Energies **2014**, *7*, 191-209; doi:10.3390/en7010191

OPEN ACCESS

energies

ISSN 1996-1073

www.mdpi.com/journal/energies

Article

Optimal PID Controller Design Based on PSO-RBFNN for Wind Turbine Systems

Jau-Woei Perng *, Guan-Yan Chen and Shan-Chang Hsieh

Department of Mechanical and Electro-Mechanical Engineering, National Sun Yat-sen University, 70 Lienhai Road, Kaohsiung 80424, Taiwan; E-Mails: d013020002@student.nsysu.edu.tw (G.-Y.C.); m013020114@student.nsysu.edu.tw (S.-C.H.)

* Author to whom correspondence should be addressed; E-Mail: jwperng@faculty.nsysu.edu.tw; Tel.: +886-07-525-2000 (ext. 4281); Fax: +886-07-525-4299.

Received: 17 September 2013; in revised form: 27 November 2013 / Accepted: 31 December 2013 / Published: 7 January 2014

Abstract: A strategy was proposed to determine the optimal operating point for the proportional-integral-derivative (PID) controller of a wind turbine, and identify the stability regions in the parameter space. The proposed approach combined particle swarm optimization (PSO) and radial basis function neural network (RBFNN) algorithms. These intelligent algorithms are artificial learning mechanisms that can determine the optimal operating points, and were used to generate the function representing the most favorable operating $k_p - k_i$ parameters from each parameter of k_d for the stability region of the PID controller. A graphical method was used to determine the 2D or 3D vision boundaries of the PID-type controller space in closed-loop wind turbine systems. The proposed techniques were demonstrated using simulations of a drive train model without time delay and a pitch control model with time delay. Finally, the 3D stability boundaries were determined the proposed graphical approach with and without time delay systems.

Keywords: wind turbine; proportional-integral-derivative (PID) control; particle swarm optimization (PSO); radial basis function (RBF); stability boundary; time delay

---

1. Introduction

In recent years, fossil fuel supplies have decreased, and average temperatures have increased. Because of the dramatic effects of these events, numerous renewable energy studies have been

conducted. Wind energy has received considerable attention, and has become increasingly widespread. Conventional wind turbines use wind-energy capturing systems. Pitch control, which is typically required to achieve a stable power output when the wind speed is above the rated wind speed, is achieved by adjusting the pitch angle of the wind turbine blades to limit the output power, protecting wind turbine gearbox and generator. Therefore, pitch control designs for variable-speed turbines are increasingly critical.

The proportional-integral-derivative (PID) controller is the most common closed-loop control system, and using a PID is the easiest and simplest way to design the pitch control system [1,2]. In addition to typical PID control systems, studies have proposed alternative methods for controlling the pitch angle. Recent studies have attempted to determine the stability region of PID control systems. In [3], stable synthesis of a PI-based pitch controller was proposed. A graphical approach was used to determine the stability region and set the parameters of the controller to achieve an arbitrary-order time delay system. A 3D stabilizing domain must be attained in the parameter space of a PID controller to execute system-stabilization analyses and computations [4–8]. Using the 3D stability region method to determine the operating point range of the controller parameters is simple and intuitive. Therefore, the controller design can be reliably modeled and readily analyzed. An alternate way to regulate the pitch angle in wind turbine generators is setting various operating points for the control system. A robust discrete-time (R–S–T) control model was developed to build an average model of the operating zone [9]. The robust control described in [10,11] yielded a superior output power in a wide operating range and under the influence of unknown disturbances.

Intelligent algorithms have also attracted attention. In [12], a PSO-RBF method was proposed for optimizing PI pitch control systems in MW-class wind turbines. After using PSO to determine the optimal PI gains, the RBF neural network (RBFNN) can be trained to locate the optimal dataset. This network can determine suitable PI gains according to the reference structure and variable wind speeds. In [13], an nonlinear time-varying evolution PSO (NTVE-PSO) technique was used as a training phase in an RBFNN to optimize the parameters of time-series predictions for various electrical models. The algorithm reported in [14] combined a fuzzy neural network and PSO, designing a controller to adjust the speed of wind energy conversion systems. This design maintained the stability of the system and achieved the desired level performance despite uncertain parameters. In [15,16], RBFNNs were used to attain system control. Other studies have used genetic algorithms (GA) to study wind energy. A GA-based optimization technique for designing a controller procedure to use in the frequency converter of a variable-speed wind turbine was reported in [17,18]. The proposed algorithm combine PSO and RBFNN to determine control system functions. Depending on the control system, the optimal $k_p - k_i$ parameters in different k_d can be determined for various conditions. Finally, the 3D stability regions of the PID controller were plotted in the parameter space.

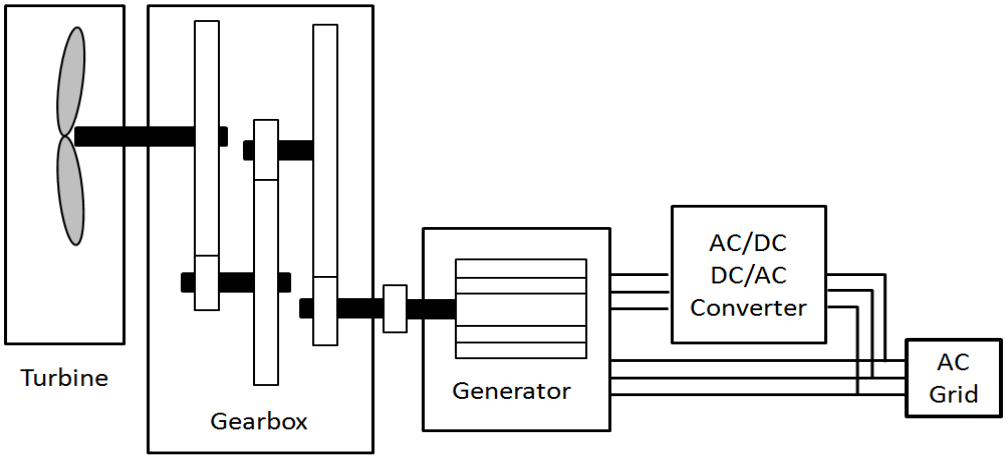
The paper is organized as follows: Section 2 introduces the wind turbine and problem formulation. Section 3 presents the use of PSO-RBFNNs in the PID-based drive train and pitch controller of a large wind turbine generator. The optimal operating point values are then identified in the center of the stability region of the PID controller. Section 4 details the simulation results (*i.e.*, the 3D stability region in the PID space for the drive train and pitch control of the wind turbine generator), and Section 5 provides a conclusion.

2. System Description and Problem Formulation

2.1. Wind Turbine System Description

In this section, the horizontal wind turbine generator is discussed. The primary structures of a wind turbine are three large blades, a powerful rotor, a strong hub, and a gearbox. The wind turbine system must be capable of operating over a wide range of wind speeds (*i.e.*, variable rotor-side and generator-side speeds). Therefore, a double-feed induction generator (DFIG) was used. Figure 1 shows the DFIG system. The AC-to-DC converter and DC-to-AC converter included ports for two currents: rotor-side and grid-side currents. The pitch-driven system was operated using hydraulic pressure, and although this system is typically used in large power systems, it included a time delay for the wind turbine generation system.

Figure 1. The DFIG architecture for the wind turbine.

