Article

Unstable System Control Using an Improved Particle Swarm Optimization-Based Neural Network Controller

Cheng-Jian Lin ^{*} , Xin-You Lin and Jyun-Yu Jhang

Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung City 411, Taiwan; kamisama0406@gmail.com (X.-Y.L.); o800920@gmail.com (J.-Y.J.)

* Correspondence: cjlin@ncut.edu.tw; Tel.: +886-4-23924505

Received: 25 September 2019; Accepted: 3 November 2019; Published: 7 November 2019

Abstract: In this study, an improved particle swarm optimization (IPSO)-based neural network controller (NNC) is proposed for solving a real unstable control problem. The proposed IPSO automatically determines an NNC structure by a hierarchical approach and optimizes the parameters of the NNC by chaos particle swarm optimization. The proposed NNC based on an IPSO learning algorithm is used for controlling a practical planetary train-type inverted pendulum system. Experimental results show that the robustness and effectiveness of the proposed NNC based on IPSO are superior to those of other methods.

Keywords: neural networks; chaos; nonlinear control; inverted pendulum system; particle swarm optimization

1. Introduction

The multilayer neural network is one of the most popular methods for solving real-time signal processing, system control, and signal prediction [1–3] problems. In general, determining the number of hidden levels and neurons for solving a problem is difficult. However, within the provided training period, a fixed structure in an artificial neural network may not provide satisfactory performance. A large neural network has some redundant and complex connections, whereas a small neural network cannot provide the optimal performance because of its limited information processing abilities. Therefore, a learning method that simultaneously tunes the network structure and parameters of a neural network was proposed.

The parameter training of a neural network design is the primary concern in various applications. A backpropagation (BP) learning method is commonly adopted and is a powerful training technique for feedforward neural networks. Since the gradient descent algorithm is used to minimize the error function in the BP training process, this algorithm can quickly trap into the local minimum, and finding a global solution is difficult. Although neural networks have a beneficial topology, the aforementioned drawbacks result in suboptimal performance. Therefore, several algorithms were used to adjust parameters in neural networks and to find the appropriate network structure. Recently, several researchers [4–9] have used various evolutionary algorithms to determine the parameters of neural networks. In [4–7], several researchers have adopted genetic algorithms (GAs), artificial bee colony (ABC), and gray wolf optimization (GWO) to adjust parameters of neural networks. However, search in GAs is extremely tedious, which is the primary disadvantage of GAs. Similar to GAs, particle swarm optimization (PSO) [8,9] is a robust numerical optimization algorithm. PSO has been successfully used for solving several optimization problems. PSO is a simple, rapid-convergence, and easy-implementation algorithm.

Chaos [10] has finite unstable dynamic nonlinear behavior and its sensitivity is shown according to initial conditions. Tang et al. [11] proposed a hierarchical structure which is based on a biological DNA structure. A biological DNA structure comprises structural genes (SGs) and regulatory sequences (RSs). SGs encode a polypeptide or RNA. RS is the leader of the beginning and end of the SG—the starting point for replication or reorganization.

Recently, several researchers [12–15] have used neural networks and evolutionary computation methods for the inverted pendulum control. Ping [12] used a neural network for adjusting 10 motion functions to complete the control of motion behavior. Al-Araji [13] proposed an online self-tuning hybrid intelligent algorithm based on the culture-bees algorithm to control the inverted pendulum. Mohamed et al. [14] adopted a hybrid of gravitational search algorithm and genetic algorithm to control the inverted pendulum. Su et al. [15] presented the parallel distributed compensation law and the event-triggering fuzzy controller for the nonlinear inverted pendulum system. However, the above-mentioned methods adopted a fixed fuzzy system or neural network architecture and then adjusted their parameters by evolutionary computation methods. The architecture of the fuzzy system or neural network is determined by trial and error.

In this study, an efficient improved particle swarm optimization (IPSO)-based neural network controller (NNC) was proposed for controlling a planetary train-type inverted pendulum system. In the proposed hierarchical neural network, the structure of a multi-level control gene in a layered approach is proposed. The value of the first-level control gene controls the activation of the parameter gene. The control genes are adjusted using a GA and the parameter genes are tuned using PSO. The proposed IPSO learning algorithm is used to automatically determine the neural network structure and adjust the NNC parameters to avoid over-fitting and to enhance global search capabilities. The experimental results show that the proposed IPSO-based NNC has a better control performance of the set-point and periodic square commands than other methods.

