CHAPTER 4

A LEARNABLE SRMCNN FOR HETERO-ASSOCIATIVE MEMORY APPLICATIONS

4.1 INTRODUCTION

The neural network with locally connected neighboring cells, is called cellular nonlinear network (CNN) as introduced by Chua and Yang [55]-[56]. This network easily implemented in VLSI architecture applied for various image processing and operation. So far, many image processing in CNN with the specific template have been reported [57]-[61]. The given networks can process the input exemplar patterns and output the correct pattern base on the desired function. Most neural networks were designed that each of the patterns to be stored in the associative memory for a local minimum of an associative energy function [59]-[61]. Among the many learning methods published in the literature, the coefficients of the template in the CNN are found by learned from the dynamic equation [62], [76], [106]-[107] a perception-learning rule [111], [133] the tabu search technique and the cost function [141], the modified BP algorithm [102]. The learning algorithm has been incorporated into CNN to generate associative memories for the learning and recognizing of image patterns.

A modified Hopfield network and one Hebbian learning algorithm, called the modified Hebbian learning algorithm, is used to perform the weight learning operation in

the CNN. The ratio memory (RM) in the Grossberg outstar structure is used to realize the weights of template in the neural network for image processing [77], [166], [169]-[175]. The learning algorithm is used to implement the 18×18 CNN for pattern learning over a fixed period. The ratio memory (RM) is also used to form the template coefficients in the CNN for image recognition. The resultant structure is called the ratio-memory CNN (RMCNN) [169]-[175]. Only used the four coefficients of the **A** template but not self-feedback coefficient for the cell C(*i*,*j*) of the proposed RMCNN. This structure is realized with associative memory that can learn and recognize three (five) patterns with Gaussian noise in 9×9 (18×18) neural array [171]-[172], [174]-[175].

The self-feedback RMCNN, is called SRMCNN, the pixel values of the nearest five neighboring cells are multiplied by the pixel value of cell C(*i*,*j*), and the products are summed for all input exemplar patterns. Then, the accumulated result is transformed into a ratio to form the coefficient of the **A** template. The proposed SRMCNN can learn and recognize 87 patterns in the 18×18 neuron array [174]. The structure of RMCNN has been implemented in CMOS technology and its function has been successfully verified in the practically image processing [171]-[172], [174]-[175].

In the thesis, the architecture with embedded ratio memory and realized the modified Hebbian learning algorithm in the SRMCNN with **B** template is presented. It can learning the input exemplar patterns and correctly output the recognized patterns. The weights of the **B** template are generated from the desired output pixel value produced by the nearest five neighboring element as associative memory for all input exemplar patterns. The learned weights are processed in the ratio with the summation of absolute coefficients on the **B** template. The efficiency of ratio memory is enhanced the feature of pattern. The simulation results of the behavior and function of SRMCNN with **A** and **B** templates for hetero-associative memory applications are demonstrated and analyzed. The capability of SRMCNN for pattern learning and recognition is improved.

The chapter is organized as follows. In Section 4.2, the architecture of the SRMCNN with the modified Hebbian learning algorithm for hetero-associative memory is presented. The learning and the embedded ratio memory operation for the **B** or **A**、**B** templar in the VLSI architecture of the SRMCNN are proposed. In Section 4.3, the simulation results of SRMCNN are demonstrated and analyzed. Some phenomena are also discussed. Finally, summaries are drawn.

4.2 SRMCNN STRUCTURE

4.2.1 SRMCNN with B Template

In a two-dimensional CNN, any cell is only connected to its neighboring cells, the connected line segments are expressed as weights, the cell state $X_{ij}(t)$ and the cell output $Y_{ij}(t)$ for a regular cells can be expressed as [55]-[56]

$$\dot{X}_{ij}(t) = -X_{ij}(t) + \sum_{C(k,l)\in Nr(i,j)} a_{ijkl}(t) Y_{kl}(t) + \sum_{C(k,l)\in Nr(i,j)} b_{ijkl}(t) u_{kl} + z_{ij} \quad (4.1)$$

$$y_{ij} = f(x_{ij}) = \frac{1}{2}\left|x_{ij} + 1\right| - \frac{1}{2}\left|x_{ij} - 1\right| \quad (4.2)$$

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), u_{kl} is the cell input from the cell C(k, l) of $Nr(i, j)$, z_{ij} is threshold of the cell C(i, j), $f(\cdot\cdot)$ is bipolar activation function, a_{ijkl} and b_{ijkl} are the weights from $Y_{kl}(t)$ and u_{kl} of the cell, respectively. $Nr(i, j)$can be expressed by the following equation.

$$Nr(i, j)=\{ C(k, l)|1\le k \le M, 1\le l \le N; |k-i| \le r, |l-j| \le r \} \quad (4.3)$$

In this thesis, the coupled **B** template and uncoupled **A** template in the CNN structure with r=1 neighborhood is adopted. The coefficients of the space-variant **B** template are corresponding to the input and four nearest neighboring coefficients for the cell (i, j). The

