A Learnable Cellular Neural Network Structure With Ratio Memory for Image Processing

Chung-Yu Wu, *Fellow, IEEE,* and Chiu-Hung Cheng, *Student Member, IEEE*

***Abstract*—In this paper, a learnable cellular neural network (CNN) with space-variant templates and ratio memory (RM) called the RMCNN, is proposed and analyzed. By incorporating both modified Hebbian learning rule and RM into CNN architecture, the RMCNN as the associative memory can generate the absolute weights and then transform them into the ratioed A-template weights as the ratio memories for recognition of noisy input patterns. It is found from simulation results that due to the feature enhancement effect of RM, the RMCNN under constant leakage on template coefficients can store and recognize more patterns than the CNN associative memories without RM, but with the same learning rule and the same constant leakage on space-variant template coefficients. For 9×9 (18×18) RMCNNs, three (five) patterns can be learned, stored and recognized. Based upon the RMCNN architecture, an experimental chip of CMOS 9×9 RMCNN is designed and fabricated by using 0.35 μm CMOS technology. The measurement results have successfully verified the correct functions of RMCNN.**

***Index Terms*—Cellular neural network (CNN), divider, multiplier, ratio memory (RM).**

I. Introduction

DUE to the advantageous feature of local connectivity, the cellular neural network (CNN) introduced by Chua and Yang [1] is very suitable for very large-scale integration (VLSI) implementation and thus enables many applications [2], [3]. So far, many research works on the applications of CNNs as neural associative memories for pattern learning, recognition and association have been explored [4]–[10]. Among them, many innovative algorithms and software simulations of CNN associated memories were reported [4]–[8]. As to the hardware implementation, special learning algorithms and digital hardware implementation for CNNs were proposed in [9] to solve the sensitivity problems caused by the limited precision of analog weights. Moreover, CMOS chip implementation of CNN associative memory was also reported in [10].

In realizing CNN associative memories, the learning circuitry can be integrated on-chip with CNNs. The major advantages of on-chip learning are: 1) no host computer is needed to perform the learning task off-line. This makes the interface of neural system chips simple for many practical applications; 2) the space-variant template coefficients can be on-chip learned without being loaded from outside to the CNN chips. Thus, long loading time, complex cell global interconnection, and analog coefficient storage elements to perform the loading operation for large numbers of space-variant template coefficients can be avoided; 3) the adaptability to the process variations of CNN chips can be enhanced.

The ratio memory (RM) of Grossberg outstar structure [11]–[13] has been used in both feedforward and feedback neural network ICs for image processing [14]–[17]. It is found that the RM in neural network ICs has the advantages of long memory time and image feature enhancement under constant leakage on stored weights [14]–[17].

In this paper, both RM and modified Hebbian learning function [18] are implemented in the CNN structure with space-variant templates and constant leakage on stored template coefficients [19] for pattern learning, storing and recognition. The proposed CNN with RM is called the RMCNN. It has the advantages of on-chip learning as mentioned above. Since most of the on-chip learning circuits can be shared with both RM and CNN core circuits, the extra chip area required for on-chip learning circuits is small. Moreover, the RMCNN can have longer template-coefficient storage time or equivalently pattern recognition time which is one of the advantages of RM. Due to the feature enhancement effect of the RM which well separates the learned weights and decreases the insignificant weights to zero, more patterns can be stored and recognized in the RMCNN as compared to the CNN associative memory without RM, but with space-variant template coefficients, the same constant leakage on template coefficients and the same learning rule. As a demonstrative example, a 9×9 RMCNN is realized in CMOS technology. Both simulation and experimental results have verified the advantageous characteristics of the RMCNN.

In Section II, both model and architecture of the RMCNN are described. In Section III, the detailed CMOS circuit design is presented. In Section IV, both MATLAB and HSPICE simulation results are demonstrated to verify the correct functions of the RMCNN. The measurement results are presented in Section V. Finally, the conclusion is given.

II. Model and Architecture

In a CNN, the cell state $X_{ij}(t)$, its derivative $\dot{X}_{ij}(t)$ and the cell output $Y_{ij}(t)$ for a regular cells can be expressed as [1]–[3]

$$\dot{X}_{ij}(t) = -X_{ij}(t) + \sum_{C(k,l)\in Nr(i,j)} a_{ijkl}(t)Y_{kl}(t) + \sum_{C(k,l)\in Nr(i,j)} b_{ijkl}(t)u_{kl} + Z_{ij} \quad (1)$$

Manuscript received February 3, 2001; revised July 12, 2001. This work was supported by the National Science Council of the Republic of China, under Grant NSC89-2218-E-009-064. This paper was recommended by Associate Editor P. Szolgay.

The authors are with the Integrated Circuits and Systems Laboratory, Department of Electronics Engineering, National Chiao Tung University, Hsinchu City, 30050 Taiwan, R. O. C. (e-mail: cywu@alab.ee.nctu.edu.tw; p8611828@alab.ee.nctu.edu.tw).

Digital Object Identifier 10.1109/TCSI.2002.805697

$$Y_{ij}(t) = f[X_{ij}(t)] = \begin{cases} X_{ij}(t), & \text{if } -1 \le X_{ij}(t) \le +1 \\ 1, & \text{if } X_{ij}(t) > +1 \\ -1, & \text{if } X_{ij}(t) < -1 \end{cases} \quad (2)$$

where $Y_{kl}(t)$ is the cell output from the cell $C(k,l)$ in the r-neighborhood system $Nr(i,j)$ of the cell $C(i,j)$ and u_{kl} is the cell input from the cell $C(k,l)$ of $Nr(i,j)$. In an $M \times N$ CNN cell array, the r-neighborhood system $Nr(i,j)$ of the cell $C(i,j)$ is defined as the set of all cells including $C(i,j)$ and its neighboring cells, which satisfy the following property:

$$Nr(i,j) = \Big\{ C(k,l) | 1 \le k \le M, 1 \le l \le N; \; |k-i| + |l-j| \le r \Big\} \quad (3)$$

where r is an integer and called the radius or the number of neighborhood layer. In (1), $a_{ijkl}(t)$ is the coefficient or weight of $\mathbf{A}$ template which correlates $Y_{kl}(t)$ to $X_{ij}(t)$, $b_{ijkl}(t)$ is the coefficient or weight of $\mathbf{B}$ template that correlates u_{kl} to $X_{ij}(t)$ and Z_{ij} is the bias or threshold of the cell $C(i,j)$. In many applications of CNNs, the template coefficients of $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{Z}$ templates are usually constants, being independent of time and cell position. However, these template coefficients can be time-variant or space-variant in general. In (2), the output function $f[X_{ij}(t)]$ called the ramp function, is a nonlinear function which limits the maximum absolute output to be 1.

To incorporate the learning capability into a CNN, its structure has to be modified to realize a learning rule and variable templates. In this work, the Hebbian rule for unsupervised learning [19] is adopted with the necessary modification to accommodate the local connectivity of CNNs. Suppose there are m exemplar patterns to be learned in a learnable CNN. The learned weight $zi_{ijkl}(0)$ at $t = 0$ in the learning period can be determined according the modified Hebbian rule by the summation of all products of two activations or outputs from two correlated pixels or nodes. Thus, $zi_{ijkl}(0)$ can be written as

$$zi_{ijkl}(0) = \sum_{p=1}^{m} u_{ij}^p u_{kl}^p \quad C(k,l) \in N_r^0(i,j) \quad (4)$$

where u_{ij}^p is the pixel activation or output of the cell $C(i,j)$ at ith row and jth column of the pth pattern out of m input patterns with the normalized value between $+1$ and -1, u_{kl}^p is the pixel activation or output of the cell $C(k,l)$ in the set of $N_r^0(i,j)$ and $N_r^0(i,j)$ is the set of r-neighboring cells $N_r(i,j)$ without the cell $C(i,j)$. The learned weight is then transformed into the ratioed weight $w_{ijkl}(0)$ [14]–[17] as

$$w_{ijkl}(0) = K \frac{zi_{ijkl}(0)}{\sum_{C(k,l) \in N_r^0(i,j)} |zi_{ijkl}(0)|} \quad (5)$$

where K is a constant. With the ratioed weight in (5), the RM[14]–[17] can be realized in a CNN. The resultant CNN with modified Hebbian rule and RM is called the RMCNN. In the RMCNN, the cell state $X_{ij}(t)$ is written as

$$\dot{X}_{ij}(t) = -X_{ij}(t) + \sum_{C(k,l) \in N_r^0(i,j)} w_{ijkl}(t) Y_{kl}(t) + K' u_{ij}^* + Z_{ij} \quad (6)$$

where u_{ij}^* is the input of the patterns to the RMCNN for processing and K' is a constant.

