Fractional Hopfield Neural Networks: Fractional Dynamic Associative Recurrent Neural Networks

Yi-Fei Pu, Zhang Yi, *Fellow, IEEE*, and Ji-Liu Zhou, *Senior Member, IEEE*

***Abstract*— This paper mainly discusses a novel conceptual framework: fractional Hopfield neural networks (FHNN). As is commonly known, fractional calculus has been incorporated into artificial neural networks, mainly because of its long-term memory and nonlocality. Some researchers have made interesting attempts at fractional neural networks and gained competitive advantages over integer-order neural networks. Therefore, it is naturally makes one ponder how to generalize the first-order Hopfield neural networks to the fractional-order ones, and how to implement FHNN by means of fractional calculus. We propose to introduce a novel mathematical method: fractional calculus to implement FHNN. First, we implement fractor in the form of an analog circuit. Second, we implement FHNN by utilizing fractor and the fractional steepest descent approach, construct its Lyapunov function, and further analyze its attractors. Third, we perform experiments to analyze the stability and convergence of FHNN, and further discuss its applications to the defense against chip cloning attacks for anticounterfeiting. The main contribution of our work is to propose FHNN in the form of an analog circuit by utilizing a fractor and the fractional steepest descent approach, construct its Lyapunov function, prove its Lyapunov stability, analyze its attractors, and apply FHNN to the defense against chip cloning attacks for anticounterfeiting. A significant advantage of FHNN is that its attractors essentially relate to the neuron's fractional order. FHNN possesses the fractional-order-stability and fractional-order-sensitivity characteristics.**

***Index Terms*— Defense against chip cloning attacks, fractional calculus, fractional Hopfield neural networks (FHNNs), fractional-order-sensitivity, fractional-order-stability.**

I. Introduction

IT IS well known that the classical first-order Hopfield neural networks (HNNs) is one of the most influential neural networks [1]–[4]. The circuit configuration of HNN's first-order neuron is based on a first-order integral circuit. Each first-order neuron of HNN consists of one operational amplifier and its related capacitor and resistors. Each first-order neuron has the same circuit configuration. There are many classical applications of HNN in content addressable memory [1], analog-to-digital converters [5], linear programming [3], and so on. Meanwhile, there are also many dynamic associative memories that are closely related to HNN, such as the Li *et al.* neural networks [6], [7], bidirectional associative memories [8], [9], and so on. Furthermore, with the widespread application of HNN, some model modifications of HNN, such as the high-order HNNs [10]–[13], fuzzy HNNs [14], [15], and stochastic HNNs [16], [17], are proposed, respectively. In addition, fractional calculus has been incorporated into artificial neural networks, mainly because of its long-term memory and nonlocality. Some researchers have made interesting attempts at fractional neural networks and gained competitive advantages over the integer-order neural networks. For instance, Özdemir *et al.* [18] proposed a new type of activation function for a complex valued neural network. Alofi *et al.* [19] studied the finite-time stability of Caputo fractional neural networks with distributed delay. Kaslik and Sivasundaram [20] discussed the stability analysis of the fractional-order neural networks of Hopfield type. Zhang *et al.* [21] discussed a fractional-order financial system based on a fractional-order 3-D Hopfield type neural network. Raja *et al.* [22]–[25] proposed stochastic techniques as well as evolutionary techniques for the solution of the fractional-order systems represented by fractional differential equations, respectively. In these approaches, feedforward artificial neural networks are employed for accurate mathematical modeling. The advantage of these approaches is that the solution of fractional differential equations is available in the domain of continuous inputs unlike the other integer-order calculation-based numerical techniques. Therefore, it naturally makes one to ponder how to generalize HNN to the fractional-order ones, and how to implement the fractional HNNs (FHNNs) by means of fractional calculus. This paper discusses a novel conceptual framework: FHNN.

In over the past 300 years, fractional calculus has been an important novel branch of mathematical analyses [26]–[31]. Fractional calculus is as old as the integer one, although till date, its application has been exclusively in the field of mathematics. It seems as if fractional calculus is a promising mathematical method for physical scientists and engineering technicians. Scientific study has shown that a fractional order or a fractional dimensional approach is now the best description for many natural phenomena. Fractional calculus is used currently in many fields such as specific physical problems [32], [33], biomedical engineering [34], diffusion processes [35]–[37], viscoelasticity theory [38], fractal dynamics [39], and fractional control [40]. Unfortunately, its major application still focuses on describing the transient state

Manuscript received November 3, 2015; revised February 19, 2016; accepted June 16, 2016. Date of publication November 3, 2015; date of current version September 15, 2017. This work was supported in part by the National Natural Science Foundation of China under Grant 61571312, in part by the Foundation Franco-Chinoise Pour La Science Et Ses Applications, in part by the Science and Technology Support Project of Sichuan Province of China under Grant 2013SZ0071, and in part by the Science and Technology Support Project of Chengdu PU Chip Science and Technology Company, Ltd. *(Corresponding author: Yi-Fei Pu.)*

The authors are with the College of Computer Science, Sichuan University, Chengdu 610065, China (e-mail: puyifei_007@hotmail.com; zhangyi@scu.edu.cn; zhoujl@scu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2016.2582512

of physical change, but seldom involves systemic evolution processes.

How to apply fractional calculus to signal analysis and processing, especially to neural networks, is an emerging field of study and few studies have been seldom performed in this area. The properties of the fractional calculus of a signal are quite different from those of its integer-order calculus [41]–[43]. Therefore, the fractional differential can nonlinearly enhance the complex texture details of an image [44]–[46] and implement texture image denoising approaches [47]–[50]. Following the success in the synthesis of a fractional differentiator in the form of an analog circuit, the emergence of a novel electrical circuit element has been named fractor [26], [43], [51]–[57]. As in our previous studies [51], [52], an ideal fractor consists of an ordinary resistor and an ordinary capacitor or inductor in the form of an analog circuit on the tree-type [26], two-circuit-type [43], [53], H-type [43], [54], net-grid-type [43], [55]–[57], and other infinite recursive structures, which are of extreme self-similar fractal structure. On this basis, the first preliminary attempt at implementation of a fractional-order neural network of the Hopfield type by means of fractional calculus was reported [57]. Another prior study [58] showed that, in fractional adaptive signal processing and fractional adaptive control, the fractional extreme point is quite different from a traditional integer-order extreme one, such as the first-order stationary point. In order to seek the fractional extreme points of the energy norm, we have generalized the integer-order steepest descent approach to a fractional approach [58]. Based on the prior studies mentioned above [26], [43], [45], [51]–[58], we propose to introduce a novel mathematical method: fractional calculus to implement FHNN. A significant advantage of FHNN is that its attractors essentially relate to the neuron's fractional order. FHNN possesses the fractional-order stability and the fractional-order-sensitivity characteristics.

The rest of this paper is organized as follows. Section II recalls the necessary theoretical background of fractional calculus and fractional neural works. Section III implements FHNN and studies its stability and convergence. First, we implement fractor in the form of an analog circuit. Second, we implement FHNN by utilizing fractor and the fractional steepest descent approach. Third, we construct the Lyapunov function of FHNN. Fourth, we analyze the attractors of FHNN. Section IV reports the experiment results and analysis. First, we deduce numerical implementation of FHNN. Second, we analyze the stability and convergence of FHNN. Third, we study the applications of FHNN to the defense against chip cloning attacks for anticounterfeiting. In Section V, the conclusions of this paper are presented.

II. Mathematical Background

This section presents a brief introduction to the necessary mathematical background of fractional calculus. The commonly used fractional calculus definitions are those of Grünwald–Letnikov, Riemann–Liouville, and Caputo [26]–[30].

The Grünwald–Letnikov definition of fractional calculus, in a convenient form, for causal signal $s(x)$, is as follows:

$$\begin{aligned} &{}_a^{G-L}D_x^v s(x) \\ &= \lim_{N\to\infty}\left\{\frac{\left(\frac{x-a}{N}\right)^{-v}}{\Gamma(-v)}\sum_{k=0}^{N-1}\frac{\Gamma(k-v)}{\Gamma(k+1)}s\left(x-k\left(\frac{x-a}{N}\right)\right)\right\} \end{aligned} \quad (1)$$

where $s(x)$ is a differintegrable function [26]–[30], $[a, x]$ is the duration of $s(x)$, v is a noninteger, $\Gamma(\alpha) = \int_0^\infty e^{-x}x^{\alpha-1}dx$ is the Gamma function, and ${}_a^{G-L}D_x^v$ denotes the Grünwald–Letnikov defined fractional differential operator.

The Riemann–Liouville definition of the v-order integral, for causal signal $s(x)$, is as follows:

$${}_a^{R-L}I_x^v s(x) = \frac{1}{\Gamma(v)}\int_a^x \frac{s(\tau)}{(x-\tau)^{1-v}}d\tau \quad (2)$$

where $v > 0$ and ${}_a^{R-L}I_x^v$ denotes the Riemann–Liouville left-sided fractional integral operator. The Riemann–Liouville definition of the v-order derivative is as follows:

$${}_a^{R-L}D_x^v s(x) = \frac{1}{\Gamma(n-v)}\frac{d^n}{dx^n}\int_a^x \frac{s(\tau)}{(x-\tau)^{v-n+1}}d\tau \quad (3)$$

where $n-1 \le v < n$, and ${}_a^{R-L}D_x^v$ denotes the Riemann–Liouville left-handed fractional differential operator. The Laplace transform of the v-order Riemann–Liouville differential operator is $L[{}_0^{R-L}D_x^v s(x)] = S^v L[s(x)] - \sum_{k=0}^{n-1} S^k[{}_0^{R-L}D_x^{v-1-k}s(x)]_{x=0}$, where S denotes the Laplace operator. When $s(x)$ is a causal signal and its fractional primitives are also required to be zero, we can simplify the Laplace transform for ${}_0^{R-L}D_x^v s(x)$ as $L[{}_0^{R-L}D_x^v s(x)] = S^v L[s(x)]$.

The Caputo definition of the v-order derivative for causal signal $s(x)$ is as follows:

$${}_a^{C}D_x^v s(x) = \frac{1}{\Gamma(n-v)}\int_a^x (x-\tau)^{n-v-1}s^{(n)}(\tau)d\tau \quad (4)$$

where $0 \le n-1 < v < n$, $n \in R$ and ${}_a^{C}D_x^v$ denotes a Caputo defined fractional differential operator. From (4), we can see that ${}_a^{C}D_x^v$ is equivalent to the successive performance of an n-order differential and an $(n-v)$-order integral of signal $s(x)$. The Laplace transform of the v-order Caputo differential operator is $L[{}_0^{C}D_x^v s(x)] = S^v L[s(x)] - \sum_{k=0}^{n-1} S^k s^{(k)}(x)|_{x=0}$. When $s(x)$ is a causal signal, and its fractional primitives are also required to be zero, we can simplify the Laplace transform for ${}_0^{C}D_x^v s(x)$ as $L[{}_0^{C}D_x^v s(x)] = S^v L[s(x)]$. In this case, the three cited definitions of fractional derivatives are equivalent. In this work, we use the equivalent notations $D_x^v = {}_0^{G-L}D_x^v = {}_0^{R-L}D_x^v = {}_0^{C}D_x^v$ in an arbitrary interchangeable manner.

Nowadays, fractional calculus has been incorporated into artificial neural networks, mainly because of its long-term memory and nonlocality. Some remarkable progress in studies of fractional neural networks not only validates them as fractional dynamic systems, but also gives interesting and practical suggestions for future research. For instance, first, the exploration of the theoretical properties for fractional neural networks is needed. Fractional neural networks employed by fractional activation functions have the modeling capability to achieve the desired parametric learning, but structural

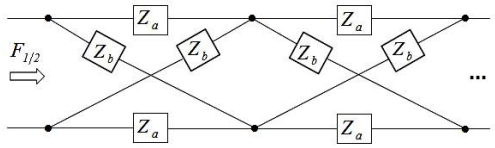

Fig. 1. 1/2-order net-grid-type fractor.

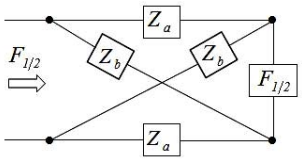

Fig. 2. Equivalent circuit of 1/2-order net-grid-type fractor.

mutation requires specific modification of the algorithm to represent additional complexity. Stability and multistability, bifurcations, and chaos of fractional neural networks should be investigated. Second, the verification of intrinsic differences in the behavior between the fractional-order neural networks and the integer-order neural networks is desired. Behavioral differences are observed in experiments, but whether or not they could be attributed to the inherent differences of the fractional-order networks remains to be seen. Experiments and analyses might also be implemented on different topologies to detect further interesting relationships between these two types of neural networks. Third, any biologically inspired computational intelligence algorithm could be utilized to solve fractional differential equations virtually. With the exception of the stochastic techniques and evolutionary computations, the combination of an artificial neural network aided with other biological-inspired methods, such as an artificial bee colony, might also solve fractional differential equations efficiently.

III. FHNN and Its Stability and Convergence

A. Implementation of Fractor in Form of an Analog Circuit

In this section, in order to implement FHNN in the form of an analog circuit, we first need to perform fractional calculus of the signal in an analog circuit process. Following the success in the synthesis of a fractional differentiator in the form of an analog circuit, the emergence of a novel electrical circuit element was named as fractor [26], [43], [51]–[57]. As in our previous studies [51], [52], an ideal fractor consists of an ordinary resistor and an ordinary capacitor or inductor in the form of an analog circuit on the tree-type, two-circuits-type, H-type, net-grid-type, and other infinite recursive structures, which are of extreme self-similar fractal structure. In this sense, fractance means the fractional-order impedance of a fractor. Consequently, let us denote the identify fractor with the symbol $-\overset{F}{\text{WW}}-$, in which F is the abbreviation for fractor [51]–[57].

In the above-mentioned ideal fractor structures, the net-grid-type fractor has an optimal performance [51]–[57]. The structural representation of the 1/2-order net-grid-type fractor is shown in Fig. 1.

In Fig. 1, $F_{1/2}$ denotes the driving-point impedance function of the 1/2-order net-grid-type fractor. From Fig. 1, we can see that the 1/2-order net-grid-type fractor is of extreme self-similar fractal structure with the series connection of infinitely repeated net-grid-type structure where $Z_{\boldsymbol{a}}$ and $Z_{\boldsymbol{b}}$ are impedances. The number of $Z_{\boldsymbol{a}}$ and $Z_{\boldsymbol{b}}$ is equal to twofold the number of layers. Then, its equivalent circuit is shown in Fig. 2.

