frontiers in Neuroscience

ORIGINAL RESEARCH
published: 29 May 2019
doi: 10.3389/fnins.2019.00390

Robust Adaptive Recurrent Cerebellar Model Neural Network for Non-linear System Based on GPSO

Jian-sheng Guan[1,2]*, *Shao-jiang Hong*[1], *Shao-bo Kang*[1], *Yong Zeng*[2], *Yuan Sun*[1] and *Chih-Min Lin*[3]*

[1] College of Electrical Engineering and Automation, Xiamen University of Technology, Xiamen, China, [2] Princess Margaret Cancer Centre, University Health Network, Toronto, ON, Canada, [3] Department of Electrical Engineering, Innovation Center for Big Data and Digital Convergence, Yuan Ze University, Taoyuan, Taiwan

A robust adaptive recurrent cerebellar model articulation controller (RARC) neural network for non-linear systems using the genetic particle swarm optimization (GPSO) algorithm is presented in this study. The RARC is used as the principal tracking controller and the robust compensation controller is designed to recover the residual of the approximation error. In the RARC neural network, the steepest descent gradient method and the Lyapunov function are used for deriving the adaptive law parameter of the system. Besides, the learning rates play an important role in these adaptive laws and they have a great effect on the functions of control systems. In this paper, the combination of the genetic algorithm with the mutation particle swarm optimization algorithm is applied to seek for the optimal learning rates of the RARC adaptation laws. The numerical simulations about the inverted pendulum system as well as the robot manipulator system are given to confirm the effectiveness and practicability of the GPSO-RARC-based control system. Compared with other control schemes, the proposed control scheme is testified to be reliable and can obtain the optimal parameter about the learning rates and the minimum root mean square error for non-linear systems.

Keywords: RARC neural network, GPSO algorithm, learning rate, robot manipulator system, non-linear systems

OPEN ACCESS

Edited by:
Hak Keung Lam,
King's College London,
United Kingdom

Reviewed by:
Bo Xiao,
Imperial College London,
United Kingdom
Aiwen Meng,
Yanshan University, China

***Correspondence:**
Jian-sheng Guan
guanjiansheng@gmail.com
Chih-Min Lin
cml@saturn.yzu.edu.tw

Specialty section:
This article was submitted to Autonomic Neuroscience, a section of the journal Frontiers in Neuroscience

Received: *02 January 2019*
Accepted: *04 April 2019*
Published: *29 May 2019*

Citation:
Guan J, Hong S, Kang S, Zeng Y, Sun Y and Lin C-M (2019) Robust Adaptive Recurrent Cerebellar Model Neural Network for Non-linear System Based on GPSO. Front. Neurosci. 13:390. doi: 10.3389/fnins.2019.00390

INTRODUCTION

Strictly speaking, almost all practical control systems are non-linear systems and there is a difference between the mathematical model and the practical system. Besides, the structure and parameters of the practical systems are generally unknown or time-varying and the disturbances acting on the system are often random and unmeasurable in many cases. The neural network has the advantages of highly parallel structure, powerful learning ability, continuous non-linear function approximation ability, fault tolerance, etc., which greatly promotes and expands the application of neural network technology in non-linear system identification and control (Hunt et al., 1992). Recently, scholars have proposed generous research papers on neural network control theory and engineering application. For a class of uncertain non-linear systems with strict feedback, an adaptive neural network controller was designed using dynamic surface control technique (Wang and Huang, 2005). For a class of uncertain Multiple-Input Multiple-Output non-linear systems with unknown control coefficient matrix and input non-linearity, a variable structure control method combining an adaptive neural network controller with backtracking and Lyapunov synthesis is proposed (Chen et al., 2010). The paper presented a control scheme for the

non-linear systems with input and state delay, which merges a radial basis function neural network, backstepping, and adaptive control (Zhu et al., 2008). As for a class of second-order non-linear systems, a wavelet adaptive backstepping control system was designed, which consists of a neural backstepping controller and a robust controller (Hsu et al., 2006).

In 1975, Albus proposed the concept of the cerebellar model articulation controller (CMAC) for the first time (Albus, 1975), which was an imitation of the cerebellum learning structure, also one of the local approximations in the neural network system. Cerebellar model network system not only has non-linear approximation ability, adaptive generalization ability and associative memory ability, but also is a kind of fast convergence neural network, which has been widely used in non-linear real-time control system (Guan et al., 2016). An efficient controller was proposed for the robot manipulators based on the structure and local learning characteristics of CMAC (Commuri et al., 1997). Considering the characteristics of non-linear uncertainty model, the paper presented an adaptive connection controller based on the monitoring system, which was composed of a supervisory controller and adaptive CMAC (Lin and Peng, 2004). Compared with fully connected neural networks, the CMAC neural network (NN) has strong structural advantages and was an effective control method for unknown dynamics non-linear systems (Commuri and Lewis, 1995). For a class of multi-input and multi-output uncertain non-linear systems, a self-organizing CMAC control system was proposed, which combines sliding mode control, compensation control and CMAC (Lin and Chen, 2009). At the same time, a TKS fuzzy CMAC controller integrates the robust compensation and adaptive law is proposed to improve the precision of position control and speed control for robot manipulators (Guan et al., 2018).

As a kind of artificial neural network, recurrent neural network takes sequence data as input, recursion in the direction of sequence evolution and all nodes are linked in a chain to form a closed loop. Therefore, it can show dynamic time series behavior. Unlike feed forward neural networks, RNNs have the characteristics of memory and parameter sharing. This performance also makes it extremely useful for speech recognition, language modeling, machine translation and other fields (Sak et al., 2014). A complex fuzzy neural network system, which can be a modified version of the fuzzy neural networks, was used for identifying and controlling non-linear dynamic systems (Zhong et al., 2017, 2018; Lam, 2018). The recursive neuron has an internal feedback loop which can capture the dynamic response of the system and further simplify the network model (Lee and Teng, 2000). The paper combines Takagi-Sugeno-Kang fuzzy model with the wavelet neural network and constructs a recurrent wavelet fuzzy neural network to identify and predict the operation of non-linear dynamic systems (Lin and Chin, 2004). For the non-linear uncertain systems, an adaptive recurrent CMAC with sliding mode control was proposed, and the performance of the system was proved on the car-following system and the chaotic system (Lin and Chen, 2006). For the motion control of the linear ultrasonic motor, an adaptive recurrent CMAC based on variable optimal learning rate and dynamic gradient descent method was studied (Peng and Lin, 2007).

Particle swarm optimization (PSO) algorithm is an evolutionary algorithm proposed by Kennedy and Eberhart in 1995 (Eberhart and Kennedy, 1995). It is derived from the simulation of bird predation and is an evolutionary computation technology based on swarm intelligence. As a new parallel optimization evolutionary algorithm, PSO can deal with a large number of non-linear, non-differentiable, multi-peak as well as non-continuous optimization and multi-peak optimization problems, which was widely used in engineering and science fields (Kennedy, 2011). In the 1970s, professor J. H. Holland first developed the model of Genetic algorithm (GA) (Holland, 1973). It is an effective optimization method with principles about genetics natural selection. GA is also very popular in the fields of optimal scheduling, computer science, combinatorial optimization as well as transportation problem since its simple and universal, strong robustness and parallel processing (Holland, 1992). The improvement of genetic algorithm in the past is generally considered as the problem of premature and convergence. An adaptive genetic algorithm with dynamic fitness function for multi-objective problems in a dynamic environment was proposed to review the performance of the algorithm (Bingul, 2007). A new kind of genetic algorithms combined with the concept of the horizontal set was proposed to control the "precocity" of the genetic algorithm (Qinghua et al., 2006). In order to improve the convergence of the genetic algorithm, an improved crossover operation is proposed, and new population diversity and individual correlation are defined (Cai and Xia, 2006). However, the research on genetic algorithm fails to fully consider the situation of individuals in each generation, which does not match the growth and improvement of individuals in the process of evolution. In the selection and cross steps of the genetic algorithm, individuals directly enter the next generation, while individuals themselves do not get improved. Individuals have to grow and adapt to the environment in order to reproduce in nature. In the PSO algorithm, each particle is related to each other, and the particle can be imitated in the natural world, and the maximum performance can be improved and the particle can be mature (Peng et al., 2008).

There are usually two options to select the appropriate learning rate of the recurrent CMAC (learning rate of the recurrent neuron, weight, the variance and the mean), one of which is to adopt human expert experience. However, the accuracy of the method is not high, and not suitable for complex and uncertain problems. The second scheme is gradient learning (Song et al., 2008; Misra and Saha, 2010). In this paper, a robust adaptive recurrent cerebellar model neural network for non-linear system based on GPSO algorithm is investigated, in order to avoid trial-and-error and improve the local optimal problems. In this system, the optimal learning rate for controller is calculated by GPSO algorithm, the adaptive recurrent cerebellar model articulation controller is used as the principal tracking controller and the robust compensation controller is designed to recover the residual of the approximation error, and the steepest descent gradient method and the Lyapunov function are used for deriving the online adaptive law parameter, so that the system

stability can be guaranteed. Finally, the proposed GPSO-RARC-based control system is applied to the inverted pendulum system and the robot manipulator system to illustrate its effectiveness. Compared with the existing research already reported in the literature, the contribution of this paper has the following three aspects: (1) This paper combines genetic algorithm and mutation particle swarm optimization algorithm to find the optimal learning rate of adaptive law to the robust adaptive recurrent cerebellar model articulation controller and reduce the system training time; (2) The proposed control scheme ensures the stability of the entire system; (3) The compensation control can eliminate the small disturbance, when there are uncertainty, the compensation control deals with the lumped uncertainty. The full text is structured as follows. After a basic introduction, the formulation of the non-linear control system is shown in section Problem Formulation. In section GPSO-RARC, a GPSO-RARC control system is developed. Section Simulation Results provides the simulation results about the manipulator system and the inverted pendulum system. Finally, in Section Conclusion some valuable conclusions are drawn from the results.