As compared with the original CNN cell state $X_{ij}(t)$ in (1), the template coefficient $a_{ijkl}(t)$ can be expressed as

$$a_{ijkl}(t) = w_{ijkl}(t) = K_A \frac{zi_{ijkl}(t)}{\sum_{C(k,l) \in N_r^0(i,j)} |zi_{ijkl}(t)|} \quad (7)$$

where $K_A = K$ is a constant. Note that the $\mathbf{A}$ template is time variant and space variant. For a RMCNN with $r = 1$ and four nearest neighboring cells, the learned $\mathbf{A}$ template of the cell $C(i,j)$ at $t = 0$ just after the learning is denoted as $\boldsymbol{A}_{ij}^1(0)$ and can be expressed as

$$\boldsymbol{A}_{ij}^1(0) = K_A^1 \begin{bmatrix} 0 & a_{ij(i-1)j} & 0 \\ a_{iji(j-1)} & 0 & a_{iji(j+1)} \\ 0 & a_{ij(i+1)j} & 0 \end{bmatrix} \quad (8)$$

$$a_{ijkl} = \frac{\sum_{p=1}^{m} \int_{T_P} u_{ij}^p u_{kl}^p dt}{sum1}$$

$$kl = (i-1)j,\ i(j-1),\ i(j+1),\ (i+1)j \quad (9)$$

$$sum1 = \sum_{kl} \left| \sum_{p=1}^{m} \int_{T_P} u_{ij}^p u_{kl}^p dt \right| \quad (10)$$

where T_P is the learning time for the RMCNN to learn pth pattern and the total learning time for the RMCNN to learn m patterns is $T_L = \sum_{p=1}^{m} T_P$.

On the other hand, the $\mathbf{B}$ template can be written as

$$\mathbf{B}_{ij}^1(t) = K_B^1 \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad \text{for } r = 1 \quad (11)$$

where K_B^1 is a constant. As may be seen from (11), the $\mathbf{B}$ template is a static and space-invariant matrix. Similarly, the threshold template Z_{ij} is also a space-invariant and static template. The boundary conditions of the boundary cells in the RMCNN can be written as

$$X_{i^*j^*}(t) = 0,\ u_{i^*j^*}(t) = 0 \quad (12)$$

where i^*j^* denote the boundary cells. The initial state $X_{ij}(0)$ of the RMCNN is set as

$$X_{ij}(0) = 0, \qquad 1 \le i \le M, 1 \le j \le N. \quad (13)$$

The architecture of the RMCNN is shown in Fig. 1 where the RM is used to realize the $\mathbf{A}$- template weights among the cells and there are only four nearest neighboring cells. The detailed block diagram of two neighboring CNN cells and their RM in the RMCNN is shown in Fig. 2. In Fig. 2, the block T1 is a V-I converter used to convert the voltage of input patterns into current. The current of input patterns is summed with the four weighted outputs from neighboring cells during the recognition period and converted into voltage through the resistor R_{ij} and the parasitic capacitor C_{ij} to form the cell state $X_{ij}(t)$. The block T2d is a V-I converter with one-half absolute-value circuit and sign-detection circuit to generate the absolute value of output current and detect the signs of $X_{ij}(t)$, respectively. The CNN cell $C(i,j)$ is formed by T1, T2d, R_{ij}, and C_{ij} as indicated in Fig. 2.

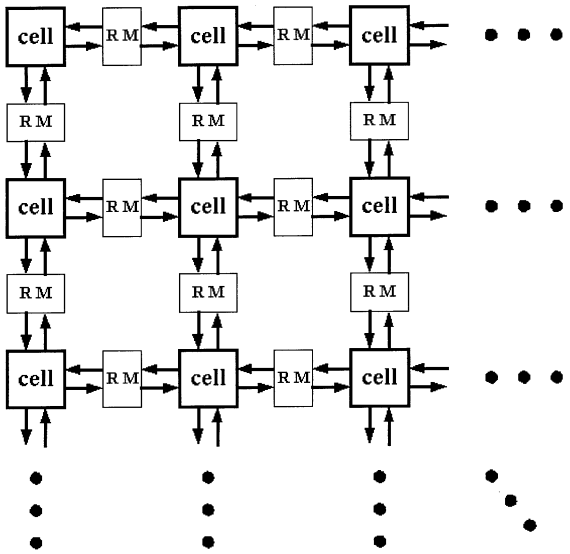


Fig. 1. The architecture of the RMCNN.

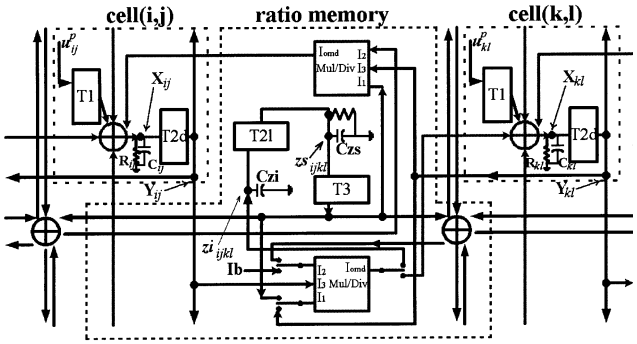


Fig. 2. The detailed architecture of two neighboring cells and their RMs in the RMCNN.

The block Mul/Div [20] in Fig. 2 is a combined four-quadrant multiplier and two-quadrant divider circuit. It is used to perform the multiplication of (4) and realize the modified Hebbian learning rule during the learning period. It is also used to realize both RM in (5) and multiplication of $w_{ijkl}(t)^* Y_{kl}(t)$ in (6) during the recognition period. The resultant absolute weight zi_{ijkl} during the learning period is stored in the capacitor Czi. The block T2l transfers the absolute value of the voltage stored in Czi to Czs and stores its sign in the latch circuit. The resistor in parallel with Czs represents the inevitable leakage associated with Czs. The block T3 is also a V-I converter to convert the voltage of Czs into current during the recognition period. The output current of T3 is sent to the sum block to perform the summing function with the currents from the other three neighboring cells. The summed current is sent to the Mul/Div block for the ratio-memory generation. The above circuits form the RM among CNN cells.

During the learning period, with $\phi_1 = 1$ and $\phi_2 = 0$, the configuration of RMCNN is shown in Fig. 3. In Fig. 3, the m input patterns are read sequentially into the cell $C(i,j)$ and the input voltage Vu_{ij}^p of the pth input pattern is sent to T1 to convert into the current Iu_{ij}^p and then to T2d to extract its absolute current value $|Iu_{ij}^p|$ and sign. Then, the converted absolute

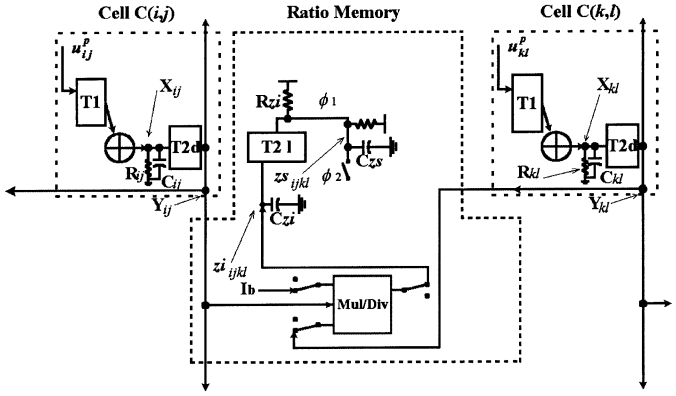


Fig. 3. Architecture the RMCNN during the learning period.

currents $|Iu_{ij}^p|$ and $|Iu_{kl}^p|$ of two neighboring cells are sent to the four-quadrant multiplier in the Mul/Div block to generate the signed product. The generated product in the current mode charges the capacitor Czi for the period T_P to form the voltage on Czi. This operation is repeated for m patterns to sum the voltages on Czi. Finally, the weight voltage $Vzi_{ijkl}(0)$ stored on Czi at $t = 0$ when the learning period ends can be written as

$$Vzi_{ijkl}(0) = \frac{1}{Czi}\sum_{p=1}^{m}\left(\int_{T_P}\frac{Iu_{ij}^p Iu_{kl}^p}{Ib}dt\right) C(k,l) \in N_r^0(i,j) \tag{14}$$

where Iu_{ij}^p is the current of the pixel at ith row and jth column of the pth pattern out of m input patterns, Iu_{kl}^p is the current of the input pattern to the cell $C(k,l)$ of $N_r^0(i,j)$ neighboring cells, Ib is a constant bias current, $Vzi_{ijkl}(0)$ is the weight voltage stored on Czi at $t = 0$ s and T_P is the learning time of each learned patterns. Comparing (14) and (4), it can be realized that the learned weight of the modified Hebbian rule is realized by (14). Through T2l, the absolute value of the weight $Vzi_{ijkl}(0)$ denoted as $abs[Vzi_{ijkl}(0)]$ is stored on the capacitor Czs, whereas the sign of $Vzi_{ijkl}(0)$ is stored in the latch circuit of T2l.

