

An Improved Particle Swarm Optimization-Based Dynamic Recurrent Neural Network for Identifying and Controlling Nonlinear Systems

Hong-Wei Ge¹, Yan-Chun Liang^{*1,2}, and Maurizio Marchese²

¹ College of Computer Science and Technology, Jilin University, Changchun 130012, China

² Department of Information and Communication Technology, University of Trento, Via Sommarive
14, 38050, Povo (TN) Italy

* Corresponding author: Yan-Chun Liang, E-mail: ycliang@jlu.edu.cn

An Improved Particle Swarm Optimization-Based Dynamic Recurrent Neural Network for Identifying and Controlling Nonlinear Systems

Hong-Wei Ge¹, Yan-Chun Liang^{*1,2}, and Maurizio Marchese²

¹ College of Computer Science and Technology, Jilin University, Changchun 130012, China

² Department of Information and Communication Technology, University of Trento, Via Sommarive 14, 38050, Povo (TN) Italy

Corresponding author: ycliang@jlu.edu.cn

Abstract

In this paper, we first present a learning algorithm for dynamic recurrent Elman neural networks based on an improved particle swarm optimization. The proposed algorithm computes concurrently both the evolution of network structure, weights, initial inputs of the context units and self-feedback coefficient of the modified Elman network. Thereafter, we introduce and discuss a novel control method based on the proposed algorithm. More specifically, a dynamic identifier is constructed to perform speed identification and a controller is designed to perform speed control for Ultrasonic Motors (USM). Numerical experiments show that the novel identifier and controller based on the proposed algorithm can both achieve higher convergence precision and speed than other state-of-the-art algorithms. In particular, our experiments show that the identifier can approximate the USM's nonlinear input-output mapping accurately. The effectiveness of the controller is verified using different kinds of speeds of constant, step, and sinusoidal types. Besides, a preliminary examination on a randomly perturbation also shows the robust characteristics of the two proposed models.

Key words: dynamic recurrent neural network; particle swarm optimization; nonlinear system

identification; system control; ultrasonic motor

1. Introduction

The design goal of a control system is to influence the behavior of dynamic systems to achieve some pre-determinate objectives. A control system is usually designed on the premise that an accurate knowledge of a given object and environmental cannot be obtained in advance. So it is both desired and relevant to find suitable methods to address the problems related to uncertain and highly complicated dynamic system identification. As a matter of fact, system identification is an important branch of research in the automatic control domain. However, the majority of methods for system identification and parameters' adjustment are based on linear analysis: therefore it is difficult to extend them to complex non-linear systems. Normally, a large amount of approximations and simplifications have to be performed and, unavoidably, they have a negative impact on the desired accuracy. Fortunately the characteristics of the Artificial Neural Network (ANN) approach, namely non-linear transformation and support to highly parallel operation, provide effective techniques for system identification and control, especially for non-linear systems [1-6]. The ANN approach has a high potential for identification and control applications mainly because: (1) it can approximate the nonlinear input-output mapping of a dynamic system; (2) it enables to model the complex systems' behavior and to achieve an accurate control through training, without a priori information about the structures or parameters of systems. Due to these characteristics, there has been a growing interest, in recent years, in the application of neural networks to dynamic system identification and control.

An Ultrasonic Motor (USM) is a typical non-linear dynamic system. It is a newly developed motor, which has some excellent performances and useful features such as high torque at low

speeds, compactness in size and thickness, no electromagnetic interference, short start-stop times, and many others. Due to the mentioned advantages, USMs have attracted considerable attention in many practical applications [7-10]. The simulation and control of an USM are crucial in the actual use of such systems. Following a conventional control theory approach, an accurate mathematical model should be first set up, in order to simulate and analyze USM's behavior. But an USM has strongly nonlinear speed characteristics that vary with the driving conditions [11] and its operational characteristics depend on many factors, including the mode shape, the resonant frequency, the contact stiffness, the frictional characteristics, the working temperature and many others. Therefore, performing effective identification and control in this case, is a challenging task using traditional methods based on mathematical models of the systems. Moreover, to construct such complex and formal models is normally demanding from the control designer's point of view.

The ANN approach can be applied advantageously to the specific tasks of USM's identification and control since it is capable to tackle nonlinear behaviors and it does not require any system's a priori knowledge. To this end, the dynamic recurrent multilayer network introduces dynamic links to memorize feedback information of the history influence. This approach has great developmental potential in the fields of system modeling, identification and control [12-16]. The Elman network is one of the simplest types among the available recurrent networks. In the following sections, a modified Elman network is employed to identify and control an USM, and a novel learning algorithm based on an improved particle swarm optimization is proposed for training the Elman network, so that the complex theoretical analysis of the operational mechanism and the exact mathematical description of the USM are avoided.

2. Modified Elman Neural Network

Figure 1 depicts the modified Elman Neural Network (ENN) which was proposed by Pham and Liu [17] based on the original Elman network introduced by Elman [18]. The modified Elman network introduces a self-feedback coefficient to improve its memorization ability. It is a type of recurrent neural network with different layers of neurons, namely: input nodes, hidden nodes, output nodes and, specific of the approach, context nodes. The input and output nodes interact with the outside environment, whereas the hidden and context nodes do not. The context nodes are used only to memorize previous activations of the hidden nodes and can be considered to function as a one-step time delay. The feed-forward connections are modifiable, whereas the recurrent connections are fixed. The modified Elman network differs from the original Elman network by having self-feedback links with fixed coefficient α in the context nodes. Thus the output of the context nodes can be described by

$$x_{cl}(k) = \alpha x_{cl}(k-1) + x_l(k-1) \quad (l=1,2,\Lambda, n) \quad (1)$$

where $x_{cl}(k)$ and $x_l(k)$ are, respectively, the outputs of the l th context unit and the l th hidden unit and α ($0 \leq \alpha < 1$) is the self-feedback coefficient. When the coefficient α is zero, the modified Elman network is identical to the original Elman network. If we assume that there are r nodes in the input layer, n nodes in the hidden and context layers, respectively, and m nodes in the output layer, then the input u is an r dimensional vector, the output x of the hidden layer and the output x_c of the context nodes are n dimensional vectors, the output y of the output layer is m dimensional vector, and the weights W^{I1} , W^{I2} and W^{I3} are $n \times n$, $n \times r$ and $m \times n$ dimensional matrices, respectively.

