stability criterion and designing a state observer, which effectively solved the finite-time quantized feedback control problem for nonlinear functions. Hu et al. [7] proposed a neural network model composed of three networks for reinforcement learning to control a robotic manipulator with unknown parameters and dead zones, and the state of the robotic manipulator was predicted by the state network of the model, the action policy was learned by the action network, and the performance index of the action policy was estimated by a critic network. Further, the effectiveness and the stability of the model was demonstrated by applying on a simulated two-link robotic manipulator. In addition, dynamic optimization design of a neural network is an effective way to ensure that a neural network always works under the optimization of structure and combination parameters [8]. Therefore, the self-organizing learning method that simultaneously optimizes network structure and parameters has received widespread attention [9–12]. The optimization methods mainly include the genetic algorithm [9], the differential evolutionary algorithm [10], and the particle swarm optimization (PSO) algorithm [11,12]. Among them, the PSO algorithm is widely used in neural network learning due to its advantages of fast convergence speed and high optimization efficiency. Han et al. [11] designed a nonlinear regression function to adjust the inertia weight of PSO to enhance the searching ability of particles, and designed a unified criterion to obtain the parameters and network size of RBF neural network. Dong et al. [12] designed time-varying learning factors to balance the exploration and utilization ability of particles in the search space. Meanwhile, the local best information was added to the evolutionary process of particles for better population diversity. Further, the improved PSO algorithm was applied to train the RBF neural network for determining optimal network parameters, which effectively improved network generalization capability.

Among these control approaches, due to the intrinsic insensitivity of the sliding mode (SM) control to parameter variations and external disturbances, the SM control has been widely used in the designs of some train control systems [13–15]. Li et al. [13] designed a SM adaptive robust controller for automatic train operation (ATO) of high-speed trains with using the robust control to reduce all uncertainties of parameter variation caused by model errors, and achieved high precision tracking ability under different wind speeds. Li et al. [14] designed the distributed neural network SM control strategy to control the speed tracking of high-speed trains according to the dispersion characteristics of train model power or braking force input, and used the historical operation data of the train to compensate the current control law output to improve the control accuracy and practical stability. However, the conventional SM control suffers an obvious drawback, which is asymptotic error convergence of the closed loop system, due to the use of linear sliding surfaces [15]. Therefore, Venkataraman et al. [16] proposed a terminal sliding mode (TSM) control method with limited time, which had been widely concerned. However, TSM control in general has two limitations, one is the singular problem in the control process, and another is that the convergence rate of the system state is slow when the system state is far from the equilibrium point. In order to solve the singularity problem in the control process, Feng et al. [17] proposed a non-singular terminal sliding mode (NTSM) control method and verified the effectiveness of the method. On the other hand, to solve the problem of convergence rate, Yu et al. [18] proposed a fast terminal sliding mode (FTSM) control method, which could achieve faster convergence even when the system state was far away from the equilibrium point. Combining the advantages of the two control methods, the concept of the non-singular fast terminal sliding mode (NFTSM) control is proposed [19]. Followed along [19], Gao et al. [20] proposed a fast-integral terminal sliding mode control method for a cascaded nonlinear dynamical system with
mismatched uncertainties, which not only avoided the singularity in the traditional terminal sliding mode, but also addressed the mismatched problems in the nonlinear control system. Yang et al. [21] proposed a time-specified nonsingular terminal sliding manifold consisting of pre-specified nonlinear functions for solving the problem of trajectory tracking for robotic airships, which can avoid the singularity problem and specify the convergence time of terminal sliding mode control. Yao et al. [22] proposed a novel nonlinear nonsingular terminal sliding manifold by considering that the unknown parameters needed to be estimated via the non-negative adaptive law. Meanwhile, the corresponding novel robust adaptive NTSM control strategy was designed to enable that the position tracking error and the velocity tracking error of the ATO system converge to zero finite time and eliminate the singularity caused by terminal sliding mode controller.

However, the nonlinear controlled objects of the above literatures are in some ways different from the train network control system, such as [14,15,22] were not concerned intensively with the train complex operating environment and the train communication network, which are the critical factors for the operation of the train. Thus, a suitable NFTSM control methodology guaranteeing robustness and other performances to the above factors must be developed to incorporate in the high-speed train network control system.

In this paper, a NFTSM control method based on the improved multi-strategy particle swarm optimization (IMPSO) algorithm and the RBF neural network (RBFNN) is proposed for the key nonlinear network control system. In order to improve the control accuracy, the multi-strategy evolution ways with a nonlinear deceasing inertia weight are adopted to improve global optimization characteristics of particle swarm, for quickly obtaining the most appropriate the tuning parameters of NFTSM controller with the elimination of sliding mode control chattering phenomenon. In addition, the IMPSO algorithm is also introduced into RBFNN to optimize the structure and parameters for better mapping the highly nonlinear characteristics of train traction braking, which ensures the safe and reliable operation of the train network control system, combined with effective delay prediction and delay compensation methods.

The rest of the paper is structured as follows: In Section 2, we design the NFTSM control strategy for a high-speed train network control system with train information transmission analysis. In Section 3, we build the multi point-mass with single-coordinate dynamic model, which describes the motion process of high-speed trains. In Section 4, we use the IMPSO algorithm with a nonlinear deceasing inertia weight to optimize the structure and parameters of RBFNN, so that the network better maps the highly nonlinear characteristics in the process of train traction braking. In Section 5, we design the NFTSM control law with proofing stability control system, and describe the flowchart of the proposed control strategy. In Section 6, we analyze the real-time performance of IMPSO-RBFNN and discuss the performance of the proposed control method compared with other control methods. Finally, conclusions and prospects are presented in Section 7.

In order to make the proposed method easier to describe and understand, we summarize the notation needed in our formulation in Appendix A.

2. The Design of the NFTSM Control Strategy for the High-Speed Train Network Control System

The operation environment of high-speed trains is complex and changeable, and there are many factors that affect the operation performance. Accurate control of the operation of trains is required to ensure smooth tracking under the desired speed of the train. The NFTSM control method has the advantages of high control accuracy, fast convergence speed, and strong robustness on the basis of ensuring the non-singular control. In order to ensure the safe and reliable operation of high-speed trains, we design a NFTSM network control system for high-speed trains, combining the optimization capability of the IMPSO algorithm and the approximation capability of RBFNN, as shown in Figure 1.
Figure 1. Improved network control system of high-speed train based on Improved multi-strategy particle swarm optimization RBF neural network (IMPSO-RBFNN).

