Memristor-Based Cellular Nonlinear/Neural Network: Design, Analysis, and Applications

Shukai Duan, *Member, IEEE*, Xiaofang Hu, *Student Member, IEEE*, Zhekang Dong, *Student Member, IEEE*, Lidan Wang, *Member, IEEE*, and Pinaki Mazumder, *Fellow, IEEE*

***Abstract*— Cellular nonlinear/neural network (CNN) has been recognized as a powerful massively parallel architecture capable of solving complex engineering problems by performing trillions of analog operations per second. The memristor was theoretically predicted in the late seventies, but it garnered nascent research interest due to the recent much-acclaimed discovery of nanocrossbar memories by engineers at the Hewlett-Packard Laboratory. The memristor is expected to be co-integrated with nanoscale CMOS technology to revolutionize conventional von Neumann as well as neuromorphic computing. In this paper, a compact CNN model based on memristors is presented along with its performance analysis and applications. In the new CNN design, the memristor bridge circuit acts as the synaptic circuit element and substitutes the complex multiplication circuit used in traditional CNN architectures. In addition, the negative differential resistance and nonlinear current–voltage characteristics of the memristor have been leveraged to replace the linear resistor in conventional CNNs. The proposed CNN design has several merits, for example, high density, nonvolatility, and programmability of synaptic weights. The proposed memristor-based CNN design operations for implementing several image processing functions are illustrated through simulation and contrasted with conventional CNNs. Monte-Carlo simulation has been used to demonstrate the behavior of the proposed CNN due to the variations in memristor synaptic weights.**

***Index Terms*— Cellular neural/nonlinear network (CNN), fault tolerance, image processing, memristor, stability.**

I. Introduction

CELLULAR nonlinear/neural network (CNN) was proposed in [1] and [2] by demonstrating how the regularity of cellular automata and local computation of neural networks can be melded together to accelerate numerous real-world computational tasks. Many artificial, physical, chemical, as well as biological systems have been represented using the CNN model [3]. Its continuous-time analog operation allows real-time signal processing at high precision, while its local interaction between the constituent processing elements (PEs) obviate the need of global buses and long interconnects, leading to several digital and analog implementations of CNN architectures in the form of very large scale integrated (VLSI) chips. In [4]–[9], several powerful applications of CNNs were reported including pattern and image analysis such as vertical line detection, noise reduction, edge detection, feature detection, and character recognition.

However, to improve the resolution in static and dynamic image analysis, the size of PEs and the connectivity between the PEs describing the control (feedforward) and feedback templates that are used as programming artifacts in the CNN model of computation [5], [6], must be significantly reduced. The current version of CNN arrays in a CMOS VLSI chip is typically limited to 16-K PEs. Emerging technologies such as resonant tunneling diode [7] and quantum dots [8] have been attempted recently to implement PEs more compactly to improve the resolution of CNN computation. However, lack of programmability associated with fixed connecting elements between the PEs as well as wide variability of tunneling currents in RTDs and quantum dots are the major limitations of these emerging technologies.

The recent advent of memristors [10], [11] has opened the possibility of significantly enhancing the resolution of on-chip CNN model of computation. The memristor was introduced [12] as a fundamental circuit element and recently nanofabrication technology has shown its superior device properties such as nonvolatility, binary as well as multiple memory states, and nanometer geometries that can be shrunk to the ultimate physical dimensions [13]–[16]. These versatile features of memristors have been exploited in showing their applications in nonvolatile memory [17], [18], artificial neural networks [19]–[23], composite circuits [24], [25], and so on. Because its conductance can change in response to the applied voltage or current like a biological synapse, the memristor has been demonstrated an artificial synapse for biological signal processing. Recently, Kim *et al.* [21] and [22] presented a compact memristor bridge synapse in which both weighting and weight programming can be performed at different time slots.

Manuscript received November 10, 2013; accepted June 21, 2014. Date of publication July 21, 2014; date of current version May 15, 2015. This work was supported in part by the Program for New Century Excellent Talents in University under Grant [2013]47, in part by the National Natural Science Foundation of China under Grant 61372139, Grant 61101233, and Grant 60972155, in part by the Spring Sunshine Plan, in part by the Research Project, Ministry of Education of China, under Grant z2011148, in part by the Technology Foundation for Selected Overseas Chinese Scholars, in part by the Ministry of Personnel in China under Grant 2012-186, in part by the University Excellent Talents Supporting Foundations of Chongqing under Grant 2011-65, in part by the University Key Teacher Supporting Foundations of Chongqing under Grant 2011-65, and in part by the Fundamental Research Funds for the Central Universities under Grant XDJK2014A009 and Grant XDJK2013B011.

S. Duan, Z. Dong, and L. Wang are with the College of Electronics and Information Engineering, Southwest University, Chongqing 400715, China (e-mail: duansk@swu.edu.cn; englishp@126.com; ldwang@swu.edu.cn).

X. Hu is with the Department of Department of Mechanical and Biomedical Engineering, City University of Hong Kong, Hong Kong (e-mail: xiaofanghs@gmail.com).

P. Mazumder is with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: mazum@eecs.umich.edu).

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.2014.2334701

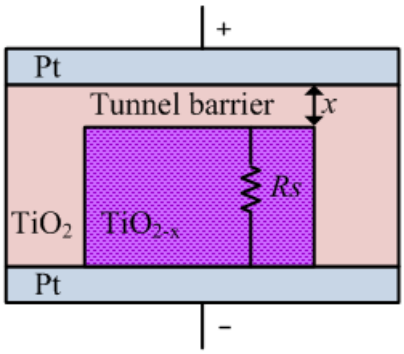

Fig. 1. Physical model of Simmons tunnel barrier memristor [13].

In this paper, a novel type of memristor-based CNN (M-CNN) is addressed. Specifically, the memristor bridge circuit is employed to realize the interactions between the neighboring cells. Different from the memristor circuit in [21] and [22], the memristor used here is a more practical model based on the experimental data. Moreover, since the memristor exhibits nonlinear current–voltage (I–V) characteristic with locally negative differential resistance, the memristor is also considered to replace the original linear resistor in a traditional CNN cell. Thus, an M-CNN, equipped with nonvolatile and programmable synapse circuits, is more versatile and compact and saves the traditional complex output function realization circuits.

In Section II, the basics of the memristor are briefly introduced. Then, the topology and dynamics of the proposed M-CNN are described in Section III. To guarantee the feasibility of implementation scheme, the stability and fault tolerance analysis are investigated in Section IV. Section V presents the numerical simulations of the uncoupled and coupled M-CNNs in image processing, including image inversion, horizontal line detection (HLD), edge extraction, noise removal, as well as a variation analysis based on Monte-Carlo methods. Finally, conclusion and discussion are presented in Section VI.

II. Memristor Basics

The memristor is a nonlinear passive device with variable resistance states. It is mathematically defined by its constitutive relationship of the charge q and the flux φ

$$\frac{d\varphi}{dt} = \frac{d\varphi(q)}{dq} \cdot \frac{dq}{dt}. \tag{1}$$

Based on the basic circuit law, (1) leads to

$$v(t) = \frac{d\varphi(q)}{dq} i(t) = M(q) i(t) \tag{2}$$

where $M(q)$ is defined as the resistance of a memristor called memristance and it is a function of the internal current i and the state variable x.

