CHAPTER 3

SELF-FEEDBACK RATIO-MEMORY CELLULAR NONLINEAR NETWORK (SRMCNN) FOR AUTO-ASSOCIATIVE MEMORY

3.1 INTRODUCTION

As introduced by Chua and Yang [55]-[56], cellular nonlinear networks (CNN) with locally connected neighboring cells have the inherent advantage of being easily implemented in VLSI for various applications. Many image operations in CNN with suitable templates have been successfully explored [57]-[58] and realized in many applications. Moreover, the CNN can be used to classify and recognize image patterns through appropriate learning algorithms. Recently, this innovative application of CNN has attracted more research effort. Some important results have been reported in the literature [59]-[88], [92]-[102], [165]-[170].

The Hebbian learning algorithm can be used to perform unsupervised learning operation in a neural network system, in which the learned pattern signal is equal to the neuron's output. One Hebbian learning algorithm, called the discrete Hebbian learning algorithm, has been incorporated into CNN with some modification terms to generate associative memories for the learning and recognizing of image patterns [165]-[170]. Modified Hebbian learning is used to implement the 18×18 CNN for pattern learning over a fixed period [171]. The ratio memory (RM) in the Grossberg out-star structure is also used to form the template coefficients in the CNN for image recognition [169]-[172]. The resultant structure is called the ratio-memory CNN (RMCNN). To determine the four coefficients of the **A** template but not self-feedback coefficient for the

cell C(*i*,*j*) of the proposed RMCNN, the pixel values of the nearest four neighboring cells are multiplied by the pixel value of cell C(*i*,*j*), and the products are summed for all input patterns. Then, the accumulated product is transformed into a ratio to form the coefficient of the **A** template. The proposed RMCNN can learn and recognize three (five) patterns in the 9×9 (18×18) neuron array. The structure of RMCNN has been implemented in CMOS technology and its function has been successfully verified [171], [174]-[175].

The modified Hebbian learning algorithm used in the RMCNN can be modified to include a self-feedback term [170]-[172]. The modified algorithm is called the modified Hebbian learning algorithm with self-feedback. In this paper, the RMCNN with the modified Hebbian learning algorithm with self-feedback is proposed and analyzed. The new RMCNN is called self-feedback RMCNN (SRMCNN). In the learning process of the proposed SRMCNN, the features from input exemplar patterns are considered to update the weights. The operation of SRMCNN retains the feature enhancement effect of the RM. Detailed analysis and simulation results has shown that the SRMCNN can recognize up to 93 noisy patterns with a 100% success rate and 98 noisy patterns with a 97% success rate after learning the input exemplar patterns in uniform (normal) noise level is 0.8 (0.3). Thus, the capacity for learning and recognizing patterns is greatly improved.

The thesis is organized as follows. In Section 2, the operational principles, the modified Hebbian learning algorithm with self-feedback, and the embedded ratio memory in the SRMCNN are presented. Section 3 describes the architecture of the SRMCNN. In Section 4, the simulation results of SRMCNN are demonstrated and analyzed. Some phenomena are also discussed. Finally, conclusions are drawn.

3.2 OPERATIONAL PRINCIPLE AND LEARNING ALGORITHM

In a CNN, the behavior of a regular cell C(*i*,*j*) and its neighboring cells C(*k*,*l*) can be expressed by the differential state equation, in terms of their input, state, and output variables as [55]-[58]

$$\dot{X}_{ij}(t) = -X_{ij}(t) + \sum_{C(k,l)\in Nr(i,j)} a_{ijkl}(t) Y_{kl}(t) + \sum_{C(k,l)\in Nr(i,j)} b_{ijkl}(t) u_{kl} + z_{ij} \tag{3.1}$$

and the equation of the cell output $Y_{ij}(t)$ is [1]-[4]

$$Y_{ij}(t) = f\left[X_{ij}(t)\right] = \begin{cases} X_{ij}(t) & \text{if } -1 \le X_{ij}(t) \le +1 \\ 1 & \text{if } X_{ij}(t) > +1 \\ -1 & \text{if } X_{ij}(t) < -1 \end{cases} \tag{3.2}$$

where $X_{ij}(t)$ represents the cell state, $Y_{kl}(t)$ is the cell output from cell C(*k*, *l*) in the r-neighborhood system *Nr*(*i*, *j*) of the cell C(*i*, *j*), u_{kl} is the cell input from cell C(*k*, *l*) in *Nr*(*i*, *j*), z_{ij} is the threshold of cell C(*i*, *j*), *f*[ ] is bipolar activation function, and a_{ijkl} (b_{ijkl}) is the weight of template **A**(**B**) that correlates $Y_{kl}(t)$ (u_{kl}) to $X_{ij}(t)$. In an M×N CNN cell array, the r-neighborhood system *Nr(i, j)* of cell *C(i, j)* is defined as a set of cells that includes cell *C(i, j)* and its neighborhood cells. The term r is an integer that represents the number of the neighborhood layers. *Nr*(*i*, *j*) can be expressed by the following equation.

$$Nr(i,j) = \{\, C(k,l) \mid 1 \le k \le M,\ 1 \le l \le N;\ |k-i| \le r,\ |l-j| \le r \,\} \tag{3.3}$$

The general architecture of the SRMCNN is depicted in Fig. 3.1 [12]-[13] where the RM is used to realize the **A**-template weights of two neighboring cells and SRM is used to realize the self-feedback weight of the cell. In the SRMCNN, a coupled **A** template**,** an uncoupled **B** template, and r = 1 neighborhood is adopted. The space-variant **A** template has a self-feedback coefficient and four nearest neighboring coefficients. The **B**

template has only one coefficient that corresponds to the input of cell *C(i,j)*. Both **A** and **B** templates of *C(i,j)* can be expressed as

$$\boldsymbol{A}_{ij} = \begin{bmatrix} 0 & a_{ij(i-1)j} & 0 \\ a_{iji(j-1)} & a_{ijij} & a_{iji(j+1)} \\ 0 & a_{ij(i+1)j} & 0 \end{bmatrix} \quad \boldsymbol{B}_{ij} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \text{ for r = 1} \tag{3.4}$$

The outermost boundary cells are called the edge cells. They are commonly used to realize fixed (Dirichlet) boundary conditions. The output and input of those boundary cells are set to zero.

The modified Hebbian learning algorithm with self-feedback is applied in the SRMCNN to determine the updated volume of the weight vector at t = 0 as

$$W_{ijkl}(0) = \sum_{p=1}^{m} u_{ij}^{p} u_{kl}^{p} \qquad C(k,l) \in N_r(i,j) \tag{3.5}$$

$$z_{ij}(0) = 0 \tag{3.6}$$

where m is the number of learning patterns, u_{ij}^{p} is the pixel value of the *i*th row and the *j*th column in the *p*th pattern of m learned patterns with the value +1 or –1, u_{kl}^{p} is the pixel of the cell *C(k,l)* of *Nr* neighboring cells including cell *C(i,j)*, $W_{ijkl}(0)$ in (3.5) is the weight associated with cell *C(i,j)* and its neighboring cells *C(k,l)*, and $z_{ij}(0)$ in (3.6) is the threshold of cell *C(i, j)* which is set to zero. Note that the self-feedback terms $W_{ijij}(0)$ is defined in (3.5).

In the learning period, the weights $W_{ijkl}(0)$ are generated in parallel from $u_{ij}^{p} u_{kl}^{p}$ and accumulated for all m learned exemplar patterns. They are updated simultaneously when an exemplar pattern is input at a given time. Then its magnitude $\left|W_{ijkl}(0)\right|$ is stored on the capacitor Czs to generate the ratio weights. According to (3.5), if the

product of $u_{ij}^{p}u_{kl}^{p}$ is positive, the weight $W_{ijkl}(0)$ of **A** template is increased. Otherwise, $W_{ijkl}(0)$ is decreased. Since the self-feedback term $u_{ij}^{p}u_{ij}^{p}$ is always positive, the self-feedback weight $W_{ijij}(0)$ is one of the largest weights among the five weights in (3.5).

In the elapsed period, starting from t=0, the leakage current $I_{leakage}$ associated with capacitor C*zs* gradually decreases the stored voltage $|W_{ijkl}(0)|$ as time elapses. Since the leakage current is almost constant, the change of $|W_{ijkl}(t)|$ on capacitor C*zs* can be written as

$$|W_{ijkl}(t)| = |W_{ijkl}(0)| - \frac{I_{leakage}}{Czs}t \tag{3.7}$$

The RM is used to generate the ratio weight a_{ijkl} of the **A** template in the recognition period. The noisy patterns are input to the SRMCNN with the ratio weights to perform the recognition operation. The derivative $\dot{X}_{ij}(t)$ of the cell state is expressed as

$$\dot{X}_{ij}(t) = -X_{ij}(t) + \sum_{C(k,l)\in Nr(i,j)} a_{ijkl}(t)Y_{kl}(t) + u_{kl} + z_{ij} \tag{3.8}$$

and the ratio weights $a_{ijkl}(t)= a_{ijkl}$ are generated according to the equation [171]-[172], [174]-[175]

$$a_{ijkl} = a_{ijkl}(t) = \frac{W_{ijkl}(t)}{\sum_{C(k,l)\in N_r(i,j)} |W_{ijkl}(t)|} \tag{3.9}$$

The ratio weight $a_{ijkl}(t)$ in (3.9) has the effect of feature enhancement. When the weight magnitude exceeds the mean value of all a_{ijkl} terms of the cell *C(i,j)*, it is increased gradually with time. Otherwise, the weight decreases gradually. Since the self-feedback weight $W_{ijij}(0)$ or $W_{ijij}(t)$ is one of the largest weights in the cell *C(i,j)*, the corresponding self-feedback ratio weight a_{ijij} is the largest in the **A** template. With a_{ijij},,

the features of patterns can be enhanced to reject the noise. Thus, the capability of recognition for noisy patterns is significantly improved by the SRMCNN.