Figure 1 shows that all train information is transmitted through a multifunction vehicle bus (MVB). ATO is the core component of information transmission and processing calculation. ATO should not only capture the effective control amount in time under the guidance of the designed control algorithm, but also reasonably arrange the control amount transmission sequence according to the accurate delay prediction method. Thus, the ATO sends the calculated control input \( u(k - \tau_{ca}) \) and forward channel timestamp \( T_{ca} \) as a process data packet to the train traction braking system, which executes the latest control law and records the actual forward channel delay \( \tau_{ca} \). The sensor periodically obtains the output \( x(k), v(k) \) according to the set sampling time and sends \( x(k), v(k), \) timestamp \( T_{sc} \), and \( \tau_{ca} \) as a process data packet back to ATO. Then, ATO will analyze the actual output of the current time through the timestamp, which combines with the current reference input signal \( (x_d(k), v_d(k)) \) to calculate the control input of the next time, so as to realize the accurate tracking under the desired speed on the basis that the auto-regressive (AR) model of the least mean square (LMS) algorithm accurately predicts the forward delay. Due to inconsistencies in the clocks of each subservice, clock correction is required at the same time the timestamp is received. If \( \tau_{ca} \) is the forward channel delay measured at different times, and \( T \) is the sampling period, then it is defined as:

\[
\tau_{ca} = \tau_{ca}/T.
\]  

(1)

3. High-Speed Train Motion Model

The high-speed train motion model is a complex nonlinear system composed of multiple operating conditions. Therefore, it is extremely vital that the accurate motion model is built for the safe and reliable operation of the train. In order to fully reflect the changing laws of trains under different motion conditions, the multi-point-mass with single-coordinate dynamic model [23] is adopted to describe the motion process of high-speed trains as:

\[
\begin{align*}
\dot{v}(k) &= \eta F(k) \\
F(k) &= u(k) - f(k) - f_l(k) \\
f(k) &= f_0(v(k)) + f_1(k) + f_2(k) + f_l(k)
\end{align*}
\]  

(2)

where \( v(k) \) is the speed of the reference vehicle, \( \eta \) is the acceleration coefficient, \( F(k) \) is the net force for the whole train, \( u(k) \) is the control force for the whole train, \( f_l(k) \) is the mutual influence of other vehicles on the reference vehicle, and \( f(k) \) is the resistance for the whole train composed of general...
will introduce the IMPSO algorithm and RBFNN into the NFTSM controller and utilize the global ability and better maps the highly nonlinear characteristics in the process of train traction braking. In order to improve the learning efficiency of RBFNN, we use the IMPSO algorithm to optimize the structure and parameters of RBFNN so that the network has self-organizing learning ability and better maps the highly nonlinear characteristics in the process of train traction braking.

4. IMPSO-RBF Neural Network

RBFNN is widely used in the control field because of its advantages such as fast learning speed, strong approximation ability, and difficulty in falling into local extremum [25]. However, the structure and parameter design of RBFNN directly affect its generalization ability with the increasing complexity of solving problems. To guarantee the accuracy of the train running speed control, we will introduce the IMPSO algorithm and RBFNN into the NFTSM controller and utilize the global optimization characteristics of particle swarm to improve the approximation ability of RBFNN to the nonlinear disturbance in the process of train traction braking, and significantly reduce chattering caused by sliding mode control and the train running speed tracking error, which realizes the safe driving and parking precision.

4.1. RBF Neural Network

Generally, an RBFNN consists of three layers: The input layer, the hidden layer, and the output layer [25]. For the input \( x \), the network output is as follows:

\[
y = \sum_{k=1}^{K} a_k \phi_k = \sum_{k=1}^{K} a_k e^{-\frac{\|x - c_k\|^2}{\sigma_k^2}},
\]

where \( a_k \), \( \phi_k \), and \( \sigma_k \) are the resistive coefficients for the \( i^{th} \) vehicle, \( \theta_i(k) \) is the gradient angle of the rail for the \( i^{th} \) vehicle, \( A \) is the parameter obtained through test, \( R_i(k) \) is the radius of the curve for the \( i^{th} \) vehicle passes, \( L_{4i}(k) \) is the length of the tunnel for the \( i^{th} \) vehicle, \( \Delta x_{di}(k) \) is the elastic-shaped variable in which tension or compression occurs in the elastic coupler of the \( i^{th} \) vehicle, \( m_j \) is the mass of the \( j^{th} \) vehicle, and \( d \) is the train rotary mass coefficient. By combining (2) and (3), the motion model of high-speed train can be given:

\[
\dot{v}(k) + \eta \left( \sum_{i=1}^{n} \left( a_{0i}(k) + a_{1i}(k)v_i(k) + a_{2i}(k)v_i^2(k) \right) \right) + \sum_{i=1}^{n} 1000 \tan \theta_i(k) + \sum_{i=1}^{n} \frac{A}{R_i(k)} + \sum_{i=1}^{n} 0.00013L_{4i}(k) + \sum_{i=1}^{n-1} \Delta x_{di}(k) \sum_{j=i+1}^{n} m_j = \eta u(k),
\]

where \( a_{2i}(k)v_i^2(k) \) is the nonlinear function of speed, which increases with the increase of train speed. Therefore, the train traction braking system has obvious multiple operating conditions and nonlinear characteristics at high speed. To guarantee the accuracy of the train running speed control, we will introduce the IMPSO algorithm and RBFNN into the NFTSM controller and utilize the global optimization characteristics of particle swarm to improve the approximation ability of RBFNN to the nonlinear disturbance in the process of train traction braking, and significantly reduce chattering caused by sliding mode control and the train running speed tracking error, which realizes the safe driving and parking precision.

4.2. RBF Neural Network
where $\omega_k$ is the output weight between the $k^{th}$ hidden neuron and the output neuron, $\phi_k$ is the output of the $k^{th}$ hidden neuron, and $c_k$ and $b_k$ are the center vector and width of the $k^{th}$ hidden neuron respectively, which can be described as:

\[
\begin{align*}
    \omega_k(t+1) &= \omega_k(t) - \beta \frac{\partial E}{\partial \omega_k} + \delta \Delta \omega_k(t) \\
    c_k(t+1) &= c_k(t) - \beta \frac{\partial E}{\partial c_k} + \delta \Delta c_k(t) \\
    b_k(t+1) &= b_k(t) - \beta \frac{\partial E}{\partial b_k} + \delta \Delta b_k(t)
\end{align*}
\]

(6)

where $E$ is the error function, $\beta$ is the learning rate, and $\delta$ is the momentum factor.

4.2. Particle Swarm Optimization Algorithm

PSO, as an optimization algorithm that utilizes the information exchange between the particles and the swarm to search cooperatively to obtain the global optimal solution, has the advantages of fast convergence speed and high optimization efficiency, and has been widely applied to optimization problems [26].

Generally, a swarm consists of a number of particles. Then, each particle carries $d$ dimensional information and has the memory ability to record the best previous position of each particle $p_i^{(k)}$. Thus, the position $x_i^{(k+1)}$ and velocity $v_i^{(k+1)}$ of the $i^{th}$ particle in the $k+1^{th}$ iteration can be described as:

\[
\begin{align*}
    x_i^{(k+1)} &= x_i^{(k)} + v_i^{(k+1)} \\
    v_i^{(k+1)} &= w_i v_i^{(k)} + c_1 r\text{and}(p_i^{(k)} - x_i^{(k)}) + c_2 r\text{and}(p_g^{(k)} - x_i^{(k)})
\end{align*}
\]