A template has only one self-feedback coefficient. **A** and **B** templates are expressed as

$$A_{ij}^{1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \boldsymbol{B}_{ij}^{1} = \begin{bmatrix} 0 & b_{ij(i-1)j} & 0 \\ b_{iji(j-1)} & b_{ijij} & b_{iji(j+1)} \\ 0 & b_{ij(i+1)j} & 0 \end{bmatrix} \tag{4.4}$$

The outermost boundary cells are called edge cells that commonly used for fixed (Dirichlet) boundary conditions. Those cells are prescribed to zeros.[57]

A CNN with ratio memory (RM) is shown in Fig. 4.1 where the RM is used to realize the weights of the neighboring cells. In the ratio memory (RM) model, the learned weights are transformed into the ratio weight Wb_{ijkl} stored in the associative memory for equilibrium equation of the cell state at an equilibrium point as

$$\mathrm{x}_{ij}(k+1) = \sum_{C(k,l)\in N_r(i,j)} Wb_{ijkl}(k) \cdot \mathrm{u}_{kl}(k) + y_{ij} + \mathrm{z}_{ij} \tag{4.5}$$

The equilibrium point x is globally asymptotical stable base on the locally stable in the sense of the energy function and globally attractive.

To realize the SRMCNN as the associate memory can store a large set of exemplar patterns for the recurrent hetero-associative memory. Using the input test exemplar patterns and the desired output pattern to training the SRMCNN, the weights of the **B** templates can be found.

The weight-learning algorithm is shown in Fig. 4.2. The modified Hebbian learning algorithm is applied in the learning block to find the increment $dw_{ij} = y_{ij}u_{kl}$ of the weight vector. The value of dw_{ij} is positive then $Wb_{ijkl}(0)$ is increased. Otherwise, $Wb_{ijkl}(0)$ is decreased. The learned weights recursive update operation with m distinct exemplar patterns was generated at time t=0 as

$$Wb_{ijkl}(0) = \sum_{p=1}^{m} y_{ij}^{p} u_{kl}^{p} \qquad C(k,l) \in N_r(i,j) \tag{4.6}$$

$$Zij(0)=0 \tag{4.7}$$

$$br_{ijkl}(0) = \frac{Wb_{ijkl}(0)}{\sum_{C(k,l)\in N_r(i,j)} \left| Wb_{ijkl}(0) \right|} \tag{4.8}$$

where m is the number of learning patterns, y_{ij}^{p} is the desired neuron output 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 inputted pixel of the cell *C(k,l)* of *Nr* neighboring cells including cell *C(i,j)*, $Wb_{ijkl}(0)$ in (4.6) is the weight associated with cell *C(i,j)* and its neighboring cells *C(k,l)*, and $z_{ij}(0)$ in (4.7) is the threshold of cell *C(i, j)* which is set to zero.

Updating the weights of the **B** template was performed in parallel and the operation is stop until all patterns are learned during the learning period. In order to limit the value of the weight, the weights $Wb_{ijkl}(0)$ are to be normalized by the ratio with the summation of the all absolute neighboring weights. The normalized ratio weights $br_{ijkl}(0)$ were stored in the ratio memory.

In the elapsed period, the leakage current $I_{leakage}$ associated with capacitor C*zs* gradually decreased the stored voltage of capacitor C*zs*. Since the leakage current is nearly constant, the change of storage voltage $br_{ijkl}(t)$ stored on the capacitor C*zs* can be written as

$$\left| br_{ijkl}(t) \right| = \left| br_{ijkl}(0) \right| - \frac{I_{leakage}}{C_{zs}} t \tag{4.9}$$

The RM was also used to normalize the **B** template coefficients in the recognition period. The patterns were based on spatial averaging operations performed on the local neighborhoods of the cell inputs. The elapsed weights $br_{ijkl}(t)$ are transformed into the ratio weight b_{ijkl} as [174]-[175]

$$b_{ijkl}(t) = \frac{br_{ijkl}(t)}{\sum_{C(k,l)\in N_r(i,j)} \left| br_{ijkl}(t) \right|} \tag{4.10}$$

where the $b_{ijkl}(t)$ is the ratio weight of template **B**. The ratio weight $b_{ijkl}(t)$ generated by the

M/D block had the effect of normalization and feature enhancement. With the weight intensity larger than the mean intensity of the weight b_{ijkl} of cell C(i,j), the weight increased gradually. Otherwise, the weight decreased gradually. The weights are correlated with the nearest neighboring of the cell input and the output of the cell. The neighboring pixels in the exemplar pattern are contributive enhanced to reject the heaviest noise for the corresponding cell. Thus, the network capability of the recognition for the noisy patterns with white-black is improved by the proposed SRMCNN.

In the recognition state, the cell outputs are adjusted by the learned ratio weights gradually to close one of the features of the training patterns. The energy function of a CNN in quadratic form [64], [138] can be expressed as

$$E = -\frac{1}{2}\sum_{ij}\sum_{kl}(b_{ijkl}u_{kl})Y_{ij} \tag{4.11}$$

The energy function is also tendency converged to its local minimum until the outputs can't be changed. The final recognized pattern is found at the minima of E in the stable state.

