Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 6913579, 19 pages
https://doi.org/10.1155/2020/6913579

Research Article

Improved Generalized Predictive Control for High-Speed Train Network Systems Based on EMD-AQPSO-LS-SVM Time Delay Prediction Model

Xiangyu Kong[1] **and Tong Zhang**[2]

[1]*College of Electrical and Information Engineering, Dalian Jiaotong University, Dalian 116028, China*
[2]*College of Locomotive and Rolling Stock Engineering, Dalian Jiaotong University, Dalian 116028, China*

Correspondence should be addressed to Tong Zhang; zhang_tong66@126.com

Received 19 August 2020; Revised 17 November 2020; Accepted 26 November 2020; Published 14 December 2020

Academic Editor: Georgios I. Giannopoulos

Various control signals of high-speed trains (HSTs) are transmitted through the train communication network. However, the time delay generated during the transmission will cause a significant threat to the stability and safe operation of the train. To overcome the effect of time delay on the train control system, based on empirical mode decomposition (EMD) and adaptive quantum particle swarm optimization (AQPSO) algorithms, a least squares support vector machine (LS-SVM) time delay prediction model is proposed in this paper. The EMD algorithm is used to decompose the time delay sequence into several subsequences, which emphasizes the different local characteristics of the time delay sequence. By improving the calculation method about the successful value of particle iteration, an AQPSO algorithm with adaptive contraction-expansion coefficient is designed to optimize the parameters of different LS-SVM models for predicting each time delay component, which improves the prediction accuracy of network delay. Further, based on actor-critic reinforcement learning algorithm, an improved generalized predictive control method is proposed for the train network system. The actor-critic network is used to predict the future output of the system, and the recursive least squares identification algorithm with the variable forgetting factor is adopted to identify the future system model parameters. Combined with the time delay predicted accurately, the control quantity is sent in advance according to the properly arranged time series, which compensates efficiently the influence of the time delay on the control system. Simulation results show that compared with other control methods, the proposed method has better robustness and stability, which ensures the safe operation of high-speed trains under various working conditions.

1. Introduction

At present, HSTs and urban track vehicles all use the train communication network (TCN) to realize train control and fault diagnosis [1]. All kinds of control signals are transmitted to the corresponding actuator through the wired train bus and the multifunction vehicle bus (MVB). However, the time delay caused by various reasons in the process of information transmission will seriously affect the safety and stability of the train control system [2]. In addition to the end-to-end time delay in the TCN, there is also the time delay generated by signal processing and control logic judgment, etc. If the time delay is too long, it will greatly affect the stability of the control system [3].

The network environment of HSTs is complex, and key systems such as traction and braking have obvious nonlinear characteristics [4]. In order to suppress the adverse effect of time delay on the control performance, it is necessary to test and study the real network characteristics of TCN, which realizes real-time and stable control according to the actual nonlinear characteristics of the train key control system. In recent years, some scholars have studied the scheduling algorithm of train network and the time delay problem of composite Ethernet [5, 6], but there are still few reports on the time delay control of TCN.

Controller design and time delay prediction are two main problems to be solved in network-controlled systems. Based on controller design, researchers have closely combined network control with sliding mode control [7], neural network control [8], H_∞ control [9], and other related theories to conduct extensive research, which provides a variety of solutions for networked control of nonlinear systems. Li et al. [7] studied the tracking control problem of networked control systems of intelligent vehicle with external disturbance and network-induced delay, and a high-order sliding mode controller was designed to reduce the effect of external disturbance, and a state observer was used to compensate the time delay disturbance in the network. Xu et al. [8] proposed a robust adaptive neural network control method to compensate the uncertainty and network delay disturbance of the system, which solves the remote-control problem of ship course with uncertain time delay. Chen et al. [9] designed an H_∞ sampling controller by constructing the Lyapunov–Krasovskii functional with delay, which realizes the networked control of asynchronous traction motor.

For the time delay prediction, some scholars use the regression model prediction method to accurately model the time delay sequence samples [10, 11]. However, the process of solving the model parameters is too complex, so it is not suitable for the case with a large range of network delay fluctuation. With the introduction of compensation devices in the feedback loop, Smith predictor can eliminate the adverse effect of network delay on the control system [12, 13]. But in practical application, when the parameters of the controlled system are unstable or disturbed, the Smith prediction model will get out of control, which leads to the control effect getting worse and even oscillating. Due to its strong nonlinear identification ability and fast operation speed, neural network can use the past data to predict the future state of the system, which realizes the time delay prediction and compensation [14, 15]. Nevertheless, the neural network prediction method is easy to fall into local extremum and relies too much on the autocorrelation coefficient of the input time delay sequence. Support vector machine (SVM) has unique advantages in dealing with nonlinear, small sample, and high-dimensional spatial recognition problems and has stronger generalization ability than neural network, which is suitable for network delay prediction with strong nonlinear characteristics [16]. Suykens and Vandewalle [17] proposed the LS-SVM algorithm to make up for the disadvantages of SVM, such as long computing time and large computing amount. Under the condition of equality constraint, the problem of convex quadratic programming in SVM was transformed into solving systems of linear equations, which greatly reduces the training time of the model. Considering the uncertainty and nonlinearity of the time delay, Tian et al. [18] introduced EMD into the LS-SVM time delay prediction model, and the time delay sequence was decomposed into several eigenmode functions for classification prediction, which reduces the modelling complexity. Since the kernel function parameters of LS-SVM have a great impact on the learning and generalization ability of the model and it is difficult to determine them uniformly, Tian et al. [19] used genetic algorithm for offline optimization of LS-SVM kernel function parameters, which effectively improves the prediction accuracy of time delay.

However, the network application environment and nonlinear controlled object of the above method are completely different from the train network control system, so it is difficult to be applied to the high-speed train network system with high real-time performance requirements. As a model-based advanced control method, generalized predictive control (GPC) has the advantages of predictive model, control optimization, cyclic rolling, keeping output variable stable, etc., which is widely used for tracking control of complex nonlinear systems such as HSTs [20], spacecraft [21], and underwater robots [22]. Li and Yan [20] designed a GPC fast algorithm based on extreme learning machine for HSTs to achieve the speed tracking control, and the extreme learning machine was used to study the parameter mapping relationship between system model and controller, which greatly reduces the computational burden of the algorithm. Chen et al. [21] proposed a GPC method with extended state observer to solve the tracking control problem of the spacecraft attitude, and according to the hyperbolic tangent function, the extended state observer was designed to estimate and compensate the uncertainties and unknown disturbances of the system, which achieves high-precision tracking of spacecraft attitude. Zhu et al. [22] designed an improved generalized predictive control (IGPC) method for the motion control of underwater robots, and the incremental proportion integration differentiation (PID) algorithm was used to optimize the generalized predictive controller in the initial stage, which improves the stability of the system.

In this paper, an IGPC method based on actor-critic reinforcement learning algorithm is proposed for the train key nonlinear network control system, and the EMD-AQPSO-LS-SVM time delay prediction model is introduced into the IGPC method for reducing the impact of time delay on control effect. The EMD algorithm is used to decompose the original time delay sequences into several intrinsic mode functions (IMFs), and by improving the particle iterative success value calculation method, an AQPSO algorithm with dynamic contraction-expansion coefficient is designed to optimize the parameters of different LS-SVM models, which improves the predictive accuracy of time delay component. Using the actor-critic reinforcement learning algorithm and the recursive least squares (RLS) method with variable forgetting factor to predict and identify the future parameters of the IGPC, respectively, the predictive controllers are designed for each real-time linear system. Combined with the accurate forward time delay prediction results, the output sequence of control signal is adjusted reasonably to compensate the influence of network delay on control performance. The effectiveness of the proposed method is verified by simulation experiments on TCN platform.

The rest of this paper is organized as follows. In Section 2, the high-speed train network control system model is introduced. In Section 3, the LS-SVM time delay prediction model is designed based on EMD and AQPSO algorithms, and the AQPSO algorithm with dynamic contraction-expansion coefficient is proposed to optimize the LS-SVM model parameters. In Section 4, the IGPC strategy for HSTs is designed based on reinforcement learning algorithm. Section 5 analyses and compares the performance of different time delay prediction models and verifies the real-time performance and effectiveness of the proposed method. Section 6 discusses the advantages and limitations of the proposed method and the future research direction. Section 7 concludes this paper.

2. High-Speed Train Network Control System

The general train network control system model can be described as follows:

$$y(k+1) = y(k) + \chi\left(u\left(k - \widehat{\tau}_{ca}\right) - f_0(y(k)) - d(k)\right), \quad (1)$$

where $y(k)$ is the speed of train, χ is the acceleration coefficient, $u(k-\widehat{\tau}_{ca})$ is the unit control force of train, $\widehat{\tau}_{ca} = (\tau_{ca}/T)$, τ_{ca} is the forward channel time delay, T is the sampling period, $d(k)$ is the unit additional resistance caused by the complicated operating environment such as wind, tunnel, and curve, and $f_0(y(k))$ is the unit general resistance of train, which can be described as [3]

$$f_0(y(k)) = \alpha_0(k) + \alpha_1(k)y(k) + \alpha_2(k)y^2(k), \quad (2)$$

where $\alpha_0(k)$ is the rolling mechanical resistance coefficient, $\alpha_1(k)$ is the other mechanical resistance coefficient, and $\alpha_2(k)$ is the external air resistance coefficient.

Combining (1) and (2), the system model can be rewritten as

$$\begin{aligned} y(k+1) = y(k) + \chi\big(u\left(k-\widehat{\tau}_{ca}\right) - \alpha_0(k) - \alpha_1(k)y(k) \\ - \alpha_2(k)y^2(k) - d(k)\big), \end{aligned} \quad (3)$$

where $\alpha_0(k)$, $\alpha_1(k)$, and $\alpha_2(k)$ with high uncertainty change constantly with the change of operating condition, which makes the train traction control system have obvious multiple working conditions and nonlinear characteristics [3]. In practical application, the high-speed train network system is composed of an automatic train operation (ATO) system, traction control system, and sensors. The network simulation system constructed in this paper is composed of two central control units (CCUs) and a human machine interface (HMI). CCU1 simulates the controller of the ATO system, CCU2 simulates the actuator of the traction control system, and HMI simulates the sensor. Each device communicates on the MVB network with process data [23]. The network system structure is shown in Figure 1.

Figure 1 shows that CCU1 simulates ATO function and sends control signal to CCU2 for realizing nonlinear control of the train traction system. CCU2 simulates the execution process of traction and braking and sends the completed information to HMI. HMI can measure the output signal of the system and send feedback to CCU1 to realize the whole network control process. In the communication process, the transmission time delay of MVB network includes two parts:

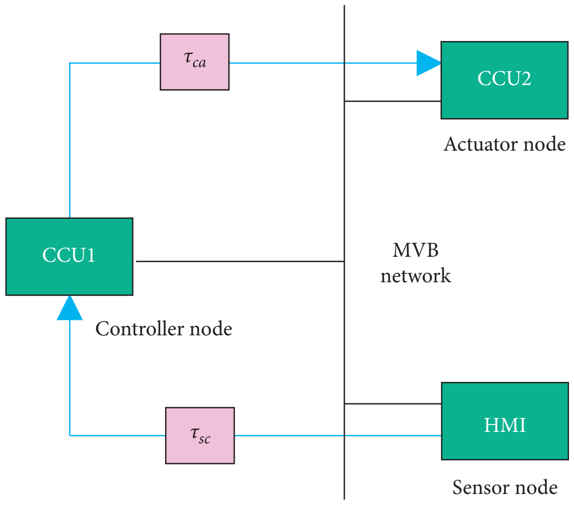


Figure 1: High-speed train network control system structure.

forward channel time delay τ_{ca} and feedback channel time delay τ_{sc}. Due to the large number of nodes and ports, irregular flow changes, uneven network load distribution, and other reasons, the information in the TCN is always in the dynamic and uncertain time-varying environment, which makes the analysis and design of the train network control system more complex and difficult [24]. With the improvement of the performance requirements of HSTs, it is necessary to obtain higher frequency system information, which will further increase the transmission delay of control signals. Therefore, there is considerable significance to adopt appropriate control methods to compensate and control the time delay generated in the TCN. The premise of controlling time delay is to master the characteristics of time delay transmission, which establishes the description or prediction model.