The major contributions of the study are illustrated as follows:

1. The proposed improved particle swarm optimization can automatically determine the neural network structure and adjust the parameters of a neural network controller.
2. A hierarchical chromosome architecture is proposed. This architecture has a multi-level control gene, which consists of the control genes and the parameter genes.
3. The chaotic local search method is proposed to improve the searching space and avoid trapping into the local minimum.

The remainder of the paper is organized as follows: Section 2 introduces the dynamics analysis of the planetary wheel inverted pendulum. The structure of the hierarchical neural network is presented in Section 3. Section 4 illustrates the optimization method of PSO and the local search method of the chaotic method. The experimental results of a planetary train-type inverted pendulum control are shown in Section 5. Section 6 presents the conclusions.

2. Dynamic System Analysis of Planetary Train-Type Inverted Pendulum

The mathematical model of the derived planetary train inverted pendulum system [16] is based on initial conditions and input signals to predict the dynamic behavior of the system. The dynamic behavior of an unstable system was used to determine the size of an actuator to select amplifier power, design mechanisms, and tune the controller based on the computer simulation. To explain the dynamic and kinematic relations, Figure 1 shows a pendulum, sun gear, and planetary gear.

Figure 1. The structure of a planetary train-type inverted pendulum.

The kinematic relationship between the planetary gears, the sun gear, and the pendulum lies in the mutual movement between the two. First, when the pendulum is stationary ($\dot{\theta}_0 = 0$), the ratio between the planetary gear motion and sun gear motion is shown as follows:

$$e = \frac{\dot{\theta}_1'}{\dot{\theta}_2} = -\frac{N_2}{N_1} = -\frac{r_2}{r_1} \quad (1)$$

where θ_1 is the sun gear angle, θ_2 is the planetary gear angle, N_1 and N_2 present the number of sun gear and planetary gear, respectively, and r_1 and r_2 are the radii of the sun gear and planetary gear, respectively. Thus, the angular velocity of the planetary gear is derived as follows:

$$\dot{\theta}_1' = e\dot{\theta}_2 \quad (2)$$

Assume that the center gear is stationary ($\dot{\theta}_2 = 0$) and the pendulum and planet gear are allowed to rotate. The planetary gear center velocity is derived as follows:

$$v = (r_1 + r_2)\dot{\theta}_0 \quad (3)$$

The angular velocity of planetary gear ($\dot{\theta}_1''$) is as follows:

$$\dot{\theta}_1'' = \frac{v}{r_1} = \frac{r_1 + r_2}{r_1}\dot{\theta}_0 \quad (4)$$

Equations (3) and (4) are combined and derived as follows:

Lagrangian mechanics were used to obtain relations among input motor torque τ_2 , center gear θ_2 , and output of the pendulum θ_0 . This method can be used to analyze mechanisms in system methods. The potential energy and kinetic of each movable part are derived as follows:

$$\begin{aligned}
 K_2 &= \frac{1}{2} I_2 \dot{\theta}_2^2 \\
 P_2 &= 0 \\
 K_1 &= \frac{1}{2} m_1 [(r_1 + r_2) \dot{\theta}_0]^2 + \frac{1}{2} I_1 \dot{\theta}_1^2 \\
 P_1 &= m_1 g (r_1 + r_2) \cos \theta_0 \\
 K_0 &= \frac{1}{2} \left(\frac{1}{3} m_0 l^2 \right) \dot{\theta}_0^2 \\
 P_0 &= m_0 g \frac{1}{2} \cos \theta_0
 \end{aligned} \tag{5}$$

where l is the pendulum length, K_0 , K_1 , and K_2 represent the kinetic energy of the pendulum, planetary gear, and center gear, respectively; P_0 and m_0 denote the potential energy and the mass of the pendulum, respectively; P_1 , I_1 and m_1 and represent the potential energy, the moment of inertia, and the mass of the planetary gear, respectively; and P_2 and I_2 are the potential energy and the moment of inertia of the center gear, respectively.

From Equation (5), we derive the Lagrangian as follows:

$$\begin{aligned}
 L &= \sum K - \sum P \\
 &= \left[\frac{1}{2} m_1 (r_1 + r_2)^2 + \frac{1}{6} m_0 l^2 \right] \dot{\theta}_0^2 \\
 &\quad + \frac{1}{2} I_1 \left(e \dot{\theta}_2 + \frac{r_1 + r_2}{r_1} \dot{\theta}_0 \right)^2 + \frac{1}{2} I_2 \dot{\theta}_2^2 \\
 &\quad - \left[m_1 (r_1 + r_2) + \frac{1}{2} m_0 \right] g \cos \theta_0
 \end{aligned} \tag{6}$$