In the recognition period, the outputs are adjusted according to (3.8) for noisy input patterns with either uniform or Gaussian (normal) noise distribution. The output pattern noise is gradually eliminated through a feedback-type interaction. The outputs of all neurons are adjusted to eliminate noise during the recognition period until no further change is detected. Finally, the SRMCNN reaches its stable state.

The energy function of a CNN in quadratic form [64] can be expressed as

$$E = -\frac{1}{2}\sum_{ij}\sum_{kl}(a_{ijkl}Y_{kl})Y_{ij} \tag{3.10}$$

When all the cells become saturated in the recognition period, we have dE/dt=0 and the SRMCNN results in a stable output with the energy function converged to its local minimum. The minima of E correspond to stable states. The final recognized pattern represents one local minimum of the energy function.

3.3 SRMCNN ARCHITECTURE

The detailed block diagram of two neighboring CNN cells and their RM in the SRMCNN is shown in Fig. 3.2(a), and the detailed block diagram of the S block is shown Fig. 3.2(b) when the SRMCNN is operated during the learning period. In Fig. 3.2(a), the block T1 is a V-I converter used to convert the voltage of input patterns into current. The block T2d is a V-I converter with a one-half absolute-value circuit and a sign-detection circuit to generate the absolute value of output current and detect the sign of the cell state $X_{ij}(t)$, respectively. The CNN cell *C(i,j)* is formed by T1, T2d, R_{ij}, and C_{ij} as indicated in Fig. 3.2(a) [171], [174]-[175].

The block M/D [171] in Fig. 3.2(a) is a combined four-quadrant multiplier and a

two-quadrant divider circuit. The block is used to realize the modified Hebbian learning algorithm with self-feedback during the learning period. It is also used to multiply perform the multiplication a_{ijkl} (t) and Y_{kl} (t) in the recognition period. The resultant absolute weight zi_{ijkl} during the learning period is stored in the capacitor *Czi* in the S block of Fig. 3.2(b). In Fig. 3.2(b), 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 *Rzs* in parallel with *Czs* is used to generated the absolute voltage from the output current of block T2L and to store the voltage on *Czs*. Block T3 is also a V-I converter to convert the voltage of *Czs* into current. The output current of T3 is sent to the sum block and summed with the currents from neighboring cells. The summed current is sent to the M/D block to generate ratio-memory. Both M/D and S blocks form the RM among CNN cells as indicated in Fig. 3.2(a).

In Fig. 3.2(a), the *m* exemplar patterns are input in order read into the cell *C(i, j)* and the input voltage Vu_{ij}^{p} of the *p*-th input pattern is sent to T1 to be converted into current $\mathrm{I}u_{ij}^{p}$ and then to T2d to extract its absolute current value $\left|\mathrm{I}u_{ij}^{p}\right|$ and sign. Then the converted absolute currents $\left|\mathrm{I}u_{ij}^{p}\right|$ and $\left|\mathrm{I}u_{kl}^{p}\right|$ from two neighboring cells are sent to the four-quadrant multiplier in the M/D block to generate the product. The generated product in the current mode charges the capacitor *Czi* for the period T_P to generate the voltage on *Czi*. This operation is repeated for *m* patterns to sum the voltages of *Czi*. Finally, the weight voltage $Vzi_{ijkl}(0)$ stored on *Czi* at $t = 0$ when the learning period ends, can be written as

$$\mathrm{V}zi_{ijkl}(0) = \frac{1}{\mathrm{C}zi}\sum_{p=1}^{m}\left(\int_{T_P}\frac{\mathrm{I}u_{ij}^{p}\,\mathrm{I}u_{kl}^{p}}{\mathrm{Ib}}dt\right) \qquad C(k,l) \in N_r(i,j) \tag{3.11}$$

where $\mathrm{I}u_{ij}^{p}$ is the current of the *p*-th input patterns sent to the cell *C(i, j)*, $\mathrm{I}u_{kl}^{p}$ is the

current of the *p*-th pattern sent to the cell *C(k, l)* of $N_r(i, j)$ neighboring cells, Ib is a constant bias current, $\mathrm{V}zi_{ijkl}(0)$ is the weight W_{ijkl} voltage stored on *Czi* at $t = 0$ sec, and T_P is the learning time of each input pattern. Through T2l, the absolute value $abs[\mathrm{V}zi_{ijkl}(0)]$ of the weight $\mathrm{V}zi_{ijkl}(0)$ is stored on the capacitor *Czs*, whereas the sign of $\mathrm{V}zi_{ijkl}(0)$ is stored in the latch circuit of T2L.

In Fig. 3.2(a), the voltage $\mathrm{V}zi_{ijkl}(0)$ weight $W_{ijkl}(0)$ is directly generated by the current product of $\mathrm{I}u_{ij}^{p}\mathrm{I}u_{kl}^{p}$ changing on the capacitor *Czi* for the period *Tp*. $\mathrm{V}zi_{ijkl}(0)$ is stored on the capacitor *Czi*. Then, the absolute value of $\mathrm{V}zi_{ijkl}(0)$ is transferred and stored on the capacitor *Czs*.

In the elapsed period, the configuration of SRMCNN is shown in Fig. 3.3(a), where *Czs* is disconnected from the block T2L as shown in Fig. 3(b). The leakage current $I_{leakage}$ associated with *Czs* gradually decreases $abs[\mathrm{V}zi_{ijkl}(0)]$ of *Czs*.

In the recognition period, the configuration of the SRMCNN is shown in Fig. 3.4 and that of S block are the same as that in the elapsed period. The voltage $\mathrm{V}u_{ij}^{t}$ of the test pattern to be recognized is input to T1 and converted into the current $\mathrm{I}u_{ij}^{t}$. The absolute weight voltage $abs[Vzs_{ijkl}(t)]$ stored on C*zs* is converted into the current $abs[Izs_{ijkl}(t)]$ through T3 and summed with the currents from other neighboring cells. The summed current, the weight current $abs[Izs_{ijkl}(t)]$, and the cell output current $I_{Ykl}(t)$ are sent to the M/D block to yield the current that corresponds to the term $a_{ijkl}(t)\ Y_{kl}(t)$ in (3.1), which is then summed with the currents from other neighboring cells, the input current $\mathrm{I}u_{ij}^{t}$, and the threshold current $\mathrm{I}z_{ij}$ to generate the cell state current $\mathrm{I}x_{ij}(t)$. The current $Ix_{ij}(t)$ is converted into the voltage $Vx_{ij}(t)$ through resistor R_{ij}. Thus, $Vx_{ij}(t)$ can be

expressed as

$$\mathrm{Vx}_{ij}(t) = \mathrm{R}_{ij}\left(\sum_{\mathrm{C}(k,l)\in N_r(i,j)} \mathrm{K_A} \frac{\mathrm{I}_{\mathrm{Y}_{kl}}(t)\mathrm{Izs}_{ijkl}(t)}{\sum_{\mathrm{C}(k,l)\in N_r(i,j)} abs[\mathrm{Izs}_{ijkl}(t)]} + \mathrm{Iu}_{ij}^{t} + \mathrm{Iz}_{ij} \right) \tag{3.12}$$

where $\mathrm{K_A}$ is the empirical gain. Ideally $\mathrm{K_A}$ = 1. The ratioed weight $\mathrm{Izs}_{ijkl}(t) \Big/ \sum_{\mathrm{C}(k,l)\in N_r(i,j)} abs[\mathrm{Izs}_{ijkl}(t)]$ in (3.12) is generated by the two-quadrant divider in the M/D block with its sign equal to the sign of $Izs_{ijkl}(t)$ latched in T2L, whereas the I_{Yij}(t) is multiplied by the ratioed weight by the four-quadrant multiplier of M/D using the latched sign of $Izs_{ijkl}(t)$ and the sign of $Y_{kl}(t)$ in T2d. The current of input patterns is summed with the five weighted outputs from neighboring cells during the recognition period and converted into a voltage through the resistor R_{ij} and the parasitic capacitor C_{ij} to form the cell state $X_{ij}(t)$.