PROBLEM FORMULATION

The nth order non-linear system can be denoted as:

$$\begin{cases} \boldsymbol{x}^{(n)}(t) = f(\underline{\boldsymbol{x}}(t)) + g(\underline{\boldsymbol{x}}(t))u(t) + \boldsymbol{d}(t) \\ y = \boldsymbol{x}(t) \end{cases} \tag{1}$$

or, equivalent to formulas

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \vdots \\ \dot{x}_n = f(x_1, x_2, \cdots, x_n) + g(x_1, x_2, \cdots, x_n)u(t) + d(t) \\ y = x_1 \end{cases} \tag{2}$$

in which $f(\underline{\boldsymbol{x}}(t)) \in \Re^m$ and $g(\underline{\boldsymbol{x}}(t)) \in \Re^{m\times m}$ represent smooth non-linear uncertain functions, which are assumed to be bounded, but functions that are assumed to be bounded, and assume $g(\underline{\boldsymbol{x}}(t))$ is invertible; $u(t) = [u_1(t), u_2(t), \cdots, u_m(t)]^T \in \Re^m$ and $x(t) = [x_1(t), x_2(t), \cdots, x_m(t)]^T \in \Re^m$ are the inputs and outputs of the control, respectively; $\underline{\boldsymbol{x}}(t) = \left[\boldsymbol{x}^T(t), \dot{\boldsymbol{x}}^T(t), \cdots, \boldsymbol{x}^{(n-1)^T}(t)\right]^T \in \Re^{mn}$ is a state vector of the system and is assumed to be measurable, and $\boldsymbol{d}(t) = [d_1(t), d_2(t), \cdots, d_m(t)]^T \in \Re^m$ is the unknown external disturbance but bounded ($|d(t)| \le D$).

The purpose of the control system is to design a controller so that the state $\boldsymbol{x}(t)$ can track a given reference value $\boldsymbol{x}_d(t)$. The tracking error was denoted as $\boldsymbol{e}(t) \underline{\underline{\Delta}} \boldsymbol{x}_d(t) - \boldsymbol{x}(t) \in \Re^m$, and the tracking error vector of the control system is defined as:

$$E \underline{\underline{\Delta}} \left[\boldsymbol{e}^T(t), \dot{\boldsymbol{e}}^T(t), \cdots, \boldsymbol{e}^{(n-1)^T}(t)\right]^T \in \Re^{mn} \tag{3}$$

If the dynamics and external disturbances of the controlled object are known (i.e., the nominal functions of $f(\underline{\boldsymbol{x}}(t))$, $g(\underline{\boldsymbol{x}}(t))$ and $\boldsymbol{d}(t)$ are known exactly), the so-called feedback linearization method can be used for the control problem. In this way, an ideal controller can be developed as:

$$u^* = \frac{1}{g(\underline{\boldsymbol{x}}(t))}\left[-f(\underline{\boldsymbol{x}}(t)) - \boldsymbol{d}(t) + \boldsymbol{x}_d{}^{(n)} + \underline{\boldsymbol{K}}^T E\right] \tag{4}$$

in which $\underline{\boldsymbol{K}} = [\boldsymbol{K}_n, \cdots, \boldsymbol{K}_2, \boldsymbol{K}_1]^T \in \Re^{mn\times m}$ is the feedback gain matrix, where $\boldsymbol{K}_i = diag(k_{i1}, k_{i2}, \cdots, k_{im}) \in \Re^{m\times m}$ ($i = 1, 2, \cdots, n$) are non-zero positive constants diagonal matrix. The following error dynamics are derived by applying the control law (4) to the system (1). Suppose $\underline{\boldsymbol{K}}$ is chosen so that all roots of the polynomial $h(s) \underline{\underline{\Delta}} s^n + k_1 s^{n-1} + \cdots + k_n$ are strictly in the open left half of the complex plane. This means that for any starting initial conditions, the trace of the reference trajectory is asymptotically achieved at $\lim_{t\to\infty} |E| = 0$.

Substituting (4) into (1) the error dynamic equation is developed as:

$$\boldsymbol{e}^{(n)} + \boldsymbol{K}_1 \boldsymbol{e}^{(n-1)} + \cdots + \boldsymbol{K}_n \boldsymbol{e} = \boldsymbol{e}^{(n)} + \underline{\boldsymbol{K}}^T E = 0 \tag{5}$$

However, the non-linear functions $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ are usually unknown and external disturbances are unknown and uncertain. In this case, the control law (4) cannot be implemented in the practical applications. In order to make the system output $\boldsymbol{x}(t)$ effectively follow the given reference track $\boldsymbol{x}_d(t)$, a GPSO-RARC control system is developed to achieve a better control performance in the following sections.

GPSO-RARC

The structure of GPSO-RARC control system consists of a sliding surface, a robust adaptive recurrent CMAC where its learning rates can be updated using the GPSO algorithm and a robust compensation controller. **Figure 1** shows the block diagram of the RARC feedback control system.

The Recurrent CMAC Model

Figure 2 Shows an RCMAC model, in which T denotes a delay time. The architecture of the RCMAC includes the inputs space, the association memory space with recurrent weights, the receptive-field space, the weight memory space and the outputs space. The following describes the propagation of signals in each space and the basic functions of each space.

1) Input space **C**: which can be described as the $\boldsymbol{c} = \left[c_1, \cdots, c_i, \cdots, c_{n_i}\right]^T \in \Re^{n_i}$, c_i is the ith input in layer 1. Based on the specific control space, all variables of the input state c_i can be quantized to discrete regions (namely, an element).
2) Association memory space (Membership function) **A**: usually several elements are accumulated into one block and the number of blocks n_k is usually no less than two. **A** represents an association memory space with n_A ($n_A = n_i \times n_k$) components. In the space, each block performs the Gaussian function as a receptive-field basis function, and is described as:

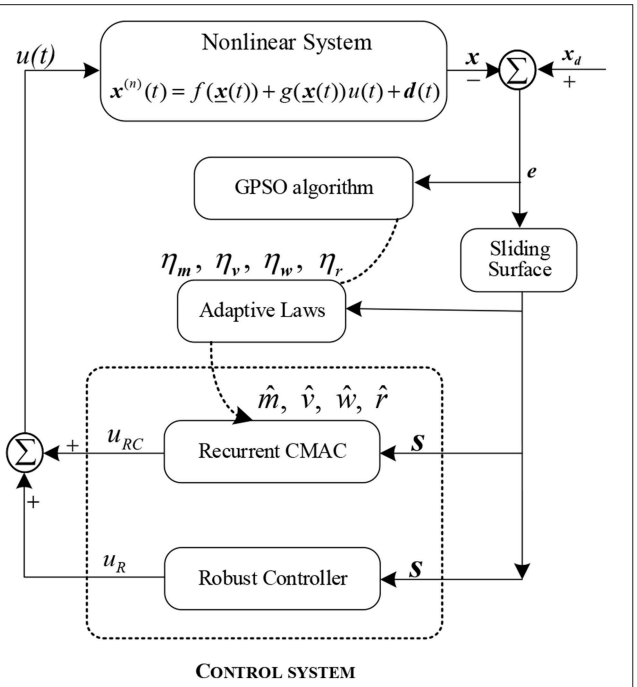

FIGURE 1 | The architecture of the GPSO-RARC control system.

$$\phi_{ik} = \exp\left[\frac{-(cr_i - m_{ik})^2}{v_{ik}^2}\right], \text{for } i = 1, 2, ..., n_i,\ k = 1,\ 2,\ \cdots, n_i \quad (6)$$

in which ϕ_{ik} denotes the kth block of the ith input cr_i with the mean m_{ik} and the variance v_{ik}. In general, the input of this block can be described as follows:

$$cr_i(t) = c_i(t) + r_{ik}\,\phi_{ik}(t - T) \quad (7)$$

in which represents the recurrent gain, $\phi_{ik}(t-T) \underline{\underline{\Delta}}\, \phi_{ikT}$ indicates the value of ϕ_{ik} through time delay T. Obviously, the input of this block includes the memory term ϕ_{ikT}, which saves the previous information about the network and presents dynamic mapping. This is the obvious difference between RCMAC and traditional CMAC. Where the variable c_1 is separated into blocks A and , while the variable c_2 is separated into blocks a and b. Shifting each variable to an element yields different blocks. For example, in **Figure 2B** the block C and D for c_1, while the block c and d for c_2 are obtained by shifting an element. In this space, each block has three adjustable parameters, named m_{ik}, v_{ik}, and r_{ik}.