Suppose the current of $Z_{\boldsymbol{a}}$ and $Z_{\boldsymbol{b}}$ is equal to $i_{\boldsymbol{a}}$ and $i_{\boldsymbol{b}}$. Let the input voltage and input current of $F_{1/2}$ be equal to $V_i(S)$ and $I_i(S)$, respectively. From Fig. 2, according to Kirchoff's current law and Kirchoff's voltage law

$$\begin{cases} Z_a i_a + Z_b i_b = V_i \\ (Z_a + F_{1/2}) i_a - (F_{1/2} + Z_b) i_b = 0. \end{cases} \tag{5}$$

According to Cramer's rule in linear algebra, it follows that:

$$\begin{cases} i_a = \dfrac{(Z_b + F_{1/2}) V_i}{\begin{vmatrix} Z_a + F_{1/2} & -(F_{1/2} + Z_b) \\ Z_a & Z_b \end{vmatrix}} \\ i_b = \dfrac{(Z_a + F_{1/2}) V_i}{\begin{vmatrix} Z_a + F_{1/2} & -(F_{1/2} + Z_b) \\ Z_a & Z_b \end{vmatrix}}. \end{cases} \tag{6}$$

Hence, $F_{1/2}$ is equal to

$$F_{1/2} = \frac{V_i}{i_a + i_b} = \frac{2Z_a Z_b + F_{1/2}(Z_a + Z_b)}{2F_{1/2} + Z_a + Z_b}. \tag{7}$$

From (7), it follows that:

$$F_{1/2} = (Z_a Z_b)^{1/2}. \tag{8}$$

For the convenience of discussion, we only discuss the issue of the capacitive fractor in the following. An inductive fractor behaves in a similar way. Suppose the initial energy of the electric element of a capacitive fractor is equal to zero. Let the resistance and capacitance of capacitive fractor be equal to r and c, respectively. Then, in the Laplace transform domain, $Z_a = r$ and $Z_b = 1/(cS)$, where S is the Laplace operator. From (8), it follows that:

$$F_{1/2} = \zeta^{1/2} S^{-1/2} \tag{9}$$

where $\zeta = r/c$. From (9), we can derive the relationship between input voltage $V_i(S)$ and input current $I_i(S)$ of the 1/2-order capacitive fractor. It follows that:

$$V_i(S) = \zeta^{1/2} S^{-1/2} I_i(S). \tag{10}$$

The inverse Laplace transform of (10) is as follows:

$$V_i(t) = \frac{\zeta^{1/2}}{\Gamma(1/2)} \int_0^t \frac{I_i(\tau)}{(t-\tau)^{1/2}} d\tau \tag{11}$$

where Γ is the Gamma function. From (4) and (11), we can see that $V_i(t)$ is in direct ratio to the 1/2-order fractional integral of $I_i(t)$. On the other hand, from (10), it follows that:

$$I_i(S) = \zeta^{-1/2} S^{1/2} V_i(S). \tag{12}$$

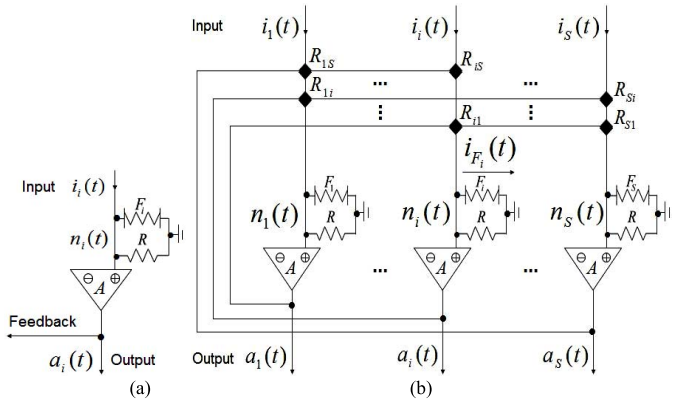

Fig. 3. FHNN model. (a) Circuit configuration for the fractional neuron of FHNN. (b) Circuit configuration of FHNN.

The inverse Laplace transform of (12) is as follows:

$$I_i(t) = \frac{1}{\zeta^{1/2}\Gamma(1/2)} \int_0^t \frac{V_i^{(1)}(\tau)}{(t-\tau)^{1/2}} d\tau. \tag{13}$$

From (4) and (13), we can see that $I_i(t)$ is in direct ratio to the 1/2-order fractional differential of $V_i(t)$.

Similarly, with regard to the v-order capacitive fractor, let us use F_v to denote its driving-point impedance function. We can further derive that $V_i(t)$ is generally in a direct ratio to the v-order fractional integral of $I_i(t)$, whereas $I_i(t)$ is generally in a direct ratio to the v-order fractional differential of $V_i(t)$ [51], [52]. Thus

$$I_i(t) = \frac{1}{\xi^v \Gamma(n-v)} \int_0^t (t-\tau)^{n-v-1} V_i^{(n)}(\tau) d\tau \tag{14}$$

where $\xi = r^{(1-p)/v}/c$, $v = q+p$ is a positive real number, q is a positive integer, $0 \le p \le 1$, $V_i(t)$ is input voltage, and $I_i(t)$ is input current of the v-order capacitive fractor [51], [52].

B. Implementation of FHNN

In this section, we implement FHNN. On the basis of the aforementioned fractor, we can implement FHNN in the form of an analog circuit by utilizing fractor and the fractional steepest descent approach [58]. FHNN model is shown in Fig. 3.

Looking at Fig. 3 and comparing FHNN with the HNN model [1]–[4], it is apparent that the circuit configuration of FHNN's fractional neuron is strikingly dissimilar to the circuit configuration of HNN's first-order neuron. In Fig. 3, each fractional neuron of FHNN consists of one operational amplifier $\boldsymbol{A}$ and its related fractor $\dashv\!\overset{F}{\wedge\!\wedge\!\wedge}\!\vdash$ and resistors. Each fractional neuron has the same circuit configuration. From (14), we know that the fractor $\dashv\!\overset{F}{\wedge\!\wedge\!\wedge}\!\vdash$ implements the v-order fractional calculus. We set $F_1 = F_{v_1}$, $F_i = F_{v_i}$ and $F_S = F_{v_S}$, i.e., the fractional order of FHNN's neuron is equal to v_1, v_i, and v_S, respectively. Thus, input current $i_{F_i}(t)$ is generally in a direct ratio to the v_i-order fractional differential of the input voltage $n_i(t)$ of the v_i-order F_{v_i}. In the Laplace transform domain, set $F_{v_i}(S)$ denotes the v_i-order $F_{v_i}(t)$. Therefore, we can derive the operation rule of FHNN from Kirchhoff's current law. It follows that:

$$I_{F_i}(S) = \frac{N_i(S)}{F_{v_i}(S)} = \sum_{j=1}^{S} T_{i,j} \cdot A_j(S) - \frac{N_i(S)}{Z_i} + I_i(S) \tag{15}$$

where $I_{F_i}(S)$, $N_i(S)$, $F_{v_i}(S)$, $A_j(S)$, and $I_i(S)$ are the Laplace transforms of $i_{F_i}(t)$, $n_i(t)$, $F_{v_i}(t)$, $a_j(t)$, and $i_i(t)$, respectively. The function $n_i(t)$ is the input voltage of the ith fractional neuron's $\boldsymbol{A}$ amplifier, $a_i(t)$ is the output voltage of the ith fractional neuron, and $i_i(t)$ is the input electric current of the ith fractional neuron. $R_{i,j}$ is the feedback resistor being connected to the output of the jth fractional neuron and the input of the ith one. Thus, the conductance is $T_{i,j} = 1/R_{i,j}$. It is assumed that the circuit of FHNN is symmetric, so that $T_{i,j} = T_{j,i}$. We set $1/Z_i = 1/R + \sum_{j=1}^{S} (1/R_{i,j})$.

Furthermore, using (14) the inverse Laplace transform of (15) is as follows:

$$\begin{aligned} i_{F_i}(t) &= K_i \frac{d^{v_i} n_i(t)}{dt^{v_i}} \\ &= \frac{K_i}{\Gamma(n-v_i)} \int_0^t (t-\tau)^{n-v_i-1} n_i^{(n)}(\tau) d\tau \\ &= \sum_{j=1}^{S} T_{i,j} \cdot a_j(t) - \frac{n_i(t)}{Z_i} + i_i(t) \end{aligned} \tag{16}$$

where $K_i = \xi_i^{-v_i}$ is a positive constant. With respect to the capacitive fractor, $\xi_i = r^{(1-p_i)/v_i}/c$, $v_i = q_i + p_i$ is a positive real number, q_i is a positive integer, and $0 \le p_i \le 1$. Multiplying both sides of (16) by Z_i, it follows that:

$$Z_i K_i \frac{d^{v_i} n_i(t)}{dt^{v_i}} = \sum_{j=1}^{S} Z_i T_{i,j} \cdot a_j(t) - n_i(t) + Z_i i_i(t). \tag{17}$$

Define $\chi_i = Z_i K_i$ as a positive constant, and set $w_{i,j} = Z_i T_{i,j}$, $b_i = Z_i i_i$. Thus, it follows that:

$$\chi_i \frac{d^{v_i} n_i(t)}{dt^{v_i}} = -n_i(t) + \sum_{j=1}^{S} w_{i,j} a_j(t) + b_i(t). \tag{18}$$

Thus, it follows that in vector form:

$$\boldsymbol{\chi} \frac{d^{v} \boldsymbol{n}(t)}{dt^{v}} = -\boldsymbol{n}(t) + \boldsymbol{W}\boldsymbol{a}(t) + \boldsymbol{b} \tag{19}$$

where $\boldsymbol{\chi} = [\ \chi_1 \ \cdots \ \chi_i \ \cdots \ \chi_S\ ]^T$, $\boldsymbol{v} = [\ v_1 \ \cdots \ v_i \ \cdots \ v_S\ ]^T$, $\boldsymbol{W} = [w_{i,j}]_{S\times S}$ is the weighting matrix of FHNN, and $\boldsymbol{b} = [b_1\ b_2 \cdots b_S]^T$. Both (18) and (19) are the state equations of FHNN.

As we know, the state equation of HNN [1]–[4] is given as $\varepsilon_i(dn_i(t)/dt) = -n_i(t) + \sum_{j=1}^{S} w_{i,j} a_j(t) + b_i(t)$, where $\boldsymbol{\varepsilon}_i = R_i c$ and c is capacitance. Comparing the state equation of FHNN with that of HNN, we can see that $d^{v_i} n_i(t)/dt^{v_i}$ appears in the former, while $dn_i(t)/dt$ exists in the latter. The former is explicitly fractional partial differential equation (FPDE), but the latter is the first-order one. Furthermore, from (1)–(4), it can be seen that since the fractional differential is nonlocal and has a weakly singular kernel, it provides an excellent method for the description of the long-term memory and the nonlocality of nonlinear dynamic processes. Therefore, an FPDE is used to describe nonlinear dynamic systems, such

as FHNN, which can be characterized by power-law nonlocality, power-law long-term memory, fractal property, and chaotic behavior, because the arbitrary order of the fractional differential represents an additional degree of freedom to fit the specific behavior. Another important characteristic is that the fractional differential depends not only on the local conditions of the evaluated time, but also on the history of the signal. This factor is often useful as FHNN has a long-term memory and any evaluation point depends on the past values of the signal. $d^{v_i}n_i(t)/dt^{v_i}$ is a nonlocal expression with respect to the nonlocality of the fractional differential. Some of the derived nonlocal voltage yields stationary charges that in turn can be converted into nonlocal conserved charges. Just as in the aforementioned discussion, $\boldsymbol{A}$ is an operational amplifier. In terms of electrical circuits, f represents the transfer function of the nonlinear amplifier $\boldsymbol{A}$ with a negligible response time. Set $f(0) = 0$. It is also convenient to define the inverse output–input relation f^{-1}. Thus it has

$$a_i(t) = f[n_i(t)]. \tag{20}$$

Thus, it follows that in vector form:

$$\boldsymbol{a}(t) = f[\boldsymbol{n}(t)]. \tag{21}$$

It is assumed that the analytic increasing function f has the inverse function f^{-1}. Thus, f^{-1} is also an analytic increasing function. Then, it follows that:

$$n_i(t) = f^{-1}[a_i(t)]. \tag{22}$$

Thus, it follows that in vector form:

$$\boldsymbol{n}(t) = f^{-1}[\boldsymbol{a}(t)]. \tag{23}$$

C. Construction of Lyapunov Function of FHNN

In this section, we construct the Lyapunov function of FHNN making use of the variable gradient method. Practical applications of FHNN heavily depend on its dynamical behaviors, such as Lyapunov stability and asymptotic stability. For the existence and uniqueness of the solution of the state equation of FHNN, it should be proved before studying its asymptotical stability; we prove the existence and uniqueness of the fractional-order equilibrium point of (18) before constructing the Lyapunov function of FHNN, respectively.

First, we prove the existence of the fractional-order equilibrium point of (18). Let us assume that the transfer function f of FHNN satisfies

$$\begin{cases} |f(n)| \le M, & n \in R \\ |f(m) - f(n)| \le K|m-n|, & m \in R \end{cases} \tag{24}$$

where M and K are two nonnegative constants and R denotes the field of real numbers. Equation (24) shows that the transfer function f of FHNN satisfies the boundedness and Lipschitz continuity. Assume further that $n^* = [n_1^* n_2^* \cdots n_S^*]^T$ is a fractional-order equilibrium point of (18), so that from (18) and (20), it follows that:

$$n_i^*(t) = \sum_{j=1}^{S} w_{i,j} f\left[n_j^*(t)\right] + b_i(t). \tag{25}$$

Thus, it follows that in vector form:

$$\boldsymbol{n}^* = \boldsymbol{W}\boldsymbol{F}(\boldsymbol{n}^*) + \boldsymbol{b} \tag{26}$$

where $\boldsymbol{F}(\boldsymbol{n}^*) = [f(n_1^*)\ f(n_2^*) \cdots f(n_S^*)]^T$. Let us assume that a mapping G satisfies

$$\boldsymbol{G}(\boldsymbol{n}) = \boldsymbol{W}\boldsymbol{F}(\boldsymbol{n}) + \boldsymbol{b} \tag{27}$$

where $\boldsymbol{n} = [n_1\ n_2 \cdots n_S]^T$. From (24), we can see that $\boldsymbol{F}(\boldsymbol{n})$ is a uniform continuous mapping from R^n to R^n. Thus, $\boldsymbol{G}(\boldsymbol{n})$ is also a uniform continuous mapping from R^n to R^n. From (24) and (27), it follows that:

$$\begin{aligned} \|\boldsymbol{G}(\boldsymbol{n})\|^2 &= \sum_{i=1}^{S}\left[\sum_{j=1}^{S} w_{i,j} f(n_j) + b_i\right]^2 \\ &\le \sum_{i=1}^{S}\left[\sum_{j=1}^{S} |w_{i,j}|M + |b_i|\right]^2 \\ &= \rho^2 \end{aligned} \tag{28}$$

where $\|\|$ denotes the Euclid norm. From (28), we can see that $\Phi = \{\boldsymbol{n} | \|\boldsymbol{n}\| \le \rho\}$ is a bounded convex set and $G(n)$ is a uniform continuous mapping from Φ to Φ. Thus, from Brouwer's fixed-point theorem, we can derive that $\exists \boldsymbol{n}^* \in \Phi$ to enable $G(n^*) = n^*$ to be set up. Thus, (25) and (26) are set up. n^* is a fractional-order equilibrium point of (18).