In (4.10), the absolute value is added in the denominator because the connection weight br_{ijkl} can be both positive and negative value and the ratio memory take the total magnitude of each weight element in the ratio memory to normalize the weight. The neuron state change in SRMCNN is

$$\begin{aligned} y_{ij} &\longrightarrow y_{ij}^{-1} \qquad if \quad \sum_{\mathrm{C}(k,l)\in N_r^0(i,j)} b_{ijkl}u_{kl} < 0 \\ &\longrightarrow y_{ij}^{+1} \qquad if \quad \sum_{\mathrm{C}(k,l)\in N_r^0(i,j)} b_{ijkl}u_{kl} > 0 \end{aligned} \tag{4.12}$$

where y_{ij}^{-1} and y_{ij}^{+1} are the two cell outputs -1 and +1, respectively. Because the ratio memory weight in SRMCNN is no longer symmetric as the weight in the original Hopfield network. The stability of the proposed SRMCNN needs to further consider.

Consider an energy function

$$E_{RM} = -\sum_{(i,j)=(1,1)}^{(n,n)} \sum_{C(k,l)\in N_r^0(i,j)} b_{ijkl} u_{kl} y_{ij} \tag{4.13}$$

Suppose y_{ij} is changed, then the energy change can be expressed as

$$\Delta E_{RM} = -\sum_{C(k,l)\in N_r^0(i,j)} b_{ijkl} u_{kl} \Delta y_{ij} - \sum_{(i,j)=(1,1)}^{(n,n)} b_{ijkl} u_{ij} \Delta y_{kl}$$

$$= \Delta E_{RM1} + \Delta E_{RM2} \tag{4.14}$$

The first term ΔE_{RM1} of (4.14) can be proven to be always non-positive as the proof method in the original Hopfield energy function *E*, if $\Delta y_{ij} \geq 0$, $\sum_{C(k,l)\in N_r^0(i,j)} b_{ijkl} u_{kl} \geq 0$ and if $\Delta y_{ij} \leq 0$, $\sum_{C(k,l)\in N_r^0(i,j)} b_{ijkl} u_{kl} \leq 0$. Consider the second energy change term ΔE_{RM2}

$$\Delta E_{RM2} = -\sum_{(i,j)=(1,1)}^{(n,n)} b_{ijkl}\, u_{ij}\, \Delta y_{kl}$$

$$= -\sum_{(i,j)=(1,1)}^{(n,n)} \frac{br_{ijkl}}{\sum_{C(k,l)\in N_r^0(i,j)} \left| br_{ijkl} \right|} u_{ij}\, \Delta y_{kl} \tag{4.15}$$

Because the connection weight in the original Hopfield model is symmetric, i.e. $w_{ijkl} = w_{klij}$, (4.15) can be written as

$$\Delta E_{RM2} = -\frac{\sum_{i,j=(1,1)}^{(n,n)} br_{klij} u_{ij}}{\sum_{C(k,l)\in N_r^0(i,j)} \left| br_{ijkl} \right|} \Delta y_{kl} \tag{4.16}$$

From (4.16), it can also be proven that ΔE_{RM2} is always non-positive. Thus, under the assumption that the connection weight is symmetric in the original Hopfield model, the stable state of the SRMCNN model is also existed.

4.2.2 SRMCNN Architecture

The detailed block diagram of the SRMCNN in the two neighboring CNN cells and their RM is shown in Fig. 4.3(a), and the detailed block diagram of the S block is shown Fig. 4.3(b) when the SRMCNN is operated during the learning period. In Fig. 4.3(a), the block T1 is a V-I converter used to convert the voltage of the state into the output current of the cell. 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 input current and detect the sign of the cell state $X_{ij}(t)$, respectively. The CNN cell *C(i,j)* is consisted with T1, R_{ij}, and C_{ij} elements [171], [138].

The block M/D [171] in Fig. 4.3(a) is a combined four-quadrant multiplier and a two-quadrant divider circuit. The block is used to realize the modified Hebbian learning algorithm during the learning period. It is also used to multiply perform the multiplication b_{ijkl}(t) and u_{kl}(t) in the recognition period. The resultant absolute weight zi_{ijkl} during the learning period is stored in the capacitor *Czi*, transfers the absolute voltage *VCzs* stored to *Czs* and stores its sign in the latch circuit in the S block of Fig. 4.3(b). 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 weight currents from neighboring cells. The summed current is sent to the M/D block to generate ratio-memory.

In Fig. 4.3(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 T2d to be converted into current $\mathrm{I}u_{ij}^{p}$. The desired output currents $\mathrm{I}y_{ij}^{p}$ are obtained by the decode circuit D from p_sel corresponding to the value of m. Then the converted absolute currents $\left|\mathrm{I}y_{ij}^{p}\right|$ and $\left|\mathrm{I}u_{kl}^{p}\right|$ from the neighboring cells are distinctly sent to the four-quadrant multiplier in the

M/D block to charges the capacitor *Czi* for the period T_P. This operation is repeated for *m* patterns to sum the produced voltages in *Czi*. Finally, the weight voltage $Vzi_{ijkl}(0)$ stored on *Czi* at $t = 0$ when the learning period ends, can be written as

$$\mathrm{V}zi_{ijkl}(0) = \frac{1}{\mathrm{C}zi}\sum_{p=1}^{m}\left(\int_{T_P}\frac{\mathrm{I}y_{ij}^{p}\,\mathrm{I}u_{kl}^{p}}{\mathrm{Ib}}dt\right) \qquad C(k,l) \in N_r(i,j) \tag{4.17}$$

where $\mathrm{I}y_{ij}^{p}$ is the current of the *p*-th desired output of 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.