3. LS-SVM Time Delay Prediction Model Based on EMD and AQPSO Algorithms

Considering the randomness and nonlinearity of train network delay, the EMD algorithm is used to decompose the time delay sequence into several time delay components, which highlights the local characteristic signals with different time scales of the original data so that the modelling complexity is reduced. Meanwhile, the LS-SVM algorithm is used to build prediction models of different time delay components, and the predicted values are combined and superposed to obtain the final prediction results. For the drawbacks of LS-SVM algorithm in which parameters are difficult to be determined, an AQPSO algorithm with adaptive contraction-expansion coefficient is proposed to optimize model parameters offline, which effectively improves the prediction accuracy of network delay.

3.1. Time Delay Sequence Processing Based on EMD Algorithm. As a new signal processing method, EMD can adaptively decompose complex time series into several IMFs easy to model, which reduces the modelling difficulty. Thus, it is suitable for the analysis and processing of nonlinear and nonstationary time series [25]. In addition, the IMF has to meet the following two conditions: (1) for a column of data, the number of extreme points and zero crossings must be equal or at most slightly different; (2) at any point, the average value of the envelope formed by local maxima and minima is zero [26]. For the forward channel delay sequence $\tau_{ca}(t)$, the EMD algorithm is presented in Algorithm 1.

3.2. Time Delay Sequence Modelling Based on LS-SVM Algorithm. As a novel SVM algorithm, LS-SVM can transform the inequality constraint problem into the solution problem of linear matrix and has the advantages of simple operation, fast training speed, and strong generalization ability, which is widely used in the prediction modelling of complex data sequences [27]. Therefore, we adopt the LS-SVM algorithm to conduct training modelling for time-delay subsequences processed by EMD. For a given training set $U = \{(x_i, y_i) | x_i \in R^n, \quad y_i \in R, i = 1, 2, \ldots, l\}$, the LS-SVM model can be built as

$$y(x) = w^{\mathrm{T}} \varphi(x) + b, \tag{4}$$

where x is the input vector, y is the output vector, w is the weight vector, b is the offset vector, $\varphi(x)$ is the nonlinear mapping function, and $\varphi(x)$ can be used to map the input space to the high-dimensional feature space, which makes the nonlinear fitting problem in the input space become the linear fitting problem in the high-dimensional feature space. According to the criteria of structural risk minimization, the objective function can be described as [17]

$$\begin{cases} \min, \quad J(w,e) = \dfrac{1}{2} w^{\mathrm{T}} w + \dfrac{1}{2}\gamma \displaystyle\sum_{i}^{l} e_i^2, \quad \gamma > 0, \\ s.t. \quad y_i = w^{\mathrm{T}} \varphi(x_i) + b + e_i, \end{cases} \tag{5}$$

where e_i is the error of estimation and γ is the regularization parameter indicating how much to penalize the error function. To solve the above optimization problem, Lagrange multiplier is introduced into (5) to obtain

$$L(w,e,\lambda,b) = J(w,e) - \sum_{i=1}^{l} \lambda_i \left(w^{\mathrm{T}} \varphi(x_i) + b + e_i + y_i \right), \tag{6}$$

where λ_i is the Lagrange multiplier; taking partial derivatives of w, e, λ, and b, respectively, the following equations can be obtained:

$$\begin{cases} \dfrac{\partial L}{\partial w} = 0 \Longrightarrow w = \displaystyle\sum_{i=1}^{l} \lambda_i \varphi(x_i), \\ \dfrac{\partial L}{\partial e_i} = 0 \Longrightarrow \lambda_i = \gamma e_i, \\ \dfrac{\partial L}{\partial \lambda_i} = 0 \Longrightarrow w^{\mathrm{T}} \varphi(x_i) + b + e_i + y_i = 0, \\ \dfrac{\partial L}{\partial b} = 0 \Longrightarrow \displaystyle\sum_{i=1}^{l} \lambda_i = 0. \end{cases} \tag{7}$$

Eliminating w and e, the optimization problem is converted to solving the following linear equations:

$$\begin{bmatrix} 0 & I^{\mathrm{T}} \\ I & \Omega + \gamma^{-1} E \end{bmatrix} \begin{bmatrix} b \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix}, \tag{8}$$

where $y = [y_1, y_2, \ldots, y_l]^{\mathrm{T}}$, $\lambda = [\lambda_1, \lambda_2, \ldots, \lambda_l]^{\mathrm{T}}$, $I = [1, 1, \ldots, 1]^{\mathrm{T}}$, E is the $l \times l$ dimensional identity matrix, $\Omega_{ij} = K(x_i, x_j)$, $i, j = 1, 2, \ldots, l$, and $K(x_i, x_j)$ is the kernel function.

According to the given time delay data set U, using (8) to solve λ and b, the LS-SVM time delay prediction model can be obtained:

$$y(x) = \sum_{i=1}^{l} \lambda_i K(x, x_i) + b, \tag{9}$$

where $K(x, x_i)$ selects radial basis function (RBF) as the kernel function:

$$K(x, x_i) = \exp\left(\frac{-\| x - x_i \|^2}{2\sigma^2} \right), \tag{10}$$

where x_i is the centre vector and σ is the Gaussian kernel width.

It can be seen from the above modelling process that λ and σ determine the learning accuracy and generalization ability of the LS-SVM model. If λ is too small and σ is too large, the penalty degree of the error function will decrease and the kernel function will tend to 1, resulting in underfitting of the model. On the contrary, λ is too large and σ is too small. As a result, the generalization ability of the model becomes worse and the kernel function tends to 0, which makes the model overfit the samples. Thus, we propose an AQPSO algorithm to optimize λ and σ of the LS-SVM model, which not only improves the prediction accuracy but also enhances the generalization ability of the model for samples with different time delays.

```
     Initialization
     Determine all local extreme points of the original signal $\tau_{ca}(t)$.
     Procedure
(1)  repeat
(2)  Obtain the upper envelopment $e_1$ and lower envelopment $f_1$ by means of cubic spline fitting of all maximum and minimum
     points
(3)  Calculate the average envelopment as $m_1 = ((e_1 + f_1)/2)$
(4)  Set $h_1(t) = \tau_{ca}(t) - m_1$
(5)  until $h_1(t)$ satisfies the IMF conditions
(6)  Let $h_1(t) = c_1(t)$ and $c_1(t)$ is the first IMF of the $\tau_{ca}(t)$
(7)  Separate $c_1(t)$ from $\tau_{ca}(t)$
(8)  Let $r_1(t) = \tau_{ca}(t) - c_1(t)$
(9)  Let $r_1(t)$ be the original signal
(10) for $i = 2$: $n$ do
(11) repeat
(12) Obtain $e_i$ and $f_i$ by means of cubic spline fitting of all maximum and minimum points
(13) Calculate $m_i = ((e_i + f_i)/2)$
(14) Set $h_i(t) = \tau_{ca}(t) - m_i$
(15) until $h_i(t)$ satisfies the IMF conditions
(16) Let $h_i(t) = c_i(t)$
(17) Separate $c_i(t)$ from $\tau_{ca}(t)$
(18) Let $r_i(t) = \tau_{ca}(t) - c_i(t)$
(19) end for
(20) Obtain $n$ IMF components $c_i(t)$ and a margin $r_n(t)$ of the $\tau_{ca}(t)$
     Output: $\tau_{ca}(t) = \sum_{i=1}^{n} c_i(t) + r_n(t)$
     End Procedure
```

Algorithm 1: EMD algorithm for $\tau_{ca}(t)$.

3.3. Parameter Optimization Based on AQPSO Algorithms. As an extension of particle swarm optimization (PSO) in quantum space, quantum particle swarm optimization (QPSO) algorithm has the advantages of few control parameters, fast convergence speed, strong search ability, etc., which is suitable for the processing of complex optimization problems [28]. Based on the δ quantum well, QPSO algorithm can obtain the probability density function of particles at a certain point by solving the Schrodinger equation, and it uses the Monte Carlo method to obtain the evolution equation of particles [29]:

$$\begin{cases} p_i^{(t)} = \beta p\text{best}_i^{(t)} + (1-\beta) g\text{best}^{(t)}, \\ x_i^{(t+1)} = p_i^{(t)} - \dfrac{L_i^{(t)}}{2} \ln\left(\dfrac{1}{\mu}\right), & \mu > 0.5, \\ x_i^{(t+1)} = p_i^{(t)} + \dfrac{L_i^{(t)}}{2} \ln\left(\dfrac{1}{\mu}\right), & \mu \le 0.5, \end{cases} \tag{11}$$

where $p_i^{(t)}$ is the centre random position with δ quantum well of the ith particle in the tth iteration, $p\text{best}_i^{(t)}$ is the best position of the ith particle in the tth iteration, $g\text{best}^{(t)}$ is the best position of the population in the tth iteration, $x_i^{(t+1)}$ is the position of the ith particle in the $t+1$th iteration, β and μ are uniformly distributed random numbers in the interval $[0,1]$, and $L_i^{(t)}$ is the δ quantum well characteristic length of the ith particle in the tth iteration, which is defined as

$$L_i^{(t)} = 2\alpha \left| \frac{1}{N} \sum_{i=1}^{N} p\text{best}_i^{(t)} - x_i^{(t)} \right|, \tag{12}$$

where N is the size of the particle population and α is the contraction-expansion coefficient, which determines the convergence performance of the Algorithm. A larger α value is conducive to improve the global searching ability of particles and prevent local convergence, and a smaller α value is beneficial to enhance the local searching ability of particles and improve the convergence accuracy. It is noted that $\alpha < 1.781$ is the convergence condition of QPSO algorithm; otherwise, it will lead to divergence of algorithm [29].

On the basis of ensuring algorithm convergence, how to control the α value is the key to improve algorithm performance and efficiency. At present, discussions and studies on this coefficient in QPSO algorithm mainly focus on fixed value [29], linear decline [30], and nonlinear decline [31]. Compared with the fixed value, the dynamic decline method can improve the global search performance of the algorithm by taking a larger α value in the early stage of iteration, while taking a smaller α value in the late stage of iteration to enhance the local search performance of the algorithm. However, these dynamic decline methods only take into account the value of contraction-expansion coefficient changing with the number of iterations, which obviously cannot fully reflect the state changes in the actual evolution process of particles and cannot handle complex and nonlinear optimization problems well to some extent. In consideration of the limitations of the above parameter evaluation methods, we introduce the concept of success rate of particle swarm iteration [32, 33]. By improving the calculation method of success value of particle iteration in literature [33], we propose an adaptive contraction-expansion coefficient evaluation strategy that comprehensively reflects the changes of particle position state.