The generated $Vx_{ij}(t)$ is sent to T2d to generate the current $abs[I_{Yij}(t)]$ and $sign[I_{Yij}(t)]$. The block T2d realizes $f[\mathrm{Vx}_{ij}(t)]$ by separating its magnitude and sign. The sign $sign[I_{Yij}(t)]$ is detected in the block T2d and the voltage is V_{SYij}.

In the proposed SRMCNN, each cell requires an extra self-feedback ratio-memory with M/D and S block to realize the self-feedback weight W_{ijij} or a_{ijij}. As in the original RMCNN, eight sets of M/D and S block are required to generate and store the ratio weight a_{ijkj} from cell $C(k,l)$ and the ratio weight a_{klij} from cell $C(i,j)$, respectively. Thus five sets of M/D and S block per cell are required in the architecture of SRMCNN. As compared with RMCNN, the increased hardware is small but the performance in pattern recognition is greatly improved.

The SRMCNN also can be integrated into the conventional CNNUM, and is called SRMCNNUM. The chip area of the cell, the core cell array, and the SRMCNNUM are

estimated by using different CMOS process technologies. Table I lists these areas.

3.4 SIMULATION RESULTS

Matlab software is used to simulate the operations of the proposed SRMCNN with 18×18 neurons, the direct neighborhood (r=1), and the modified Hebbian learning algorithm with self-feedback. The 18x18 SRMCNN can process patterns with 324 pixels. In each pattern, a black pixel is expressed by +1 whereas a white pixel by -1. To elucidate the effect of leakage current in the simulation, a constant leakage current of 0.8fA is applied to the capacitor *Czs* of 2pF, the stored voltage Vzs_{ijkl} will gradually decreased. The capacitance of 2pF is chosen as a compromise between the weight storage time and the capacitor chip area.

The totals of 98 exemplar patterns to be processed in the SRMCNN are classified into four groups. Group 1 includes 35 (No.1~No.35) Chinese characters with vertical-horizontal lines of two-pixel width. Group 2 includes 52 (No.36~No.87) English characters (capital and small letters) with the slant lines. Group 3 includes six (No.88~No.93) patterns with vertical-horizontal grid lines. Group 4 includes five (No.94~No.98) patterns with slant lines only. It has the most complicated patterns. Variations of the selected weights in the **A** template during various operation periods in some selected cells are examined to verify the RM phenomenon in the SRMCNN. In Table 3.2 (a), the generated ratio weights $W_{ijkl}(0)$ and $a_{ijkl}(t)$ of the cells (3,1), (6,2), (9,3), (11,17) and (15,5) after the learning period and the elapsed period with the learned 36 (No. 1~No 36) exemplar patterns are listed. In Table II (b), the weights of the cells (5,10), (8,6), (11,2), (13,16), and (16,12) with learned 98 exemplar patterns. As shown in both Tables 3.2(a) and 3.2(b), the learned **A** templates for different input exemplar

patterns are different. The **A** template for larger number learned patterns has fewer elements than that for small number of learned patterns. Moreover, the constant leakage current can enhance the larger ratioed weights while suppressing the smaller is to zero. For N larger ratio weights, they are enhanced to 1/N during the elapsed period. The effect is called the feature enhancement effect [171]-[172], [174]-[175].

Due to the feature enhancement effect, the variations of the ratio weights of two **A** templates of $A_{5,10}$ and $A_{11,2}$ versus the elapsed time factor is shown in Figs. 3.6(a) and 3.6(b), respectively, during the elapsed period.. The elapsed time factor is normalized by the elapsed time of 50 seconds. As seen from Figs. 3.5(a) and 3.5(b), the value of the weight is increased to 1 or 1/N whereas the others are decayed to zero. For example, the **A** template weights of the cell *C(5,10)*, $A_{5,10}$ =[0.15 0.25 0.25 0.18 0.18] at elapsed time factor=1 is changed to $A_{5,10}$ =[0 0.5 0.5 0 0] at elapsed time factor=14 as shown in Fig. 3.5(a). Similarly, The weights $A_{11,2}$ =[0.22 0.18 0.24 0.13 0.22] at elapsed time factor=1 is changed to $A_{11,2}$ =[0 0 1 0 0] at elapsed time factor=14 as shown in Fig. 3.5(b). It is found that the success rate of pattern recognition is related to the elapsed time factor. The minimum required elapsed time factors that yield the maximum success recognition rate of with different numbers of the learned patterns are given in Table 3.3. The minimum required elapsed time factors are from 8 to 18 for different patterns, which corresponds a range from 400 to 900 seconds. Note that the maximum elapsed time is generally proportional to the number of the learned patterns and their complexity.

One hundred noisy test patterns and two types of noise are used in simulations to determine the success rate of pattern recognition. One type is the uniform distribution random noise at the levels between 0 and 0.05n, where n is a noise level factor. The other type is the normal distribution random noise with a noise standard variance of 0.05m, where m is a noise variance factor. As verified by the simulation results, the

18×18 SRMCNN can learn 93 patterns and successfully recognize the corresponding 93 noisy patterns of Groups 1, 2, and 3 with uniform (normal) distribution noise at a level of n=16 (variance of m=6). The success rate is 100%. The simulation shows that the learned **A** template already catch the features of all three groups of patterns. Thus, actually more than 93 patterns in the same groups can be recognized correctly. Fig. 3.6(a) shows some noisy test patterns with uniform noise level of 0.8, whereas Fig. 3.6(b) shows the correctly recognized patterns. The success rate versus the noise level factor n and the noise variance factor m for 93 (No.1~No.93) noisy test patterns with uniform and normal distribution noise are shown in Figs. 3.7(a) and 3.7(b), respectively. The Figures show that the success rate decreases as the noise level increases beyond 0.8 and the noise variance exceeds 0.3.

The success rate versus the noise level factor n and the noise variance factor m for 98 noisy patterns of the four groups with uniform and normal distribution noise shown in the Figs. 3.8(a) and 3.8(b), respectively. The success rate is 97% for uniform noise levels of 0.8 and a normal noise variance of 0.25. The rate is rapidly decreased at noise levels over 1.0 or noise variances over 0.25. Analysis indicates that the two patterns include only slanted lines within Group 4, as shown in Fig. 3.9, cannot be completely recognized. Accordingly, the success rate is degraded to 97%. If only five patterns in Group 4 are learned and recognized under uniform noise, the success rate can reach 100% when the uniform noise level is 0.8, as shown in Fig. 3.8(c).

All the simulation results concerning the success rate for various numbers of patterns and different types of noise are summarized in Table 3.4. Those simulation results indicate that the SRMCNN has a better learning and recognition capability if the learned patterns are simpler and the noise is lower. For complex patterns like those of Group 4, the numbers of pattern learning and recognition should be decreased to yield a

100% success rate.

If one pattern with vertical–horizontal lines in Group 3 is added to Group 4, the success rate is decreased to 90% due to the learning of a different type of pattern from those five patterns with slant lines only.

The patterns not already learned are included in the noisy patterns to be recognized to verify the effect of learning on that of recognition in the SRMCNN. It is found that almost no unlearned patterns can be recognized correctly. Thus, pattern learning is required to recognize a correct pattern.

To investigate the recognition convergence of SRMCNN, a noisy pattern with a uniform noise level of 0.8, as shown in Fig. 3.10(a) is recognized as the stable pattern in Fig. 3.10(b). The value corresponding to the energy function of each iteration in the recognition operation during the recognition period is shown in Fig. 3.10(c). It can be seen that the value of energy function is decreased to the minimum value and the correct pattern is generated after two iterations.

It has been shown that the number of connection weights in the SRMCNN is much less then that in the Hopfield neural network and the SRMCNN can achieve higher capabilities with 93 patterns. The 18×18 SRMCNN has 1620 weight connections while the 18×18 Hopfield network has 104652. The circuit complexity of SRMCNN is approximately 1/65 of that of the Hopfield network.

For comparisons, conventional CNN associative memories have been proposed with the learned weights of the **A** template processed without RM and leakage during the recognition operation [59]-[61]. It is shown that the maximum numbers of stored and recognized patterns is 25 (12) for a 9×9 CNN with 49 (25) weight connections. The 18×18 RMCNN without a self-feedback weight in the A template can learn and recognize five patterns [171]. The proposed SRMCNN with RM and self-feedback

weight can enhance the feature of the exemplar patterns and significantly improve the capability of recognition. As shown in the simulation results, the 18×18 SRMCNN can learn and recognize 93 noisy patterns with five weights connection. This verifies the improved recognition capability of the SRMCNN.

Using the same learning algorithm but without RM and leakage current, 15 exemplar patterns can be learned in the 18×18 CNN and only 6 (11) patterns could be correctly recognized from input noisy patterns with a uniform noise level of 0.5 (0.3). The success rate of recognition is 40% (73%). This verifies the importance of the effect of RM on the learning and recognition capability of the SRMCNN.