(7)

where $i = 1, 2, \ldots, \text{Num}$ with Num being the swarm size, $x_i^{(k)}$ is the position of the $i^{th}$ particle in the $k^{th}$ iteration, $v_i^{(k)}$ is the corresponding velocity, $p_i^{(k)}$ is the best position of the $i^{th}$ particle in the $k^{th}$ iteration, $p_g^{(k)}$ is the best position obtained by the swarm in the $k^{th}$ iteration, $w$ is the inertia weight, $c_1$ and $c_2$ are the acceleration constants, and rand is the random value uniformly distributed in (0,1). The $d$ dimensional information carried by each particle consists of the parameters to be optimized in the RBFNN and the NFTSM controller. To optimize the structure of RBFNN, the position update formula in (6) is represented in a discrete form as:

\[
x_i^{(k+1)} = \begin{cases} 
0, r\text{and} \geq f(v_i^{(k+1)}) \\
1, r\text{and} < f(v_i^{(k+1)})
\end{cases}
\]

(8)

where $j = 1, 2, \ldots, d$ with $d$ being the dimensionality of the searching space, $x_{ij}^{(k+1)}$ is the $j^{th}$ position of the $i^{th}$ particle in the $k+1^{th}$ iteration and its value represents the number of neurons in the corresponding RBF neural network, and $v_{ij}^{(k+1)}$ is the corresponding velocity; the sigmoid function $f(v_{ij}^{(k+1)})$ can be described as:

\[
f(v_{ij}^{(k+1)}) = \frac{1}{1 + e^{-v_{ij}^{(k+1)}}}
\]

(9)

4.3. Improved Multi-Strategy Particle Swarm Optimization Algorithm

The PSO algorithm easily falls into local optimum for solving complex multimodal optimization problems [27]. Considering the complexity of the controlled object in this paper, the adaptive multi-strategy evolutionary behavior with a nonlinear deceasing inertia weight is adopted to adjust the particle optimization velocity. In the optimization process, the evolutionary behavior of the particles is determined by calculating the immediate value, the future value, and the comprehensive value...
of different strategies, and the strategy behavioral mutation algorithm was proposed to avoid the algorithm falling into the local optimal solution.

4.3.1. Improved Multi-Strategy Evolutionary Behavior

The evolutionary behavior of learning based on individual optimal and globally optimal individual is as follows [27]:

\[
\begin{align*}
    v_i^{(k+1)} &= w(k)v_i^{(k)} + \text{rand}(p_i^{(k)} - x_i^{(k)}) + \text{rand}(p_g^{(k)} - x_i^{(k)}) \\
    w(k) &= w_{\text{max}} - \frac{k}{k_{\text{max}}}(w_{\text{max}} - w_{\text{min}})
\end{align*}
\]

(10)

where \(w_{\text{max}}\) is the maximum inertia weight, \(w_{\text{min}}\) is minimum inertia weight, \(k\) is the current iteration times, and \(k_{\text{max}}\) is the total number of iterations. It is noted that if the inertia weight \(w(k)\) is large enough, the algorithm has good global searching ability.

However, when \(w(k)\) is relatively small, the algorithm has good local searching performance [28]. Thus, the performance of PSO could be improved by setting the value of \(w(k)\) appropriately for a tradeoff between its local and global searching abilities. According to the problem in approach [27] that the linear deceasing \(w(k)\) in (10) easily leads to the algorithm falling into the local extremum at the beginning of the iteration process, we propose the strategy of the nonlinear deceasing \(w(k)\) for avoiding the algorithm falling into the local optimal solution to some extent, which can be described as:

\[
    w(k) = (w_{\text{max}} - w_{\text{min}}) \tan(0.875(1 - \frac{k}{k_{\text{max}}})^{0.5}) + w_{\text{min}},
\]

(11)

where the decline rate of \(w(k)\) is relatively quick in the searching process, which is conducive to the particle swarm to explore the optimal solution at a relatively small velocity at the end of search and effectively avoids the algorithm falling into the local extremum.

The evolutionary behavior of learning based on group average knowledge is as follows [27]:

\[
    v_i^{(k+1)} = w(k)v_i^{(k)} + \text{rand}\left(\sum_{i=1}^{\text{Num}} x_i^{(k)}\right) / \text{Num} - x_i^{(k)}).
\]

(12)

The evolutionary behavior of learning based on global optimal individual and other individuals’ knowledge is as follows [27]:

\[
    v_i^{(k+1)} = p_g^{(k)} + \text{rand}(x_{r_1}^{(k)} - x_i^{(k)}),
\]

(13)

where \(r_1\) and \(r_2\) are different integers in \([1, \text{Num}], r_1 \neq r_2\).

The evolutionary behavior of learning based on other individuals’ knowledge is as follows [27]:

\[
    v_i^{(k+1)} = x_{r_1}^{(k)} + \text{rand}(x_{r_2}^{(k)} - x_i^{(k)}),
\]

(14)

where \(r_1, r_2,\) and \(r_3\) are different integers in \([1, \text{Num}], r_1 \neq r_2 \neq r_3\).

4.3.2. Multi-Strategy Value Comparison

In the process of individual evolution, the evolutionary strategy will directly affect the convergence of optimal algorithm, and the evolutionary strategy of the individual is determined by the selection
probability of each strategy with calculating the immediate value, the future value, and the comprehensive value of each strategy in this paper, which can be described as [27]:

\[
\begin{align*}
value - I_i(k) &= \frac{f(x_{i(k-1)}) - \min(f(x_{i(k-1)}, f(x_i)))}{f(x_{i(k-1)})} \\
value - F_i(k) &= \frac{N^p_i(k)}{M^p_i(k)} + \frac{C_0 \log(k)}{N^p_i(k)} + (1 - \mu)\left(\frac{N^p_i(k)}{M^p_i(k)} + \frac{\log(\text{Num}_k)}{\sum_{i=1}^{M} N^p_i(k)}\right), \\
value_i(k) &= value - I_i(k) + value - F_i(k) + P_i(k), 1 \leq i \leq M \\
P_i(k+1) &= P_{\min} + \frac{value_i(k)}{\sum_{i=1}^{M} value_i(k)} (1 - MP_{\min})
\end{align*}
\]

where \(value - I_i(k), value - F_i(k), \) and \(value_i(k)\) are the immediate value, the future value, and the comprehensive value of each evolutionary strategy, respectively; \(f(x_i, k)\) is the fitness in the \(k^{th}\) iteration; \(f(x_i, k-1)\) is the fitness in the previous iteration; \(N^p_i(k)\) and \(M^p_i(k)\) are the number of success and total number of executions of the \(i^{th}\) strategy used by individual before the \(k^{th}\) iteration, respectively; \(N^p_i(k)\) and \(M^p_i(k)\) are the number of success and total number of execution of the \(i^{th}\) strategy used by all the individual before the \(k^{th}\) iteration, respectively; \(\mu\) and \(C_0\) are the constant; \(P_i(k)\) is the probability of each strategy adopted in the \(k^{th}\) iteration; \(P_{\min}\) is the minimum selection probability of each strategy; and \(M\) is the total number of strategy and \(0 < P_{\min} < 1/M\). The comprehensive value will comprehensively evaluate the influence of different strategies on the global optimal solution, which accelerates the convergence process.

4.3.3. Strategy Behavioral Mutation Algorithm

In order to avoid the algorithm falling into the local optimal solution, when the same strategy is always used in the evolutionary process and the optimization effect is poor, the mutation probability formula of strategy [27] should be used to adjust the selection probability of each evolutionary strategy.

\[
\sigma^2 = \frac{\sum_{i=k-n+1}^{k} (f_i - \bar{f})^2}{n},
\]

where \(n\) is the number of fitness before the \(k^{th}\) iteration, \(f_i\) is the fitness of the \(i^{th}\) iteration, and \(\bar{f}\) is the mean value of \(n\) fitness. If \(\sigma^2\) is less than a certain threshold \(\lambda\), the selection probability of the strategy is set as \(P_{\min}\).