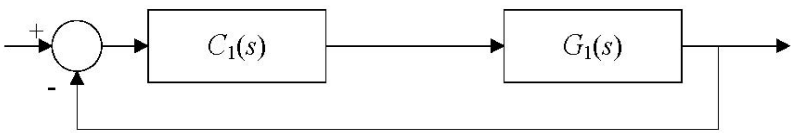


2.2. Problem Formulation for Identifying the Stability Region of the PID Controller in a Wind Turbine Pitch Control System

2.2.1. Case A: System Transfer Function without Time Delay

Figure 2 shows the block-diagram of the drive train control system incorporating a PID controller, where, $C_1(s)$ is the PID rotor torque controller, and $G_1(s)$ is the drive train model of the wind turbine system.

Figure 2. Block-diagram of the drive train control in wind turbine system.



The transfer function parameters were obtained as described in [19]. The wind turbine (HWP330/33) was manufactured by James Howden Ltd. (Renfrew, Renfrewshire, Scotland, UK); it comprises three blades and a 330-kW generator. The function can be written as follows:

$$G_1(s)=\frac{a_1 s+a_0}{b_5 s^5+b_4 s^4+b_3 s^3+b_2 s^2+b_1 s+b_0} \tag{1}$$

where the numerator and denominator are calculated as follows:

$$a_1 s+a_0=\left(\Omega N^2 k_{\mathrm{LSS}} k_{\mathrm{HSS}} k_g\right)s+0$$

$$\begin{aligned} & b_5 s^5+b_4 s^4+b_3 s^3+b_2 s^2+b_1 s+b_0 \\ = \; & N^2 k_{\mathrm{HSS}}\left(I_r s^2+k_{LSS}\right)\left[I_g s^2\left(\tau_{el} s+1\right)+k_g s\right]+I_r s^2 k_{\mathrm{LSS}}\left[\left(I_g s^2+k_{\mathrm{HSS}}\right)\left(\tau_{el} s+1\right)+k_g s\right] \\ = \; & \left[I_r I_g \tau_{el}\left(N^2 k_{\mathrm{HSS}}+k_{\mathrm{LSS}}\right)\right]s^5+\left[I_r I_g\left(N^2 k_{\mathrm{HSS}}+k_{\mathrm{LSS}}\right)\right]s^4+\left[k_{\mathrm{HSS}}\left(I_r k_{\mathrm{LSS}}\tau_{el}+I_r N^2 k_{\mathrm{LSS}}\tau_{el}\right)\right]s^3 \\ & +\left[k_{\mathrm{LSS}}\left(I_g k_{\mathrm{HSS}} N^2+I_r k_{\mathrm{HSS}}+I_r k_g\right)\right]s^2+\left(N^2 k_{\mathrm{HSS}} k_{\mathrm{LSS}} k_g\right)s+0 \end{aligned} \tag{2}$$

The parameters in this transfer function are listed in Table 1 [19]. The PID controller has the following form:

$$C_1(s)=\frac{\left(k_p s+k_i+k_d s^2\right)}{s} \tag{3}$$

where k_p is the proportional gain, k_i is the integral gain, and k_d is the derivative gain. According to Equations (1) and (3), the open-loop transfer function is calculated as follows:

$$C_1(s)G_1(s)=\frac{\left(k_p s+k_i+k_d s^2\right)\left(a_1 s+a_0\right)}{s\left(b_5 s^5+b_4 s^4+b_3 s^3+b_2 s^2+b_1 s+b_0\right)} \tag{4}$$

Thus, the denominator of the closed loop transfer function for the drive train control system is:

$$\begin{aligned} \Delta(s) &= s\left(b_5 s^5+b_4 s^4+b_3 s^3+b_2 s^2+b_1 s+b_0\right)+\left(k_p s+k_i+k_d s^2\right)\left(a_1 s+a_0\right) \\ &= b_5 s^6+b_4 s^5+b_3 s^4+\left(b_2+a_1 k_d\right)s^3+\left(b_1+a_1 k_p+a_0 k_d\right)s^2+\left(b_0+a_0 k_p+a_1 k_i\right)s+a_0 k_i \end{aligned} \tag{5}$$

Assuming $s=jy$, and $y>0$. Equation (5) may be expressed as follows:

$$\Delta(jy)=jy\left(b_4 y^4+b_2 y^2-a_1 k_d y^2+a_0 k_p+a_1 k_i+b_0\right)+\left(-b_5 y^6+b_3 y^4-\left(b_1+a_1 k_p+a_0 k_d\right)y^2+a_0 k_i\right) \tag{6}$$

The real and imaginary parts of the quasi-polynomial form in Equation (6) can be obtained by Equations (7) and (8), respectively, as follows:

$$\begin{aligned} \delta_r(y) &= c_1 k_p+c_2 k_i+c_3 k_d+c_4 \\ &= -a_1 y^2 k_p+a_0 k_i-a_0 y^2 k_d+\left(-b_5 y^6+b_3 y^4-b_1 y^2\right) \\ &= 0 \end{aligned} \tag{7}$$

$$\begin{aligned} \delta_i(y) &= d_1 k_p+d_2 k_i+d_3 k_d+d_4 \\ &= a_0 y k_p+a_1 y k_i-a_1 y^3 k_d+\left(b_4 y^5+b_2 y^3+b_0 y\right) \\ &= 0 \end{aligned} \tag{8}$$

In Equations (7) and (8), parameters $c_1, c_2, c_3, c_4, d_1, d_2, d_3$ and d_4 are as follows:

$$\begin{aligned} c_1 &= -a_1 y^2 \\ c_2 &= a_0 \\ c_3 &= -a_0 y^2 \\ c_4 &= -b_5 y^6 + b_3 y^4 - b_1 y^2 \end{aligned} \tag{9}$$

and:

$$\begin{aligned} d_1 &= a_0 y \\ d_2 &= a_1 y \\ d_3 &= -a_1 y^3 \\ d_4 &= \left(b_4 y^4 - b_2 y^2 + b_0\right) y \end{aligned} \tag{10}$$

Finally, k_p and k_i can be obtained as follows:

$$k_p = \frac{c_2\left(d_3 k_d + d_4\right) - d_2\left(c_3 k_d + c_4\right)}{c_1 d_2 - c_2 d_1} \tag{11}$$

$$k_i = \frac{c_1\left(d_3 k_d + d_4\right) - d_1\left(c_3 k_d + c_4\right)}{c_2 d_1 - c_1 d_2} \tag{12}$$

The transfer function in [19] may be expressed as follows:

$$\mathrm{G}_1(s) = \frac{2.28\times10^{19} s}{9.941\times10^{12} s^5 + 4.971\times10^{14} s^4 + 1.024\times10^{17} s^3 + 7.534\times10^{17} s^2 + 5.685\times10^{18} s} \tag{13}$$

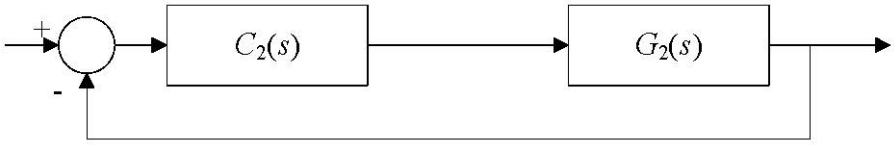
Table 1. Drive train parameters [19].