The weight $W_{ijkl}(0)$ is directly generated by the current product of $\mathrm{I}y_{ij}^{p}\mathrm{I}u_{kl}^{p}$ changing on the capacitor *Czi*, $\mathrm{V}zi_{ijkl}(0)$ is also generated for the period *Tp*. 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. 4.4(a), where *Czs* is disconnected from the block T2L as shown in Fig. 4.4(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. 4.5 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 T2d and converted into the pixel current $\mathrm{I}u_{ij}^{t}$ and the sign voltage Vsx_{ij}. 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 the other neighboring cells. The summed current, the weight current $abs[Izs_{ijkl}(t)]$, and the cell input current $I_{ukl}(t)$ are sent to the M/D block to yield the current that corresponds

to the term $b_{ijkl}(t)\ u_{kl}(t)$ in (4.1), 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

$$\mathrm{V}x_{ij}(t) = \mathrm{R}_{ij}\left(\sum_{\mathrm{C}(k,l)\in N_r(i,j)} \mathrm{K}_{\mathrm{A}} \frac{\mathrm{I}_{u_{kl}}(t)\mathrm{I}zs_{ijkl}(t)}{\sum_{\mathrm{C}(k,l)\in N_r(i,j)} abs[\mathrm{I}zs_{ijkl}(t)]} + \mathrm{I}y_{ij}^{t} + \mathrm{I}z_{ij} \right) \tag{4.18}$$

where K_{A} is the empirical gain. Ideally $\mathrm{K}_{\mathrm{A}} = 1$.

The ratioed weight $\mathrm{I}zs_{ijkl}(t) \Big/ \sum_{\mathrm{C}(k,l)\in N_r(i,j)} abs[\mathrm{I}zs_{ijkl}(t)]$ in (4.10) 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 $I_{uij}\ (t)$ 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 $u_{kl}(t)$ 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 T1 to generate the output current $I_{Yij}(\mathrm{t})$.

4.2.3 SRMCNN with A and B Template Architecture

The SRMCNN structure with the coupled **A** and **B** templates with r=1 neighborhood for hetero-associative memory applications is proposed. The space-variant **A** template has a self-feedback coefficient and four nearest neighboring coefficients. The coefficients of the space-variant **B** template are corresponding to the input and four nearest neighboring coefficients for the cell (i, j). Both **A** and **B** templates are expressed as

$$\boldsymbol{A}_{ij} = \begin{bmatrix} 0 & a_{ij(i-1)j} & 0 \\ a_{iji(j-1)} & a_{ijij} & a_{iji(j+1)} \\ 0 & a_{ij(i+1)j} & 0 \end{bmatrix} \boldsymbol{B}_{ij}^{1} = \begin{bmatrix} 0 & b_{ij(i-1)j} & 0 \\ b_{iji(j-1)} & b_{ijij} & b_{iji(j+1)} \\ 0 & b_{ij(i+1)j} & 0 \end{bmatrix} \tag{4.19}$$

The outermost boundary cells are called edge cells that commonly used for fixed (Dirichlet) boundary conditions. Those cells are prescribed to zeros.

The modified Hebbian learning algorithm as a supervised Hebbian learning, is applied in **A** template as B template for the SRMCNN in (4.6) to determine the updated volume of the weight vector at t = 0 as

$$Wa_{ijkl}(0) = \sum_{p=1}^{m} u_{ij}^{p} y_{kl}^{p} \qquad C(k,l) \in N_r(i,j) \tag{4.20}$$

Using the input test exemplar patterns and the desired output pattern to training the SRMCNN, the weights of the **A** and **B** templates can be found. The learned weights Wa_{ijkl} and Wb_{ijkl} in (4.6) are transformed into the ratio weight as normalize operation stored in the associative memory for the equilibrium equation of the cell state at an equilibrium point as

$$\mathrm{x}_{ij}(k+1) = \sum_{C(k,l)\in N_r(i,j)} Wa_{ijkl}(k) \cdot \mathrm{y}_{kl}(k) + \sum_{C(k,l)\in N_r(i,j)} Wb_{ijkl}(k) \cdot \mathrm{u}_{kl}(k) + \mathrm{z}_{ij} \tag{4.21}$$

The equilibrium state x is globally asymptotical stable base on the locally stable in the sense of the energy function and globally attractive.