$$\alpha(t) = (\alpha_{\max} - \alpha_{\min})P_s(t) + \alpha_{\min}, \tag{13}$$

where $\alpha_{\max}$ and $\alpha_{\min}$ are the maximum value and minimum value of the contraction-expansion coefficient, respectively, and $P_s(t)$ is the success rate of population in the t^{th} iteration, which can be described as follows:

$$P_s(t) = \frac{1}{N}\sum_{i=1}^{N} S_i(t), \tag{14}$$

where $S_i(t)$ is the success value of the i^{th} particle in the t^{th} iteration, which is expressed as in literature [33]:

$$S_i(t) = \begin{cases} 1, & f\left(p\text{best}_i^{(t)}\right) < f\left(p\text{best}_i^{(t-1)}\right) \& f\left(p\text{best}_i^{(t)}\right) < f\left(g\text{best}^{(t)}\right), \\ 0.5, & f\left(g\text{best}^{(t)}\right) \leq f\left(p\text{best}_i^{(t)}\right) < f\left(p\text{best}_i^{(t-1)}\right), \\ 0, & f\left(p\text{best}_i^{(t)}\right) = f\left(p\text{best}_i^{(t-1)}\right), \end{cases} \tag{15}$$

where $f(p\text{best}_i^{(t)})$ is the fitness of the optimal position of the i^{th} particle in the t^{th} iteration and $f(g\text{best}^{(t)})$ is the fitness of the global optimal position of the population in the t^{th} iteration. From (15), it can be seen that $S_i(t)$ value greater than 0 represents the success of optimization; otherwise, it represents the failure of optimization [33]. However, the above method is only a simple piecewise constant function, which ignores the comparison of the state changes between the optimal position of the particle and the optimal position of the population during the evolution process. As a result, the calculation of $P_s(t)$ and $\alpha(t)$ is not accurate enough, which makes the algorithm unable to effectively balance the global and local search capabilities and easy to fall into the local optimal solution. In view of the inaccuracy of the above calculation method about $S_i(t)$, we make a more detailed evaluation of the particle position and state, which improves the calculation method again as follows:

$$S_i(t) = \begin{cases} 1, & f\left(p\text{best}_i^{(t)}\right) < f\left(p\text{best}_i^{(t-1)}\right) \& f\left(p\text{best}_i^{(t)}\right) < f\left(g\text{best}^{(t)}\right), \\ 1 - \dfrac{f\left(p\text{best}_i^{(t)}\right) - f\left(g\text{best}^{(t)}\right)}{f\left(p\text{best}_i^{(t)}\right)} & f\left(g\text{best}^{(t)}\right) \leq f\left(p\text{best}_i^{(t)}\right) < f\left(p\text{best}_i^{(t-1)}\right), \\ 0, & f\left(p\text{best}_i^{(t)}\right) = f\left(p\text{best}_i^{(t-1)}\right). \end{cases} \tag{16}$$

From (15), it can be seen that when the particle iteration is successful, the closer its optimal position is to the global optimal solution, the closer $f(p\text{best}_i^{(t)})$ is to $f(g\text{best}^{(t)})$, which makes $S_i(t)$ larger. On the contrary, the optimal position of particles is far from the global optimal solution, and there is a big gap between $f(p\text{best}_i^{(t)})$ and $f(g\text{best}^{(t)})$, which makes $S_i(t)$ smaller. Therefore, the calculation accuracy of $P_s(t)$ and $\alpha(t)$ is obviously improved after considering the change of particle position state so that an AQPSO algorithm that can dynamically balance global and local search capabilities can be obtained.

During the algorithm iteration, larger $P_s(t)$ indicates that the distance between the position of population and the global optimal solution is far, so $\alpha(t)$ should be improved to enhance the global searching ability of the algorithm, which improves the diversity of population. Smaller $P_s(t)$ indicates that the position of population is close to the global optimal solution, and $\alpha(t)$ should be reduced to enhance the local search ability of the algorithm, which ensures the convergence accuracy of the optimization algorithm. To optimize the regularization parameters and kernel function width of the LS-SVM model, the AQPSO algorithm is proposed in Algorithm 2. The fitness function to evaluate the candidate solution of the algorithm is constructed as follows:

$$f\left(x_i^{(t)}(\gamma,\sigma)\right) = \frac{1}{2}\sum_{j}^{l}\left(y_j^* - y_j(\gamma,\sigma)\right)^2, \tag{17}$$

where l is the total number of samples, y_j^* is the actual value of the j^{th} sample, and $y_j(\gamma,\sigma)$ is the predicted value of the j^{th} sample.

3.4. LS-SVM Time Delay Prediction Model Based on EMD and AQPSO Algorithms

3.4. LS-SVM Time Delay Prediction Model Based on EMD and AQPSO Algorithms. Assume that the TCN time delay

```
     Initialization
     Set the size of population $N$, the maximum number of iterations $T_0$, the tolerable error $lr$, and the random positions of the
     particles $x_i^{(t)}$.
     Procedure
(1)  while $t \le T_0 \| f(g\text{best}^{(t)}) > lr$ do
(2)    for $i = 1: N$ do
(3)  Calculate the fitness value of ith particle by (17)
(4)    if $f(x_i^{(t)}(\gamma, \sigma)) < f(p\text{best}_i^{(t-1)})$
(5)  Update the $p\text{best}_i^{(t)}$
(6)    end if
(7)    if $f(p\text{best}_i^{(t)}) < f(g\text{best}^{(t)})$
(8)  Update the $g\text{best}^{(t)}$
(9)    end if
(10) Calculate the iteration success value of each particle $S_i(t)$ by (16)
(11)   end for
(12) Calculate the success rate of population $P_s(t)$ by (14)
(13) Calculate the contraction-expansion coefficient $\alpha(t)$ by (13)
(14)   for $i = 1: N$ do
(15) Calculate the $\delta$ quantum well characteristic length of each particle $L_i^{(t)}$ by (12)
(16) Update the position of each particle $x_i^{(t+1)}$ by (11)
(17)   end for
(18) end while
     Output: the current global optimal solution $g\text{best}^{(t)}$
     End Procedure
```

Algorithm 2: AQPSO algorithm for the LS-SVM model.

sequence $\tau_{ca}(t)$ can be expressed as the following time delay sequence after EMD processing:

$$\tau = [\tau_1, \tau_2, \ldots, \tau_l], \tag{18}$$

where l is the sequence length. Sorting and transforming the time delay sequence τ, the input and output training sets of the LS-SVM model can be obtained:

$$X = \begin{bmatrix} \tau_1 & \tau_2 & \cdots & \tau_m \\ \tau_2 & \tau_3 & \cdots & \tau_{m+1} \\ \cdots & \cdots & \cdots & \cdots \\ \tau_{l-m} & \tau_{l-m+1} & \cdots & \tau_{l-1} \end{bmatrix}, \tag{19}$$

$$Y = \begin{bmatrix} \tau_{m+1} \\ \tau_{m+2} \\ \cdot \\ \tau_l \end{bmatrix}, \tag{20}$$

where X is the input training set, Y is the output training set, and m is the embedded dimension. According to (19) and (20), the LS-SVM model can use the time delay data of the previous m moment to predict the time delay data of the next moment, and the LS-SVM time delay prediction model based on EMD and AQPSO algorithms is shown in Figure 2.

The establishment process of hybrid delay prediction model in Figure 2 is as follows. First, the TCN time delay sequence is decomposed into several IMFs and a remain (R) by EMD algorithm. Second, all IMFs and R data are normalized, and the input and output training sets of each LS-SVM model are generated by using (19) and (20). In the next step, AQPSO algorithm is used to optimize kernel function parameters of different LS-SVM models offline and establish prediction models of each time delay component. Finally, the prediction results of each time delay component are unnormalized, and the final prediction results are obtained by summing the equal weights.

4. IGPC Strategy for HSTs Based on Reinforcement Learning Algorithm

Considering that HSTs are disturbed by many factors such as operating conditions, line ramps, and curve resistance, the model parameters are highly uncertain, which has a great impact on the control effect. Thus, the actor-critic neural network in reinforcement learning is used to approximate the actual operating conditions of the train and predict the actual output of the train traction system in the future. Meanwhile, the LS-SVM hybrid time delay prediction model is adopted to predict the forward channel time delay to overcome its influence on the control performance. In combination with the prediction results, the RLS algorithm with variable forgetting factor is used to identify the time-varying model parameters, which improves the IGPC law for achieving the smooth control parameter mutation situation and time delay compensation effectively in real time. The nonlinear network control structure is shown in Figure 3.

As shown in Figure 3, the ATO system sends the calculated control quantity $u(k - \hat{\tau}_{ca})$ and forward timestamp T_{ca} into a process data packet to the train traction control system through the MVB network, and the actuators in the traction control system execute the latest control law and record the actual forward channel time delay τ_{ca}, and the sensor node periodically acquires the output $y(k)$ according

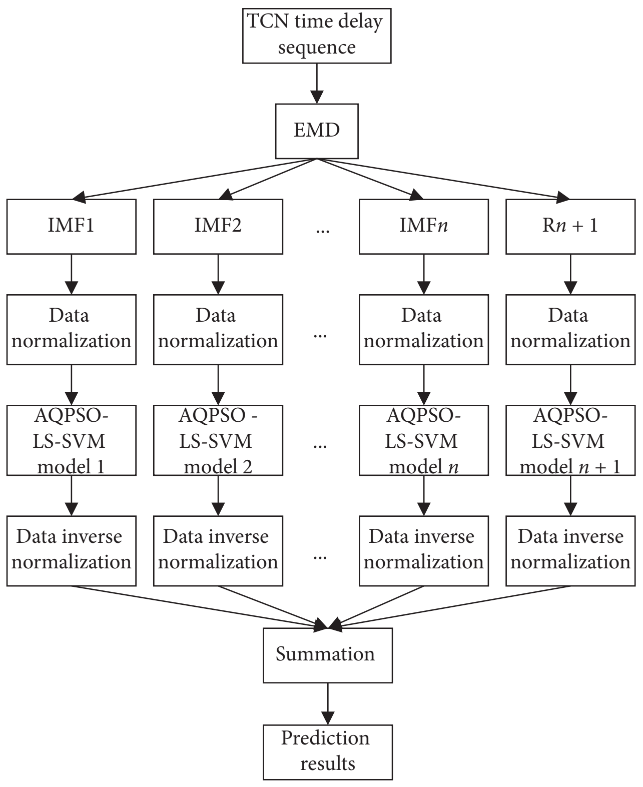


Figure 2: LS-SVM time delay prediction model based on EMD and AQPSO algorithms.

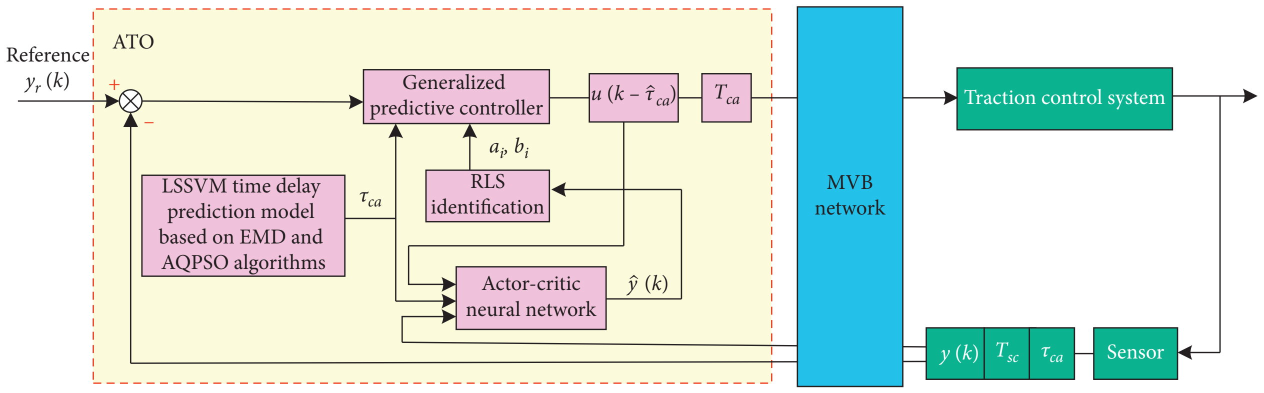


Figure 3: IGPC structure diagram of nonlinear network system for HSTs.

to the set sampling time and sends $y(k)$, feedback timestamp T_{sc}, and τ_{ca} into a process data packet to the ATO system through the MVB network. The ATO system can analyse the actual output $y(k)$ combined with the speed and control quantity at the previous moment, and the actor-critic neural network is used to predict the future output $\hat{y}(k)$ of the system. Furthermore, combined with the RLS identification and IGPC algorithm, the appropriate control quantity $u(k)$ is obtained in advance. Meanwhile, with the prediction results of LS-SVM hybrid time delay prediction model, the time series are arranged reasonably to send out the appropriate $u(k-\hat{\tau}_{ca})$ in advance, which compensates the influence of τ_{ca} on the control performance. In addition, due to the inconsistent clock of each subdevice, it is also necessary to perform clock correction while receiving the timestamp.