The mathematical model of the modified Elman neural network can be described as follows

$$x(k) = f(W^{I1}x_c(k) + W^{I2}u(k-1)), \quad (2)$$

$$x_c(k) = \alpha x_c(k-1) + x(k-1), \quad (3)$$

$$y(k) = g(W^{I3}x(k)), \quad (4)$$

where $f(x)$ is often taken as the sigmoidal function

$$f(x) = \frac{1}{1 + e^{-x}} \quad (5)$$

and $g(x)$ is often taken as a linear function, that is

$$y(k) = W^{I3}x(k) \quad (6)$$

Let the k th desired output of the system be $y_d(k)$. We can then define the error as

$$E(k) = \frac{1}{2}(y_d(k) - y(k))^T (y_d(k) - y(k)). \quad (7)$$

Differentiating E with respect to W^{I3} , W^{I2} and W^{I1} respectively, according to the gradient descent method, we obtain the following equations

$$\Delta w_{ij}^{I3} = \eta_3 \delta_i^0 x_j(k) \quad (i = 1, 2, \Lambda, m; j = 1, 2, \Lambda, n), \quad (8)$$

$$\Delta w_{jq}^{I2} = \eta_2 \delta_j^h u_q(k-1) \quad (j = 1, 2, \Lambda, n; q = 1, 2, \Lambda, r), \quad (9)$$

$$\Delta w_{jl}^{I1} = \eta_1 \sum_{i=1}^m (\delta_i^0 w_{ij}^{I3}) \frac{\partial x_j(k)}{\partial w_{jl}^{I1}} \quad (j = 1, 2, \Lambda, n; l = 1, 2, \Lambda, n), \quad (10)$$

which form the learning algorithm for the modified Elman neural network, where η_1 , η_2 and η_3 are learning steps of W^{I1} , W^{I2} and W^{I3} , respectively, and

$$\delta_i^0 = (y_{d,i}(k) - y_i(k)) g'_i(\cdot), \quad (11)$$

$$\delta_j^h = \sum_{i=1}^m (\delta_i^0 w_{ij}^{I3}) f'_j(\cdot), \quad (12)$$

$$\frac{\partial x_j(k)}{\partial w_{jl}^{I1}} = f'_j(\cdot) x_l(k-1) + \alpha \frac{\partial x_j(k-1)}{\partial w_{jl}^{I1}}. \quad (13)$$

If $g(x)$ is taken as a linear function, then $g'_i(\cdot) = 1$. Clearly, Eqs. (10) and (13) possess recurrent characteristics.

From the above dynamic equations it can be seen that the output at an arbitrary time is influenced by the past input-output because of the existence of feedback links. If a dynamic system is identified or controlled by an Elman network with an artificially imposed structure, and the gradient descent learning algorithm is used to train the network, this may give rise to a number of problems:

- the initial input of the context unit is artificially provided, which in turn induces larger errors of system identification or control at the initial stage.
- searches are easy to get locked into local minima.
- the self-feedback coefficient α is determined artificially or experimentally by a lengthy trial-and-error process, which induces a lower learning efficiency.
- if the network structure and weights are not trained concurrently, we have first to determine the number of the nodes of the hidden layer and then train the related weights. At the end these weights may not obey the Kosmogorov theorem therefore a good performance of dynamic approximation can not be guaranteed [19].

In order to face the above critical points, we propose a learning algorithm for the ENN based on an Improved Particle Swarm Algorithm (IPSO) to enhance the identification capabilities and control performances of the models. We call the proposed integration algorithm as IPBEA (IPSO-based ENN Learning Algorithm). Furthermore, a novel identifier and a novel controller, based on IPBEA, are designed to identify and control non-linear systems. In the following, they are named IPBEI (IPSO-based ENN Identifier) and IPBEC (IPSO-based ENN Controller) respectively.

3. Improved Particle Swarm Optimization (IPSO)

Particle Swarm Optimization (PSO), originally developed by Kennedy and Elberhart [20], is a

method for optimizing hard numerical functions based on the metaphor of the social behavior of flocks of birds and groups of fish. PSO can be briefly described as an evolutionary computation technique based on swarm intelligence. A swarm consists of individuals, called “particles”, which change their positions over time. Each particle represents a potential solution to the problem. In a PSO system, particles fly around in a multi-dimensional search space. During its flight each particle adjusts its position according to its own experience and the experience of its neighboring particles, making use of the best position encountered by itself and its neighbors. The overall effect is that particles tend to move towards most promising solution areas in the multi-dimensional search space, while maintaining the ability to search a wide area around the localized solution areas. The performance of each particle is measured according to a pre-defined fitness function, which is related to the problem being solved and indicates how good a candidate solution is. The PSO has been found to be robust and fast in solving non-linear, non-differentiable, multi-modal problems. The mathematical abstract and executive steps of PSO are as follows.

Let the i th particle in a D -dimensional space be represented as $X_i = (x_{i1}, K, x_{id}, K, x_{iD})$. The best previous position (which possesses the best fitness value) of the i th particle is recorded and represented as $P_i = (p_{i1}, K, p_{id}, K, p_{iD})$, which is also called $pbest$. The index of the best $pbest$ among all the particles is represented by the symbol g . The location P_g is also called $gbest$. The velocity for the i th particle is represented as $V_i = (v_{i1}, K, v_{id}, K, v_{iD})$. The basic concept of the particle swarm optimization consists of, at each time step, changing the velocity and location of each particle towards its $pbest$ and $gbest$ locations according to Eqs. (14) and (15), respectively:

$$V_i(k+1) = wV_i(k) + c_1r_1(P_i - X_i(k))/\Delta t + c_2r_2(P_g - X_i(k))/\Delta t \quad (14)$$

$$X_i(k+1) = X_i(k) + V_i(k+1)\Delta t \quad (15)$$

where w is the inertia coefficient which is a constant in interval $[0, 1]$ and can be adjusted in the direction of linear decrease [21]; c_1 and c_2 are learning rates which are nonnegative constants; r_1 and r_2 are generated randomly in the interval $[0, 1]$; Δt is the time interval, and commonly be set as unit; $v_{id} \in [-v_{\max}, v_{\max}]$, and v_{\max} is a designated maximum velocity. The termination criterion for iterations is determined according to whether a maximum generation number or a designated value of the fitness is reached. PSO has attracted broad attention in the fields of evolutionary computing, optimization and many others [22-24].

The method described above can be considered as the conventional particle swarm optimization, in which as time goes on, some particles become quickly inactive because they are similar to the *gbest* and lose their velocities. In the subsequent generations, they will have less contribution to the search task for their very low global and local search activity. In turn, this will induce the emergence of a state of premature convergence, defined technically as prematurity. To improve on this specific issue, we introduce an adaptive mechanism to enhance the performance of PSO: our improved algorithm is called Improved Particle Swarm Optimization (IPSO).

In our proposed algorithm, first the prematurity state of the algorithm is judged against the following conditions after each given generation. Let's define

$$\bar{f} = \frac{1}{n} \sum_{i=1}^n f_i, \quad \sigma_f^2 = \frac{1}{n} \sum_{i=1}^n (f_i - \bar{f})^2 \quad (16)$$

Where f_i is the fitness value of the i th particle, n is the number of the particles in the population, \bar{f} is the average fitness of all the particles, and σ_f^2 is the variance, which reflects the convergence degree of the population. Moreover, we define the following indicator, τ^2 :

$$\tau^2 = \frac{\sigma_f^2}{\bar{f}^2} \quad (17)$$

If τ^2 is less than a small given threshold, decided by the algorithm's user, and the theoretical

global optimum or the expectation optimum has not been found, the algorithm is considered to get into a premature convergence state.

In the case, we identify those inactive particles by use of the inequality

$$\frac{f_g - f_i}{\max\{(f_g - f_j), (j = 1, \Lambda, n)\}} \leq \theta \quad (18)$$

where f_g is the fitness of the best particle g_{best} and θ is a small given threshold decided by the user. In this paper, τ^2 and θ are taken as 0.005 and 0.01 respectively.

