Supervised and Reinforcement Group-Based Hybrid Learning Algorithms for TSK-type Fuzzy Cerebellar Model Articulation Controller

Jyun-Yu Jhang*, Cheng-Jian Lin, Lingling Li*****

**Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu 300, Taiwan.*
*** Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung 411, Taiwan, (e-mail: cjlin@ncut.edu.tw)*
****Electrical Apparatus Institute, Hebei University of Technology, Tianjin 300130, China*

Abstract: In this study, a Takagi–Sugeno–Kang (TSK)-type fuzzy cerebellar model articulation controller (T-FCMAC) based on a group-based hybrid learning algorithm (GHLA) was proposed for solving various problems. The proposed T-FCMAC model was mainly derived from a traditional cerebellar model articulation controller and the TSK-type fuzzy model. For supervised learning, the proposed GHLA was developed by combining an improved quantum particle swarm optimization algorithm and the Nelder–Mead method for adjusting the parameters of a T-FCMAC. The fuzzy C-mean clustering technique was adopted to improve the performance of quantum particle swarm optimization. A fitness threshold was used to determine the number of clusters in fuzzy C-mean clustering. The grouping concept was also used to improve the search ability and increase the convergence rate. Moreover, because exact training data may be expensive or even impossible to obtain in some real-world applications, a reinforcement GHLA (R-GHLA) was proposed. Experimental results revealed the performance and applicability of the proposed GHLA and R-GHLA.

Keywords: Fuzzy CMAC, Nelder–Mead, fuzzy C-mean, particle swarm optimization, control, reinforcement learning.

1. INTRODUCTION

The cerebellar model articulation controller (CMAC) was proposed by Albus (Albus, 1975). There are three major limitations in the CMAC proposed by Albus. First, the CMAC requires an enormous amount of memory to solve high-dimensional problems (Reay, 1995). Second, the CMAC requires a more rigorous theory for function approximation. Third, it is difficult for the CMAC to select the appropriate memory structure parameters (Chow and Menozzi, 1994). Therefore, the fuzzy concept has been integrated into the CMAC in many studies (Lin and Li, 2015; Zhang and Qian, 2000), and the resulting structure is known as fuzzy CMAC (FCMAC).

The two most commonly used fuzzy models are Mamdani-type (Kaur and Kaur, 2012) and Takagi–Sugeno–Kang (TSK)-type fuzzy models (Kaur and Kaur, 2012). The main motivation for developing the TSK-type fuzzy model was to reduce the number of rules required by the Mamdani-type fuzzy model, especially for complex and high-dimensional problems. The linear equation of the input variables in the consequent (then part) of the TSK-type fuzzy model was used to replace the fuzzy sets of the Mamdani rule. In this study, a new TSK-type fuzzy CMAC (T-FCMAC) was proposed and its network structure was mainly derived from the traditional CMAC and TSK-type fuzzy models.

The backpropagation (BP) algorithm (Wang et al., 2016) is commonly used to adjust the parameters of the models during parameter learning. As the BP algorithm is based on the gradient descent method, some drawbacks usually occur due to characteristics such as slow convergence rate and frequently being trapped in local minima. Moreover, popular parameter learning approaches have been used in the stochastic gradient descent (SGD) method (Leon, 1998; John et al., 2011). Although SGD can be implemented easily and has a high convergence rate, it requires a number of hyperparameters. Therefore, to overcome the aforementioned drawbacks, several evolutionary learning algorithms have been proposed, such as the genetic algorithm (GA) (Lin et al., 2016), the particle swarm optimization (PSO) algorithm (Eberhart and Kennedy, 1995), the differential evolution algorithm (Ilonen et al., 2003).

Recently, PSO has been widely used in various fields. Traditional PSO systems converge quickly, but can easily fall into local minima. Thus, many researchers focused on improving the traditional PSO. In 2004, to combine quantum theory and PSO algorithms, a quantum PSO (QPSO) algorithm was proposed by (Sun et al., 2004). QPSO performs well on a wide range of continuous optimization problems (Xi et al., 2008).

For PSO and QPSO, the positions of "individual cognition" and "social interaction" are required for updating the position of each particle. That is, the system calculates the personal best (Pbest) position and global best (Gbest) position. If the effect of "social interaction" is very strong, then the particles can easily fall into a local optimum. However, several

improved PSO and QPSO algorithms have been used for solving this problem. Multi-swarm optimization has been widely used to overcome diversity loss, such as niching (Li, 2012) and cooperative coevolution (Li and Yao, 2012). Moreover, using dynamic PSO for changing the constitution of a swarm dynamically seems to be an effective strategy (Hsieh, Sun, Liu, and Tsai, 2009). In this study, a group-based hybrid learning algorithm (GHLA) was proposed to overcome the aforementioned drawbacks.

This study proposed a T-FCMAC with a GHLA for solving prediction and control problems. GHLA was developed by combining an improved QPSO (IQPSO) algorithm and the Nelder–Mead (NM) method for adjusting the parameters of a T-FCMAC. The fuzzy C-mean clustering technique was adopted to improve the performance of QPSO. The grouping concept was used to reform the search ability and greatly increase the convergence rate. Both the time series prediction and the water bath control problems were considered to evaluate the performance of the proposed GHLA. For some real-world applications, it is usually difficult and expensive to obtain relevant training data. To solve the aforementioned problem, a reinforcement GHLA (R-GHLA) was proposed in this study. The proposed R-GHLA records the number of failures as a design fitness function method and uses a GHLA for adjusting parameters of the T-FCMAC. Experimental results revealed that the proposed method is better than other methods in terms of supervised learning and reinforcement learning.

2. STRUCTURE OF THE T-FCMAC

In this section, a novel T-FCMAC is presented. The architecture of the T-FCMAC is illustrated in Fig. 1 and comprises nonlinear input mapping, internal mapping, TSK-type output, and defuzzification.

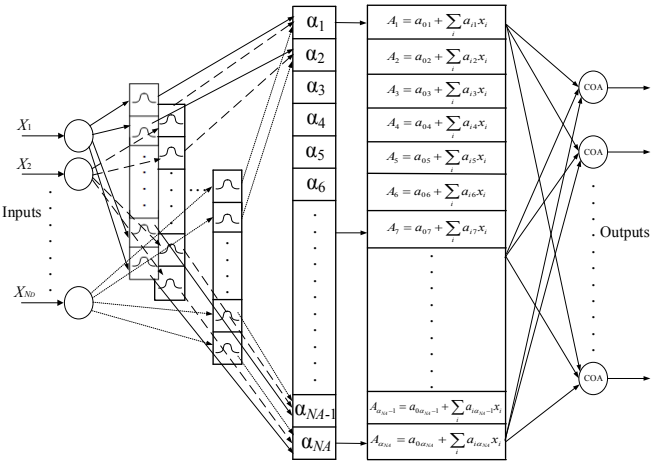


Fig. 1. Architecture of the T-FCMAC.