In the elapsed period, the leakage current I_{leakage} associated with Czs gradually decreases $abs[Vzi_{ijkl}(0)]$ of Czs. Since the leakage current is nearly constant, the change of $abs[Vzs_{ijkl}(t)]$ can be written as

$$abs\left[Vzs_{ijkl}(t)\right] = abs\left[Vzs_{ijkl}(0)\right] - \frac{I_{\text{leakage}}}{Czs}t. \tag{15}$$

In the recognition period with $\phi_1 = 0$ and $\phi_2 = 1$, the architecture of RM is shown in Fig. 4. In Fig. 4, the voltage Vu_{ij}^{*p} of the pth pattern to be recognized is input to T1 and converted into the current Iu_{ij}^{*p}. The absolute weight voltage $abs[Vzs_{ijkl}(t)]$ stored on Czs is converted into the current $abs[Izs_{ijkl}(t)]$ through T3 and summed with the currents from other neighboring cells. The summed current, the current $abs[Izs_{ijkl}(t)]$ and the cell output current $IY_{kl}(t)$ are sent to the Mul/Div block to obtain the current corresponding to $w_{ijkl}(t)Y_{kl}(t)$ in (6), which is then summed with the currents from other neighboring cells, the input current Iu_{ij}^{*p} and the threshold current Iz_{ij} to form the cell state current $Ix_{ij}(t)$.

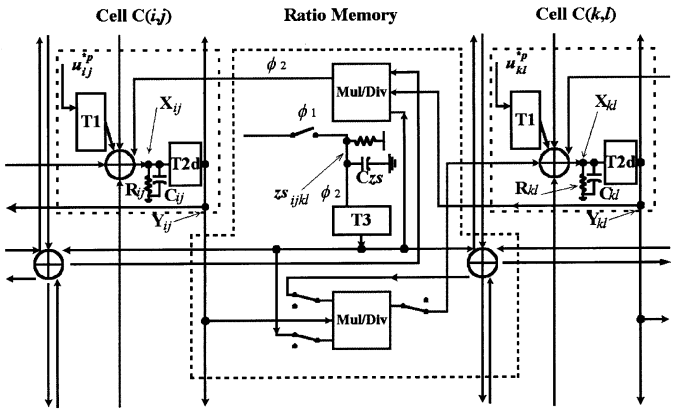


Fig. 4. The architecture of the RMCNN during the recognition period.

The current $Ix_{ij}(t)$ is converted into the voltage $Vx_{ij}(t)$ through the resistor R_{ij}. Thus, $Vx_{ij}(t)$ can be expressed as shown in (16), at the bottom of the page, where K_A^1 is the empirical gain. Ideally $K_A^1 = 1$. The ratioed weight $Izs_{ijkl}(t)/(\sum_{C(k,l)\in N_r^0(i,j)} abs[Izs_{ijkl}(t)])$ in (16) is generated by the two-quadrant divider in the block Mul/Div with its sign equal to the sign of $Izs_{ijkl}(t)$ latched in T2l whereas the multiplication of $IY_{ij}(t)$ and the ratioed weight is generated by the four-quadrant multiplier of Mul/Div by using the latched sign of $Izs_{ijkl}(t)$ and the sign of $Y_{kl}(t)$ in T2d. Comparing (16) with (6), it can be seen that the function of (6) is realized in the architecture of Fig. 4.

The generated $Vx_{ij}(t)$ is sent to T2d to generate the current $abs[IY_{ij}(t)]$ and $sign[IY_{ij}(t)]$ as shown in (17)–(18), at the bottom of the page, where the $Gm2d$ is the transconductance of T2d. It can be seen from (2), (17) and (18) that the block T2d realize $f[Vx_{ij}(t)]$ by separating its magnitude and sign. The sign $sign[IY_{ij}(t)]$ is detected in the block T2d and its voltage is $V_{\mathrm{SY}ij}$.

Generally, the learned template $\mathbf{A}$ matrix is asymmetrical. According to the simulation result, the learned template $\mathbf{A}$ leads to stable behavior. The above-mentioned stability will be formally proved later.

III. CMOS Circuit Realization

A. V-I Converters and Sign Detectors

The CMOS circuits of T2d and T2l are shown in Fig. 5 where Fig. 5(a) shows the circuits of V-I converter with the one-half absolute-value circuit. The V-I converter which is also used in the blocks T1 and T3 is a CMOS differential amplifier M1 $\sim$ M7 with the source resistance to increase the linear range. The two source resisters are realized by M5 and M6 devices operated in the linear region with the gate bias voltage Vbvic1. The output current Iovic is sent to the one-half absolute-value circuit formed by M8 $\sim$ M13 to generate the absolute-value current Ioabs with the unified flow direction. In Fig. 5(a), Vbvic1, Vbvic, Vbabsn, and Vbabsp are constant bias voltages.

The sign of Vzi_{ijkl} is detected and latched by the CMOS dynamic latch circuits of Fig. 5(b) in the block of T2l whereas the sign of the input voltage $V_{Xij}(V_{Yij})$ is detected by the four cascaded CMOS inverters in the block of T2d and its output voltage is denoted as $V_{\mathrm{SX}ij}(V_{\mathrm{SY}ij})$ in Fig. 5(c). When the input signal Vzi_{ijkl} or $V_{Xij}(V_{Yij})$ is larger than the inverter threshold voltage (1.5 V), the output of the latch circuit in Fig. 5(b) or the detect circuit in Fig. 5(c) is high (3 V). Otherwise. The circuit output becomes low (0 V) when the input signal Vzi_{ijkl} or $V_{Xij}(V_{Yij})$ is smaller than the threshold voltage (1.5 V). To avoid the effect of the inverter threshold-voltage variations, the input signal levels are kept well separated from the threshold voltage.

In the learning period, ϕ_1 is High (3 V) and ϕ_2 is low (0 V) in Fig. 5(b). The signs of the input voltages u_{ij}^p and u_{kl}^p are detected by the circuit of Fig. 5(c) in T2d and used to determine the sign of zi_{ijkl} in (4) whereas the sign of the voltage Vzi_{ijkl} is detected by the circuit of Fig. 5(b). In the recognition period, the sign of zi_{ijkl} or equivalently the sign of w_{ijkl} denoted as $V_{\mathrm{SW}ijkl}$ in Fig. 5(b) is further latched by setting ϕ_1 low (0 V) and ϕ_2 high (3 V). The latched sign is used in generating the first term in (16).

Fig. 6 is the HSPICE simulation result of the V-I converter with the one-half absolute-value circuit, which is designed by using 0.35 μm single-poly quadruple-metal (SPQM) N-well CMOS technology. It can be seen from Fig. 6 that

$$Vx_{ij}(t) = R_{ij}\left(\sum_{C(k,l)\in N_r^0(i,j)} K_A^1 \frac{I_{Y_{kl}}(t) Izs_{ijkl}(t)}{\sum_{C(k,l)\in N_r^0(i,j)} abs\left[Izs_{ijkl}(t)\right]} + Iu_{ij}^{*p} + Iz_{ij}\right) \tag{16}$$

$$abs\left[I_{Y_{ij}}(t)\right] = \begin{cases} Gm2d\, abs\left[Vx_{ij}(t)\right], & \text{if } -V_L \le Vx_{ij}(t) \le +V_U \\ Gm2d\, V_U, & \text{if } Vx_{ij}(t) > V_U \\ -Gm2d\, V_L, & \text{if } Vx_{ij}(t) < -V_L \end{cases} \tag{17}$$

$$sign\left[I_{Y_{ij}}(t)\right] = \begin{cases} 0\text{ V}, & \text{if } Vx_{ij}(t) < 1.5\text{ V} \\ 3\text{ V}, & \text{if } Vx_{ij}(t) > 1.5\text{ V} \end{cases} \tag{18}$$