The Simmons tunnel barrier model is the most accurate physical model of TiO_2/TiO_{2-x} memristor, reported by the Hewlett-Packard Lab [13]. As shown in Fig. 1, the memristor is made up of two TiO_2 and TiO_{2-x} components sandwiched between the two platinum electrodes. The memristance magnitude is determined by the electron tunnel barrier, in series with a resistor. In this case, the state variable x is the width of the Simmons tunnel barrier and its dynamics are represented by [13]

$$\frac{dx(t)}{dt} = \begin{cases} c_{\mathrm{OFF}} \sinh\left(\frac{i}{i_{\mathrm{OFF}}}\right) \exp\left[-\exp\left(\frac{x-a_{\mathrm{OFF}}}{w_c} - \frac{|i|}{b}\right) - \frac{x}{w_c}\right], & i > 0 \\ c_{\mathrm{ON}} \sinh\left(\frac{i}{i_{\mathrm{ON}}}\right) \exp\left[-\exp\left(-\frac{x-a_{\mathrm{ON}}}{w_c} - \frac{|i|}{b}\right) - \frac{x}{w_c}\right], & i < 0 \end{cases} \tag{3}$$

where c_{OFF}, c_{ON}, i_{OFF}, i_{ON}, a_{OFF}, a_{ON}, w_c, and b are the fitting parameters, in which the parameters c_{OFF} and c_{ON} influence the change in magnitude of x, the parameters i_{OFF} and i_{on} reflect the current thresholds, and a_{OFF} and a_{on} are the upper and the lower bounds of x, respectively. Equation (3) is also interpreted as the velocity of the oxygen vacancy drift in TiO_{2-x} material. About this accurate model, two problems have been discussed [15]: 1) there is no explicit relationship between the current and the voltage of the memristive device and 2) it is too complicated to be used in numerical and mathematical analysis. Afterward, an alternative model with simpler expressions, which can reflect the same physical behavior, is proposed [15]. In this simplified model, the derivation of the state variable x is given by

$$\frac{dx(t)}{dt} = \begin{cases} k_{\mathrm{OFF}}\left(\frac{i(t)}{i_{\mathrm{OFF}}} - 1\right)^{a_{\mathrm{OFF}}} \cdot f_{\mathrm{OFF}}(x), & 0 < i_{\mathrm{OFF}} < i \\ 0, & i_{\mathrm{ON}} < i < i_{\mathrm{OFF}} \\ k_{\mathrm{ON}}\left(\frac{i(t)}{i_{\mathrm{ON}}} - 1\right)^{a_{\mathrm{ON}}} \cdot f_{\mathrm{ON}}(x), & i < i_{\mathrm{ON}} < 0 \end{cases} \tag{4}$$

where k_{OFF} is a positive constant and k_{ON} is a negative constant, a_{OFF} and a_{ON} are fitting parameters, and i_{OFF} and i_{ON} are the current thresholds. Correspondingly, there are two window functions $f_{\mathrm{ON}}(x)$ amd $f_{\mathrm{OFF}}(x)$ to represent the dependence of the derivation on the state variable x and guarantee the effective range of x, i.e., $x \in [x_{\mathrm{ON}},\ x_{\mathrm{OFF}}]$

$$f_{\mathrm{OFF}}(x) = \exp\left[-\exp\left(\frac{x - a_{\mathrm{OFF}}}{w_c}\right)\right] \tag{5a}$$

$$f_{\mathrm{ON}}(x) = \exp\left[-\exp\left(\frac{x - a_{\mathrm{ON}}}{w_c}\right)\right]. \tag{5b}$$

The relationship between the current through and the voltage across the memristor can be written as

$$v(t) = \left[R_{\mathrm{ON}} + \frac{R_{\mathrm{OFF}} - R_{\mathrm{ON}}}{x_{\mathrm{OFF}} - x_{\mathrm{ON}}}(x - x_{\mathrm{ON}})\right] \cdot i(t) \tag{6}$$

where $M(x) = R_{\mathrm{ON}} + (R_{\mathrm{OFF}} - R_{\mathrm{ON}}/x_{\mathrm{OFF}} - x_{\mathrm{ON}})(x - x_{\mathrm{ON}})$ is the memristance, in which R_{ON} and R_{OFF}, respectively, denote the low and high resistances of the memristor. Note that positive voltage or current can increase the undoped region width x, and thus increase the memristance; while negative excitation leads to memristance decrease.

A few fundamental simulations have been done to observe the behavior of the memristor subjected to an applied

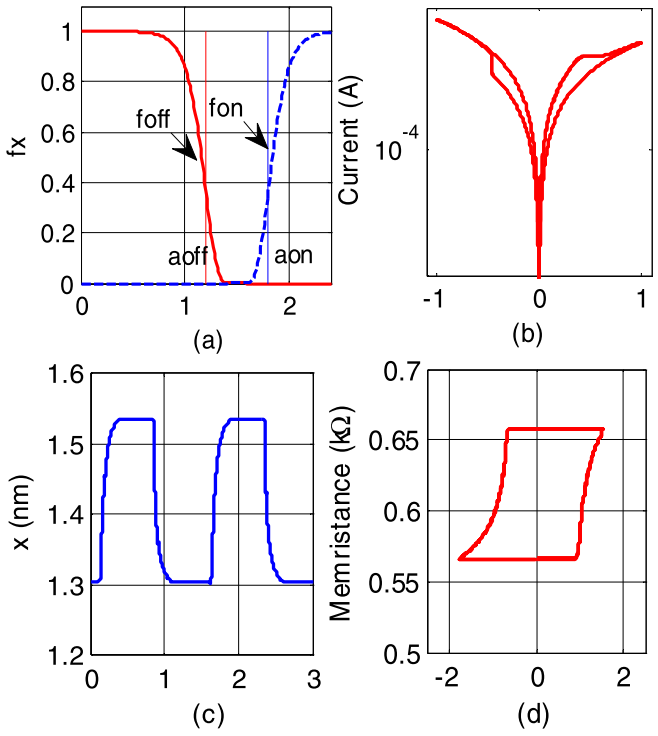

Fig. 2. Characteristic curves of the memristor under $v = \sin(2\pi f)$. (a) Window functions $f_{\mathrm{OFF}}(x)$ (red solid line) and $f_{\mathrm{ON}}(x)$ (blue dashed line). (b) Voltage and current relationship. (c) Change of the state variable x. (d) Relationship between memristance and the current through the device.

sine voltage $v = \sin(2\pi f)$ with results shown in Fig. 2. Fig. 2(a)–(d) shows the window functions $f_{\mathrm{ON}}(x)$ and $f_{\mathrm{OFF}}(x)$, the I–V relationship with current presented in logarithm coordinate, the change of the state variable x, and the memristance change versus the current, respectively. In this simulation, the memristor model parameters are set as: $R_{\mathrm{OFF}} = 1$ kΩ, $R_{\mathrm{on}} = 50$ Ω, $i_{\mathrm{OFF}} = 115$ μA, $i_{\mathrm{ON}} = 8.9$ μA, $a_{\mathrm{OFF}} = 1.2$ nm, $a_{\mathrm{ON}} = 1.8$ nm, and $w_c = 107$ pm.

III. M-CNN

A. Description of the M-CNN

An M-CNN is constituted of an $M \times N$ rectangular array of cells c(i, j) located at site (i, j), $i = 1, 2, 3, \ldots, M$; $j = 1, 2, 3, \ldots, N$, as shown in Fig. 3. A cell has $(2r + 1)^2$ neighboring cells, where $r \in [1, 2, 3, \ldots]$ is the radius of the neighborhood. It only communicates with its closest neighbors, which is the defined local interconnection of CNNs. In this paper, we consider the most commonly used case $r = 1$ for our study, i.e., one cell only transmits information among its eight closest neighbors.

Like in a standard CNN [1], all cells have identical circuit structure and parameter values. Fig. 4 shows a schematic implementation of the proposed M-CNN cell containing a capacitor, a memristor state resistor, and two variable-gain voltage-controlled current sources (VCCSs). Memristor bridge circuits were to used implement the VCCSs for weight setting as well as the weighting operation. Besides, a memristor (M) is employed to replace the linear state resistor. Therefore, the original state-output conversion part is removed.

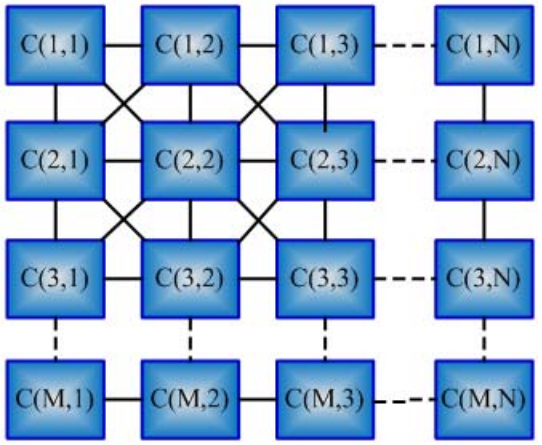

Fig. 3. Topology of an M-CNN, in which the squares represent cells with identical structure.