Fig. 1 indicates that learned information is stored in the receptive field functions and TSK-type output vectors. A one-dimensional Gaussian basis function can be given as follows:

$$\mu(x) = e^{-(\frac{x-m}{\sigma})^2} \tag{1}$$

where x represents the specific input state, m represents the corresponding center, and σ represents the corresponding variable. A Gaussian basis function with N_D dimensions is given as follows:

$$\alpha_j = \prod_{i=1}^{N_D} e^{-\left(\frac{x_i - m_{ij}}{\sigma_{ij}}\right)^2} \tag{2}$$

where $\prod$ represents the product operation, α_j represents the j-th element of the association vector, x_i represents the input value of the i-th dimension for a specific input state x, m_{ij} represents the mean of the Gaussian functions, σ_{ij} represents the deviation of the Gaussian functions, and N_D represents the number of receptive field functions for each input state. Each fuzzy hypercube cell of the receptive field functions is inferred to produce a partial fuzzy output by applying the value of its corresponding association vector as an input matching degree. The output of Layer 2 is defuzzified into a scalar output y by using the centroid of area approach. Then, the actual output y is derived as follows:

$$y = \frac{\sum_{j=1}^{N_L} \alpha_j \left(\alpha_{0j} + \sum_{i=1}^{N_D} a_{ij} x_j\right)}{\sum_{j=1}^{N_L} \alpha_j} \tag{3}$$

The j-th element of the TSK-type output vectors is described as follows:

$$a_{0j} + \sum_{i=1}^{N_D} a_{ij} x_i \tag{4}$$

where a_{oj} and a_{ij} denote the scalar value, N_D denotes the number of the input dimensions, N_L denotes the number of fuzzy hypercube cells, and x_i denotes the i-th input dimension. Based on the aforementioned model structure, a GHLA is presented in the next section to determine the appropriate model structure and its adjustable parameters.

3. PROPOSED HYBRID LEARNING ALGORITHM FOR THE T-FCMAC

3.1 Proposed Improved QPSO

In the traditional QPSO algorithm, the particle position is defined as follows:

$$Mbest = \frac{1}{P} \sum_{i=1}^{N\_P} Pbest_i \tag{5}$$

$$P_i = \varphi \times Pbest_i + (1 - \varphi) \times Gbest \tag{6}$$

$$x_i(t+1) = P_i \pm \beta |Mbest - x_i(t)| \times ln\left(\frac{1}{u}\right), i = 1, 2, \ldots, N\_P \tag{7}$$

where φ and u are random numbers that are uniformly distributed over the interval (0,1), P_i is a local attractor, Mbest is the mean best position of the population, and N_P denotes the number of particle. Parameter β is known as the contraction expansion coefficient. In the iteration process, the plus (+) or minus (−) sign is decided based on the random numbers. When the numbers are higher than 0.5, the "−" sign is used. Otherwise, the "+" is used.

Fuzzy C-means clustering (FCM) was proposed by Bezdek (1984). Let the input data be as $X = \{x_1, x_2, \ldots, x_{N\_P}\}$, where N_P denotes the number of data. $\mu_{C_i}(x_j)$ represents the membership degree of data x_j in the i-th cluster. If the position of a particle is near to the center of a cluster, it has a higher membership degree. The sum of each cluster

membership degree should be exactly equal to one as presented below:

$$\sum_{i=1}^{N_C} \mu_{c_i}(x_j) = 1 \tag{8}$$

where $\mu_{c_i}(x_j) \in [0,1]$, $\mu_{C_i}(x_j)$ represents the membership degree of the fuzzy subsets, and N_C denotes the number of clusters.

The mean best value was used to evaluate the Mbest position, thus making the QPSO algorithm more efficient than the traditional PSO algorithm (Eberhart and Kennedy, 1995). The Mbest position is obtained by simply averaging the personal best positions of all particles. That is, it is considered that all particles are equal and that all particles exert the same influence on the Mbest value. The primary idea of this method is that the mean best position determines the search scope or creativity of the particle (Sun et al., 2004). In the traditional Mbest computational method, the positions of all particles are used to obtain an average position. In this study, the center position is obtained using the fuzzy C-mean clustering method instead of the traditional Mbest computational method. Based on the variety of the cluster center positions used for increasing the search scope, the particle swarm quickly converges to the global optimum. The procedure for the proposed improved QPSO is as follows:

Step 1. *Initialization of swarm positions*: Initialize a population (array) of particles with random positions in an N-dimensional problem by using a uniform probability distribution function. The parameters of T-FCMAC are coded into a particle. The parameters contain the uncertainty mean m_{nj}, standard deviation σ_{nj}, and scalar value of TSK a_{nj}. The coding format is shown in Fig. 2.

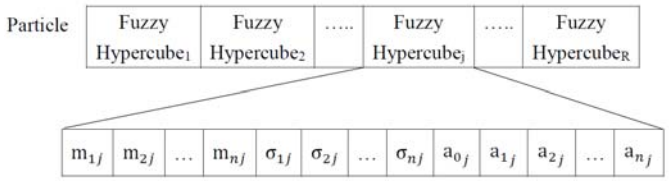


Fig. 2. Coding the adjustable parameters of a T-FCMAC into a particle.

Step 2. *Updating a membership value:* The membership degree $\mu_{c_i}(x)$ of particle each depends on the position of the particle and the cluster center:

$$\mu_{c_i}(x) = \frac{1}{\sum_{k=1}^{N_C}\left(\frac{\|x-v_i\|^2}{\|x-v_k\|^2}\right)} \tag{9}$$

where x is the position of particles, N_C denotes the number of clusters, and v_i represents the *i*-th clustering center.

Step 3. *Computing cluster center:* Equation (10) is used to update the cluster center c_i (Bezdek, Ehrlich, and Full 1984) and return to Step 3 to keep the cluster center moving until the termination condition is attained.

$$v_i = \frac{\sum_{j=1}^{N\_P}(\mu_{c_i}(x_j))^2 x_j}{\sum_{j=1}^{N\_P}(\mu_{c_i}(x_j))^2} \tag{10}$$

Step 4. *Updating the position of a particle*: New particle positions are generated as follows:

$$\begin{cases} x_j^1(t+1) = P_j \pm \beta\left|v_1 - x_j(t)\right| \times ln\left(\frac{1}{u}\right) \\ x_j^2(t+1) = P_j \pm \beta\left|v_2 - x_j(t)\right| \times ln\left(\frac{1}{u}\right) \\ \quad\vdots \\ x_j^i(t+1) = P_j \pm \beta\left|v_i - x_j(t)\right| \times ln\left(\frac{1}{u}\right) \\ \quad\vdots \\ x_j^{N_C}(t+1) = P_j \pm \beta\left|v_{N_C} - x_j(t)\right| \times ln\left(\frac{1}{u}\right) \end{cases} \tag{11}$$

where P_j denotes the local attractor, β represents the contraction expansion coefficient, v_i is used to replace the traditional *Mbest* position, and the factor u is randomly generated within the range [0,1]. Moreover, the particle swarms are generated based on the number of clusters.

Step 5. *Evaluation of a particle's fitness value*: In this step, the fitness value of each particle is evaluated. The *j*-th particle with the best fitness value in all clusters is selected using the following equation:

$$top\_particle_j = max\left(fit(x_j^1), fit(x_j^2), \ldots, fit(x_j^{N_C})\right) \tag{12}$$

Step 6. *Updating the Pbest and global best (Gbest) values*: This step compares the fitness value of each particle with the *Pbest* fitness value of each particle. If the current fitness value is better than the *Pbest* value, then the current fitness value is used to replace the *Pbest* value and the current position replaces the *Pbest* position in an *N*-dimensional space. The updating approach of the *Gbest* value also adopts the aforementioned method.

Step 7. *Repeating the evolutionary cycle*: Go back to Step 3 until a termination criterion is satisfied.