where θ_0 and θ_2 are two independent variables, which are generalized coordinates. Moreover, two dynamic equations are derived using the Lagrange equation as follows:

$$\begin{aligned}
 \tau_2 &= \frac{d}{dt} \left(\frac{\partial L}{\partial \dot{\theta}_2} \right) - \frac{\partial L}{\partial \theta_2} \\
 &= (e^2 I_1 + I_2) \ddot{\theta}_2 + e \left(\frac{r_1 + r_2}{r_1} \right) I_1 \ddot{\theta}_0
 \end{aligned} \tag{7}$$

$$\begin{aligned}
 \tau_0 &= \frac{d}{dt} \left(\frac{\partial L}{\partial \dot{\theta}_0} \right) - \frac{\partial L}{\partial \theta_0} \\
 &= \left[\left(\frac{r_1 + r_2}{r_1} \right)^2 I_1 + m_1 (r_1 + r_2)^2 + \frac{1}{3} m_0 l^2 \right] \ddot{\theta}_0 \\
 &\quad + e \left(\frac{r_1 + r_2}{r_1} \right) I_1 \ddot{\theta}_2 - \left[m_1 (r_1 + r_2) + \frac{1}{2} m_0 \right] g \sin \theta_0
 \end{aligned} \tag{8}$$

Equation (8) shows that the pendulum is not applied with external torque, therefore zero is assigned to the variable τ_0 .

3. Structure of a Hierarchical Neural Network

In this section, the structure of a hierarchical neural network is introduced. Figure 2 shows the structure of a multi-level control gene in a layered approach. The value of the first-level control gene controls the activation of the parameter gene. To illustrate the status of the control genes, each open control gene was assigned a value of 1 and each of the closed genes was designated as 0. Although the inactive genes were turned off for a period of time, they still exist in the chromosome. This is called hierarchical chromosome architecture.

Figure 2. An example of chromosome representation.

Figure 2 shows that a chromosome comprises six control genes in the form of bits and six parametric genes in the form of integers, where X represents a chromosome with a length of 2 and a solution of 5 and 8. In addition, multi-layered chromosomes are formulated by increasing the level of stratification in the chromosome. Figure 3 illustrates an example of a three-level chromosome.

Figure 3. A three-level chromosome.

A hierarchical chromosome representation is shown in Figure 4. Each chromosome contains a control gene and a connection gene. A control gene in the form of bits activates layers and neurons. A connection gene, which is a real value representation, has connection weights and neuron biases. Figure 5 presents the structure of a hierarchical neural network, where w_1 , w_2 , and w_3 represent the weights of neural networks.

Figure 4. Hierarchical structure of the chromosome.

This particular treatment ensures that the structural chromosome contains both active and inactive genes. Inactive genes remain in the chromosomal structure and can continue to exist in the next generation. In the chromosome, the natural genetic variation avoids falling into local optimization and leads to early convergence. Therefore, it balances the use of its accumulated knowledge with the exploration of new search spaces. In addition, this structure allows for greater genetic variation in the chromosome and allows multiple simultaneous genetic changes to maintain high viability.

Figure 5. Structure of a hierarchical neural network.

4. Proposed Improved Particle Swarm Optimization

This section introduces the review of particle swarm optimization, local search of chaos, and the IPSO algorithm.

4.1. Review of Particle Swarm Optimization (PSO)

In PSO [17,18], the trajectory of each particle is based on its flight experience and search space to update the speed of each particle. The position and speed vector of the i th particle are represented as $X_i = (x_{i1}, x_{i2}, x_{i3}, \dots, x_{id})$ and $V_i = (v_{i1}, v_{i2}, v_{i3}, \dots, v_{id})$, respectively. The optimal position of each particle is defined as $P_i = (p_{i1}, p_{i2}, p_{i3}, \dots, p_{id})$, and the optimal position of all particles is defined as $P_g = (p_{g1}, p_{g2}, p_{g3}, \dots, p_{gd})$. The new speed and position of the particle used for the next evaluation is derived as follows:

$$v_{vid}^{k+1} = \omega \times v_{vid}^k + c_1 \times \text{rand}(\cdot) \times (P_{id} - x_{id}^k) + c_2 \times \text{rand}(\cdot) \times (P_{gd} - x_{id}^k) \quad (9)$$

$$x_{id}^{k+1} = x_{id}^k + v_{id}^{k+1} \quad (10)$$

where c_1 and c_2 are acceleration coefficients. Moreover, two separately generated random numbers (i.e., $\text{rand}(\cdot)$) are between 0 and 1. The first term in Equation (9) illustrates the speed of the previous generation and provides momentum for the particle search space. The second term in Equation (9) is a cognitive component that facilitates the movement of particles toward their own local optimal positions. The third term in Equation (9) illustrates the social components and synergistic effects of the particle's global optimal solution.