3) Receptive-field space (Hypercube) : H regions composed of blocks (called Aa and Bb) are called receptive-fields. The kth multidimensional receptive field function is described as follow:

$$h_k(\boldsymbol{cr}, \boldsymbol{m}_k, \boldsymbol{v}_k, \boldsymbol{r}_k) = \prod_{i=1}^{n_i}\phi_{ik} = \prod_{i=1}^{n_i}\exp\left(-\left(\frac{cr_i - m_{ik}}{v_{ik}}\right)^2\right) \quad (8)$$

$$\text{for } i = 1, 2, ..., n_i, \text{ and } k = 1, 2, ..., n_k$$

in which $\boldsymbol{cr} = [cr_1, cr_2, \cdots, cr_{n_i}]^T \in R^{n_i}$, $\boldsymbol{m}_k = [m_{1k}, m_{2k}, \cdots, m_{n_ik}]^T \in R^{n_i}$ and $\boldsymbol{v}_k = [v_{1k}, v_{2k}, \cdots, v_{n_ik}]^T \in R^{n_i}$. Meanwhile, the multidimensional receptive-field functions can also be expressed in a vector form:

$$\begin{aligned}\boldsymbol{H} &= \left[h_{11}, \ldots, h_{1n_k}, h_{21}, \ldots, h_{2n_k}, \ldots, h_{n_i1}, \ldots, h_{n_in_k}\right]^T \in \Re^{n_in_k} \\ &= \left[h_1, \ldots, h_l, \ldots, h_{n_l}\right]^T \in \Re^{n_l}\end{aligned} \quad (9)$$

in which h_{ik} is associated with the ith layer and kth block, the field is activated while the input is in the kth receptive-field. At the same time, one or more of the same weights are activated by nearby inputs, and the corresponding blocks export similar outputs. The correlation is a very useful feature of the RCMAC, which is a local generalization.

4) Weight memory (RCMAC output weight) $\boldsymbol{W}$: in this space, the parameter $w_{n_in_k}$ is the weight which parameterizes the RCMAC mapping(connects to h_{ik}), which can be represented by the following formula:

$$\begin{aligned}\boldsymbol{w} &= \left[w_{11}, \ldots, w_{1n_k}, w_{21}, \ldots, w_{2n_k}, \ldots, w_{n_i1}, \ldots, w_{n_in_k}\right]^T \in \Re^{n_in_k} \\ &= \left[w_1, \ldots, w_l, \ldots, w_{n_l}\right]^T \in \Re^{n_l}\end{aligned} \quad (10)$$

in which w_l is automatically adjusted from the initial value via the online algorithm.

5) Output space $\boldsymbol{Y}$: The outputs of RCMAC are the sum of the activated receptive field multiplied by the corresponding weight, expressed as:

$$u_{RCMAC} = y_o = \sum_{i=1}^{n_i}\sum_{k=1}^{n_k} w_{ik}h_{ik} \quad (11)$$

and the outputs with the RCMAC can be described with the following vector form as:

$$u_{RCMAC} = \boldsymbol{y} = \boldsymbol{w}^T\boldsymbol{H} \quad (12)$$

Since the recurrent unit of RCMAC contains the past value of the receptive-field basis function, the results of the control network have the features of dynamic characteristics and simple structure. If the time delay T=0, the system will return to a conventional CMAC mode. Moreover, the RCMAC will be simplified as a recurrent neural network, in case of each block carries only an element, and each input space has only one layer. Therefore, RCMAC is a generalization of recurrent neural networks, but it is more general, faster to learn and more to recall than the latter.

Adaptive Law for RCMAC Control System

The robust adaptive RCMAC control system includes an adaptive recurrent CMAC and a robust controller which is shown in **Figure 1**, and output of the system as the following:

$$u(t) = u_{RARC} = u_{ARCMAC} + u_R \quad (13)$$

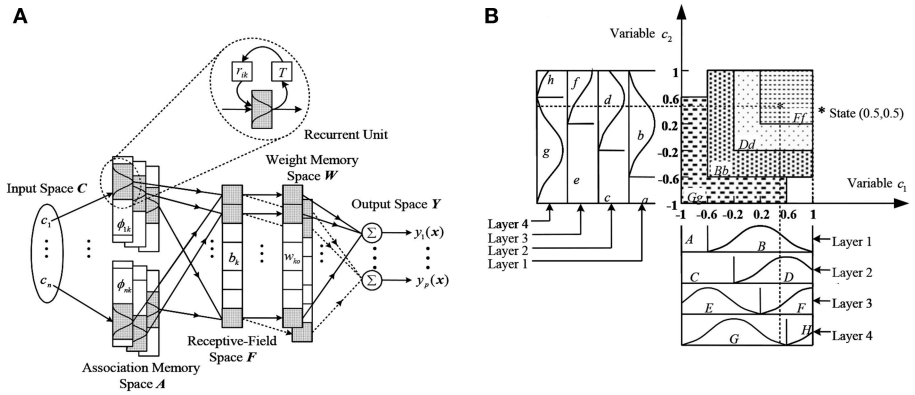

FIGURE 2 | (A) Architecture of an RCMAC. **(B)** Organization of a 2-D RCMAC.

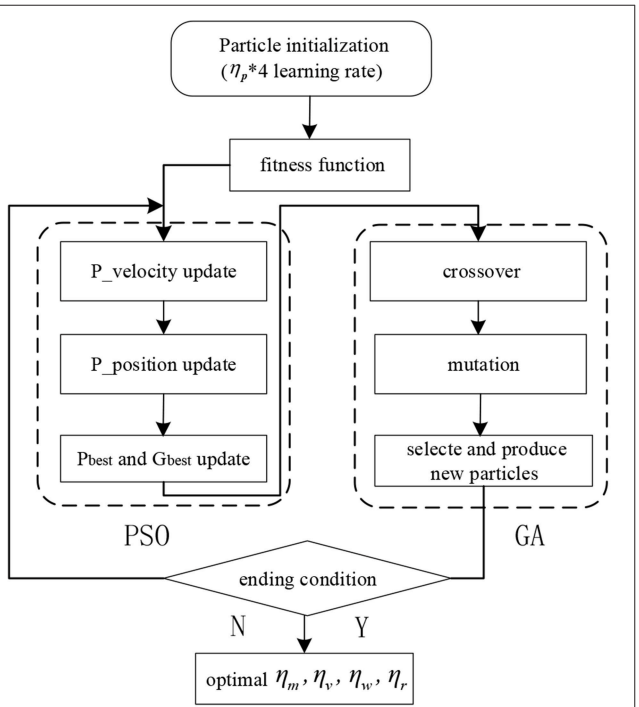

FIGURE 3 | The flowchart of the GPSO algorithm.

in which u_{ARCMAC} is the output of the developed adaptive RCMAC and u_R is the output of the robust compensation. u_{ARCMAC} is the main controller of RCMAC, which is used to approximate the ideal controller in formula (4). The parameters of RCMAC are adjusted online by the adaptive laws. u_R is the robust controller used to efficiently restrain the influence of residual approximation error between the RCMAC and the ideal controller, and guarantees the L_2-stability of the control system.

A sliding surface $\boldsymbol{s}(t)$ can be defined as follow:

$$\boldsymbol{s}(t) = \boldsymbol{e}^{(n-1)} + \boldsymbol{K_1 e}^{(n-2)} + \cdots + \boldsymbol{K_{n-1} e} + \boldsymbol{K_n} \int_0^t \boldsymbol{e}(\tau) d\tau \quad (14)$$

in which $\boldsymbol{s}(t) = \left[s_1(t), s_2(t), \cdots s_m(t)\right]^T \in \Re^m$, taking the derivative about (14), and substituting with (1) and (13)

$$\begin{aligned} \dot{\boldsymbol{s}}(t) &= \boldsymbol{e}^{(n)} + \boldsymbol{K_1 e}^{(n-1)} + \cdots + \boldsymbol{K_n \dot{e}} = \boldsymbol{e^{(n)}} + \underline{\boldsymbol{K}}^T E \\ &= \boldsymbol{x}_d{}^{(n)} - f(\underline{\boldsymbol{x}}) - g(\underline{\boldsymbol{x}})u(t) - \boldsymbol{d}(t) + \underline{\boldsymbol{K}}^T E \end{aligned} \quad (15)$$

then define $L = \frac{1}{2}s^2(t)$ as the cost function, and its derivative is $\dot{L} = \boldsymbol{s}(t)\dot{\boldsymbol{s}}(t) \leq 0$ and substituting (13) in it

$$\begin{aligned} \dot{L} &= \boldsymbol{s}(t)\dot{\boldsymbol{s}}(t) = \boldsymbol{s}(t)[\boldsymbol{x}_d{}^{(n)} - f(\underline{\boldsymbol{x}}) - g(\underline{\boldsymbol{x}})(u_{RCMAC} + u_R) \\ &\quad -\boldsymbol{d}(t) + \underline{\boldsymbol{K}}^T E] \end{aligned} \quad (16)$$