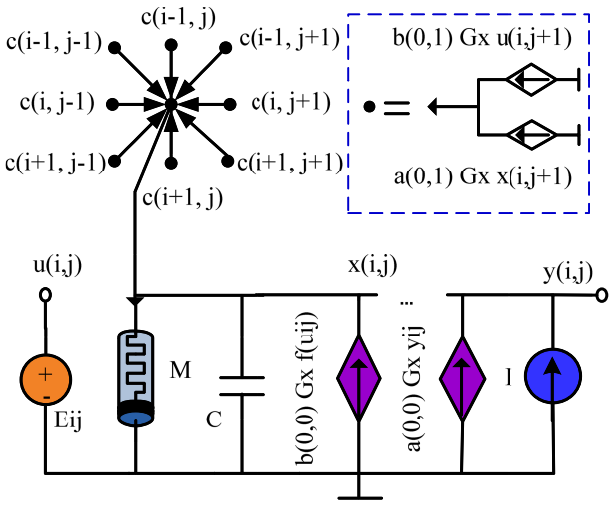

Fig. 4. Schematic diagram of an M-CNN cell.

In a CNN, each cell c(i, j) at ith row and jth column can receive eight input currents from a set of neighboring cells c(k', l') located at k'th row and l'th column of the array, where $(k', l') \in \{(i-1, j-1), (i-1, j), (i-1, j+I), (i, j-1), (i, j+1), (i+1, j-1), (i+1, j), (i+1, j+1)\}$. These inputs are represented by solid dots in Fig. 4. Specially, each solid dot represents an input current that is a summation of two VCCSs of the corresponding neighboring cells, as given in the dashed square. Likewise, each cell c(i, j) also provides eight output currents to its eight neighboring cells.

Note that the total input current of each cell contains not only the currents from its eight neighbors, but also from itself and an independent current source. Therefore, as a matter of simplicity, a cell itself is also considered as a neighbor cell included in its neighborhood in CNNs [1].

The dynamics of a cell c(i, j) usually depends on: a set of 18 weights called template elements, an independent current source denoted by I, an independent voltage source denoted by u_{ij}, as well as its own state x_{ij}. The template elements make up a 3×3 feedback template labeled by A, and a 3×3 control template labeled by B. They determine the

gains of the interconnections between the neighboring cells. The independent current offers an offset current and the independent voltage source continuously provides the input for the network. In addition, because of the usage of the memristor bridge circuits and the memristor state resistor, the state x_{ij} can achieve stability itself and thus as the output directly, i.e., $y_{ij} = x_{ij}$ (detailed description is given later).

Based on the above descriptions and Kirchhoff's current law, the dynamics of each cell can be governed by

$$C\frac{dx_{ij}(t)}{dt} = -m(x_{ij}(t)) + \sum_{c(k,l)} (a_{ij,kl}x_{kl}(t) + b_{ij,kl}u_{kl}) + I \quad (7)$$

where $c(k,l) \in N_r(i,j)$ denotes the r-neighborhood of c(i,j) and here $r = 1$. As described above, for the cell c(i,j), its neighboring cells c(k,l) contain all the neighbors (k',l') and itself.

The term $m(\cdot)$ is the current flowing through the memristor (M) in the form of

$$m(x_{ij}(t)) = \frac{v_M}{M(t)} = \frac{x_{ij}(t)}{M(t)} \quad (8)$$

where $M(t)$ is the memristance of the memristor state resistor. In the following section, the analog implementation scheme of the M-CNN based on the mathematical model (8) will be presented.

B. Implementation of Synaptic Connections With Memristor Bridge Circuits

The template elements (weights) play a crucial role in signal and image processing applications of the CNN. To execute different functions, the templates should be updated correspondingly. In traditional circuit implementation of CNNs, the weights and the weighting operation are achieved through a number of amplifiers and multipliers [7]. For an amplifier, the amplifying gain is fixed once the circuit is built, thereby not being easy to alter. Moreover, the multipliers are implemented with at least eight CMOS transistors that operate in a nonlinear way and hence consume significant power. Therefore, serious nonlinearity is unavoidable in the multiplication processing (weighting operation) [21].

Memristor bridge circuit was recently reported as a candidate of artificial synapse to replace the amplifiers and multipliers to implement weight programming and weighting operation [21], [22]. Such a circuit, consisting of four identical memristors (M_1, M_2, M_3, and M_4), is capable of performing zero, positive and negative synaptic weights, as shown in Fig. 5. When a pulse V_{in} (positive or negative) is applied at the input port, the memristance of each memristor changes correspondingly depending on its polarity. The output voltage between the positive and the negative terminals is governed by

$$V_W = V_+ - V_- = \left(\frac{M_2}{M_1 + M_2} - \frac{M_4}{M_3 + M_4}\right)V_{\text{in}}. \quad (9)$$

In the form of the relationship between the synaptic weight ω and the synaptic input signal V_{in}, (9) can be rewritten as

$$V_W = \omega V_{\text{in}} \quad (10)$$

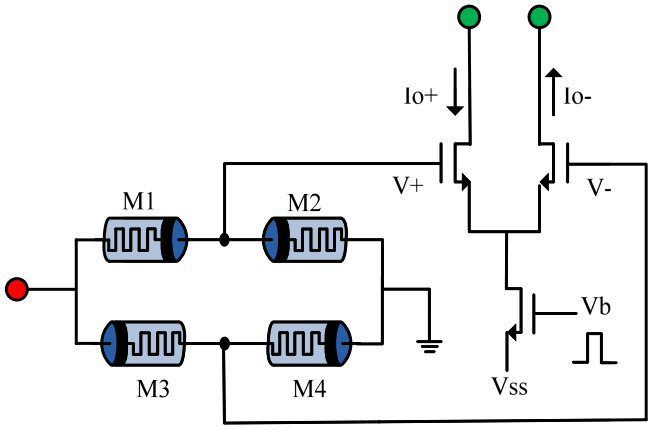

Fig. 5. Memristor bridge circuit [21].

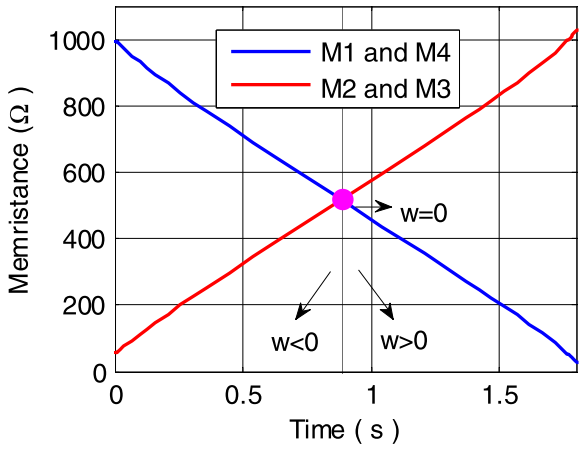

Fig. 6. Change of the memristors in the bridge circuit.

where the synaptic weight ω represents $M_2/(M_1 + M_2) - M_4/(M_3 + M_4)$ within the range $[-0.9, 0.9]$. If $M_2/M_1 > M_4/M_3$, the synaptic weight is positive; if $M_2/M_1 < M_4/M_3$, it is negative; otherwise, it is zero and called balanced state. Fig. 6 shows the change of memristors along with the varying programming time in the bridge structure and indicates the three regions of the weights. In the simulation, M_1 and M_4 were initially set as R_{OFF}, while M_2 and M_3 were R_{ON}. The amplitude of the programming voltage is 1 V.

In the bridge circuit, the input port is shared by the weight programming signal V_p and the synaptic input signal V_{in} for weighting operation at different time slots. The unintended change of the memristance during the weighting operation is negligible as long as the weighting signal amplitude is smaller than the thresholds [13]–[16] or the pulse is very narrow [21], [22]. Therefore, the weighting factor ω is a constant and thus the relationship between the synaptic input and synaptic output is linear, which has been discussed in [21] and [22].

The differential amplifier on the right of Fig. 5 performs the conversion of voltage to current with a transconductance parameter g_m. The currents at the positive and negative terminals are given by

$$\begin{cases} I_{o+} = -\frac{1}{2}g_m V_M = -\frac{1}{2}g_m \omega V_{\text{in}} \\ I_{o-} = \frac{1}{2}g_m V_M = \frac{1}{2}g_m \omega V_{\text{in}}. \end{cases} \quad (11)$$

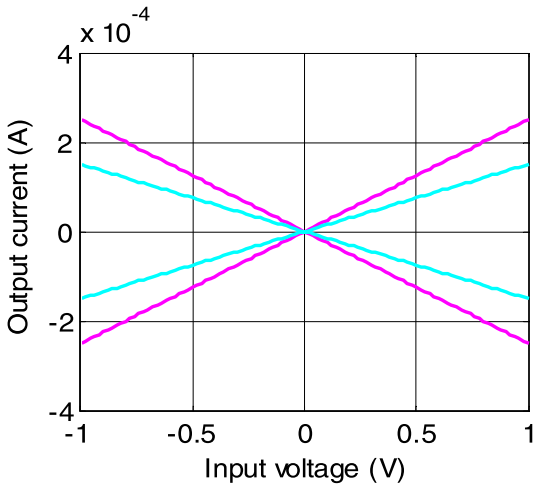

Fig. 7. Synaptic weight processing via the bridge circuit.