Finally the inactive particles are chosen to mutate by using a Gauss random disturbance on them according to formula (19), while at the same time only one of the best particles is retained.

$$p_{ij} = p_{ij} + \beta_{ij} \quad (j = 1, \Lambda, D) \quad (19)$$

where p_{ij} is the j th component of the i th inactive particle; β_{ij} is a random variable and follow a Gaussian distribution with zero mean and constant variance 1, namely $\beta_{ij} \sim N(0,1)$.

4. IPSO-based Learning Algorithm for Elman Neural Network (IPBEA)

Let's denote the location vector of a particle as X , and its ordered components as self-feedback coefficients, i.e. initial inputs of the context unit and weights. In the proposed algorithm, a “particle” consists of two parts:

- the first part is named “head”, and comprises the self-feedback coefficients;
- the second part is named “body”, and includes the initial inputs of the context unit and all the weights.

As far as the network shown in Figure 1 is concerned (where there are r nodes in the input layer, n nodes in the hidden and context layers, and m nodes in the output layer) the corresponding “particle” structure can be illustrated as in Figure 2. There $\tilde{X}_c^0 = (x_{c,1}^0, \Lambda, x_{c,n}^0)$ is a permutation of

the initial inputs of the context unit, \tilde{W}^{I1} , \tilde{W}^{I2} and \tilde{W}^{I3} are the respective permutations of the expansion of weight matrices W^{I1} , W^{I2} and W^{I3} by rows. Therefore, the number of the elements in the body is $n+n \cdot n+r \cdot n+n \cdot m$. While coding the parameters, we define a lower and an upper bound for each parameter being optimized. This restricts the search space of each parameter, and thus the specified bounds need to be validated against the given problem's context. In our experiments, related to the USMs, the lower and upper bounds for weight matrices are taken as -2 and +2 respectively.

In IPBEA searching process, two additional operations are introduced: namely, the “structure developing” operation and the “structure degenerating” operation. They realize the evolution of the network structure, and more specifically, they determine the number of neurons of the hidden layer. Adding (“structure developing”) or deleting (“structure degenerating”) neurons in the hidden layer is judged against the developing probability p_a and the degenerating probability p_d , respectively. If a neuron is added, the weights related to the neuron are added synchronously: such values are randomly set according to their initial range. If the degenerating probability p_d passes the Bernoulli trials, a neuron of the hidden layer is randomly deleted, and the weights related to the neuron are set to zero synchronously. In order to maintain the dimensionality of the particle, the maximal number of the neurons in the hidden layer is given, and taken as 10 in this paper. The evolution of the self-feedback coefficient α in the part of the head lies on the probability p_e . The probabilities p_a , p_d and p_e are given by the following equation

$$p_a = p_d = p_e = e^{-\frac{1}{N_g \cdot \gamma}} \quad (20)$$

where N_g represents the number of generations that the maximum fitness has not been changed, and is taken as 50; γ is an adjustment coefficient, which is taken as 0.03 in this paper. It is worth mentioning here that the probabilities p_a , p_d and p_e are adaptive with the change of N_g .

The elements in the body part are updated according to Eqs. (14) and (15) in each iteration, while the element α is updated (always using Eqs. (14) and (15)) only if p_e passes the Bernoulli trials.

Moreover, the inertia coefficient w is adjusted in the direction of linear decrease using the following equation, according to Reference [21]:

$$w(k) = w_{\max} - w_{\min} \times (k/gen_{\max}) \quad (21)$$

where $w(k)$ is the inertia coefficient in the k th iteration, w_{\max} and w_{\min} are the maximum and the minimum of the inertia coefficient, respectively, and gen_{\max} is the maximum generation of iterations. In our experiments, related to the USMs, we have defined $w_{\max} = 1.5$, and $w_{\min} = 1.0$.

5. USM Speed identification using the IPSO-based Elman Neural Network

In this section, we present and discuss a dynamic identifier to perform the identification of non-linear systems based on the IPSO-based Elman Neural Network proposed in the previous section. We named the novel dynamic identifier as IPSO-based ENN Identifier (IPBEI). The proposed model can be used to identify highly non-linear systems. In the following, we have considered a simulated dynamic system of the ultrasonic motor as an example of a highly nonlinear system. In fact, non-linear and time-variant characteristics are inherent of an ultrasonic motor.

Numerical simulations have been performed using the model of IPBEI for the speed identification of a longitudinal oscillation USM [25] shown in Figure 3. Some parameters on the USM model in our experiments are taken as follows: driving frequency 27.8 kHz , amplitude of driving voltage 300 V , allowed output moment $2.5 \text{ kg} \cdot \text{cm}$, rotation speed 3.8 m/s . The parameters on the IPSO are taken as: population scale 80, learning rates $c_1 = 1.9$ and $c_2 = 0.8$. The initial number of the neurons in the hidden layer is 10. The Block diagram of the identification model of the motor is shown in Figure 4.

In the simulated experiments, the Elman neural network is trained on line by the IPSO algorithm. The fitness of a particle is evaluated by the reciprocal of the mean square error, namely

$$f_j(k) = \frac{1}{E_j(k)} = p / \sum_{i=k-p+1}^k (y_d(i) - y_j(i))^2 \quad (22)$$

where $f_j(k)$ is the fitness value of the particle j at time k , p is the width of the identification window (taken as 1 in this paper), $y_d(i)$ is the expected output at time i , and $y_j(i)$ is the actual output corresponding to the solution found by particle j at time i . The smaller the identification window, the higher identification precision will be obtained. The iterations continue until a termination criterion is met, where a sufficiently good fitness value or a predefined maximum number of generation is achieved in the allowed time interval. Upon the identification of a sampling step, the particles produced in the last iteration are stored as an initial population for the next sampling step: only 20% of them are randomly initialized. In fact, these stored particles are good candidate guesses for the solution for the next step, especially if the system is close to the desired steady-state. In our numerical experiments, the use of the described techniques has significantly reduced the number of generations needed to calculate an acceptable solution.

In order to show the effectiveness and accuracy of the identification by the proposed method, a durative external moment of $1 \text{ N}\cdot\text{m}$ is applied in the time window $[0.4, 0.7\text{s}]$ to simulate an external disturbance. The curve of the actual motor speed is shown as curve a in Figure 5, and curve b is a zoom of curve a at the stabilization stage. Figures 6 to 11 show the respective identification results. The proposed IPBEI model is compared with the original Elman model using the gradient descent-based learning algorithm. In all the following figures, the motor curve is the actual speed curve of the USM, represented by the solid line and the symbol “×”; the Elman curve is the speed curve identified using the Elman model with gradient-descent-based learning algorithm, and represented by the solid line and the symbol “●”; the IPBEI curve is the speed curve identified using the IPBEI model, and represented by the short dot line and the symbol “▲”. The Elman error

curve is the error curve identified using the Elman model with gradient-descent-based learning algorithm, and the IPBEI error curve is the error curve identified using the IPBEI model, in which the error is the difference between the identification result and the actual speed.