In the ratio memory (RM) model, the normalized weights ar_{ijkl} and br_{ijkl} are elapsed for a selected time period, the ratioed weights a_{ijkl} and b_{ijkl} are found by divided with the summation of the absolute neighboring weights, respectively. Then the resultants are stored in the capacitor storage as associative memory. The realized SRMCNN can learn a large set of exemplar patterns. The weights are stored in the associative memory. The proposed system is able to use for the recurrent hetero-associative memory applications. The capability of the image processing of the SRMCNN with **A** and **B** template are more and more improved

4.3 SIMULATION RESULTS

Using Matlab software to simulate the proposed SRMCNN with 18×18 neurons, direct neighborhood (r=1), and the modified Hebbian learning algorithm for associative memory. The SRMCNN would process the pattern with the block pixel expressed by +1 and the white pixel was -1. In the system, the constant leakage current 0.8fA and the associative memory storage capacitor Czs of 2pF are used in the simulation.

It was found that the correct of pattern recognition is related to the elapsed time factor of the ratio memory. The elapsed time factor is the normalized elapsed time to 50 second. Observe the result from the error function that the values of error are reached to zero at the elapsed time factor over 9, which is equal to 450 second, as shown in Fig. 4.6.

Due to the feature enhancement effort, its weights for the **B** template are transformed into ratio weights during the ended of the learning period. As verified from the simulating results, The SRMCNN with **B** template able to learn nine patterns and successfully recognized the corresponding eight noisy patterns with white-black noise. The success rate can reach to 100%. Fig. 4.7 shows the test patterns with noise for the English character A. The SRMCNN with **B** template can recognize those patterns and output the desired correct pattern for the character A. While the learning patterns expanded the more characters or the clearly distinct from the output then the recognition rate is largely reduced.

The SRMCNN with **A** and **B** template can be recognized the complex patterns after through the learning. The ratioed weights of **A** and **B** template are produced from the learned weights divided with the absolute summation for its five neighboring weights, respectively. The proposed SRMCNN can learn five exemplars patterns to product the weights of **A** and **B** template for hetero-associative memory learning. The elapsed learn weights are transforming into the ratioed weights from the summing for absolute the five

nearest neighboring weights are stored in the associative memory.

The exemplar patterns are presented the normal, expand, left-rotate, right-rotate, and reversal of the A character, as shown in Fig. 4.8. The desired SRMCNN outputs are the pattern A. While the test pattern of the expanded A character into the SRMCNN as the first one shown in the Fig. 4.9. The processing procedure of the recognition is presented from the 2nd to 8th patterns, the last one is the final recognized correctly output pattern, as shown in the Fig. 4.9. Another test case, using the left-rotate A character in the SRMCNN to process the image recognition. Fig. 4.10 has shown the resultant of the recognition in sample for the 150 times iterations. In the Fig. 4.10 to show, the correct pattern at the last one in the figure is recognized. Thus, the SRMCNN with **A** and **B** template can be recognized by the learned ratio weights for the complex test example pattern.

4.4 SUMMARY

In this thesis, the self-feedback ratio memory cellular nonlinear network with **B** template for associative memory is proposed and analyzed. In the SRMCNN, the five weights in the **B** template are generated and updated from a set of exemplar patterns, and the ratioed weights are in the ratio of the absolute summation of its neighborhood was stored in the associative memory. From the simulation results, it has been verified that the 18×18 SRMCNN can learn and recognize nine patterns with white-black noise for the character A. The recognition rate was reduced for more than one-desired patterns of the English characters. The SRMCNN able to learn and recognize more patterns as compared to the CNN without RM, the capability is similar Hopfield neural network but with less network complexity.

The SRMCNN with **A** and **B** template through the exemplar patterns learn and the

ratio operate, its weights are stored in the associative memory. The SRMCNN have the capability to process the singularity test patterns and output correctly the recognized pattern. As the results, input test patterns as normal, expanded, left-rotate, right-rotate, and reversal of the English character A, the system is able to recognize the correct pattern after the some iterations period. Therefore, the proposed SRMCNN with the modified Hebbian learning algorithm capability for the pattern learning and recognition has improved. Moreover, the proposed SRMCNN can easily implement and fabricated by the mixed signal technology in VLSI for various applications.