According to the steepest gradient descent algorithm, the parameters of RCMAC, $\dot{\hat{w}}_k$, $\dot{\hat{m}}_{ik}$, $\dot{\hat{v}}_{ik}$ and $\dot{\hat{r}}_{ik}$ can be updated by the tuning laws as below:

$$\begin{aligned} \dot{\hat{w}}_k &= -\eta_w \frac{\partial \boldsymbol{s}(t)\, \dot{\boldsymbol{s}}(t)}{\partial \hat{w}_k} = -\eta_w \frac{\partial \boldsymbol{s}(t)\, \dot{\boldsymbol{s}}(t)}{\partial u_{RCMAC}} \frac{\partial u_{RCMAC}}{\partial \hat{w}_k} \\ &= \eta_w \boldsymbol{s}(t) \boldsymbol{g} h_k \end{aligned} \quad (17)$$

$$\begin{aligned} \dot{\hat{m}}_{ik} &= -\eta_m \frac{\partial \boldsymbol{s}(t)\dot{\boldsymbol{s}}(t)}{\partial u_{RCMAC}} \frac{\partial u_{RCMAC}}{\partial h_k} \frac{\partial h_k}{\partial \phi_{ik}} \frac{\partial \phi_{ik}}{\partial \hat{m}_{ik}} \\ &= \eta_m \boldsymbol{s}(t) \boldsymbol{g} \hat{w}_k h_k \frac{2(cr_i - m_{ik})}{\hat{v}_{ik}^2} \end{aligned} \quad (18)$$

$$\begin{aligned} \dot{\hat{v}}_{ik} &= -\eta_v \frac{\partial \boldsymbol{s}(t)\dot{\boldsymbol{s}}(t)}{\partial u_{RCMAC}} \frac{\partial u_{RCMAC}}{\partial h_k} \frac{\partial h_k}{\partial \phi_{ik}} \frac{\partial \phi_{ik}}{\partial \hat{v}_{ik}} \\ &= \eta_v \boldsymbol{s}(t) \boldsymbol{g} \hat{w}_k h_k \frac{2(cr_i - m_{ik})^2}{\hat{v}_{ik}^3} \end{aligned} \quad (19)$$

$$\begin{aligned} \dot{\hat{r}}_{ik} &= -\eta_r \frac{\partial \boldsymbol{s}(t)\dot{\boldsymbol{s}}(t)}{\partial u_{RCMAC}} \frac{\partial u_{RCMAC}}{\partial h_k} \frac{\partial h_k}{\partial \phi_{ik}} \frac{\partial \phi_{ik}}{\partial \hat{r}_{ik}} \\ &= \eta_r \boldsymbol{s}(t) \boldsymbol{g} \hat{w}_k h_k \frac{2(cr_i - m_{ik})}{\hat{v}_{ik}^2} \phi_{ikT} \end{aligned} \quad (20)$$

where learning-rates η_w, η_m, and η_r are positive for $\dot{\hat{w}}_k$, $\dot{\hat{m}}_{ik}$, $\dot{\hat{v}}_{ik}$, and $\dot{\hat{r}}_{ik}$, respectively.

Genetic Particle Swarm Optimization (GPSO) Algorithm

Particle swarm optimization algorithm (PSO) is a swarm intelligence algorithm designed by simulating hunting behavior of birds. The PSO moves the individuals in the population to a good region according to the adaptability of the environment. However, instead of using evolutionary operators, each individual fly in the D-dimensional search space at a certain speed and is regarded as a non-volume particle, and dynamically adjusts according to the flight experience of itself and its companions. The ith particle is represented as $X_i = (x_{i1}, x_{i2}, \ldots x_{iD})$, the best position (with the best adaptive value) it has experienced is $P_i = (p_{i1}, p_{i2}, \ldots p_{iD})$, also known as p_{best}. The index number of the best position experienced by all particles in the population is denoted by the symbol g, namely P_g, also known as g_{best}. The velocity of the particle i is $V_i = (v_{i1}, v_{i2} \ldots, v_{iD})$. For each generation, its d dimension ($1 \leq d \leq D$) is changed according to the following equation:

$$v_{i,d}(k+1) = \omega \cdot v_{i,d}(k) + c_1 \cdot r_1 \cdot (p_{best} - x_{i,d}) + c_2 \cdot r_2 \cdot (g_{best} - x_{i,d}) \quad (21)$$

$$x_{i,d}(k+1) = x_{i,d}(k) + v_{i,d}(k+1) \quad (22)$$

where c_1 and c_2 are the learning factors, which are also called acceleration constant, ω is the inertia factor, r_1 and r_2 are the uniform random numbers within the range of [0,1]. The right side of the formula (21) consists of three parts. The first part is the inertia or momentum part, which reflects the movement habit of the particle, which means that the particle has a tendency to maintain its previous speed. The second part is the cognition part, which reflects the memory or remembrance of the particle's own historical experience, which represents the tendency of the particle to approach its best position in history. The third part is the social part, which reflects the group history experience of synergy and knowledge sharing between particles.

Particle swarm optimization is simple to calculate and converges quickly, but it lacks mutation ability and is easy to diverge. Genetic algorithm has strong global search ability and high efficiency, but it is prone to premature convergence and poor local search ability. Therefore, a GPSO algorithm is proposed in this paper, which integrates the crossover and mutation operations of the genetic algorithm into the optimization iteration process of particle swarm, and adopts adaptive crossover and adaptive mutation to enhance the ability of the population to jump out of the local optimal solution.

Firstly, the main parameters of PSO are improved in the GPSO algorithm. The linear decreasing method is adopted for inertia weight ω , so that the algorithm can have strong global optimization ability in the early stage of search and detailed local search in the late stage of search. The iterative formula is shown in equation (23):

$$\omega(k) = \omega_{start} - (\omega_{start} - \omega_{end})k/k_{\max} \quad (23)$$

where ω_{start} is the weight of initial inertia, ω_{end} is the weight of terminate inertia; k is the current number of iterations; $k_{\max}$ is the maximum number of iterations.

In order to make the algorithm have a strong global search ability in the early iteration process, it can converge to the global optimal quickly in the later stage, the value of learning factors in this paper is evaluated by asymmetric linear variation, as shown in equations (24) and (25):

$$c_1 = c_{1s} - (c_{1s} - c_{1e})k/k_{\max} \quad (24)$$

$$c_2 = c_{2s} - (c_{2s} - c_{2e})k/k_{\max} \quad (25)$$

where c_{1s}, c_{2s} and c_{1e}, c_{2e} are the initial and terminate iterative value of learning factors of c_1, c_2 respectively.

Secondly,the crossover operation of GA is applied to PSO in the GPSO algorithm. Particles in the population are selected and randomly paired, and then paired particles are crossed with selected probability p_c. For cross particles x_i and x_j, the calculation process is shown in equations (26) and (27):

$$\begin{cases} x_i^{k+1} = \alpha_1 x_i^k + (1-\alpha_1)x_j^k \\ x_j^{k+1} = (1-\alpha_1)x_i^k + \alpha_1 x_j^k \end{cases} \quad (26)$$

$$\begin{cases} v_i^{k+1} = \alpha_2 v_i^k + (1-\alpha_2)v_j^k \\ v_j^{k+1} = (1-\alpha_2)v_i^k + \alpha_2 v_j^k \end{cases} \quad (27)$$

where α_1, α_2 are two random numbers within the interval [0, 1], and equations (26) and (27) represent the crossover operation of the position and velocity of the paired particles, respectively.

Then, the mutation operation of GA is applied to PSO in GPSO algorithm. The optimal position of each particle varies with the selected probability p_m. Assuming that the D-dimensional variable of the individual optimal value p_i is p_i^d, the variation operation of p_i^d is carried out with the strategy of random perturbation. The variable β applied is subject to the normal distribution with mean value 0 and variance 1, and its mutation formula is shown in (28).

$$p_i^d = p_i^d(1 + 0.5\beta) \quad (28)$$

The selection of crossover probability p_c and mutation probability p_m is one of the important factors affecting the optimization ability of the algorithm. If the p_c is too small, the generation speed of new individuals will slow down during the iteration. If the p_c is too large, the good individuals that have been generated in the population may be damaged. If p_m is too small, then the ability to generate new individuals by mutation operation will be weakened, which is not conducive to maintaining the diversity of the population. If p_m is too large, it is similar to the random search algorithm. Therefore, this paper proposes an adaptive crossover and adaptive mutation strategy to make p_c and p_m automatically adjust according to the evolutionary state of the population.

The rate and mutation probability are defined as shown in equations (29) and (30):

$$p_c = \begin{cases} \frac{(p_{c1}-p_{c2})(f_{\max}-f')}{f_{\max}-f_{avg}} & f' \geq f_{avg} \\ p_{c1} & f' < f_{avg} \end{cases} \quad (29)$$

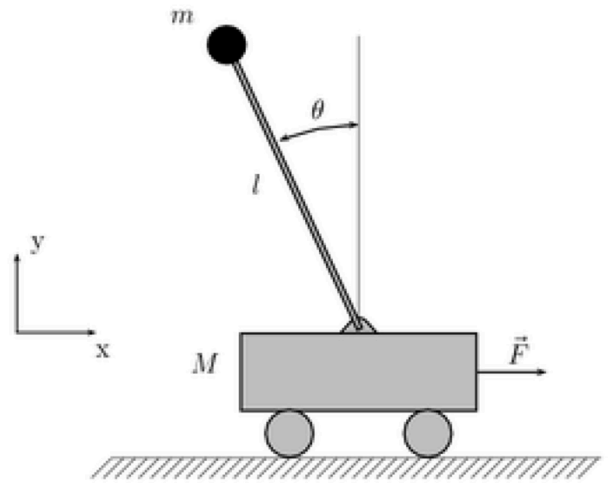

FIGURE 4 | Single inverted pendulum system.