3.2 Proposed GHLA

In this subsection, an efficient GHLA, which involves a combination of the group concept, IQPSO, and the NM method, is presented. The flowchart of the GHLA is displayed in Fig. 4 and described as follows:

Step 1. *Initialization population and parameter set:* Initialize ND + 1 particles and calculate the fitness value Fit_i of the particles. ND represents the number of dimensions. Set the fitness threshold Fit_{th} and the parameters of C-mean clustering in IQPSO.

Step 2. *NM method:* The NM method finds local minimum through a sequence of elementary geometric transformations, including reflection (α), expansion (β), contraction (γ) and shrinkage (δ) should satisfy by $\alpha>0$, $\beta>1$, $0<\gamma<1$, and $0<\delta<1$. The Pbest particles are sorted based on their fitness values by using the NM method to update the position of the worst Pbest particle.

Step 3. *Group formation:* For group formation, particles are first sorted based on their performance in descending order. The first particle in the sorted swarm is the best-performing one. This particle forms the first group and functions as the 1-th group leader (which is denoted as L_1). The group leader is

used to replace the Gbest position to make a swarm of particles to have a higher number of reference points, and the new formula is as follows:

$$P_i = \varphi \times pbest_i + (1-\varphi) \times L_k \tag{13}$$

For the leader L_k of the k-th group, the particle positions ranked after L_k are checked one by one against L_k until a particle is obtained whose similarity degree is larger than a predefined threshold $\rho_{th}$with respect to L_k. The threshold ρ_{th} used for group formation is expressed as follows:

$$\rho_{th} = \omega \times \sqrt{\frac{\|P_{\text{best}} - P_{\text{worst}}\|^2}{N_D}} \tag{14}$$

where ND is the number of particle dimensions; ω is set to 0.2(ND + 1); and P_{best} and P_{worst} represent the best and worst particles in the initial swarm, respectively. The similarity degree between the leader of i-th group and the group size of the i-th group is given by the following expression:

$$\rho(L_i,\ P_{z_i}) = \sqrt{\frac{\|L_i - P_{z_i}\|^2}{N_D}},\ i = 1,\ 2,\ \ldots, g \tag{15}$$

where L_i represents the leader of i-th group, P_{z_i} represents the group size of i-th group, and g is the number of groups. The diagram of grouping is presented in Fig. 3.

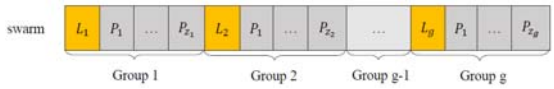


Fig. 3. Schematic of grouping.

Step 4. *Improved QPSO algorithm:* Calculate the particle position and the fitness values using IQPSO and update Pbest and Gbest.

Step 5. *Check if the termination condition is attained:* If $G > G_{th}$, then exit the process. Otherwise, go to step 6. G_{th} represents the number of generations.

Step 6. *Determine the number of clusters:* The number of clusters (N_C) in IQPSO is determined using the fitness threshold Fit_{th}. If $Fit_i < Fit_{th}$, then increase one cluster (i.e., NC + 1) and return to step 2. Otherwise, return to step 2.

3.3 Experimental Results

In this section, the examples that were used to evaluate the T-FCMAC using the GHLA are discussed. To evaluate the performance of the proposed method, two experiments were conducted—the prediction of chaotic time series and the control of a water bath temperature system. In this study, the fitness function is defined as follows:

$$Fit = \frac{1}{1+\sqrt{\frac{1}{N_t}\sum_{k=1}^{N_t}\left(y_k - y_k^d\right)^2}} = \frac{1}{1+RMSE} \tag{16}$$

where RMSE denotes the root-mean-square error, N_t represents the number of the training data, and y_k and y_k^d represent the model output and the desired output of the k-th data item, respectively. All the programs were developed in C++ using Visual Studio 2013, and each problem was simulated on a Core 2 Quad 2.83 GHz desktop computer.

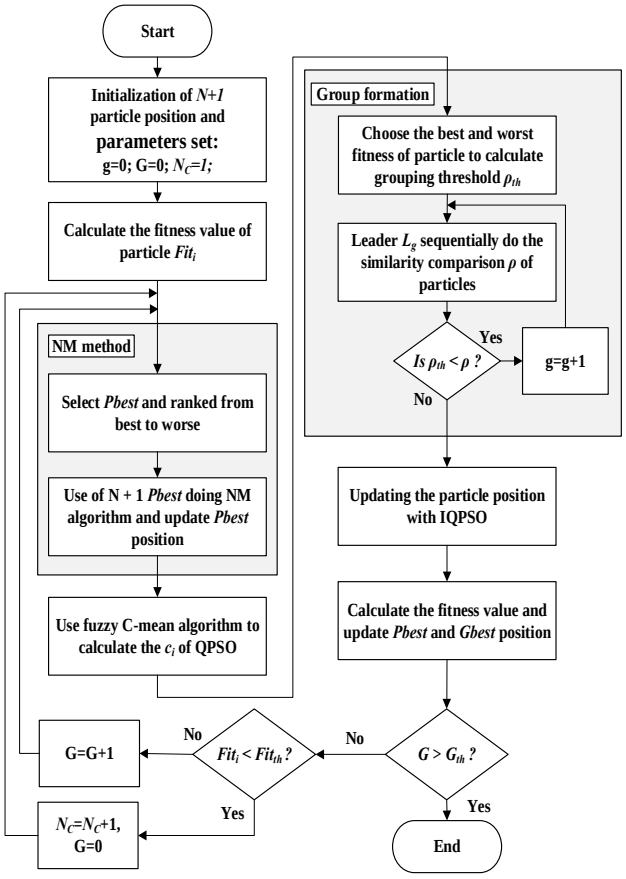


Fig. 4. Flowchart of the proposed GHLA.

Example 1: Prediction of the Chaotic Time Series

The Mackey-Glass equation was used to the generate time series data. This chaotic time series is a nonlinear, delay-differential equation whose dynamics exhibit a chaotic behavior. The Mackey-Glass chaotic time series $x(t)$ in consideration here is generated from the following delay-differential equation:

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{17}$$

where $\tau = 17$ and $x(0) = 1.2$. Crowder (Cowder, 1990) extracted 1000 input–output data pairs $\{x, y_d\}$ that consist of four past values of $x(t)$ as follows:

$$[x(t-18), x(t-12), x(t-6), x(t); x(t+6)] \tag{18}$$

There are four inputs to the T-FCMAC that correspond to the values of x(t), and there is one output representing the value $x(t+\Delta t)$, where Δt is a time prediction into the future. The first 500 pairs, that is, from $x(1)$ to $x(500)$, form the training dataset. The remaining 500 pairs, that is, from $x(501)$ to $x(1000)$, form the testing dataset used for validating the GHLA.

Table 1. Parameter values of the proposed method.

Parameter	Number of generation G_{th}	Number of particle N_P	Number of fuzzy hypercube	Fitness threshold Fit_{th}	NM-parameter $\alpha, \beta, \gamma, \delta$
Value	1000	40	3	0.99	1, 2, 0.5, 0.5

Table 2. Fitness value generated using different numbers of clusters.