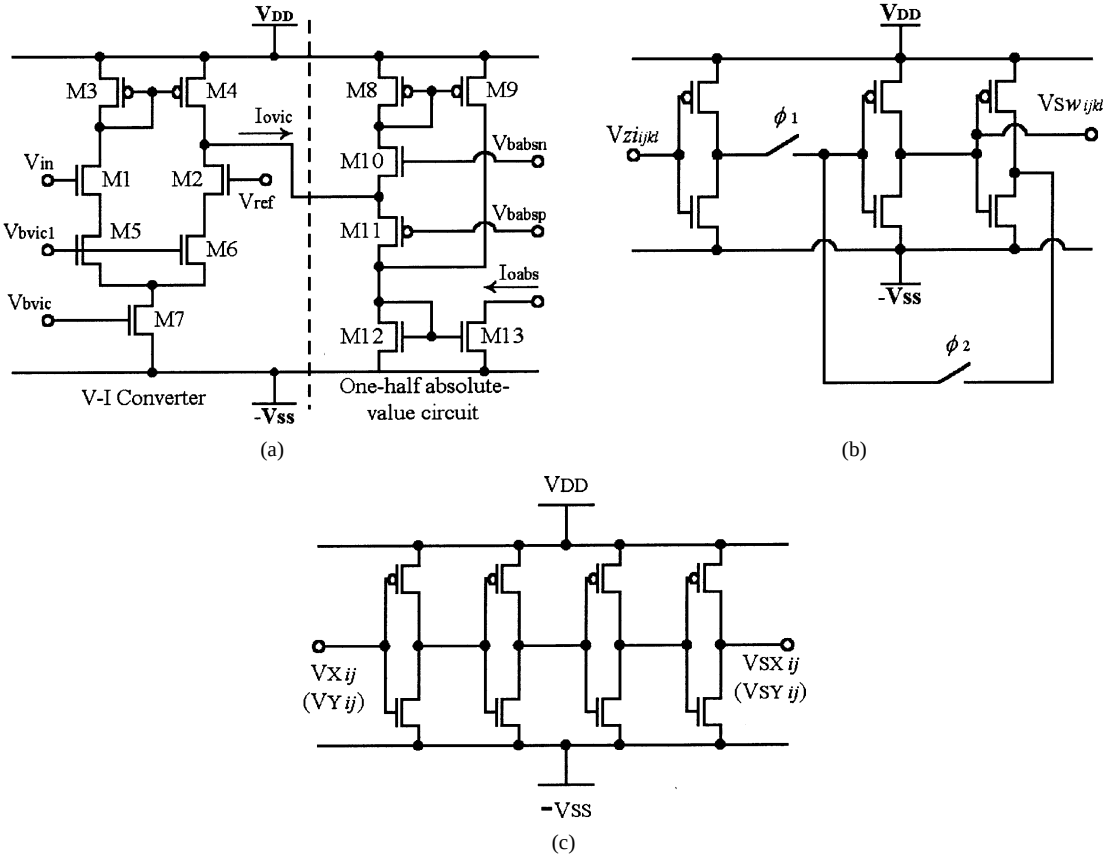


Fig. 5. (a) The circuit of V-I converter of the blocks T1 and T3 and the one-half absolute-value circuit used in the blocks T2l and T2d. (b) Latch circuit used in the block T2l. (c) The sign detector circuit used in the block T2d.

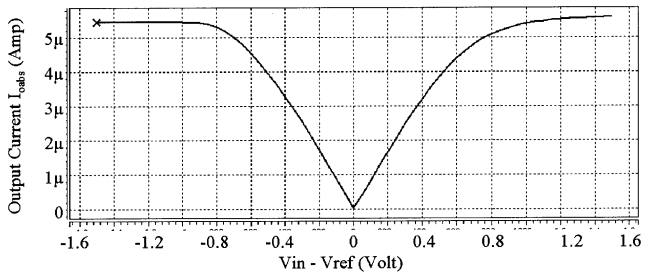


Fig. 6. The HSPICE simulation result of the circuit in Fig. 5(a).

the voltage Vin of the cell state X_{ij} is converted into positive current Ioabs. The maximum linearity error of Ioabs is 15% at Vin $-$ Vref $= \pm 0.8$ V. It is found that this error is acceptable in the RMCNN.

B. Combined Analog Multiplier and Divider

The combined four-quadrant analog multiplier and two-quadrant divider in Fig. 2 can be realized in the current mode by the CMOS circuit shown in Fig. 7 [20]. In Fig. 7, the currents I_1 and I_3 for multiplication are input through the pMOS current sources M14i/M14 and M15i/M15/M16, respectively, whereas the current I_2 as the divider is input through M24i/M24. The parasitic vertical PNP bipolar junction transistors (BJTs) Q1, Q2, Q3, and Q4 are adopted to perform the functions of multiplication and division by using the relation between emitter current I_E and base-emitter voltage $V_{\rm BE}$ as

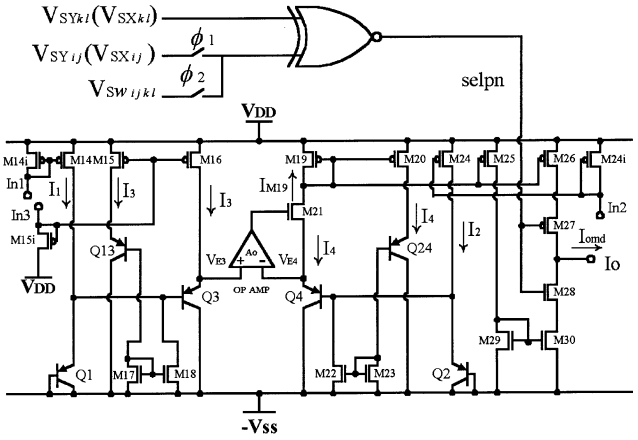


Fig. 7. CMOS circuit of the block Mul/Div.

$$I_E = I_S \exp\left(\frac{V_{\rm BE}}{V_T}\right) \text{ or } V_{\rm BE} = V_T \ln\left(\frac{I_E}{I_S}\right) \qquad (19)$$

where I_S is the emitter saturation current and V_T is the thermal voltage. The OP AMP Ao has a closed-loop feedback via the nMOS device M21. Thus, the emitter voltage V_{E3} and V_{E4} are virtually the same. With the buffered direct injection circuit [21], the output current I_4 can be readout through the pMOS current mirrors M19, M25 and M26, and the nMOS current mirror M29 and M30, to form the output current $I_{\rm omd}$.

Since $V_{E3} = V_{E4}$, we have

$$V_{\mathrm{BE1}} + V_{\mathrm{BE3}} = V_{\mathrm{BE2}} + V_{\mathrm{BE4}}. \tag{20}$$

Using the equation in (19), the relation among I_{E1}, I_{E2}, I_{E3}, and I_{E4} can be obtained from (20) as

$$I_{E4} = \frac{I_{E1} I_{E3}}{I_{E2}}. \tag{21}$$

Neglecting the base currents, the output current I_4 can be expressed in terms of I_1, I_2, and I_3 as

$$I_4 \cong \frac{I_1 I_3}{I_2}. \tag{22}$$

In (22), only the magnitudes of the input current signals are used to form the magnitude of the output current signal. The sign of I_4 should be determined to realize the complete function of four-quadrant multiplier and two-quadrant divider. In Fig. 7, the signal "selpn" is used to determine the sign of the output current I_{omd}. The signal "selpn" is obtained from the XNOR gate with the three different input signs. In the learning period, ϕ_1 is high and ϕ_2 is low. The output "selpn" is determined by the sign voltages VSX_{kl} and VSX_{ij} from the block T2d to realize the sign of $Iu_{ij}^{p} Iu_{kl}^{p}$ as in (14). In the recognition period, ϕ_1 is low and ϕ_2 is high. "selpn" is determined by the voltage VSY_{kl} and VSW_{ijkl} from the block T2d and T2l, respectively, to determine the sign of $IY_{kl} Izs_{ijkl}$ in (16). If the signal "selpn" is high (low), the sign is negative (positive) and the MOS device M28 (M27) is turned on to make Iomd $= -I_4(+I_4)$.

The BJT devices used in Fig. 7 are the parasitical vertical BJT in the 0.35-μm N-well CMOS process. The current gain β of the parasitical BJTs is about $6 \sim 17$. It is not large enough to neglect the effect of the base currents of the BJTs Q3 and Q4 to the emitter currents of the BJTs Q1 and Q2, respectively. Thus, extra circuits are needed to further bypass the base currents from entering the emitters of Q1 and Q2. In Fig. 7, the BJTs Q13 and Q24 have the same emitter currents I_3 and I_4 as Q3 and Q4, respectively. Thus, Q13 (Q24) has the same base current as Q3 (Q4). The current mirror circuits M17/M18 (M22/M23) are used to mirror the base current of Q13 (Q24) to Q3 (Q4). Thus the base current of Q3 (Q4) is bypassed from Q1 (Q2) and the relation $I_{E1} = I_1$ and $I_{E2} = I_2$ can be more accurately maintained to realize (22).

In the learning period, the Mul/Div block functions as a multiplier to implement the multiplication function $Iu_{ij} Iu_{kl}$. The HSPICE simulation results of the multiplier function of the Mul/Div circuit in Fig. 7 are shown in Fig. 8 where the device parameters of 0.35 μm SPQM N-well CMOS technology are used. It is found that in the actual operation range of I_1 from 0.5 μA to 6 μA and I_3 from 1.2 μA to 6 μA with I_2 kept at 20 μA, the multiplication error can be kept under 5.5%. In the recognition period, since the gain K_A^1 in (8) is chosen as 4 in IC design, most of the $Y_{ij}(t)$ from the neighboring cell is kept at the maximum absolute value as in (17). Thus, most of the corresponding input current I_1 of the Mul/Div block becomes a constant current and the Mul/Div block is functioned as a divider. The HSPICE simulation results of the divider function of the Mul/Div circuit in Fig. 7 are shown in Fig. 9. In the actual operation range of I_3 from 1.2 to 6 μA and I_2 from 0.3 to 6 μA with I_1 kept at 6 μA, the output current can be as high as 60 μA. Under the condition $I_3 < I_2$ which is the actual operation condition, the division error can be kept under 10%.

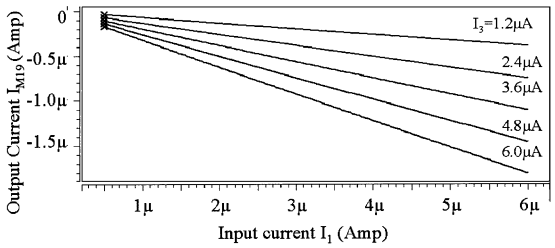


Fig. 8. HSPICE simulation results of the Mul/Div circuit with $I_2 = 20\ \mu$A.