4.1. *Actor-Critic Neural Network Multistep Prediction.*

Neural network has become a common method to deal with nonlinear systems, due to its good fitting characteristics. The control method based on neural network can effectively solve the problems caused by the uncertainty and nonlinearity of the system [34]. Based on the neural network technology, reinforcement learning is a control method with stronger learning ability and higher robustness. It emphasizes that agents modify their action strategies through the return of the environment after each action in the interaction with the external environment so as to achieve optimization decision, which has been widely applied in artificial intelligence and intelligent control [35, 36]. In reinforcement learning algorithm based on the actor-critic structure, the actor network is responsible for learning optimal decisions so that the agent can choose the best one for different environmental states, and the critic network is responsible for fitting the value function, which enhances the intelligence ability of the agent to the environment, and the combination of the two networks ensures the effectiveness and robustness of the reinforcement learning algorithm [37, 38]. In [37], an actor-critic-based reinforcement learning algorithm is used to approximate the value function and uncertainty of system, respectively, which effectively solves the deterministic nonlinear discrete-time tracking control problem in the presence of input constraints. In [38], an actor-critic reinforcement learning algorithm based on fractional gradient descent RBF neural network is proposed to control the inverted pendulum system, which improves the convergence speed and stability.

Referring to the above literature design ideas, we adopt the RBF neural network to construct the critic and actor neural network for approximating the value function and action function, respectively, which learns the nonlinear characteristics of the traction system and predicts the output of the system in future.

4.1.1. Design of Critic Neural Network. The following state value function is defined to represent the discount sum of the expected revenue of a certain strategy at the beginning of the k time [37]:

$$\begin{aligned} V(k) &= R(k+1) + \gamma_0 R(k+2) + \gamma_0^2 R(k+3) + \cdots \\ &= \sum_{q=0}^{\infty} \gamma_0^q R(k+q+1), \end{aligned} \tag{21}$$

where $\gamma_0 \in [0,1]$ is the discount factor, which determines the present value of the benefits. If $\gamma_0 = 0$, it represents the agent that is short-sighted and only concerned with maximizing immediate benefits. Instead, as γ_0 gets closer to 1, the discounted returns will take more account of future returns, which means agents will become more far-sighted. $R(k)$ is the utility function, and it is designed as [37]

$$R(k) = \begin{cases} 0, & e^2(k) \le \xi, \\ 1, & e^2(k) > \xi, \end{cases} \tag{22}$$

where $e(k) = y_r(k) - y(k)$ represents the error between the desired speed of train and the actual speed of train, ξ is the threshold, and $R(k)$ represents the current system performance index, that is, $R(k) = 0$ stands for ideal tracking performance and $R(k) = 1$ indicates poor tracking performance.

Time difference (TD) error $e_c(k)$ reflects the actor neural network selected action decisions of the degree of excellence. Rewriting (21) into a recursive form, it is defined as [36]

$$e_c(k) = \gamma_0 V(k) - (V(k-1) - R(k)), \tag{23}$$

where $e_c(k) > 0$ indicates that the actual effect is better than expected, and the next decision will be more inclined to choose this action; $e_c(k) < 0$ means that the actual effect is worse than expected, and the next decision will reduce the propensity to choose this action.

$V(k)$ is approximated by the RBF neural network as

$$V(k) = \widehat{w}_c^{\mathrm{T}}(k)\varphi_c(x(k)), \tag{24}$$

where $\widehat{w}_c(k)$ is the estimate of the ideal network weight $w_c(k)$; the weight error is defined as $\widetilde{w}_c(k) = \widehat{w}_c(k) - w_c(k)$, $\varphi_c(x(k))$ is the Gaussian basis function.

Substituting (24) into (23), we obtain

$$e_c(k) = R(k) + \gamma_0 \widehat{w}_c^{\mathrm{T}}(k)\varphi_c(x(k)) - \widehat{w}_c^{\mathrm{T}}(k-1)\varphi_c(x(k-1)). \tag{25}$$

The cost function of critic neural network is defined as

$$E_c(k) = \frac{1}{2} e_c^{\mathrm{T}}(k) e_c(k). \tag{26}$$

Partial derivatives of (26) can be obtained as follows:

$$\frac{\partial E_c(k)}{\partial \widehat{w}_c(k)} = \frac{\partial E_c(k)}{\partial e_c(k)} \frac{\partial e_c(k)}{\partial \widehat{w}_c(k)} = \frac{\partial e_c(k)}{\partial \widehat{w}_c(k)} e_c(k). \tag{27}$$

Take the partial of (25) as

$$\frac{\partial e_c(k)}{\partial \widehat{w}_c(k)} = \gamma_0 \varphi_c(x(k)). \tag{28}$$

Substituting (25) and (28) into (27), we get

$$\frac{\partial E_c(k)}{\partial \widehat{w}_c(k)} = \gamma_0 \varphi_c(x(k)) \left[R(k) + \gamma_0 \widehat{w}_c^{\mathrm{T}}(k)\varphi_c(x(k)) - \widehat{w}_c^{\mathrm{T}}(k-1)\varphi_c(x(k-1)) \right]. \tag{29}$$

According to the gradient descent method, the weight update law for $\widehat{w}_c(k)$ is given by

$$\widehat{w}_c(k+1) = \widehat{w}_c(k) - \eta_c \frac{\partial E_c(k)}{\partial \widehat{w}_c(k)}, \tag{30}$$

where η_c is the learning rate of the critic neural network weight.

Substituting (29) into (30), (30) is rewritten as

$$\widehat{w}_c(k+1) = \widehat{w}_c(k) - \eta_c \gamma_0 \varphi_c(x(k)) \left[R(k) + \gamma_0 \widehat{w}_c^{\mathrm{T}}(k)\varphi_c(x(k)) - \widehat{w}_c^{\mathrm{T}}(k-1)\varphi_c(x(k-1)) \right]. \tag{31}$$

Note that there are no convergence guarantees with the weight update since it is an approximation to gradient descent but it proved successful in the simulations in this paper. One can refer to [38, 39] for an exact gradient descent algorithm with improved convergence guarantees.

4.1.2. Design of Actor Neural Network. Actor neural network can use historical data to predict the dynamic output of the system in the future; its ideal model is as follows:

$$y(k) = w_a(k)\varphi_a(x(k)) + \varepsilon(k), \tag{32}$$

where $w_a(k)$ is the ideal network weight, $\varphi_a(x(k))$ is the Gaussian basis function (the network input is selected as $x(k) = [y(k-1),\dots,y(k-n_y),u(k-1),\dots,u(k-n_u)]$), n_y and n_u are order of output and control sequence, $y(k)$ is the train speed at the next moment, and $\varepsilon(k)$ is the error of estimation.

The actual output of actor neural network is as follows:

$$\hat{y}(k) = \hat{w}_a^{\mathrm{T}}(k)\varphi_a(x(k)), \tag{33}$$

where $\hat{w}_a(k)$ is the estimate of $w_a(k)$ and $\hat{y}(k)$ is the predicted speed; the weight error is defined as $\tilde{w}_a(k) = \hat{w}_a(k) - w_a(k)$.

The error of actor neural network is defined as

$$\begin{aligned} e_a(k) &= \hat{w}_c^{\mathrm{T}}(k)\varphi_c(x(k)) + \hat{w}_a^{\mathrm{T}}(k)\varphi_a(x(k)) - w_a(k)\varphi_a(x(k)) \\ &\quad - \varepsilon(k), \\ &= \hat{w}_c^{\mathrm{T}}(k)\varphi_c(x(k)) + \tilde{w}_a(k)\varphi_a(x(k)) - \varepsilon(k). \end{aligned} \tag{34}$$

The cost function of actor neural network is defined as

$$E_a(k) = \frac{1}{2}e_a^{\mathrm{T}}(k)e_a(k). \tag{35}$$

From (35), the gradient is derived as

$$\frac{\partial E_a(k)}{\partial \hat{w}_a(k)} = \frac{\partial E_a(k)}{\partial e_a(k)}\frac{\partial e_a(k)}{\partial \hat{w}_a(k)} = \frac{\partial e_a(k)}{\partial \hat{w}_a(k)}e_a(k). \tag{36}$$

Take the partial of (34) as

$$\frac{\partial e_a(k)}{\partial \hat{w}_a(k)} = \varphi_a(x(k)). \tag{37}$$

Substituting (34) and (37) into (36), we obtain

$$\frac{\partial E_a(k)}{\partial \hat{w}_a(k)} = \varphi_a(x(k))\left[\hat{w}_c^{\mathrm{T}}(k)\varphi_c(x(k)) + \tilde{w}_a(k)\varphi_a(x(k)) - \varepsilon(k)\right]. \tag{38}$$

According to the gradient descent method, the updating law of actor neural network weight is as follows:

$$\hat{w}_a(k+1) = \hat{w}_a(k) - \eta_a\frac{\partial E_a(k)}{\partial \hat{w}_a(k)}, \tag{39}$$

where η_a is the learning rate of the actor neural network weight.

Substituting (38) into (39), (39) is rewritten as

$$\hat{w}_a(k+1) = \hat{w}_a(k) - \eta_a\varphi_a(x(k))\left[\hat{w}_c^{\mathrm{T}}(k)\varphi_c(x(k)) + \tilde{w}_a(k)\varphi_a(x(k)) - \varepsilon(k)\right]. \tag{40}$$

Note that the convergence proof of the actor-critic network during learning is provided in [37, 40].

In the ATO system, the control output sequence needs to be reconstructed and sent in advance. Considering that the forward channel time delay is generally several times the sampling period, the number of recursive prediction steps is taken as $d = (\hat{\tau}_{ca-\max}/T)$, i.e., the maximum forward channel time delay under the current configuration is divided by the sampling period. On the basis of obtaining relevant input and output information at k moment, actor-critic neural network can be used to perform real-time online recursive prediction of the output sequence within d period. The prediction process is shown in Figure 4.

Figure 4 shows that the actor and critic networks are represented by the RBF neural network, and the external environment is composed of relevant input and output sequences of the system at k moment. Actor networks can use the output and control of previous moments to predict future train speed $\hat{y}(k)$ in one step and update the next decision based on the TD error $e_c(k)$ obtained by the critic network. Meanwhile, the critic network can use the same $e_c(k)$ to adjust the state value function, and the critic network computes the $e_c(k)$ using the $R(k)$ and $V(k)$ to prepare for the next prediction after the actor network outputs the predictions to the environment. It is noted that the actor network does not get $R(k)$ directly, and the critic network does not get $\hat{y}(k)$ directly in the actor-critic neural network prediction process.

4.2. Online Model Parameter Identification with RLS Algorithm. IGPC needs to obtain the linear model parameters of the controlled object. Thus, based on the output of future time predicted by actor-critic network and output and control quantity of past time, the RLS algorithm with variable forgetting factor is used to identify the model parameters of IGPC strategy at different times. The model parameters are $\hat{\theta}(k) = [a_1,\dots,a_{n_a},b_1,\dots,b_{n_b}]^{\mathrm{T}}$, and the corresponding identification formula is as follows [41]:

$$\hat{\theta}(k) = \hat{\theta}(k-1) + K(k)\left[\hat{y}(k)t - nh^{\mathrm{T}}q(k)h\hat{\theta}(k-1)\right], \tag{41}$$

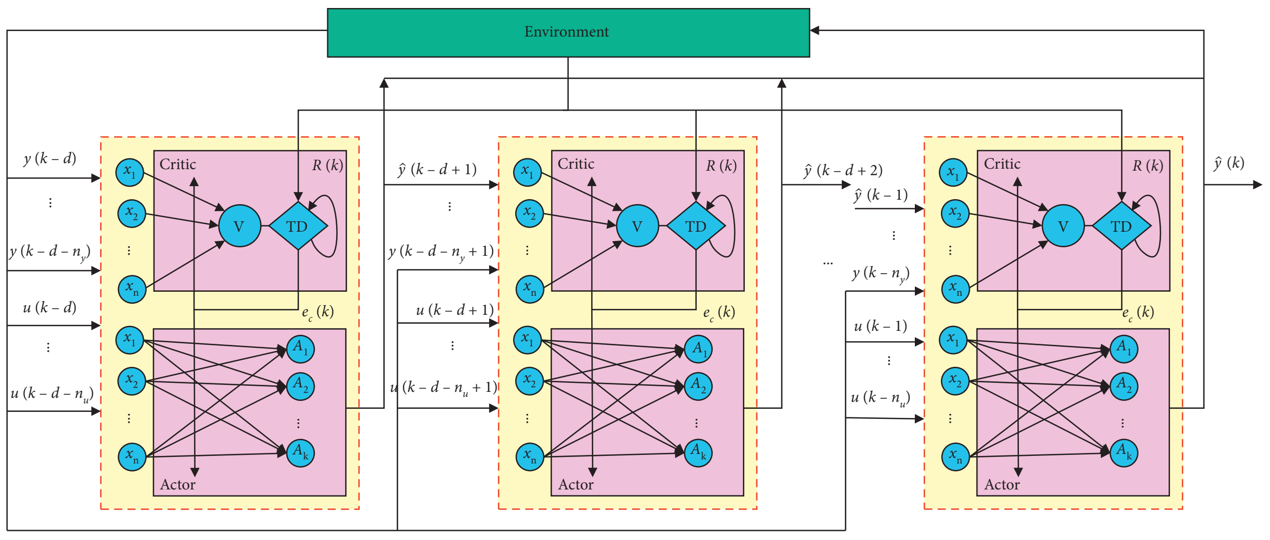


Figure 4: Actor-critic neural network multistep prediction process.