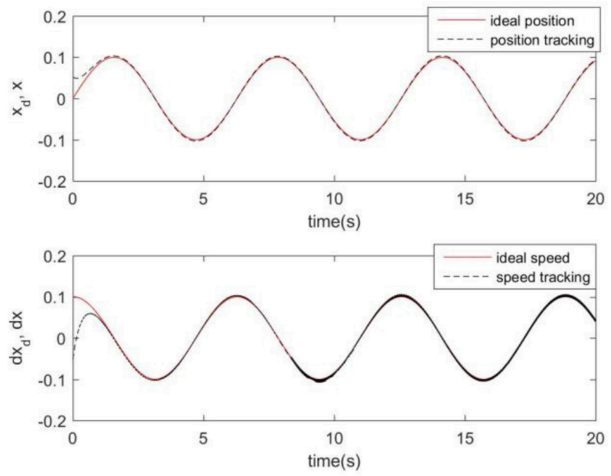

FIGURE 5 | Simulated pendulum systems on angle and angular speed based on RCMAC.

$$p_m = \begin{cases} \frac{(p_{m1}-p_{m2})(f_{\max}-f)}{f_{\max}-f_{avg}} & f \geq f_{avg} \\ p_{m1} & f < f_{avg} \end{cases} \tag{30}$$

where p_{c1}, p_{c2}, p_{m1}, and p_{m2} are constants, f' represents the fitness value corresponding to the better individuals compared with the two individuals with crossover operation; f refers to the fitness function value of the mutant operational particle, f_{avg} refers to the average value of the fitness function value of the entire population at present. From the formula, it can be seen that the probability of crossover operation and mutation operation of individuals whose fitness function value is lower than the population average is relatively high, which ensures the population diversity. At the same time, when $f_{\max} - f_{avg}$ decreases, the individual in the population tends to converge to the local optimal solution. Meanwhile, the probability of individual crossover and mutation will increase, which enhances the ability of the population to generate new individuals and urges them to jump out of the local optimal solution.

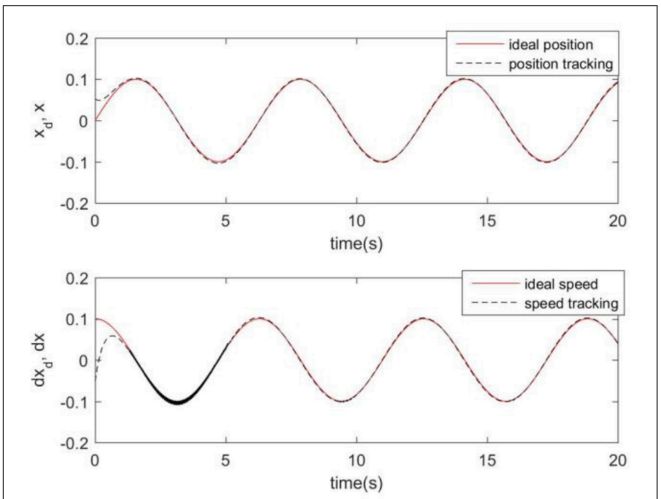

FIGURE 6 | Simulated pendulum systems on angle and angular speed based on RARC_PSO.

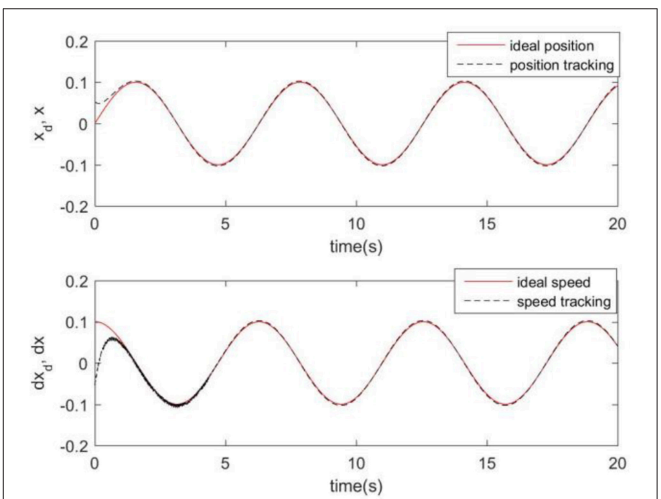

FIGURE 7 | Simulated pendulum systems on angle and angular speed based on RARC_GA.

Finally, the fitness function $f_{itness} = \sum_{i=1}^{m} \left\| e_i(t) \right\|^2$ is chosen as a cost function, to evaluate the performance of learning rates in the GPSO algorithm. The flowchart of GPSO is shown in **Figure 3**.

Robust Compensation Control

There is unavoidably the approximation error between the adaptive recurrent CMAC(ARCMAC) and the ideal controller, an ideal controller can be formulated as the sum of ARCMAC and the approximate error:

$$u^* = u_{ARCMAC} + \boldsymbol{\varepsilon}(t) \tag{31}$$

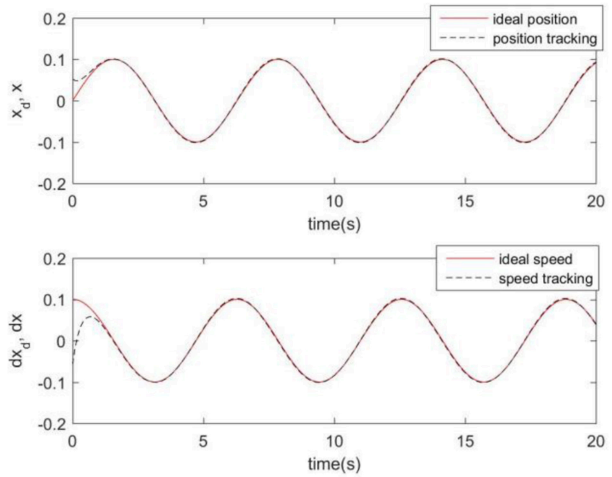

FIGURE 8 | Simulated pendulum systems on angle and angular speed based on RARC_GPSO.

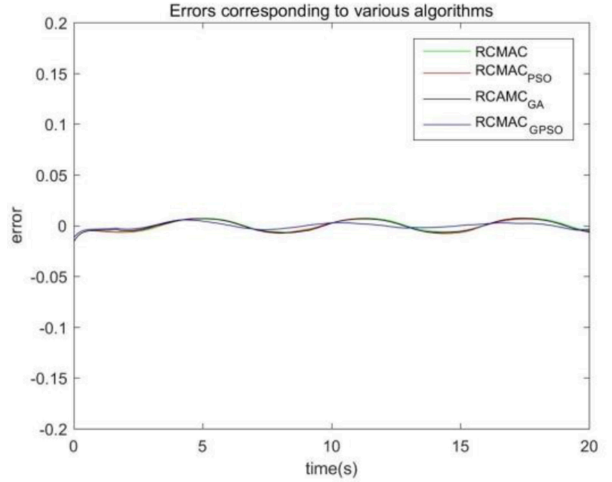

FIGURE 9 | Tracking error for angle on four schemes.

Substituting (13) in (1), yield:

$$\boldsymbol{x}^{(n)} = f(\underline{\boldsymbol{x}}) + g(\underline{\boldsymbol{x}})(u_{ARCMAC} + u_R) + \boldsymbol{d}(t) \tag{32}$$

using the product of (4) and $g(x)$ subtract (32), yield:

$$g(\underline{\boldsymbol{x}})(u^* - u_{ARCMAC} - u_R) = \boldsymbol{e}^{(n)} + \underline{\boldsymbol{K}}^T E = \dot{\boldsymbol{s}}(t) \tag{33}$$

The robust controller can reduce the influence of the approximation error between the ARCMAC and the ideal controller, thus achieving the tracking performance of L_2. Assuming that $\boldsymbol{\varepsilon}(t)$ exists and satisfies L_2 bounded, consider the specified L_2 tracking performance (Chen and Lee, 1996):

$$\sum_{i=1}^{m} \int_0^T s_i^2(t)\, dt \le \sum_{i=1}^{m} [s_i^2(0)/g_{0i}] + \sum_{i=1}^{m} r_i^2 \int_0^T \varepsilon_i^2(t)\, dt \tag{34}$$

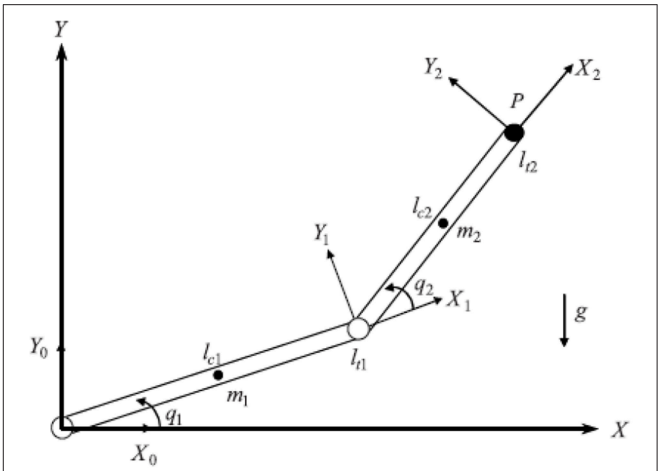

FIGURE 10 | Two-link robot manipulator's architecture.