5. Non-Singular Fast Terminal Sliding Mode Control

5.1. Control Law Design

Supposing that \(x_d(k) \in \mathbb{R}^n\) and \(v_d(k) \in \mathbb{R}^n\) are the desired position and speed of the train, respectively. Then, the position tracking error and the speed tracking error can be defined as:

\[
\begin{align*}
e_s(k) &= x_d(k) - x(k) \\
e_v(k) &= v_d(k) - v(k)
\end{align*}
\]

Then, combining (2)–(4), the error system can be described as:

\[
\begin{align*}
\dot{e}_s(k) &= e_v(k) \\
\dot{e}_v(k) &= \frac{d}{\eta} [u(k - \xi_{ca}) - f_0(v(k)) - d_f(k)]
\end{align*}
\]
where \( d_f(k) = f_r(k) + f_c(k) + f_t(k) + f_i(k) \) represents the additional disturbance received by the train during operation.

In order to ensure that the (x(k), v(k)) converges to the equilibrium point within finite-time and avoid the singular problem, the FTSM surface [29] is designed as:

\[
s(k) = e_c(k) + a_1\text{sign}^{\gamma_1}e_c(k) + a_2\text{sign}^{\gamma_2}e_c(k),
\]

(19)

where \( a_1 = \text{diag}[a_{11}, a_{12}, \ldots, a_{1n}] \) and \( a_2 = \text{diag}[a_{21}, a_{22}, \ldots, a_{2n}] \) are positive diagonal matrices, and \( \text{sign}^{\gamma}e_c(k) = [\text{sign}(e_{c1}(k))\varepsilon_{c1}(k)]^\gamma, \ldots, \text{sign}(e_{cn}(k))\varepsilon_{cn}(k)]^\gamma \) for \( i = 1, 2, \gamma_1 \geq 1 \) and \( 0 < \gamma_2 < 1 \).

Comparing with the conventional TSM, the FTSM surface has faster convergence capability attributed to the additional nonlinear term \( a_1\text{sign}^{\gamma_1}e_c(k) \).

Differentiating (19) with respect to time, yields

\[
\dot{s}(k) = \dot{e}_c(k) + a_1\gamma_1|e_c(k)|^{\gamma_1-1}e_c(k) + a_2\gamma_2|e_c(k)|^{\gamma_2-1}e_c(k).
\]

(20)

Taking use of (18) to substitute for \( \dot{e}_c(k) \), (20) can be rewritten as:

\[
\eta^{-1}\dot{s}(k) = \eta^{-1}\dot{v}_d(k) - [u(k - \tau_{ca}) - f_0(v(k)) - d_f(k)]
+ \eta^{-1}a_1\gamma_1|e_c(k)|^{\gamma_1-1}e_c(k) + \eta^{-1}a_2\gamma_2|e_c(k)|^{\gamma_2-1}e_c(k).
\]

(21)

Then, we introduce the auxiliary variable \( v_r(k) = s(k) + v(k) \), such that its time derivative \( \dot{v}_r(k) = \dot{s}(k) + \dot{v}(k) \). In terms (17) and (18), we have

\[
\begin{cases}
\dot{v}_r(k) = v_d(k) + a_1\gamma_1|e_c(k)|^{\gamma_1-1}e_c(k) + a_2\gamma_2|e_c(k)|^{\gamma_2-1}e_c(k) \\
\dot{v}_r(k) = \dot{v}_d(k) + a_1\gamma_1|e_c(k)|^{\gamma_1-1}e_c(k) + a_2\gamma_2|e_c(k)|^{\gamma_2-1}e_c(k)
\end{cases}
\]

(22)

By employing (22), we can rewrite (21) as:

\[
\eta^{-1}\dot{s}(k) = \eta^{-1}\dot{v}_r(k) - [u(k - \tau_{ca}) - \omega\phi(v(k)) - d_f(k)] + \epsilon,
\]

(23)

where \( \omega\phi(v(k)) \) is the estimates of general resistance of train operation \( f_0(v(k)) \), and \( \epsilon \) is the reconstruction error of the RBFNN.

**Assumption 1.** The uncertain item \( d_f(k) + \epsilon \) has an upper bound as follows: \( ||d_f(k) + \epsilon|| \leq \rho \), where \( \rho \) is the known positive constant [22,29].

In order to realize the fast tracking of a desired speed in a finite-time, in terms of the FTSM controller design procedure in approach [29], the control low with the estimates of RBFNN can be designed as:

\[
u(k - \tau_{ca}) = \eta^{-1}\dot{v}_r(k) + \omega\phi(v(k)) + K_1s(k) + K_2\text{sign}(s(k)),
\]

(24)

where \( K_1 \in \mathbb{R}^{m \times n} \) and \( K_2 \in \mathbb{R}^{m \times n} \) are positive diagonal matrices to be designed. \( K_1s(k) \) is to relieve the possible chattering phenomena and the control term \( K_2\text{sign}(s(k)) \) is employed to suppress the external disturbance. In order to further improve the control effect, the above parameters are optimized by the IMPSO algorithm, which effectively enhances the effectiveness and robustness of the control algorithm.

However, regarding the controller in (24), we can find out that some singular phenomena occur when \( e_c(k) \rightarrow 0 \), due to the existence of singular item \( a_2\gamma_2|e_c(k)|^{\gamma_2-1}e_c(k) \) in \( \dot{v}_r(k) \). With the aim to settle the singular problem, referring to the approach [29], we design the saturation function to limit the singular item \( \dot{v}_r(k) \):

\[
sat(\dot{v}_r(k)) = \begin{cases}
\dot{v}_r(k), & |\dot{v}_r(k)| \leq \omega \\
\omega\text{sign}(\dot{v}_r(k)), & |\dot{v}_r(k)| > \omega
\end{cases}
\]

(25)
where $\dot{s}(k) = \alpha_2 \gamma_2 |e_s(k)|^{\gamma - 1} e_s(k)$, and $\omega$ is the positive constant to be designed. As a consequence, the control law in (24) can be rewritten as:

$$u(k - \tau_{ca}) = \eta^{-1} s(k) + \omega \phi(v(k)) + K_1 s(k) + K_2 \text{sign}(s(k)), \quad (26)$$

where $\dot{s}(k) = \dot{s}(k) + \text{sat}(\dot{s}(k))$ through defining $\theta_s(k) = \dot{s}(k) + \alpha_1 |e_s(k)|^{\gamma - 1} e_s(k)$.

### 5.2. Stability Analysis

**Lemma 1** [30]. Consider a nonlinear system $\dot{x} = f(x)$, $f(0) = 0$, $x \in \mathbb{R}^n$, $x(0) = 0$, whereby the equilibrium point $x = 0$ is built on the assumption that, as a continuous function, $V(x) : D \rightarrow \mathbb{R}$ is defined on an open neighborhood $U \subseteq D$ of the origin, such that the following conditions hold: (1) $V(x)$ is positive definite; (2) there are real numbers $c > 0$ and $\alpha \in (0, 1)$, such that $\dot{V}(x) + c V(x) \leq 0$, $x \in U \setminus \{0\}$

Then, the nonlinear system is locally finite-time stable. Furthermore, the given time $T(x)$ satisfies $T(x_0) \leq \frac{V(x_0)}{c(1-\alpha)}$ for all $x_0$ in a number of open neighborhoods of the origin. What is more, if $D = \mathbb{R}^n$ and $V(x)$ is unbounded, the system is globally finite-time stable.