Second, we construct the Lyapunov function of FHNN making use of the variable gradient method. From (18)–(21), we can see that FHNN is a fractional nonlinear system. The Lyapunov function of FHNN is to solve a kind of fractional asymptotic stability. From (18), define an analytic function $\rho_i(t)$

$$\rho_i(t) = \chi_i \frac{d^{v_i} n_i(t)}{dt^{v_i}} = -n_i(t) + \sum_{j=1}^{S} w_{i,j} a_j(t) + b_i(t). \tag{29}$$

$\rho_i(t)$ is a differintegrable function [26]–[30]. According to the characteristics of fractional calculus

$$D_t^v \psi(t) = \sum_{n=0}^{\infty} \frac{D_t^n \psi(t)}{\Gamma(1+n-v)} t^{n-v} \tag{30}$$

where D_t^v is the v-order fractional differential, $\psi(t)$ is analytic function, and D_t^n is the n-order differential operator. Then, from (30) and Faà di Bruno formula [31], it follows that:

$$\begin{aligned} D_t^v \psi[\varphi\ (t)] = {} & \frac{t^{-v}}{\Gamma\ (1-v)} \psi[\varphi\ (t)] \\ & + \sum_{n=1}^{\infty} \binom{v}{n} \frac{t^{n-v} n!}{\Gamma(1+n-v)} \sum_{m=1}^{n} D_\varphi^m \psi \\ & \times \sum \prod_{k=1}^{n} \frac{1}{P_k!} \left(\frac{D_t^k \varphi(t)}{k!}\right)^{P_k} \end{aligned} \tag{31}$$

where $\psi = \psi[\varphi\ (t)]$ is the composite function, $\binom{v}{n} = ((-1)^{-n}\Gamma\ (n-v)/\Gamma\ (-v)\Gamma\ (1+n))$, D_φ^m and D_t^k are the integer-order differential operators, and P_k satisfies

$$\begin{cases} \sum\limits_{k=1}^{n} k\mathrm{P}_k = n \\ \sum\limits_{k=1}^{n} \mathrm{P}_k = m. \end{cases} \tag{32}$$

The third summation notation $\sum$ in (31) denotes the summation of the corresponding $\{\prod_{k=1}^{n}(1/\mathrm{P}_k!)[(D_t^k\varphi\ (t)/k!)]^{\mathrm{P}_k}\}|_{m=1\to n}$ of all of the combinations of $\mathrm{P}_k|_{m=1\to n}$ that satisfy the requirement of (32). From (3) and (31), we can see that the fractional differential of the composite function is equal to an infinite sum. Therefore, we suggest constructing the Lyapunov function of FHNN as follows:

$$\begin{aligned}E[a(t)] &= -\sum_{i=1}^{S} D_t^{-v_i}\left\{\left(D_{n_i}^1 f\right)\left[D_t^{1-v_i}(\rho_i(t)/\chi_i)\right]\right\}^2 \\ &= -\sum_{i=1}^{S} D_t^{-v_i}\left[D_t^1 a_i(t)\right]^2 \\ &= \frac{-t^{v_i}}{\Gamma\ (1+v_i)}\sum_{i=1}^{S}\left(D_t^1 a_i\right)^2 - \sum_{i=1}^{S}\sum_{n=1}^{\infty}\binom{-v_i}{n} \\ &\quad\times\frac{t^{n+v_i}n!}{\Gamma(1+n+v_i)}\sum_{m=1}^{n} D_{D_t^1 a_i}^m\left(D_t^1\boldsymbol{a}_i\right)^2 \\ &\quad\times\sum\prod_{k=1}^{n}\frac{1}{P_k!}\left(\frac{D_t^{k+1}a_i}{k!}\right)^{P_k} \end{aligned} \tag{33}$$

where $D_t^1 a_i(t) = (D_{n_i}^1 f)[D_t^{1-v_i}(\rho_i(t)/\chi_i)]$, $D_t^{-v_i}$ is the v_i-order fractional integral operator. From (29) and (33), we can see that to enable (33) to be set up, $D_t^{v_i}n_i(t)$ must be an inverse operation of $D_t^{-v_i}n_i(t)$. Therefore, from the combination rules of fractional calculus [26]–[30], the additional condition to guarantee the existence of Lyapunov function is $w_{i,j} = w_{j,i}$. FHNN's weighting matrix $\boldsymbol{W}$ is a symmetric matrix. For $D_{D_t^1 a_i}^1(D_t^1 a_i)^2 = 2D_t^1 a_i$, $D_{D_t^1 a_i}^2(D_t^1 a_i)^2 = 2$, and $D_{D_t^1 a_i}^m(D_t^1 a_i)^2 \overset{m\geq 3}{\equiv} 0$. Thus, (33) can be simplified as follows:

$$\begin{aligned}&E[\boldsymbol{a}(t)] \\ &= \frac{-t^{v_i}}{\Gamma\ (1+v_i)}\sum_{i=1}^{S}\left(D_t^1 a_i\right)^2 - \binom{-v_i}{1}\frac{2t^{1+v_i}}{\Gamma\ (2+v_i)} \\ &\quad\times\sum_{i=1}^{S}\left(D_t^1 a_i\right)\left(D_t^2 a_i\right) - \sum_{i=1}^{S}\sum_{n=2}^{\infty}\binom{-v_i}{n}\frac{2t^{n+v_i}n!}{\Gamma(1+n+v_i)} \\ &\quad\times\left\{D_t^1 a_i\left[\prod_{k=1}^{n}\frac{1}{P_k!}\left(\frac{D_t^{k+1}a_i}{k!}\right)^{P_k}\right]_{m=1}\right. \\ &\qquad\left. + \left[\prod_{k=1}^{n}\frac{1}{P_k!}\left(\frac{D_t^{k+1}a_i}{k!}\right)^{P_k}\right]_{m=2}\right\}. \end{aligned} \tag{34}$$

From (33) or (34), implement the v_i-order fractional differential of $E[a_i(t)]$ on both sides of (33) or (34), thus it follows that:

$$D_t^{v}E = \frac{d^{v}E}{dt^{v}} = -\sum_{i=1}^{S}\left[D_t^1 a_i(t)\right]^2 \leq 0. \tag{35}$$

Hence, we can see that $(d^{\boldsymbol{v}}E/dt^{\boldsymbol{v}})$ is a negative semidefinite function and $E(t)$ is a valid Lyapunov function of FHNN. When $D_t^{v_i}E = (d^{v_i}E/dt^{v_i}) = 0$, the system energy of FHNN is unchanging and its system has reached steady state. From (35), we can obtain an equivalent form of $D_t^{v_i}E = (d^{v_i}E/dt^{v_i}) = 0$ as follows:

$$D_t^1 a_i(t) = \frac{da_i(t)}{dt} = 0. \tag{36}$$

Thus, it follows that in vector form:

$$D_t^1\boldsymbol{a}(t) = \frac{d\boldsymbol{a}(t)}{dt} = 0. \tag{37}$$

From (35)–(37), we can determine the equilibrium points of FHNN's Lyapunov function according to LaSalle's invariance theorem and the fractional steepest descent approach [58], as the fractional steepest descent approach is different from the first-order steepest descent approach. Its every optimal searching adjustment step is on the negative direction of the Lyapunov function's fractional gradient but not of its first-order one. The equilibrium points of FHNN's Lyapunov function are the potential attractors of FHNN.

D. Analysis of Attractors of FHNN

In this section, we analyze the attractors of FHNN.

As in the aforementioned discussion, we can see that the equilibrium points of FHNN's Lyapunov function are the potential attractors of FHNN. From (30) and (33)

$$D_{a_i}^{v_i}E = \frac{d^{v_i}E}{da_i^{v_i}} = -\sum_{n=0}^{\infty}\left[\frac{a_i^{n-v_i}D_t^{n-v_i}\left[D_t^1 a_i(t)\right]^2}{\Gamma(1+n-v_i)}\right]. \tag{38}$$

Therefore, it follows that in vector form:

$$D_a^{v}E = \frac{d^{v}E}{d\boldsymbol{a}^{v}} = -\sum_{i=1}^{S}\sum_{n=0}^{\infty}\left[\frac{a_i^{n-v_i}D_t^{n-v_i}\left[D_t^1 a_i(t)\right]^2}{\Gamma(1+n-v_i)}\right] \tag{39}$$

where $\boldsymbol{v} = [\,v_1\ \cdots\ v_i\ \cdots\ v_S\,]^T$. From (36) and (38), we have $D_{a_i}^{v_i}E = (d^{v_i}E/da_i^{v_i}) = 0$, only when $D_t^1 a_i(t) = (da_i(t)/dt) = 0$. In vector form, we have $(d^{v}E/da^{v}) = 0$, when $D_t^1\boldsymbol{a}(t) = (d\boldsymbol{a}(t)/dt) = 0$. Thus, when $\boldsymbol{a}(t)$ satisfies (37), the equilibrium points of FHNN's Lyapunov function are the attractors of FHNN. Note that a significant advantage of FHNN is that its attractors essentially relate to the neuron's fractional order. FHNN possesses the fractional-order stability and the fractional-order-sensitivity characteristics.

E. Implementation of Training Algorithm for FHNN

In this section, we implement a training algorithm based on the supervised Hebb rule for FHNN.

We can implement a training algorithm based on the Hebb rule for FHNN. As we know, the Hebb rule and Storkey rule are two efficient training algorithms for HNN [59]–[61]. It is desirable for a learning rule to be both local and incremental [62]. Similar to HNN, in fact, a design procedure based on FHNN's Lyapunov function is used to determine FHNN's weighting matrix. Suppose we want to store a set of prototype patterns in an FHNN. When an input pattern is presented to an FHNN, its output should then converge to the prototype pattern closest to the input pattern. Let us assume that the prototype patterns of FHNN are $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_q, \ldots, \boldsymbol{p}_Q\}$, where the elements of the vectors $\boldsymbol{p}_q = [p_{q_1}\cdots p_{q_i}\cdots p_{q_j}\cdots p_{q_S}]^T$ are

restricted to ± 1. Assume further that $Q \ll S$, so that the state space is large enough and the prototype patterns are well distributed in this space. In order for an FHNN to be able to recall the prototype patterns, the prototype patterns must be the minima of FHNN's Lyapunov function. We propose an appropriate quadratic performance index as follows:

$$J(\boldsymbol{a}) = -\sum_{i=1}^{S} D_t^{-v_i} \left\{ \frac{\left(D_{n_i}^1 f\right)^2}{\chi_i^2} \left[D_t^{1-v_i} \left(\sum_{j=1}^{S} \sum_{q=1}^{Q} p_{q_i} p_{q_j} a_j \right) \right]^2 \right\} \tag{40}$$

where the elements of the vectors $\boldsymbol{a} = [a_1 \cdots a_j \cdots a_S]^T$ are restricted to ± 1 and $(D_{n_i}^1 f)^2/\chi_i^2$ is a positive constant. From (40), we evaluate the performance index at a random input pattern a, which is presumably not close to any prototype pattern. $\sum_{j=1}^{S} \sum_{q=1}^{Q} p_{q_i} p_{q_j} a_j$ in (40) is an inner product between a prototype pattern and the input pattern. The inner product will increase as the input pattern moves closer to a prototype pattern. However, if the input pattern is not close to any prototype pattern, all the terms of $\sum_{j=1}^{S} \sum_{q=1}^{Q} p_{q_i} p_{q_j} a_j$ in (40) will be small. Thus, $J(a)$ will be largest (least negative) when a is not close to any prototype pattern, and will be smallest (most negative) when a is equal to any one of the prototype patterns. Assume that the prototype patterns are orthogonal. We further evaluate the performance index at one of the prototype patterns as follows:

$$\begin{aligned} J(p_k) &= -\sum_{i=1}^{S} D_t^{-v_i} \left\{ \frac{\left(D_{\boldsymbol{n}_i}^1 f\right)^2}{\chi_i^2} \left[D_t^{1-v_i} \left(\sum_{j=1}^{S} \sum_{q=1}^{Q} p_{q_i} p_{q_j} p_{k_j} \right) \right]^2 \right\} \\ &= -\sum_{i=1}^{S} D_t^{-v_i} \left[\frac{\left(D_{\boldsymbol{n}_i}^1 f\right)^2}{\chi_i^2} \left(D_t^{1-v_i} S\right)^2 \right]. \end{aligned} \tag{41}$$

From the properties of the fractional calculus and (41), we can derive that $J(a)$ is minimized at the prototype patterns. We use the supervised Hebb rule to compute FHNN's weighting matrix (with target patterns being the same as input patterns) as follows:

$$\boldsymbol{W} = \sum_{q=1}^{Q} \boldsymbol{p}_q (\boldsymbol{p}_q)^T \tag{42}$$

where $\boldsymbol{W}$ is the weighting matrix of FHNN, and $w_{i,j} = \sum_{q=1}^{Q} p_{q_i} p_{q_j}$. From (20), we can see that if f is a high-gain arc-tangent function, $D_t^{1-v_i}(n_i) = D_t^{1-v_i}[f^{-1}(a_i)] = 0$. From (29), (33), and (42), and setting the bias $b_i(t)$ to zero, FHNN's high-gain Lyapunov function is as follows:

$$\begin{aligned} E(\boldsymbol{a}) &= -\sum_{i=1}^{S} D_t^{-v_i} \left(D_t^1 a_i\right)^2 \\ &= -\sum_{i=1}^{S} D_t^{-v_i} \left\{ \frac{\left(D_{n_i}^1 f\right)^2}{\chi_i^2} \left[D_t^{1-v_i} \left(\sum_{j=1}^{S} \sum_{q=1}^{Q} p_{q_i} p_{q_j} a_j \right) \right]^2 \right\} \\ &= J(\boldsymbol{a}). \end{aligned} \tag{43}$$

Thus, FHNN's high-gain Lyapunov function is indeed equal to the quadratic performance index for the content-addressable memory problem. FHNN output will tend to converge to the stored prototype patterns. In particular, from Fig. 3, (18), and (19), we can see that the diagonal elements of FHNN's weighting matrix are set to zero. From (42), since the elements of each $\boldsymbol{p}_q$ are restricted to ± 1, all of the diagonal elements of $\boldsymbol{W}$ will be equal to Q, that is, the number of prototype patterns. Thus, we can zero the diagonal by subtracting Q times the identity matrix as follows:

$$\boldsymbol{W} = \sum_{q=1}^{Q} \boldsymbol{p}_q (\boldsymbol{p}_q)^T - Q\boldsymbol{I} \tag{44}$$

where $\boldsymbol{I}$ is the identity matrix. Note that there will be at least two minima of the performance index for each prototype pattern. If $\boldsymbol{p}_q$ is a prototype pattern, then $-\boldsymbol{p}_q$ will be also in the space spanned by the prototype patterns. Therefore, each prototype pattern will be one of the corners of the hypercube $\{\boldsymbol{a} : -1 < a_j < 1\}$. These corners will include the prototype patterns, but they will also include some linear combinations of the prototype patterns. There will also be a number of other minima (spurious patterns) of the FHNN's Lyapunov function that do not correspond to the prototype patterns. We can use an improved design method [6] that is guaranteed to minimize the number of spurious patterns.

IV. Experiment and Analysis

A. Numerical Implementation of FHNN

In this section, we achieve the numerical implementation of FHNN before analyzing its stability and convergence.

First, suppose there are only two fractional neurons of FHNN. Thus, $S = 2$. From Fig. 3, we can see that the output of either FHNN's fractional neuron feeds back to the input of the other through a feedback resistor. Thus, we set $R_{1,2} = R_{2,1} = 1\ \Omega$ (ohm), $R_{1,1} = R_{2,2} = \infty\ \Omega$, and also set $R = 1\ \Omega$ and $i_1 = i_2 = 0$ on FHNN's fractional neuron (Fig. 3). Thus, it follows that $T_{1,2} = T_{2,1} = 1$ S (siemens), $T_{1,1} = T_{2,2} = 0$ S, $Z_1 = Z_2 = 1/2$, and $b_1 = b_2 = 0$. Furthermore, we set $r = 1\ \Omega$, $c = 1$ nF (nanofarad) of fractor, $F_i = F_{v_i}$ in (14) and (16). Thus, it follows that $K_i = \xi_i^{-v_i} = (r^{1-p/v_i}/c)^{v_i} = 1$, $\chi_1 = \chi_2 = Z_1 K_1 = Z_2 K_2 = 1/2$, $w_{1,2} = w_{2,1} = 1/2$, and $w_{1,1} = w_{2,2} = 0$. Therefore, we have the weighting matrix in (19) as follows:

$$\boldsymbol{W} = \begin{bmatrix} 0 & 1/2 \\ 1/2 & 0 \end{bmatrix}. \tag{45}$$

Suppose the transfer function of operational amplifier $\boldsymbol{A}$ is $f(\tau) = 2/\pi \tan^{-1}(\gamma \pi \tau/2)$. Thus, from (20) and (22), it follows, respectively that:

$$a_1(t) = \frac{2}{\pi} \tan^{-1} \left[\frac{\gamma \pi n_1(t)}{2} \right] \tag{46}$$

$$a_2(t) = \frac{2}{\pi} \tan^{-1} \left[\frac{\gamma \pi n_2(t)}{2} \right] \tag{47}$$

$$n_1(t) = \frac{2}{\gamma \pi} \tan \frac{\pi a_1(t)}{2} \tag{48}$$

$$n_2(t) = \frac{2}{\gamma \pi} \tan \frac{\pi a_2(t)}{2} \tag{49}$$

where γ is the gain coefficient of the transfer function of the operational amplifier $\boldsymbol{A}$.

From (1), when $\Delta t \to 0$, we have

$$D_t^v s(t) \cong \sum_{k=0}^{n} \frac{\Gamma(k-v)}{\Delta t^v \Gamma(-v)\Gamma(k+1)} s(t-k\Delta t) \cong \sum_{k=0}^{n} \frac{(-1)^k \Gamma(1+v)}{\Delta t^v \Gamma(k+1)\Gamma(v-k+1)} s(t-k\Delta t) \quad (50)$$

where n is a large positive integer. Equation (50) is the approximate calculation of the v-order fractional calculus when $\Delta t \to 0$ [45], i.e., fractional backward difference. Therefore, from (50), when $v = n$, it follows that:

$$D_t^n s(t) \cong \sum_{k=0}^{n} \frac{\Gamma(k-n)}{\Delta t^n \Gamma(-n)\Gamma(k+1)} s(t-k\Delta t). \quad (51)$$

Equation (51) is the approximate calculation of the n-order integer-order calculus when $\Delta t \to 0$. Thus, from (51), when $v = 1$, it becomes

$$D_t^1 s(t) \cong \frac{1}{\Delta t}[s(t) - s(t-\Delta t)]. \quad (52)$$

Equation (52) is the approximate calculation of the first-order differential when $\Delta t \to 0$, i.e., the first-order backward difference. From (50) and (52), we can see that the fractional difference has nonlocal characteristics. $D_t^v s(t)$ is not only correlated to $s(t)$ and $s(t-\Delta t)$, but also to $s(t-k\Delta t)$. It is quite different from the first-order calculus.