Cluster number	Cluster center	Fitness value
1	c_1	0.978
2	c_1	0.968
	c_2	0.971
3	c_1	0.977
	c_2	0.993
	c_3	0.985

In this experiment, the parameter values in GHLA were set as presented in Table 1. The number of clusters was set to one for initialization and for determining whether the fitness value satisfies the specified threshold Fit_{th}. Finally, the group number was set to three; the results are presented in Table 2. The experimental results and errors of the PSO algorithm, QPSO algorithm, and proposed method are displayed in Fig. 5. By analyzing the figure, it can be noted that the proposed method has better forecasting results and lesser errors than the PSO and QPSO algorithms. Fig. 6 displays the learning curves of the PSO algorithm (Eberhart and Kennedy, 1995), QPSO algorithm (Sun et al., 2004), WQPSO algorithm (Xi et al., 2008), NM method (Nelder and Mead, 1965), and the proposed method. Table 3 presents the RMSE values obtained when the PSO algorithm (Eberhart and Kennedy, 1995), QPSO algorithm (Sun et al., 2004), WQPSO algorithm (Xi et al., 2008), NM method (Nelder and Mead, 1965), and the proposed method were used.

Table 3. Comparison results of various PSO methods.

Method	PSO	QPSO	WQPSO	NM
RMSE	211.6×10-4	98.27×10-4	70.29×10-4	105.8×10-4
Method	Improved QPSO	Group-based NM	Group-based improved QPSO	Proposed method
RMSE	61.74×10-4	29.77×10-4	11.89×10-4	8.454×10-4

Table 4 presents the RMSE value obtained when the model proposed by Kim and Kim (Kim and Kim, 1997), the adaptive neuro-fuzzy inference system (ANFIS; Jang, 1993), GEFREX (Russo, 2000), and the proposed method were used. The experimental results reveal that the proposed method is better than the other methods in terms of RMSE.

Table 4. Comparison results of various models.

Method	Kim and Kim	ANFIS	GEFREX	Proposed method
RMSE	260×10-4	70×10-4	61×10-4	8.454×10-4

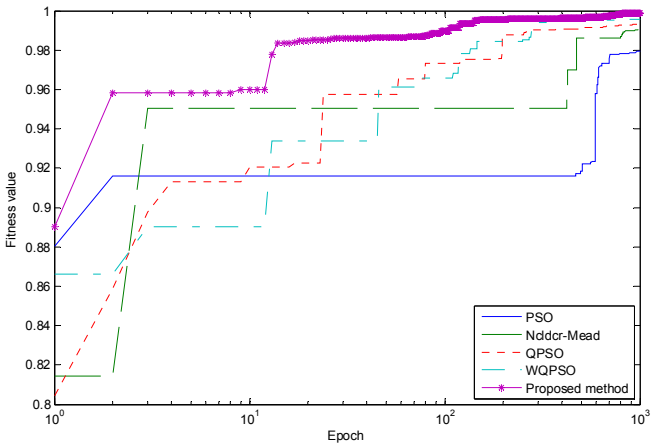
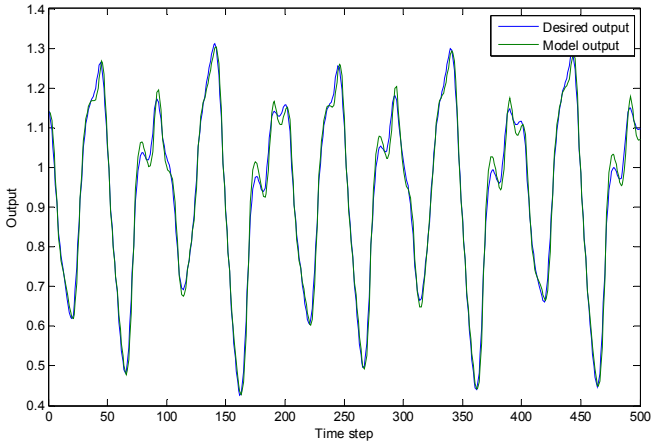
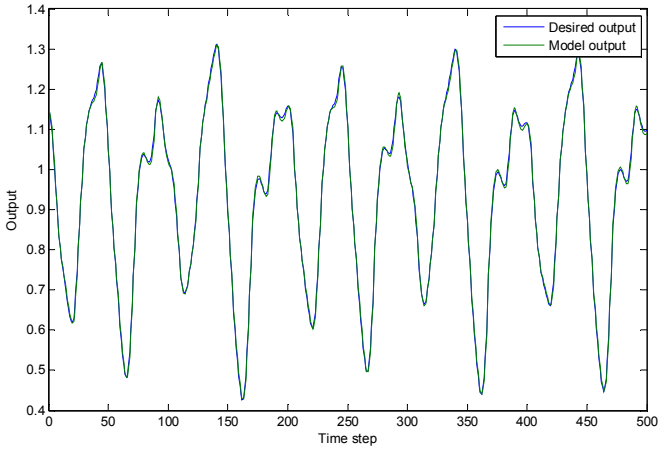


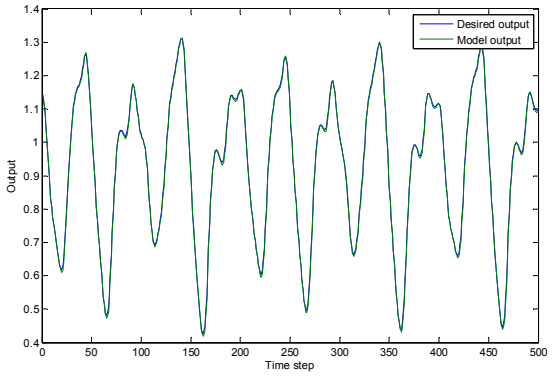
Fig. 5. Learning curves of the proposed method, PSO algorithm, NM method, QPSO algorithm, and WQPSO algorithm.



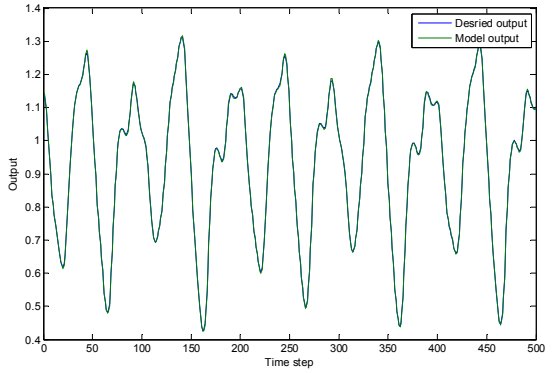
(a)



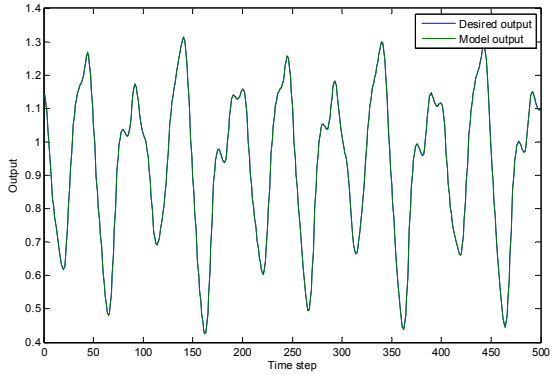
(b)



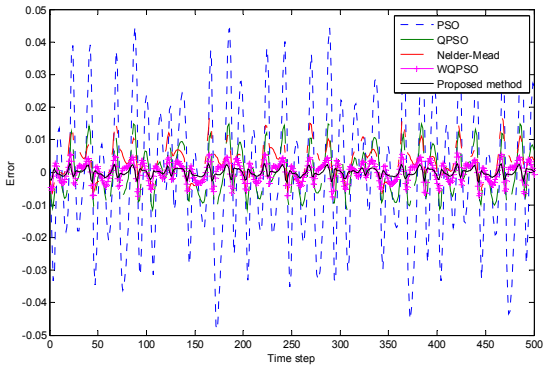
(c)



(d)



(e)



(f)

Fig. 6. Forecasting results using the (a) PSO algorithm, (b) QPSO algorithm, (c) NM method, (d) WQPSO algorithm, and (e) the proposed method. (f) Forecasting errors using different learning algorithms.