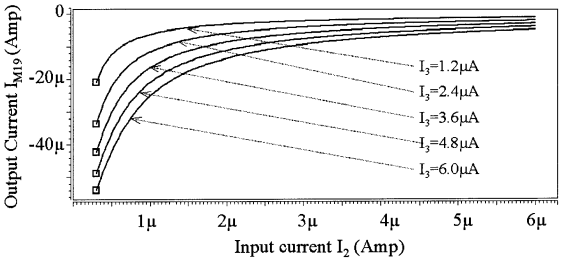


Fig. 9. HSPICE simulation results of the Mul/Div circuit with $I_1 = 6\ \mu$A.

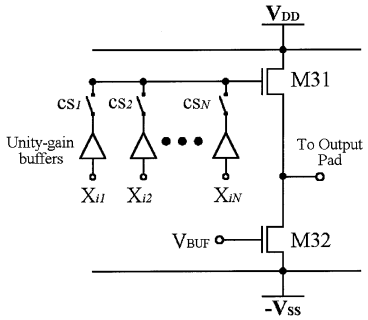


Fig. 10. The CMOS readout circuit for the cell state signal Xij.

The above errors of the Mul/Div circuit are also dependent on the variations of device parameters. However, it is found from simulation results that these errors have insignificant effects on the operation of RMCNN because of on-chip learning and RM operation.

C. The Complete Circuit

By using the above CMOS circuits as building blocks, the architecture of the RMCNN in Fig. 2 is implemented with the array size of 9×9. In the implemented 9×9 CMOS RMCNN, the capacitors Czi and Czs for absolute weight voltage storage are realized by the nMOS gate capacitors. Because of the current-mode output signals of the blocks in Fig. 2, the summing and distribution block is realized by directly connecting the output nodes of the related blocks to the input of the master stage in a CMOS current mirror to perform the summing function and the mirrored output current is distributed out through the multiple slave stages.

A layer of the boundary cells is designed to surround the 9×9 regular cell array. In the boundary cells, both state $X_{i^*j^*}(t)$ and input $u_{i^*j^*}(t)$ are zero. Thus, the output $Y_{i^*j^*}(t)$ of the boundary cell is also zero. Since the boundary cells have to send

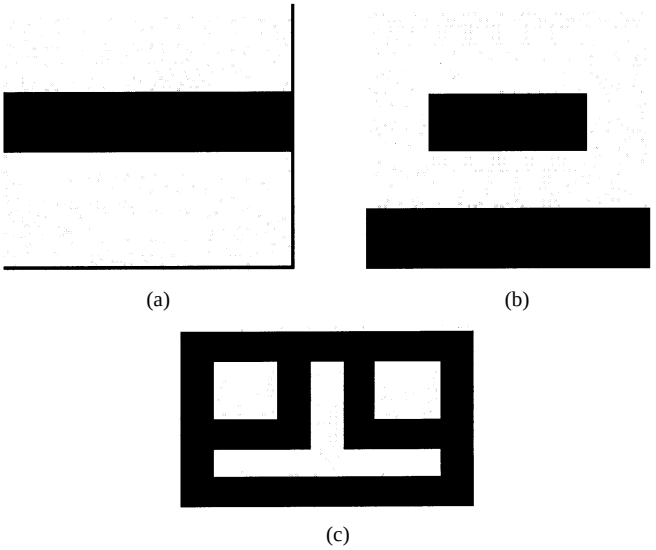


Fig. 11. The correct patterns of Chinese characters (a) "One." (b) "Two." (c) "Four."

a zero signal voltage to the neighboring regular cells or other boundary cells, it can be realized by setting the weights from boundary cell to other cells to be zero. Thus, the associated RM blocks can be removed.

To readout the neuron state signal Xij, suitable readout circuit shown in Fig. 10 is designed in the 9×9 CMOS RMCNN. In the readout circuit, the inputs of nMOS-input CMOS single-stage OP AMPs used as unity-gain buffers are connected to the node Xij in Fig. 2. The buffer output is connected to the input of the source-follower driver through the switch controlled by the column select control signal CSj. In the readout operation, CSj is raised to high column by column so that Xij is sent to the input of nMOS source follower M31 and M32 with M32 biased by $V_{\rm BUF}$ as the current source. Through the source follower, the neuron state signal can be readout column by column to the output pad and the large off-chip load.

IV. Simulation Results

A. 9×9 RMCNN

The MATLAB software is used to simulate the behavior of the RMCNN as an associative memory. In the MATLAB simulation, 9×9 neurons are used to form the RMCNN with $r = 1$. Thus, it can process patterns with 81 pixels. To consider the leakage current effect, a constant leakage current of 0.8 fA is applied to the capacitor Czs of 2 pF so the voltage Vzs_{ijkl} is decreased as in (19). The 2-pF capacitor Czs is implemented on the chip. The value of 2 pF is chosen as a compromise between weight storage time and capacitor chip area. The test patterns applied for learning and recognition are the patterns of Chinese characters "one," "two," and "four," as shown in Fig. 11. The learned $\mathbf{A}$ templates in (8) are space-variant templates. In Table I, one of the learned $\mathbf{A}$ template of the cell C(4,4) at $t = 0$ s denoted as $\mathbf{A}^1_{44}(0\ \text{s})$, is listed with the corresponding learned matrix $\mathbf{zi}^1_{44}(0\ \text{s})$ of the absolute weights. Due to the leakage current, both absolute and ratioed weights are changed with time as shown in Fig. 12(a) and 12(b), respectively. For the

TABLE I
The Learned Absolute and Ratioed Weights for the RMCNN

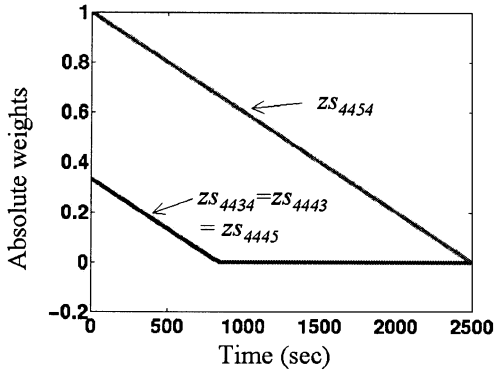
RMCNN	The learned **A** template	The corresponding absolute weights
9×9 $r=1$	$\mathbf{A}^1_{44}(0\ \text{s}) = \begin{bmatrix} 0 & -\frac{1}{6} & 0 \\ \frac{1}{6} & 0 & \frac{1}{6} \\ 0 & \frac{1}{2} & 0 \end{bmatrix}$	$\mathbf{zi}^1_{44}(0\ \text{s}) = \begin{bmatrix} 0 & -\frac{1}{3} & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & \frac{1}{1} & 0 \end{bmatrix}$
	$\mathbf{A}^1_{44}(850\ \text{s}) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$	$\mathbf{zi}^1_{44}(850\ \text{s}) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \frac{2}{3} & 0 \end{bmatrix}$
18×18 $r=1$	$\mathbf{A}^1_{44}(0\ \text{s}) = \begin{bmatrix} 0 & \frac{3}{16} & 0 \\ \frac{3}{16} & 0 & \frac{5}{16} \\ 0 & \frac{5}{16} & 0 \end{bmatrix}$	$\mathbf{zi}^1_{44}(0\ \text{s}) = \begin{bmatrix} 0 & \frac{3}{5} & 0 \\ \frac{3}{5} & 0 & \frac{1}{1} \\ 0 & \frac{1}{1} & 0 \end{bmatrix}$
	$\mathbf{A}^1_{44}(1500\ \text{s}) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 \end{bmatrix}$	$\mathbf{zi}^1_{44}(1500\ \text{s}) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \frac{1}{5} \\ 0 & \frac{1}{5} & 0 \end{bmatrix}$
	$\mathbf{A}^1_{104}(0\ \text{s}) = \begin{bmatrix} 0 & \frac{1}{12} & 0 \\ \frac{5}{12} & 0 & \frac{1}{4} \\ 0 & \frac{1}{4} & 0 \end{bmatrix}$	$\mathbf{zi}^1_{104}(0\ \text{s}) = \begin{bmatrix} 0 & \frac{1}{5} & 0 \\ \frac{1}{1} & 0 & \frac{3}{5} \\ 0 & \frac{3}{5} & 0 \end{bmatrix}$
	$\mathbf{A}^1_{104}(1500\ \text{s}) = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\mathbf{zi}^1_{104}(1500\ \text{s}) = \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{5} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

ratioed weights, the decreasing absolute weights lead to the effect of feature enhancement [14]–[17] which makes the smaller (larger) ratioed weights approach 0 (1) at $t = 850$ s as shown in Fig. 12(b) and Table I. After the elapsed period of 850 s, the absolute weights zi_{4434}, zi_{4443} and zi_{4445} have been decreased to 0. Note that the time for the ratioed weights to become 1 or 0 depends on the leakage current. The larger (smaller) leakage current makes the time shorter (longer).