3.5 SUMMARY

In this chapter, the ratio memory cellular nonlinear network with self-feedback (SRMCNN) is proposed and analyzed. In the SRMCNN, the modified Hebbian learning algorithm with self-feedback is applied to the generation the absolute weights from the sets of input exemplar patterns and then transform them into ratio weights through the ratio memory to form the coefficients of space-variant **A** template. With RM and the modified Hebbian learning algorithm with self-feedback, the SRMCNN can be used as the associative memory for learning, recognizing, and recovering patterns. The simulation results have shown that the 18×18 SRMCNN with five weights connection can learn and recognize 93 noisy patterns with a 100% success rate at a uniform distribution level of 0.8 and a normal distribution variance of 0.3. This has successfully verified the correct function and superior performance of SRMCNN in the patterns recognition.

The proposed SRMCNN with the feature enhancement effect of the RM under constant leakage on the template coefficients can learn and recognize patterns with fewer

weight connections than that of the Hopfield neural network. Moreover, the proposed SRMCNN with the self-feedback ratio weight can learn and recognize more patterns than the CNN associative memories with RM and without RM, given the same learning algorithm and the same constant leakage in the coefficients of space-variant templates. Simulation results have successfully verified the correct function of 18×18 SRMCNN. Since the proposed SRMCNN has the advantages in learning, storing, and recognizing image patterns, it is suitable for appropriate applications of nanoelectronic associative memory systems for real-time image processing.

Table 3.1. Estimated Chip Areas of Cell, Core Cell Array, and SRMCNNUM for Different Types of CMOS Technology.

Circuit \ CMOS Technologies	0.25um	90nm	65nm
cell	$7x10^4um^2$	$9x10^3um^2$	$45x10^2um^2$
core cell array (128x128)	$7x10^8um^2$	$9x10^7um^2$	$45x10^6um^2$
SRMCNNUM (128x128)	$11x10^8um^2$	$1.4x10^8um^2$	$75x10^6um^2$

Table 3.2 Generated Ratio Weights of Some Neuron Cells in The 18×18 SRMCNN for (a) 36 Learned Patterns, and (b) 98 Learned Patterns After Different Operation Periods.

	$W_{ijkl}(0)$			$a_{ijkl}(t)$		
C(3,1)	0	0.26 0.26 0.26	0.23	0	0.33 0.33 0.33	0
C(6,2)	0.18	0.23 0.23 0.19	0.19	0	0.5 0.5 0	0
C(9,3)	0.21	0.2 0.24 0.21	0.14	0	0 1 0	0
C(11,17)	0.19	0.2 0.23 0.23	0.15	0	0 0.5 0.5	0
C(15,5)	0.04	0.33 0.33 0.31	0	0	043 0.43 0.14	0

(a)

	$W_{ijkl}(0)$			$a_{ijkl}(t)$		
C(5,10)	0.21	0.17 0.25 0.17	0.2	0	0 1 0	0
C(8,6)	0.19	0.18 0.28 0.17	0.18	0	0 1 0	0
C(11,2)	0.18	0.22 0.24 0.22	0.13	0	0 1 0	0
C(13,16)	0.14	0.21 0.28 0.16	0.2	0	0 1 0	0
C(16,12)	0.27	0.06 0.29 0.15	0.23	0	0 1 0	0

(b)

Table 3.3. Matlab Simulation Results of Minimum Required Elapsed Time Factor: for Maximum Success Rate of Recognition.

Learned patterns	Min elapsed time factor
1-5	8
1-15	15
1-36	15
1-61	17
1-87	17
1-98	18
94-98	14

Table 3.4. Success Rate for Various Sets of Learned Patterns with Noise.

No. of Learned patterns	Noise of input patterns	Recognized patterns	Correct patterns	Success rate
1-93	Uniform noise Level is 0.8	1-93	93	100%
1-93	Normal noise Variance is 0.3	1-93	93	100%
1-98*	Uniform noise Level is 0.8	1-98	95	97%
1-98*	Normal noise Variance is 0.25	1-98	95	97%
95~96*	Uniform noise Level is 0.8	95~96	2	100%
94~98*	Uniform noise Level is 0.8	94~98	5	100%
88, 94~98*	Uniform noise Level is 0.8	88, 94~98	5	90%

- Including 2 patterns in Group 4 as shown in Fig. 10.