Figure 6 shows the speed identification curves for the initial stage, with the exclusion of the first 10 sampling data, obtained from different methods, The maximal identification error using the proposed method is not larger than 0.004, which is obviously superior to the maximal identification error 3.5 obtained by using the gradient-descent-based learning algorithm. Figure 6 shows that, in this experiment, the proposed IPBEI model can approximate the optimum solution very fast and it has the ability to produce good solutions in the sampling interval, since the results at the initial stage, except for the first 10 sampling data, are stable.

Figures 7 and 8 show respectively the speed identification curves and error curves for the disturbance stage. Figures 9 and 10 respectively show the speed identification curves and error curves for the stabilization stage. These results demonstrate the full power and potential of the proposed method. If we compare the identification errors obtained with the two methods, we see that in the gradient descent-based learning algorithm they are ca. 0.005, while in the proposed method they are about an order of magnitude smaller, i.e. less than 0.0004. In other words, the identification error of the IPBEI is about 8% that of the Elman model trained by the gradient descent algorithm, and the identification precision is more than 99.98%. Such precise results are mainly attributed to the fast convergence of the improved PSO learning algorithm. Besides, the identifier IPBEI runs on line, and the samples are identified one by one.

Our simulated experiments of the identification algorithms have been carried out on a PC with Pentium IV 2.8 GHz processor and 512MB memory. There were 21000 sampling data and the whole identification time has been about 6.2 seconds. The average CPU-time for the identification of a sampling data has been about 0.03 msec.

The proposed on-line identification model and strategy can be successfully used to identify

highly non-linear systems. Moreover, the on-line learning and estimation approach can identify and update the parameters required by the model to ensure model accuracy when condition changes.

6. USM Speed control using the IPSO-based Elman Neural Network

In this section, we present and discuss a novel controller specially designed to control non-linear systems using the IPSO-based Elman network, which we name IPSO-based ENN Controller (IPBEC). The proposed on-line control strategy and model can be used for any type of non-linear systems especially when a direct controller cannot be designed due to the complexity of the process and related system model. The USM used in Section 5 is still considered as an example of a highly nonlinear system to test the performance of the proposed controller. The optimized control strategy and model is illustrated in Figure 11.

In the developed IPBEC, the Elman network is trained on line by the IPSO algorithm, proposed in section 3, and the driving frequency is taken as the control variable. The fitness of a particle is evaluated computing the deviation of the control result over the expected result from a desired trajectory, which is formulated as follows

$$f_j(i) = 1/e_j(k)^2 = 1/(y_d(i) - y_j(i))^2 \quad (23)$$

where $f_j(k)$ is the fitness value of the particle j at sampling time i , $y_d(i)$ is the expected output at time i and $y_j(i)$ is the actual output corresponding to the solution found by particle j at time i . In order to deal with real-time control, the algorithm stops after a maximum allowed time has passed. In our experiments, during each discrete sampling interval the control algorithm is allowed to run for 1 *ms*, which is equal both to the time of the sampling interval and to the time available for calculating the next control action. In a similar way to Section 5, after a sampling step the produced particles in the last iteration are stored as an initial population for the next sampling

step, and only 20% of the particles are randomly initialized. Control results (figure 12 to 16) show that the proposed algorithm can approximate the optimal solution rapidly, and the identified solutions are accurate and acceptable for practical and real-time control problems.

Figure 12 shows the USM speed control curves using three different control strategies when the control speed is taken as 3.6 m/s . In the figure, the dotted line a represents the speed control curve based on the method presented by Senjyu et al.[26], the solid line b represents the speed control curve using the method presented by Shi et al.[27] and the solid line c represents the speed curve using the method proposed in this paper. Simulation results show that the stable speed control curves obtained by using the three methods possess different fluctuation behaviors. The existing neural-network-based methods for USM control have lower convergent precision and it is therefore more difficult to obtain the accurate control input for the USM. From Figure 12 it can be seen that the amplitude of the speed fluctuation using the proposed method is significantly smaller at the steady state than the other methods. The fluctuation degree is defined as

$$\zeta = (V_{\max} - V_{\min}) / V_{\text{ave}} \times 100\% \quad (24)$$

where V_{\max} , V_{\min} and V_{ave} represent the maximum, minimum and average values of the speeds. As reported in Figure 12, the maximum fluctuation values when using the methods proposed by Senjyu and Shi are 5.7% and 1.9% respectively, whereas they are reduced significantly to 0.06% when using the method proposed in this paper. The control errors when using the methods proposed by Senjyu and Shi are about 0.1 and 0.034 respectively, while the error is kept within 0.001 (again more than one order of magnitude smaller) when using the proposed IPBEC. Figure 13 shows an enlargement of the control curve using the IPBEC controller.

The speed control curves of the referenced values varying with time are also examined to further verify the control effectiveness of the novel method. Figure 14 shows the speed control curves, where the reference speed varies first step-wise and then follows a sinusoidal behavior: the solid line represents the reference speed curve while the dotted line represents the speed control

curve based on the method proposed in this paper. Figure 15 shows an enlargement of Figure 14 in the time window [10.8s, 11.2s], where there is a trough of the reference speed curve. From the two figures it can be seen that the proposed controller performed successfully and possesses a high control precision.

For the sake of verifying preliminarily the robustness of the proposed control system, we examine the response of the system when an instantaneous perturbation is added into the control system. The speed reference curve is the same to that in Figure 14. Figure 16 shows the speed control curve when the driving frequency is subject to an instantaneous perturbation (5% of the driving frequency value) at time = 6 seconds. From the figure it can be seen that the control model possesses a rapid adaptive behavior against the randomly instantaneous perturbation on the frequency of the driving voltage. Such behavior suggests that the controller presented here exhibits a robust anti-noise performance and can handle a variety of operating conditions without losing the ability to track accurately a desired course.

7. Conclusions

The proposed learning algorithm for Elman neural networks based on the improved PSO overcomes some known shortcoming of ordinary gradient descent methods, namely (1) their sensitivity to the selection of initial values and (2) their propensity to lock into a local extreme point. Moreover, training dynamic neural networks by IPSO does not need to calculate the dynamic derivatives of weights, which reduces significantly the calculation complexity of the algorithm. Besides, the speed of convergence is not dependent on the dimension of the identified and controlled system, but is only dependent on the model of neural networks and the adopted learning algorithm. The proposed learning algorithm guarantees the rationality of the algorithm and realizes concurrently the evolution of network construct, weights, initial inputs of the context unit and self-feedback coefficient of the Elman network. In this paper, we have described, analyzed and discussed an identifier IPBEI and a controller IPBEC designed to identify and control non-linear

systems on line. When the system is disturbed by an external noise, it can learn on line and adapt in real-time to the nonlinearity and uncertainty. Our numerical experiments show that the designed identifier and controller can achieve both higher convergence precision and speed, relative to current state-of-the-art other methods. The identifier IPBEI can approximate with high precision (error less than 0.0004) the nonlinear input-output mapping of the USM, and the effect and applicability of the controller IPBEC are verified using different kinds of speeds of constant, step, and sinusoidal types. Besides, the preliminary examination on a random perturbation also shows the robust characteristics of the two models. The methods described in this paper can provide effective approaches for non-linear dynamic systems identification and control. More detailed theoretical analyses on the robustness and convergence for the identification and speed control of the USM using the proposed methods are currently being investigated.