4.2. Local Search Using Chaotic Method

To improve the searching space and avoid falling into the local minimum, the chaotic method is used in PSO. A logistic equation in [19] was used for developing hybrid PSO and is defined as follows:

$$x_{n+1} = \mu \times x_n(1 - x_n), \quad 0 \leq x_0 \leq 1 \quad (11)$$

where μ is a control parameter. Although Equation (11) is deterministic, it exhibits chaotic dynamics for $\mu = 4$ and $x_0 \in \{0, 0.25, 0.5, 0.75, 1\}$. This equation shows that the initial conditions are very important dependencies on the sensitivity of the chaotic sequence. Subtle differences in the initial values of chaotic variables can cause significant differences in their long-term trajectories. In the search space,

the trajectory of chaotic variables can be ergodically traversed. The chaotic local search (CLS) process is defined as follows:

$$cx_i^{(k+1)} = 4cx_i^{(k)}(1 - cx_i^{(k)}), \quad i = 1, 2, \dots, n \quad (12)$$

In Equation (12), k is k th iteration and cx_i is the i th chaotic variable. $cx_i^{(k)}$ is set to between 0 and 1 for $cx_i^{(0)} \in (0, 1)$ and $cx_i^{(0)} \notin \{0.25, 0.5, 0.75\}$.

The CLS process is as follows:

Step 1: Decision variables $x_i = (x_{i1}, x_{i2}, x_{i3}, \dots, x_{in})$ are located in the interval $(x_{\min,i}, x_{\max,i})$. The value of $cx_i^{(k)}$ is set between 0 and 1 and is shown as follows:

$$cx_i^{(k)} = \frac{x_i^{(k)} - x_{\min,i}}{x_{\max,i} - x_{\min,i}} \quad (13)$$

Step 2: Using Equation (12), determine the chaotic variable $cx_i^{(k+1)}$ in the next iteration.

Step 3: The conversion of the chaotic variable $cx_i^{(k+1)}$ turns into the decision variables $dx_i^{(k)}$

$$dx_i^{(k)} = x_{\min,i} + cx_i^{(k+1)}(x_{\max,i} - x_{\min,i}) \quad (14)$$

Step 4: The evaluation of the new solution is found by using decision variable $x_i^{(k+1)}$. If the new solution is superior to the previous solution, then the previous solution is updated as follows:

$$x_i^{(k+1)} = dx_i^{(k)} \quad (15)$$

Step 5: Go to $i + 1$.

4.3. Proposed Improved Particle Swarm Optimization

Figure 6 shows the parameter adjustment of the chromosome. The control genes are adjusted using a GA [20,21], whereas the parameter genes are tuned using PSO. The GA is a stochastic search procedure based on natural selection with the use of reproduce, crossover, and mutation operations.

Figure 6. The adjustable parameters of the chromosome.

The reproduction operation is used to copy individual strings based on the evaluated fitness values. Each individual is assigned a fitness value using a fitness assignment method where a large value indicates satisfactory fitness. In this study, a roulette-wheel selection method [22] was used. The individuals with the optimal performance in the first half of the population [23] are directly reproduced in the next generation. By performing crossover operations for individuals in the first half of the parent generation, the other half of the next generation is generated. For each generation, the reproduction operation keeps the same chromosomes in the first half of the population.

Figure 7 shows the two-point crossover operation. The figure shows the generation of new individuals by exchanging site values between sites selected by parents. Figure 7 shows the two-point crossover operation. The figure shows the generation of new individuals by exchanging site values between the selected sites from parents. When the crossover operation is performed, the unsatisfactory individuals are replaced by the newly generated offspring.

Figure 7. Two-point crossover operation.

During mutation, the allele of a gene is randomly altered. Figure 8 shows the mutation of an individual. In contrast to conventional symbiotic evolution [22], a mutation value is created using a constant range. In mutation, new genetic materials are used in the population. Because mutation is a random search operation, it must be used sparingly.

Figure 8. Mutation operation.

By replicating better control genes in this generation and using them for crossover and mutation operations, the better control genes in the next generation are produced. Therefore, the GA is used to develop a hierarchical neural network. Figure 9 shows the flowchart of the proposed IPSO algorithm. The IPSO algorithm is illustrated as follows:

Step 1: Initially, randomly generate P particles forming the swarm.