where $h(k) = [-\hat{y}(k-1), \ldots, -\hat{y}(k-n_a), \hat{u}(k-1), \ldots, \hat{u}(k-n_b)]$ is the set of system input and output samples, n_a and n_b represent the input and output orders respectively, and $K(k)$ is the gain matrix, which can be represented as

$$K(k) = P(k-1)h(k)\left[\lambda_0(k) + h^{\mathrm{T}}(k)P(k-1)h(k)\right]^{-1}, \tag{42}$$

where $P(k)$ is the covariance matrix of the error and $\lambda_0(k) \in (0,1]$ is the forgetting factor, which is a parameter to correct the performance index in order to prevent "data saturation." Its value is determined by error $\varepsilon(k)$ and parameter σ. $\lambda_0(k)$ and $P(k)$ can be expressed as follows:

$$\begin{aligned} \lambda_0(k) &= 1 - \varepsilon(k)^2\left[\sigma\left(1 + h^{\mathrm{T}}(k)P(k-1)h(k)\right)\right]^{-1}, \\ P(k) &= \left[\frac{P(k-1) - K(k)h^{\mathrm{T}}(k)P(k-1)}{\lambda_0(k)}\right], \end{aligned} \tag{43}$$

and we set the initial value as $P(0) = 10^6\mathrm{I}$, where I is the unit matrix.

4.3. Design of Improved Generalized Predictive Controller.

The train network control system identified at different moments is the following controlled autoregressive moving average (CARIMA) model [42]:

$$A(z^{-1})y(k) = B(z^{-1})u(k-\hat{\tau}_{ca}) + \frac{C(z^{-1})\zeta(k)}{\Delta}, \tag{44}$$

where $\Delta = 1 - z^{-1}$ is the difference operator, z^{-1} is the delay operator, $\zeta(k)$ is the white noise sequence with a mean of 0, and $A(z^{-1})$, $B(z^{-1})$, and $C(z^{-1})$ are the following polynomials of z^{-1}; the order is n_a, n_b, and n_c, respectively.

$$\begin{cases} A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + +a_{n_a} z^{-n_a}, \\ B(z^{-1}) = b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + +b_{n_b} z^{-n_b}, \\ C(z^{-1}) = 1 + c_1 z^{-1} + c_2 z^{-2} + \cdots + +c_{n_c} z^{-n_c}, \end{cases} \tag{45}$$

where a_i is the polynomial coefficient, in which $i = 1, 2, \ldots, n_a$, b_i is the polynomial coefficient, in which $i = 1, 2, \ldots, n_b$, and c_i is the polynomial coefficient, in which $i = 1, 2, \ldots, n_c$.

Define the following variables:

$$\begin{cases} E_j(z^{-1}) = e_1 + e_2 z^{-1} + \cdots + e_j z^{-(j-1)}, \\ F_j(z^{-1}) = f_1 + f_2 z^{-1} + \cdots + f_{n+1} z^{-n}, \\ G_j(z^{-1}) = g_1 + g_2 z^{-1} + \cdots + g_j z^{-(j-1)}, \\ H_j(z^{-1}) = h_1 + h_2 z^{-1} + \cdots + h_{n-1} z^{-(n-2)}, \end{cases} \tag{46}$$

where e_i is the polynomial coefficient, in which $i = 1, 2, \ldots, j$, f_i is the polynomial coefficient, in which $i = 1, 2, \ldots, n+1$, g_i is the polynomial coefficient, in which $i = 1, 2, \ldots, j$, and h_i is the polynomial coefficient, in which $i = 1, 2, \ldots, n-1$.

Solve the Diophantine equation:

$$\left\{1 = E_j(z^{-1})A(z^{-1})\Delta + z^{-j}F_j(z^{-1})E_j(z^{-1})B(z^{-1}) = G_j(z^{-1}) + z^{-j}H_j(z^{-1}).\right. \tag{47}$$

According to (47), we can obtain $F_j(z^{-1})$, $G_j(z^{-1})$, and $H_j(z^{-1})$. The performance index of CARIMA is defined as follows:

$$J = E\left\{\sum_{j=1}^{P}\left[y(k+j) - y_r(k+j)\right]^2 + \sum_{j=1}^{M}\lambda\left[\Delta u(k+j-\hat{\tau}_{ca}-1)\right]^2\right\}, \tag{48}$$

where P is the prediction horizon, M is the control horizon, and λ is the control weighting factor; the performance index J is minimized to derive the control law:

$$\begin{cases}\Delta u(k-\hat{\tau}_{ca}) = \mathrm{K}_1\left[Y_r - f(Fy(k) + H\Delta U)\right],\\ u(k-\hat{\tau}_{ca}) = u(k-\hat{\tau}_{ca}-1) + \Delta u(k-\hat{\tau}_{ca}),\end{cases} \tag{49}$$

where $\Delta U = [\Delta u(k-\hat{\tau}_{ca}),\ \Delta u(k-\hat{\tau}_{ca}+1),\ldots,\Delta u(k-\hat{\tau}_{ca}+M)]^{\mathrm{T}}$, $Y_r = [y_r(k), y_r(k+1),\ldots,y_r(k+P)]^{\mathrm{T}}$, and K_1 is the first row of the matrix $(G^{\mathrm{T}}G+\lambda\mathrm{I})^{-1}G^{\mathrm{T}}$. The above process is based on the latest acquired data packet information, which can realize the rolling optimization of each real-time sublinear model.

5. Simulation and Analysis

To verify the effectiveness of the proposed method, CRH3 train [2] is taken as the controlled object. The main parameters of the CRH3 train are as follows: the maximum running speed is 350 km/h, the sustained running speed is 300 km/h, the total train weight is 400 tons, and the rotary turning coefficient is 0.06. Control parameter settings are shown in Table 1. The TCN simulation platform is built by CCU1, CCU2, and HMI. The model of the train traction control system is set up in CCU2. The latest speed information is analysed in CCU1, and the LS-SVM hybrid delay prediction algorithm, actor-critic neural network algorithm, recursive RLS algorithm with variable forgetting factor, and IGPC algorithm are inserted into it. The task execution period of each device was configured to be 50 ms, and all the transceiver data modules, algorithm, and model modules were executed in the same task to achieve synchronous calculation [23]. In practical application, the characteristic period of each port is usually selected as a multiple of 64 ms. Therefore, the sampling period T is selected as 64 ms.

5.1. Construction of TCN Simulation Platform. Setting up an operating environment consistent with the actual train operating environment and the functions of each node, the semiphysical simulation platform of TCN is shown as in Figure 5. During the test, the task period is set to 50 ms, the load rate is set to 45%, and the characteristic period of the source port is set to 64 ms and 128 ms, respectively.

Figure 5 shows that all vehicle control signals are transmitted through the MVB network. As the controller node, CCU1 can simulate the ATO system, while as the actuator node, CCU2 can simulate the train traction control system. In the control process, CCU1 sends the control instruction to CCU2, which simulates the train traction and braking execution process and sends the completed execution information to the sensor node. As the sensor node, HMI can measure the output signal of the system and feed it back to CCU1. When the data are transmitted through the MVB network, the computer connects the network analyser through RS-232 serial line and Ethernet data line, tests the forward and feedback channel delay of the MVB network, and analyses the delay data sample according to the captured process data.

5.2. Performance Analysis of Different Time Delay Prediction Models. In order to verify the prediction performance of the proposed time delay prediction model, we compare the proposed model with the LS-SVM model based on EMD (EMD-LS-SVM) [25], LS-SVM model [27], Elman neural network model (ELMANNN) [43], and the least mean square algorithm based on AR model (LMS-AR) [44]. For all prediction models, the training set is 500 sets of data, the test set is 50 sets of data, the input end of each model is $X = [\tau(k-1), \tau(k-2), \tau(k-3)]$, and the output end is $Y = \tau(k)$. The parameters of the EMD-LS-SVM model are set as follows: $\gamma = 1854.945$ and $\sigma^2 = 0.842$. The parameters of the LS-SVM model are set as follows: $\gamma = 943.571$ and $\sigma^2 = 0.598$. The parameters of the ELMANNN model are set as follows: the maximum number of training is 5000, the training target is 0.01, the learning rate is 0.1, and the number of neurons is 50. The parameters of the LMS-AR model are set as follows: the order number is 20 and the convergence factor is 7.8×10^{-7}. When the characteristic period is 64 ms, the time delay prediction results of different models are shown in Figure 6(a). Figure 6(b) shows the time delay prediction results of different models when the characteristic period is 128 ms. In order to better measure the prediction effect of each model on the time delay series, the root mean square error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), and prediction time are used to evaluate the overall performance of each prediction model. The performance indexes of each prediction model under different characteristic periods are recorded in Table 2. The RMSE, MAE, and MAPE are defined as follows:

$$\begin{cases} e_{\mathrm{RMSE}} = \sqrt{\dfrac{1}{L}\displaystyle\sum_{k=1}^{L}\left(\tau^*(k) - \hat{\tau}(k)\right)^2},\\[2ex] e_{\mathrm{MAE}} = \sqrt{\dfrac{1}{L}\displaystyle\sum_{k=1}^{L}\left|\tau^*(k) - \hat{\tau}(k)\right|},\\[2ex] e_{\mathrm{MAPE}} = \dfrac{1}{L}\displaystyle\sum_{k=1}^{L}\left|\dfrac{\tau^*(k) - \hat{\tau}(k)}{\tau^*(k)}\right| \times 100\%,\end{cases} \tag{50}$$

where L is the length of the time delay sequence, $\tau^*(k)$ is the real time delay value at the k moment, and $\hat{\tau}(k)$ is the predicted delay value at the k moment.

Figure 6 shows that with the increase of the characteristic period of the source port, the decoding and access time of

Table 1: Control parameters.