Acknowledgment

The first two authors are grateful to the support of the National Natural Science Foundation of China (60673023,60433020), the science-technology development project of Jilin Province of China (20050705-2), the doctoral funds of the National Education Ministry of China (20030183060), and “985” project of Jilin University of China. The last two authors would like to thank the support of the European Commission under grant No. TH/Asia Link/010 (111084) and the Erasmus Mundus programme of the EU.

Appendix A. The mathematic model of the Longitudinal Oscillation USM

The mathematic model of the longitudinal oscillation USM [25] used in this paper is represented by the following state space equations:

$$F = -K\Delta l = -K[u(t)] - x_0 - (y(l,t) - y_0) \tan \alpha, \quad (\text{A.1})$$

$$\left\{ \begin{array}{l} EI \frac{\partial^4 y(x,t)}{\partial x^4} + \rho_0 S \frac{\partial^2 y(x,t)}{\partial t^2} + \frac{\partial}{\partial x} (F(x,t) \frac{\partial y(x,t)}{\partial x}) = P \delta(x-l) \cos \alpha \\ y(0,t) = y'(0,t) = y''(l,t) = y'''(l,t) = 0 \\ y(l,0) = y_0, \quad \dot{y}(l,0) = \dot{y}_0 \end{array} \right. , \quad (\text{A.2})$$

$$v_y = \frac{\partial y(l,t)}{\partial t} \cos \alpha, \quad (\text{A.3})$$

$$P = -F_r \operatorname{sgn}(\Delta x) = \begin{cases} -(F \sin \alpha + \mu_c F \cos \alpha) \operatorname{sgn}(\Delta x) & (\Delta v \geq 0) \\ -(F \sin \alpha - \mu_c F \cos \alpha) \operatorname{sgn}(\Delta x) & (\Delta v < 0) \end{cases}, \quad (\text{A.4})$$

$$F_r = F \sin \alpha + \mu F \cos \alpha, \quad (\text{A.5})$$

$$J \frac{d\Omega}{dt} = F_r(t)R - M, \quad (\text{A.6})$$

where $u(t) = A \sin(pt)$, $\Delta v = v_y - v_r$, $v_r = r\Omega$.

Explanations of the above symbols are given in the following nomenclature.

Nomenclature

F	Elastic force of the stator in the longitudinal direction
p	Driving frequency
A	Amplitude of driving voltage
K	Elastic coefficient in the longitudinal direction of the piece
α	Angle between the rotor surface and the vertical
l	Length of the stator
x_0	Initial displacement of the stator tip in the longitudinal direction
y_0	Initial displacement of the stator tip in the flexural direction
E	Young's modulus of elasticity
I	Area moment of inertia about an axis normal to the plane
ρ_0	Density of material
S	Area of the cross-section of the beam
$F(x,t)$	Axial compressive force assumed to be equal to F
P	Resultant force parallel to the rotor surface
\dot{y}_0	Initial velocity of the stator tip in the flexural direction
μ_c	Coefficient of slipping friction between the stator tip and the rotor
v_y	Velocity of the stator tip
v_r	Linear speed of the rotor at the contact point
r	Radius from the center of the plate to the contact point
F_r	Driving force produced by the resultant force parallel to the rotor surface
μ	Friction coefficient during the sticking phase
Ω	Angular velocity of the rotor
J	Moment of inertia of the rotor
M	External moment
R	Radius of the rotor
$\delta(x)$	Dirac delta function
$\operatorname{sgn}(\cdot)$	Sign function

References

- [1] Hayakawa T, Haddad WM, Bailey JM, Hovakimyan N. Passivity-based neural network adaptive output feedback control for nonlinear nonnegative dynamical systems. *IEEE Transactions on Neural Networks* 2005; 16(2):387-398.
- [2] Li YM, Liu YG, Liu XP. Active vibration control of a modular robot combining a back-propagation neural network with a genetic algorithm. *Journal of Vibration and Control* 2005; 11(1):3-17.
- [3] Sunar M, Gurain AMA, Mohandes M. Substructural neural network controller. *Computers & Structures* 2000; 78(4):575-581.
- [4] Xu X, Liang YC, Shi XH, Liu SF. Identification and bimodal speed control of ultrasonic motors using input-output recurrent neural networks. *Acta Automatica Sinica* 2003; 29(4):509-515.
- [5] Yu W, Li XO. System identification using adjustable RBF neural network with stable learning algorithms. *Lecture Notes in Computer Science* 2004; 3174: 212-217.
- [6] Wang D, Huang J. Neural network-based adaptive dynamic surface control for a class of uncertain nonlinear systems in strict-feedback form. *IEEE Transactions on Neural Networks* 2005; 16(1):195-202.
- [7] Sashida T, Kenjo T. *An Introduction to Ultrasonic Motors*. Oxford: Clarendon Press, 1993.
- [8] Lino A, Suzaki K, Kasuga M, Suzuki M, Yamanaka T. Development of a self-oscillating ultrasonic micro-motor and its application to a watch. *Ultrasonics* 2000; 38(1-8):54-59.
- [9] Hemsell T, Wallaschek J. Survey of the present state of the art of piezoelectric linear motors. *Ultrasonics* 2000; 38(1-8):37-40.
- [10] Lin FJ, Wai RJ, Hong CM. Identification and control of rotary traveling-wave type ultrasonic motor using neural networks. *IEEE Transactions on Control Systems Technology* 2001; 9(4):672-680.
- [11] Senjyu T, Miyazato H, Yokoda S, Uezato K. Speed control of ultrasonic motors using neural network. *IEEE Transactions on Power Electronics* 1998; 13(3):381-387.
- [12] Xiong ZH, Zhang J. A batch-to-batch iterative optimal control strategy based on recurrent neural network models. *Journal of Process Control* 2005; 15(1):11-21.
- [13] Lin FJ, Wai RJ, Chou WD, Hsu SP. Adaptive backstepping control using recurrent neural

network for linear induction motor drive. IEEE Transactions on Industrial Electronics 2002; vol. 49(1):134-146.