Parameters	Symbols	Values	Unit
Rotor/low speed shaft angular velocity	Ω	3.92	rad/s
Lumped inertia of generator and high-speed shaft	I_g	3.8	kg·m^2
Lumped inertia of rotor and low-speed shaft	I_r	190,120	kg·m^2
Low-speed shaft stiffness	k_{LSS}	12.6×10^6	Nm/rad
High-speed shaft stiffness	k_{HSS}	301×10^3	Nm/rad
Generator torque/speed coefficient	k_g	668	Nm/rad/s
Gearbox ratio	N	47.37	-
Generator electrical time	τ_{el}	0.02	s

2.2.2. Case B: System Transfer Function with Time Delay

Because the generation system of the wind turbine included a time delay feature, a PID controller was incorporated in the pitch control system of the wind turbine. The transfer function used for the pitch control system of the wind turbine and the turbine parameters are discussed in [3]. The wind turbine modeling was carried out using FAST [20]. Three blades and a 275 kW generator were considered. Figure 3 shows a block diagram of the pitch angle control system, where $C_2(s)$ is the PID pitch angle controller and $G_2(s)$ is the model of wind turbine system.

Figure 3. Block-diagram of the pitch angle controller for the wind turbine system.



The transfer function of the pitch control system of the horizontal-axis wind turbine can be written as follows:

$$\mathrm{G}_2(s)=\frac{a_2s^2+a_1s+a_0}{b_4s^4+b_3s^3+b_2s^2+b_1s+b_0}e^{-\tau s} \tag{14}$$

where $\tau=0.25$, parameters a_0, a_1, a_2, b_0, b_1, b_2, b_3 and b_4 represent the time constant of the wind turbine model, and depend on the device configuration.

The controller, which was modified from a PI to a PID [1], is presented as Equation (3).

Thus, the open-loop transfer function of the pith control system of the wind turbine is as follows:

$$C_2(s)G_2(s)=\frac{\left(k_ps+k_i+k_ds^2\right)\left(a_2s^2+a_1s+a_0\right)}{s\left(b_4s^4+b_3s^3+b_2s^2+b_1s+b_0\right)}e^{-\tau s} \tag{15}$$

The denominator of the closed-loop transfer function of the wind turbine pitch control system can be generated as follows:

$$\begin{aligned}\Delta(s) &= s\left(b_4s^4+b_3s^3+b_2s^2+b_1s+b_0\right)+\left(k_ps+k_i+k_ds^2\right)\left(a_2s^2+a_1s+a_0\right)e^{-\tau s}\\ &= \left(b_4s^5+b_3s^4+b_2s^3+b_1s^2+b_0s\right)+\\ &\quad\left[a_2k_ds^4+\left(a_1k_d+a_2k_p\right)s^3+\left(a_2k_i+a_1k_p+a_0k_d\right)s^2+\left(a_0k_p+a_1k_i\right)s+a_0k_i\right]e^{-\tau s}\end{aligned} \tag{16}$$

Assuming $s=jy$ and $y>0$, Equation (16) may be divided into real and imaginary quasi-polynomial parts, which can be calculated using Equations (17) and (18), respectively, as follows:

$$\begin{aligned}\delta_r(y) &= c_1k_p+c_2k_i+c_3k_d+c_4\\ &= \left(b_3y^4-b_1y^2\right)+\left[\left(a_0y-a_2y^3\right)\sin(\tau y)-a_1y^2\cos(\tau y)\right]k_p\\ &\quad+\left[a_1y\sin(\tau y)+\left(a_0-a_2y^2\right)\cos(\tau y)\right]k_i\\ &\quad+\left[\left(a_2y^4-a_0y^2\right)\cos(\tau y)-a_1y^3\sin(\tau y)\right]k_d\\ &= 0\end{aligned} \tag{17}$$

$$\begin{aligned}\delta_i(y) &= d_1k_p+d_2k_i+d_3k_d+d_4\\ &= \left(b_4y^4+b_2y^2+b_0\right)y+\left[\left(a_0-a_2y^2\right)y\cos(\tau y)+a_1y^2\sin(\tau y)\right]k_p\\ &\quad+\left[a_1y\cos(\tau y)+\left(a_2y^2-a_0\right)\sin(\tau y)\right]k_i\\ &\quad+\left[\left(a_0y^2-a_2y^4\right)\sin(\tau y)-a_1y^3\cos(\tau y)\right]k_d\\ &= 0\end{aligned} \tag{18}$$

Finally, c_1, c_2, c_3, c_4, d_1, d_2, d_3 and d_4 in Equations (17) and (18) are obtained as follows:

$$\begin{aligned} c_1 &= y\left(a_0 - a_2 y^2\right)\sin\left(\tau y\right) - a_1 y^2 \cos\left(\tau y\right) \\ c_2 &= a_1 y \sin\left(\tau y\right) + \left(a_0 - a_2 y^2\right)\cos\left(\tau y\right) \\ c_3 &= \left(a_2 y^4 - a_0 y^2\right)\cos\left(\tau y\right) - a_1 y^3 \sin\left(\tau y\right) \\ c_4 &= b_3 y^4 - b_1 y^2 \end{aligned} \tag{19}$$

and:

$$\begin{aligned} d_1 &= \left(a_0 - a_2 y^2\right) y \cos\left(\tau y\right) + a_1 y^2 \sin\left(\tau y\right) \\ d_2 &= a_1 y \cos\left(\tau y\right) + \left(a_2 y^2 - a_0\right)\sin\left(\tau y\right) \\ d_3 &= \left(a_0 y^2 - a_2 y^4\right)\sin\left(\tau y\right) - a_1 y^3 \cos\left(\tau y\right) \\ d_4 &= y\left(b_4 y^4 + b_2 y^2 + b_0\right) \end{aligned} \tag{20}$$

k_p and k_i can be can be solved using Equations (11) and (12). Table 2 lists the parameters of the wind turbine generator from [3].

Table 2. Wind turbine parameters [3].

Parameters	Values	Unit
Rated power	275	kW
Rotor diameter	27	m
Tower height	42	m
Operating conditions wind speed	15	m/s
Pitch angle	0	deg
q_{op}	$[6.848 \times 10^{-3}\ 8.915 \times 10^{-2}]^T$	-
τ	0.25	s

After using the FAST [20] aeroelastic computer-aided engineering tool to perform numerical linearization of the horizontal-axis wind turbine, the transfer function was as follows:

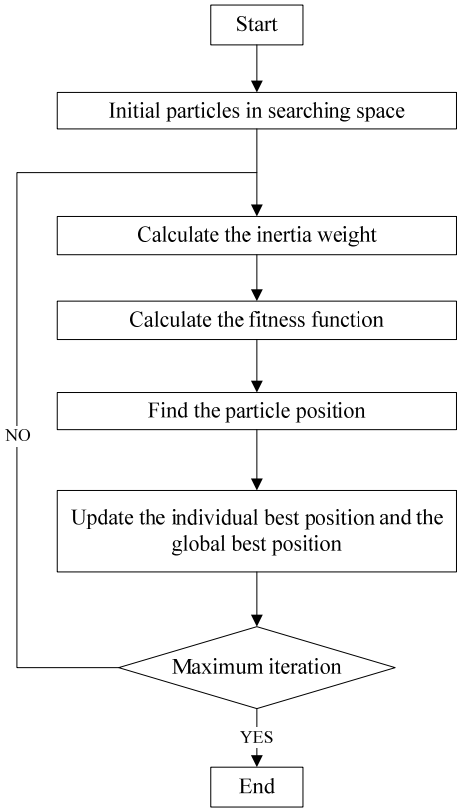
$$G_2(s) = \frac{-0.6219s^2 - 8.7165s - 2911}{s^4 + 5.018s^3 + 691.3s^2 + 1949s + 1.15\times10^5} e^{-0.25s} \tag{21}$$

3. Design of PSO-RBFNN for a PID-Based Pitch Controller for Large Wind Turbine Generator

3.1. SIWPSO Algorithm

Recent studies of artificial intelligence (AI) have discussed particle swarm optimization (PSO), an AI algorithm used for iteratively determining the optimal positions and fitness values. The primary objective of the algorithm is to simulating the foraging behaviors of birds. For example, the algorithm first simulates a flock of birds randomly searching for food in space. Assuming the search region presents only one foraging site, none of the birds know the location of the favorable foraging position; thus, the birds use various biological features to exchange information. After repeated attempts, they locate the most favorable foraging location. Figure 4 shows a flowchart depicting the typical PSO.