**Lemma 2** [31]. Consider a vector $b = [b_1, b_2, \ldots, b_n]$, the following inequality holds: $\|b\| \leq \sum_{i=1}^{n} |b_i|$

**Theorem 1.** Consider the network control system of high-speed train in (4). Then, the position and the speed tracking errors will converge to 0 in the finite-time, if the FTSM surface is given by (19), the control law is given by (26).

**Proof.** Select the following Lyapunov function candidate:

$$V(k) = \frac{1}{2} s^T(k) \eta^{-1} s(k). \quad (27)$$

The time derivative of $V(k)$ is derived as:

$$\dot{V}(k) = s^T(k) \eta^{-1} \dot{s}(k). \quad (28)$$

Using (23), we obtain

$$\dot{V}(k) = s^T(k) \eta^{-1} \dot{s}(k) - [u(k - \tau_{ca}) - \omega \phi(v(k)) - d_f(k)] + \varepsilon. \quad (29)$$

By substituting for $u(k - \tau_{ca})$ from (26) into (29), we can obtain that

$$\dot{V}(k) = -s^T(k) K_1 s(k) - s^T(k) K_2 \text{sign}(s(k)) + s^T(k) d_f(k) + \varepsilon. \quad (30)$$

Next, (30) can be simplified to

$$\dot{V}(k) = -s^T(k) K_1 s(k) - s^T(k) K_2 \text{sign}(s(k)) + s^T(k) d_f(k) + \varepsilon \leq -s^T(k) K_1 s(k) - \lambda_{\min}(K_2) \|s(k)\| + \|d_f(k)\| + \varepsilon \|s(k)\|. \quad (31)$$

Considering the Assumption 1, we choose the $K_1 > 0$ and the gains $K_2$ to satisfy that $\lambda_{\min}(K_2)$ is a little larger than the upper bound $\rho$, such that

$$\dot{V}(k) \leq -s^T(k) K_1 s(k) \leq 0, \quad (32)$$

where (32) indicates that $s(k)$ is bounded in accordance with Lyapunov theorem. In addition, considering that $s(k)$ is defined with error vectors $e_s(k)$ and $e_v(k)$, we can conclude that both $e_s(k)$ and
where $L$ is the train stopping time. Furthermore, $x_d(k)$ and $v_d(k)$ are set to be bounded such that $x(k)$ and $v(k)$ are bounded as well.

To demonstrate the finite-time stability of the network control system, we choose the $K_1 > 0$ and utilize Lemma 2, (31) can be rewritten as:

$$
\dot{V}(k) \leq -s^T(k)K_2 s(k) + s^T(k)(d_f(k) + \epsilon)
$$

$$
\leq -\sum_{i=1}^{n} K_2|s_i(k)| + \sum_{i=1}^{n} \rho|s_i(k)|
$$

$$
\leq -[\min(K_2) - \rho]\sum_{i=1}^{n} |s_i(k)|
$$

$$
\leq -[\min(K_2) - \rho]\sqrt{\frac{2}{m_2}}\sqrt{\sum_{i=1}^{n} |s_i(k)|^2}
$$

$$
\leq -[\min(K_2) - \rho]\sqrt{\frac{2}{m_2}}V(k)^{1/2}
$$

By setting $c = [\min(K_2) - \rho]\sqrt{\frac{2}{m_2}}, \alpha = 1/2$, and making use of Lemma 1, it is clear that the network control system is stable within finite-time when $(e_x(k), e_v(k))$ are in $G$. For the convenience of proving, we separate the state space $(e_x(k), e_v(k))^T \in \mathbb{R}^2$ into two opposite areas, as follows:

$$
G = \left\{ (e_x(k), e_v(k)) : |\delta_x(k)| = a_2\gamma_2|e_x(k)|^{\gamma_2-1}|e_v(k)| \leq \omega \right\}
$$

$$
Q = \left\{ (e_x(k), e_v(k)) : |\delta_x(k)| = a_2\gamma_2|e_x(k)|^{\gamma_2-1}|e_v(k)| > \omega \right\}.
$$

If the error states $(e_x(k), e_v(k))$ are in $Q$, there are two different situations: $e_v(k) > 0; e_v(k) < 0$. According to (18), we can conclude that when $e_v(k) > 0$, $e_x(k)$ will increase monotonically until getting and passing the boundary between $G$ and $Q$ along a one-way path, while monotonically decreasing in the same way when $e_v(k) < 0$.

On the basis of the analysis as mentioned above, the error states $(e_x(k), e_v(k))$ will pass from $Q$ to $G$ in the finite-time instead of staying in $G$ all the time. Once the error states $(e_x(k), e_v(k))$ reach $G$, the network control system will satisfy the condition of the finite-time stability given by (33). Therefore, whether the error states $(e_x(k), e_v(k))$ are in $Q$ or $G$, the position and the speed tracking errors will converge to the sliding mode surface $s(k) = 0$ and reach 0 in the finite-time. The proof is completed. □

5.3. Controller Preprocessing

The most important step in our control strategy is the controller preprocessing. Thus, the flowchart of the proposed control strategy is as shown in Figure 2.

In this strategy, the IMPSO algorithm is adopted to optimize the structure and parameters of RBFNN for better mapping the highly nonlinear characteristics of train traction braking. In addition, it is also introduced into the NFTSM controller to obtain the tuning parameters of the controller and suppresses the chattering phenomenon from the sliding mode controller. In the flowchart of the proposed control strategy, the fitness function is described as:

$$
g(x) = \sum_{k=1}^{L} |e_d(k) - v(k)|,
$$

where $L$ is the train stopping time.
According to the actual position and speed of the train obtained by the simulation sensor, the latest traction braking force values are calculated and then assigned to the traction braking simulation unit. The IMPSO algorithm, RBFNN algorithm, and NFTSM control algorithm are implanted into it. According to the actual position and speed of the train obtained by the simulation sensor, the latest traction braking force values are calculated and then assigned to the traction braking simulation unit to perform corresponding actions, where the control input multistage saturation limit is described as:

$$u(k) = \begin{cases} 
0.6u(k) + 1000, & u(k) > 4000 \\
0.8u(k) + 200, & 800 \leq u(k) \leq 4000 \\
8u(k), & -800 < u(k) < 800 \\
0.8u(k) - 200, & -4000 \leq u(k) \leq -800 \\
0.6u(k) - 1000, & u(k) < -4000 
\end{cases} \quad (36)$$