- [14]Tang WS, Wang J. A recurrent neural network for minimum infinity-norm kinematic control of redundant manipulators with an improved problem formulation and reduced architecture complexity. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 2001; 31(1):98-105.
- [15]Tian LF, Wang J, Mao ZY. Constrained motion control of flexible robot manipulators based on recurrent neural networks. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 2004; 34(3):1541-1552.
- [16]Cong S, Gao XP. Recurrent neural networks and their application in system identification. Systems Engineering and Electronics 2003; **25**(2):194-197.
- [17]Pham DT, Liu X. Dynamic system modeling using partially recurrent neural networks. J. of Systems Engineering 1992; 2:90-97.
- [18]Elman JL. "Finding structure in time. Cognitive Science 1990; 14(2):179-211.
- [19]He ZY, Han YQ, Wang HW. A modular neural networks based on genetic algorithm for FMS reliability optimization. International Conference on Machine Learning and Cybernetics. Piscataway, NJ: IEEE Press. Xi'an, China, 2003.
- [20]Kennedy J, Eberhart R. Particle swarm optimization. Proceedings of the IEEE International Conference on Neural Networks. Piscataway, NJ: IEEE Press. Perth, Australia, 1995.
- [21]Shi Y, Eberhart R. A modified particle swarm optimizer. IEEE International Conference on Evolutionary Computation. Piscataway, NJ: IEEE Press. Anchorage, Alaska, USA, 1998.
- [22]Angeline PJ. Evolutionary optimization versus particle swarm optimization: philosophy and performance differences. Evolutionary Programming 1998; 7:601-610.
- [23]Clerc M, Kennedy J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation 2002; 6:58-73.
- [24]Trelea IC. The particle swarm optimization algorithm: convergence analysis and parameter selection. Information Processing Letters 2003; 85:317-325.
- [25]Xu X, Liang YC, Lee HP, Lin WZ, Lim SP, Lee KH. Mechanical modeling of a longitudinal oscillation ultrasonic motor and temperature effect analysis. Smart Materials and Structures 2003; 12(4):514-523.

- [26] Senjyu T, Miyazato H, Yokoda S, Uezato K. Speed control of ultrasonic motors using neural network. *IEEE Transactions on Power Electronics* 1998; 13(3):381-387.
- [27] Shi XH, Liang YC, Lee HP, Lin WZ, Xu X, Lim SP. Improved Elman networks and applications for controlling ultrasonic motors. *Applied Artificial Intelligence* 2004; 18(7): 603-629.

Figure captions

Figure 1. Architecture of the modified Elman network

Figure 2. Structure of a particle

Figure 3. Schematic diagram of the motor

Figure 4. Block diagram of identification model of the motor

Figure 5. Actual speed curve of the USM with a durative external disturbance

Figure 6. Speed identification curves for the initial stage

Figure 7. Speed identification curves for the disturbance stage

Figure 8. Speed identification error curves for the disturbance stage

Figure 9. Speed identification curves for the stabilzition stage

Figure 10. Speed identification error curves for the stabilzition stage

Figure 11. Block diagram of the speed control system

Figure 12. Comparison of speed control curves using different schemes

Figure 13. An enlargement of the control curve using the IPBEC controller

Figure 14. Speed control curves with varied reference speeds

Figure 15. An enlargement of Figure 14 in the time windows [10.8s, 11.2s]