Figure 4. Flow chart of PSO algorithm.



The desired PSO parameters are *m* particles searching a *d*-dimensional space. At the t_{th} iteration, the equations are:

$$\begin{aligned} X_i(t) &= \left(x_{i1}(t), x_{i2}(t), \cdots, x_{id}(t)\right) \\ V_i(t) &= \left(v_{i1}(t), v_{i2}(t), \cdots, v_{id}(t)\right) \end{aligned} \tag{22}$$

where X_i is the i_{th} particle position, and V_i is the i_{th} particle velocity.

The individual optimal position P_i is given by:

$$P_i(t) = \left(p_{i1}(t), p_{i2}(t), \cdots, p_{id}(t)\right) \tag{23}$$

The global optimal position P_g is then given by:

$$P_g(t) = \left(p_{g1}(t), p_{g2}(t), \cdots, p_{gd}(t)\right) \tag{24}$$

The stochastic inertia weight in the PSO (SIWPSO) algorithm is an evolutionary computational method developed by Eberhart and Shi [21,22]. The fitness function equation for the integrated absolute error (IAE) is computed by integrating the absolute value of the error. After performing a SIWPSO run, the IAE delivers the optimal solution to the real objective function to calculate its real fitness. The fitness function is formulated as follows:

$$\begin{aligned} T_{IAE} &= f\left(k_p, k_i, k_d\right) \\ &= \int_0^T \left|1 - y\left(k_p, k_i, k_d, t\right)\right| dt \end{aligned} \tag{25}$$

Finally, the updated velocity and position are given by:

$$\begin{aligned} V_{in}(t+1) &= w(t)V_{in}(t)+\beta_1\gamma_1\left(P_{in}(t)-X_{in}(t)\right)+\beta_2\gamma_2\left(P_{gn}(t)-X_{in}(t)\right) \\ X_{in}(t+1) &= X_{in}(t)+V_{in}(t+1) \end{aligned} \quad (26)$$

where n is the n_{th} dimension of the search space. The inertia weight $w(t)$ is randomly selected according to a uniform distribution in the range of 0.5–1.0, exhibiting a mean value of 0.75. γ_1 and γ_2 are uniform distributions in the range of 0–1. In this study, the individual coefficient β_1 and global coefficient β_2 in Equation (26) are set to 1.494.

The profiles of the four PSOs were compared regarding of IAE [23]. The algorithms compared the following: the PSO, constriction factor approach in PSO, linearly decreasing inertia weight in PSO, and SIWPSO. In this study, the dimension for the $k_p - k_i$ search space was set to 2. Table 3 shows the detailed parameter values for various PSOs. Figure 5 shows a detailed comparison of the PSO algorithms based on their IAE values without a time delay, whereas Figure 6 shows the same with time delay. Performance comparisons of 40 iterations for each PSO method clearly demonstrated that the convergence rate was fastest in SIWPSO and slowest in the standard PSO algorithm. Table 4 (without time delay) and Table 5 (with time delay) compare the IAE performance of the four PSO methods.

Table 3. The PSO parameters.

Parameters	PSO	LDWPSO	CFAPSO	SIWPSO
Dimension			2	
Number of Particles			5	
β_1, β_2	2, 2	2, 2	2.05, 2.05	1.494, 1.94
Weight	0.4	0.4–0.9	0.7298	0.5–1.0
Fitness Function			IAE	
Max Iteration			50	

Figure 5. The IAE fitness value obtained by Equation (13) for each iteration of the four PSO methods.

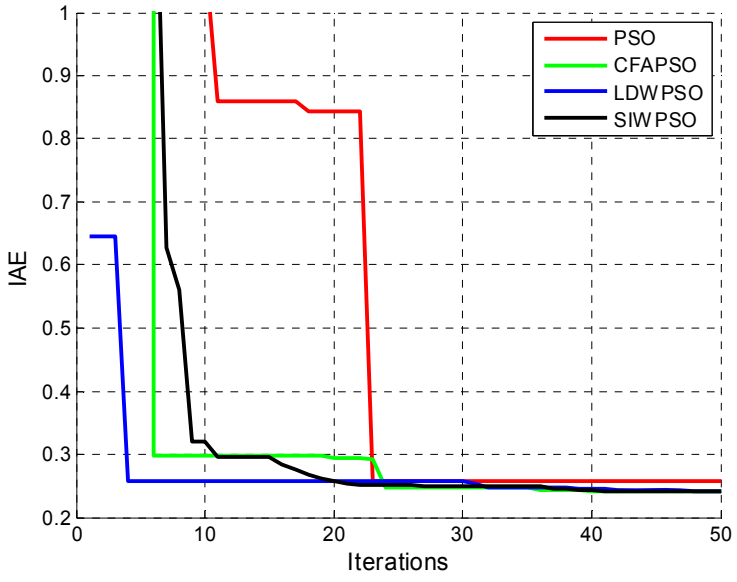


Figure 6. Plots of the IAE fitness values in Equation (21) for each iteration of the four PSO methods.

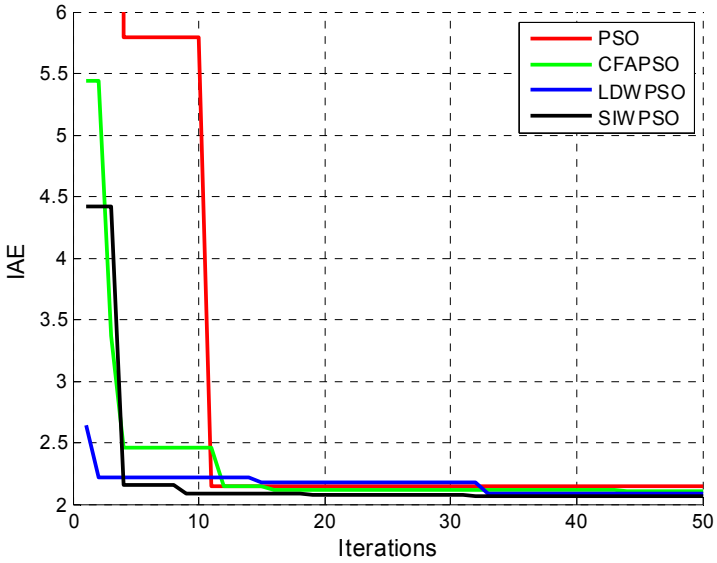


Table 4. The IAE values obtained by Equation (13) for each iteration of the four PSO methods.