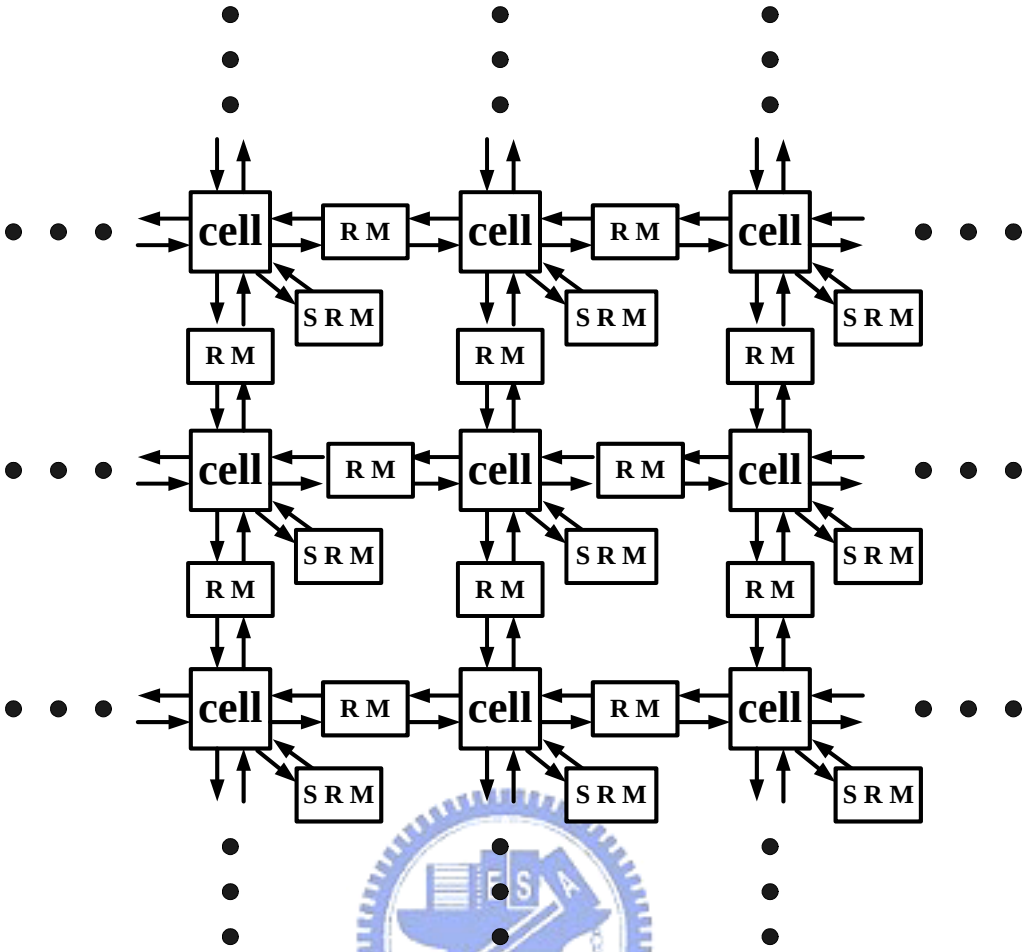


Fig. 3.1. General architecture of the SRMCNN

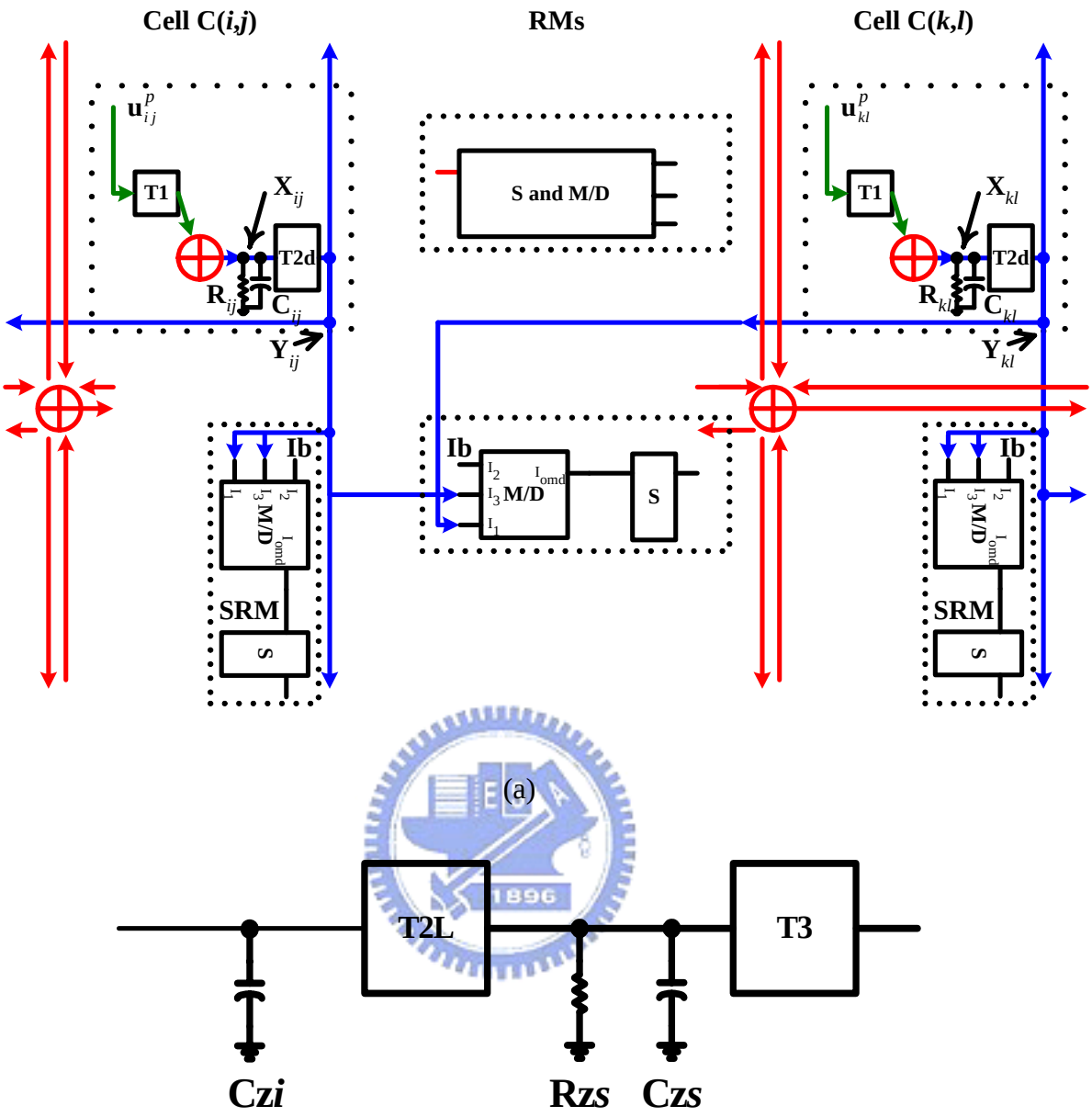


Fig. 3.2. Detailed architecture of (a) two neighboring cells and their ratio memories (RM) and (b) the S block in the SRMCNN during the learning period.

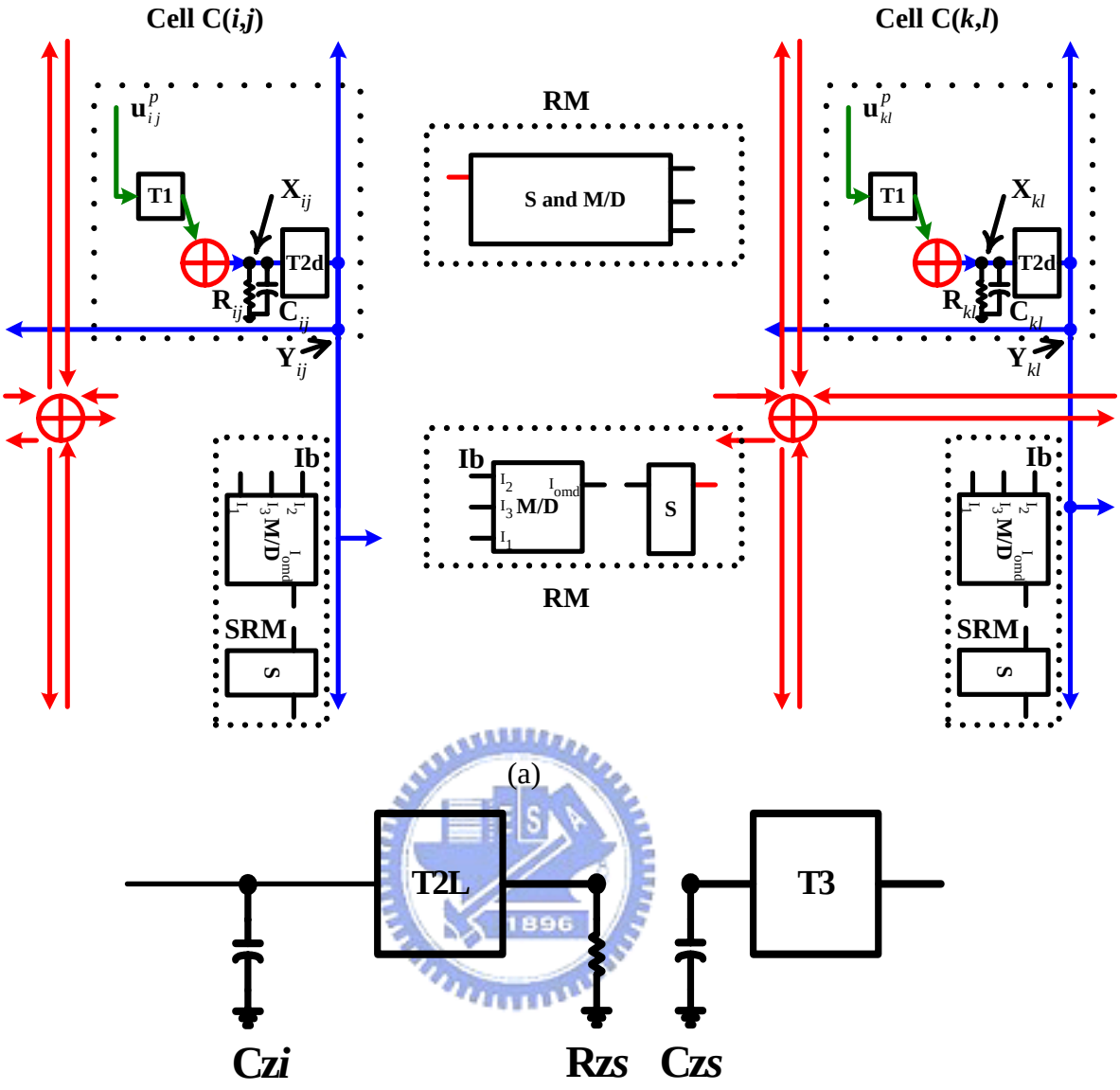


Fig. 3.3. Detailed architecture of (a) two neighboring cells and their ratio memories (RM) and (b) the S block in the SRMCNN during the elapsed period.