Figure 16. Speed control for randomly instantaneous disturbance

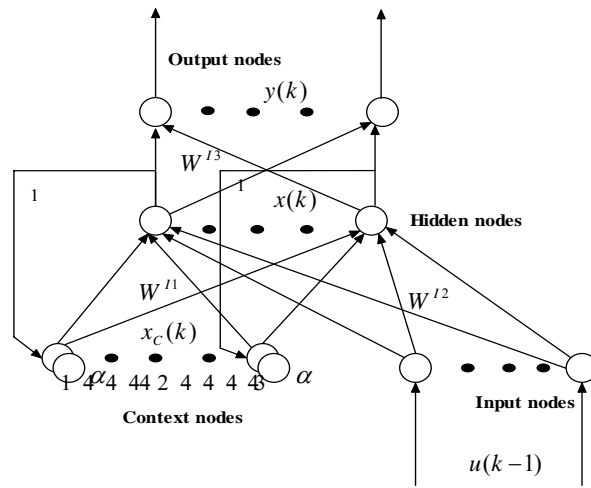


Figure 1. Architecture of the modified Elman network

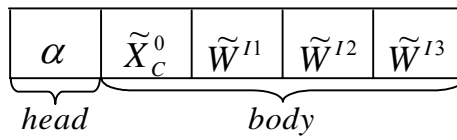


Figure 2. Structure of a particle

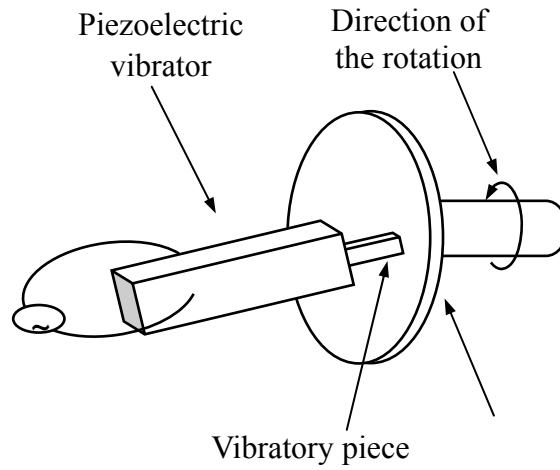


Figure 3. Schematic diagram of the motor

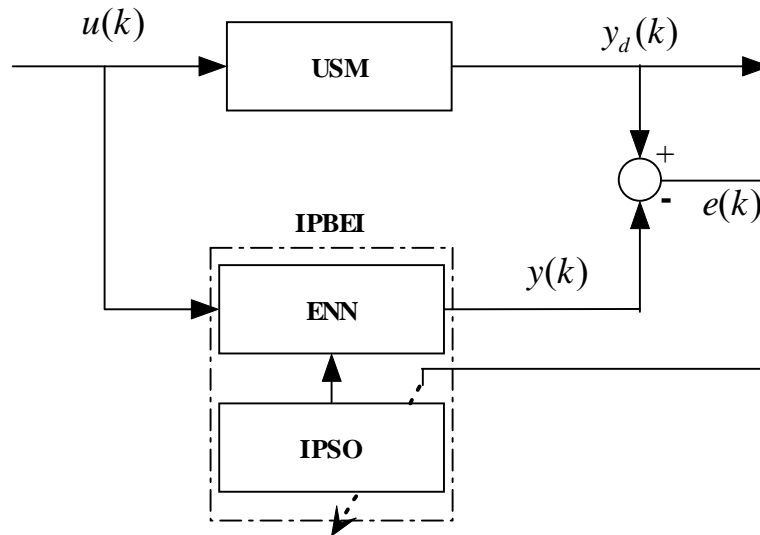


Figure 4. Block diagram of identification model of the motor

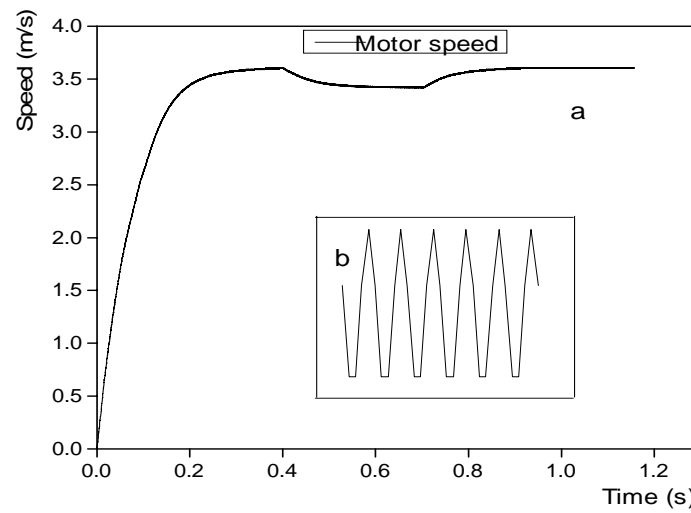


Figure 5. Actual speed curve of the USM with a durative external disturbance

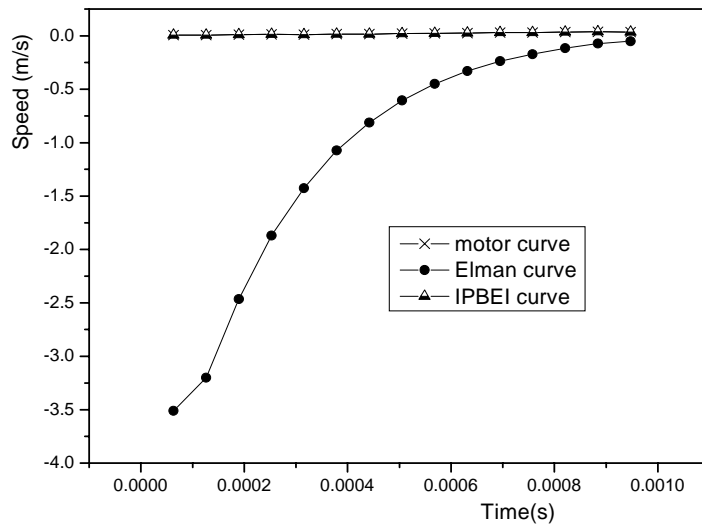


Figure 6. Speed identification curves for the initial stage

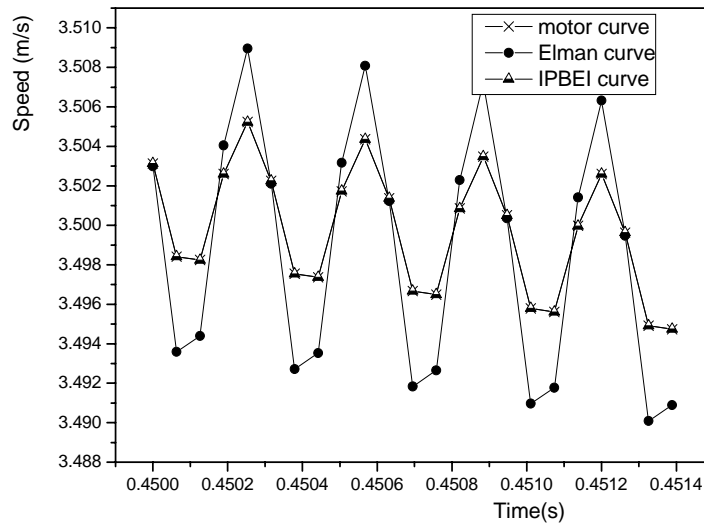


Figure 7. Speed identification curves for the disturbance stage

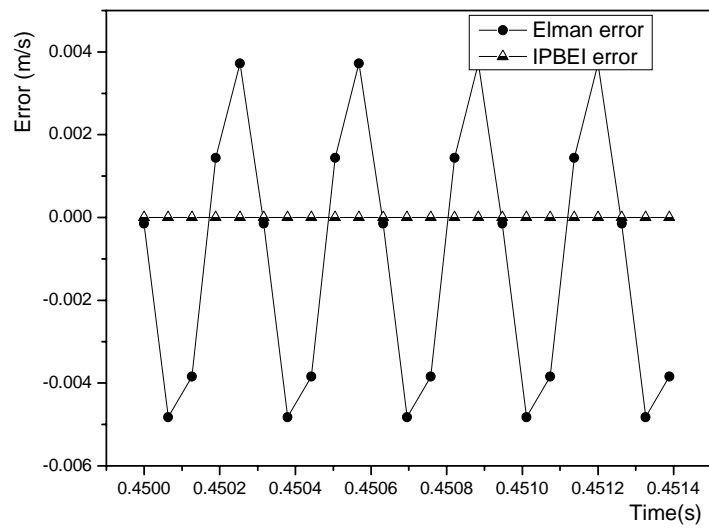


Figure 8. Speed identification error curves for the disturbance stage