Methods	k_p	k_i	IAE
PSO	0.3821	1.0883	0.2572
CFAPSO	0.3979	1.2774	0.2421
LDWPSO	0.4996	1.38	0.2419
SIWPSO	0.5226	1.4619	0.2415

Table 5. The IAE values obtained by Equation (21) for each iteration of the four PSO methods.

Methods	k_p	k_i	IAE
PSO	1.0017	−22.0900	2.1428
CFAPSO	1.3309	−20.1238	2.1098
LDWPSO	1.242	−20.6178	2.086
SIWPSO	1.0242	−20.5914	2.0624

3.2. RBFNN Algorithm

The RBFNN introduced in [12,24] is used in self-learning systems composed of large numbers in a simple data set. The algorithm comprises three layers: an input layer, a hidden layer exhibiting a nonlinear activation function and an output layer. Figure 7 displays the architecture of a typical RBF network. Figure 8 displays a flowchart depicting the typical RBFNN.

Each input node corresponds to an element of the input layer, and each hidden node consist a radial-activated function, that includes local perception nodes. The activation function is described as follows:

$$\begin{aligned} \varphi_j(n) &= \varphi_j\left\{h(n), \lambda_j(n), \sigma_j(n)\right\} \\ &= e^{-\frac{\left\|h(n)-\lambda_j(n)\right\|^2}{\sigma_j^2(n)}} \end{aligned} \tag{27}$$

Figure 7. Architecture of a RBF network.

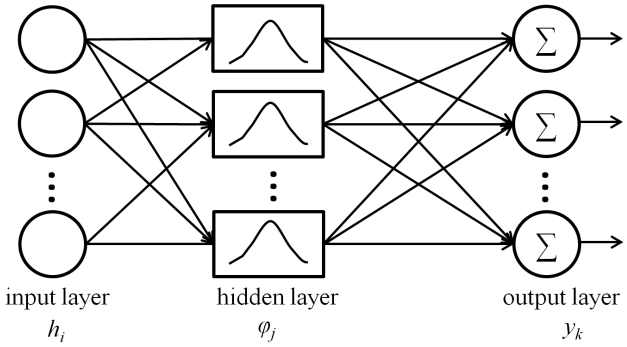
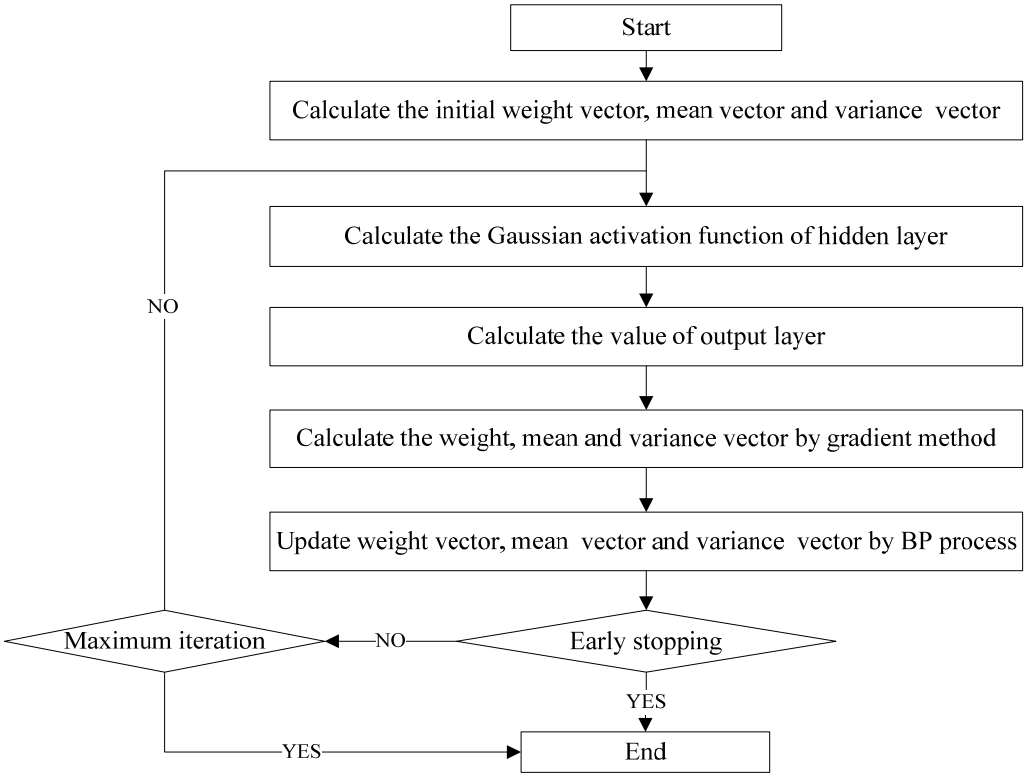


Figure 8. Flow chart of a RBFNN algorithm.



The activation function used in this study was a Gaussian distribution. The terms λ_j and σ_j^2 are the mean and variance of the j Gaussian distribution function, respectively. According to this nonlinear function, the amplitude of the output from each node of the hidden layer is in the range $0 < \varphi < 1$. Therefore, each hidden node corresponds to an element of the hidden layer, and each output node consist a linear function. Output y_k of the RBFNN is a sum of the weight values multiplied by the hidden node output:

$$y_k = \alpha_k \left(\sum_{j=1}^{J} \omega_{kj} \varphi_j + \theta_k \right) \tag{28}$$

where α_k is the output activation function and θ_k is the threshold value. The α_k was a linear unit and $\theta_k = 0$. The output layer provides a summation to each output node. Here, $\alpha_k = 1$.

The RBFNN can was trained using back-propagation (BP) as a supervised learning process. The output error was used to update their weight values ω_j, center values λ_j and width values σ_j. as follows:

$$\begin{aligned}\omega_j(n+1)&=\omega_j(n)+\Delta\omega_j(n)\\ \lambda_j(n+1)&=\lambda_j(n)+\Delta\lambda_j(n)\\ \sigma_j(n+1)&=\sigma_j(n)+\Delta\sigma_j(n)\end{aligned} \tag{29}$$

According to the chain rule, the supervised-learning BP process is as follows:

$$\begin{aligned}\omega_j(n+1)&=\omega_j(n)+\mu e(n)\varphi_j(n)\\ \lambda_j(n+1)&=\lambda_j(n)+\mu e(n)\omega_j(n)\varphi_j(n)\frac{\left[h(n)-\lambda_j(n)\right]}{\sigma_j^2(n)}\\ \sigma_j(n+1)&=\sigma_j(n)+\mu e(n)\omega_j(n)\varphi_j(n)\frac{\left\|h(n)-\lambda_j(n)\right\|^2}{\sigma_j^3(n)}\end{aligned} \tag{30}$$

where μ is the learning rate, and the error output can be written as follows:

$$e(n)=y(n)-y_d(n) \tag{31}$$

The iteration process optimized the RBF weight, center, and width values. The RBFNN was applied to the PSO approach to optimize the operating point of the PID control parameters. The RBF parameters were adopted: hidden neurons = 7, learning rate = 0.01, training times = 5000, and number of training data = 21.

In this study, the early stopping rule was used in the upper bounds to allow the RBFNN algorithms to converge. When the mean squared error generated by the error output began to increase, the RBFNN algorithm was stopped.

3.3. Design Procedure

The objective was to optimize the PID controller function, (*i.e.*, to use the SIWPSO-based RBFNN algorithm to optimizing the PID operating point in the wind turbine generator). The optimization procedure for the proposed SIWPSO-based RBFNN was designed as follows.