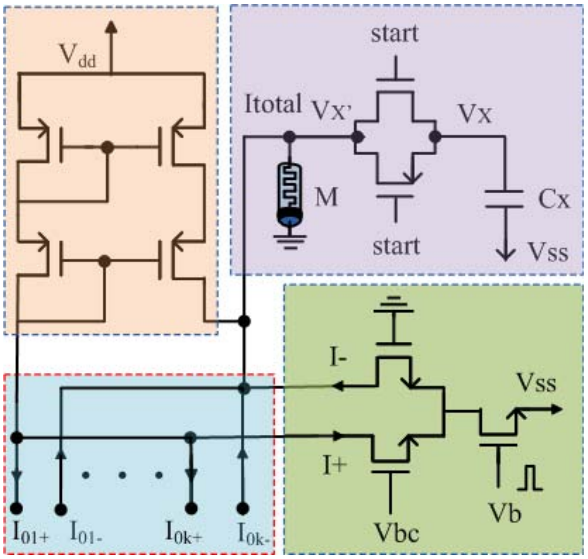

Fig. 8. Schematic diagram of an analog implementation of the M-CNN cell. The bottom left indicates synaptic inputs from the neighboring cells. The bottom right is a bias circuit offering an offset current. These currents are summed up through the active load in the top left. The top right realizes the current-to-voltage conversion and integration.

Fig. 7 shows the weighting operation of the bridge circuits. Four lines represent four levels of the synaptic weights as examples. In general, the synaptic inputs are voltage, which eases the intrachip distribution of multiplication factors (template coefficients) and the intrachip distribution of the state variable to all the neuron synapses. However, the synaptic output is usually in the form of current, which facilitates the summation at the state nodes [5]. The transconductance gain of the differential amplifier, g_m, can be set by the bias current and the transistor size of the amplifier according to the required template coefficients.

C. Implementation of M-CNN Cells

As described above, a cell c(i, j) may receive 18 synaptic inputs from its neighboring cells including itself. These synaptic inputs are the weighted input $b_{ij,kl}u_{kl}$ and the weighted state $a_{ij,kl}x_{kl}$, and they are summed up and injected into the center cell c(i, j) through the state node, namely, the memristor (M) shown in Fig. 4.

Fig. 8 shows an analog implementation schematic diagram of the proposed M-CNN cell. The weighted signals $(I_{01}, \ldots, I_{0k})$ from k memristor bridge circuits input into the center cell from the ports at the lower right corner. The circuit located at the right bottom is a bias circuit that provides the direct current I. The synaptic currents and the bias current are aggregated by connecting all the output terminals together through a differential circuit at the left top. The total current is fed into the memristor and through an integrator at the right top, and then converted into the cell state variable x_{ij} also the cell output y_{ij}, which is finally transmitted to neighboring cells.

The total current I_{total} is given by

$$I_{\text{total}} = \underbrace{I_{01} + I_{02} + I_{03} + \cdots + I_{0k}}_{I_0} + I \tag{12}$$

where $I_{0i} = I_{0i-} - I_{0i+}$ is the output current of the differential amplifier connected with ith memristor bridge circuit. Thus, I_0 that denotes the sum of the synaptic currents can be rewritten as

$$\begin{aligned} I_0 &= g_m(\omega_1 V_{\text{in1}} + \omega_2 V_{\text{in2}} + \omega_3 V_{\text{in3}} + \cdots \omega_k V_{\text{ink}}) \\ &= \sum_k g_m \omega_k V_{\text{ink}}. \end{aligned} \tag{13}$$

Therefore, the memristor bridge circuits can implement the accumulation terms in (7) in a more compact and simpler form. Note that, if both of the feedback and the control templates are nonzero, then $k = 18$; if one template is zero, then $k = 9$, which is a simple implementation method. Another choice is that if the memristor bridge synapses are time-multiplexed, then the 18 weighted values associated with the two templates are implemented using only nine of these synapse circuits. For instance, the control term $\sum b_{ij,kl}u_{kl}$ is first calculated in each computation cycle with the result stored in an analog current-mode memory. After that, the nine synapses are used to calculate the feedback term $\sum a_{ij,kl}x_{kl}$ and the result is added to the former result, finally realizing the two accumulation terms.

The total current flows through the state memristor and the output voltage corresponds to the derivation of the cell state, i.e., dx_{ij}/dt. Then, through an integrator, the cell state x can be obtained

$$Vx = Cx \cdot M\left(\sum_k g_m \omega_k V_{\text{ink}} + I\right). \tag{14}$$

Note that, besides performing the summation function, the active load part also plays the role of restricting the range of the state memristor voltage as [22]

$$Vx = \begin{cases} Cx \cdot (V_{\text{dd}} - 2V_{\text{th}}), & I_{\text{total}} \geq \frac{V_{\text{dd}} - 2V_{\text{th}}}{R_{\text{out}}} \\ Cx \cdot M \cdot I_{\text{total}}, & \frac{-V_{\text{ss}} + 2V_{\text{th}}}{R_{\text{out}}} \leq I_{\text{total}} \leq \frac{V_{\text{dd}} - 2V_{\text{th}}}{R_{\text{out}}} \\ Cx \cdot (-V_{\text{ss}} + 2V_{\text{th}}), & I_{\text{total}} \leq \frac{-V_{\text{ss}} + 2V_{\text{th}}}{R_{\text{out}}}. \end{cases} \tag{15}$$

The original output function is no longer necessary, because the state can achieve stability (bounded) and support the binary output, which leads to the further reduction of the CNN circuit.

IV. Mathematical Analysis

A. Stability

By definition, a CNN is stable if it has a steady output as time goes to infinity [9]. In circuit analysis, a continuous-time autonomous network is stable if its solution is bounded and its energy function is damped. Since the state variable (solution) of a CNN (a specific network) is limited by the circuit parameters, it is easy to guarantee its boundness. Then, in general, the energy degradation is considered as the key point to prove the stability of a CNN [9]. According to the Lyapunov stability theorem, the energy function of the M-CNN is chosen as

$$E(t) = \sum_{(i,j)} \int_0^{x_{ij}} m(s)ds - \frac{1}{2}\sum_{(i,j)}\sum_{(k,l)} a_{ij,kl}x_{kl}(t)x_{ij}(t) - \sum_{(i,j)}\sum_{(k,l)} b_{ij,kl}u_{kl}(t)x_{ij}(t) - \sum_{(i,j)} Ix_{ij}(t). \quad (16)$$

Theorem 1: The function $E(t)$ is bounded by

$$\max|E(t)| \le E_{\max} \quad (17)$$

where

$$E_{\max} = \frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}\left|a_{ij,kl}x_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}\sum_{(k,l)}\left|b_{ij,kl}u_{kl}(t)x_{ij}(t)\right| + MN\left(|I| + \max_{i,j}\left|\int_0^{x_{ij}} m(s)ds\right|\right) \quad (18)$$

for an $M \times N$ CNN.

Proof: By the definition of $E(t)$, one can obtain

$$\begin{aligned}|E(t)| &\le \frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}\left|a_{ij,kl}x_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}\sum_{(k,l)}\left|b_{ij,kl}u_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}|Ix_{ij}(t)| + \sum_{(i,j)}\left|\int_0^{x_{ij}} m(s)ds\right| \\ &\le \frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}\left|a_{ij,kl}x_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}\sum_{(k,l)}\left|b_{ij,kl}u_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}|I|\cdot|x_{ij}(t)| + \sum_{(i,j)}\left|\int_0^{x_{ij}} m(s)ds\right| \\ &\le \frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}\left|a_{ij,kl}x_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}\sum_{(k,l)}\left|b_{ij,kl}u_{kl}(t)x_{ij}(t)\right| + MN(|I|) + MN\left(\max_{i,j}\left|\int_0^{x_{ij}} m(s)ds\right|\right) \\ &\le \frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}\left|a_{ij,kl}x_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}\sum_{(k,l)}\left|b_{ij,kl}u_{kl}(t)x_{ij}(t)\right| + MN\left(|I| + \max_{i,j}\left|\int_0^{x_{ij}} m(s)ds\right|\right) = E_{\max}. \end{aligned} \quad (19)$$

It follows from (18) and (19) that $E(t)$ is bounded as claimed in (17). It is worth noting that the output voltage of a cell is constrained between the two boundaries in circuit implementation [as described in (15)]. Like in most of the CNN applications, the two limitations are used to represent its binary outputs, i.e., 1 and -1 in image processing, respectively. Thus, in the mathematical model of an M-CNN, the range of the state x_{ij} is guaranteed within $[-1, 1]$.