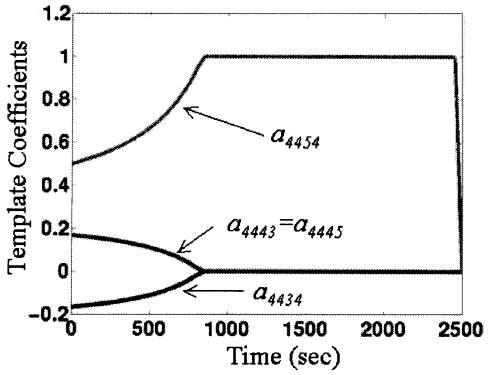
After the three patterns in Fig. 11 have been learned in the learning period and elapsed in the elapsed period of 850 s in this case, both correct patterns in Fig. 11 and 300 noisy patterns are applied to the 9×9 RMCNN with $r = 1$ for recognition and recovery. The noisy test patterns are the learned patterns with noise where the noise level is normally distributed with mean 0 and standard deviation 0.25. It is found that all of input correct or noisy patterns can be recognized and recovered correctly by the RMCNN after the three patterns have learned and elapsed for 850 s. After 2500 s, all the ratioed weights are decayed to 0 as shown in Fig. 12(b). Thus, the RMCNN cannot recognize input patterns. The total recognizable time in this case is 1650 s from 850 s to 2500 s. If only any two patterns in Fig. 11 are learned by the 9×9 RMCNN with $r = 1$, both correct and noisy input patterns can be recognized for 2500 s right after the two patterns are learned.

When the noise standard deviation of the noisy test patterns is increased to 0.3 (0.4), the average probability of accurate recognition is decreased to 97% (60%). Thus, the degradation of recognition rate is increased with the increase of noise standard deviation.

As compared to the 9×9 CNN without RM, but with the same learning rule and constant leakage on the coefficient of



(a)



(b)

Fig. 12. For the 9×9 RMCNN, a) the simulated absolute weights of the cell C(4,4) stored in the capacitor Czs versus time and b) the corresponding ratioed weights of the cell C(4,4) versus time.

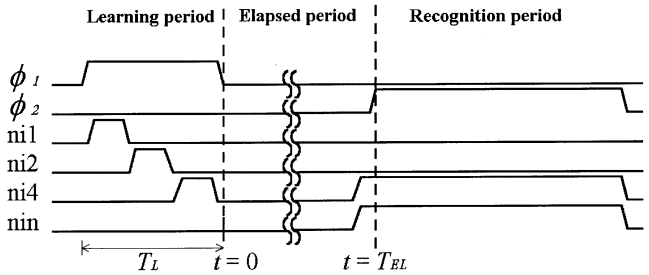


Fig. 13. The control-timing diagram in the HSPICE simulation of the 9×9 RMCNN with $r = 1$.

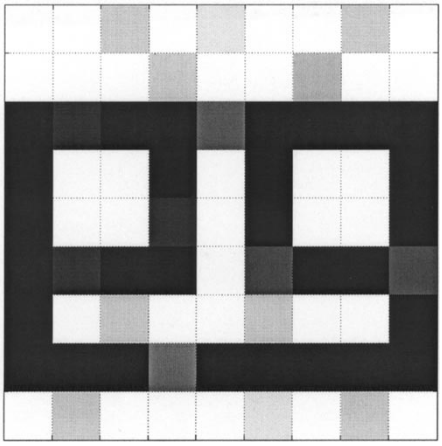


Fig. 14. The noisy pattern of the Chinese character "four" corresponding to the correct pattern in Fig. 12(c).

space-variant templates, only the two patterns "one" and "two" in Fig. 11(a) and 11(b), respectively, can be learned and only correct input patterns can be recognized from 0 to 1200 s. If the pattern "four" in Fig. 11(c) is used, only one pattern can be learned and recognized by the 9×9 CNN without RM.

From the above results, it is found that some ratioed weights are not well separated after the three patterns are learned by the 9×9 RMCNN. Thus, the pattern recognition and recovery is not successful. After 850 s, the feature-enhancement effect makes the ratioed weights well separated and the insignificant weights are decreased to zero. Thus, the pattern recognition and recovery can be performed successfully even with three input noisy patterns. If only two patterns are learned, the ratioed weights are well separated right after learning. Thus, no elapsed time is required for pattern recognition and recovery. If the CNN has only absolute weights without RM, there is no feature-enhancement effect under constant leakage and the number of recognizable patterns is reduced to one for complicated pattern and two for simple patterns. Besides, the noisy input patterns cannot be recovered without the feature-enhancement effect of RM.

The above results on the storage capacity of 9×9 RMCNN are obtained with the patterns in our test set. If different patterns are used, the results might be different.

The HSPICE simulation of the complete CMOS 9×9 RMCNN circuit designed in Section III is performed by using the device parameters of 0.35-μm SPQM N-well CMOS technology. The control-timing diagram in the HSPICE simulation is shown in Fig. 13. In the learning period, the signal ϕ_1 is set to high level (3 V) and ϕ_2 to low level (0 V). Thus, the circuit architecture of Fig. 3 can be formed. The control signals ni1, ni2, and ni4 in Fig. 13 are sequentially set to a high level so that the three patterns in Fig. 11 with the black (white) signal level of 2.5 V (0.5 V), are input to the circuit for learning. In the elapsed period, ϕ_1 and ϕ_2 are set to low (0 V). The learned absolute weight is stored on the capacitor Czs and decayed with time under the inevitable leakage current. In the recognition period, $\phi_2 = 1$ and $\phi_1 = 0$ and the circuit architecture of Fig. 4 is formed. Both control signals ni4 and nin in Fig. 13 are set to high level so that the noisy pattern shown in Fig. 14 can be applied to the 9×9 RMCNN for recognition. After enough elapsed time, the column select signals CSj are sequentially set to high level. Thus, the state Xij for $i = 1$ to 9 can be read out column by column from $j = 1$ to $j = 9$. The HSPICE simulated output waveforms for the noisy pattern are shown in Fig. 15 where the high (low) voltage of 1.2 V (0 V) represents black (white) level. It can be seen from Fig. 15 that the recognized result is the recovered correct pattern in Fig. 11(c). Thus, the above MATLAB simulation results have been verified in those HSPICE simulations on the real circuits.

B. 18 × 18 RMCNN

The behavior simulation of the 18×18 RMCNN with $r = 1$ is also performed. The patterns used for learning and recognition are the patterns of five Chinese characters shown in Fig. 16. One of the learned A templates $\mathbf{A}^1_{44}(0\text{ s})$ and $\mathbf{A}^1_{104}(0\text{ s})$ of the cells C(4,4) and C(10,4) at $t = 0$ s are listed in Table I with their corresponding absolute weights $\mathbf{zi}^1_{44}(0\text{ s})$ and $\mathbf{zi}^1_{104}(0\text{ s})$, respectively. Due to the leakage current, the smaller absolute weights

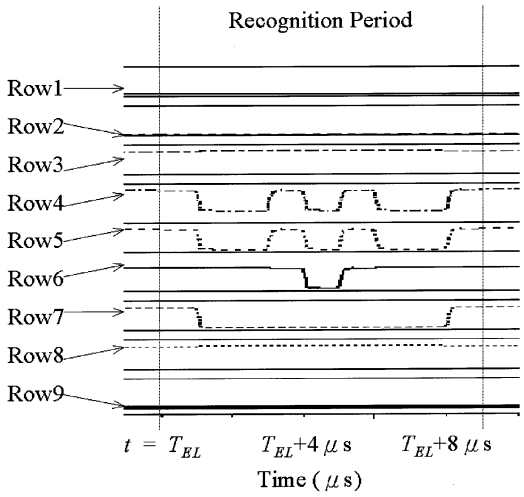


Fig. 15. HSPICE simulation output waveforms of neuron state Xij in the 9×9 RMCNN with the input noisy pattern "four" for recognition.

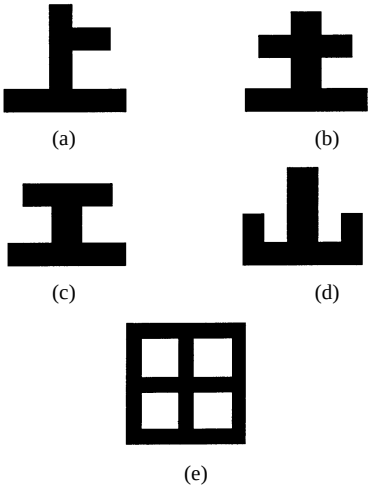


Fig. 16. Correct patterns of five Chinese characters: (a)"Up"; (b)"Soil"; c)"Work"; (d)"Mountain"; (e)"Farm" which are learned and stored in the 18×18 RMCNN.

are decayed to zero at $t = 1500$ s as shown in Fig. 17(a) and Table I. But the feature-enhancement effect [14]–[17] makes the coefficients of template converge to 1, 0.5 or 0 as shown in Fig. 17(b) and Table I. In $\mathbf{A}_{44}^{1}$, two largest terms in $\mathbf{zi}_{44}^{1}$ are left at $t = 1500$ s. Thus, the corresponding coefficients of $\mathbf{A}_{44}^{1}(0\text{ s})$ are converged to 0.5 instead of 1 according to (16).