Parameters	Value
$\alpha_0(k)$	$\alpha_0(k) \in [0.53, 0.58]$
$\alpha_1(k)$	$\alpha_1(k) \in [0.0039, 0.0041]$
$\alpha_2(k)$	$\alpha_2(k) \in [0.000114, 0.0001176]$
χ	$\chi = (1/(1+0.06))$
$d(k)$	$d(k) = \sin((0.01 + 0.1 * \text{rand})k)$
γ_i	$\gamma_1 = 48.883, \gamma_2 = 39.932, \gamma_3 = 4172.392, \gamma_4 = 1365.056,$ $\gamma_5 = 384.804, \gamma_6 = 2746.093, \gamma_7 = 0.063, \gamma_8 = 139.116$
σ_i^2	$\sigma_1^2 = 0.007, \sigma_1^2 = 0.014, \sigma_3^2 = 0.237, \sigma_4^2 = 0.162,$ $\sigma_5^2 = 1.084, \sigma_6^2 = 328.113, \sigma_7^2 = 1.146, \sigma_8^2 = 291.540$
γ_0	$\gamma_0 = 0.9$
ξ	$\xi = 0.5$
w_c	$w_c = [0.82,\ 0.91,\ 0.13,\ 0.91,\ 0.63]$
w_a	$w_a = [0.10,\ 0.28,\ 0.55,\ 0.96,\ 0.97]$
η_c, η_a	$\eta_c = 0.5$, $\eta_a = 0.5$
n_y, n_u	$n_y = 2$, $n_u = 2$
n_a, n_b	$n_a = 2$, $n_b = 2$
d	$d = 3$
P	$P = 5$
M	$M = 3$
λ	$\lambda = 0.1$

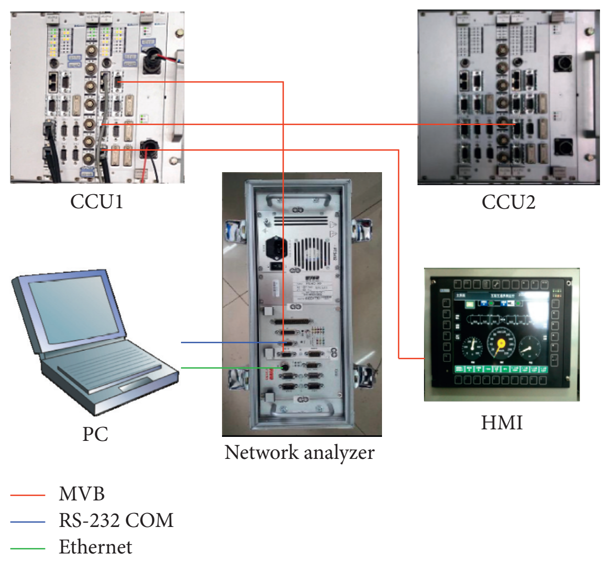


Figure 5: TCN semiphysical simulation platform.

MVB process data becomes longer, resulting in the severe time delay jitter of the system. Compared with other prediction models, the proposed prediction model can maintain high prediction accuracy in the face of different degrees of delay jitter, which effectively overcomes the impact of network delay on the train control system and ensures the real-time performance of control signal transmission.

Table 2 shows that the prediction accuracy of the proposed prediction model is significantly higher than that of other prediction models. The main reason is that after the original delay sequence is processed by EMD, different local characteristics of time delay components are highlighted, and the components of frequency components and waveform changes become simpler and more regular, which effectively reduces the difficulty to predict. At the same time, the proposed model optimized by AQPSO algorithm has more accurate kernel function parameters, which effectively improves the prediction accuracy compared with the EMD-LS-SVM model. On the other hand, since the prediction results of the proposed model are composed of the predicted values of different components generated by EMD, the algorithm takes a little longer to execute but the prediction accuracy is greatly improved, and the one-step delay prediction time of the proposed model is far less than the task execution period, which can meet the real-time requirements of high-speed train network control system.

5.3. Comparison of Optimization Performance of Different PSO Algorithms. To validate the AQPSO algorithm optimization performance, we select the QPSO algorithm based on linear contraction-expansion coefficient (QPSO-LDCE) [30], QPSO algorithm based on nonlinear decreasing contraction-expansion coefficient (QPSO-NDCE) [31], PSO algorithm based on swarm success rate descending inertia weight (SSRDIWPSO) [32], and PSO algorithm based on adaptive inertia weight (AIWPSO) [33] to compare with the proposed algorithm. The above algorithms are simulated in CCU. The hardware configurations of CCU are Intel (R) Core (TM) I5-7300HQ cpu@2.50 GHz processor and 8.00 GB memory. For all optimization algorithms, the maximum iteration time is selected as 30 times, the population number is selected as 50, and the tolerance error is set as 0.01. After testing, the parameters of AQPSO algorithm are set as follows: $\alpha_{\max} = 1$ and $\alpha_{\min} = 0.5$. The parameters of QPSO-LDCE algorithm are set as follows: $m = 1$ and $n = 0.5$. The parameters of QPSO-NDCE algorithm are set as follows: $\alpha_{\text{initial}} = 1$ and $n = 2$. The parameters of SSRDIWPSO algorithm are set as follows: $w_{\text{start}} = 0.9$ and

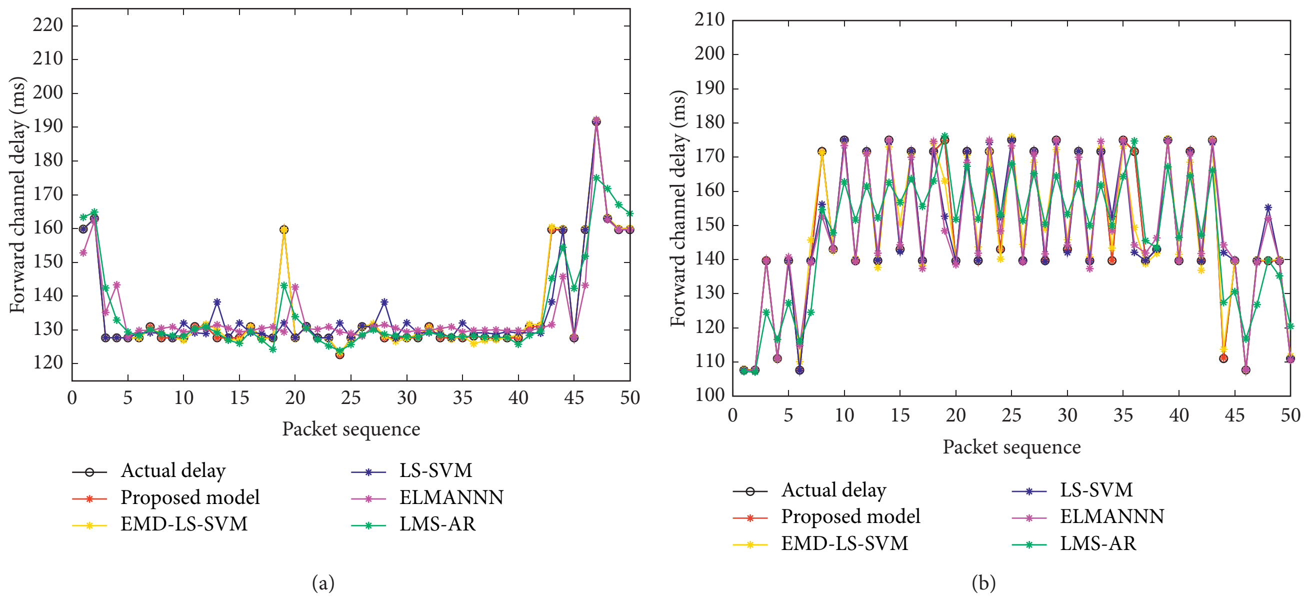


Figure 6: The prediction results of time delay under different characteristic periods. (a) The prediction results of time delay under 64 ms characteristic period. (b) The prediction results of time delay under 128 ms characteristic period.

Table 2: Comparison of performance indexes of different prediction models.

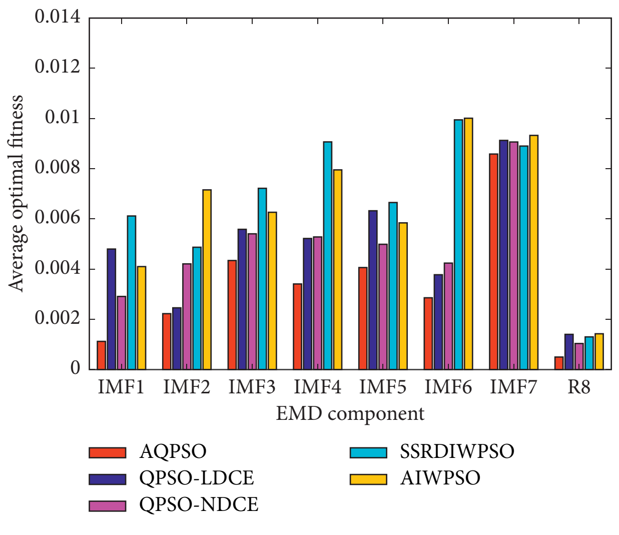
Evaluation index		Proposed model	EMD-LS-SVM	LS-SVM	ELMANNN	LMS-AR
Characteristic period: 64 ms	RMSE	0.034	0.773	5.755	7.208	5.619
	MAE	0.026	0.539	2.507	4.131	3.123
	MAPE	0.019	0.41	1.81	2.938	2.267
	Time (ms)	3.473	3.877	0.54	1.833	0.062
Characteristic period: 128 ms	RMSE	0.085	4.399	7.851	8.169	9.862
	MAE	0.062	2.323	2.901	3.774	11.203
	MAPE	0.044	1.532	1.988	2.597	6.126
	Time (ms)	3.886	3.983	0.521	1.806	0.056

$w_{\text{end}} = 0.4$. The parameters of AIWPSO algorithm are set as follows: $w_{\max} = 1$ and $w_{\min} = 0$. For the time delay component model, each algorithm runs independently for 20 times. Figure 7(a) shows the average optimal fitness of different PSO algorithms, and Figure 7(b) shows the average running time of different PSO algorithms.

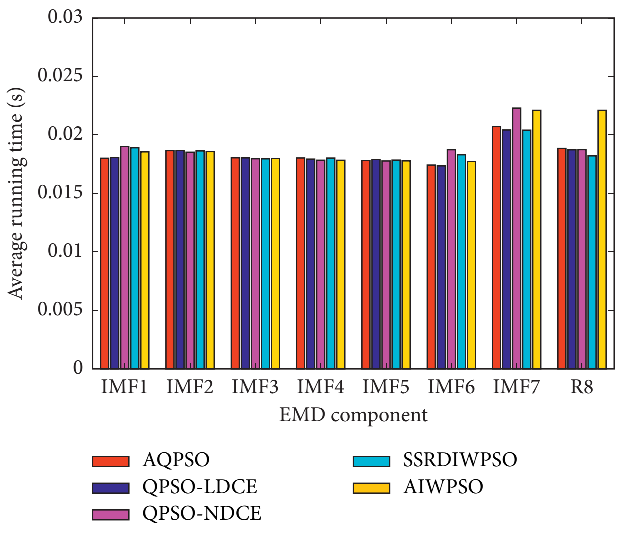
Figure 7 shows that compared with other PSO algorithms, the AQPSO algorithm has better convergence performance in optimizing parameters of models with different time delay components. The main reason is that PSO algorithm is extended to the quantum space and represents the motion state of particles in the form of wave function, which simplifies the evolution mode and enables the particle to appear at any position in the entire search solution space with a certain probability. Thus, the AQPSO algorithm has a stronger global search performance compared with SSRDIWPSO and AIWPSO algorithms. On the other hand, the AQPSO algorithm comprehensively considers the position and state changes of particles in the iterative process and designs the adaptive contraction-expansion coefficient so that the proposed algorithm can dynamically balance the global and local searching ability of particles. Therefore, the AQPSO algorithm has higher average convergence precision and stability compared with QPSO-LDCE and QPSO-NDCE algorithms. In terms of running time, the proposed algorithm does not increase significantly compared with other PSO algorithms, and the running time of each algorithm under a given tolerance error is basically the same.

5.4. Time Delay Compensation Effect of Different Characteristic Periods. In order to analyse the influence of time delay compensation on the control method under different characteristic periods, the characteristic period of the source port is set as 64 ms and 128 ms, respectively, the sine wave trajectory is selected as the reference input, the sampling time is set as 64 ms, and the initial control quantity is set as 0. Figure 8(a) represents the time delay compensation effect when the characteristic period is 64 ms, and Figure 8(b) represents the time delay compensation effect when the characteristic period is 128 ms.

Figure 8 shows that when the characteristic period is 64 ms, the proposed method can track sine wave quickly and accurately with almost no overshooting after adding time delay compensation, and the traction and braking effect is very ideal. If the time delay is not controlled, the output will occasionally oscillate and be unstable at different time points.