TABLE 1 | Initial of learning rate and RMSE.

Algorithms	Lr_m	Lr_v	Lr_w	Lr_r	Rmse
RCMAC	0.2	0.3	0.5	0.2	0.0158
RARC_ PSO	0.073	0.059	0.689	0.208	0.0153
RARC_GA	0.021	0.249	0.094	0.519	0.0150
RARC_GPSO	0.050	0.889	0.722	0.291	0.0056

Here r_i is a prescribed positive attenuation constant. The following formula describes the design of a robust controller:

$$u_R(t) = (2R^2)^{-1}(R^2 + I)\, \boldsymbol{s}(t) \tag{35}$$

where $R = diag(r_1, r_2, \cdots, r_m) \in \Re^{m\times m}$ and I is the unity matrix, then further state and prove the following theorem.

Theorem I: while the nth-order MIMO non-linear systems described in (1), the RARC control system is designed as in (13), in which u_{RCMAC} is shown as (12) with the online parameter learning algorithms (17)-(20), and (35) describe the design of the robust controller. Then the desired L_2 tracking performance in (34) can be achieved for the specified attenuation levels r_i , i = 1,2, $\cdots$,m.

Proof: The following formula gives the Lyapunov function:

$$V(\boldsymbol{s}(t)) = \frac{1}{2}\boldsymbol{s}^T(t)\boldsymbol{s}(t) \tag{36}$$

Taking the derivative of the Lyapunov function and using (31), (33) and (35), as below:

$$\begin{aligned}
\dot{V}(\boldsymbol{s}(t)) &= \boldsymbol{s}^T(t)\dot{\boldsymbol{s}}(t) \\
&= \boldsymbol{s}^T(t) g[\boldsymbol{e}(t) - (2R^2)^{-1}(R^2 + I)\, \boldsymbol{s}(t)] \\
&= \sum_{i=1}^{m} g_{0i}[s_i(t)\varepsilon_i(t) - s_i^2(t)\frac{r_i^2+1}{2r_i^2}] \\
&= \sum_{i=1}^{m} g_{0i}[s_i(t)\varepsilon_i(t) - \frac{s_i^2(t)}{2} - \frac{s_i^2(t)}{2\, r_i^2})] \\
&= \sum_{i=1}^{m} g_{0i}[-\frac{s_i^2(t)}{2} - \frac{1}{2}(\frac{s_i(t)}{r_i} - r_i\varepsilon_i(t))^2 + \frac{r_i^2\varepsilon_i^2(t)}{2}] \\
&\le \sum_{i=1}^{m} g_{0i}[-\frac{s_i^2(t)}{2} + \frac{r_i^2\varepsilon_i^2(t)}{2}]
\end{aligned} \tag{37}$$

Assuming $\varepsilon_i(t) \in L_2[0,T], \forall T \in [0,\infty)$, taking the integration of the above equation from $t = 0$ to t= T, yields:

$$V(T) - V(0) \leq \sum_{i=1}^{m} g_{0i}[-\frac{1}{2}\int_0^T s_i^2(t)\,dt + \frac{r_i^2}{2}\int_0^T \varepsilon_i^2(t)\,dt\,] \quad (38)$$

Since $V(T) \geq 0$, the following inequality can be derived from (38):

$$\frac{1}{2}\sum_{i=1}^{m} g_{0i}\int_0^T s_i^2(t)\,dt \leq V(0) + \frac{1}{2}\sum_{i=1}^{m} g_{0i}\, r_i^2 \int_0^T \varepsilon_i^2(t)\,dt \quad (39)$$

using (36), the above inequality is equivalent to the following:

$$\sum_{i=1}^{m}\int_0^T s_i^2(t)\,dt \leq \sum_{i=1}^{m}[s_i^2(0)/g_{0i}] + \sum_{i=1}^{m} r_i^2 \int_0^T \varepsilon_i^2(t)\,dt \quad (40)$$

and the proof is completed.

Moreover, in (24), in the case of $\int_0^T \varepsilon_i^2(t)\,dt < \infty$ then $\int_0^T s_i^2(t)\,dt < \infty$ for all T, so the L_2 stability of the closed-loop system is guaranteed.

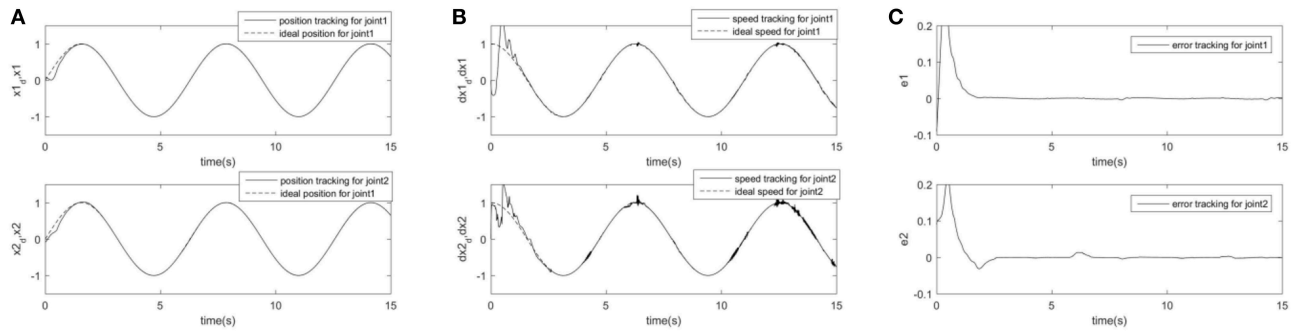

FIGURE 11 | Simulated results of a two-joint manipulator on RCMAC. **(A)** Position tracking for link1 and link2; **(B)** Speed tracking for link1 and link2; **(C)** Tracking error for link1 and link2.

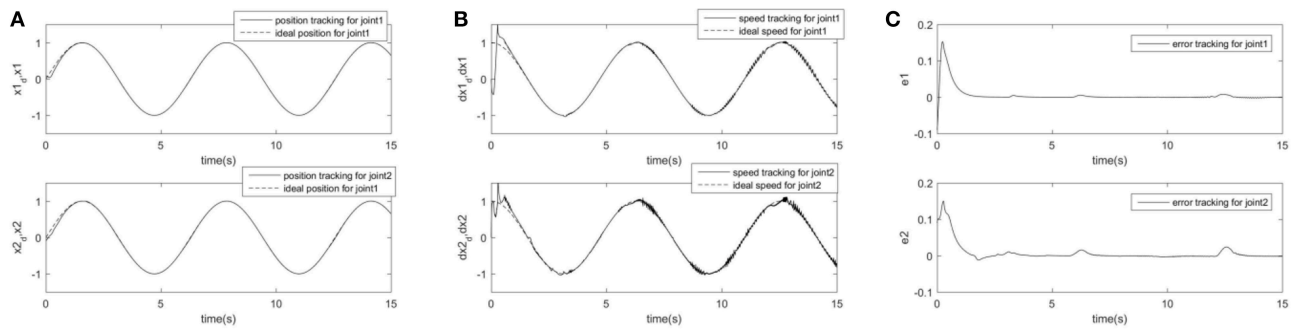

FIGURE 12 | Simulated results of a two-joint manipulator on RARC_GA. **(A)** Position tracking for link1 and link2; **(B)** Speed tracking for link1 and link2; **(C)** Tracking error for link1 and link2.

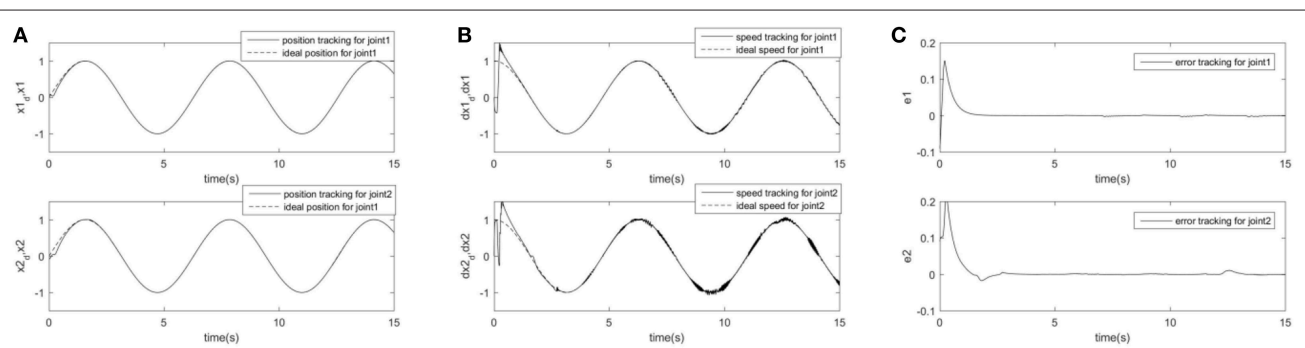

FIGURE 13 | Simulated results of a two-joint manipulator on RARC_PSO. **(A)** Position tracking for link1 and link2; **(B)** Speed tracking for link1 and link2; **(C)** Tracking error for link1 and link2.