Theorem 2: The function $E(t)$ is monotonically decreasing, namely

$$\frac{dE(t)}{dt} \le 0. \quad (20)$$

Proof: If the feedback template is symmetric, i.e., $a_{ij,kl} = a_{kl,ij}$, the derivative of the energy function with respect to time t is given by

$$\frac{dE(t)}{dt} = -\sum_{(i,j)}\sum_{(k,l)} a_{ij,kl}x_{kl}(t)\frac{dx_{ij}(t)}{dt} - \sum_{(i,j)}\sum_{(k,l)} b_{ij,kl}u_{kl}(t)\frac{dx_{ij}(t)}{dt} - \sum_{(i,j)} I\frac{dx_{ij}(t)}{dt} + \sum_{(i,j)}\frac{dx_{ij}(t)}{dt}\frac{d}{dx_{ij}(t)}\int_0^{x_{ij}} m(s)ds. \quad (21)$$

According to the definition of the CNNs, one has

$$a_{ij,kl} = 0, b_{ij,kl} = 0 \text{ for } c(k,l) \notin N_r(i,j). \quad (22)$$

Afterward

$$\begin{aligned}\frac{dE(t)}{dt} &= -\sum_{(i,j)}\frac{dx_{ij}(t)}{dt}\left\{\sum_{(k,l)} a_{ij,kl}x_{kl}(t) + \sum_{(k,l)} b_{ij,kl}u_{kl}(t) + I - \frac{d}{dx_{ij}(t)}\int_0^{x_{ij}} m(s)ds\right\} \\ &= -\sum_{(i,j)}\frac{dx_{ij}(t)}{dt}\left\{\sum_{(k,l)}(a_{ij,kl}x_{kl}(t) + b_{ij,kl}u_{kl}(t)) + I - m(x_{ij}(t))\right\}. \end{aligned} \quad (23)$$

Substituting (7) into (23), and recalling $C > 0$, one can obtain

$$\frac{dE(t)}{dt} = -\sum_{(i,j)} C\left(\frac{dx_{ij}(t)}{dt}\right)^2 \le 0 \quad (24)$$

which means the energy function is monotonic decreasing. Meanwhile, since the energy function is bounded under certain

constraints (in Theorem 1), the state variable in the M-CNN is bounded. Thus, such a network always generates a dc output, in other words, the M-CNN is stable.

B. Fault Tolerance

In practice, some faulty cells may exist in a CNN because of device fault or other reasons. In general, a CNN is expected to work normally even with the presence of faulty cells, namely, being capable of fault tolerance. Theoretically, a CNN is considered to possess fault tolerance if it is still stable when some cells do not work. In this paper, the fault tolerance for the M-CNN is verified.

Definition: The stuck-at-α fault is defined that state of faulty cell is not changeable along with the inputs and outputs of other cells, where α is a constant with $|\alpha| \leq 1$ [26].

In image processing, the normalized binary output of a CNN processor (cell) can be 1 and -1 (or 0), which denote the white and the black pixels, respectively. A stuck-at-α faulty cell always keeps its state variable unchanged, and therefore, the corresponding pixel value is constant depending on the value of α.

Then, the dynamics of an M-CNN cell with a single stuck-at-α fault can be given by

$$C\frac{dx_{ij}(t)}{dt} = -m(x_{ij}(t)) + \sum_{c(k,l)} (\bar{a}_{ij,kl}x_{kl}(t) + b_{ij,kl}u_{kl}) + I + a_f\alpha \tag{25}$$

where a_f and $\bar{a}$ denote the faulty and the rest templates, respectively.

An energy function $V(t)$ for the M-CNN with single stuck-at-α fault is chosen as

$$V(t) = \sum_{(i,j)}\int_0^{x_{ij}} m(s)ds - \frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}\bar{a}_{ij,kl}x_{kl}(t)x_{ij}(t) - \sum_{(i,j)}\sum_{(k,l)} b_{ij,kl}u_{kl}(t)x_{ij}(t) - \sum_{(i,j)} Ix_{ij}(t) - \sum_{(i,j)} a_f\alpha x_{ij}(t). \tag{26}$$

Theorem 3: The energy function $V(t)$ for an $M \times N$ M-CNN is bounded by

$$\max|V(t)| \leq V_{\max} \tag{27}$$

where

$$V_{\max} = \frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}\left|\bar{a}_{ij,kl}x_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}\sum_{(k,l)}\left|b_{ij,kl}u_{kl}(t)x_{ij}(t)\right| + MN\left(|I| + \max_{i,j}\left|\int_0^{x_{ij}} m(s)ds\right| + |a_f\alpha|\right). \tag{28}$$

Proof: Based on the definition of $V(t)$ in (26), one has

$$\begin{aligned}
|V(t)| &\leq \frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}\left|\bar{a}_{ij,kl}x_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}\sum_{(k,l)}\left|b_{ij,kl}u_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}|Ix_{ij}(t)| + \sum_{(i,j)}\left|\int_0^{x_{ij}} m(s)ds\right| + \sum_{(i,j)}|a_f\alpha\cdot x_{ij}(t)| \\
&\leq \frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}\left|\bar{a}_{ij,kl}x_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}\sum_{(k,l)}\left|b_{ij,kl}u_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}|I|\cdot|x_{ij}(t)| + \sum_{(i,j)}\left|\int_0^{x_{ij}} m(s)ds\right| + \sum_{(i,j)}\left|a_f\alpha\right|\cdot|x_{ij}(t)| \\
&\leq \frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}\left|\bar{a}_{ij,kl}x_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}\sum_{(k,l)}\left|b_{ij,kl}u_{kl}(t)x_{ij}(t)\right| + MN(|I|) + MN\left(\max_{i,j}\left|\int_0^{x_{ij}} m(s)ds\right|\right) + MN(|a_f\alpha|) \\
&\leq \frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}\left|\bar{a}_{ij,kl}x_{kl}(t)x_{ij}(t)\right| + \sum_{(i,j)}\sum_{(k,l)}\left|b_{ij,kl}u_{kl}(t)x_{ij}(t)\right| + MN\left(|I| + \max_{i,j}\left|\int_0^{x_{ij}} m(s)ds\right| + |a_f\alpha|\right) \\
&= V_{\max}.
\end{aligned} \tag{29}$$

Therefore, $V(t)$ is bounded as declared in (27).

Theorem 4: The energy function $V(t)$ is monotonic decreasing, namely

$$\frac{dV(t)}{dt} \leq 0. \tag{30}$$

Proof: With the same assumption for the feedback template as in Theorem 2, the derivative of $V(t)$ with respect to time t is given by

$$\frac{dV(t)}{dt} = -\sum_{(i,j)}\sum_{(k,l)}\bar{a}_{ij,kl}x_{kl}(t)\frac{dx_{ij}(t)}{dt} - \sum_{(i,j)}\sum_{(k,l)} b_{ij,kl}u_{kl}(t)\frac{dx_{ij}(t)}{dt} - \sum_{(i,j)} I\frac{dx_{ij}(t)}{dt} + \sum_{(i,j)}\frac{dx_{ij}(t)}{dt}\frac{d}{dx_{ij}(t)}\int_0^{x_{ij}} m(s)ds - \sum_{(i,j)} a_f\alpha\frac{dx_{ij}(t)}{dt}. \tag{31}$$

0.8	0.7	1.0	-0.1
1.0	1.0	1.0	1.0
1.0	0.9	0.7	0.8
-0.1	1.0	0.8	1.0

(a)

0.8	1.0	1.0	0.6
1.0	1.0	1.0	1.0
-1.0	0.9	-1.0	-0.8
-0.9	-1.0	-0.7	-0.8

(b)

-0.8	1.0	-0.1	-0.6
1.0	1.0	1.0	-0.1
-1.0	0.9	-1.0	-0.8
-0.9	-1.0	-0.7	-0.8

(c)

-0.9	-1.0	1.0	1.0
-1.0	1.0	-1.0	1.0
1.0	-1.0	0.7	0.8
0.9	1.0	0.8	1.0

(d)

-0.9	-1.0	-0.9	-1.0
-1.0	1.0	-1.0	-1.0
1.0	-1.0	1.0	1.0
0.7	1.0	1.0	0.8

(e)

-0.8	-0.9	-1.0	-0.6
-1.0	1.0	-1.0	-1.0
-1.0	-0.8	-1.0	-0.8
-0.9	-1.0	-0.7	-0.8

(f)

Fig. 9. (a)–(f) Six different sets of initial conditions in which the initial states of cell c(2, 2) are the same [1].