Step 1: Based on the transfer function of the wind turbine generation system, set the range of control parameter k_d.

Step 2: Set the values for k_d and for the initial SIWPSO particles in k_p-k_i parameter space. Specify the maximal and minimal values for position and velocity. Set the maximal iteration value.

Step 3: Initiate the movement of the particles, using a random position and velocity.

Step 4: Calculate the IAE fitness value for each SIWPSO particle.

Step 5: Select the optimal local and global values for each particle based on the minimal IAE fitness values, and update the optimal individual and global positions in k_p-k_i parameter space.

Step 6: Repeat Steps 3–4 until the maximal number of iterations in PSO is reached.

Step 7: Modify k_d and repeat Step 6 until all k_d parameters are calculated.

Step 8: Based on the PSO training results, optimize the operating values of the PID control parameters, and calculate the RBF parameters (initial weight vector, mean vector, and variance vector).

Step 9: Calculate the values of the hidden and output layers. According to the error output, use the BP process to update the weight vector, mean vector, and variance vector.

Step 10: Repeat Steps 8–9 until the maximum number of iterations in RBF is reached.

4. Simulation Results

4.1. Stability Region and Optimal $k_p - k_i - k_d$ Operating Region without Time Delay System

This section describes how the stability boundary of the PID controller was determined for Case A. The stability boundary at $k_d = 0$ in plane $k_p - k_i$ was defined by Equation (13). To measure the response of the optimal operating point of the PID controller in the drive train system, the SIWPSO $k_p - k_i$ gain was tested for each k_d. Regarding the system that lacked a time delay, the stability boundary at $k_d = 0$ was determined as shown in Figure 9. The unit-step response is used for performance verification of the drive train system in Figure 10. It is obvious that the SIWPSO method (red line) produced the most favorable performance response. According to the result in Figure 9, region R1 was a stable region (shaded), but region R2 is unstable. Test #1 and Test #2 are the two test operating points for the drive train system. Table 6 presents the optimal operating point for the drive train system, SIWPSO yielded the minimal IAE, indicating that SIWPSO identified the optimal operating point for the system that lacked a time delay.

Figure 9. Stability regions of PID controller for Equation (13).

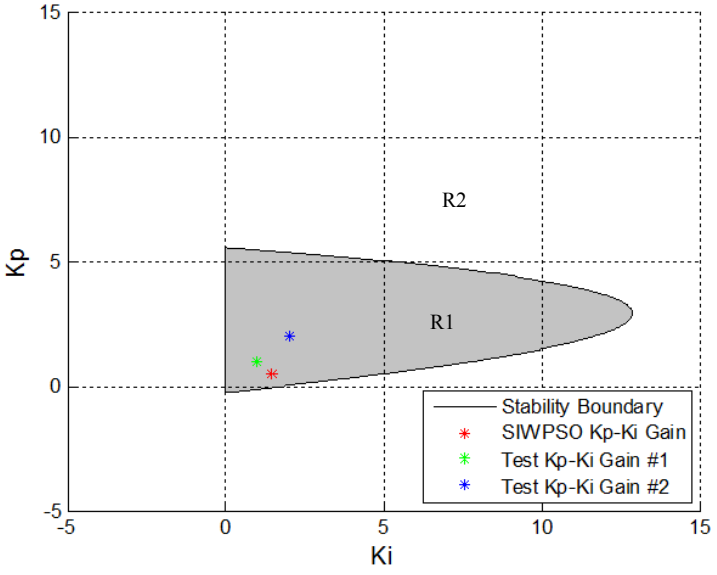
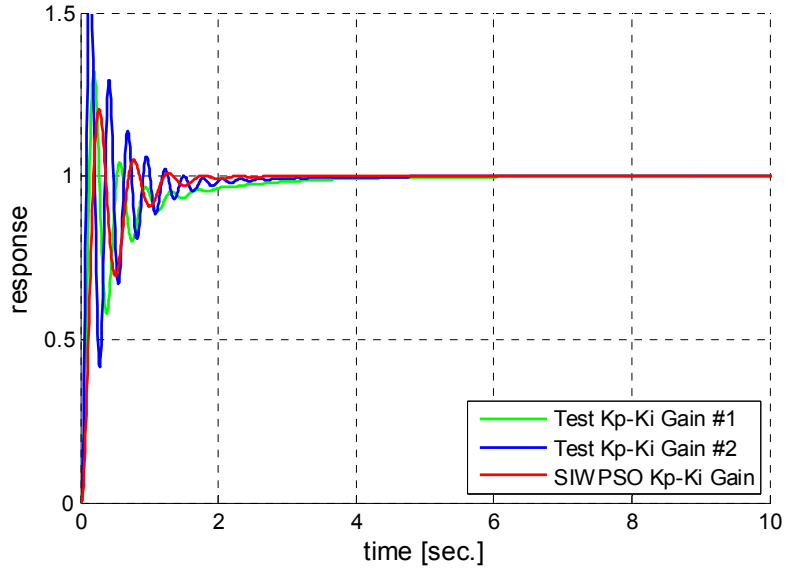


Figure 10. Performance verification of the drive train system.



A grid for k_d was used to locate the stability regions in 3D. Therefore, Figure 11 shows the possible stability regions of the PID controller for the given system. The proposed method facilitated intuitively calculating parameter space, stabilizing the PID controller for the drive train system. The 3D plot is displayed in Figure 12. In Case A, the intersection of the stability boundary (transparent shaded), RBF training set (green circle) and optimal trajectories were determined. The red-line and blue-line trajectories were generated by using PSO-RBF in $k_p - k_i - k_d$ space. The trajectories were enclosed in the determined stability boundary.

Table 6. The IAE values for the drive train system.

Operating points	k_p	k_i	IAE
Test #1	1	1	0.3278
Test #2	2	2	0.3073
SIWPSO	0.5226	1.4619	0.2415

Figure 11. The stabilizing in the $k_p - k_i - k_d$ space for Equation (13).

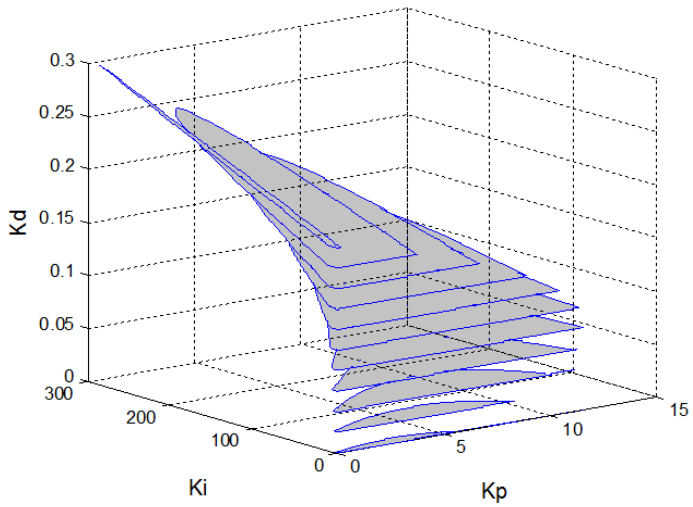
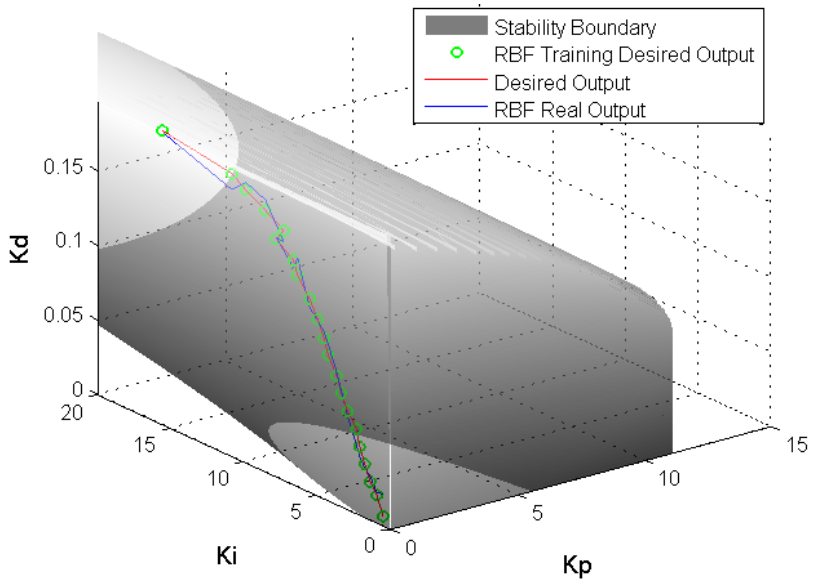