Step 2: Evaluate the fitness values from each particle.

$$f_i = \alpha \times RMSE + \beta \times W_R \quad (16)$$

where $RMSE$ and W_R represents the root mean square error and the weight ratio of connected neurons, respectively; the evaluation ratio α and β are set as 0.7 and 0.3, respectively.

Step 3: Update the local best particle and the global best particle.

Step 4: Adjust control genes by using reproduce, crossover, and mutation operations.

Step 5: Use Equations (9) and (10) for updating the parameter genes.

Step 6: Use CLS to update the optimal particle.

Step 7: If the learning process has been completed, the best individual is derived. Otherwise, go back to Step 2.

Figure 9. Flowchart of the proposed improved particle swarm optimization (IPSO) algorithm.

5. Experimental Results

The performance of IPSO-based NNC in practically unstable nonlinear control applications is evaluated using the VisSim environment with a RT-DAC4/PCI motion control card and hardware-in-the-loop. VisSim is a simulation of complex nonlinear dynamic systems in a Windows environment. In this study, based on the VisSim environment, the IPSO-based NNC was implemented in a planetary train inverted pendulum control system.

Figure 10 shows the block diagram of the control system. The encoder was used for sensing the pendulum angle. The angle signal was used as a feedback signal. The motor torque is used to control the swing angle of the pendulum in the equilibrium state to indicate that the control is completed.

Figure 10. Block diagram of the control system.

This study compared the proposed IPSO-based NNC with a proportional–integral–derivative (PID) controller, PSO-based NNC, and GA-based NNC. The PID controller is shown in the following:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de}{dt} \quad (17)$$

where $u(t)$ represents the controller's output and $e(t)$ represents the difference between the controller's desired and measured values. The control gains obtained through extensive experimental experience are $K_p = 830$, $K_i = 0$ and $K_d = 0.3284$.

5.1. Control of Different Reference Trajectories

To validate the control abilities of the proposed IPSO-based NNC, the set-point and periodic square commands are used as different reference trajectories in this experiment. The initial parameters of the particle swarm optimization are the number of particles, the single inertia weight w , acceleration

constants C_1 and C_2 , and number of generations (see Table 1). The different inertia weights will lead to the different performance of the proposed algorithm. Thus, in this study, the inertia weight values were set as 0.3, 0.4, 0.6, 0.8, and 0.9. The performance evaluations of the inertia weight ($\omega = 0.8$) were more efficient than that of the inertia weights ($\omega = 0.3, 0.4, 0.6$ and 0.9). Therefore, the inertia weight ($\omega = 0.8$) was adopted in this study.

Table 1. Initial parameters of PSO.

Number of Particles	Inertia Weight (ω)	C_1	C_2	Generation
30	0.3~0.9	2	2	3000

Figure 11 presents the real control system of a planetary train inverted pendulum. The performance measurement of the controller comprised set-point regulation and a square command tracking capability. In the set-point command experiment, the proposed method was used to control the inverted pendulum for following the set-point reference. In this study, the set-point reference is equal to zero. Figure 12a–d shows the comparison results of the regulation of IPSO-based NNC, PID controller, PSO-based NNC, and GA-based NNC. In order to validate the performance of set-point regulation, the sum of absolute error (SAE) are defined in the following:

$$SAE_{\theta_0} = \sum |\theta_0^{ref} - \theta_0| \text{ and } SAE_{\dot{\theta}_0} = \sum |\dot{\theta}_0^{ref} - \dot{\theta}_0| \quad (18)$$

where θ_0 and θ_0^{ref} represent the actual angle of the pendulum and the referred angle of the pendulum, respectively, and $\dot{\theta}_0$ and $\dot{\theta}_0^{ref}$ denote the actual speed of the pendulum and the referred speed of the pendulum, respectively. Table 1 shows the experimental results of the sum of the absolute error for the pendulum angle (SAE_{θ_0}) and pendulum speed ($SAE_{\dot{\theta}_0}$).

In the periodic square wave signal command experiment, we used a square wave with an amplitude of ± 0.02 and a frequency of 0.5 Hz to evaluate the tracking ability of the proposed method. Figure 13a–d show the regulation performance of the proposed IPSO-based NNC, PID controller, PSO-based NNC and GA-based NNC, respectively. In this experiment, a sampling rate of 0.1 s is used. Table 2 presents the summary of the experimental results. In this table, the comparison results of various controllers in the set-point and periodic square commands show that the proposed IPSO-based NNC outperforms the other controllers.

Figure 11. A real control system of planetary train inverted pendulum.