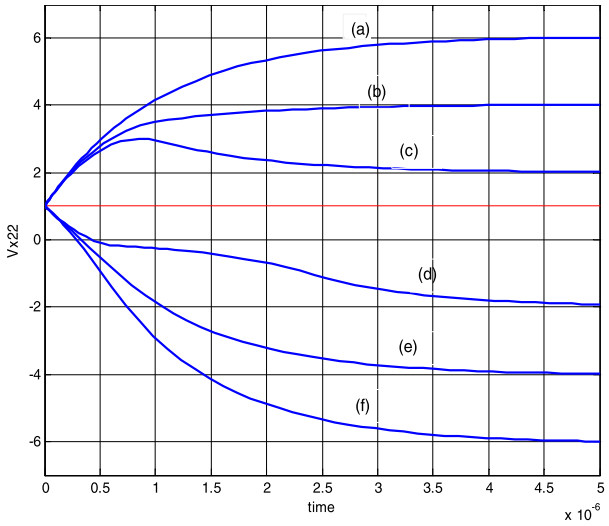

Fig. 10. The transient behaviors of cell c(2,2) in the memristor-based CNN. Curves (a)–(f) denote the transient behaviors of the corresponding initial conditions in (a)–(f) in Fig.9, respectively.

Similar to the derivation of (23), one can obtain the following from (31) as:

$$\frac{dV(t)}{dt} = -\sum_{(i,j)} \frac{dx_{ij}(t)}{dt} \left\{ \sum_{(k,l)} \bar{a}_{ij,kl} x_{kl}(t) + \sum_{(k,l)} b_{ij,kl} u_{kl}(t) + I - \frac{d}{dx_{ij}(t)} \int_0^{x_{ij}} m(s)ds + a_f \alpha \right\}$$

$$= -\sum_{(i,j)} \frac{dx_{ij}(t)}{dt} \left\{ -m(x_{ij}(t)) + \sum_{(k,l)} (\bar{a}_{ij,kl} x_{kl}(t) + b_{ij,kl} u_{kl}(t)) + I + \alpha_f \alpha \right\}. \quad (32)$$

Substituting (25) into (32), and recalling $C > 0$, one can obtain

$$\frac{dV(t)}{dt} = -\sum_{(i,j)} C \left(\frac{dx_{ij}(t)}{dt} \right)^2 \leq 0. \quad (33)$$

Then, it can be concluded that the M-CNN with a single stuck-at-α fault is still stable, which indicates the fault tolerance property.

V. Computer Simulations

In this section, the M-CNN has been numerically analyzed on software MATLAB, including its stability, fault tolerance property, and two typical applications in image processing.

A. Stability Analysis

The transient behavior of the cell $c(2,2)$ in a 4×4 M-CNN subjected to six sets of initial conditions (shown in Fig. 9) are shown in Fig. 10. It can be observed that the output (state) of the M-CNN cells can reach stability and keep it ultimately. This is in accordance with the theoretical derivation in Section IV and essential for sequent applications. In this simulation, the templates and the offset current is chosen as

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \quad B = 0, \quad I = 0. \quad (34)$$

B. Analysis of Fault Tolerance Property

To verify the fault tolerance property, a 21×21 M-CNN is considered to implement the image inversion function. The corresponding standard templates and the offset current (or threshold) are chosen as

$$A = \begin{bmatrix} 0 & 2 & 0 \\ 0 & 2 & 0 \\ 0 & 2 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad I = 4. \quad (35)$$

Fig. 11 shows the time evolvement for image inversion of the M-CNN with a single stuck-at-0 faulty cell c(8,5). It can be observed that although there is a stuck-at-0 faulty cell, the M-CNN can still accomplish the image inversion task satisfactorily, i.e., inversing the original black diamond into the white one successfully. It should be mentioned that the black-line boundary of each subfigure is added to present the simulation results better, but it does not belong to the objective image.

C. Applications of the M-CNN

Two kinds of M-CNNs are used to perform several typical applications in image processing: 1) uncoupled M-CNN for HLD and edge extraction and 2) coupled M-CNN for noise removal.

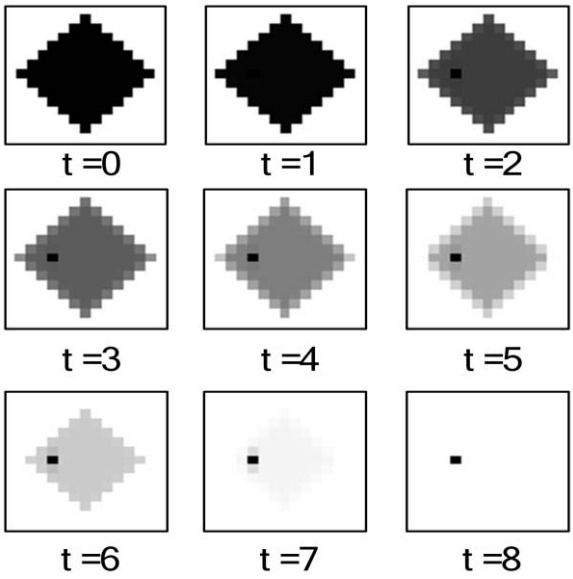

Fig. 11. Image inversion of the M-CNN with a single stuck-at-0 faulty cell at (8, 5).

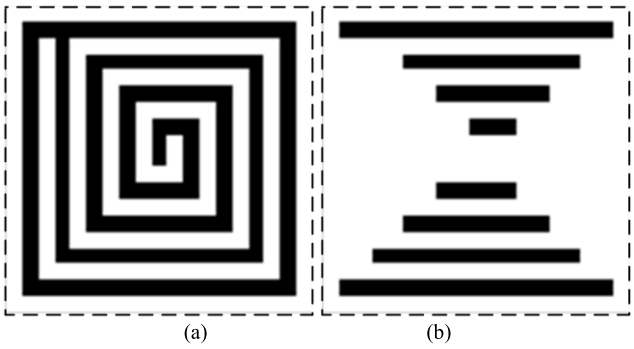

Fig. 12. HLD with uncoupled M-CNN. (a) Input and (b) output images.

1) Uncoupled M-CNN for HLD and Edge Extraction: A CNN is considered as uncoupled if all of the elements of the feedback template A are zeroes or only with one nonzero element at center indicating self-feedback [9]. HLD is one of the functions an uncoupled CNN can complete. It means that the uncoupled CNN will horizontally delete isolated points and the remaining horizontal lines should contain at least two points. For this example, the templates and the offset current are given by [21]

$$A = [0\ 1\ 0], \quad B = [1\ 1\ 1], \quad I = -1. \tag{36}$$

Fig. 12(a) and (b) shows the original image and the result, respectively.

Another application is edge extraction, which is a common operation in image processing. Edge extraction is an important case of feature extraction because the edges of an image contain most of the information regarding the shape of the image. Here, using the templates in (37), a practical example of a license plate [shown in Fig. 13(a)] was, respectively, input into the M-CNN and the standard CNN to get its edges extracted. The processed results are shown in Fig. 13(b) and (c),

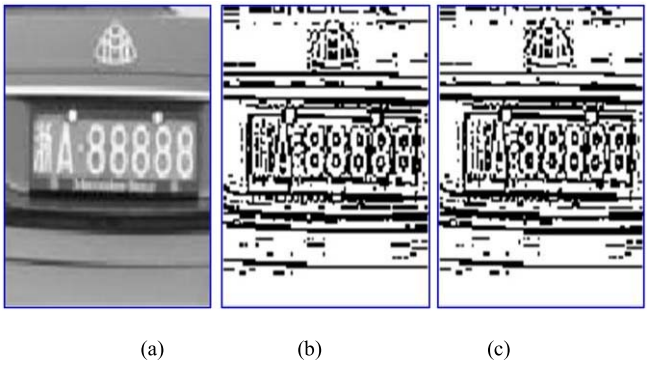

Fig. 13. Illustration of image processing performance for edge extraction. (a) Original (input) images. (b) Output of the M-CNN. (c) Output of a traditional CNN.