Example 2: Control of the Water Bath Temperature System

In this example, the temperature control of a water bath system (Tanomaru and Omatu, 1992) is given as follows:

$$\frac{dy_{(t)}}{dt} = \frac{u(t)}{C} + \frac{Y_0 - y(t)}{RC} \tag{21}$$

where t denotes the continuous time, $y(t)$ is the system output temperature in $°C$, $u(t)$ is Jowing heating in the system, Y_0 is the room temperature, C is the thermal capacity of the equivalent system in $J°C^{-1}$, and R is the equivalent thermal resistance between the system builders and surroundings in K/W. By assuming that R and C are essentially constant, the system in Eq. (21) can be rewritten in a discrete time form. The obtained system is as follows:

$$y(k+1) = e^{-\propto T_s} y(k) + \frac{\delta/\alpha(1-e^{-\alpha T_s})}{1+e^{0.5y(k)-40}} y(k) + [1-e^{-\alpha T_s}]y_0 \tag{22}$$

where α and δ are constant values describing R and C, respectively. The system parameters used in this example are $\alpha = 1.0015e^{-4}$, $\delta = 8.67973e^{-3}$, and $y_0 = 25.0(°C)$, which were obtained from a real water bath plant (Tanomaru and Omatu, 1992). The plant input *u(t)* is limited between 0 and 5 V. The sampling period is $T_s = 30$. The task is to control the water bath system to follow three set points:

$$y_{ref} = \begin{cases} 35°C, \ for \ k \ \leq 40 \\ 55°C, \ for \ 40 < k \leq 80 \\ 75°C, \ for \ 80 < k \leq 120 \end{cases} \tag{23}$$

Table 5. Parameter values of the proposed method.

Parameter	Number of generation G_{th}	Number of particle N_P	Number of fuzzy hypercube	Fitness threshold Fit_{th}	NM-parameter $\alpha, \beta, \gamma, \delta$
Value	1000	36	5	0.65	1, 2, 0.5, 0.5

Table 6. Fitness value generated using different numbers of cluster centers.

Cluster number	Cluster center	Fitness value
1	c_1	0.637
2	c_1	0.626
	c_2	0.653

In this experiment, the parameter values of the proposed method are set and presented in Table 5. The number of clusters is set to one for initialization and for determining whether the fitness value satisfies the specified threshold Fit_{th}. Finally, the group number is defined to be three, and the results are presented in Table 6. The proposed method was compared with other aforementioned methods. The experimental results and errors of the PSO algorithm, QPSO

algorithm, and proposed method are presented in Fig. 7. As presented in this figure, the proposed method has better forecasting results and errors than the PSO and the QPSO. After learning, Fig. 8 displays the learning curves of the PSO algorithm (Eberhart and Kennedy, 1995), QPSO algorithm (Sun et al., 2004), WQPSO algorithm (Xi et al., 2008), NM method (Nelder and Mead, 1965), and the proposed method. Table 7 presents the RMSE values obtained using the PSO algorithm, QPSO algorithm, WQPSO algorithm, NM method, and the proposed method. Table 8 presents the RMSE values obtained using a GA-NFS controller (Lin, 2004), PID controller (Anderson, 1987), fuzzy controller (Juang and Lin, 1998), ANFIS controller (Jang, 1993), and the proposed method. The experimental results reveal that the proposed method is better than other methods in terms of RMSE.

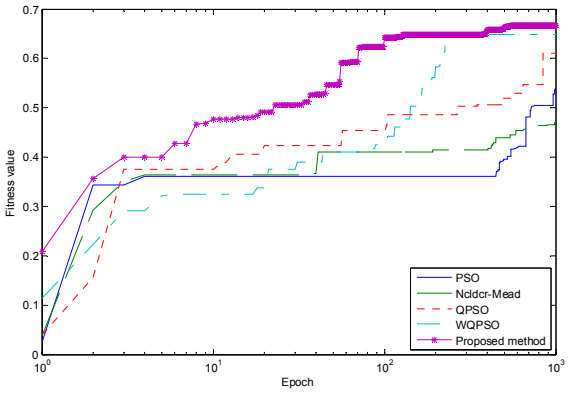


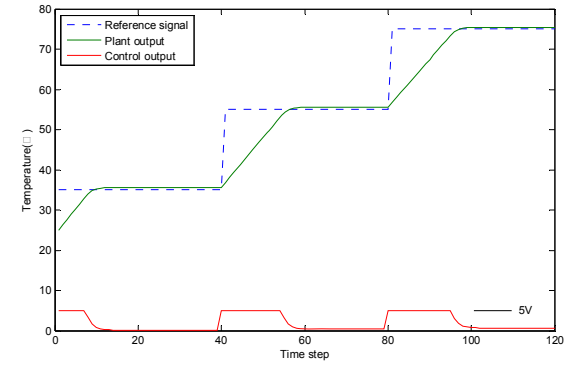
Fig. 7. Learning curves of the proposed method, PSO algorithm, NM method, QPSO algorithm, and WQPSO algorithm.

Table 7. Comparison result of the proposed method and other methods.

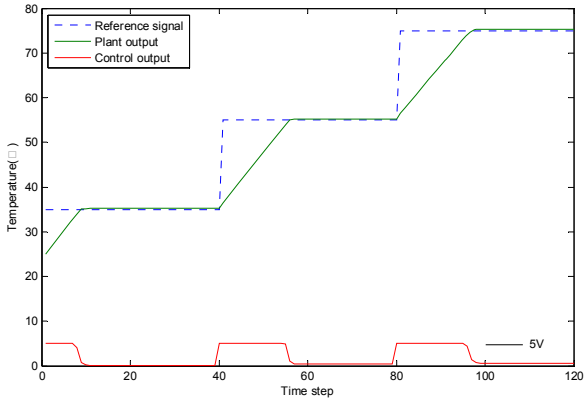
Method	PSO	QPSO	WQPSO	NM
RMSE	0.851766	0.70834	0.637786	1.12977
Method	Improved QPSO	Group-based NM	Group-based improved QPSO	Proposed method
RMSE	0.64274	0.60557	0.542865	0.49955

Table 8. Comparison results of various models.