6. Simulation and Analysis

In order to verify the effectiveness of the above methods, we select the CRH3 (eight-section group) train [32] as the controlled object and the velocity range is 0 to 350 km/h. The desired speed curve includes all operating conditions of traction, braking, and inertia. The main parameters of CRH3 are shown in Table 1. The simulation ATO controller model is established in the central control unit (CCU). The IMPSO algorithm, RBFNN algorithm, and NFTSM control algorithm are implanted into it. According to the actual position and speed of the train obtained by the simulation sensor, the latest traction braking force values are calculated and then assigned to the traction braking simulation unit to perform corresponding actions, where the control input multistage saturation limit is described as:

$$u(k) = \begin{cases} 
0.6u(k) + 1000, & u(k) > 4000 \\
0.8u(k) + 200, & 800 \leq u(k) \leq 4000 \\
8u(k), & -800 < u(k) < 800 \\
0.8u(k) - 200, & -4000 \leq u(k) \leq -800 \\
0.6u(k) - 1000, & u(k) < -4000 
\end{cases} \quad (36)$$

**Table 1.** The main parameters of CRH3.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total train weight (ton)</td>
<td>400</td>
</tr>
<tr>
<td>Maximum operating speed (km/h)</td>
<td>350</td>
</tr>
<tr>
<td>Sustained operating speed (km/h)</td>
<td>300</td>
</tr>
<tr>
<td>Rotary mass coefficient</td>
<td>$d = 0.06$</td>
</tr>
<tr>
<td>Unit general resistance (N/KN)</td>
<td>$f_0(v(k)) = 0.53 + 0.0039v(k) + 0.00014v^2(k)$</td>
</tr>
</tbody>
</table>

**Figure 2.** Flowchart of the proposed control strategy.
The task execution period of each device is configured to be 50 ms, and all the modules of the data receiving and sending, algorithm, and model are executed in the same task to achieve synchronous calculation. In practical application, the characteristic period of each port is usually selected as a multiple of 64 ms. Therefore, the sampling period of 64 ms is selected here.

After the simulation test, the parameters of the IMPSO algorithm can be set as follows [27,33]: The total number of iterations $k_{\text{max}}$ is 80, the swarm size $\text{Num}$ is 80, the maximum inertia weight $w_{\text{max}}$ is 0.9, the minimum inertia weight $w_{\text{min}}$ is 0.4, $P_{\text{min}}$ is 0.1, $C_{0}$ is 2, $\mu$ is 0.5, $n$ is 4, and $\lambda$ is 1. The control system parameters optimized by the fitness function are shown in Table 2.

### Table 2. Control system parameters.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m_i$</td>
<td>50 + $\Delta m_i$, $\Delta m_i \in [-7, 7]$</td>
<td>ton</td>
</tr>
<tr>
<td>$\theta_i$</td>
<td>$\theta_i \in [0.9, 3]$</td>
<td>$\circ$</td>
</tr>
<tr>
<td>$a_{0i}, a_{1i}, a_{2i}$</td>
<td>$a_{1i} \in [0.0038, 0.00393]$, $a_{2i} \in [0.000112, 0.0001156]$</td>
<td>-</td>
</tr>
<tr>
<td>$R_i$</td>
<td>$R_i \in [2, 7.2]$</td>
<td>km</td>
</tr>
<tr>
<td>$l_{i1}$</td>
<td>$l_{i1} \in [0.6]$</td>
<td>km</td>
</tr>
<tr>
<td>$\Delta x_{i1}$</td>
<td>$\Delta x_{i1} \in [1.1, 5]$</td>
<td>mm</td>
</tr>
<tr>
<td>$A$</td>
<td>600</td>
<td>-</td>
</tr>
<tr>
<td>$K_1, K_2$</td>
<td>$K_1 = 30.85$, $K_2 = 31.56$</td>
<td>-</td>
</tr>
<tr>
<td>$\gamma_1, \gamma_2$</td>
<td>$\gamma_1 = 1.06$, $\gamma_2 = 0.06$</td>
<td>-</td>
</tr>
<tr>
<td>$a_{1i}, a_{2i}$</td>
<td>$a_{1i} = 0.81$, $a_{2i} = -2.31$</td>
<td>-</td>
</tr>
<tr>
<td>$\omega$</td>
<td>0.45</td>
<td>-</td>
</tr>
<tr>
<td>$\beta$</td>
<td>0.05</td>
<td>-</td>
</tr>
<tr>
<td>$\delta$</td>
<td>0.35</td>
<td>-</td>
</tr>
<tr>
<td>neurons</td>
<td>13</td>
<td>-</td>
</tr>
</tbody>
</table>

#### 6.1. Real-Time Performance Analysis of the IMPSO-RBFNN

In order to verify the real-time performance of RBFNN after IMPSO optimization, the above algorithm is simulated in CCU, which works with the Intel(R) Core (TM) i5-7300HQ CPU@2.50 GHz processor and 8.00 GB memory, the real-time performance of IMPSO-RBFNN is shown in Table 3.

### Table 3. Real-time performance of RBFNN.

<table>
<thead>
<tr>
<th>Neurons</th>
<th>Approximation Error</th>
<th>Training Time/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.9962</td>
<td>$1.7044 \times 10^{-5}$</td>
</tr>
<tr>
<td>6</td>
<td>0.5955</td>
<td>$2.7368 \times 10^{-5}$</td>
</tr>
<tr>
<td>9</td>
<td>0.5128</td>
<td>$2.8476 \times 10^{-5}$</td>
</tr>
<tr>
<td>12</td>
<td>0.4171</td>
<td>$3.0462 \times 10^{-5}$</td>
</tr>
<tr>
<td>13</td>
<td>0.3580</td>
<td>$3.2155 \times 10^{-5}$</td>
</tr>
<tr>
<td>14</td>
<td>0.3862</td>
<td>$3.3030 \times 10^{-5}$</td>
</tr>
<tr>
<td>17</td>
<td>0.4019</td>
<td>$3.7213 \times 10^{-5}$</td>
</tr>
</tbody>
</table>

Table 3 shows that the training time slowly increases with the increase of hidden neurons. However, the approximation error changes obviously with the change of hidden neurons. Specially, the real-time performance of the RBFNN is the best when the number of neurons is 13, and its average training time is far less than the task period (50 ms), thus as long as the sampling time is chosen larger than 50 ms, on-line real-time control could be realized.

#### 6.2. Delay Compensation Effect of Different Characteristic Periods

In order to analyze the delay compensation effect of different characteristic periods, the experiments compare the output effect of the proposed method with and without delay compensation. Setting the
initial value of the control input and the speed output as 0. When the characteristic period is 64 ms, the speed response curve of tracking sine wave is shown in Figure 3a. Figure 3b is the characteristic period of 128 ms.

![Figure 3a](image1.png)

![Figure 3b](image2.png)

**Figure 3.** Comparison of delay compensation effect of different characteristic periods: (a) Characteristic period 64 ms, (b) characteristic period 128 ms.

Figure 3 shows that our method can quickly and accurately track the speed response curve of various operating modes when time delay compensation is added; there is almost no-overshooting and the traction and braking effects are very ideal. Moreover, when the characteristic period is set as 128 ms, the forward and feedback channel delay continue to increase, adding up to more than 300 ms. If the delay control method is not added, the output will occasionally oscillate at different time points and become unstable. The reason for the oscillation is that the delay varies randomly and
the time when the control quantity arrives at the actuator is not fixed, which leads to the failure of timely and effective control at many moments. The larger the characteristic period, the larger the delay and the more serious the oscillation. Therefore, it is very important to adopt effective methods for compensating the influence of delay, which is the key of fast and stable control of traction braking.