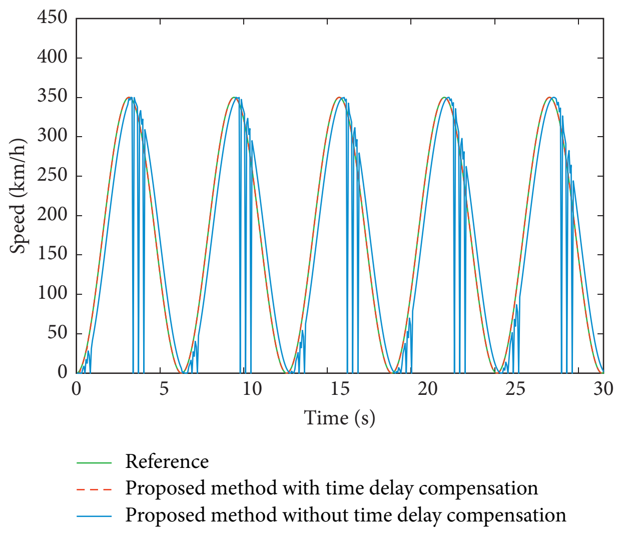


(a)

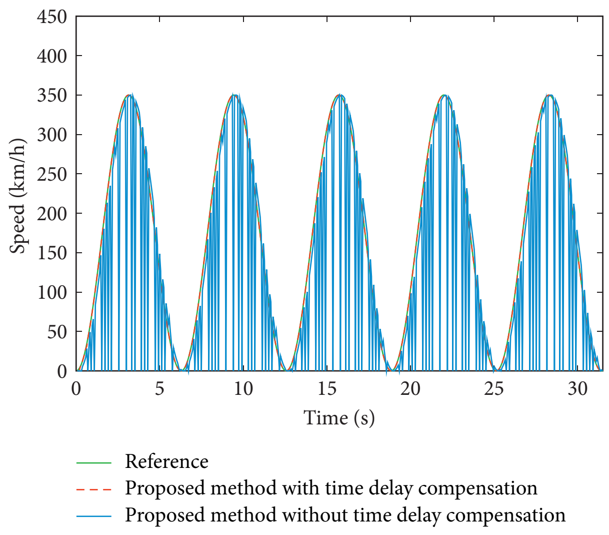


(b)

Figure 7: The optimization results of different PSO algorithms. (a) The average optimal fitness of different PSO algorithms. (b) The average running time of different PSO algorithms.



(a)



(b)

Figure 8: The time delay compensation effect under different characteristic periods. (a) The time delay compensation effect under 64 ms characteristic period. (b) The time delay compensation effect under 128 ms characteristic period.

When the characteristic period is 128 ms, the forward and feedback channel delay will continue to increase, and the sum will exceed 300 ms. If the delay control strategy is not applied, the output oscillation is very severe. The reason for its oscillation is that the time delay is random change and the time when the control quantity arrives at the actuator is not fixed, which leads to the failure of control method at many moments. It is worth noting that the characteristic period is larger, the time delay is greater, and the oscillation is more serious. In the future, the data transmission of the train will be more and more frequent, and the time delay of the forward channel will be several times or even more than the sampling period. Thus, the key to realize fast and stable control of traction and braking is that we should adopt effective methods to compensate the influence of time delay on the train control system.

5.5. Comparison of Tracking Effects between Different Control Methods.

In order to verify the real-time performance and tracking performance of the proposed method, the nonsingular fast terminal sliding mode control method (NFTSM) [2], RBF neural network adaptive control method (RBFNN) [34], and proportional-integral-differential control method (PID) [45] are selected for comparison with the proposed method. The parameters of the NFTSM control method are set as follows: $\beta = 0.05$, $\delta = 0.35$, $K_1 = 0.042$,

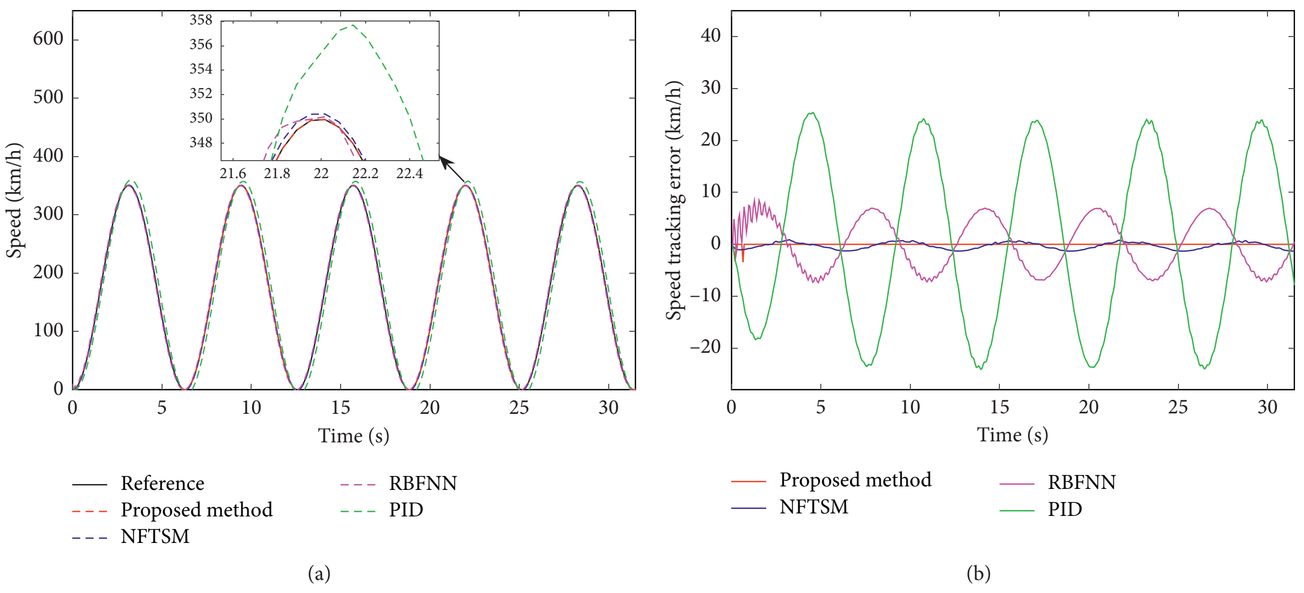


Figure 9: The speed tracking results of different control methods under sine wave. (a) The speed tracking results of different control methods. (b) The speed tracking errors of different control methods.

$K_2 = 1.012$, $\gamma_1 = 1.047$, $\gamma_2 = 0.467$, $\alpha_1 = 0.411$, $\alpha_2 = 0.891$, and $\varpi = 1.225$. The parameters of the RBFNN control method are set as follows: $c_1 = -0.1$, $\beta = 0.06$, $\gamma = 0.001$, $G = 50000$, and $\varepsilon_f = 0.003$. The parameters of the PID control method are set as follows: $k_p = 50$, $k_i = 20$, and $k_d = 5$. The characteristic period of the source port is 64 ms, the reference input selects the sine wave and the actual working condition, respectively, the sampling time is set to 64 ms, and the initial control quantity is set to 0. Figures 9(a) and 9(b), respectively, show the speed tracking results and errors of each control method under sine wave. Figures 10(a) and 10(b), respectively, show the speed tracking results and errors of each control method under actual working condition. Table 3 records the statistical results of tracking errors of different control methods under actual working conditions.

Figure 9 shows that compared with the proposed method and NFTSM method, the tracking effect of RBFNN adaptive control method and PID control method is poorer. Meanwhile, the speed tracking error fluctuation is bigger with the rapid change of sine wave, which illustrates that these control methods are difficult to be applied to high-speed train network control system with strong nonlinear change and uncertainty. Compared with other control methods, the proposed method has the smaller speed tracking error and has higher tracking accuracy and faster response speed at different moments, which can accurately track the rapid change of sine wave trajectory with better real-time performance. The advantage of the proposed method lies in the fact that the actor-critic neural network used in the control process to continuously interact with the environment gives full play to the learning ability of reinforcement learning algorithm to complex system. Thus, the actor-critic neural network can accurately predict the output of the traction control system in future, which effectively improves the control accuracy of the proposed method.

Figure 10 shows that during the alternations of tracking signals from acceleration to braking, compared with other control methods, the control performance of the proposed method is good in the whole speed and braking phase. The proposed method realizes smooth transition in different static working points and has faster response speed and dynamic performance, which can guarantee the high-precision tracking of the train at a given speed. Therefore, the proposed method fully meets demands to the safe operation of HSTs under complex conditions.

It can be seen from Table 3 that the proposed method has more ideal tracking performance and real-time performance compared with other control methods, which can provide effective means for the safe and reliable operation of HSTs. In addition, the proposed method is simple in structure, small in computation, and easy to be applied to train communication engineering.

6. Discussion

To predict the train network delay with the uncertainty and nonlinearity, the LS-SVM time delay prediction model based on EMD and AQPSO algorithms is designed. After EMD processing, the original time delay sequence is transformed from long correlation sequence to short correlation sequence, which highlights the different local characteristics of the original signal and effectively reduces the modelling complexity. By improving the calculation method about the successful value of particle iteration, an AQPSO algorithm with adaptive contraction-expansion coefficient is proposed to optimize the parameters of LS-SVM model, which enhances the prediction performance of the time delay prediction model and overcomes the effect of time delay on the train control system.

Considering the nonlinear characteristics of the train in the process of traction and braking, the actor-critic network is used in the control process to continuously interact with

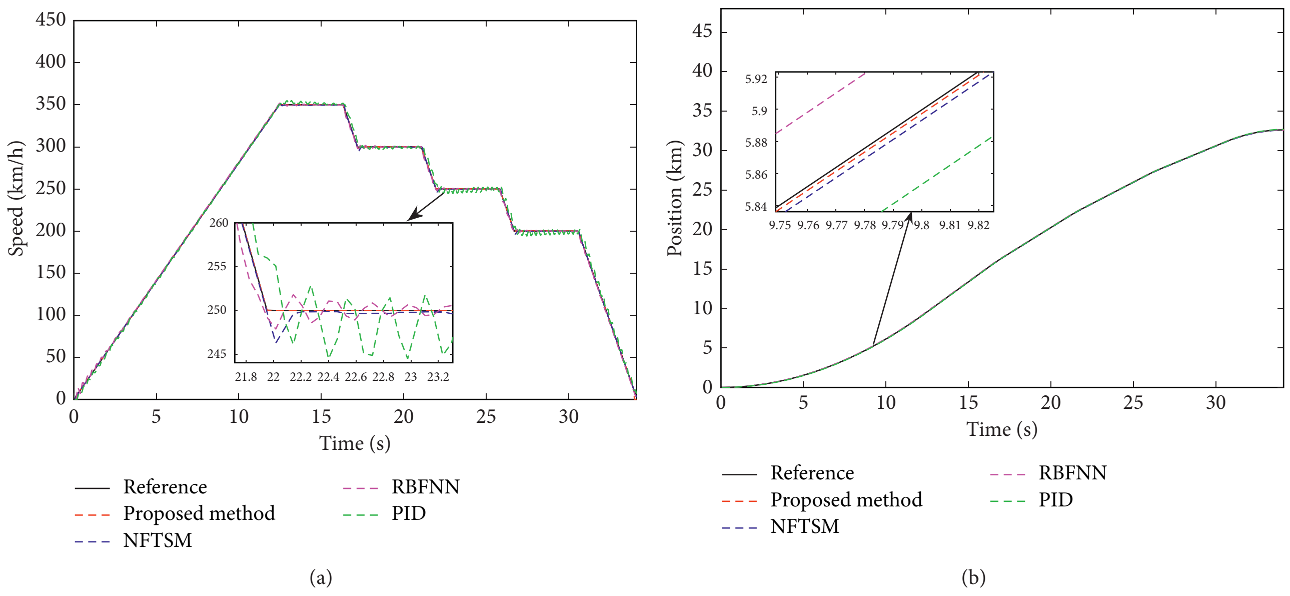


Figure 10: The speed tracking results of different control methods under actual working conditions. (a) The speed tracking results of different control methods. (b) The position tracking results of different control methods.

Table 3: Tracking error statistics.

Evaluation index		Proposed method	NFTSM	RBFNN	PID
Speed tracking error	RMSE	0.565	0.55	1.481	3.218
	MAE	0.054	0.316	1.096	2.343
Position tracking error	RMSE	2.5×10^{-3}	2.31×10^{-2}	3×10^{-2}	4.06×10^{-2}
	MAE	2.4×10^{-3}	1.88×10^{-2}	2.43×10^{-2}	3.69×10^{-2}

the environment and accurately predict the future output of the system, and the RLS identification algorithm with the variable forgetting factor is adopted to identify the future system model parameters, which realizes the predictive control of the nonlinear train network system. Further, combined with effective time delay prediction and compensation methods, the IGPC scheme for the train key nonlinear network system is implemented.