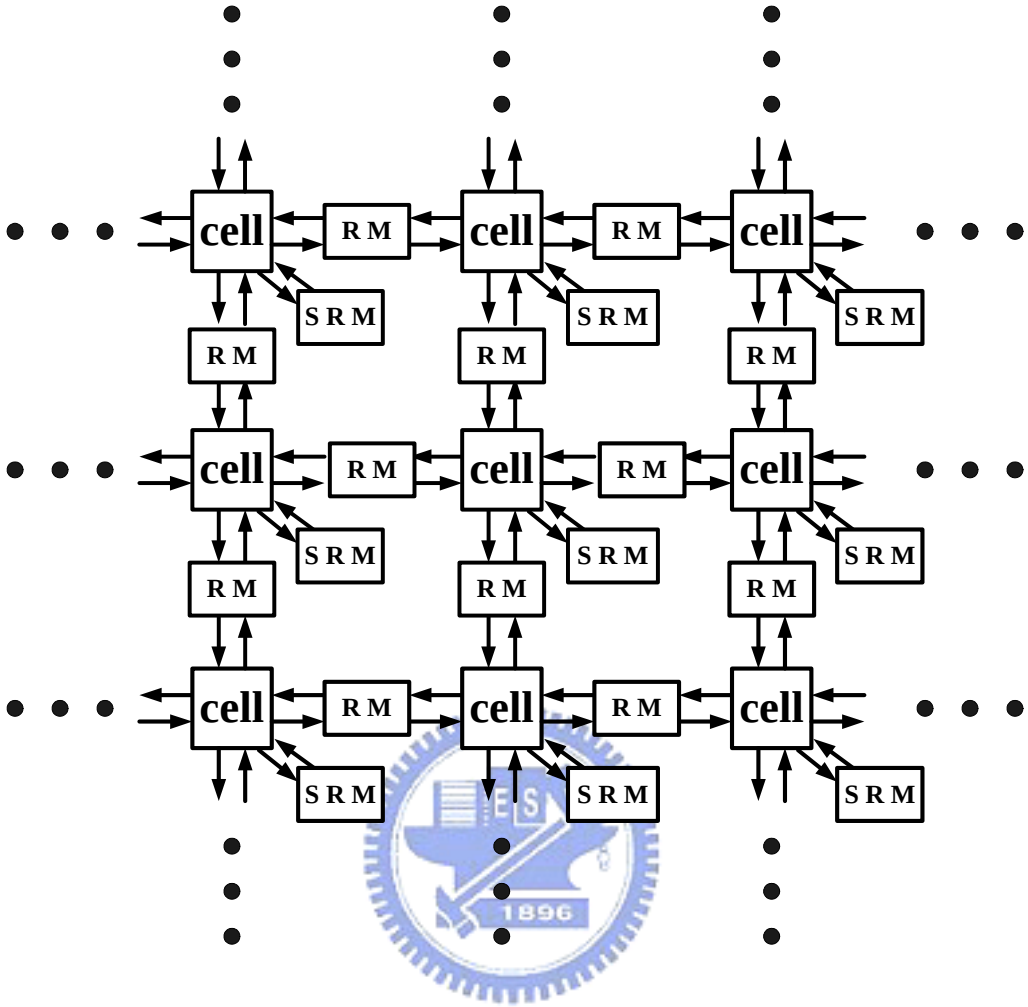


Fig. 4.1. General architecture of the SRMCNN

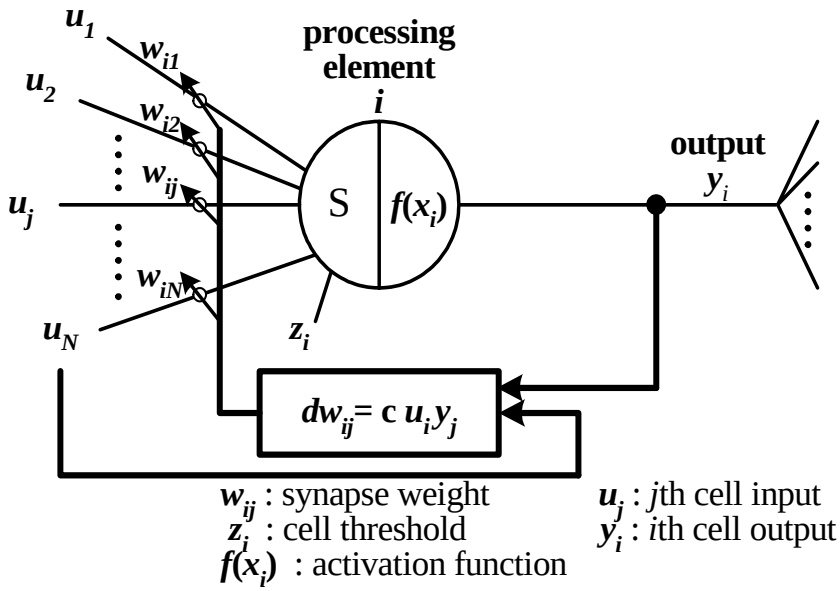


Fig.4.2 Illustration for weights learning algorithm