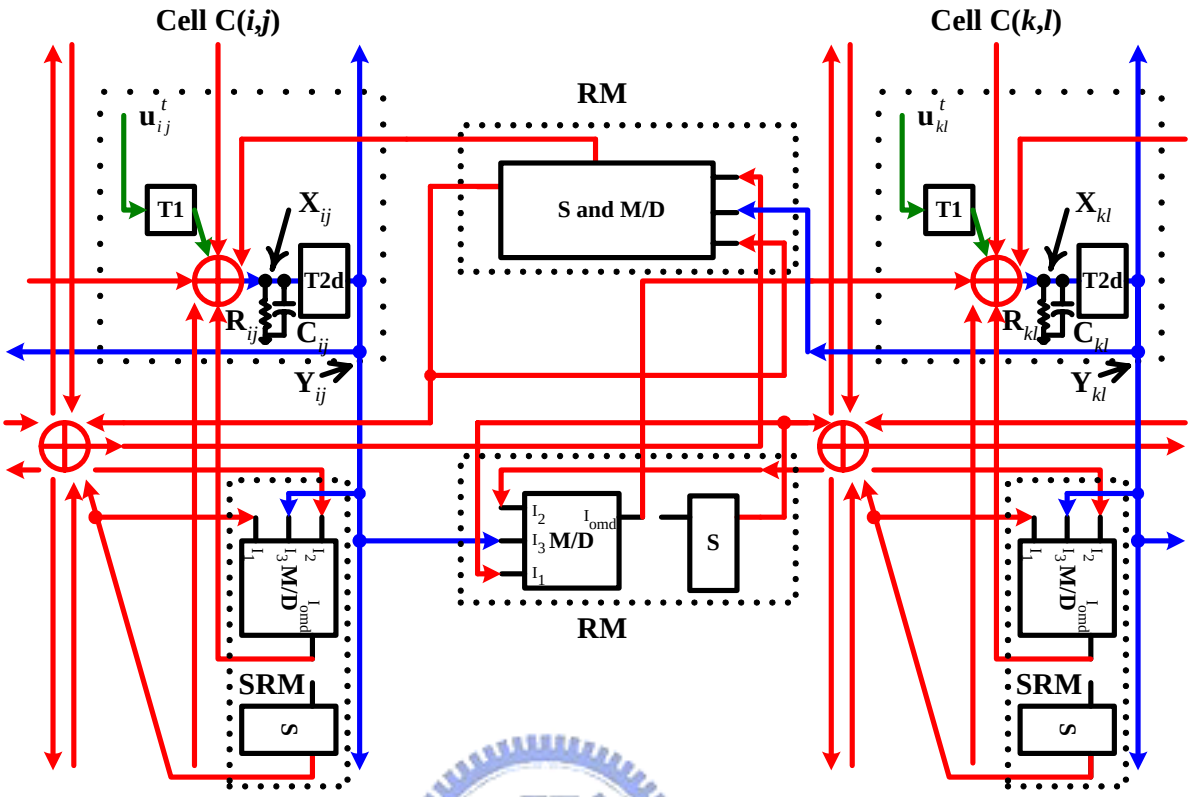
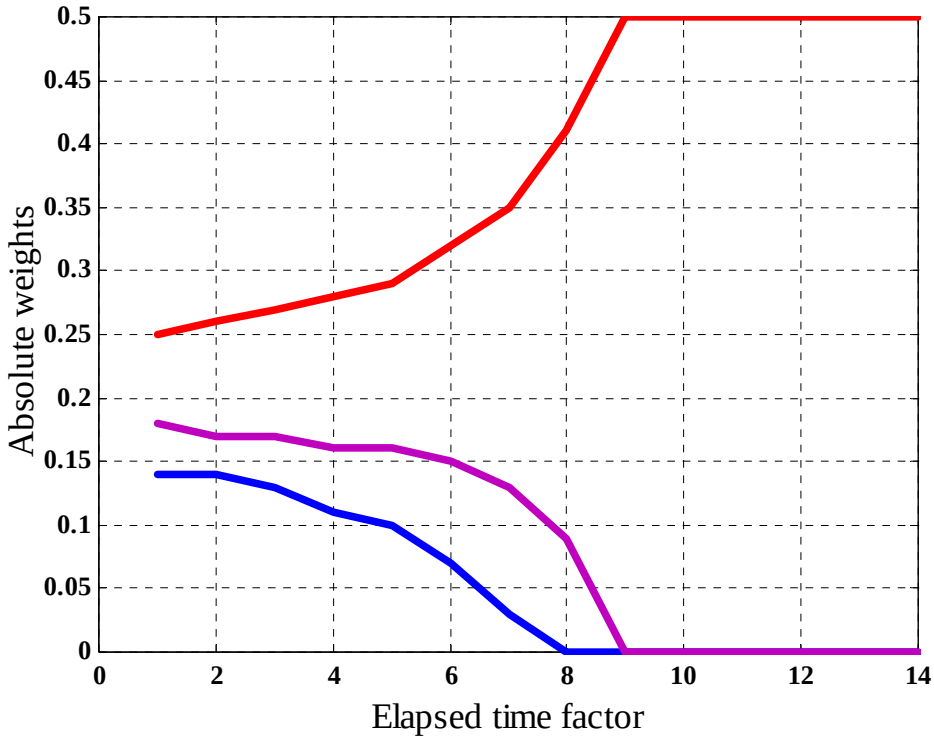
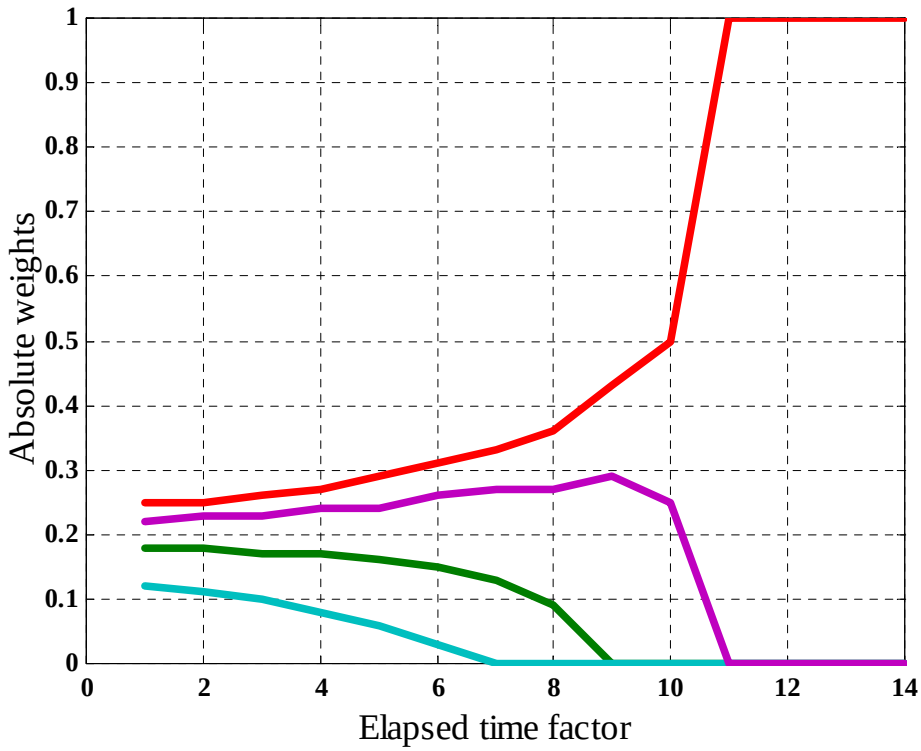


Fig. 3.4 Detailed architecture of two neighboring cells and their ratio memories (RM) in the SRMCNN during the recognition period.

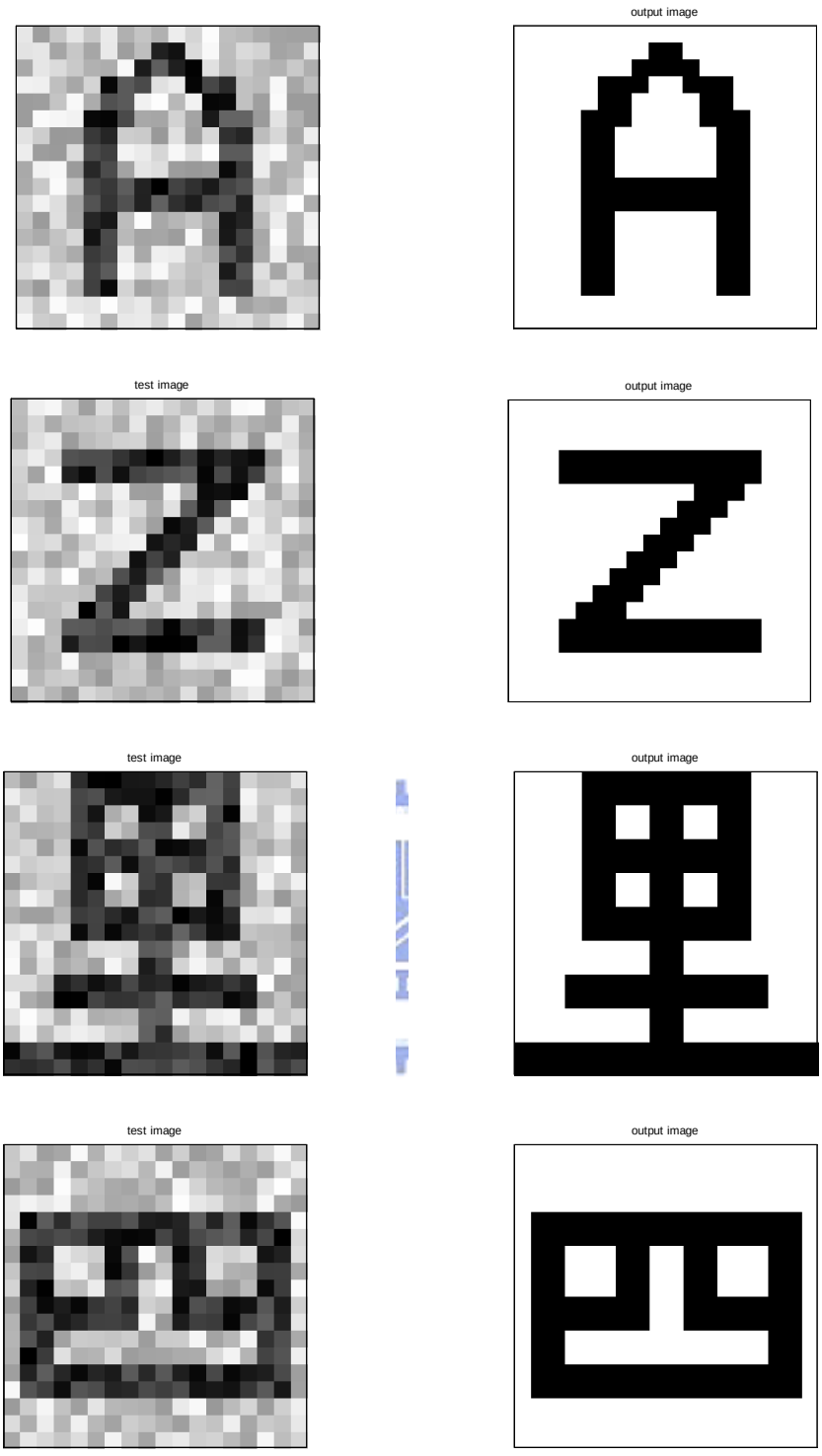


(a)