In this paper, we compensate the time delay generated in TCN without further considering the actual impact of packet loss on the train control system. However, in the process of time delay testing, it is found that when the port characteristic period increases, a small number of packets are lost in the data transmission through MVB network, so an effective method should be proposed to suppress the influence of both time delay and packet loss. Thus, we will design a more efficient and robust train network control method for such situations in the future.

7. Conclusions

In this paper, a LS-SVM time delay prediction model based on EMD and AQPSO algorithm is proposed to accurately predict the forward channel time delay for compensating the effect of network delay on train control performance. Based on the actor-critic reinforcement learning algorithm, an IGPC method is designed for HSTs, and the actor-critic reinforcement learning network is used to predict the output of the system by multiple steps in future moment, and according to the output prediction, the RLS algorithm can identify the system model parameters in the future. Combined with the forward delay prediction results, the suitable control quantity is sent in advance, which realizes the time delay compensation control of the train nonlinear network control system. Simulation results show that the proposed method can track the change of reference signal quickly and has good real-time performance, robustness, and stability. The research in this paper provides reference for the optimal control of train communication network and plays an important role in further enhancing the economy, safety, and reliability of high-speed train operation.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was funded by the Natural Science Foundation of Liaoning Province (grant no. 20180551003).

References

[1] B. Chen, "Development and reference of the TCMS on power centralized EMU abroad," *Railway Locomotive & Car*, vol. 39, no. 1, pp. 7–14, 2019.

[2] X. Kong and T. Zhang, "Non-singular fast terminal sliding mode control of high-speed train network system based on improved particle swarm optimization algorithm," *Symmetry*, vol. 12, no. 2, p. 205, 2020.

[3] T. Zhang and X. Kong, "Adaptive fault-tolerant sliding mode control for high-speed trains with actuator faults under strong winds," *IEEE Access*, vol. 8, pp. 143902–143919, 2020.

[4] S. Long, L. Meng, Y. Wang, J. Miao, and X. Li, "A discrete-space train movement model for a high-speed train under temporary speed restriction," *Mathematical Problems in Engineering*, vol. 2020, no. 1–4, 11 pages, Article ID 5386406, 2020.

[5] T. X. Qin, J. L. Song, and X. W. Dai, "Research on application of improved WFQ algorithm in train network scheduling," *Application Research of Computers*, vol. 36, no. 11, pp. 3302–3305, 2019.

[6] Y. Xiong, H. Huang, S. Y. Li, and Q. Xu, "Research on high-speed train control system based on Ethernet," *Electric Drive for Locomotives*, vol. 4, pp. 41–44, 2019.

[7] M. Li, Y. Chen, A. Zhou, W. He, and X. Li, "Adaptive tracking control for networked control systems of intelligent vehicle," *Information Sciences*, vol. 503, pp. 493–507, 2019.

[8] G. P. Xu, X. K. Zhang, and G. Q. Zhang, "Robust adaptive control for ship course-keeping by considering the communication delay," *Journal of Harbin Engineering University*, vol. 38, no. 1, pp. 59–65, 2017.

[9] G. Chen, X. Wang, C. S. Luo, and S. P. Xiao, "H_∞ sampled-data control of networked traction control system and its application," *Journal of Central South University (Science and Technology)*, vol. 50, no. 1, pp. 91–99, 2019.

[10] T. Liu, S. Jiang, and F. Pan, "Online multi-step prediction for the random delay of the forward channel in networked control system," *Information and Control*, vol. 46, no. 5, pp. 620–626, 2017.

[11] H. F. Chen, P. F. Huang, Z. X. Liu, and Z. Q. Ma, "Time delay prediction for space telerobot system with a modified sparse multivariate linear regression method," *Acta Astronautica*, vol. 166, pp. 330–341, 2019.

[12] T. Liu, S. L. Hao, D. W. Li, W. H. Chen, and Q. G. Wang, "Predictor-based disturbance rejection control for sampled systems with input delay," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 2, pp. 772–780, 2017.

[13] A. P. Batista and F. G. Jota, "Performance improvement of an NCS closed over the internet with an adaptive smith predictor," *Control Engineering Practice*, vol. 71, no. 2, pp. 34–43, 2018.

[14] T. Wang, H. J. Gao, and J. B. Qiu, "A combined adaptive neural network and nonlinear model predictive control for multirate networked industrial process control," *IEEE Transactions on Neural Networks & Learning Systems*, vol. 27, no. 2, pp. 416–425, 2017.

[15] S. F. Li, Z. Z. Qiu, L. P. Fan, and L. N. Zhao, "Modelling and simulation of networked control systems based on improved BP network," *Journal of System Simulation*, vol. 30, no. 6, pp. 288–296, 2018.

[16] A. K. Varma and D. Mitra, "Statistical feature-based SVM wideband sensing," *IEEE Communications Letters*, vol. 24, no. 3, pp. 581–584, 2020.

[17] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, 1999.

[18] Z. D. Tian, X. W. Gao, and K. Li, "Time-delay prediction method of networked control system based on EMD and LS-SVM," *Acta Electronica Sinica*, vol. 42, no. 5, pp. 868–874, 2014.

[19] Z. D. Tian, C. Zhang, S. J. Li, Y. H. Wang, and Y. Sha, "Time-delay prediction based on phase space reconstruction and least squares support vector machine," *Acta Electronica Sinica*, vol. 45, no. 5, pp. 1044–1051, 2017.

[20] Z. Q. Li and K. Yan, "A fast generalized predictive control algorithm for controlling braking process of high speed EMU," *Computer Simulation*, vol. 37, no. 6, pp. 104–110, 2020.

[21] J. Chen, L. Sun, G. Sun, Y. Li, and B. Zhu, "Generalized predictive tracking control of spacecraft attitude based on hyperbolic tangent extended state observer," *Advances in Space Research*, vol. 66, no. 2, pp. 335–344, 2020.

[22] C. L. Zhu, Q. J. Zeng, P. C. Xu, and X. Q. Dai, "Research on optimization design of predictive controller for observational underwater vehicle," *Computer Simulation*, vol. 37, no. 6, pp. 332–337, 2020.

[23] T. Zhang, "Real-time control method for communication network of high-speed EMU based on T-S fuzzy model," *China Railway Science*, vol. 39, no. 3, pp. 93–99, 2018.

[24] B. H. Wang, X. Yan, L. D. Wang, and X. Wang, "Optimization of MVB periodic data scheduling based on load balancing," *China Railway Science*, vol. 38, no. 5, pp. 114–120, 2017.

[25] B. Zhu, S. Ye, P. Wang, K. He, T. Zhang, and Y.-M. Wei, "A novel multiscale nonlinear ensemble leaning paradigm for carbon price forecasting," *Energy Economics*, vol. 70, pp. 143–157, 2018.

[26] Q.-Q. Chen, S.-W. Dai, and H.-D. Dai, "A rolling bearing fault diagnosis method based on EMD and quantile permutation entropy," *Mathematical Problems in Engineering*, vol. 2019, Article ID 3089417, 8 pages, 2019.

[27] S. Esmaeili, H. Sarma, T. Harding, and B. Maini, "A data-driven model for predicting the effect of temperature on oil-water relative permeability," *Fuel*, vol. 236, pp. 264–277, 2019.

[28] T. Zuo, H. Min, Q. Tang, and Q. Tao, "A robot SLAM improved by quantum-behaved particles swarm optimization," *Mathematical Problems in Engineering*, vol. 2018, Article ID 1596080, 11 pages, 2018.

[29] J. Sun, W. Fang, X. Wu, V. Palade, and W. Xu, "Quantum-behaved particle swarm optimization: analysis of individual particle behavior and parameter selection," *Evolutionary Computation*, vol. 20, no. 3, pp. 349–393, 2012.

[30] C. Wang, W. Li, G. Xin, Y. Wang, and S. Xu, "Prediction model of corrosion current density induced by stray current based on QPSO-driven neural network," *Complexity*, vol. 2019, no. 2, 15 pages, Article ID 3429816, 2019.

[31] W. Y. Huang, X. J. Xu, X. B. Pan, Y. J. Sun, and S. Li, "Control strategy of contraction-expansion coefficient in quantum-behaved particle swarm optimization," *Application Research of Computers*, vol. 33, no. 9, pp. 2592–2595, 2016.

[32] A. O. Adewumi and A. M. Arasomwan, "An improved particle swarm optimiser based on swarm success rate for global optimisation problems," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 28, no. 3, pp. 441–483, 2016.

[33] X. J. Li, J. Xu, E. Z. Zhu, and Y. W. Zhang, "A novel computation method for adaptive inertia weight of task scheduling algorithm," *Journal of Computer Research and Development*, vol. 53, no. 9, pp. 1990–1999, 2016.

[34] C. Liu, Z. Zhao, and G. Wen, “Adaptive neural network control with optimal number of hidden nodes for trajectory tracking of robot manipulators,” *Neurocomputing*, vol. 350, pp. 136–145, 2019.

[35] K. Zhang, H. G. Zhang, Y. L. Cai, and R. Su, “Parallel optimal tracking control schemes for mode-dependent control of coupled markov jump systems via integral RL method,” *IEEE Transactions on Automation Ence and Engineering*, vol. 17, no. 3, pp. 1332–1342, 2020.

[36] H. Zhang, Y. Mu, and C. Liu, “Decentralized tracking optimization control for partially unknown fuzzy interconnected systems via reinforcement learning method,” *IEEE Transactions on Fuzzy Systems*, p. 1, 2020.

[37] B. Kiumarsi and F. L. Lewis, “Actor–critic-based optimal tracking for partially unknown nonlinear discrete-time systems,” *IEEE Transactions on Neural Networks & Learning Systems*, vol. 26, no. 1, pp. 140–151, 2017.

[38] H. Xue, Z. P. Shao, Q. L. Fang, and X. J. Liu, “Reinforcement learning based fractional gradient descent RBF neural network control of inverted pendulum,” *Control and Decision*, 2019.

[39] M. Fairbank, S. Li, X. Fu, E. Alonso, and D. Wunsch, “An adaptive recurrent neural-network controller using a stabilization matrix and predictive inputs to solve a tracking problem under disturbances,” *Neural Networks*, vol. 49, no. 1, pp. 74–86, 2014.

[40] T. Dierks and S. Jagannathan, “Online optimal control of nonlinear discrete-time systems using approximate dynamic programming,” *Journal of Control Theory and Applications*, vol. 9, no. 3, pp. 361–369, 2011.

[41] J. Meng, M. Boukhnifer, and D. Diallo, “Comparative study of lithium-ion battery open-circuit-voltage online estimation methods,” *IET Electrical Systems in Transportation*, vol. 10, no. 2, pp. 162–169, 2020.

[42] L. L. Rodrigues, O. A. C. Vilcanqui, E. R. Conde, J. M. Guerero, and A. J. Sguarezi Filho, “Generalized Predictive Control applied to the DFIG power control using state-space model and voltage constraints,” *Electric Power Systems Research*, vol. 182, Article ID 106227, 2020.

[43] J. X. Zhang, Z. T. Shi, X. Yang, and J. Zhao, “Wheel slip tracking control of vehicle based on Elman neural network,” *Journal of Huazhong University of Science and Technology (Natural Science Edition)*, vol. 48, no. 6, pp. 64–69, 2020.

[44] A. Shakeel, T. Tanaka, and K. Kitajo, “Time-series prediction of the oscillatory phase of EEG signals using the least mean square algorithm-based AR model,” *Applied Sciences*, vol. 10, no. 10, p. 3616, 2020.

[45] V. Balaji, M. Balaji, M. Chandrasekaran, M. K. A. A. Khan, and I. Elamvazuthi, “Optimization of PID control for high speed line tracking robots,” *Procedia Computer Science*, vol. 76, pp. 147–154, 2015.