Then, from (18), (19), and (45)

$$\frac{d^{v_1} n_1(t)}{dt^{v_1}} = -2n_1(t) + a_2(t) \quad (53)$$

$$\frac{d^{v_2} n_2(t)}{dt^{v_2}} = -2n_2(t) + a_1(t). \quad (54)$$

Thus, from (50), (53) and (54), using the fractional forward difference, when $\Delta t \to 0$, we have, respectively

$$n_1(t+\Delta t) \cong -2n_1(t)\Delta t^{v_1} + a_2(t)\Delta t^{v_1} - \sum_{k=0}^{n} \frac{\Gamma(k-v_1+1)}{\Gamma(-v_1)\Gamma(k+2)} n_1(t-k\Delta t) \quad (55)$$

$$n_2(t+\Delta t) \cong -2n_2(t)\Delta t^{v_2} + a_1(t)\Delta t^{v_2} - \sum_{k=0}^{n} \frac{\Gamma(k-v_2+1)}{\Gamma(-v_2)\Gamma(k+2)} n_2(t-k\Delta t). \quad (56)$$

Then (55) and (56) are taken into (46) and (47), resulting in the numerical computation of $a_1(t+\Delta t)$ and $a_2(t+\Delta t)$, respectively.

Furthermore, from (33) and (50) we have the numerical computation of FHNN's Lyapunov function as follows:

$$E[\boldsymbol{a}(t)] = -\sum_{i=1}^{2} D_t^{-v_i}\left[D_t^1 a_i(t)\right]^2 \cong -\sum_{i=1}^{2}\sum_{k=0}^{n} \frac{(-1)^k \Gamma(1-v_i)\Delta t^{v_i}}{\Gamma(k+1)\Gamma(1-v_i-k)} \times \left[D_t^1 a_i(t-k\Delta t)\right]^2. \quad (57)$$

Thus, taking (46), (47), (52), (55), and (56) into (57), we can numerically compute the Lyapunov function of FHNN. In order to keep the stability and convenience of its numerical computation, we set $n = 60$ and $\Delta t = 0.0001$ in the following examples. Because it is impossible to obtain the first 60 system initial values before running FHNN, we set $n_i(0+k\Delta t) \overset{k=1\sim 59}{=} n_i(0)$. Hence, there are 59 arbitrary initial values for the system initial states of FHNN. In order to display its actual operation law, we start showing the experimental results of FHNN's Lyapunov function from the 61st calculation in the following examples.

Second, we assume there are three fractional neurons of FHNN, thus, $S = 3$. In order to avoid any of the eight corners of the hypercube being the saddle point of FHNN, we set the weighting matrix $\boldsymbol{W}$ of FHNN to be an asymmetric matrix. To keep the circuit of FHNN symmetric, we set $R_{1,2} = R_{2,1} = 1\ \Omega$, $R_{1,3} = R_{3,1} = 2\ \Omega$, $R_{2,3} = R_{3,2} = 3\ \Omega$, and $R_{1,1} = R_{2,2} = R_{3,3} = \infty\ \Omega$. We also set $R = 1\ \Omega$ and $i_1 = i_2 = i_3 = 0$ on FHNN's fractional neuron in Fig. 3. Thus, it results that $T_{1,2} = T_{2,1} = 1$ S, $T_{1,3} = T_{3,1} = 1/2$ S, $T_{2,3} = T_{3,2} = 1/3$ S, $T_{1,1} = T_{2,2} = T_{3,3} = 0$ S, and $b_1 = b_2 = b_3 = 0$. Furthermore, we also set $r = 1\ \Omega$, $c = 1$ nF of fractor, $F_i = F_{v_i}$ in (14) and (16). Thus, it has $Z_1 = 2/5$, $Z_2 = 3/7$, $Z_3 = 6/11$, $K_i = \xi_i^{-v_i} = (r^{1-p/v_i}/c)^{v_i} = 1$, $\chi_1 = Z_1 K_1 = 2/5$, $\chi_2 = Z_2 K_2 = 3/7$, $\chi_3 = Z_3 K_3 = 6/11$, $w_{1,2} = 2/5$, $w_{1,3} = 1/5$, $w_{2,1} = 3/7$, $w_{2,3} = 1/7$, $w_{3,1} = 3/11$, $w_{3,2} = 2.11$, and $w_{1,1} = w_{2,2} = w_{3,3} = 0$. Therefore, we have the weighting matrix in (19) as follows:

$$\boldsymbol{W} = \begin{bmatrix} 0 & 2/5 & 1/5 \\ 3/7 & 0 & 1/7 \\ 3/11 & 2/11 & 0 \end{bmatrix}. \quad (58)$$

Thus, from (18), (19), and (58)

$$\frac{d^{v_1} n_1(t)}{dt^{v_1}} = -\frac{5}{2} n_1(t) + a_2(t) + \frac{1}{2} a_3(t) \quad (59)$$

$$\frac{d^{v_2} n_2(t)}{dt^{v_2}} = -\frac{7}{3} n_2(t) + a_1(t) + \frac{1}{3} a_3(t) \quad (60)$$

$$\frac{d^{v_3} n_3(t)}{dt^{v_3}} = -\frac{11}{6} n_3(t) + \frac{1}{2} a_1(t) + \frac{1}{3} a_2(t). \quad (61)$$

Then, from (20), (33), and (50), we have the numerical computation of $a_1(t+\Delta t)$, $a_2(t+\Delta t)$, $a_3(t+\Delta t)$, and $E(t+\Delta t)$, respectively. Similarly, we can achieve the numerical implementation of FHNN when S is equal to any positive integer.

B. Analysis of Stability and Convergence of FHNN

In this section, we analyze the stability and convergence of FHNN. We evaluate the convergence trajectory performance of FHNN's output and its Lyapunov function, and further study its equilibrium points and attractors.

Example 1: suppose there are only two fractional neurons of FHNN, and each neuron has the same fractional order. Thus, $S = 2$ and $v_1 = v_2 = v$. We set $\gamma = 1.40$. From (57), the Lyapunov function of FHNN can be shown as in Fig. 4 when $v = 1.50$, $v = 3.50$, $v = 5.50$, and $v = 7.50$.

From (46) and (47), it can be seen that the output voltage of FHNN is limited to $\{\boldsymbol{a} : -1 \le a_i \le 1\}$ by the transfer function

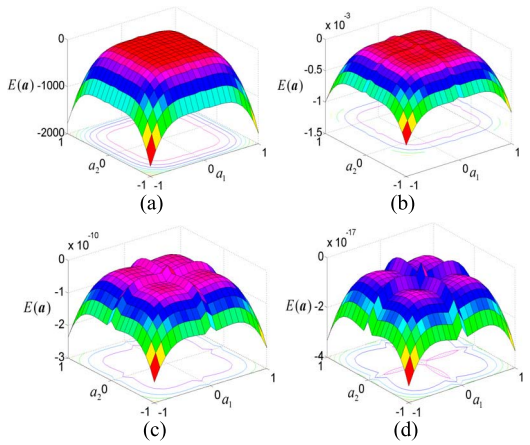

Fig. 4. Lyapunov function of FHNN ($\gamma = 1.40$). (a) $v = 1.50$. (b) $v = 3.50$. (c) $v = 5.50$. (d) $v = 7.50$.

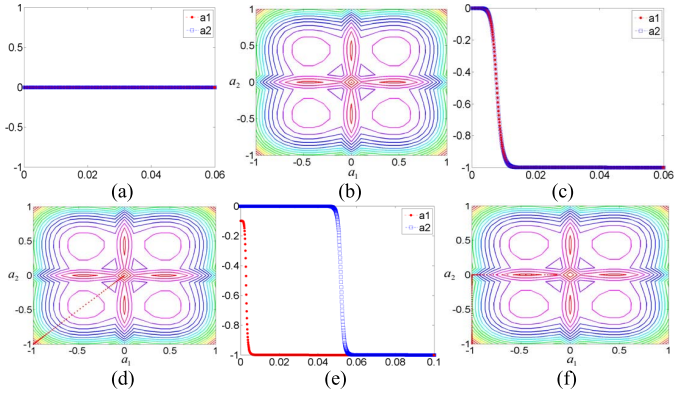

Fig. 5. Contour line of FHNN's Lyapunov function ($v = 7.50$, $\gamma = 1.40$). (a) Convergence trajectory of FHNN's output (saddle point: $n_1(0) = 0.00$, $n_2(0) = 0.00$). (b) Contour line of FHNN's Lyapunov function ($n_1(0) = 0.00$, $n_2(0) = 0.00$). (c) Convergence trajectory of FHNN's output ($n_1(0) = -0.0001$, $n_2(0) = -0.0001$). (d) Contour line of FHNN's Lyapunov function ($n_1(0) = -0.0001$, $n_2(0) = -0.0001$). (e) Convergence trajectory of FHNN's output ($n_1(0) = -0.10$, $n_2(0) = 0.00$). (f) Contour line of FHNN's Lyapunov function ($n_1(0) = -0.10$, $n_2(0) = 0.00$).

TABLE I

CORRESPONDING RELATIONSHIP BETWEEN FHNN'S FRACTIONAL ORDER AND ITS CONVERGENCE

Fractional Order v	Convergent *yes / no*	Fractional Order v	Convergent *yes / no*
$0 < v < 1$	*yes*	$4 < v < 5$	*no*
$v = 1$	*yes*	$v = 5$	*no*
$1 < v < 2$	*yes*	$5 < v < 6$	*yes*
$v = 2$	*no*	$v = 6$	*no*
$2 < v < 3$	*no*	$2k-2 < v < 2k-1$	*no*
$v = 3$	*no*	$v = 2k-1$	*no*
$3 < v < 4$	*yes*	$2k-1 < v < 2k$	*yes*
$v = 4$	*no*	$v = 2k$	*no*

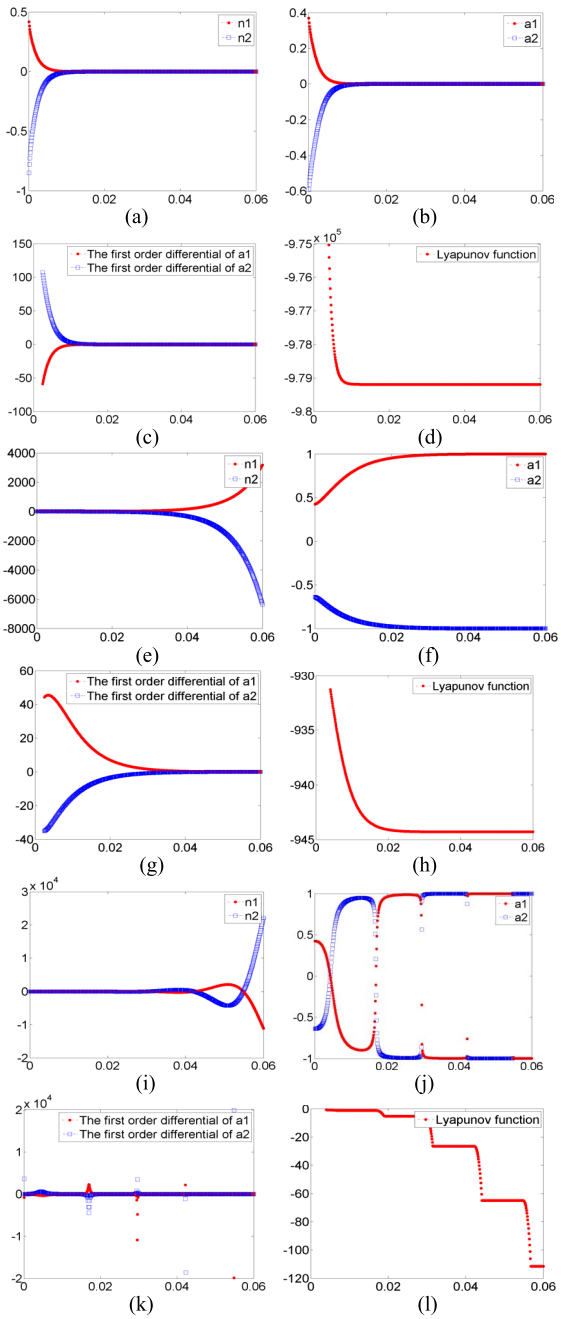

Fig. 6. Effect of fractional-order v of neuron on stability and convergence of FHNN ($\gamma = 1.40$, $n_1(0) = 0.50$, $n_2(0) = -1.00$). (a) $n_1(t)$ and $n_2(t)$ ($v = 0.50$). (b) $a_1(t)$ and $a_2(t)$ ($v = 0.50$). (c) First-order differential of $a_1(t)$ and $a_2(t)$ ($v = 0.50$). (d) Lyapunov function of FHNN ($v = 0.50$). (e) $n_1(t)$ and $n_2(t)$ ($v = 1.50$). (f) $a_1(t)$ and $a_2(t)$ ($v = 1.50$). (g) First-order differential of $a_1(t)$ and $a_2(t)$ ($v = 1.50$). (h) Lyapunov function of FHNN ($v = 1.50$). (i) $n_1(t)$ and $n_2(t)$ ($v = 2.50$). (j) $a_1(t)$ and $a_2(t)$ ($v = 2.50$). (k) First-order differential of $a_1(t)$ and $a_2(t)$ ($v = 2.50$). (l) Lyapunov function of FHNN ($v = 2.50$).

of the operational amplifier $\boldsymbol{A}$. Thus, from Fig. 4, we can see that the Lyapunov function of FHNN is limited to a minimum at any of the four corners of a hypercube. Therefore, we have the minimum of FHNN's Lyapunov function when (a_1, a_2) is equal to $(-1, -1)$, $(-1, 1)$, $(1, -1)$, and $(1, 1)$, respectively. From Fig. 4, we can also see that $(0, 0)$ is the saddle point of FHNN's Lyapunov function, and $a_1 = 0$ or $a_2 = 0$ is the ridge of its Lyapunov function. The ridge of FHNN's Lyapunov function is more pronounced when the fractional-order v of FHNN's neuron is greater.

In the above example, we select $v = 7.50$ and $\gamma = 1.40$. The contour line of FHNN's Lyapunov function can be shown as in Fig. 5.