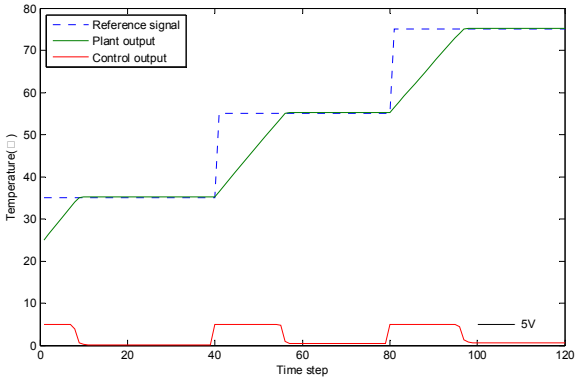
Method	GA-NFS controller	PID Controller	Fuzzy Controller	ANFIS controller	Proposed method
RMSE	0.51426	0.82391	0.79522	0.56338	0.49955



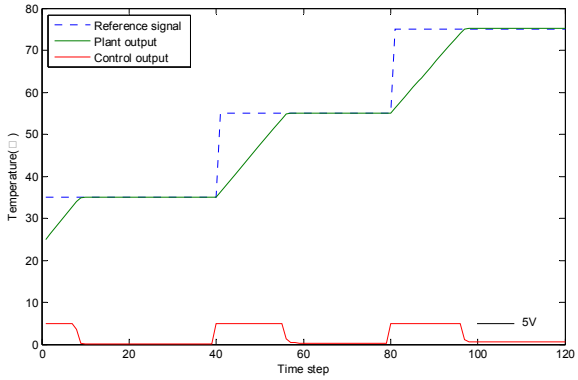
(a)



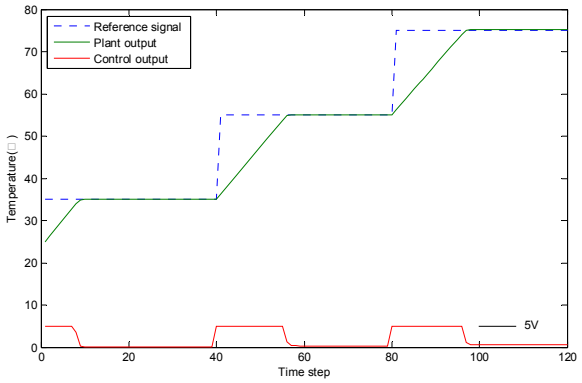
(b)



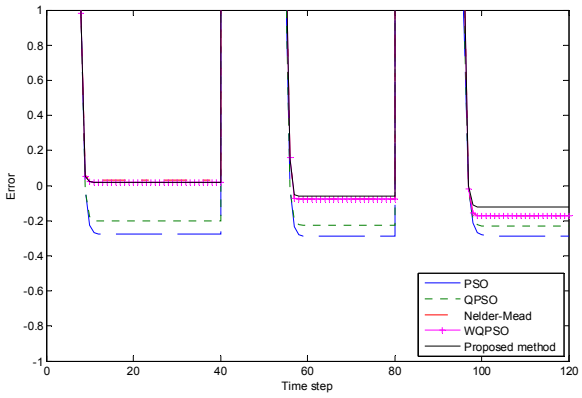
(c)



(d)



(e)

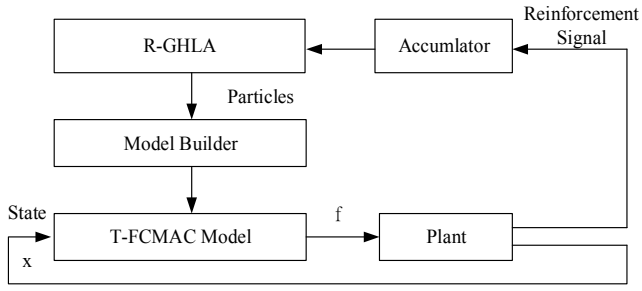


(f)

Fig. 8. Forecasting results using the (a) PSO algorithm, (b) QPSO algorithm, (c) NM method, (d) WQPSO algorithm, and (e) proposed method. (f) The forecasting errors using different learning algorithms.

4. R-GHLA FOR T-FCMAC

Unlike the supervised learning problem, in which the correct "target" output values are given for each input pattern, the reinforcement learning problem has very simple "evaluative" or "critical" information, rather than the "instructive" information, available for learning. In an extreme case, there is only a single bit of information to indicate whether the output is right or wrong. Fig. 9 displays the proposed method using the reinforcement evolutionary learning system. The training environment of the controller interacts with reinforcement learning problems. In this section, the reinforcement signal indicates whether a success or a failure has occurred.



Figsd. 9. Schematic of R-GHLA for the T-FCMAC.

An accumulator is used for relative performance evaluation, as shown in Fig. 9, and accumulates the number of time steps before a failure occurs. In this section, the feedback acts as an accumulator to determine from how long the experiment is a "success." The obtained result is used for the fitness evaluation of the R-GHLA. That is, the accumulator indicates the "fitness" of the current T–FCAMC.

An accumulator is used to evaluate the fitness function that determines from how long the experiment is a "success." Thus, based on a defined fitness function, a fitness value is assigned to each string in the population where a higher fitness value implies a better fit. In this section, a number of time steps were used before failure occurs to define the fitness function. The goal is to maximize the fitness value.

The fitness function is defined as follows:

$$\text{Fitness Value (i)} = \text{TIME–STEP(i)} \tag{21}$$

where *TIME–STEP*(*i*) represents how long the experiment is a "success" with the *i*-th population. Equation (21) suggests that long-time steps before failure occurs (to keep the desired control goal longer) imply a higher fitness value.

The experiment was conducted using a ball and beam system (Hauser et al., 1992), as shown in Fig. 10. The beam was rotated in a vertical plane when a torque was applied at the center of rotation. The ball was free to roll along the beam; however, the ball should remain in contact with the beam.

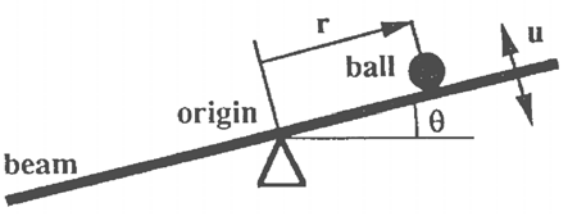


Fig. 10. Ball and beam system.

The ball and beam system can be written in the state space form as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ B(x_1 x_4^2 - G\sin x_3) \\ x_4 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u, y = x_1 \tag{26}$$

where $x = (x_1, x_2, x_3, x_4)^T \equiv (r, \dot{r}, \theta, \dot{\theta})^T$, x represents the state of the system, x_1 is the position of the ball, u is the angular acceleration ($\ddot{\theta}$), B = 0.7143 is a constant and G = 9.81 m/s^2 is the acceleration of gravity. Moreover, $y = x_1 \equiv r$ and y represents the output of the system. The purpose of control u is such that the closed-loop system output y will converge to zero from different initial conditions.

Based on the input–output linearization algorithm, the control law u is determined as follows:

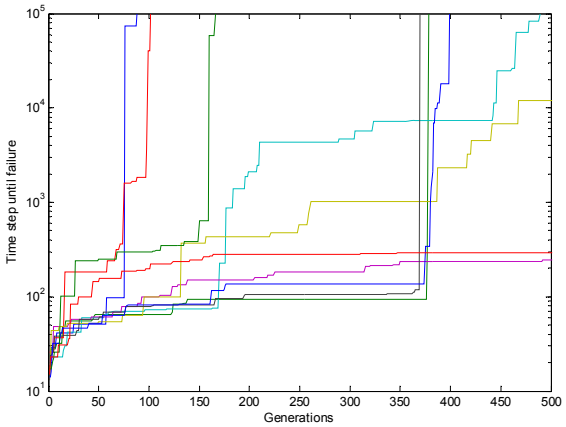
For state x, compute $v = -\alpha_3\varphi_4(x) - \alpha_2\varphi_3(x) - \alpha_1\varphi_2(x) - \alpha_0\varphi_1(x)$, where $\varphi_1(x) = x_1$, $\varphi_2(x) = x_2$, $\varphi_3(x) = -BG\sin x_3$, $\varphi_4(x) = -BGx_4\cos x_3$, and α_i are selected so that $s^4 + \alpha_3 s^3 + \alpha_2 s^2 + \alpha_1 s + \alpha_0$ is a Hurwitz

polynomial. Compute $a(x) = -BG\cos x_3$ and $b(x) = BGx_4^2 \sin x_3$. Then, $u = [v - b(x)]/a(x)$.