(a)

(b)

Figure 12. Cont.

(c)

(d)

Figure 12. Regulation comparisons of (a) proposed IPSO-based NNC, (b) PID controller, (c) PSO-based NNC, and (d) GA-based NNC.

(a)

(b)

(c)

Figure 13. Cont.

(d)

Figure 13. Tracking comparisons of (a) proposed IPSO-based NNC, (b) PID controller, (c) PSO-based NNC, and (d) GA-based NNC.

Table 2. Comparison results of various controllers in the set-point and periodic square commands.

Reference Trajectories	Methods	Proposed IPSO-Based NNC		PID Controller		PSO-Based NNC		GA-Based NNC	
		SAE_{θ_0}	$SAE_{\dot{\theta}_0}$	SAE_{θ_0}	$SAE_{\dot{\theta}_0}$	SAE_{θ_0}	$SAE_{\dot{\theta}_0}$	SAE_{θ_0}	$SAE_{\dot{\theta}_0}$
The set-point command		32.1526	67.4174	34.0881	73.3770	33.1843	69.1064	33.3609	69.5228
The periodic square command		32.5641	71.2465	34.4442	80.4085	33.6248	72.8291	33.6294	73.1649

5.2. Robustness Test of IPSO-Based NNC

To validate the usefulness of the proposed control system, two different cases, including the influence of impulse noise (i.e., add unknown impulse noise to the plant output) and effect of change in plant dynamics (i.e., add dynamic value to the plant input), are used in this experiment. First, one impulse noise value, 0.1 rad, is added to the plant output at the 10th second when a set-point of 0.04 rad is adopted in this case. Experimental results show that the proposed method recovers quickly when pulse noise occurs, as illustrated in Figure 14a. Second, one general characteristic of many industrial-control processes is that their parameters tend to change in an unpredictable way. In this experiment, $0.5 \cdot u$ ($k=0.005$) is added to the plant input between the 7th second and the 15th second, in order to test the robustness of the proposed controller. Figure 14b shows the recovering results of the proposed method.

(a)

(b)

Figure 14. (a) The effect of one impulse noise, (b) the effect of a change in plant dynamics.

6. Stability Analysis of IPSO-Based NNC

Consider the n th-order nonlinear system (Figure 15) that can be expressed in the canonical form

$$\begin{aligned} x^{(n)} &= f(x, \dot{x}, \dots, x^{(n-1)}) + bu, \\ y &= x, \end{aligned} \quad (19)$$

where f is an unknown continuous function, b is the control gain (For simplicity $b = 1$ is taken in the subsequent development), and $u \in R$, $y \in R$ is the input and output of the plant, respectively. We assume that the state vector $X = (x_1, x_2, \dots, x_n)^T = (x, \dot{x}, \dots, x^{(n-1)})^T \in R^n$ is available for measurement. The

control objective is to force the plant state vector y to follow a specified desired trajectory, y^d , under the constraint that all signals involved must be bounded.

Figure 15. The overall scheme of stable IPSO-based NNC.

Defining the tracking error vector, $e = y - y^d$ the problem is thus to obtain a feedback control law, $u = u_c + u_s$, where u_c and u_s are control outputs of the NNC controller and the supervisor, respectively.

The global stability of the control system is a basic requirement for solving nonlinear control problems. Because the general evolution algorithm has become characteristic of random search, some search points may cause learning processes to be unstable. In this study, the supervisor $u_s(t)$ is designed (Figure 15) to guarantee global stability of the closed-loop system in the sense that the error state variables must be uniformly bounded, i.e., $|e(t)| \leq D < \infty$ for all $t \geq 0$, where D is a design parameter specified by the designer.

Let an error metric be defined $r(t) = M^T e(t)$, where $M^T = [\alpha^{(n-1)}, (n-1)\alpha^{(n-2)}, \dots, 1] \in R^n$. $r(t) = 0$ defines a time-varying hyperplane in R^n in which the tracking error vector decays exponentially to zero. The time derivative of the error metric can be expressed as:

$$\dot{r}(t) = -\dot{y}^d(t) + M_v^T e(t) - f(x(t)) + u(t) \quad (20)$$

where

$$M_v^T = [0, \alpha^{(n-1)}, (n-1)\alpha^{(n-2)}, \dots, (n-1)\alpha] \in R^n \quad (21)$$