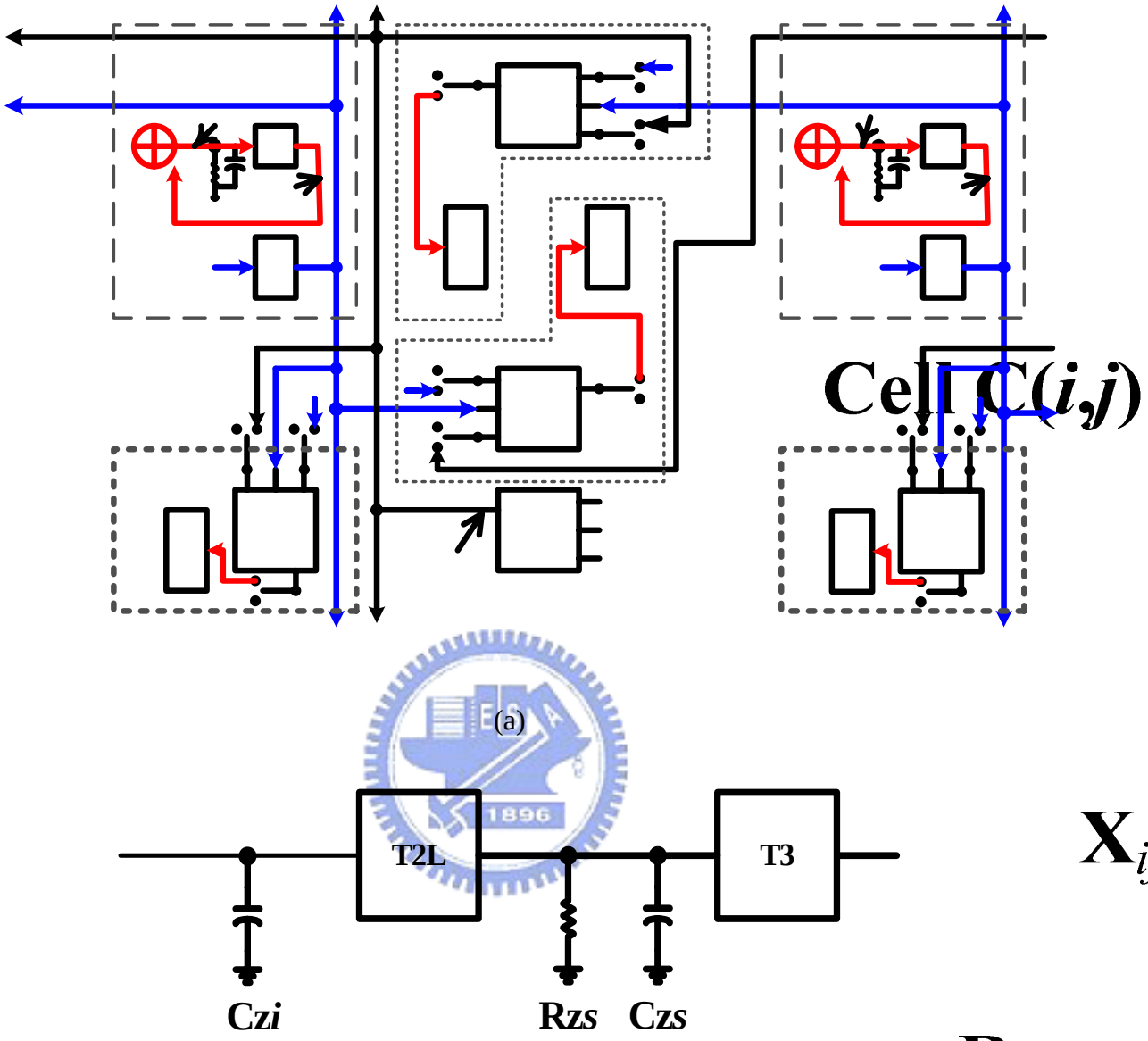


Fig. 4.3 Detailed architecture of (a) two neighboring cells and their ratio memories (RM) and (b) the S block in the SRMCNN during the learning period.