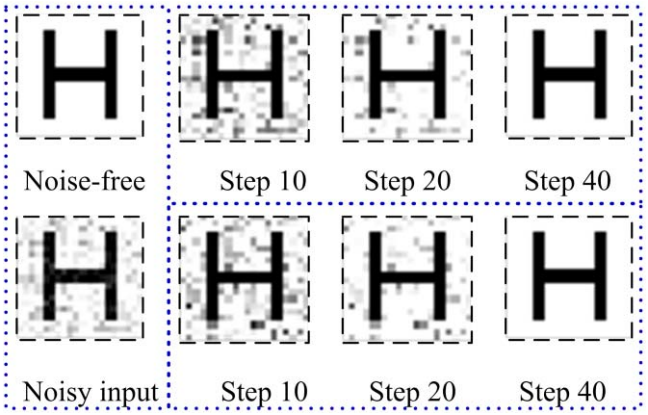

Fig. 14. Image processing performance for noise removal is illustrated. The left two image are noise-free image and input image with 0.02 Gaussian white noise. On the right, the first row presents the process of the M-CNN and the second row shows that of the standard CNN.

respectively

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}, \quad I = -1. \tag{37}$$

Fig. 13(a)–(c) shows the original image and the processed results obtained from the M-CNN and the standard CNN, respectively. Note that in the edge extraction the feedback template elements are all zeroes, which means there is no feedback including self-feedback needed. The experiential results have also verified that the M-CNN model possesses similar processing performance in image processing with the standard CNN: the average pixel difference between Fig. 13(b) and (c) is 1.47%.

2) Coupled M-CNN for Noise Removal ($\sigma = 0.02$): In CNNs, linear image processing with feedback and control templates is equivalent to spatial convolution with infinite-impulse response kernels [3]. For the image that is obtained from the real world by a camera or some other equipment, it is inevitable to be polluted by some noise. Thus, noise removal is a necessary and important operation. Now, we consider a 19×19 image (top left corner of Fig. 14), which is polluted by Gaussian white noise with variance $\sigma = 0.02$,

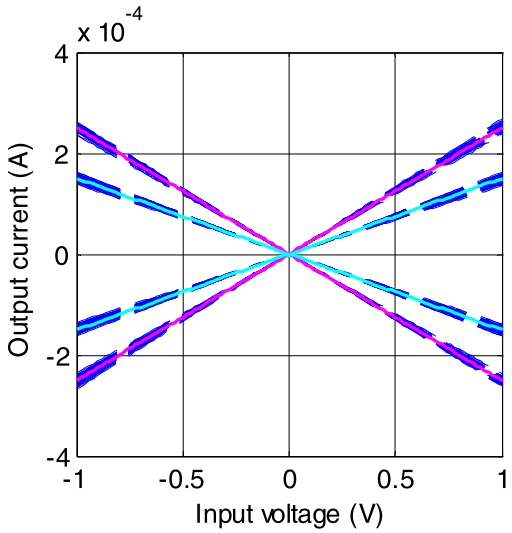

Fig. 15. Synaptic weight processing via the memristor bridge circuit with memristor variations.

its mean value $m = 0$ (bottom left corner), as the input of a 19×19 M-CNN and a standard CNN, respectively. The processed results are illustrated on the first and the second rows correspondingly. The employed templates and threshold are given by (38). This simulation indicates that the M-CNN has satisfactory processing performance for noise removal as that of the standard CNN even with much simpler circuit structure

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad I = 0. \qquad (38)$$

D. Impact of Variations of the Memristors on the M-CNN

The variations of the memristor synaptic weights may result from the process variation on memristors, the weight-setting inaccuracy, and weighting operation influence, and so on. Taking the four ideal lines (corresponding to four levels of weights), for instance, assume that the overall variation impacts on the memristor weights follow a Gaussian distribution with variance $\sigma = 0.02$, 100 lines denoting the deviations of the ideal weights can be generated, as shown in Fig. 15.

In this section, we take the HLD, for example, to evaluate the performance of the M-CNN under variations in the memristors. Since for the CNNs, the image processing is a procedure of iteratively finding stable solution, we at first assume that during each iteration, the memristor synaptic weight (templates) varies following a Gaussian distribution with variance $\sigma = 0.02$ and then Monte-Carlo simulations are conducted 100 times for which the results are shown in Fig. 16.

Furthermore, Fig. 17 shows the average error (blue square) and the average running time (green star) of 100 Monte-Carlo simulations versus the increasing memristor weights variation. It can be observed that the variations of the memristor weights do not influence the processing result very much, because the small deviation is negligible compared with the binary quantization of the output signals. For instance, even with

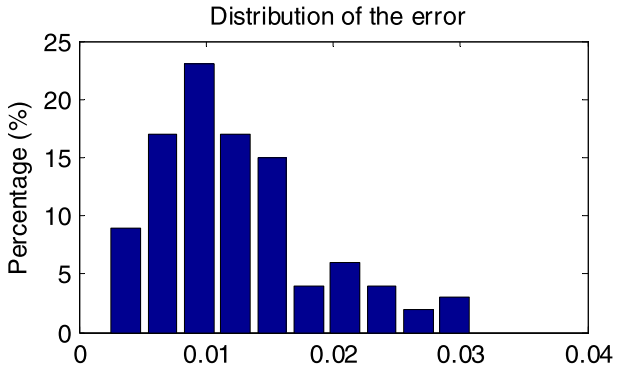

Fig. 16. Histogram of the errors of 100 times of simulations for HLD with memristor weights variations.

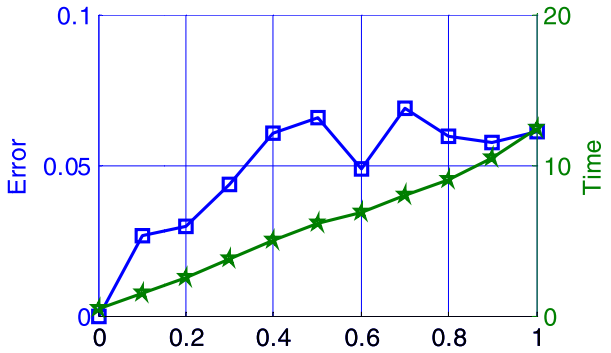

Fig. 17. Average error (blue square) and average running time (green star) in seconds versus memristor weights variation's variance in 100 simulations for HLD.

variance of 1.0, the finial output is the same as the original one. However, since the image processing is a series of iterative operations, if some variations arise in the templates, the process will thus need more iterations to get the stable solution.

VI. Conclusion

The memristor has been extensively studied in biological sciences and electrical engineering as a means to compactly implement synaptic function in neural networks. This paper presents a novel implementation scheme for M-CNNs along with detailed mathematical analysis. In the proposed M-CNN, memristors are employed not only to implement the synaptic weights (the templates) in programmable bridge structure, but also to replace the original linear state resistor. The weight programming and weighing process can both be performed in the memristor bridge circuit, and the original sigmoid output circuit can be eliminated to further reduce the size of the PE as well as to improve the speed of computation. Therefore, the proposed architecture is more compact and versatile, as well as suitable for VLSI implementation.

Mathematical analysis including stability and fault tolerance has been presented using the Lyapunov theorem, thereby providing a theoretical foundation for M-CNN implementation. The simulation results indicate the performance of M-CNN in image processing for numerous common functions such as HLD, edge extraction, and noise removal. Finally, the

effects of variations of memristors characteristics have been investigated by performing Monte-Carlo simulation. It was observed that variations of memristor characteristics influence the performance of an M-CNN somewhat in speed without impairing the quality of the final results.

REFERENCES

[1] L. O. Chua and L. Yang, "Cellular neural networks: Theory," *IEEE Trans. Circuits Syst.*, vol. CAS-35, no. 10, pp. 1257–1272, Oct. 1988.

[2] L. O. Chua and L. Yang, "Cellular neural networks: Applications," *IEEE Trans. Circuits Syst.*, vol. CAS-35, no. 10, pp. 1273–1290, Oct. 1988.