After the five patterns in Fig. 16 have been learned and elapsed for the elapsed period of 1500 s, both correct patterns and 500 noisy patterns with noise levels normally distributed with mean 0 and standard deviation 0.25, are applied to the 18×18 RMCNN for recognition and recovery. The recognizable time for the five correct patterns is from 1500 s after these patterns have been learned to 2500 s. If only any four patterns in Fig. 16 are learned, the recognizable time for correct patterns is from 1250 s to 2500 s. For the recognition of noisy patterns, 98% accuracy on recognition and recovery can be achieved for five patterns whereas 99% accuracy for four patterns. In the case of five learned patterns, when the noise standard deviation is 0.3, the average probability of accurate recognition is down to 85%. The average probability of accurate recognition is only 50% when the noise standard deviation is increased to 0.35.

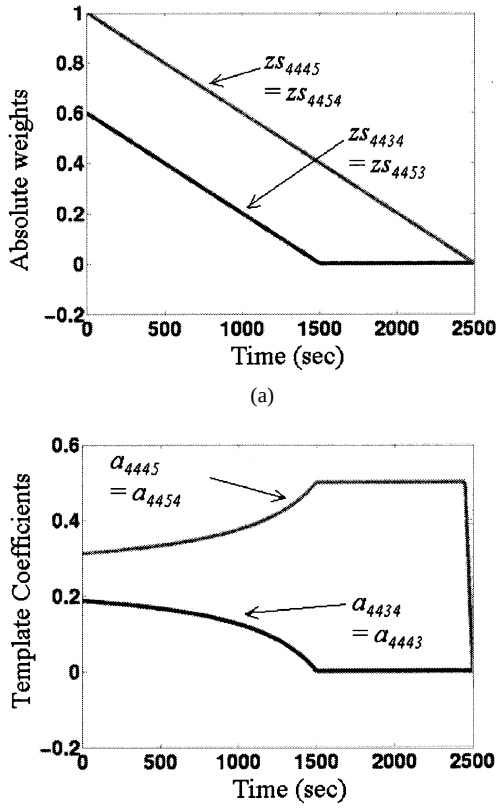


Fig. 17. For the 18×18 RMCNN, (a) the absolute weights $\mathbf{ZS}_{44}^{1}(t)$ and (b) the ratioed weights $\mathbf{A}_{44}^{1}(t)$ of cell C(4,4) versus time.

From the simulated recognition rates of 9×9 and 18×18 RMCNNs, it is realized that the 18×18 RMCNN can learn more patterns, but the tolerance to the noise standard deviation is lower as compared to that of 9×9 RMCNN.

In the 18×18 CNN without RM, but with the same learning rule and constant leakage on the coefficients of space-variant templates, only two patterns can be learned and only the correct patterns can be recognized at $t = 0$ s. But after elapsed time $T_{EL} = 2$ s, the correct patterns cannot be recognized. Thus, 18×18 CNN without RM has less capability in pattern learning, storing and recognizing. This is quite different from the case of RMCNNs where increasing the size from 9×9 to 18×18, the number of noisy patterns for learning and recognition is increased from three to five. The main reason for this difference is that as the number of stored patterns is increased with the array size of CNNs, the total number of space-variant templates is increased. The CNN associative memories without RM cannot keep all these templates well separated. Thus exact pattern recognition and recovery cannot be realized. But the feature-enhance effect of RMCNN retains the simple feature of the space-variant template coefficients and keeps them well separated. Thus, more patterns can be learned, stored and recognized.

Due to the unique feature-enhancement effect, the RMCNN can learn, store, recognize, and recover the same number of black and white (B/W) patterns with less weight connections among neurons as compared with the Hopfield neural network with RM and constant leakage on template coefficients [17]. For example, 18×18 RMCNN can process five B/W patterns as

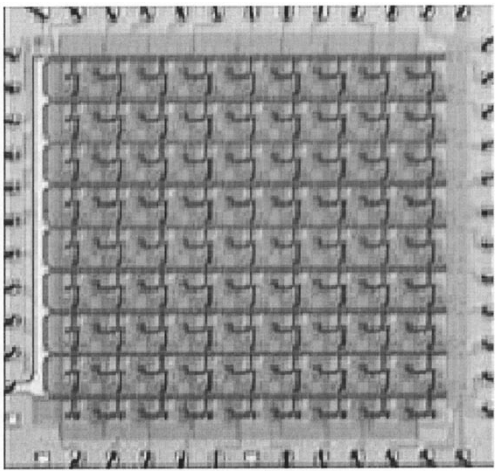


Fig. 18. Photograph of the fabricated CMOS 9 × 9 RMCNN chip.

9×9 Hopfield neural network with RM. But 18×18 RMCNN has 1296 weight connections while the 9×9 Hopfield network with RM has 6480 weight connections. The circuit complexity of RMCNN is about one fifth of the Hopfield network with RM.

As compared to other CNN associative memories without RM and without leakage on the stored template coefficients during the recognition operation [4], [5], [7], [8], the maximum numbers of stored and recognizable patterns are 25 (12) for 9×9 CNN with 49 (25) synaptic connections per cell [4], [7], [8] and two for 6×6 CNN with three synaptic connections per cell [5]. It is found from this work that both the leakage on the stored template coefficients and the noise of input patterns has a strong effect on the maximum number of stored and recognizable patterns.

V. Experimental Results

To verify the function of RMCNN, the experimental chip of 9×9 CMOS RMCNN circuit using the proposed architecture is designed and fabricated by using 0.35 μm single-poly quadruple-metal (SPQM) N-well CMOS technology. The photograph of the fabricated chip is shown in Fig. 18. It includes 9×9 regular cells, one surrounding layer of boundary cells, 144 RMs, and 9 rows of readout circuits. To compensate for the inevitable process variation effects on circuit parameter and guarantee the correct operation of the RMCNN chip, the gain K_A^1 is set to 4 as realized by the current ratio of the current mirrors M19/M25 and M19/M26 in Fig. 7.

Firstly, the three correct patterns in Fig. 11 are learned and the learned absolute weights are stored on the 2-pF capacitor Czs of the fabricated 9×9 RMCNN chip. As expected, the fabricated 9×9 RMCNN chip cannot recognize the correct test patterns just after it has learned the three patterns. After about 10 min, three noisy patterns, are input to the 9×9 RMCNN chip for recognition. The measured output waveforms of the cell state for the noisy pattern "four" are shown in Fig. 19 where the minimum readout time of a cell state signal is 1 μs. In Fig. 19, the first two signal waveforms are the column select signals CS1 and CS9, which select the responding first and ninth column to readout circuits. Other signal waveforms are the measured cell state outputs of each row. The high (low) signal level of

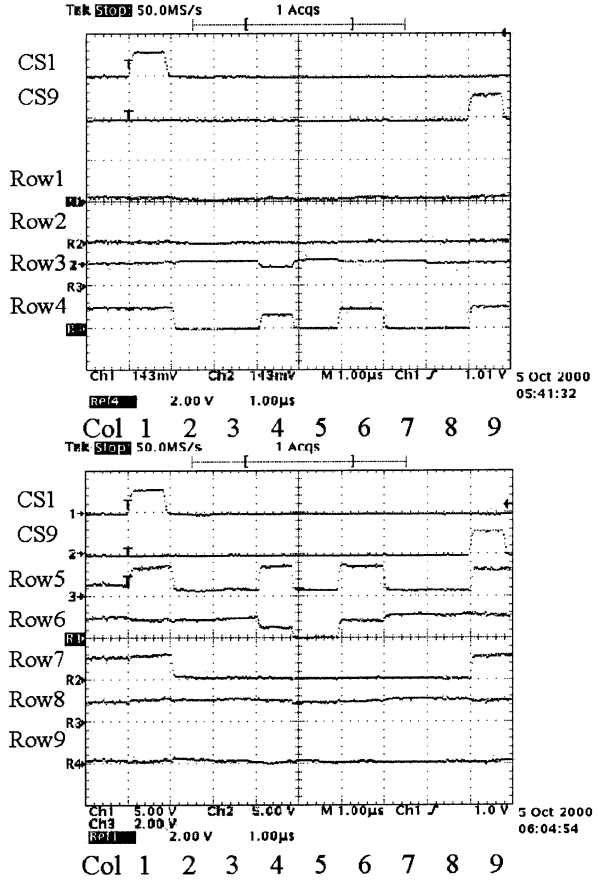


Fig. 19. Measured output waveforms for the input noisy pattern "four" in the fabricated CMOS 9×9 RMCNN chip.