SIMULATION RESULTS

Single Inverted Pendulum System

There is the single inverted pendulum system on the vehicle, as shown in **Figure 4**, and its dynamic equation is as follows (Mori et al., 1976):

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f(\boldsymbol{x}) + g(\boldsymbol{x})u \end{aligned} \tag{41}$$

where, $f(\boldsymbol{x}) = \frac{g\sin x_1 - mlx_2{}^2\cos x_1 \sin x_1/(m_c+m)}{l(4/3-m\cos^2 x_1/(m_c+m))}$, $g(\boldsymbol{x}) = \frac{\cos x_1/(m_c+m)}{l(4/3-m\cos^2 x_1/(m_c+m))}$, x_1 and x_2 are the angle and angular velocity of the pendulum, respectively, $g = 9.8m/s^2$ is the acceleration of gravity, $m_c = 1kg$ is the mass of the car, is the mass of the pendulum, $l = 0.5m$ is half the length of the pendulum, u is the control input.

The tracking reference signals are $x_d(t) = 0.1sin(t)$, the initial conditions for this system are set as $x(0) = \pi/60$, $\dot{x}(0) = 0$, the robust compensation $R = 0.05$, the $k_1 = 3$, $k_2 = 1$ then the inputs are $s(t) = 3e + \dot{e} + \int_0^t edt$ and $\dot{s}(t) = \ddot{e} + 3\dot{e} + e$.

Considering the practical application, the off-line training time of the three optimization algorithms is set to 2 s, then the learning rate obtained by off-line training is taken as the initial value of the learning rate parameter of the system controller. For comparison, the original RCMAC control system, the RARC control system based on GA algorithm, the RARC control system based on PSO algorithm, and the RARC control system based on GPSO algorithm are applied to this single inverted pendulum system. Their simulation results are shown in **Figures 5–9**. The state responses $x(t)$ and $\dot{x}(t)$ of normal RCMAC and RARC based on PSO (RARC_ PSO) are shown in **Figures 5**, **6**, respectively and state responses of RARC based on GA (RARC_GA) and RARC based on GPSO (RARC_GPSO) are plotted in **Figures 7**, **8**, respectively. Moreover, the tracking errors of various algorithms are depicted in **Figure 9**, and the Root Mean Square Error (RMSE) are presented in **Table 1**, respectively. Eventually, the simulation results show the proposed RARC_GPSO control system can effectively achieve favorable control for the single inverted pendulum system and can get better tracking performance than others, especially in the tracking of the angular speed.

Two Link Robot Manipulator System

The controlled object is an n-joint robot manipulator, and its non-linear dynamic equation is (Lewis et al., 2003):

$$M(x)\ddot{x} + V(x,\dot{x})\dot{x} + G(x) + F(\dot{x}) + \tau_d = \tau \tag{42}$$

where $M(x) \in R^{n\times n}$ indicates inertia matrix which is the symmetrical positive definite, $V(x,\dot{x}) \in R^n$ means the term of centrifugal force and Coriolis force, $\dot{M}(x) - 2V(x,\dot{x})$ is the oblique symmetric matrix, $G(x) \in R^n$ is the gravity, $F(\dot{x}) \in R^n$ is the term of friction, $\tau_d(t) \in R^n$ is an unknown external disturbance, $\tau(t) \in R^n$ is the joint torque vector applied by the actuator, and $x \in R^n$ represents the vector of the joint variable.

A two-joint system is presented as follow, as shown in **Figure 10**, and the similar design process can be extended to any n-joint system. The specific system parameters of the two joint manipulators are described as below:

$$M(x) = \begin{bmatrix} l_2{}^2 m_2 + l_1{}^2 (m_1 + m_2) + 2l_1 l_2 m_2 \cos(x_2) & l_2{}^2 m_2 + l_1 l_2 m_2 \cos(x_2) \\ l_2{}^2 m_2 + l_1 l_2 m_2 \cos(x_2) & l_2 m_2 \end{bmatrix} \tag{43}$$

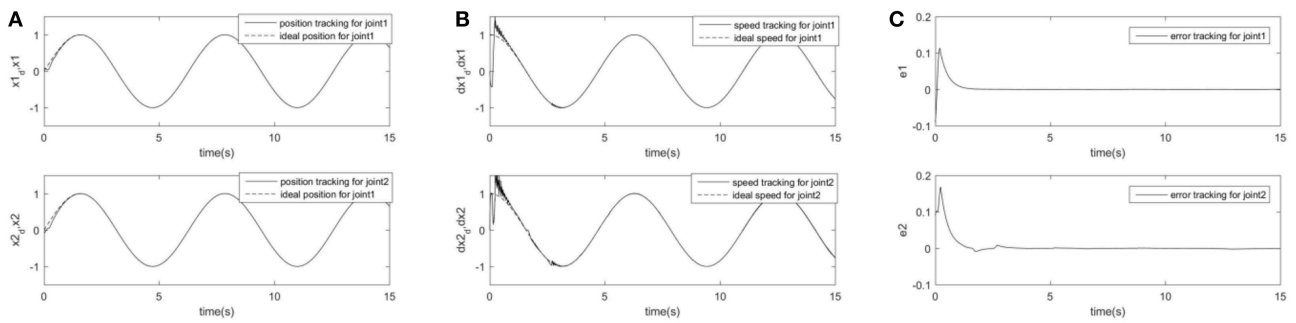

FIGURE 14 | Simulated results of a two-joint manipulator on RARC_GPSO. **(A)** Position tracking for link1 and link2; **(B)** Speed tracking for link1 and link2; **(C)** Tracking error for link1 and link2.

TABLE 2 | Initial of learning rate and RMSE.

Algorithms	Lr_m1	Lr_v1	Lr_w1	Lr_r1	Lr_m2	Lr_v2	Lr_w2	Lr_r2	Rmse1	Rmse2
RCMAC	0.1	0.1	0.5	0.1	0.1	0.1	0.5	0.1	0.0442	0.0369
RARC_PSO	0.1639	0.575	0.583	0.032	0.233	0.439	0.544	0.137	0.0194	0.0308
RARC_GA	0.459	0.563	0.456	0.044	0.471	0.586	0.492	0.087	0.0174	0.0279
RARC_GPSO	0.429	0.838	0.600	0.085	0.613	0.704	0.770	0.174	0.0145	0.0185

$$V(x,\dot{x}) = \begin{bmatrix} -l_1 l_2 m_2 \dot{x}_2 \sin(x_2) & -l_1 l_2 m_2 (\dot{x}_1 + \dot{x}_2) \sin(x_2) \\ m_2 l_1 l_2 \sin(x_2) & 0 \end{bmatrix} \quad (44)$$

$$G(x) = \begin{bmatrix} (m_1 + m_2) l_1 g \cos(x_1) + l_2 m_2 \cos(x_1 + x_2) \\ m_2 l_2 g \cos(x_1 + x_2) \end{bmatrix} \quad (45)$$

where x_1, x_2, m_1, m_2 and l_1, l_2 are the angle, mass and length of joint 1 and 2, respectively. g means the gravity acceleration.

In addition, the dynamics of the manipulator also includes the non-linear viscous and dynamic friction terms of $F(\dot{x})$ and the unknown disturbance τ_d, as follow:

$$\begin{cases} l_1 & = 1.0 \text{ m } l_2 = 1.0 \text{ m} \\ m_1 & = 0.8 \text{ kg } m_2 = 2.3 \text{ kg} \\ g & = 9.8 \text{ m/s}^2 \\ F(\dot{x}) & = 0.02 sgn(\dot{x}) \\ \tau_d & = [0.2 \sin(t) \;\; 0.2 \sin(t)] \end{cases} \quad (46)$$

The initial state of the system is $[x_{1d}, \dot{x}_{1d}, x_{2d}, \dot{x}_{2d}]^T = [0.09\ 0\ -0.09\ 0]^T$, and the expected trajectory are represented as:

$$x_{1d}(t) = \sin t \;\; x_{2d}(t) = \sin t \quad (47)$$

in the robust compensation $R = 0.5 * I$, $k_1 = 3$ and $k_2 = 1$.

In order to further prove the superiority and robustness of RARC_GPSO control system, the other three neural network control schemes (normal RCMAC NN, RARC_PSO NN and RARC_GA NN) are adopted to compare the position and velocity tracking of the manipulator's joints, as shown in **Figures 11–13**. The control performance of the robust adaptive RCMAC control system based on GPSO in the two-joint manipulator is shown in **Figure 14**. It is obvious that the performance of RARC_GPSO is better than that of the other three control methods in the velocity tracking and position tracking, and it has the best speed of error convergence. **Table 2** represents RMSE of the four controllers designed above, which reconfirm that the robust adaptive RARC_GPSO is more excellent than the others in robot manipulator control.