[3] L. O. Chua and T. Roska, *Cellular Neural Networks and Visual Computing*. Cambridge, U.K.: Cambridge Univ. Press, 2002, pp. 272–275.

[4] T. Yang, *Cellular Neural Network and Image Processing*, 1st ed. Commack, NY, USA: Nova, 2002.

[5] R. Dominguez-Castro *et al.*, "A 0.8-μm CMOS two-dimensional programmable mixed-signal focal-plane array processor with on-chip binary imaging and instructions storage," *IEEE J. Solid-State Circuits*, vol. 32, no. 7, pp. 1013–1026, Jul. 1997.

[6] J. M. Cruz and L. O. Chua, "A 16 $\times$ 16 cellular neural network universal chip: The first complete single-chip dynamic computer array with distributed memory and with gray-scale input-output," *Analog Integr. Circuits Signal Process*, vol. 15, no. 3, pp. 227–237, 1998.

[7] P. Mazumder, S.-R. Li, and I. E. Ebong, "Tunneling-based cellular nonlinear network architectures for image processing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 17, no. 4, pp. 487–495, Apr. 2009.

[8] W. H. Lee and P. Mazumder, "Motion detection by quantum-dots-based velocity-tuned filter," *IEEE Trans. Nanotechnol.*, vol. 7, no. 3, pp. 357–362, May 2008.

[9] S.-R. Li, P. Mazumder, and L. O. Chua, "Cellular neural/nonlinear networks using resonant tunneling diode," in *Proc. 4th IEEE Conf. Nanotechnol.*, Aug. 2004, pp. 164–167.

[10] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *Nature*, vol. 453, no. 7191, pp. 80–83, 2008.

[11] R. Williams, "How we found the missing memristor," *IEEE Spectr.*, vol. 45, no. 12, pp. 28–35, Dec. 2008.

[12] L. O. Chua, "Memristor—The missing circuit element," *IEEE Trans. Circuit Theory*, vol. 18, no. 5, pp. 507–519, Sep. 1971.

[13] M. D. Pickett *et al.*, "Switching dynamics in titanium dioxide memristive devices," *J. Appl. Phys.*, vol. 106, no. 7, p. 074508, 2009.

[14] H. Abdalla and M. D. Pickett, "SPICE modeling of memristors," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2011, pp. 1832–1835.

[15] S. Kvatinsky, E. G. Friedman, A. Kolodny, and U. C. Weiser, "TEAM: Threshold adaptive memristor model," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 1, pp. 211–221, Jan. 2013.

[16] F. Alibart, L. Gao, B. D. Hoskins, and D. B. Strukov, "High precision tuning of state for memristive devices by adaptable variation-tolerant algorithm," *Nanotechnology*, vol. 23, no. 7, p. 075201, 2012.

[17] X. Hu, S. Duan, L. Wang, and X. Liao, "Memristive crossbar array with applications in image processing," *Sci. China, Inform. Sci.*, vol. 55, no. 2, pp. 461–472, 2012.

[18] S. Duan, X. Hu, L. Wang, C. Li, and P. Mazumder, "Memristor-based RRAM with applications," *Sci. China, Inform. Sci.*, vol. 55, no. 6, pp. 1446–1460, 2012.

[19] S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder, and W. Lu, "Nanoscale memristor device as synapse in neuromorphic systems," *Nano Lett.*, vol. 10, no. 4, pp. 1297–1301, 2010.

[20] X. Hu, S. Duan, and L. Wang, "A novel chaotic neural network using memristive synapse with applications in associative memory," *Abstract Appl. Anal.*, vol. 2012, pp. 1–19, Nov. 2012, doi: 10.1155/2012/405739.

[21] H. Kim, M. P. Sah, C. Yang, T. Roska, and L. O. Chua, "Memristor bridge synapses," *Proc. IEEE*, vol. 100, no. 6, pp. 2061–2070, Jun. 2012.

[22] S. P. Adhikari, C. Yang, H. Kim, and L. O. Chua, "Memristor bridge synapse-based neural network and its learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 9, pp. 1426–1435, Sep. 2012.

[23] F. Corinto, A. Ascoli, Y.-S. Kim, and K.-S. Min, "Cellular nonlinear networks with memristor synapses," in *Memristor Networks*. New York, NY, USA: Springer-Verlag, 2014, pp. 267–291.

[24] L. Wang, E. Drakakis, S. Duan, and P. He, "Memristor model and its application for chaos generation," *Int. J. Bifurcation Chaos*, vol. 22, no. 8, p. 1250205, 2012.

[25] S. Liu, L. Wang, S. Duan, C. Li, and J. Wang, "Memristive device based filter and integration circuits with applications," *Adv. Sci. Lett.*, vol. 8, no. 1, pp. 194–199, 2012.

[26] L. Wang, X. Yang, and S. Duan, "Analysis of fault tolerance of cellular neural networks and applications to image processing," in *Proc. 3rd Int. Conf. Natural Comput.*, Washington, DC, USA, 2007, pp. 252–256.

Shukai Duan (M'10) received the Ph.D. degree in computer science from Chongqing University, Chongqing, China, in 2006.

He has been with the College of Electronic and Information Engineering, Southwest University, Chongqing, since 1996, where he was an Associate Professor from 2005 to 2009, and a Post-Doctoral Researcher from 2007 to 2009, and has been a Professor since 2010. He was a Visiting Professor with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, USA, from 2010 to 2011, and University of Windsor, Windsor, ON, Canada, in 2013. He is currently a Ph.D. Candidate Supervisor and leading a research group of three Ph.D. students and 15 master students. He has authored four books and more than 100 papers in refereed journals and conferences. His current research interests include memristor devices and memrisitive systems, nonlinear circuits and systems, artificial neural networks, chaos and chaotic circuit, and intelligent signal processing. He has taken charge of more than 10 national, provincial or ministry level research projects, including the National Natural Science Foundation of China and the Program for New Century Excellent Talents through the Ministry of Education.

Dr. Duan serves as an IEEE CIS Member.

Xiaofang Hu (S'14) received the B.E. and M.E. degrees in electronics and information engineering from the Southwest University, Chongqing, China, in 2009 and 2012, respectively. She is currently pursuing the Ph.D. degree in mechanical and biomedical engineering with the City University of Hong Kong, Hong Kong.

Her current research interests include memristor and memrisitive systems, artificial neural networks, intelligent control, and design and analysis of chaotic systems with memristor.

Zhekang Dong (S'14) received the B.S. degree in electronics from Southwest University, Chongqing, China, in 2012, where he is currently pursuing the degree with the School of Electronic and Information Engineering.

His current research interests include memristor and memristive systems, artificial neural networks, and design and analysis of nonlinear systems based on memristor and computer simulation.

Lidan Wang (M'10) received the B.E. degree in automatic control from the Nanjing University of Science and Technology, Nanjing, China, in 1999, and the Ph.D. degree in computer software and theory from the Chongqing University, Chongqing, China, in 2008.

She is currently a Professor with the School of Electronics and Information Engineering, Southwest University, Chongqing, China. She has authored more than 20 academic papers in electronic circuits and chaotic communications. Her current research interests include the nonlinear system and chaotic circuit, artificial neural networks and FPGA technology, and memristor and memristive systems.

Pinaki Mazumder (F'99) received the Ph.D. degree from the University of Illinois at Urbana-Champaign, Champaign, IL, USA, in 1988.

He was the Lead Program Director with the Emerging Models and Technologies Program, University of Michigan, Ann Arbor, MI, USA, through the U.S. National Science Foundation. He has served in Industrial Research and Development Centers, including AT&T Bell Laboratories, Murray Hill, NJ, USA, where he started the CONES Project entitled the first C modeling-based VLSI synthesis tool, and India's premiere electronics company, Bharat Electronics Ltd., Bangalore, India, in 1985, where he had developed several high-speed and high-voltage analog integrated circuits intended for consumer electronics products. He is currently a Professor with the Department of Electrical Engineering and Computer Science, University of Michigan. He has authored more than 200 technical papers and four books on various aspects of very-large-scale integration (VLSI) research works. His current research interests include current problems in nanoscale CMOS VLSI design, CAD tools, and circuit designs for emerging technologies, including Quantum MOS and resonant tunneling devices, semiconductor memory systems, and physical synthesis of VLSI chips.

Dr. Mazumder was a recipient of the Digital's Incentives for Excellence Award, BF Goodrich National Collegiate Invention Award, and DARPA Research Excellence Award. He is an AAAS Fellow.