From Fig. 5(a) and (b), we can see that the output of FHNN (a_1, a_2) is identically equal to $(0, 0)$ when the input

TABLE II

CORRESPONDING RELATIONSHIP BETWEEN FHNN'S INPUT AND ITS OUTPUT. (a) $0.00 < v < 1.00$, $\gamma = 1.40$. (b) $1 < v < 2$, $\gamma = 1.40$. (c) $3 < v < 4$, $\gamma = 1.40$. (d) $2k-1 < v < 2k$, $\gamma = 1.40$

(a)

Input n_1, n_2	Output a_1, a_2	Input n_1, n_2	Output a_1, a_2
−1.0, −1.0	0.0, 0.0	0.5, 0.0	0.0, 0.0
−0.5, −1.0	0.0, 0.0	1.0, 0.0	0.0, 0.0
0.0, −1.0	0.0, 0.0	−1.0, 0.5	0.0, 0.0
0.5, −1.0	0.0, 0.0	−0.5, 0.5	0.0, 0.0
1.0, −1.0	0.0, 0.0	0.0, 0.5	0.0, 0.0
−1.0, −0.5	0.0, 0.0	0.5, 0.5	0.0, 0.0
−0.5, −0.5	0.0, 0.0	1.0, 0.5	0.0, 0.0
0.0, −0.5	0.0, 0.0	−1.0, 1.0	0.0, 0.0
0.5, −0.5	0.0, 0.0	−0.5, 1.0	0.0, 0.0
1.0, −0.5	0.0, 0.0	0.0, 1.0	0.0, 0.0
−1.0, 0.0	0.0, 0.0	0.5, 1.0	0.0, 0.0
−0.5, 0.0	0.0, 0.0	1.0, 1.0	0.0, 0.0

(b)

Input n_1, n_2	Output a_1, a_2	Input n_1, n_2	Output a_1, a_2
−1.0, −1.0	−1.0, −1.0	0.5, 0.0	1.0, 1.0
−0.5, −1.0	−1.0, −1.0	1.0, 0.0	1.0, 1.0
0.0, −1.0	−1.0, −1.0	−1.0, 0.5	−1.0, 1.0
0.5, −1.0	1.0, −1.0	−0.5, 0.5	−1.0, 1.0
1.0, −1.0	1.0, −1.0	0.0, 0.5	1.0, 1.0
−1.0, −0.5	−1.0, −1.0	0.5, 0.5	1.0, 1.0
−0.5, −0.5	−1.0, −1.0	1.0, 0.5	1.0, 1.0
0.0, −0.5	−1.0, −1.0	−1.0, 1.0	−1.0, 1.0
0.5, −0.5	1.0, −1.0	−0.5, 1.0	−1.0, 1.0
1.0, −0.5	1.0, −1.0	0.0, 1.0	1.0, 1.0
−1.0, 0.0	−1.0, −1.0	0.5, 1.0	1.0, 1.0
−0.5, 0.0	−1.0, −1.0	1.0, 1.0	1.0, 1.0

(c)

Input n_1, n_2	Output a_1, a_2	Input n_1, n_2	Output a_1, a_2
−1.0, −1.0	−1.0, −1.0	0.5, 0.0	1.0, 1.0
−0.5, −1.0	−1.0, −1.0	1.0, 0.0	1.0, 1.0
0.0, −1.0	−1.0, −1.0	−1.0, 0.5	−1.0, 1.0
0.5, −1.0	1.0, −1.0	−0.5, 0.5	−1.0, 1.0
1.0, −1.0	1.0, −1.0	0.0, 0.5	1.0, 1.0
−1.0, −0.5	−1.0, −1.0	0.5, 0.5	1.0, 1.0
−0.5, −0.5	−1.0, −1.0	1.0, 0.5	1.0, 1.0
0.0, −0.5	−1.0, −1.0	−1.0, 1.0	−1.0, 1.0
0.5, −0.5	1.0, −1.0	−0.5, 1.0	−1.0, 1.0
1.0, −0.5	1.0, −1.0	0.0, 1.0	1.0, 1.0
−1.0, 0.0	−1.0, −1.0	0.5, 1.0	1.0, 1.0
−0.5, 0.0	−1.0, −1.0	1.0, 1.0	1.0, 1.0

(d)

Input n_1, n_2	Output a_1, a_2	Input n_1, n_2	Output a_1, a_2
−1.0, −1.0	−1.0, −1.0	0.5, 0.0	1.0, 1.0
−0.5, −1.0	−1.0, −1.0	1.0, 0.0	1.0, 1.0
0.0, −1.0	−1.0, −1.0	−1.0, 0.5	−1.0, 1.0
0.5, −1.0	1.0, −1.0	−0.5, 0.5	−1.0, 1.0
1.0, −1.0	1.0, −1.0	0.0, 0.5	1.0, 1.0
−1.0, −0.5	−1.0, −1.0	0.5, 0.5	1.0, 1.0
−0.5, −0.5	−1.0, −1.0	1.0, 0.5	1.0, 1.0
0.0, −0.5	−1.0, −1.0	−1.0, 1.0	−1.0, 1.0
0.5, −0.5	1.0, −1.0	−0.5, 1.0	−1.0, 1.0
1.0, −0.5	1.0, −1.0	0.0, 1.0	1.0, 1.0
−1.0, 0.0	−1.0, −1.0	0.5, 1.0	1.0, 1.0
−0.5, 0.0	−1.0, −1.0	1.0, 1.0	1.0, 1.0

of FHNN (n_1, n_2) is equal to $(0, 0)$. Hence, $(0, 0)$ is the saddle point of the Lyapunov function of FHNN. In addition, from Fig. 5(c) and (d), we can also see that when (n_1, n_2) is equal to $(-0.0001, -0.0001)$, which is very close to the saddle point $(0, 0)$, (a_1, a_2) converges to one of the equilibrium points $(-1, -1)$ of FHNN. From the discussion in (35)–(37) mentioned above, we determine the equilibrium points of FHNN's Lyapunov function according to LaSalle's invariance theorem and the fractional steepest descent approach [58]. Each of its optimal searching adjustment step is in the negative direction of Lyapunov function's fractional gradient but not of its first-order one. Thus, its convergence trajectory easily passes through the first-order local minimum and the maximum points of FHNN's Lyapunov function. Furthermore, from Fig. 5(e) and (f), we can further see that when (n_1, n_2) is equal to $(-0.10, 0.00)$, which is on the ridge of FHNN's Lyapunov function, the convergence trajectory of (a_1, a_2) passes along the ridge of FHNN's Lyapunov function $a_2 = 0$ first, and then passes along the boundary of the hypercube $a_1 = 0$. It finally converges to one of the equilibrium points of FHNN, i.e., $(-1, -1)$.

In order to analyze the effect of FHNN's fractional-order v on its stability and convergence, we set the fractional order of the neuron to $v = 0.50$, $v = 1.50$, and $v = 2.50$, respectively. Its convergence trajectory performance can be as shown in Fig. 6.

From Fig. 6(a)–(d), it can be seen that when $v = 0.50$, $n_1(t)$ and $n_2(t)$ converge to zero gradually. Meanwhile, $a_1(t)$, $a_2(t)$ and their first-order differential converge to zero correspondingly. From Fig. 6(e)–(h), it can also be seen that when $v = 1.50$, $n_1(t)$ and $n_2(t)$ nonlinearly amplify step-by-step. Consequently, $a_1(t)$ and $a_2(t)$ are limited to $\{\boldsymbol{a} : -1 \le a_i \le 1\}$ by the transfer function of the operational amplifier $\boldsymbol{A}$. Therefore, (a_1, a_2) converges to $(1, -1)$, and their first-order differential converges to zero accordingly. From Fig. 6(i)–(l), it can be further seen that when $v = 2.50$, $n_1(t)$ and $n_2(t)$ nonlinearly amplify gradually, while at the same time, alternating between positive and negative. Affected by the crossfade alternation of $n_1(t)$ and $n_2(t)$, $a_1(t)$ and $a_2(t)$ periodically alternate between 1 and -1, respectively. Similarly, we can summarize the corresponding relationship between FHNN's fractional order and its convergence using mathematical induction. It can be shown as in Table I.

From Table I, we can see that FHNN is convergent when $0.00 < v < 1.00$, $v = 1.00$, and $2k-1 < v < 2k$, where k is a positive integer. Particularly, FHNN is the classical first-order HNN when $v = 1.00$. HNN is a special case of FHNN.

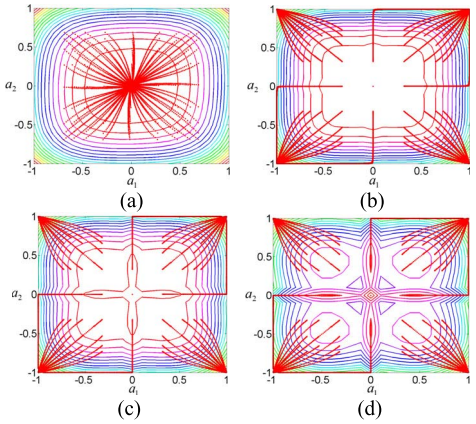

Fig. 7. Convergence trajectory of FHNN in 2-D space. (a) $0.00 < v < 1.00$, $\gamma = 1.40$. (b) $1.00 < v < 2.00$, $\gamma = 1.40$. (c) $3.00 < v < 4.00$, $\gamma = 1.40$. (d) $2k-1 < v < 2k$, $\gamma = 1.40$.

Based on the aforementioned discussion, the output of FHNN is limited to $\{\boldsymbol{a} : -1 \le a_i \le 1\}$ by the transfer function of the operational amplifier $\boldsymbol{A}$. Hence, we focus on discussing the case in which the input voltage of FHNN is limited to $\{\boldsymbol{n} : -1 \le n_i \le 1\}$. For the case of $\{\boldsymbol{n} : -1 \le n_i \le -1\}$ or $\{\boldsymbol{n} : 1 < n_i\}$ the convergence trajectory performance of FHNN is similar. Then, from (46), (47), and (55)–(57), we have the convergence trajectory performance of FHNN. Then, we have the corresponding relationship between FHNN's input and its output. It can be shown as in Table II.

From Table II, we can see that corresponding to various inputs (n_1, n_2), output (a_1, a_2) converges to $(0, 0)$ when $0.00 < v < 1.00$. Furthermore, output (a_1, a_2) converges to $(-1, -1)$, $(-1, 1)$, $(1, -1)$ or, $(1, 1)$ when $1.00 < v < 2.00$, $3.00 < v < 4.00$ and $2k-1 < v < 2k$, where k is a positive integer. Hence, $(-1, -1)$, $(-1, 1)$, $(1, -1)$, and $(1, 1)$ are four attractors of FHNN when $1.00 < v < 2.00$, $3.00 < v < 4.00$ and $2k-1 < v < 2k$. Furthermore, it should be noted that FHNN is the classical first-order HNN when $v = 1.00$. HNN is a special case of FHNN. As we all know, the convergence rule of HNN is related to the operational amplifier's gain coefficient γ and weighting matrix $\boldsymbol{W}$ of HNN [1]–[4]. To avoid repetition, its convergence rule is not described in this paper. Then, the convergence trajectory of FHNN in 2-D space can be as shown in Fig. 7.

TABLE III

CORRESPONDING RELATIONSHIP BETWEEN FHNN'S INPUT AND ITS OUTPUT ($S = 3$, $2k - 1 < v < 2k$, $\gamma = 1.40$)

Input n_1, n_2, n_3	Output a_1, a_2, a_3	Input n_1, n_2, n_3	Output a_1, a_2, a_3
−1.0, −1.0, −1.0	−1.0, −1.0, −1.0	1.0, 0.0, 0.0	1.0, 1.0, 1.0
0.0, −1.0, −1.0	−1.0, −1.0, −1.0	−1.0, 1.0, 0.0	−1.0, 1.0, −1.0
1.0, −1.0, −1.0	1.0, −1.0, −1.0	0.0, 1.0, 0.0	1.0, 1.0, 1.0
−1.0, 0.0, −1.0	−1.0, −1.0, −1.0	1.0, 1.0, 0.0	1.0, 1.0, 1.0
0.0, 0.0, −1.0	−1.0, −1.0, −1.0	−1.0, −1.0, 1.0	−1.0, −1.0, 1.0
1.0, 0.0, −1.0	1.0, 1.0, −1.0	0.0, −1.0, 1.0	−1.0, −1.0, 1.0
−1.0, 1.0, −1.0	−1.0, 1.0, −1.0	1.0, −1.0, 1.0	1.0, −1.0, 1.0
0.0, 1.0, −1.0	1.0, 1.0, −1.0	−1.0, 0.0, 1.0	−1.0, −1.0, 1.0
1.0, 1.0, −1.0	1.0, 1.0, −1.0	0.0, 0.0, 1.0	1.0, 1.0, 1.0
−1.0, −1.0, 0.0	−1.0, −1.0, −1.0	1.0, 0.0, 1.0	1.0, 1.0, 1.0
0.0, −1.0, 0.0	−1.0, −1.0, −1.0	−1.0, 1.0, 1.0	−1.0, 1.0, 1.0
1.0, −1.0, 0.0	1.0, −1.0, 1.0	0.0, 1.0, 1.0	1.0, 1.0, 1.0
−1.0, 0.0, 0.0	−1.0, −1.0, −1.0	1.0, 1.0, 1.0	1.0, 1.0, 1.0

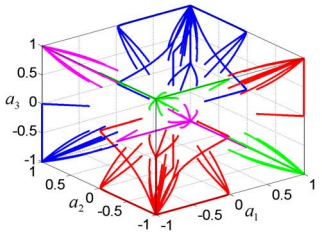

Fig. 8. Convergence trajectory of FHNN in 3-D space ($S = 3$, $2k - 1 < v < 2k$, $\gamma = 1.40$).

From Fig. 7, we can see that, on the one hand, $(0, 0)$ is the single attractor of FHNN when $0.00 < v < 1.00$. On the other hand, $(-1, -1)$, $(-1, 1)$, $(1, -1)$, and $(1, 1)$ are four different equilibrium points or attractors, and $(0, 0)$ is the saddle point of FHNN's Lyapunov function when $1.00 < v < 2.00$, $3.00 < v < 4.00$, and $2k - 1 < v < 2k$. Meanwhile, FHNN's fractional-order v can also affect the rate of FHNN's convergence. Furthermore, the convergence trajectory of FHNN easily passes through the first-order local minimum and maximum points of FHNN's Lyapunov function. When the input is on the ridge of FHNN's Lyapunov function, the convergence trajectory of the output passes along the ridge of FHNN's Lyapunov function first, and then passes along the boundary of the hypercube. It finally converges to one of FHNN's equilibrium points or attractors. Compared with the classical first-order HNN, it is known that, in general, a double-neuron HNN usually associatively memorizes two characteristics [1]–[4]. Hence, we can further see that FHNN has a stronger associative memory than HNN. As mentioned earlier, the double-neuron FHNN can associatively memorize four characteristics at most.

Example 2: suppose there are three fractional neurons of FHNN, and each of its neuron has the same fractional order. Thus, $S = 3$ and $v_1 = v_2 = v_3 = v$. We also set $\gamma = 1.40$. First, suppose that the weighting matrix $\boldsymbol{W}$ of FHNN is a symmetric matrix. From (58)–(61), the corresponding relationship between FHNN's input and its output can be as shown in Table III.

From Table III, we can see that corresponding to various inputs (n_1, n_2, n_3), the output (a_1, a_2, a_3) converges to $(-1, -1, -1)$, $(1, -1, -1)$, $(-1, 1, -1)$, $(1, 1, -1)$, $(-1, -1, 1)$, $(1, -1, 1)$, $(-1, 1, 1)$, or $(1, 1, 1)$ when $2k - 1 < v < 2k$, where k is a positive integer. Then, the convergence trajectory of FHNN in 3-D space can be as shown in Fig. 8.