Considering $V = \frac{1}{2}r^2(t)$ as the Lyapunov function candidate. According to [24], we can derive the results as follows:

$$\begin{aligned} \dot{V}(t) &= r(t)\dot{r}(t) = r(t)[-y^d + M_v^T e(t) - f(x(t)) + u_c(t) + u_s(t)] \\ &\leq r(t)[-y^d + M_v^T e(t) - |f(x(t))| + u_c(t) + u_s(t)] \end{aligned} \quad (22)$$

If condition $|f(x(t))| \leq C$ ($C > 0$ is a constant) is assumed, the supervisory $u_s(t)$ can be expressed as:

$$u_s(t) = k_s[-k_d r(t) + y^d(t) - M_v^T e(t) - \text{sgn}(r(t))C - u_c(t)] \quad (23)$$

where $k_d > 0$ is a constant; $k_s = 1$ if $|e(t)| \geq D$, and $k_s = 0$ if $|e(t)| < D$. Because all terms in Equation (23) can be determined, the supervisory control $u_s(t)$ can be implemented. Then, substitute Equation (23) into Equation (22) and consider the $k_s = 1$ case, we can derive the results as follows:

$$\begin{aligned}\dot{V}(t) &\leq r(t) \left[-k_d r(t) + |f(x(t))| - \operatorname{sgn}(r(t))C \right] \\ \dot{V}(t) &\leq r(t) \left[-k_d r(t) + |f(x(t))| - \operatorname{sgn}(r(t))C \right] \\ &\leq -k_d r^2(t) - |r(t)| \left[C - |f(x(t))| \right] \\ &\leq -k_d r^2(t) \leq 0\end{aligned}\quad (24)$$

Therefore, using the supervisory $u_s(t)$ in Equation (23), we can derive $V(t) \rightarrow 0$, i.e., $r(t) \rightarrow 0$, which in turn implies $|e(t)| \leq D$.

7. Conclusions

This study proposed an IPSO-based NNC for an unstable nonlinear system control. The proposed method can automatically determine the number of hidden levels and neurons and use the chaos method to enhance the ability of PSO to optimize neural network parameters. The proposed IPSO-based NNC was successfully applied to the control system of a planetary train-type inverted pendulum. Experimental results show that the proposed IPSO-based NNC has a smaller SAE than PID controllers, PSO-based NNC, and GA-based NNC. Moreover, the proposed method achieved 3.2% and 2.4% higher performance, respectively, in SAE_{θ_0} and $SAE_{\dot{\theta}_0}$ than the traditional PSO method.

The trained NNC controller can successfully control an unstable nonlinear system, but the weights of the NNC have no physical meaning. Future works will design a new neural fuzzy network which has learning ability, and its fuzzy rules will have a similar thinking logic to that of humans.

Author Contributions: Revised manuscript, C.-J.L. and X.-Y.L.; conceptualization and methodology, C.-J.L., X.-Y.L. and J.-Y.J.; software, C.-J.L. and X.-Y.L.; writing—original draft preparation, C.-J.L.

Funding: This research was funded by Ministry of Science and Technology of the Republic of China, Taiwan grant number MOST 108-2634-F-005-001.

Conflicts of Interest: The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

1. Maksimenko, V.A.; Kurkin, S.A.; Pitsik, E.N.; Musatov, V.Y.; Runnova, A.E.; Efremova, T.Y.; Hramov, A.E.; Pisarchik, A.N. Artificial Neural Network Classification of Motor-Related EEG: An Increase in Classification Accuracy by Reducing Signal Complexity. *Complexity* **2018**, 2018. [\[CrossRef\]](#)
2. Yang, X.C.; Yung, M.H.; Wang, X. Neural-network-designed pulse sequences for robust control of singlet-triplet qubits. *Phys. Rev. A* **2018**, 97, 042324. [\[CrossRef\]](#)
3. Eggenberger, K.; Lindauer, M.; Hutter, F. Neural Networks for Predicting Algorithm Runtime Distributions. *Int. Jt. Conf. Artif. Intell.* **2018**, 1442–1448. [\[CrossRef\]](#)
4. Waghulde, N.P.; Patil, N.P. Genetic Neural Approach for Heart Disease Prediction. *Int. J. Adv. Comput. Res.* **2014**, 4, 778–784.
5. Zhining, Y.; Yunming, P. The Genetic Convolutional Neural Network Model Based on Random Sample. *Int. J. U E Serv. Sci. Technol.* **2015**, 8, 317–326.
6. Li, L.; Lin, C.J.; Huang, M.L.; Kuo, S.C.; Chen, Y.-R. Mobile Robot Navigation Control Using Recurrent Fuzzy CMAC Based on Improved Dynamic Artificial Bee Colony. *Adv. Mech. Eng.* **2016**, 8, 1–10. [\[CrossRef\]](#)
7. Turabieh, H. A Hybrid ANN-GWO Algorithm for Prediction of Heart Disease. *Am. J. Oper. Res.* **2016**, 6, 136–146. [\[CrossRef\]](#)
8. Chen, J.F.; Do, Q.H.; Hsieh, H.N. Training Artificial Neural Networks by a Hybrid PSO-CS Algorithm. *Algorithms* **2015**, 8, 292–308. [\[CrossRef\]](#)