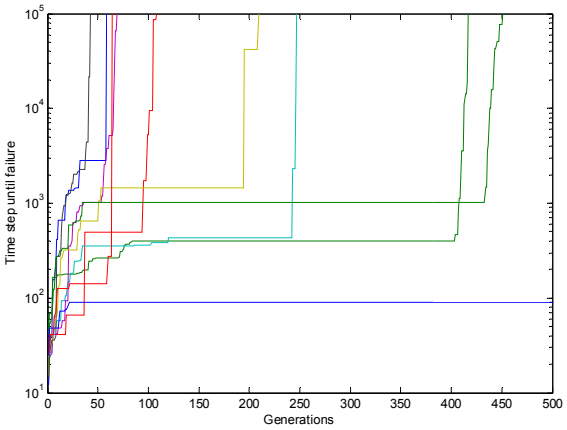
The four input variables $(r, \dot{r}, \theta, \dot{\theta})$ and the output u are normalized between 0 and 1 over the following ranges: $r = [-5,5], \dot{r} = [-3,3], \theta = [-1,1], \dot{\theta} = [-2,2],\ and\ u = [-70,70]$. In this experiment, the initial values were set to (0, 0, 0, 0). A total number of ten runs was performed. Each run was initiated at the same initial state. In this experiment, the initial state was set to $r = -1.2, \dot{r} = -0.01, \theta = 0.58, \dot{\theta} = 0.58$ Fig. 11 presents that the proposed T-FCMAC controller balances the ball using various algorithms based on evolutionary reinforcement learning. Ten runs were conducted for each algorithm, and the termination condition of each run was to maintain a balanced the system in 10^5 time steps or to satisfy the number of generations. Table 9 presents the initial parameters during the learning process. The performance evaluation includes the average number of generations, the best number of generations, and the worst number of generations in successful runs. Experimental results are presented in Table 10. Fig. 11(a) and (b) illustrate the learning curves of ten runs obtained using the PSO and QPSO algorithms. During the learning process, the control results had three and one failures to balance the ball and beam system as illustrated in Fig. 11(a) and Fig. 11 (b), respectively. Fig. 11 (c) indicates that all ten runs were able to balance the ball and beam system, but the WQPSO algorithm required more generations to converge. Fig. 11 (d) demonstrates that the proposed method is superior to other algorithms in terms of the convergence rate, and the average number of generations was 37. Fig. 12 shows the position deviation of the ball when the system was controlled using various methods. The R-GHLA can effectively keep the ball by decreasing the deviation in position and can then reach the balance of convergence the position to zero.

Table 9. Initial parameters during the learning process.

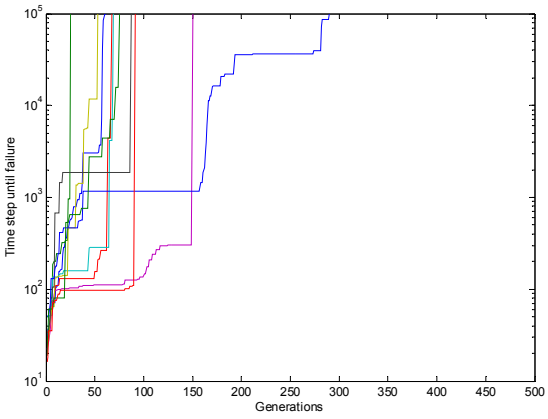
Parameter	Number of Generation G_{th}	Number of Particle	Number of Hypercube	NM–parameter $\alpha, \beta, \gamma, \delta$
Value	1000	21	5	1, 2, 0.5, 0.5



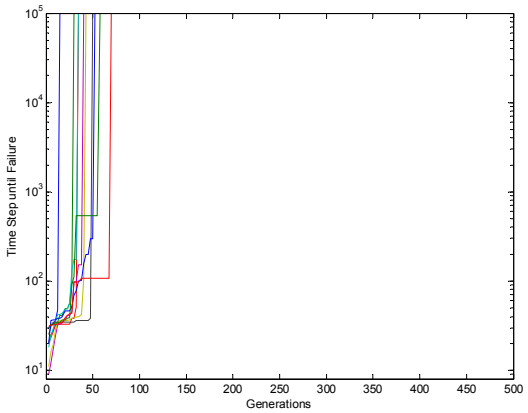
(a)



(b)



(c)



(d)

Fig. 11. Performance of the (a) PSO algorithm, (b) QPSO algorithm, (c) WQPSO algorithm, and (d) proposed method on the ball and beam system.

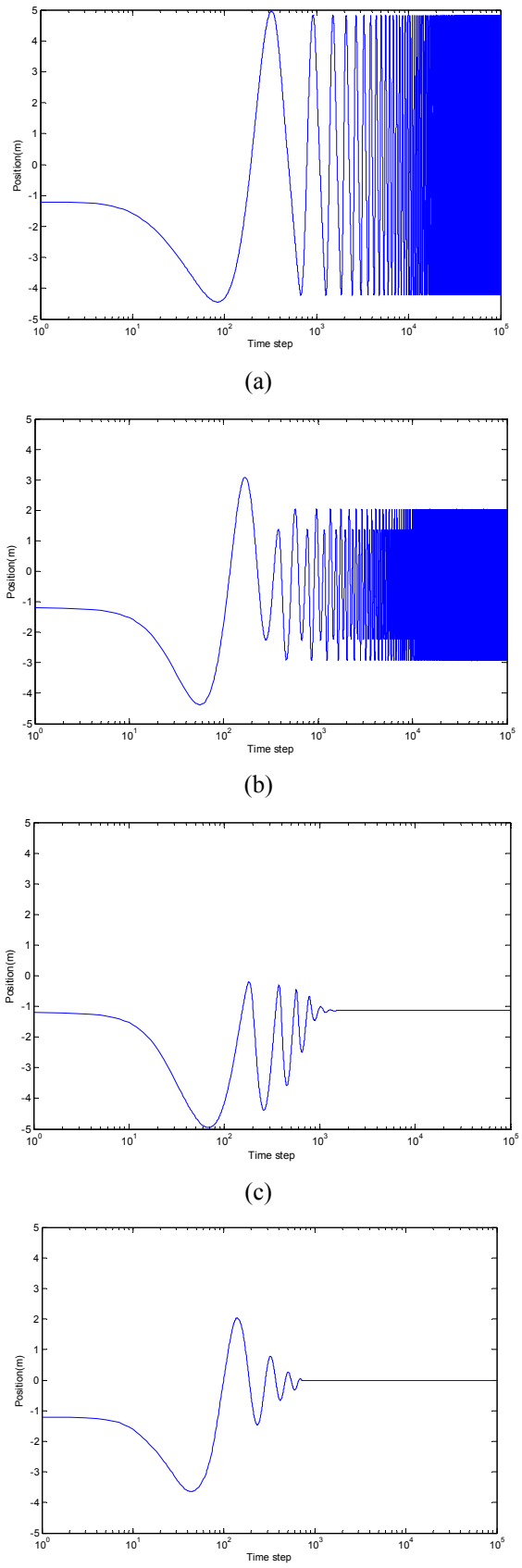


Fig. 12. Position deviation of the ball obtained using trained (a) PSO algorithm, (b) QPSO algorithm, (c) WQPSO algorithm, and (d) proposed method.