TABLE IV

CORRESPONDING RELATIONSHIP BETWEEN FHNN'S INPUT AND ITS OUTPUT ($\gamma = 1.40$). (a) $v_1 = 0.50$, $v_2 = 1.50$. (b) $v_1 = 1.50$, $v_2 = 0.50$

(a)

Input n_1, n_2	Output a_1, a_2	Input n_1, n_2	Output a_1, a_2
−1.0, −1.0	−0.0685, −1.0	0.5, 0.0	0.0685, 1.0
−0.5, −1.0	−0.0685, −1.0	1.0, 0.0	0.0685, 1.0
0.0, −1.0	−0.0685, −1.0	−1.0, 0.5	0.0685, 1.0
0.5, −1.0	−0.0685, −1.0	−0.5, 0.5	0.0685, 1.0
1.0, −1.0	−0.0685, −1.0	0.0, 0.5	0.0685, 1.0
−1.0, −0.5	−0.0685, −1.0	0.5, 0.5	0.0685, 1.0
−0.5, −0.5	−0.0685, −1.0	1.0, 0.5	0.0685, 1.0
0.0, −0.5	−0.0685, −1.0	−1.0, 1.0	0.0685, 1.0
0.5, −0.5	−0.0685, −1.0	−0.5, 1.0	0.0685, 1.0
1.0, −0.5	−0.0685, −1.0	0.0, 1.0	0.0685, 1.0
−1.0, 0.0	−0.0685, −1.0	0.5, 1.0	0.0685, 1.0
−0.5, 0.0	−0.0685, −1.0	1.0, 1.0	0.0685, 1.0

(b)

Input n_1, n_2	Output a_1, a_2	Input n_1, n_2	Output a_1, a_2
−1.0, −1.0	−1.0, −0.0685	0.5, 0.0	1.0, 0.0685
−0.5, −1.0	−1.0, −0.0685	1.0, 0.0	1.0, 0.0685
0.0, −1.0	−1.0, −0.0685	−1.0, 0.5	−1.0, −0.0685
0.5, −1.0	1.0, 0.0685	−0.5, 0.5	−1.0, −0.0685
1.0, −1.0	1.0, 0.0685	0.0, 0.5	1.0, 0.0685
−1.0, −0.5	−1.0, −0.0685	0.5, 0.5	1.0, 0.0685
−0.5, −0.5	−1.0, −0.0685	1.0, 0.5	1.0, 0.0685
0.0, −0.5	−1.0, −0.0685	−1.0, 1.0	−1.0, −0.0685
0.5, −0.5	1.0, 0.0685	−0.5, 1.0	−1.0, −0.0685
1.0, −0.5	1.0, 0.0685	0.0, 1.0	1.0, 0.0685
−1.0, 0.0	−1.0, −0.0685	0.5, 1.0	1.0, 0.0685
−0.5, 0.0	−1.0, −0.0685	1.0, 1.0	1.0, 0.0685

From Fig. 8, we can see that $(-1, -1, -1)$, $(1, -1, -1)$, $(-1, 1, -1)$, $(1, 1, -1)$, $(-1, -1, 1)$, $(1, -1, 1)$, $(-1, 1, 1)$, and $(1, 1, 1)$ are eight different equilibrium points or attractors, and $(0, 0, 0)$ is the saddle point of FHNN's Lyapunov function when $2k - 1 < v < 2k$, where k is positive integer. In Fig. 8, we use different colored (red, blue, green, and purple) convergence trajectories to demonstrate eight domains of attraction corresponding to the aforementioned eight attractors of FHNN, respectively. Compared with the classical first-order HNN, it is known that in general, the trio-neuron HNN usually associatively memorizes two characteristics [1]–[4]. Hence, we can further see that FHNN has a stronger associative memory than HNN. Based on the aforementioned discussion, the trio-neuron FHNN can associatively memorize eight characteristics at most.

Example 3: suppose there are two fractional neurons of FHNN, but each neuron has a different fractional order. Thus, $S = 2$ and $v_1 \neq v_2$. We set $v_1 = 0.50$, $v_2 = 1.50$ and $v_1 = 1.50$, $v_2 = 0.50$, respectively. We also set $\lambda = 1.40$. Then, from (46), (47), and (55)–(57), the corresponding relationship between the input and output of FHNN can be as shown in Table IV.

From Table IV, we can see that on the one hand, corresponding to various inputs (n_1, n_2), the output (a_1, a_2) converges to $(-0.0685, -1.0)$ or $(0.0685, 1.0)$ when $v_1 = 0.50$ and $v_2 = 1.50$. On the other hand, corresponding to various inputs (n_1, n_2), the output (a_1, a_2) converges to $(-1.0, -0.0685)$ or $(1.0, 0.0685)$, when $v_1 = 1.50$ and $v_2 = 0.50$, respectively. Thus, it can be seen that the coordinates of FHNN's equilibrium points or attractors have a $\pi/2$ counterclockwise rotation while the values of v_1 and v_2 have been exchanged. Then, FHNN's Lyapunov function and convergence trajectory of FHNN's output are as shown in Fig. 9.

From Fig. 9, it can be seen that, on the one hand, $(-0.0685, -1.0)$ and $(0.0685, 1.0)$ are two different equilibrium points of FHNN's Lyapunov function or attractors of FHNN when $v_1 = 0.50$ and $v_2 = 1.50$. We have the global maximum of FHNN's Lyapunov function at the points of $(-0.0685, -1.0)$ and $(0.0685, 1.0)$. On the other hand, $(-1.0, -0.0685)$ and $(1.0, 0.0685)$are two different attractors of FHNN when $v_1 = 1.50$ and $v_2 = 0.50$. We have the global maximum of FHNN's Lyapunov function at the points of $(-1.0, -0.0685)$ and $(1.0, 0.0685)$. In either case, $(0, 0)$ is

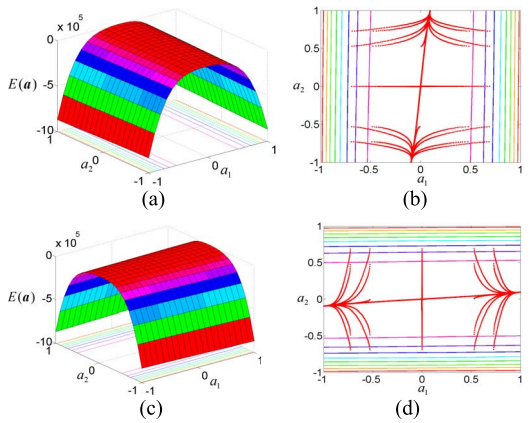

Fig. 9. FHNN's Lyapunov function and convergence trajectory of FHNN's output ($\gamma = 1.40$). (a) Lyapunov function ($v_1 = 0.50$, $v_2 = 1.50$). (b) Convergence trajectory ($v_1 = 0.50$, $v_2 = 1.50$). (c) Lyapunov function ($v_1 = 1.50$, $v_2 = 0.50$). (d) Convergence trajectory ($v_1 = 1.50$, $v_2 = 0.50$).

TABLE V

CORRESPONDING RELATIONSHIP BETWEEN INPUT AND OUTPUT OF FHNN ($\gamma = 1.40$). (a) $v_1 = 0.25$, $v_2 = 1.50$. (b) $v_1 = 0.75$, $v_2 = 1.50$

(a)

Input n_1, n_2	Output a_1, a_2	Input n_1, n_2	Output a_1, a_2
−1.0, −1.0	−0.1672, −1.0	0.5, 0.0	0.1672, 1.0
−0.5, −1.0	−0.1672, −1.0	1.0, 0.0	0.1672, 1.0
0.0, −1.0	−0.1672, −1.0	−1.0, 0.5	0.1672, 1.0
0.5, −1.0	−0.1672, −1.0	−0.5, 0.5	0.1672, 1.0
1.0, −1.0	−0.1672, −1.0	0.0, 0.5	0.1672, 1.0
−1.0, −0.5	−0.1672, −1.0	0.5, 0.5	0.1672, 1.0
−0.5, −0.5	−0.1672, −1.0	1.0, 0.5	0.1672, 1.0
0.0, −0.5	−0.1672, −1.0	−1.0, 1.0	0.1672, 1.0
0.5, −0.5	−0.1672, −1.0	−0.5, 1.0	0.1672, 1.0
1.0, −0.5	−0.1672, −1.0	0.0, 1.0	0.1672, 1.0
−1.0, 0.0	−0.1672, −1.0	0.5, 1.0	0.1672, 1.0
−0.5, 0.0	−0.1672, −1.0	1.0, 1.0	0.1672, 1.0

(b)

Input n_1, n_2	Output a_1, a_2	Input n_1, n_2	Output a_1, a_2
−1.0, −1.0	−0.0322, −1.0	0.5, 0.0	0.0322, 1.0
−0.5, −1.0	−0.0322, −1.0	1.0, 0.0	0.0322, 1.0
0.0, −1.0	−0.0322, −1.0	−1.0, 0.5	0.0322, 1.0
0.5, −1.0	−0.0322, −1.0	−0.5, 0.5	0.0322, 1.0
1.0, −1.0	−0.0322, −1.0	0.0, 0.5	0.0322, 1.0
−1.0, −0.5	−0.0322, −1.0	0.5, 0.5	0.0322, 1.0
−0.5, −0.5	−0.0322, −1.0	1.0, 0.5	0.0322, 1.0
0.0, −0.5	−0.0322, −1.0	−1.0, 1.0	0.0322, 1.0
0.5, −0.5	−0.0322, −1.0	−0.5, 1.0	0.0322, 1.0
1.0, −0.5	−0.0322, −1.0	0.0, 1.0	0.0322, 1.0
−1.0, 0.0	−0.0322, −1.0	0.5, 1.0	0.0322, 1.0
−0.5, 0.0	−0.0322, −1.0	1.0, 1.0	0.0322, 1.0

TABLE VI

CORRESPONDING RELATIONSHIP BETWEEN INPUT AND OUTPUT OF FHNN ($S = 3$, $\gamma = 1.40$). (a) $v_1 = 0.25$, $v_2 = 0.75$, $v_3 = 1.50$. (b) $v_1 = 0.25$, $v_2 = 1.50$, $v_3 = 0.75$. (c) $v_1 = 0.75$, $v_2 = 0.25$, $v_3 = 1.50$. (d) $v_1 = 0.75$, $v_2 = 1.50$, $v_3 = 0.25$. (e) $v_1 = 1.50$, $v_2 = 0.25$, $v_3 = 0.75$. (f) $v_1 = 1.50$, $v_2 = 0.75$, $v_3 = 0.25$

(c)

Input n_1, n_2, n_3	Output a_1, a_2, a_3	Input n_1, n_2, n_3	Output a_1, a_2, a_3
−1.0, −1.0, −1.0	−0.0805, −0.0132, −1.0	1.0, 0.0, 0.0	0.0805, 0.0132, 1.0
0.0, −1.0, −1.0	−0.0805, −0.0132, −1.0	−1.0, 1.0, 0.0	0.0805, 0.0132, 1.0
1.0, −1.0, −1.0	−0.0805, −0.0132, −1.0	0.0, 1.0, 0.0	0.0805, 0.0132, 1.0
−1.0, 0.0, −1.0	−0.0805, −0.0132, −1.0	1.0, 1.0, 0.0	0.0805, 0.0132, 1.0
0.0, 0.0, −1.0	−0.0805, −0.0132, −1.0	−1.0, −1.0, 1.0	0.0805, 0.0132, 1.0
1.0, 0.0, −1.0	−0.0805, −0.0132, −1.0	0.0, −1.0, 1.0	0.0805, 0.0132, 1.0
−1.0, 1.0, −1.0	−0.0805, −0.0132, −1.0	1.0, −1.0, 1.0	0.0805, 0.0132, 1.0
0.0, 1.0, −1.0	−0.0805, −0.0132, −1.0	−1.0, 0.0, 1.0	0.0805, 0.0132, 1.0
1.0, 1.0, −1.0	−0.0805, −0.0132, −1.0	0.0, 0.0, 1.0	0.0805, 0.0132, 1.0
−1.0, −1.0, 0.0	−0.0805, −0.0132, −1.0	1.0, 0.0, 1.0	0.0805, 0.0132, 1.0
0.0, −1.0, 0.0	−0.0805, −0.0132, −1.0	−1.0, 1.0, 1.0	0.0805, 0.0132, 1.0
1.0, −1.0, 0.0	−0.0805, −0.0132, −1.0	0.0, 1.0, 1.0	0.0805, 0.0132, 1.0
−1.0, 0.0, 0.0	−0.0805, −0.0132, −1.0	1.0, 1.0, 1.0	0.0805, 0.0132, 1.0

(d)

Input n_1, n_2, n_3	Output a_1, a_2, a_3	Input n_1, n_2, n_3	Output a_1, a_2, a_3
−1.0, −1.0, −1.0	−0.1556, −1.0, −0.0133	1.0, 0.0, 0.0	0.1556, 1.0, 0.0133
0.0, −1.0, −1.0	−0.1556, −1.0, −0.0133	−1.0, 1.0, 0.0	0.1556, 1.0, 0.0133
1.0, −1.0, −1.0	−0.1556, −1.0, −0.0133	0.0, 1.0, 0.0	0.1556, 1.0, 0.0133
−1.0, 0.0, −1.0	−0.1556, −1.0, −0.0133	1.0, 1.0, 0.0	0.1556, 1.0, 0.0133
0.0, 0.0, −1.0	−0.1556, −1.0, −0.0133	−1.0, −1.0, 1.0	−0.1556, −1.0, −0.0133
1.0, 0.0, −1.0	−0.1556, −1.0, −0.0133	0.0, −1.0, 1.0	−0.1556, −1.0, −0.0133
−1.0, 1.0, −1.0	0.1556, 1.0, 0.0133	1.0, −1.0, 1.0	−0.1556, −1.0, −0.0133
0.0, 1.0, −1.0	0.1556, 1.0, 0.0133	−1.0, 0.0, 1.0	0.1556, 1.0, 0.0133
1.0, 1.0, −1.0	0.1556, 1.0, 0.0133	0.0, 0.0, 1.0	0.1556, 1.0, 0.0133
−1.0, −1.0, 0.0	−0.1556, −1.0, −0.0133	1.0, 0.0, 1.0	0.1556, 1.0, 0.0133
0.0, −1.0, 0.0	−0.1556, −1.0, −0.0133	−1.0, 1.0, 1.0	0.1556, 1.0, 0.0133
1.0, −1.0, 0.0	−0.1556, −1.0, −0.0133	0.0, 1.0, 1.0	0.1556, 1.0, 0.0133
−1.0, 0.0, 0.0	−0.1556, −1.0, −0.0133	1.0, 1.0, 1.0	0.1556, 1.0, 0.0133

(e)

Input n_1, n_2, n_3	Output a_1, a_2, a_3	Input n_1, n_2, n_3	Output a_1, a_2, a_3
−1.0, −1.0, −1.0	−0.0177, −0.0567, −1.0	1.0, 0.0, 0.0	0.0177, 0.0567, 1.0
0.0, −1.0, −1.0	−0.0177, −0.0567, −1.0	−1.0, 1.0, 0.0	−0.0177, −0.0567, −1.0
1.0, −1.0, −1.0	−0.0177, −0.0567, −1.0	0.0, 1.0, 0.0	0.0177, 0.0567, 1.0
−1.0, 0.0, −1.0	−0.0177, −0.0567, −1.0	1.0, 1.0, 0.0	0.0177, 0.0567, 1.0
0.0, 0.0, −1.0	−0.0177, −0.0567, −1.0	−1.0, −1.0, 1.0	0.0177, 0.0567, 1.0
1.0, 0.0, −1.0	−0.0177, −0.0567, −1.0	0.0, −1.0, 1.0	0.0177, 0.0567, 1.0
−1.0, 1.0, −1.0	−0.0177, −0.0567, −1.0	1.0, −1.0, 1.0	0.0177, 0.0567, 1.0
0.0, 1.0, −1.0	−0.0177, −0.0567, −1.0	−1.0, 0.0, 1.0	0.0177, 0.0567, 1.0
1.0, 1.0, −1.0	−0.0177, −0.0567, −1.0	0.0, 0.0, 1.0	0.0177, 0.0567, 1.0
−1.0, −1.0, 0.0	−0.0177, −0.0567, −1.0	1.0, 0.0, 1.0	0.0177, 0.0567, 1.0
0.0, −1.0, 0.0	−0.0177, −0.0567, −1.0	−1.0, 1.0, 1.0	0.0177, 0.0567, 1.0
1.0, −1.0, 0.0	0.0177, 0.0567, 1.0	0.0, 1.0, 1.0	0.0177, 0.0567, 1.0
−1.0, 0.0, 0.0	−0.0177, −0.0567, −1.0	1.0, 1.0, 1.0	0.0177, 0.0567, 1.0

(f)

Input n_1, n_2, n_3	Output a_1, a_2, a_3	Input n_1, n_2, n_3	Output a_1, a_2, a_3
−1.0, −1.0, −1.0	−0.0327, −1.0, −0.0614	1.0, 0.0, 0.0	0.0327, 1.0, 0.0614
0.0, −1.0, −1.0	−0.0327, −1.0, −0.0614	−1.0, 1.0, 0.0	0.0327, 1.0, 0.0614
1.0, −1.0, −1.0	−0.0327, −1.0, −0.0614	0.0, 1.0, 0.0	0.0327, 1.0, 0.0614
−1.0, 0.0, −1.0	−0.0327, −1.0, −0.0614	1.0, 1.0, 0.0	0.0327, 1.0, 0.0614
0.0, 0.0, −1.0	−0.0327, −1.0, −0.0614	−1.0, −1.0, 1.0	−0.0327, −1.0, −0.0614
1.0, 0.0, −1.0	0.0327, 1.0, 0.0614	0.0, −1.0, 1.0	−0.0327, −1.0, −0.0614
−1.0, 1.0, −1.0	0.0327, 1.0, 0.0614	1.0, −1.0, 1.0	−0.0327, −1.0, −0.0614
0.0, 1.0, −1.0	0.0327, 1.0, 0.0614	−1.0, 0.0, 1.0	−0.0327, −1.0, −0.0614
1.0, 1.0, −1.0	0.0327, 1.0, 0.0614	0.0, 0.0, 1.0	0.0327, 1.0, 0.0614
−1.0, −1.0, 0.0	−0.0327, −1.0, −0.0614	1.0, 0.0, 1.0	0.0327, 1.0, 0.0614
0.0, −1.0, 0.0	−0.0327, −1.0, −0.0614	−1.0, 1.0, 1.0	0.0327, 1.0, 0.0614
1.0, −1.0, 0.0	−0.0327, −1.0, −0.0614	0.0, 1.0, 1.0	0.0327, 1.0, 0.0614
−1.0, 0.0, 0.0	−0.0327, −1.0, −0.0614	1.0, 1.0, 1.0	0.0327, 1.0, 0.0614

the saddle point of FHNN. Comparing Figs. 7 and 9, we can further see that there are only two, but not four, attractors of FHNN whenever its two neurons satisfy $0 < v_i < 1$ and $2k - 1 < v_j < 2k$, where k is the positive integer, and $i \neq j$. Therefore, we can change the number of the attractors of FHNN by means of altering the fractional order of the neuron.