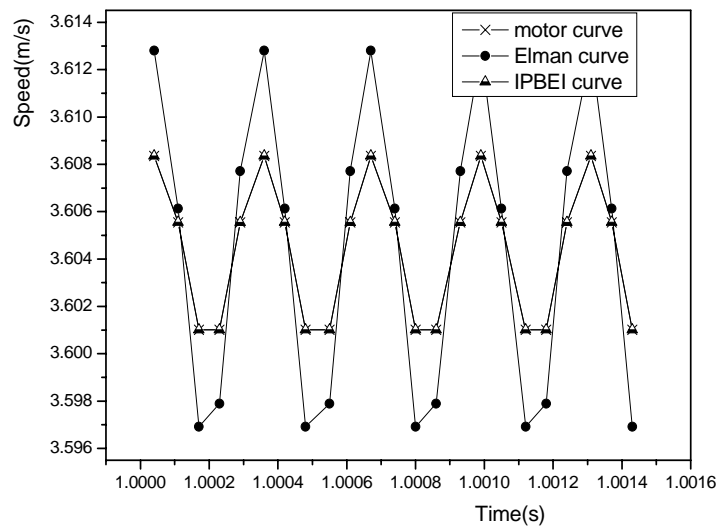


Figure 9. Speed identification curves for the stabilization stage

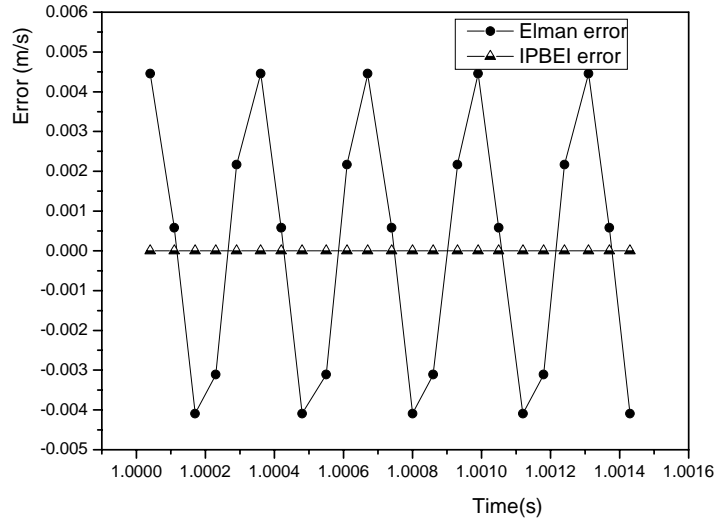


Figure 10. Speed identification error curves for the stabilization stage

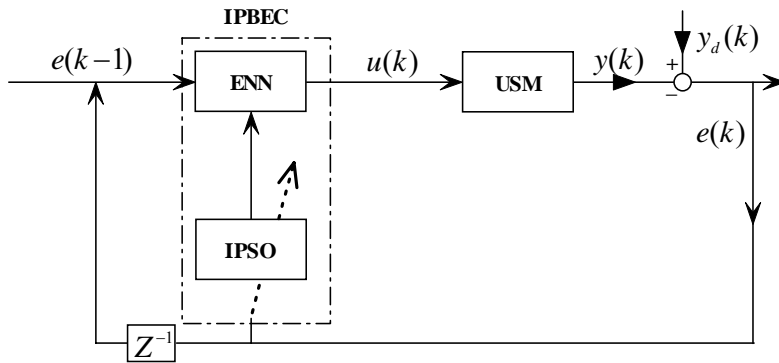


Figure 11. Block diagram of the speed control system

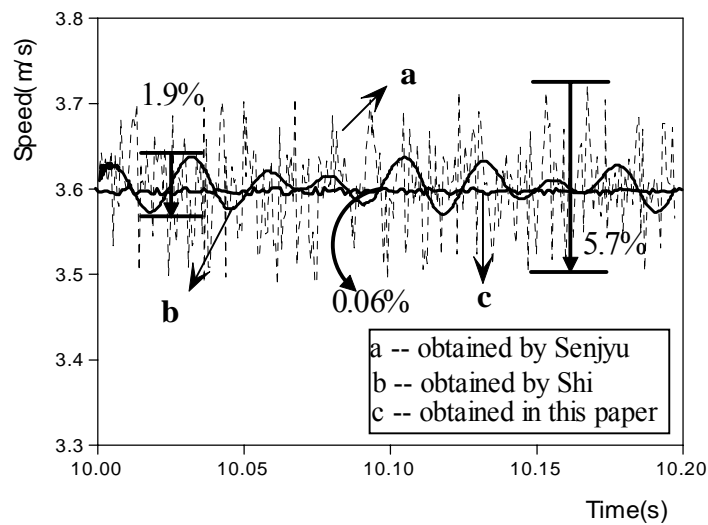


Figure 12. Comparison of speed control curves using different schemes

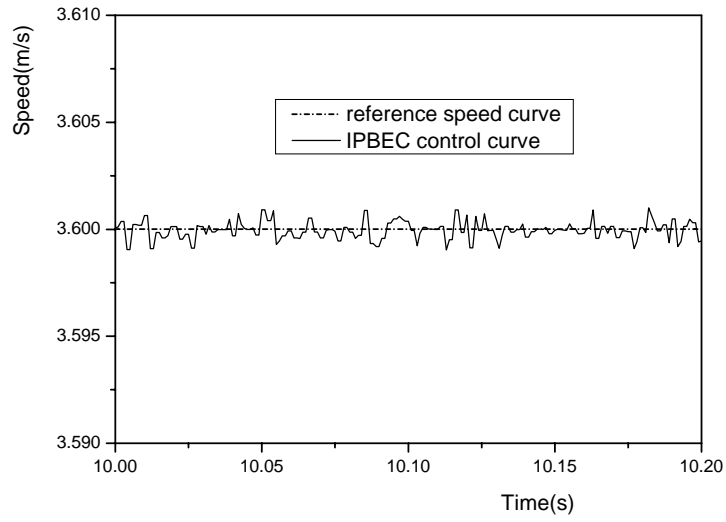


Figure 13. An enlargement of the control curve using the IPBEC controller

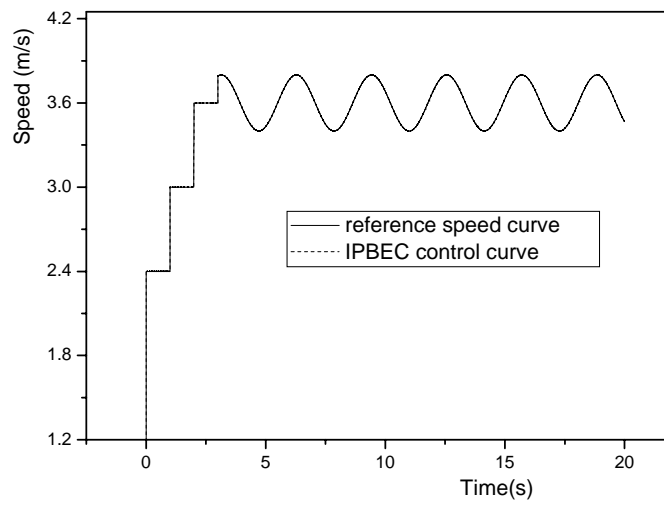


Figure 14. Speed control curves with varied reference speeds

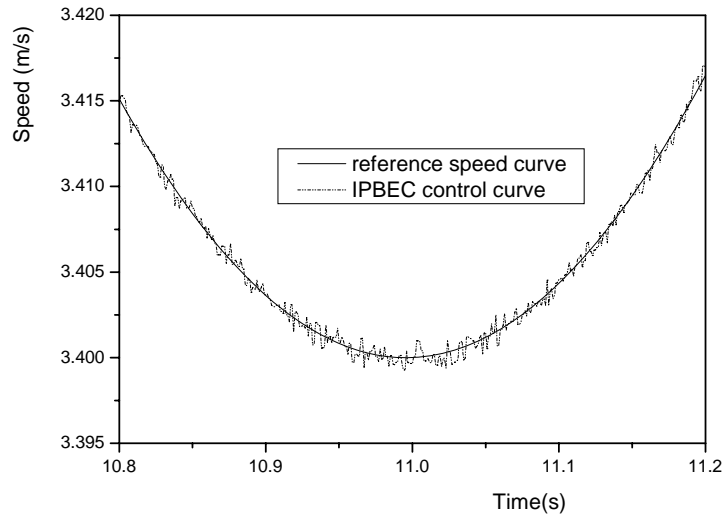


Figure 15. An enlargement of Figure 14 in the time window [10.8s, 11.2s]

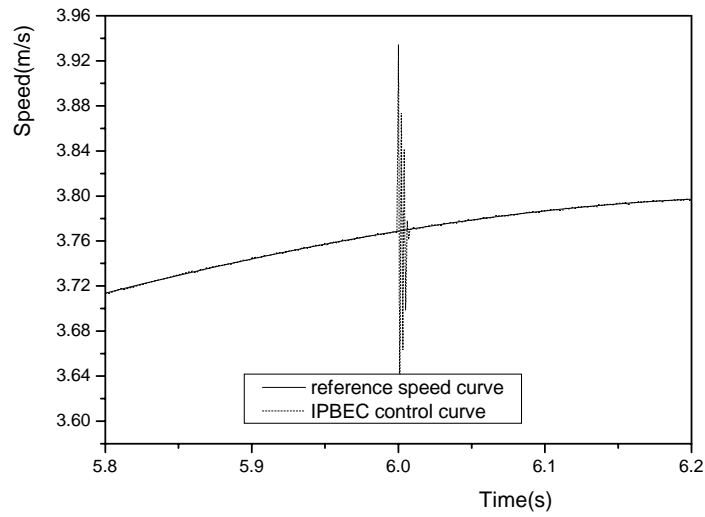


Figure 16. Speed control for randomly instantaneous disturbance