The CPU training times are also considered to evaluate the proposed method. The performance evaluation includes the average CPU time, the best CPU time, and the worst CPU time. Table 11 presents that the proposed method has lower CPU times than those of other algorithms.

Table 10. Comparison between the number of generations obtained using different methods and the proposed method.

Method	Mean	Best	Worst
PSO	297	88	898
QPSO	151	44	546
WQPSO	98	25	290
TDGAR	210	30	324
CQGAF	187	23	298
Proposed method	37	16	76

Table 11. Comparison between the CPU time of different methods and the proposed method.

Method	Mean	Best	Worst
PSO	137.98	85.75	288.92
QPSO	70.74	29.37	175.63
WQPSO	49.59	18.14	94.34
TDGAR	39.58	11.35	75.76
CQGAF	32.51	8.27	62.21
Proposed method	19.13	8.72	39.16

5. CONCLUSIONS

A T-FCMAC was proposed in this study for solving prediction and control problems. Both the supervised GHLA and R-GHLA were proposed to adjust the parameters of the T-FCMAC. The proposed GHLA is based on an IQPSO algorithm and the NM method. The advantages of the proposed GHLA and R-GHLA not only improve the search ability of the algorithm but also can quickly find global optima. The experimental results demonstrated that the convergence rate and the RMSE of the proposed method are better than those of other methods due to the GHLA and R-GHLA.

To obtain high speed operation in a real-time application, GHLA and R-GHLA controllers will be implemented on field programmable gate arrays in a future work.

REFERENCES

Albus, J. S. (1975). A new approach to manipulator control: the cerebellar model articulation controller (CMAC). J. Dyn. Sys., Meas., page 220-227.

Anderson, C.W. (1987). Strategy learning with multilayer connectionist representations. Proc. 4th Int. Workshop on Machine Learning, page 103-114.

Bezdek, J., Ehrlich, R., and Full, W. (1984). FCM: the fuzzy C-means clustering algorithm. Computers & Geosciences, volume 10, page 91-203.

Chow, M.Y. and Menozzi, A. (1994). A self-organized CMAC controller. Proceedings of 1994 IEEE International Conference on Industrial Technology, page 68-72.

Cowder, R. S. (1990). Predicting the Mackey-glass time series with cascade-correlation learning. Proceedings of the 1990 Connectionist Models Summer School, page 117-123.

Eberhart, R. and Kennedy, J. (1995). Particle swarm optimization. IEEE International Conference on Neural Networks, volume 4, page 1942-1948.

Hauser, J., Sastry, S., and Kokotovic, P. (1992). Nonolinear control via approximate input–output linearization: the ball and beam example. IEEE Trans. Automatic Control, volume 37, page 392-398.

Hsieh, S. T., Sun, T. Y., Liu, C. C., and Tsai, S. J. (2009). Efficient population utilization strategy for particle swarm optimizer. IEEE Trans Syst Man Cybernet Part B Cybernet, volume 39, page 56-444.

Ilonen, J., Kamarainen, J. K., and Lampinen, J. (2003). Differential evolution training algorithm for feed-forward neural networks. Neural Process. Lett., volume 7, page 93-105.

Jang, J.S.R. (1993). ANFIS: adaptive-network-based fuzzy inference system. IEEE Trans. Syst. Man Cyber. volume 23, page 665–685.

John, D., Elad, H., and Yoram, S. (2011). Adaptive subgradient methods for online learning and stochastic optimization. The Journal of Machine Learning Research, volume 12, page 2121-2159.

Juang, C.F. and Lin, C.T. (1998). An on-line self-constructing neural fuzzy inference network and its applications. IEEE Trans, Fuzzy System, volume 6, page 12-32.

Juang, C. F. (2005). Combination of online clustering and Q-value based GA for reinforcement fuzzy system design. IEEE Trans. Fuzzy Syst., volume 13, page 289-302.

Kaur, A. and Kaur. (2012). Comparison of Mamdani-type and Sugeno-type fuzzy inference systems for air conditioning system. International Journal of Soft Computing and Engineering, volume 2, page 323-325.

Kim, D. and Kim, C. (1997). Forecasting time series with genetic fuzzy predictor ensemble. IEEE Trans. Fuzzy Syst., volume 5, page 523-535.

Léon, B. (1998). Online algorithms and stochastic approximations. Online Learning and Neural Networks, Cambridge University Press, page 9-42.Lin, C. J. (2004). A GA-based neural fuzzy system for temperature control. IEEE Trans. Fuzzy Syst., volume 134, page 311–333.

pLin, C.M. and Li, H.Y. (2015). Dynamic petri fuzzy cerebellar model articulation control system design for magnetic levitation system. IEEE Trans. Contr. Syst. Technol., volume 23, page 693-699.

Lin, C. T. and Jou, C. P. (2000). GA-based fuzzy reinforcement learning for control of a magnetic bearing system. IEEE Trans. Syst., Man, Cybern. B, Cybern., volume 30, page 276-289.

Lin, H.Y., Lin, C.J., and Huang, M.L. (2016). Optimization of printed circuit board component placement using an efficient hybrid genetic algorithm. Applied Intelligence, volume 45, page 622-637.

Li, X. (2010). Niching without niching parameters: particle swarm optimization using a ring topology. IEEE Trans Evol Comput, volume 14, page 69-150.

Li, X. and Yao, X. (2012). Cooperatively coevolving particle swarms for large scale optimization. IEEE Trans Evol Comput, volume 16, page 24-210.

Nelder, J. A. and Mead, R. (1965). A simplex method for function minimization. The Computer Journal, volume 7, page 308-313.

Reay, D. (1995). Nonlinear channel equalization using associative memory neural networks. Proceedings of the International Workshop on Applied Neural Networks Telecom, page 17-24.

Russo, M. (2000). Genetic fuzzy learning. IEEE Trans. Evol. Comput. volume 4, page 259-273.

Sun, J., Feng, B., and Xu, W.B. (2004). Particle swarm optimization with particles having quantum behaviour. IEEE Proceedings of Congress on Evolutionary Computation, page 325-331.

Sun, J., Xu, W. B., and Feng, B. (2004). A global search strategy of quantum-behaved particle swarm optimization. Proceedings of the 2004 IEEE Conference on Cybernetics and Intelligent Systems, page 111-116.

Tanomaru, J. and Omatu, S. (1992). Process control by on-line trained neural controllers. IEEE Trans. Ind. Electron, volume 39, page 511-521.

Wang, J.G., Tai, S.C., and Lin, C.J. (2016). Transmission map estimation of weather degraded images using a hybrid of recurrent fuzzy CMAC and weighted strategy. Optical Engineering, volume 55, page 083104-1~083104-14.

Xi, M., Sun, J., and Xu, W. (2008). An improved quantum-behaved particle swarm optimization algorithm with weighted mean. Applied Mathematics and Computation, volume 205, page 751-759.

Zhang K. and Qian, F. (2000). Fuzzy CMAC and its application. Proc. 3rd World Congr. Intell. Control and Autom., page 944-947.