Furthermore, we set $v_1 = 0.25$, $v_2 = 1.50$ and $v_1 = 0.75$, $v_2 = 1.50$. Then, the corresponding relationship between the input and the output of FHNN can be shown as in Table V.

From Table V, it can be seen that, on the one hand, $(-0.1672, -1.0)$ and $(0.1672, 1.0)$ are two different equilibrium points of FHNN's Lyapunov function or attractors of FHNN when $v_1 = 0.25$ and $v_2 = 1.50$. On the other hand, $(-0.0322, -1.0)$ and $(0.0322, 1.0)$ are two different attractors of FHNN when $v_1 = 0.75$ and $v_2 = 1.50$. Therefore, we can change the value of FHNN's attractors by means of altering the fractional order of the neuron.

Example 4: suppose there are three fractional neurons of FHNN, but each neuron has a different fractional order. Thus, $S = 3$ and $v_1 \neq v_2 \neq v_3$. We also set $\gamma = 1.40$. Then, from (58)–(61), the corresponding relationship between the input and the output of FHNN according to various permutations of v_1, v_2, and v_3 can be shown as in Table VI.

From Table VI, we can see that, on the one hand, with regard to the multineuron FHNN, the equilibrium points of FHNN's Lyapunov function or attractors of FHNN varies with the fractional-order of the neuron. On the other hand, comparing Table III, Fig. 8 and Table VI, we can further see that there are only two, but not eight, attractors of FHNN whenever any two neurons of the trio-neuron FHNN satisfy $0 < v_i < 1$ and $2k - 1 < v_j < 2k$, where k is a positive integer, and $i \neq j$. Therefore, we can simultaneously change both the value and the number of the attractors of FHNN by means of altering the fractional order of the neuron. Furthermore, in all the above-mentioned cases, $(0, 0, 0)$ is the saddle point of FHNN.

C. Application of FHNN to Defense Against Chip Cloning Attacks for Anticounterfeiting

In this section, we analyze the application of FHNN to the defense against chip cloning attacks for anticounterfeiting. From the aforementioned discussion, it can be seen that FHNN has a stronger associative memory than HNN. We can obviously apply FHNN to pattern recognition, similar to HNN. To avoid repetition, the application of FHNN to pattern recognition is not described in this paper. Therefore, we propose a novel promising application case of FHNN in brief.

We apply FHNN to defense against chip cloning attacks for anticounterfeiting. Copyright is in crisis nowadays. Levies or legal penalties only patch the holes in an already leaky system. The flaw lies not only in the attitude toward copyright in our society, but also in the anticounterfeiting technology. In many cases, identification and qualification are important methods for anticounterfeiting. Encryption and digital watermarking technology have become mature, but

TABLE VII

CORRESPONDING RELATIONSHIP BETWEEN INPUT AND OUTPUT OF FHNN ACCORDING TO WEAK VARIATION OF FRACTIONAL ORDER OF THE NEURON ($\gamma = 1.40$). (a) $v_1 = 0.498$, $v_2 = 1.50$. (b) $v_1 = 0.499$, $v_2 = 1.50$. (c) $v_1 = 0.501$, $v_2 = 1.50$. (d) $v_1 = 0.502$, $v_2 = 1.50$

(a)

Input n_1, n_2	Output a_1, a_2	Input n_1, n_2	Output a_1, a_2
−1.0, −1.0	−0.0690 , −1.0	0.5 ,0.0	0.0690 ,1.0
−0.5, −1.0	−0.0690 , −1.0	1.0 ,0.0	0.0690 ,1.0
0.0, −1.0	−0.0690 , −1.0	−1.0 ,0.5	0.0690 ,1.0
0.5, −1.0	−0.0690 , −1.0	−0.5 ,0.5	0.0690 ,1.0
1.0, −1.0	−0.0690 , −1.0	0.0 ,0.5	0.0690 ,1.0
−1.0, −0.5	−0.0690 , −1.0	0.5 ,0.5	0.0690 ,1.0
−0.5, −0.5	−0.0690 , −1.0	1.0 ,0.5	0.0690 ,1.0
0.0, −0.5	−0.0690 , −1.0	−1.0 ,1.0	0.0690 ,1.0
0.5, −0.5	−0.0690 , −1.0	−0.5 ,1.0	0.0690 ,1.0
1.0, −0.5	−0.0690 , −1.0	0.0 ,1.0	0.0690 ,1.0
−1.0, 0.0	−0.0690 , −1.0	0.5 ,1.0	0.0690 ,1.0
−0.5, 0.0	−0.0690 , −1.0	1.0 ,1.0	0.0690 ,1.0

(b)

Input n_1, n_2	Output a_1, a_2	Input n_1, n_2	Output a_1, a_2
−1.0, −1.0	−0.0688 , −1.0	0.5 ,0.0	0.0688 ,1.0
−0.5, −1.0	−0.0688 , −1.0	1.0 ,0.0	0.0688 ,1.0
0.0, −1.0	−0.0688 , −1.0	−1.0 ,0.5	0.0688 ,1.0
0.5, −1.0	−0.0688 , −1.0	−0.5 ,0.5	0.0688 ,1.0
1.0, −1.0	−0.0688 , −1.0	0.0 ,0.5	0.0688 ,1.0
−1.0, −0.5	−0.0688 , −1.0	0.5 ,0.5	0.0688 ,1.0
−0.5, −0.5	−0.0688 , −1.0	1.0 ,0.5	0.0688 ,1.0
0.0, −0.5	−0.0688 , −1.0	−1.0 ,1.0	0.0688 ,1.0
0.5, −0.5	−0.0688 , −1.0	−0.5 ,1.0	0.0688 ,1.0
1.0, −0.5	−0.0688 , −1.0	0.0 ,1.0	0.0688 ,1.0
−1.0, 0.0	−0.0688 , −1.0	0.5 ,1.0	0.0688 ,1.0
−0.5, 0.0	−0.0688 , −1.0	1.0 ,1.0	0.0688 ,1.0

(c)

Input n_1, n_2	Output a_1, a_2	Input n_1, n_2	Output a_1, a_2
−1.0, −1.0	−0.0683 , −1.0	0.5 ,0.0	0.0683 ,1.0
−0.5, −1.0	−0.0683 , −1.0	1.0 ,0.0	0.0683 ,1.0
0.0, −1.0	−0.0683 , −1.0	−1.0 ,0.5	0.0683 ,1.0
0.5, −1.0	−0.0683 , −1.0	−0.5 ,0.5	0.0683 ,1.0
1.0, −1.0	−0.0683 , −1.0	0.0 ,0.5	0.0683 ,1.0
−1.0, −0.5	−0.0683 , −1.0	0.5 ,0.5	0.0683 ,1.0
−0.5, −0.5	−0.0683 , −1.0	1.0 ,0.5	0.0683 ,1.0
0.0, −0.5	−0.0683 , −1.0	−1.0 ,1.0	0.0683 ,1.0
0.5, −0.5	−0.0683 , −1.0	−0.5 ,1.0	0.0683 ,1.0
1.0, −0.5	−0.0683 , −1.0	0.0 ,1.0	0.0683 ,1.0
−1.0, 0.0	−0.0683 , −1.0	0.5 ,1.0	0.0683 ,1.0
−0.5, 0.0	−0.0683 , −1.0	1.0 ,1.0	0.0683 ,1.0

(d)

Input n_1, n_2	Output a_1, a_2	Input n_1, n_2	Output a_1, a_2
−1.0, −1.0	−0.0680 , −1.0	0.5 ,0.0	0.0680 ,1.0
−0.5, −1.0	−0.0680 , −1.0	1.0 ,0.0	0.0680 ,1.0
0.0, −1.0	−0.0680 , −1.0	−1.0 ,0.5	0.0680 ,1.0
0.5, −1.0	−0.0680 , −1.0	−0.5 ,0.5	0.0680 ,1.0
1.0, −1.0	−0.0680 , −1.0	0.0 ,0.5	0.0680 ,1.0
−1.0, −0.5	−0.0680 , −1.0	0.5 ,0.5	0.0680 ,1.0
−0.5, −0.5	−0.0680 , −1.0	1.0 ,0.5	0.0680 ,1.0
0.0, −0.5	−0.0680 , −1.0	−1.0 ,1.0	0.0680 ,1.0
0.5, −0.5	−0.0680 , −1.0	−0.5 ,1.0	0.0680 ,1.0
1.0, −0.5	−0.0680 , −1.0	0.0 ,1.0	0.0680 ,1.0
−1.0, 0.0	−0.0680 , −1.0	0.5 ,1.0	0.0680 ,1.0
−0.5, 0.0	−0.0680 , −1.0	1.0 ,1.0	0.0680 ,1.0

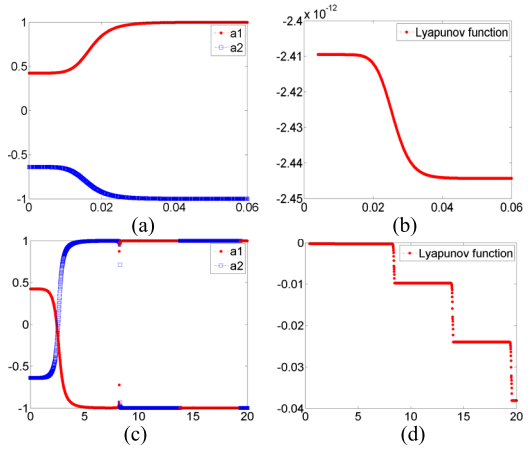

Fig. 10. FHNN's stability and convergence in neighborhood of $v = 6.00$ ($\gamma = 1.40$, $n_1(0) = 0.50$, $n_2(0) = -1.00$). (a) Time response curve of FHNN's output when$v = 5.999999$. (b) Time response curve of FHNN's Lyapunov function when $v = 5.999999$. (c) Time response curve of FHNN's output when $v = 6.000000$. (d) Time response curve of FHNN's Lyapunov function when $v = 6.000000$.

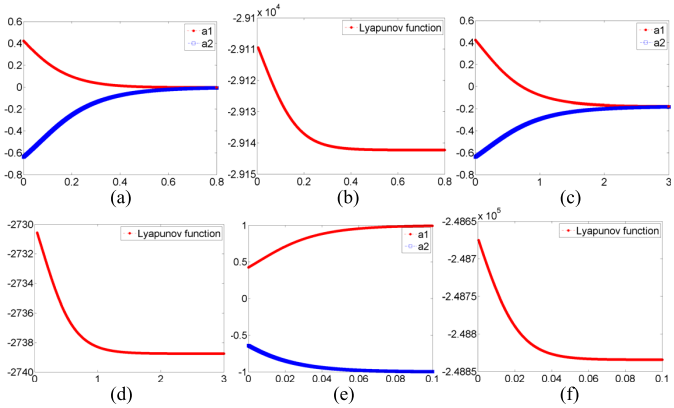

Fig. 11. FHNN's stability and convergence in neighborhood of $v = 1.00$ ($\gamma = 1.40$, $n_1(0) = 0.50$, $n_2(0) = -1.00$). (a) Time response curve of FHNN's output when$v = 0.99$. (b) Time response curve of FHNN's Lyapunov function when $v = 0.99$. (c) Time response curve of FHNN's output when $v = 1.00$. (d) Time response curve of FHNN's Lyapunov function when $v = 1.00$. (e) Time response curve of FHNN's output when $v = 1.01$. (f) Time response curve of FHNN's Lyapunov function when $v = 1.01$.

there is no effective method to identify the piracy of electronic copies. Defense against chip cloning attacks technology for anticounterfeiting is an emerging discipline that has not been studied yet indepth. Based on the aforementioned features of FHNN, we can analyze the defense against chip cloning attacks properties of FHNN.

In the first case, suppose there are only two fractional neurons of FHNN, and each neuron has the same fractional order. Thus, $v_1 = v_2 = v$. From (45)–(47) and (55)–(57), FHNN's stability and convergence in the neighborhood of even-order $v = 2k$ can be as shown in Fig. 10.

From Fig. 10, we can see that in the neighborhood of even-order $v = 2k$, FHNN is convergent when v is only less than $2k$ part per million, but it is not convergent when $v = 2k$, where k is a positive integer. Furthermore, FHNN's stability and convergence in the neighborhood of $v = 1.00$ can be as shown in Fig. 11.

In Fig. 11, in order to get distinguish further, we set the weighting matrix in (19) as $W = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$. From Fig. 11, we can see that the attractors of FHNN are $(0, 0)$, $(-0.1823, -0.1823)$ and $(1.00, -1.00)$ when $v = 0.99$, $v = 1.00$ and $v = 1.01$, respectively.

In the second case, suppose there are also two fractional neurons of FHNN, but each of its neurons has a different fractional order. Thus, $v_1 \neq v_2$. The corresponding relationship between FHNN's input and its output according to the weak variation in the fractional order of the neuron can be shown as in Table VII.

From Tables IV and VII, we can further see that the value of the attractors of FHNN change very little while the fractional order of the neuron v_1 varies weakly.

From the aforementioned two cases, it can be seen that the value of the attractors of FHNN essentially relate to the fractional order of the neuron v_i. Furthermore, from (18), it can be also seen that the fractional order of the neuron v_i relates to its v_i-order fractor in essence. With regard to the neuron's v_i-order fractor, from (14), its driving-point impedance function F_{v_i} is generally in a direct ratio to $\xi_i = r^{(1-p_i)/v_i}/c$, where $v_i = q_i + p_i$ is a positive real number, q_i is a positive integer, and $0 \leq p_i \leq 1$. In other words, the values of the attractors of FHNN are changing while the values of the resistors or capacitors of the fractor vary.

Actually, according to the latest electronic manufacturing technology, we have not been able to manufacture two resistors or capacitors with an identical value. It is luck in the midst of sadness. To date, no one has been able to manufacture two FHNNs with identical values of attractors. Therefore, we can apply FHNN for defense against chip cloning attacks for anticounterfeiting, and this will be discussed in our future work.

V. Conclusion

How to apply fractional calculus to signal analysis and processing, especially to neural networks, is an emerging field of study and few studies have beenperformed in this area. Fractional calculus has been incorporated into artificial neural networks, mainly because of its long-term memory and nonlocality. Therefore, it is natural to think about how to generalize the first-order HNNs to the fractional-order ones, and how to implement FHNN by means of fractional calculus. This paper is mainly to discuss a novel conceptual framework: FHNNs. Therefore, it naturally makes one to ponder how to generalize HNN to the fractional-order ones, and how to implement FHNN by means of fractional calculus. This paper presents a novel conceptual framework: FHNN. We propose to introduce a novel mathematical method: fractional calculus to implement FHNN. We implement FHNN by utilizing fractor and the fractional steepest descent approach, construct its Lyapunov function, and further analyze its attractors. We apply fractional calculus to implement FHNN, mainly because of its long-term memory and nonlocality. The main contribution of our work is to propose FHNN in the form of an analog circuit by utilizing fractor and the fractional steepest descent approach, construct its Lyapunov function, prove its Lyapunov stability, analyze its attractors, and apply FHNN to the defense against chip cloning attacks for anticounterfeiting. The arbitrary order of FHNN represents an additional degree-of-freedom to fit a specific behavior such as power-law long-term memory or power-law nonlocality. A significant advantage of FHNN is that its attractors essentially relate to the neuron's fractional order. FHNN possesses the fractional-order stability and the fractional-order sensitivity characteristics. We can apply FHNN to the defense against chip cloning attacks for anticounterfeiting.