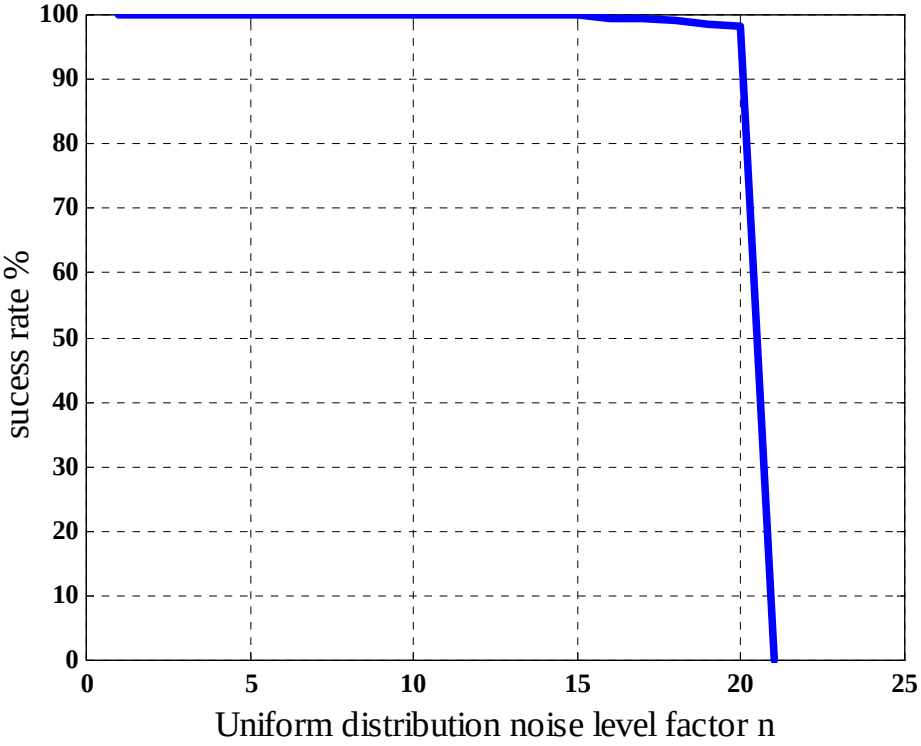
(b)

Fig. 3.5 Variations of the ratioed weights (a) $A_{5,10}$ and (b)$A_{11,2}$ under constant leakage current.

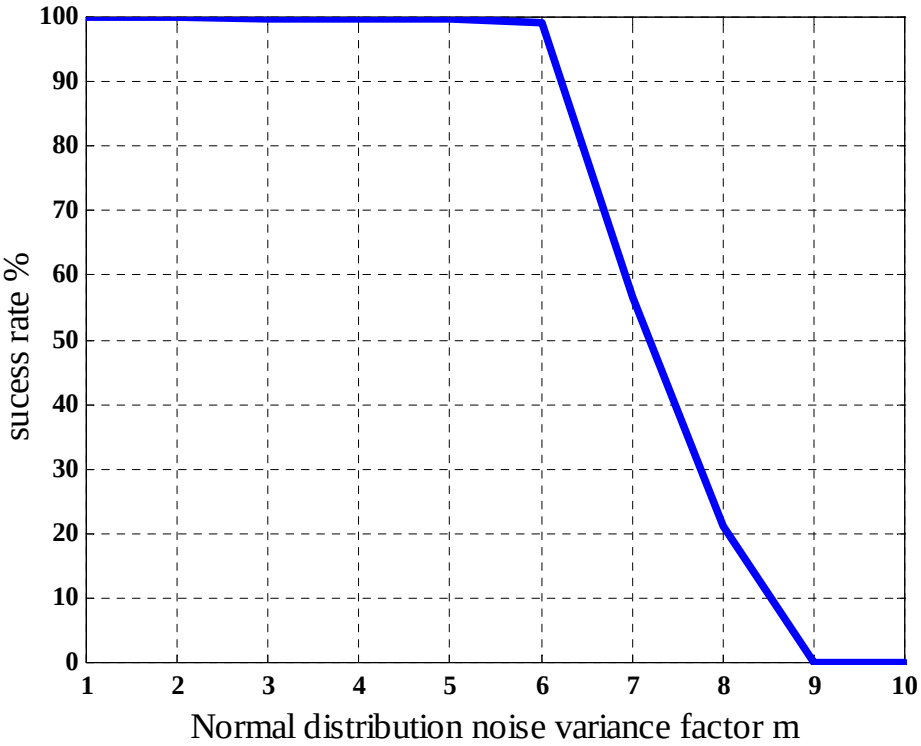


(a) (b)

Fig.3.6 (a) Input test patterns with uniform noise level of 0.8.
(b) Recognized output patterns.

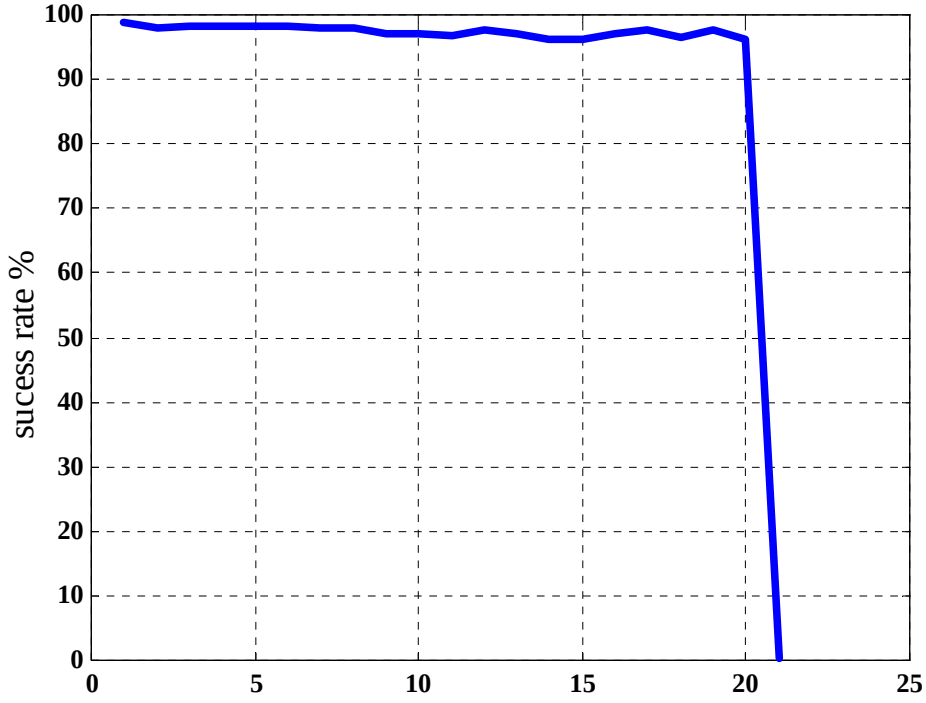


(a)

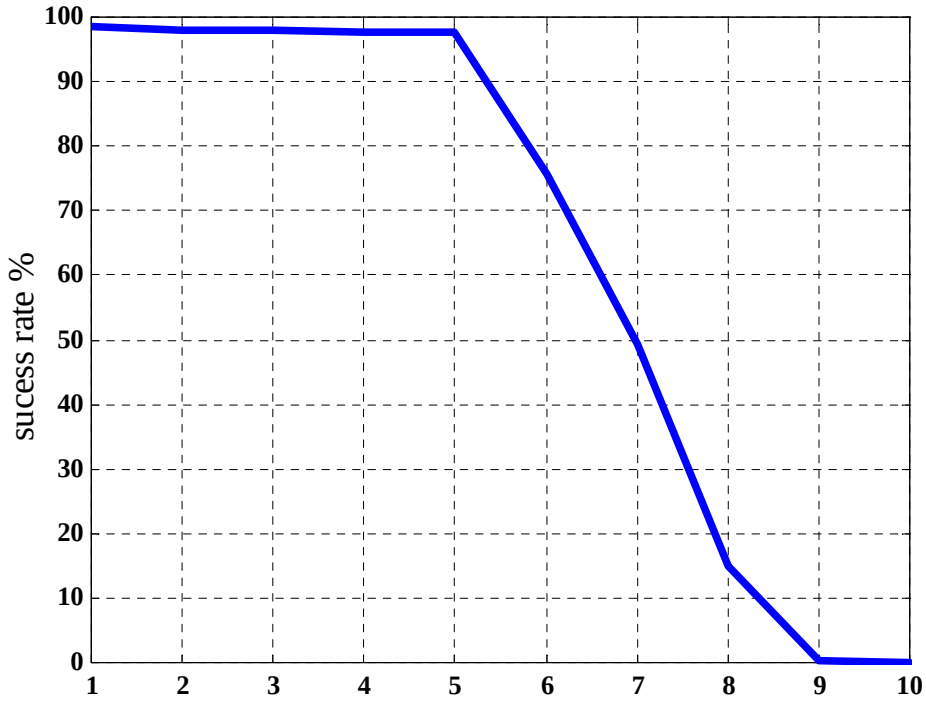


(b)