6.3. Compared with Other Control Methods

In order to verify the effectiveness and robustness of the proposed method under different reference signals, the RBFNN adaptive control method [5] with \( c_1 = -0.2, \beta = 3, \gamma = -0.005, G = 50000, \) and \( \epsilon_f = -2; \) the NFTSM control method [29] with \( K_1 = 1, K_2 = 1, \gamma_1 = 1.2, \gamma_2 = 0.8, \alpha_1 = 0.5, \) and \( \alpha_2 = 5, \) and \( \omega = 0.02; \) and the sliding mode adaptive robust (SMAR) \( H_\infty \) control method [13] with \( c = 5.5, k_s = 0.8, \) and \( \eta = 0.8 \) are selected to be compared with the proposed method. The speed tracking effect of various control methods when the reference signal is a sine wave is shown in Figure 4. To simulate the real operating condition of the train, the control performances of various methods when the reference signal is a variety of operating modes were analyzed, as shown in Figure 5. In addition, the tracking error statistics are shown in Table 4.

![Figure 4. Speed tracking under sine wave.](image)

Figure 4 shows that, under the frequently changing sine wave tracking trajectory, our method still maintains the advantages of accurate tracking and fast response, which has better control performances compared with other control methods. Figure 5 shows that the desired speed trajectory includes all operating conditions of traction, braking, and inertia. Compared with the RBFNN, NFTSM, and SMAR control methods, our method has better control performances in the whole acceleration and braking phase, and can achieve the smooth switching at different steady-state operating points, which fully meets the high precision tracking requirements of high-speed trains. Thus, it can be obviously seen that the NFTSM control method based on IMPSO-RBFNN optimization has a fast response speed and a good dynamic effect on the train nonlinear network control system, and can be applied to the complex nonlinearity-controlled object, which has ideal robustness and stability. Meanwhile, this method is simple in structure and easy to realize in train communication engineering.

Table 4 shows that, compared with the RBFNN, NFTSM, and SMAR control methods, our method has smaller speed and position tracking error, which has greater advantages in the control process.
Thus, it can be obviously seen that the NFTSM control method based on IMPSO-RBFNN optimization has a fast response speed and a good dynamic effect on the train nonlinear network control system, and can be applied to the complex nonlinearity-controlled object, which has ideal robustness and stability. Meanwhile, this method is simple in structure and easy to realize in train communication engineering.

(a) 

(b) 

Figure 5. Cont.
6.4. Discussion

Our method utilizes the improved RBF neural network to obtain the real-time and accurate nonlinear characteristics of the train operation, and the control input is also adjusted quickly and effectively with this characteristic. In addition, owing to the introduction of nonlinear item on the sliding surface, the proposed control method can improve the finite-time convergence rate of system states. Meanwhile, the singular problem is settled via introducing a saturation function into the control input and the external disturbances are suppressed by designing robust control items. Thus, this control scheme can be applied to the complex nonlinearity-controlled object, which has ideal robustness and stability. Further, this method is simple in structure and easy to realize in train communication engineering.

In this paper, the TCN forward delay is compensated, and the actual impact of packet loss on the control system is not further considered. Experiments show that data transmission of MVB has a certain degree of packet loss, it needs to propose effective methods to suppress the effects of time delay and packet loss simultaneously. In addition, the influence of the strong wind and rainfall environment on the aerodynamic performance of the high-speed train was not studied. Thus, considering the above impact of multi-factors on control system, we will design a more efficient and robust train network control method for specific circumstances in the future.
7. Conclusions and Prospects

In this paper, we propose a non-singular fast terminal sliding mode control strategy for the train network key control system. The strategy utilizes RBF neural networks approximation and particle swarm optimization to design the NFTSM controller. The structure and parameters of RBFNN can be optimized and the ideal controller parameters of NFTSM can be obtained by an improved multi-strategy particle swarm optimization algorithm proposed in this paper, which not only approximates the nonlinear process of traction and braking quickly and accurately but also suppresses the chattering phenomenon of the controller effectively. The simulation results show that, compared with other control methods, the proposed method can quickly track the changes of reference signals and has better real-time performance, robustness, and stability. The research of this paper provides a reference for realizing the symmetrical bus control by ATO at both ends of the train and the optimal control of the train communication network, which has potential application prospects in further enhancing the economy, safety, and reliability of high-speed train operation.

Author Contributions: X.K. suggested the idea of the work and performed the experiments. T.Z. designed the experimental method. Both authors wrote and critically revised the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Natural Science Foundation of Liaoning Province, grant number 20180551003

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

Abbreviations

The following abbreviations are used in this manuscript:

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBFNN</td>
<td>RBF Neural Network</td>
</tr>
<tr>
<td>IMPSO</td>
<td>Improved Multi-strategy Particle Swarm Optimization</td>
</tr>
<tr>
<td>NFTSM</td>
<td>Non-Singular Fast Terminal Sliding Mode</td>
</tr>
<tr>
<td>ATO</td>
<td>Automatic Train Operation</td>
</tr>
<tr>
<td>MVB</td>
<td>Multifunction Vehicle Bus</td>
</tr>
<tr>
<td>TCN</td>
<td>Train Communication Network</td>
</tr>
<tr>
<td>SM</td>
<td>Sliding Mode</td>
</tr>
<tr>
<td>PSO</td>
<td>Particle Swarm Optimization</td>
</tr>
<tr>
<td>TSM</td>
<td>Terminal Sliding Mode</td>
</tr>
<tr>
<td>NTSM</td>
<td>Non-Singular Terminal Sliding Mode</td>
</tr>
<tr>
<td>FTSM</td>
<td>Fast Terminal Sliding Mode</td>
</tr>
<tr>
<td>NFTSM</td>
<td>Non-Singular Fast Terminal Sliding Mode</td>
</tr>
<tr>
<td>AR</td>
<td>Auto Regressive</td>
</tr>
<tr>
<td>LMS</td>
<td>Least Mean Square</td>
</tr>
<tr>
<td>CCU</td>
<td>Central Control Unit</td>
</tr>
<tr>
<td>SMAR</td>
<td>Sliding Mode Adaptive Robust</td>
</tr>
</tbody>
</table>