Figure 12. The 3D stability boundary in the $k_p - k_i - k_d$ parameter space.



4.2. Stability Region and Optimal $k_p - k_i - k_d$ Operating Region with Time Delay System

This section describes how the stability boundary of the PID controller was determined for Case B. The stability boundary at k_d in $k_p - k_i$ plane was defined by Equation (21). The response of the optimal operating point of the PID for the wind turbine generator was determined by testing the SIWPSO $k_p - k_i$ gain for each k_d. Regarding the time delay system, Figure 13 shows how the stability boundary of $k_d = 0$ was obtained. This boundary comprised three regions as follows (R1, R2 and R3). Of these region, R1 (shaded area) was stable, but R2 and R3 were unstable according to the criteria established in [3]. Figure 14 shows that the unit-step response was used to verify the performance of the wind turbine generator system, presents the optimal operating point for the wind turbine generator. According to the result in Table 7, the minimal IAE value is obtained by using SIWPSO, indicating that SIWPSO identified the optimal operating point for the time delay system.

Figure 13. Stability regions of PID controller for Equation (21).

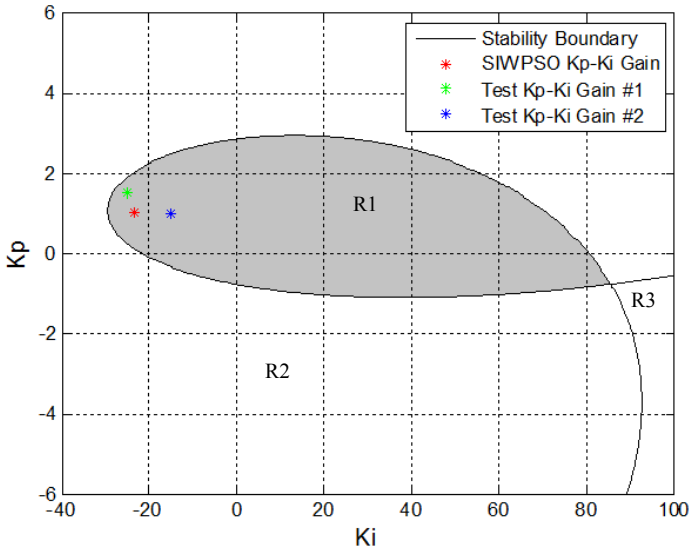


Figure 14. The performance verification of the wind turbines generator system.

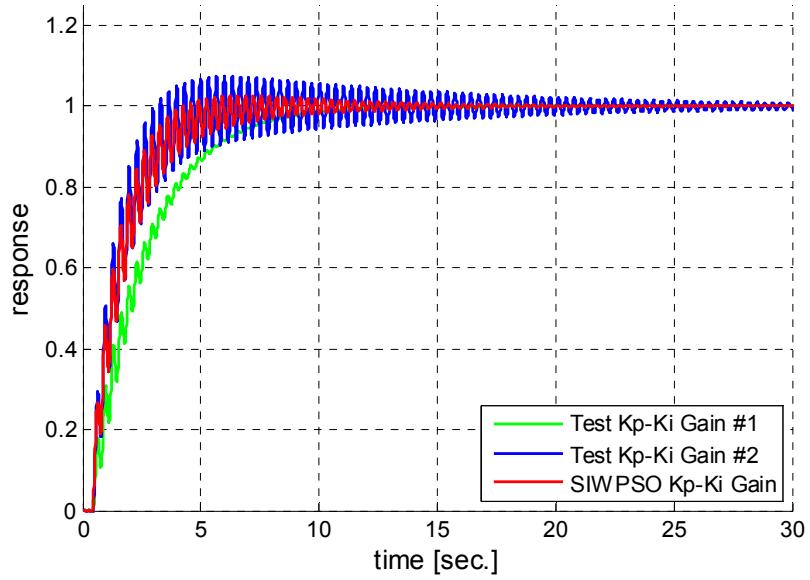


Table 7. The IAE values for the wind turbine generator.

Operating points	k_p	k_i	IAE
Test #1	1.5	−25	2.2026
Test #2	1	−15	2.6837
SIWPSO	1.0242	−20.5914	2.0624

A k_d grid is used to identify the stabilizing regions in 3D. Figure 15 shows the possible PID stability regions for the given system. The proposed method was intuitive, and readily calculates the parameter space stabilizing the PID controller for the given system. Figure 16 displays the 3D plot. As previously described, the intersection of the stability boundary (transparent-shaded), RBF training set (green-circle), and optimal trajectories were also calculated.

Figure 15. The stabilizing region in $k_p - k_i - k_d$ parameter space for Equation (21).