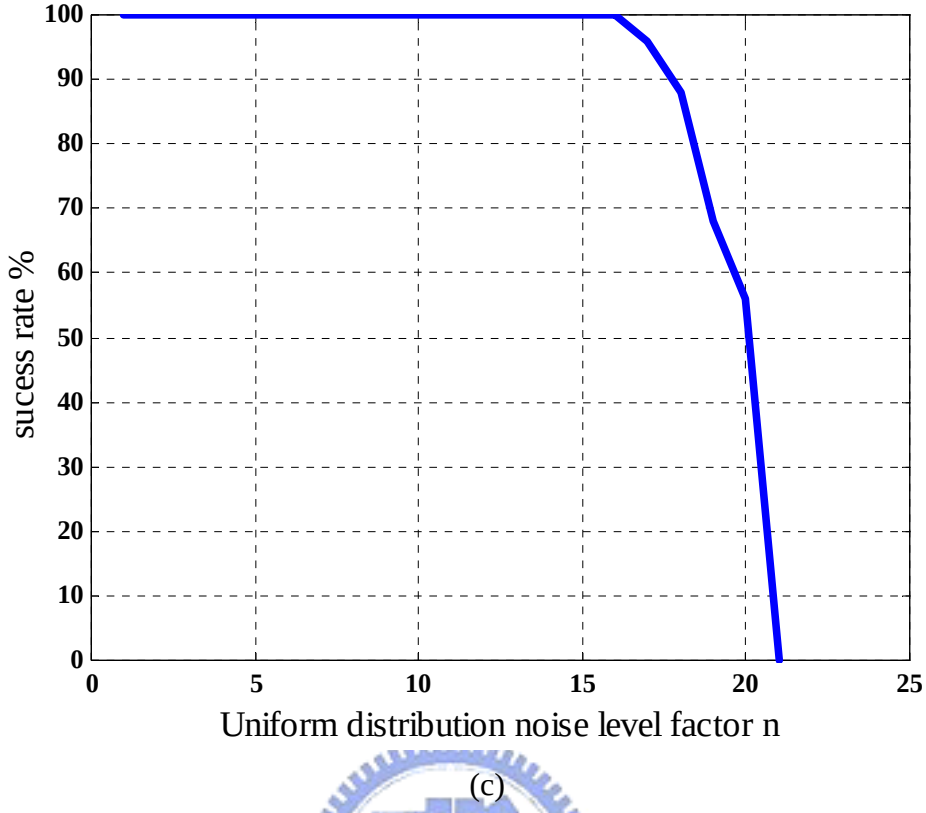
Fig. 3.7. Success rate versus (a) uniform distribution noise level of 0.05n and (b) normal distribution noise variation of 0.05m for 93 (No.1~No.93) noisy test patterns.


sucess rate %
100
90
80
70
60
50
40
30
20
10
0
0
5
10
15
20
25
Uniform distribution noise level factor n

(a)

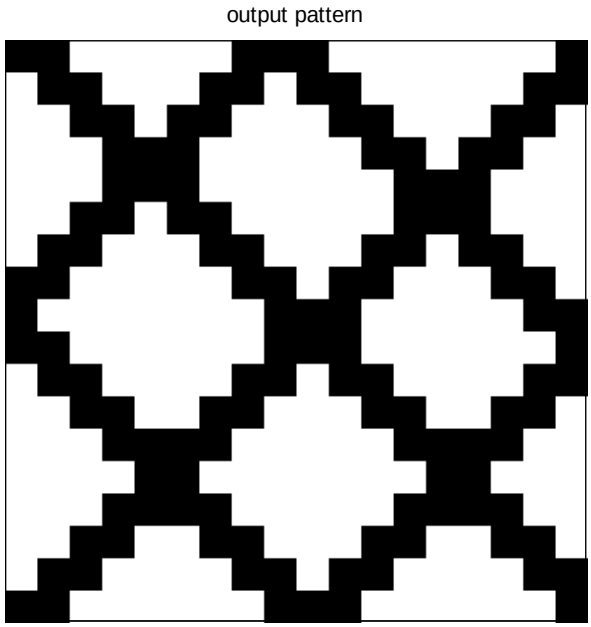

sucess rate %
100
90
80
70
60
50
40
30
20
10
0
1
2
3
4
5
6
7
8
9
10
Normal distribution noise variance factor m

(b)

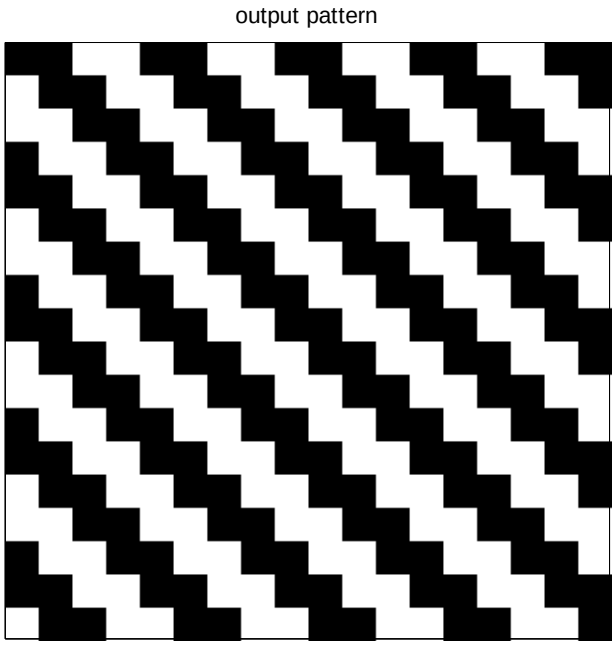


(c)

Fig. 3.8 Success rate versus (a) uniform distribution noise level of 0.05n for 98 (No.1~No.98) noisy test patterns, (b) normal distribution noise variance of 0.05m for 98 (No.1~No.98) noisy test patterns, and (c) uniform distribution noise level of 0.05n for five (No.94~No.98) noisy test patterns.

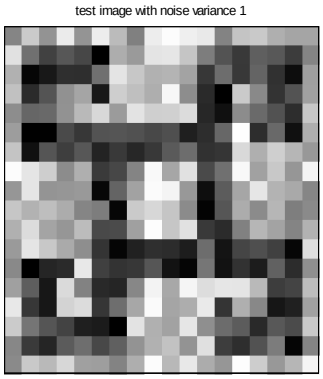


(a)

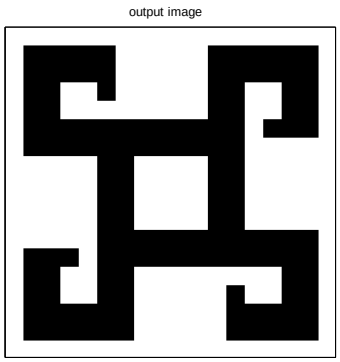


(b)

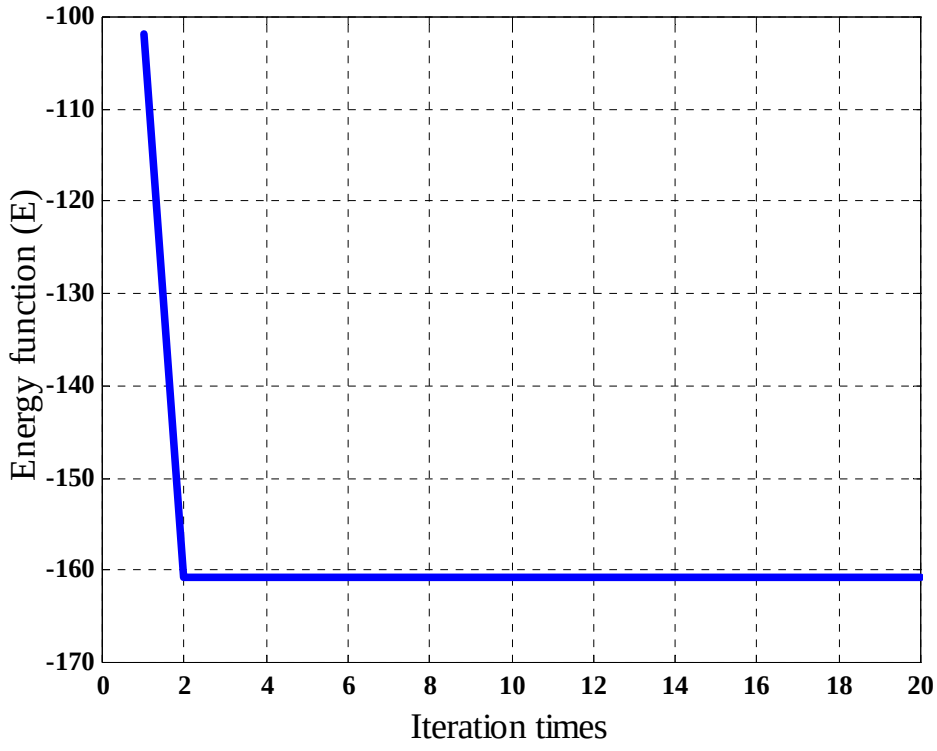
Fig. 3.9. Two specific patterns in Group 4 with only slant lines



(a)



(b)



(c)

Fig. 3.10. (a) Input noisy test pattern, (b) output stable pattern, and (c) energy function during the recognition period.