Appendix A

In this appendix, we list the symbols of notations in this paper, as shown in Table A1.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(v(k))</td>
<td>speed of the train</td>
</tr>
<tr>
<td>(x(k))</td>
<td>position of the train</td>
</tr>
<tr>
<td>(v_d(k))</td>
<td>the desired speed of the train</td>
</tr>
<tr>
<td>(x_d(k))</td>
<td>the desired position of the train</td>
</tr>
<tr>
<td>(F(k))</td>
<td>the net force for the train</td>
</tr>
<tr>
<td>(u(k))</td>
<td>the control force for the train</td>
</tr>
<tr>
<td>Notation</td>
<td>Meaning</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
</tr>
<tr>
<td>$\eta_d$</td>
<td>acceleration coefficient</td>
</tr>
<tr>
<td>$f_l(k)$</td>
<td>the mutual influence of other vehicles on the reference vehicle</td>
</tr>
<tr>
<td>$f(k)$</td>
<td>the resistance for the train</td>
</tr>
<tr>
<td>$f_0(v(k))$</td>
<td>general resistance</td>
</tr>
<tr>
<td>$f_r(k)$</td>
<td>ramp resistance</td>
</tr>
<tr>
<td>$f_c(k)$</td>
<td>curve resistance</td>
</tr>
<tr>
<td>$f_t(k)$</td>
<td>tunnel resistance</td>
</tr>
<tr>
<td>$a_0(k), a_1(k)$ and $a_2(k)$</td>
<td>the resistive coefficients for the $i^{th}$ vehicle</td>
</tr>
<tr>
<td>$\theta_i(k)$</td>
<td>the gradient angle of the rail for the $i^{th}$ vehicle</td>
</tr>
<tr>
<td>$A$</td>
<td>the parameter obtained through test</td>
</tr>
<tr>
<td>$R_i(k)$</td>
<td>the radius of the curve for the $i^{th}$ vehicle passes</td>
</tr>
<tr>
<td>$L_i(k)$</td>
<td>the length of the tunnel for the $i^{th}$ vehicle</td>
</tr>
<tr>
<td>$\Delta x_i(k)$</td>
<td>the shaped variable in the elastic coupler of the $i^{th}$ vehicle</td>
</tr>
<tr>
<td>$m_j$</td>
<td>the mass of the $i^{th}$ vehicle</td>
</tr>
<tr>
<td>$a_{0i}(k), a_{1i}(k)$ and $a_{2i}(k)$</td>
<td>the resistive coefficients for the $i^{th}$ vehicle</td>
</tr>
<tr>
<td>$\phi_k$</td>
<td>the center vector of the $k^{th}$ hidden neuron</td>
</tr>
<tr>
<td>$b_k$</td>
<td>the width of the $k^{th}$ hidden neuron</td>
</tr>
<tr>
<td>$E$</td>
<td>the error function</td>
</tr>
<tr>
<td>$\beta$</td>
<td>the learning rate</td>
</tr>
<tr>
<td>$\delta$</td>
<td>the momentum factor</td>
</tr>
<tr>
<td>$Num$</td>
<td>the swarm size</td>
</tr>
<tr>
<td>$x_i(k)$</td>
<td>the position of the $i^{th}$ particle in the $k^{th}$ iteration</td>
</tr>
<tr>
<td>$v_i(k)$</td>
<td>the velocity of the $i^{th}$ particle in the $k^{th}$ iteration</td>
</tr>
<tr>
<td>$p_i(k)$</td>
<td>the best position of the $i^{th}$ particle in the $k^{th}$ iteration</td>
</tr>
<tr>
<td>$p_{ig}(k)$</td>
<td>the best position obtained by the swarm in the $k^{th}$ iteration</td>
</tr>
<tr>
<td>$w$</td>
<td>the inertia weight</td>
</tr>
<tr>
<td>$c_1$ and $c_2$</td>
<td>the acceleration constants</td>
</tr>
<tr>
<td>$rand$</td>
<td>the random value uniformly distributed in $(0,1)$</td>
</tr>
<tr>
<td>$f(c_i^{(k+1)})$</td>
<td>the sigmoid function</td>
</tr>
<tr>
<td>$w_{\text{max}}$</td>
<td>the maximum inertia weight</td>
</tr>
<tr>
<td>$w_{\text{min}}$</td>
<td>the minimum inertia weight</td>
</tr>
<tr>
<td>$k_{\text{max}}$</td>
<td>the total number of iterations</td>
</tr>
<tr>
<td>$r_1, r_2$ and $r_3$</td>
<td>the different integers in $[1, \text{Num}]$</td>
</tr>
<tr>
<td>value–$I_i(k)$</td>
<td>the immediate value</td>
</tr>
<tr>
<td>value–$F_i(k)$</td>
<td>the future value</td>
</tr>
<tr>
<td>value–$f(x_i, k)$</td>
<td>the comprehensive value</td>
</tr>
<tr>
<td>$f(x_i, k)$</td>
<td>the fitness in the $k^{th}$ iteration</td>
</tr>
<tr>
<td>$N_{i}^{p}(k)$</td>
<td>the number of success of the $i^{th}$ strategy used by individual before the $k^{th}$ iteration</td>
</tr>
<tr>
<td>$M_{i}^{p}(k)$</td>
<td>the total number of executions of the $i^{th}$ strategy used by individual before the $k^{th}$ iteration</td>
</tr>
<tr>
<td>$N_{i}^{n}(k)$</td>
<td>the number of success of the $i^{th}$ strategy used by all the individual before the $k^{th}$ iteration</td>
</tr>
<tr>
<td>$M_{i}^{n}(k)$</td>
<td>the total number of executions of the $i^{th}$ strategy used by all the individual before the $k^{th}$ iteration</td>
</tr>
<tr>
<td>$\mu$</td>
<td>the constant</td>
</tr>
<tr>
<td>$C_0$</td>
<td>the constant</td>
</tr>
<tr>
<td>$P_{p}(k)$</td>
<td>the probability of each strategy adopted in the $k^{th}$ iteration</td>
</tr>
<tr>
<td>$P_{\text{min}}$</td>
<td>the minimum selection probability of each strategy</td>
</tr>
<tr>
<td>$M$</td>
<td>the total number of strategy</td>
</tr>
<tr>
<td>$f$</td>
<td>the mean value of $n$ fitness</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>the threshold</td>
</tr>
<tr>
<td>$T_{ca}$</td>
<td>forward channel timestamp</td>
</tr>
<tr>
<td>$\tau_{ca}$</td>
<td>forward channel delay</td>
</tr>
<tr>
<td>$T_{sc}$</td>
<td>timestamp</td>
</tr>
</tbody>
</table>
Table A1. Cont.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>sampling period</td>
</tr>
<tr>
<td>$e_x(k)$</td>
<td>position tracking error</td>
</tr>
<tr>
<td>$e_v(k)$</td>
<td>speed tracking error</td>
</tr>
<tr>
<td>$d_f(k)$</td>
<td>additional disturbance</td>
</tr>
<tr>
<td>s(k)</td>
<td>FTSM surface</td>
</tr>
<tr>
<td>$a_1$ and $a_2$</td>
<td>positive diagonal matrices</td>
</tr>
<tr>
<td>$\gamma_1$ and $\gamma_2$</td>
<td>the constant</td>
</tr>
<tr>
<td>$v_r(k)$</td>
<td>auxiliary variable</td>
</tr>
<tr>
<td>$\epsilon$</td>
<td>the reconstruction error of the RBFNN</td>
</tr>
<tr>
<td>$\rho$</td>
<td>known positive constant</td>
</tr>
<tr>
<td>$K_1$ and $K_2$</td>
<td>positive diagonal matrices</td>
</tr>
<tr>
<td>$\delta_1(k)$</td>
<td>the singular item</td>
</tr>
<tr>
<td>$\omega$</td>
<td>positive constant</td>
</tr>
<tr>
<td>$V(k)$</td>
<td>Lyapunov function</td>
</tr>
<tr>
<td>g(x)</td>
<td>fitness function</td>
</tr>
<tr>
<td>L</td>
<td>the train stop time</td>
</tr>
</tbody>
</table>

References


© 2020 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 (http://creativecommons.org/licenses/by/4.0/).