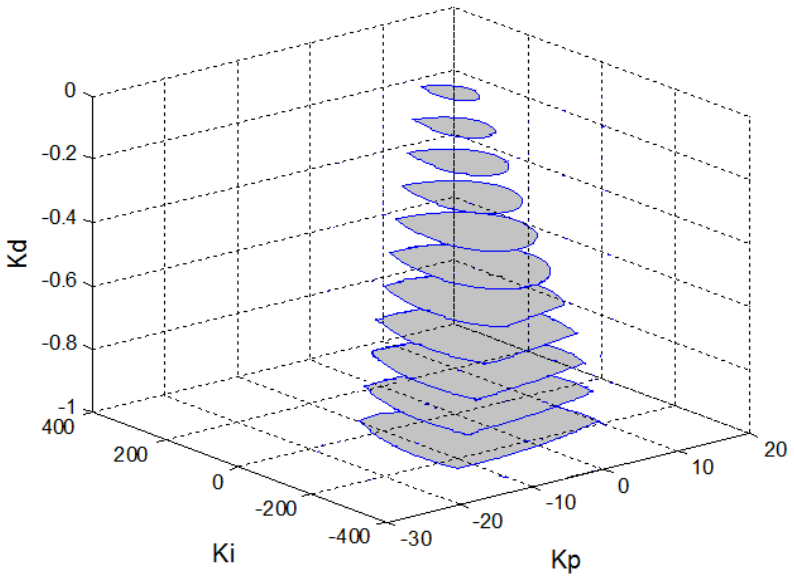
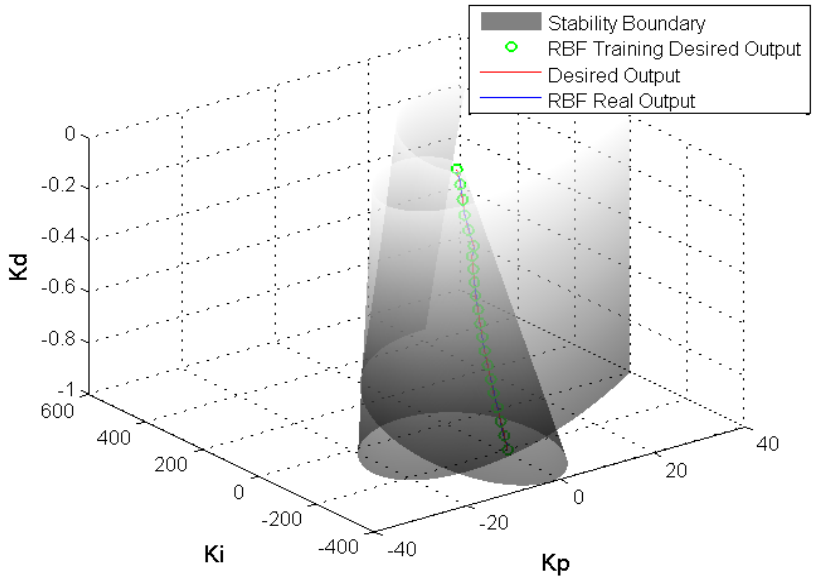


Figure 16. The 3D stability boundary in $k_p - k_i - k_d$ parameter space.



5. Conclusions

This study used a graphical approach to determine the boundaries of a PID controller. The proposed PSO-RBF algorithms were successfully used to control a wind turbine generator system. This technique was used to determined the system stability boundaries and optimal operating points of the PID controller of wind turbine generator system. The proposed method was effective in systems with and without time delay. Finally, the proposed graphical approach can provides 2D or 3D vision boundaries of the resulting PID-type controller space for close-loop wind turbine systems.

Acknowledgments

The authors would like to thank the National Science Council of the Republic of China, Taiwan for financially supporting this research under Contract No. NSC 102-3113-P-110-006.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Jauch, C.; Islam, S.M.; Sorensen, P.; Jensen, B.B. Design of a wind turbine pitch angle controller for power system stabilization. *Renew. Energy* **2007**, *85*, 2334–2349.
2. Vidal, Y.; Acho, L.; Luo, N.; Zapateiro, M.; Pozo, F. Power control design for variable-speed wind turbines. *Energies* **2012**, *5*, 3033–3050.
3. Wang, J.; Tse, N.; Gao, Z. Synthesis on PI-based pitch controller of large wind turbines generator. *Energy Convers. Manag.* **2011**, *52*, 1288–1294.
4. Li, Y.; Sheng, A.; Wang, Y. Synthesis of PID-type controllers without parametric models: A graphical approach. *Energy Convers. Manag.* **2008**, *49*, 2392–2402.

5. Bozorg, M.; Termeh, F. Domains of PID controller coefficients which guarantee stability and performance for LTI time-delay systems. *Automatica* **2011**, *47*, 2122–2125.
6. Liang, T.; Chen, J.; Lei, C. Algorithm of robust stability region for interval plant with time delay using fractional order $PI^{\lambda}D^{\mu}$ controller. *Commun. Nonlinear Sci. Numer. Simul.* **2012**, *17*, 979–991.
7. Madady, A.; Alikhani, H.R.R. Stabilization of control loops consisting of FOPDT process and parameter-dependent PID controller. *J. Process Control* **2012**, *22*, 1688–1701.
8. Soylemez, M.T.; Munro, N.; Baki, H. Fast calculation of stabilizing PID controllers. *Automatica* **2003**, *39*, 121–126.
9. Camblong, H. Digital robust control of a variable speed pitch regulated wind turbine for above rated wind speeds. *Control Eng. Pract.* **2008**, *16*, 946–958.
10. Geng, H.; Yang, G. Output power control for variable-speed variable-pitch wind generation systems. *IEEE Trans. Energy Convers.* **2010**, *25*, 494–503.
11. Corcuera, A.D.D.; Pujana-Arrese, A.; Ezquerra, J.M.; Segurola, E.; Landaluze, J. H-infinity based control for load mitigation in wind turbines. *Energies* **2012**, *5*, 938–967.
12. Poultangari, I.; Shahnazi, R.; Sheikhan, M. RBF neural network based PI pitch controller for a class of 5-MW wind turbines using particle swarm optimization algorithm. *ISA Trans.* **2012**, *51*, 641–648.
13. Lee, C.M.; Ko, C.N. Time series prediction using RBF neural networks with a nonlinear time-varying evolution PSO algorithm. *Neurocomputing* **2009**, *73*, 449–460.
14. Lin, W.M.; Hong, C.M.; Cheng, F.S. Design of intelligent controllers for wind generation system with sensorless maximum wind energy control. *Energy Convers. Manag.* **2011**, *52*, 1086–1096.
15. Yilmaz, A.S.; Özer, Z. Pitch angle control in wind turbines above the rated wind speed by multi-layer perceptron and radial basis function neural networks. *Expert Syst. Appl*. **2009**, *36*, 9767–9775.
16. Chen, Y.; Xu, J.; Yang, B.; Zhao, Y.; He, Y. A novel method for prediction of protein interaction sites based on integrated RBF neural networks. *Comput. Biol. Med.* **2012**, *42*, 402–407.
17. Hasanien, H.M.; Muyeen, S.M. Design optimization of controller parameters used in variable speed wind energy conversion system by genetic algorithms. *IEEE Trans. Sustain. Energy* **2012**, *3*, 200–208.
18. Das, D.C.; Roy, A.K.; Sinha, N. GA based frequency controller for solar thermal diesel-wind hybrid energy generation/energy storage system. *Electr. Power Energy Syst*. **2012**, *43*, 262–279.
19. Anderson, C.G.; Richon, J.B.; Campbell, T.J. An aerodynamic moment-controlled surface for gust load alleviation on wind turbine rotors. *IEEE Trans. Control Syst. Technol.* **1998**, *6*, 577–595.
20. FAST: An Aeroelastic Computer-Aided Engineering (CAE) Tool for Horizontal Axis Wind Turbines. Available online: http://wind.nrel.gov/designcodes/simulators/fast (accessed on 3 January 2014).
21. Shi, Y.; Eberhart, R.C. Particle Swarm Optimization: Developments, Applications and Resources. In Proceedings of the 2001 Congress on Evolutionary Computation, Seoul, Korea, 27–30 May 2001; Volume 1, pp. 81–86.

22. Shi, Y.; Eberhart, R.C. Tracking and Optimizing Dynamic Systems with Particle Swarms. In Proceedings of the 2001 Congress on Evolutionary Computation, Seoul, Korea, 27–30 May 2001; Volume 1, pp. 94–97.
23. Fang, H.; Chen, L.; Shen, Z. Application of an improved PSO algorithm to optimal tuning of PID gains for water turbine governor. *Energy Convers. Manag.* **2011**, *52*, 1763–1770.
24. Lin, C.T.; Lee, C.S.G. *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*, International ed.; Prentice Hall: Taipei, Taiwan, August 2003; pp. 328–330.