TABLE II
Summary on the Characteristics of the Fabricated CMOS 9×9 RMCNN Chip

Technology	0.35µm Single Poly Quadrant Metal N-well CMOS
Resolution	9 × 9 cells
Single pixel area	350µm × 350µm
CNN array size (include pads)	3800µm × 3900µm
Power supply	3V
Total quiescent power dissipation	120mW
Dynamic power dissipation of the array	120mW ~ 140mW (Depending on image input)
Minimum readout time of a pixel	1 µs

1.2 V (0 V) represents black (white) level. As may be realized from the waveforms of Fig. 19, the noisy pattern "four" has been recovered to the correct pattern shown in Fig. 11(c). Similarly, the noisy patterns corresponding to Figs. 11(a) and 11(b) can be recognized and recovered correctly. If the input is the correct pattern, it still can be recognized.

The characteristics of the fabricated CMOS 9×9 RMCNN chip are summarized in Table II. The chip area of one single pixel including one regular neuron cell and two RMs is $350\ \mu\text{m} \times 350\ \mu\text{m}$. The chip area of a single RM block including the capacitors Czi and Czs is $220\ \mu\text{m} \times 130\ \mu\text{m}$. The chip area of 2-pF Czs capacitor is $50\ \mu\text{m} \times 25\ \mu\text{m}$. The quiescent power dissipation is 120 mW whereas the dynamic power dissipation is 120 mW $\sim$ 140 mW. The total readout time of the CMOS 9×9 RMCNN is 9 μs.

VI. Conclusion

In this paper, RMCNN is proposed and analyzed. In the RMCNN, the modified Hebbian learning rule is adopted to generate the absolute weights from a set of exemplar patterns and then transform them into ratioed weights to form the RM as space-variant $\mathbf{A}$ template coefficients. With RM and Hebbian learning rule, the RMCNN can be used as the associative memory for pattern learning, recognition and recovery. It is found from the simulation result that the CMOS 9×9 RMCNN can learn and recognize three patterns whereas the 18×18 RMCNN can learn and recognize five patterns. Due to the feature enhancement effect of the RM under constant leakage on the absolute template coefficients, the RMCNN can learn and recognize the same number of patterns with less weight connections as compared to the Hopfield neural network with the same RM and constant leakage on the template coefficients. Moreover, the proposed RMCNN can learn and recognize more patterns as compared to the CNN associative memories without RM, but with the same learning rule and the same constant leakage on the coefficients of space-variant templates. Based upon the designed architecture and CMOS circuits of the RMCNN, an experimental chip of CMOS 9×9 RMCNN has been designed and fabricated by using 0.35 μm CMOS technology. The experimental results has successfully verified the correct function of 9×9 RMCNN. Since the proposed RMCNN has the advantageous features in learning, storing, and recognizing image patterns, it is suitable for many applications of neural associative memory in real-time image processing.

Acknowledgment

The authors would like to thank the Chip Implementation Center (CIC), National Science Council (NSC), Taiwan, R.O.C., for their support in chip fabrication, and the reviewers for their valuable suggestions.

References

[1] L. O. Chua and L. Yang, "Cellular neural networks: Theory," *IEEE Trans. Circuits Syst.*, vol. 35, pp. 1257–1272, Oct. 1988.

[2] ——, "Cellular neural networks: Applications," *IEEE Trans. Circuits Syst.*, vol. 35, pp. 1273–1290, Oct. 1988.

[3] T. Roska, "Analog events and a dual computing structure using analog and digital circuits and operators," in *Discrete Event Systems: Models and Applications*, P. Varaiya and A. B. Kurzhanski, Eds. New York: Springer-Verlag, 1988, pp. 225–238.

[4] D. Liu and A. N. Michel, "Cellular neural networks for associative memories," *IEEE Trans. Circuits Syst. II*, vol. 40, pp. 119–121, Feb. 1993.

[5] M. Brucoli, L. Carnimeo, and G. Grassi, "An approach to the design of space-varying cellular neural networks for associative memories," in *Proc. 37th Midwest Symp. Circuits and Systems*, vol. 1, 1994, pp. 549–552.

[6] A. Lukianiuk, "Capacity of cellular neural networks as associative memories," in *Proc. 4th IEEE Int. Workshop Cellular Neural Networks and Their Applications*, 1996, pp. 37–40.

[7] H. Kawabata, M. Nanba, and Z. Zhang, "On the associative memories in cellular neural networks," in *Proc. IEEE Int. Conf. Systems, Man and Cybernetics*, vol. 1, 1997, pp. 929–933.

[8] P. Szolgay, I. Szatmari, and K. Laszlo, "A fast fiexed point learning method to implement associative memory on CNNs," *IEEE Trans. Circuits Syst. I*, vol. 44, pp. 362–366, Apr. 1997.

[9] R. Perfetti and G. Costantini, "Multiplierless digital learning algorithm for cellular neural networks," *IEEE Trans. Circuits Syst. I*, vol. 48, pp. 630–635, May 2001.

[10] A. Paasio, K. Halonen, and V. Porra, "CMOS implementation of associative memory using cellular neural network having adjustable template coefficients," in *Proc. IEEE Int. Symp. Circuits and Systems*, vol. 6, 1994, pp. 487–490.

[11] S. Grossberg, "Nonlinear difference-differential equations in prediction and learning theory," *Proc. Nat. Acad. Sci. USA*, vol. 58, pp. 1329–1334, 1967.

[12] ——, *The Adaptive Brain I: Cognition, Learning, Reinforcement and Rhythm*. Amsterdam, North-Holland: Elsevier, 1986.

[13] ——, *The Adaptive Brain II: Vision, Speech, Language and Motor Control*. Amsterdam, North-Holland: Elsevier, 1986.

[14] C.-Y. Wu and J.-F. Lan, "CMOS current-mode neural associative memory design with on-chip learning," *IEEE Trans. Neural Networks*, vol. 7, pp. 167–181, Jan. 1996.

[15] J.-F. Lan and C.-Y. Wu, "CMOS current-mode outstar neural networks with long-period analog ratio memory," in *Proc. Int. Symp. Circuits and Systems*, vol. 3, 1995, pp. 1676–1679.

[16] ——, "Analog CMOS current-mode implementation of the feedforward neural network with on-chip learning and storage," in *Proc. IEEE Int. Conf. Neural Networks*, 1995, pp. 645–650.

[17] C.-Y. Wu and C.-H. Cheng, "The design of CMOS modified Hopfield neural network for pattern recognition," in *Proc. Int. Symp. Multimedia Information*, 1997, pp. 585–590.

[18] J. A. Freeman and D. M. Skapura, *Neural Networks—Algorithms, Applications and Programming Techniques*. Reading, MA: Addison-Wesley, 1992.

[19] C.-Y Wu and C.-H Cheng, "The design of cellular neural network with ratio memory for pattern learning and recognition," in *Proc. IEEE 6th Int. Workshop Cellular Neural Network and their Applications*, 2000, pp. 301–307.

[20] ——, "A new analog multiplier-divider with compact structure for CMOS neural network applications," in *Proc. 1st Asia Pacific Conf. on ASICs*, 1999, pp. 315–317.

[21] C.-Y Wu, C.-C Hsieh, F.-W Jih, T.-P Sun, and S.-J Yang, "A new share-buffered direct-injection readout structure for infrared detector," in *Proc. SPIE Infrared Technology XIX*, vol. 2020, 1993, pp. 57–64.

Chung-Yu Wu (S'76–M'76–SM'96–F'98) was born in 1950. He received the M.S. and Ph.D. degrees from the Department of Electronics Engineering, National Chiao-Tung University, Hsinchu, Taiwan, R.O.C., in 1976, and 1980, respectively.

Since 1980, he has served as a Consultant to high-tech industry and research organizations and has built up strong research collaborations with high-tech industries. From 1980 to 1983, he was an Associate Professor at National Chiao-Tung University. During 1984 to 1986, he was a Visiting Associate Professor in the Department of Electrical Engineering, Portland State University, Portland, OR. Since 1987, he has been a Professor at National Chiao-Tung University. From 1991 to 1995, he was rotated to serve as the Director of the Division of Engineering and Applied Science on the National Science Council, Taiwan, R.O.C. Currently, he is the Centennial Honorary Chair Professor at National Chiao-Tung University. He has more than 250 published technical papers in international journals and conferences. He also has 19 patents including nine U.S. patents. His research interests focus on nanoelectronics, low-voltage low-power mixed-mode circuits and systems for giga-scale systems applications, cellular nonlinear networks and neural sensors, RF communication circuits and systems, biochips, and bioelectronics.

Dr. Wu is a member of Eta Kappa Nu and Phi Tau Phi Honorary Scholastic Societies. He was a recipient of the IEEE Third Millennium Medal, the Outstanding Academic Award by the Ministry of Education in 1999, the Outstanding Research Award by the National Science Council in 1989–1990, 1995–1996 and 1997–1998, the Outstanding Engineering Professor by the Chinese Engineer Association in 1996 and the Tung-Yuan Science and Technology Award in 1997.

Chiu-Hung Cheng (S'97) was born in Taipei, Taiwan, R.O.C., in 1971. He received the B.S. and M.S. degrees from the Department of Electronics Engineering, National Chiao-Tung University, Hsinchu, Taiwan, R.O.C., in 1995 and 1997, respectively, where he is currently working toward the Ph.D. degree in the Department of Electronics Engineering.

His research interests include analog integrated circuits design, and CNN integrated circuits design.