From the aforementioned discussion, we can also see that there are many other problems that need to be further studied. For example, how to construct an FHNN that is convergent when $2k-2 < v < 2k-1$, how to construct the arbitrary-order fractor, how to construct fractional chaotic neural networks, how to implement the analog circuit realization of defense against chip cloning attacks based on FHNN, and so on. These will be discussed in our future work.

References

[1] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci. USA*, vol. 79, no. 8, pp. 2554–2558, 1982.

[2] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proc. Nat. Acad. Sci. USA*, vol. 81, no. 10, pp. 3088–3092, 1984.

[3] J. J. Hopfield and D. W. Tank, "'Neural' computation of decisions in optimization problems," *Biol. Cybern.*, vol. 52, no. 3, pp. 141–152, 1985.

[4] J. J. Hopfield and D. W. Tank, "Computing with neural circuits: A model," *Science*, vol. 233, no. 4764, pp. 625–633, 1986.

[5] D. W. Tank and J. J. Hopfield, "Simple 'neural' optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit," *IEEE Trans. Circuits Syst.*, vol. 33, no. 5, pp. 533–541, May 1986.

[6] J.-H. Li, A. N. Michel, and W. Porod, "Analysis and synthesis of a class of neural networks: Linear systems operating on a closed hypercube," *IEEE Trans. Circuits Syst.*, vol. 36, no. 11, pp. 1405–1422, Nov. 1989.

[7] A. N. Michel and J. A. Farrell, "Associative memories via artificial neural networks," *IEEE Control Syst. Mag.*, vol. 10, no. 3, pp. 6–17, Apr. 1990.

[8] B. Kosko, "Adaptive bidirectional associative memories," *Appl. Opt.*, vol. 26, no. 23, pp. 4947–4960, 1987.

[9] B. Kosko, "Bidirectional associative memories," *IEEE Trans. Syst., Man, Cybern.*, vol. 18, no. 1, pp. 49–60, Jan./Feb. 1988.

[10] T. Samad and P. Harper, "High-order Hopfield and tank optimization networks," *Parallel Comput.*, vol. 16, nos. 2–3, pp. 287–292, 1990.

[11] E. B. Kosmatopoulos and M. A. Christodoulou, "Structural properties of gradient recurrent high-order neural networks," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 42, no. 9, pp. 592–603, Sep. 1995.

[12] X. Liu, K. L. Teo, and B. Xu, "Exponential stability of impulsive high-order Hopfield-type neural networks with time-varying delays," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1329–1339, Nov. 2005.

[13] X. Liu and Q. Wang, "Impulsive stabilization of high-order Hopfield-type neural networks with time-varying delays," *IEEE Trans. Neural Netw.*, vol. 19, no. 1, pp. 71–79, Jan. 2008.

[14] H. Huang, D. W. C. Ho, and J. Lam, "Stochastic stability analysis of fuzzy Hopfield neural networks with time-varying delays," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 52, no. 5, pp. 251–255, May 2005.

[15] H. Li, B. Chen, Q. Zhou, and W. Qian, "Robust stability for uncertain delayed fuzzy Hopfield neural networks with Markovian jumping parameters," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 1, pp. 94–102, Feb. 2009.

[16] R. Yang, H. Gao, and P. Shi, "Novel robust stability criteria for stochastic Hopfield neural networks with time delays," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 2, pp. 467–474, Mar. 2009.

[17] B. Zhang, S. Xu, G. Zong, and Y. Zou, "Delay-dependent exponential stability for uncertain stochastic Hopfield neural networks with time-varying delays," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 6, pp. 1241–1247, Jun. 2009.

[18] N. Özdemir, B. B. İskender, and N. Y. Özgür, "Complex valued neural network with Möbius activation function," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 16, no. 12, pp. 4698–4703, 2011.

[19] A. Alofi, J. Cao, A. Elaiw, and A. Al-Mazrooei, "Delay-dependent stability criterion of Caputo fractional neural networks with distributed delay," *Discrete Dyn. Nature Soc.*, vol. 2014, no. 1, Jan. 2014, Art. no. 529358.

[20] E. Kaslik and S. Sivasundaram, "Dynamics of fractional-order neural networks," in *Proc. Int. Joint Conf. Neural Netw.*, San Jose, CA, USA, Jul./Aug. 2011, pp. 611–618.

[21] R. Zhang, D. Qi, and Y. Wang, "Dynamics analysis of fractional order three-dimensional Hopfield neural network," in *Proc. 6th Int. Conf. Natural Comput.*, Yantai, China, Aug. 2010, pp. 3037–3039.

[22] M. A. Z. Raja, J. A. Khan, and I. M. Qureshi, "A new stochastic approach for solution of Riccati differential equation of fractional order," *Ann. Math. Artif. Intell.*, vol. 60, nos. 3–4, pp. 229–250, 2010.

[23] M. A. Z. Raja, J. A. Khan, and I. M. Qureshi, "Swarm intelligence optimized neural network for solving fractional-order systems of Bagley–Torvik equation," *Eng. Intell. Syst.*, vol. 19, no. 1, pp. 41–51, 2011.

[24] M. A. Z. Raja, J. A. Khan, and I. M. Qureshi, "Evolutionary computation technique for solution of Riccati differential equation of arbitrary order," *World Acad. Sci. Eng. Technol.*, vol. 3, no. 10, pp. 303–309, 2009.

[25] M. A. Z. Raja, J. A. Khan, and I. M. Qureshi, "Solution of fractional order system of Bagley–Torvik equation using evolutionary computational intelligence," *Math. Problems Eng.*, vol. 2011, Jan. 2011, Art. no. 765075.

[26] K. B. Oldham and J. Spanier, *The Fractional Calculus: Theory and Applications of Differentiation and Integration to Arbitrary Order*. New York, NY, USA: Academic, 1974.

[27] A. C. McBride and G. F. Roach, Eds., *Fractional Calculus*. New York, NY, USA: Halsted, 1986.

[28] K. Nishimoto, *Fractional Calculus: Integrations and Differentiations of Arbitrary Order*. New Haven, CT, USA: Univ. of New Haven Press, 1989.

[29] S. G. Samko, A. A. Kilbas, and O. I. Marichev, *Fractional Integrals and Derivatives: Theory and Applications*. Yverdon-les-Bains, Switzerland: Gordon and Breach, 1993.

[30] I. Podlubny, *Fractional Differential Equations: An Introduction to Fractional Derivatives, Fractional Differential Equations, to Methods of Their Solution and Some of Their Applications*. San Diego, CA, USA: Academic, 1998.

[31] M. Abramowitz and I. A. Stegun, Eds., *Handbook of Mathematical Functions: With Formulas, Graphs, and Mathematical Tables*. Washington, DC, USA: U.S. Department Commerce Press, 1964.

[32] P. L. Butzer and U. Westphal, "An introduction to fractional calculus," in *Applications of Fractional Calculus in Physics*. Singapore: World Scientific, 2000.
[33] S. Kempfle, I. Schäfer, and H. Beyer, "Fractional calculus via functional calculus: Theory and applications," *Nonlinear Dyn.*, vol. 29, nos. 1–4, pp. 99–127, 2002.
[34] R. L. Magin, "Fractional calculus in bioengineering, part 1," *Critical Rev. Biomed. Eng.*, vol. 32, nos. 1–4, pp. 1–378, 2004.
[35] N. Özdemir and D. Karadeniz, "Fractional diffusion-wave problem in cylindrical coordinates," *Phys. Lett. A*, vol. 372, no. 38, pp. 5968–5972, 2008.
[36] N. Özdemir, O. P. Agrawal, D. Karadeniz, and B. B. İskender, "Analysis of an axis-symmetric fractional diffusion-wave problem," *J. Phys. A, Math. Theor.*, vol. 42, no. 35, 2009, Art. no. 355208.
[37] Y. Povstenko, "Solutions to the fractional diffusion-wave equation in a wedge," *Fract. Calculus Appl. Anal.*, vol. 17, no. 1, pp. 122–135, 2014.
[38] R. C. Koeller, "Applications of fractional calculus to the theory of viscoelasticity," *J. Appl. Mech.*, vol. 51, no. 2, pp. 299–307, 1984.
[39] Y. A. Rossikhin and M. V. Shitikova, "Applications of fractional calculus to dynamic problems of linear and nonlinear hereditary mechanics of solids," *Appl. Mech. Rev.*, vol. 50, no. 1, pp. 15–67, 1997.
[40] S. Manabe, "A suggestion of fractional-order controller for flexible spacecraft attitude control," *Nonlinear Dyn.*, vol. 29, no. 1, pp. 251–268, 2002.
[41] C.-C. Tseng, "Design of fractional order digital FIR differentiators," *IEEE Signal Process. Lett.*, vol. 8, no. 3, pp. 77–79, Mar. 2001.
[42] Y. Chen and B. M. Vinagre, "A new IIR-type digital fractional order differentiator," *Signal Process.*, vol. 83, no. 11, pp. 2359–2365, 2003.
[43] Y.-F. Pu, "Research on application of fractional calculus to latest signal analysis and processing," Ph.D. dissertation, School Electron. Inf., Sichuan Univ., Chengdu, China, 2006.
[44] Y.-F. Pu, W. Wang, J.-L. Zhou, Y. Wang, and H. Jia, "Fractional differential approach to detecting textural features of digital image and its fractional differential filter implementation," *Sci. China F, Inf. Sci.*, vol. 51, no. 9, pp. 1319–1339, 2008.
[45] Y.-F. Pu, J.-L. Zhou, and X. Yuan, "Fractional differential mask: A fractional differential-based approach for multiscale texture enhancement," *IEEE Trans. Image Process.*, vol. 19, no. 2, pp. 491–511, Feb. 2010.
[46] Y.-F. Pu and J.-L. Zhou, "A novel approach for multi-scale texture segmentation based on fractional differential," *Int. J. Comput. Math.*, vol. 88, no. 1, pp. 58–78, 2011.
[47] J. Bai and X. C. Feng, "Fractional-order anisotropic diffusion for image denoising," *IEEE Trans. Image Process.*, vol. 16, no. 10, pp. 2492–2502, Oct. 2007.
[48] Y.-F. Pu *et al.*, "Fractional partial differential equation denoising models for texture image," *Sci. China Inf. Sci.*, vol. 57, no. 7, pp. 1–19, 2014.
[49] Y.-F. Pu, P. Siarry, J.-L. Zhou, and N. Zhang, "A fractional partial differential equation based multiscale denoising model for texture image," *Math. Methods Appl. Sci.*, vol. 37, no. 12, pp. 1784–1806, 2014.
[50] Y.-F. Pu, J.-L. Zhou, P. Siarry, N. Zhang, and Y.-G. Liu, "Fractional partial differential equation: Fractional total variation and fractional steepest descent approach-based multiscale denoising model for texture image," *Abstract Appl. Anal.*, vol. 2013, pp. 1–19, Aug. 2013.
[51] Y.-F. Pu, "Material performance measurement of a promising circuit element: Fractor-part I: Driving-point impedance function of the arbitrary-order fractor in its natural implementation," *Mater. Res. Innov.*, vol. 19, no. S10, pp. 176–182, 2015.
[52] Y.-F. Pu, "Material performance measurement of a promising circuit element: Fractor-part II: Measurement units and physical dimensions of fractance and rules for fractors in series and parallel," *Mater. Res. Innov.*, vol. 19, no. S10, pp. 183–189, 2015.
[53] Y.-F. Pu *et al.*, "A recursive two-circuits series analog fractance circuit for any order fractional calculus," *Proc. SPIE*, vol. 6027, pp. 509–519, Jan. 2006.
[54] Y.-F. Pu *et al.*, "Structuring analog fractance circuit for 1/2 order fractional calculus," in *Proc. 6th IEEE Int. Conf. ASIC*, Shanghai, China, Oct. 2005, pp. 1136–1139.
[55] Y.-F. Pu, X. Yuan, K. Liao, J.-L. Zhou, N. Zhang, and Y. Zeng, "A recursive net-grid-type analog fractance circuit for any order fractional calculus," in *Proc. IEEE Conf. Mechatronics Autom.*, Niagara Falls, ON, Canada, Jul. 2005, pp. 1375–1380.
[56] Y.-F. Pu, X. Yuan, K. Liao, and J.-L. Zhou, "Implement any fractional order neural-type pulse oscillator with net-grid-type analog fractance circuit," *J. Sichuan Univ.*, vol. 38, no. 1, pp. 128–132, 2006.
[57] Y.-F. Pu, "Implement any fractional order multilayer dynamic associative neural network," in *Proc. 6th IEEE Conf. ASIC*, Shanghai, China, Oct. 2005, pp. 635–638.
[58] Y.-F. Pu, J.-L. Zhou, Y. Zhang, N. Zhang, G. Huang, and P. Siarry, "Fractional extreme value adaptive training method: Fractional steepest descent approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 4, pp. 653–662, Apr. 2015.
[59] F. M. Ham and I. Kostanic, *Principles of Neurocomputing for Science and Engineering*. New York, NY, USA: McGraw-Hill, 2001.
[60] D. O. Hebb, *The Organization of Behavior*. New York, NY, USA: Wiley, 1949.
[61] A. J. Storkey, "Increasing the capacity of a Hopfield network without sacrificing functionality," in *Proc. 7th Int. Conf. Artif. Neural Netw.*, Lausanne, Switzerland, Oct. 1997, pp. 451–456.
[62] A. J. Storkey and R. Valabregue, "The basins of attraction of a new Hopfield learning rule," *Neural Netw.*, vol. 12, no. 6, pp. 869–876, 1999.

Yi-Fei Pu received the Ph.D. degree from the College of Electronics and Information Engineering, Sichuan University, Chengdu, China, in 2006.

He is currently a Full Professor and a Doctoral Supervisor with the College of Computer Science and the College of Software Engineering, Sichuan University, and is the Chief Technology Officer of Chengdu PU Chip Science and Technology Company, Ltd, Chengdu. He is elected for the Thousand Talents Program of Sichuan Province. He has firstly authored about 20 papers indexed by SCI in journals, such as the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the IEEE ACCESS, *Mathematic Methods in Applied Sciences*, *Science in China Series F: Information Sciences* and *Science China Information Sciences*, and holds 11 Chinese inventive patents, as the first or single inventor. He focuses on the application of fractional calculus and fractional partial differential equation to signal analysis and signal processing. He held several research projects, such as the National Nature Science foundation of China, and the Returned Overseas Chinese Scholars Project of Education Ministry of China.

Zhang Yi (SM'01–F'15) received the Ph.D. degree from the Institute of Mathematics, Chinese Academy of Science, Beijing, China, in 1994.

He is a Full Professor, a Doctoral Supervisor, and the Dean of the School of Computer Science and Technology with Sichuan University, Chengdu, China. He is supported by the Program for New Century Excellent Talents in University, an Outstanding Expert of Sichuan Province, and gained the National Prize for progress in Science and Technology issued by the Ministry of Education of China in 2012. He has held some projects, such as the National Key Basic Research and Development Plan (973 Plan), the National High Technology Research and Development Program 863, and the National Nature Science foundation of China. He has authored over 30 papers in the IEEE TRANSACTIONS series, and 128 papers are indexed by SCI.

Ji-Liu Zhou (SM'10) received the Ph.D. degree from Sichuan University, Chengdu, China, in 1999.

He is a Full Professor and Doctoral Supervisor with the School of Computer Science and Technology, Sichuan University. He is the Academic and Technical Leader and the Outstanding Expert of Sichuan Province, and has held 17 state or provincial scientific projects, including key projects supported by the National Science Foundation. He has authored over 100 papers, of which more than 80 papers are indexed by SCI, EI, or ISTP. His current research interests include image processing, artificial intelligence, fractional differential application on the latest signal, and image processing.