CONCLUSION

A robust adaptive RCMAC control system has been successfully proposed for non-linear MIMO systems in this paper. The main findings of this study are the development of a GPSO-based RCMAC with the adaptive law for updating parameters, and the learning rates can be optimized to best value based on the GPSO algorithm. The control system includes an ARCMAC which is developed to simulate the ideal controller, and a robust controller which is designed to compensate for the difference between ARCMAC and ideal controller. In this design, the optimal learning rate and adaptive learning algorithm of controller parameters are derived, and the L_2-stability of the system is proved by Lyapunov function. Furthermore, the simulation results also prove the effectiveness of the control system. The GPSO-based RCMAC has dynamic characteristics because it considers the past value of received field basis function in associative memory space, so it has outstanding performance in general motion control and trajectory tracking. If the control scheme is applied to classification, the effect is not very good mainly because each sample in the classification is not necessarily linked. The next research plan will refer to the framework of fuzzy theory in the control of nonlinear systems (Zhong et al., 2019), and constantly improve the algorithm to make it universal.

AUTHOR CONTRIBUTIONS

JG designed the experiment and drafted the manuscript. SH and SK performed the experiment. YZ processed the data. YS and C-ML modified the paper. All authors read and approved the final manuscript.

ACKNOWLEDGMENTS

This work was supported in part by the Natural Science Foundation of Fujian Province, China (No. 2017J01782), in part by Science and Technology Guiding Project of Xiamen city (No. 3502Z20179020), in part by Science and Technology Planning Project of Longyan city (No. 2017LY90), and in part by High-level Talent Programs of Xiamen university of technology (No. YKJ18003R).

REFERENCES

Albus, J. S. (1975). A new approach to manipulator control: the cerebellar model articulation controller (CMAC). *J. Dyn. Syst. Measure. Control* 97, 220–227.

Bingul, Z. (2007). Adaptive genetic algorithms applied to dynamic multiobjective problems. *Appl. Soft Comput.* 7, 791–799. doi: 10.1016/j.asoc.2006.03.001

Cai, L., and Xia, L. (2006). Improvement on crossover operation of genetic algorithms. *Syst. Eng. Electron.* 28, 925–928. doi: 10.3321/j.issn:1001-506X.2006.06.039

Chen, B. S., and Lee, C. H. (1996). H∞ tracking design of uncertain nonlinear SISO systems: adaptive fuzzy approach. *IEEE Trans. Fuzzy Syst.* 4, 32–43.

Chen, M., Ge, S. S., and How, B. V. (2010). Robust adaptive neural network control for a class of uncertain MIMO nonlinear systems with input nonlinearities. *IEEE Transact. Neural Networks* 21, 796–812. doi: 10.1109/TNN.2010.2042611

Commuri, S., Jagannathan, S., and Frank Lewis, L. (1997). CMAC neural network control of robot manipulators. *J. Robot. Syst.*14, 465–482.

Commuri, S., and Lewis, F. L. (1995). "Control of unknown nonlinear dynamical systems using CMAC neural networks: structure, stability, and passivity," in *Intelligent Control, 1995. Proceedings of the 1995 IEEE International Symposium* (Monterey, CA). doi: 10.1109/ISIC.1995.525048

Eberhart, R., and Kennedy, J. (1995). "A new optimizer using particle swarm theory," in *Proceedings of the Sixth International Symposium on Micro Machine and Human Science, 1995* (Nagoya). doi: 10.1109/MHS.1995.494215

Guan, J., Lin, C. M., Ji, G. L., Qian, L. W., and Zheng, Y. M. (2018). Robust adaptive tracking control for manipulators based on a TSK fuzzy cerebellar model articulation controller. *IEEE Access* 6, 1670–1679. doi: 10.1109/ACCESS.2017.2779940

Guan, J. S., Lin, L. Y., Ji, G. L., Lin, C. M., Le, T. L., and Egyetem, R. Ó. (2016). Breast tumor computer-aided diagnosis using self-validating cerebellar model neural networks. *Acta Polytech. Hung.* 13, 39–52. doi: 10.12700/aph.13.4.2016.4.3

Holland, J. H. (1973). Genetic algorithms and the optimal allocation of trials. *SIAM J. Comp.* 2, 88–105. doi: 10.1137/0202009

Holland, J. H. (1992). Genetic algorithms. *Sci. Am.* 267, 66–73. doi: 10.1038/scientificamerican0792-66

Hsu, C. F., Lin, C. M., and Lee, T. T. (2006). Wavelet adaptive backstepping control for a class of nonlinear systems. *IEEE Transact. Neural Networks* 17, 1175–1183. doi: 10.1109/TNN.2006.878122

Hunt, K. J., Sbarbaro, D., Zbikowski, R., and Gawthrop, P. J. (1992). Neural networks for control systems—a survey." *Automatica* 28, 1083–1112.

Kennedy, J. (2011). *Particle Swarm Optimization. Encyclopedia of Machine Learning*. Boston, MA: Springer.

Lam, H. K. (2018). A review on stability analysis of continuous-time fuzzy-model-based control systems: from membership-function-independent to membership-function-dependent analysis. *Eng. Appl. Artif. Intellig.* 67, 390–408. doi: 10.1016/j.engappai.2017.09.007

Lee, C. H., and Teng, C. C. (2000). Identification and control of dynamic systems using recurrent fuzzy neural networks. *IEEE Transact. Fuzzy Syst.* 8, 349–366. doi: 10.1109/91.868943

Lewis, F. L., Dawson, D. M., and Abdallah, C. T. (2003). *Robot Manipulator Control: Theory and Practice*. Boca Raton, FL: CRC Press.

Lin, C. J., and Chin, C. C. (2004). Prediction and identification using wavelet-based recurrent fuzzy neural networks. *IEEE Transact. Syst. Man Cyber. B* 34, 2144–2154. doi: 10.1109/TSMCB.2004.833330

Lin, C. M., and Chen, C. H. (2006). Adaptive RCMAC sliding mode control for uncertain nonlinear systems. *Neural Comput. Appl.* 15, 253–267. doi: 10.1007/s00521-006-0027-0

Lin, C. M., and Chen, T. Y. (2009). Self-organizing CMAC control for a class of MIMO uncertain nonlinear systems. *IEEE Transact. Neural Networks* 20, 1377–1384. doi: 10.1109/TNN.2009.2013852

Lin, C. M., and Peng, Y. F. (2004). Adaptive CMAC-based supervisory control for uncertain nonlinear systems. *IEEE Transact. Syst. Man Cyber. B* 34, 1248–1260. doi: 10.1109/TSMCB.2003.822281

Misra, J., and Saha, I. (2010). Artificial neural networks in hardware: a survey of two decades of progress. *Neurocomputing* 74, 239–255. doi: 10.1016/j.neucom.2010.03.021

Mori, S., Nishihara, H., and Furuta, K. (1976). Control of unstable mechanical system control of pendulum. *Int. J. Control* 23, 673–692.

Peng, X. B., Gui, W. H., Huang, Z. W., Hu, Z. H., and Li, Y.G. (2008). GAPSO: effective genetic particle swarm algorithm and its application. *J. Syst. Simul.* 20, 5025–5027. doi: 10.16182/j.cnki.joss.2008.18.068

Peng, Y. F., and Lin, C. M. (2007). Adaptive recurrent cerebellar model articulation controller for linear ultrasonic motor with optimal learning rates. *Neurocomputing* 70, 2626–2637. doi: 10.1016/j.neucom.2006.05.018

Qinghua, L., Shida, Y., and Youlin, R. (2006). Improving optimization for genetic algorithms based on level set. *J. Comp. Res. Dev.* 43, 1624–1629.

Sak, H., Senior, A., and Beaufays, F. (2014). "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Fifteenth Annual Conference of the International Speech Communication Association* (Singapore).

Song, Q., Wu, Y., and Soh, Y. C. (2008). Robust adaptive gradient-descent training algorithm for recurrent neural networks in discrete time domain. *IEEE Transact. Neural Networks* 19, 1841–1853. doi: 10.1109/TNN.2008.2001923

Wang, D., and Huang, J. (2005). Neural network-based adaptive dynamic surface control for a class of uncertain nonlinear systems in strict-feedback form. *IEEE Transact. Neural Networks* 16, 195–202. doi: 10.1109/TNN.2004.839354

Zhong, Z., Wai, R. J., Shao, Z., and Xu, M. (2017). Reachable set estimation and decentralized controller design for large-scale nonlinear systems with time-varying delay and input constraint. *IEEE Transact. Fuzzy Syst.* 25, 1629–1643. doi: 10.1109/TFUZZ.2016.2617366

Zhong, Z., Zhu, Y., and Lam, H. K. (2018). Asynchronous piecewise output-feedback control for large-scale fuzzy systems via distributed event-triggering schemes. *IEEE Transact. Fuzzy Syst.* 26, 1688–1703. doi: 10.1109/TFUZZ.2017.2744599

Zhong, Z., Zhu, Y., Lin, C-M., and Huang, T. (2019). A fuzzy control framework for interconnected nonlinear power networks under TDS attack: Estimation and compensation. *J. Franklin Inst.* doi: 10.1016/j.jfranklin.2018.12.012. [Epub ahead of print].

Zhu, Q., Fei, S., Zhang, T., and Li, T. (2008). Adaptive RBF neural-networks control for a class of time-delay nonlinear systems. *Neurocomputing* 71, 3617–3624. doi: 10.1016/j.neucom.2008.04.012

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.