9. Jhang, J.Y.; Lin, C.J.; Lin, C.T.; Young, K.Y. Navigation Control of Mobile Robots through an Interval Type-2 Fuzzy Controller Based on Dynamic-Group Particle Swarm Optimization. *Int. J. Control Autom. Syst.* **2018**, *16*, 2446–2457. [CrossRef]
10. Liu, B.; Wang, L.; Jin, Y.H.; Tang, F.; Huang, D.X. Improved Particle Swarm Optimization Combined with Chaos. *Chaos Solitons Fractals* **2005**, *25*, 1261–1271. [CrossRef]
11. Tang, K.S.; Man, K.F.; Liu, Z.F.; Kwong, S. Minimal Fuzzy Memberships and Rules Using Hierarchical Genetic Algorithms. *IEEE Trans. Ind. Electron.* **1998**, *45*, 162–169. [CrossRef]
12. Ping, Z. Tracking problems of a spherical inverted pendulum via neural network enhanced design. *Neurocomputing* **2013**, *106*, 137–147. [CrossRef]
13. Al-Araji, A.S. An Adaptive Swing-up Sliding Mode Controller Design for a Real Inverted Pendulum System Based on Culture-Bees Algorithm. *Eur. J. Control* **2019**, *45*, 45–56. [CrossRef]
14. Magdy, M.; Marhomy, A.E.; Attia, M.A. Modeling of inverted pendulum system with gravitational search algorithm optimized controller. *Ain Shams Eng. J.* **2019**, *10*, 129–149. [CrossRef]
15. Su, X.; Xia, F.; Liu, J.; Wu, L. Event-triggered fuzzy control of nonlinear systems with its application to inverted pendulum systems. *Automatica* **2018**, *94*, 236–248. [CrossRef]
16. Cho Chieh Tech. Enterprise Ltd. 2015. Available online: <http://www.chochieh.com.tw> (accessed on 8 October 2018).
17. Chiou, J.S.; Tran, H.K.; Shieh, M.Y.; Nguyen, T.N. Particle Swarm Optimization Algorithm Reinforced Fuzzy Proportional-integral-derivative for a Quadrotor Attitude Control. *Adv. Mech. Eng.* **2016**, *8*, 1–7. [CrossRef]
18. Lee, H.; Kim, K.; Kwon, Y.; Hong, E. Real-Time Particle Swarm Optimization on FPGA for the Optimal Message-Chain Structure. *Electronics* **2018**, *7*, 274. [CrossRef]
19. Tsai, J.T.; Liu, T.K.; Chou, J.H. Hybrid Taguchi-Genetic Algorithm for Global Numerical Optimization. *IEEE Trans. Evolut. Comput.* **2004**, *8*, 365–377. [CrossRef]
20. Idrissi, J.; Hassan, R.; Youssef, G.; Mohamed, E. Genetic Algorithm for Neural Network Architecture Optimization. In Proceedings of the 2016 3rd International Conference on Logistics Operations Management (GOL), Fez, Morocco, 23–25 May 2016. [CrossRef]
21. Islam, B.; Baharudin, Z.; Raza, M.Q.; Nallagownden, P. Optimization of Neural Network Architecture Using Genetic Algorithm for Load Forecasting. In Proceedings of the 2014 5th International Conference on Intelligent and Advanced Systems (ICIAS), Kuala Lumpur, Malaysia, 3–5 June 2014. [CrossRef]
22. Kumar, R.; Jyotishree. Blending Roulette Wheel Selection & Rank Selection in Genetic Algorithms. *Int. J. Mach. Learn. Comput.* **2012**, *2*, 365–370.
23. Lin, H.Y.; Lin, C.J.; Huang, M.L. Optimization of Printed Circuit Board Component Placement Using an Efficient Hybrid Genetic Algorithm. *Appl. Intell.* **2016**, *45*, 622–637. [CrossRef]
24. Wang, L.X. Stable adaptive fuzzy control of nonlinear systems. *IEEE Trans. Fuzzy Syst.* **1993**, *1*, 146–155. [CrossRef]