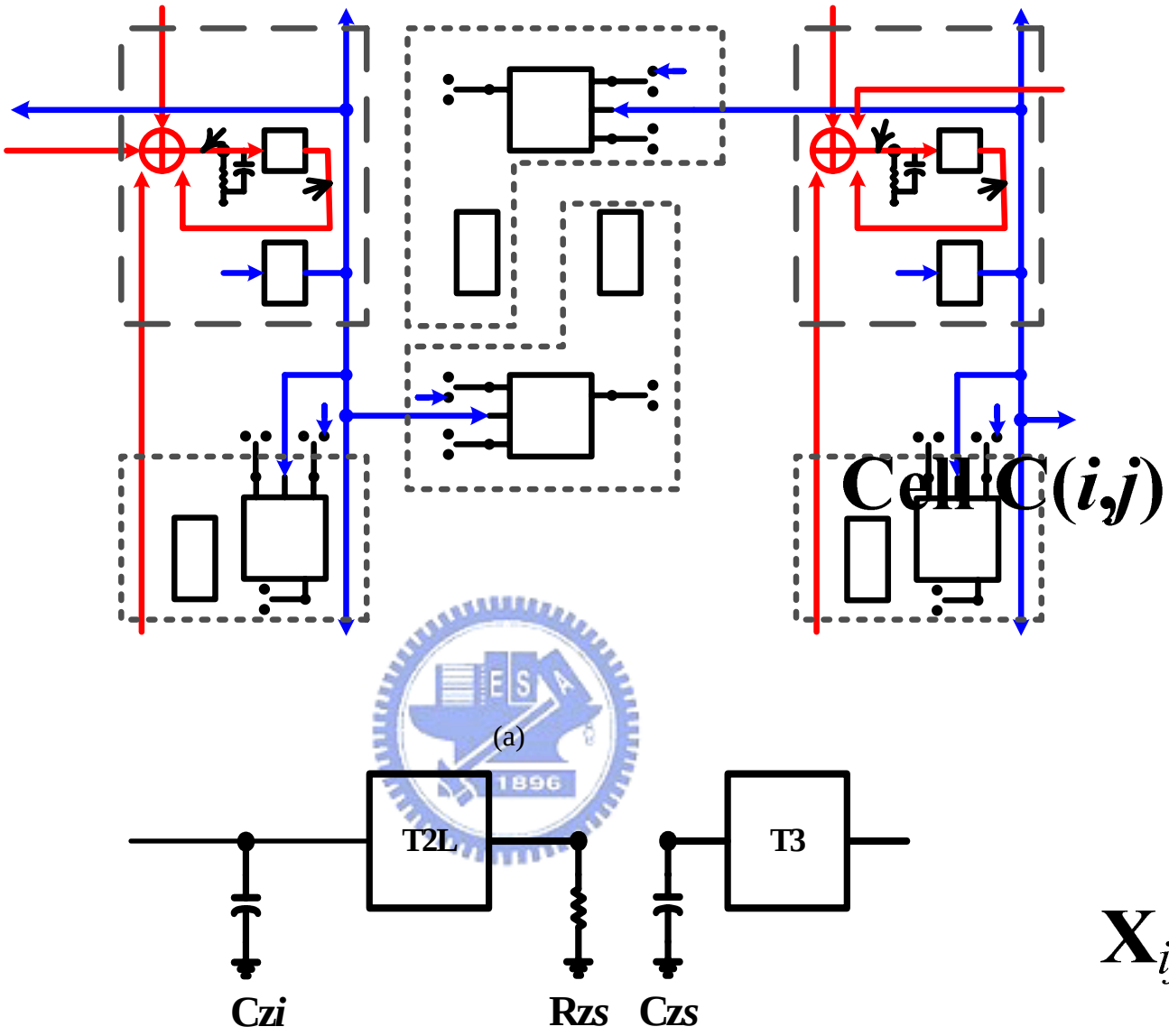


Fig. 4.4 Detailed architecture of (a) two neighboring cells and their ratio memories (RM) and (b) the S block in the SRMCNN during the elapsed period.

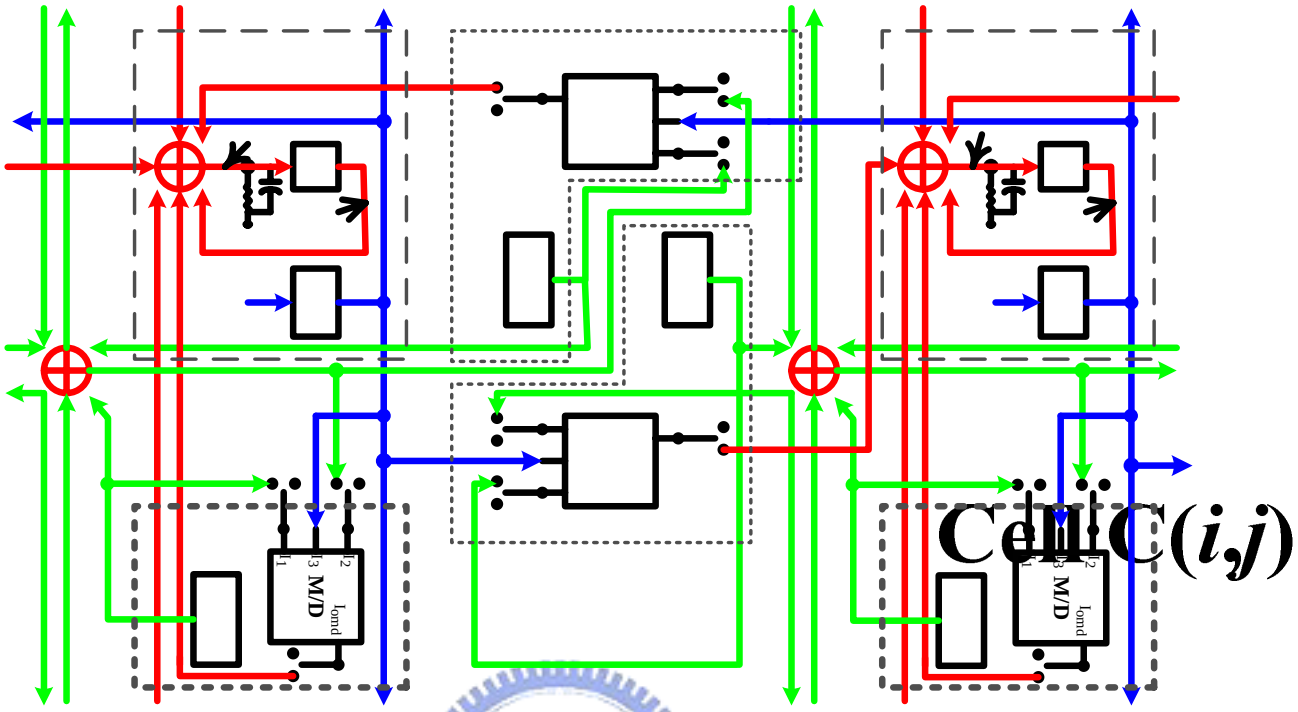


Fig. 4.5 Detailed architecture of two neighboring cells and their ratio memories (RM) in the SRMCNN during the recognition period.

$\mathbf{X}_{ij}$

$\mathbf{R}_{ij}$ C

$\mathbf{u}_{ij}^{t}$

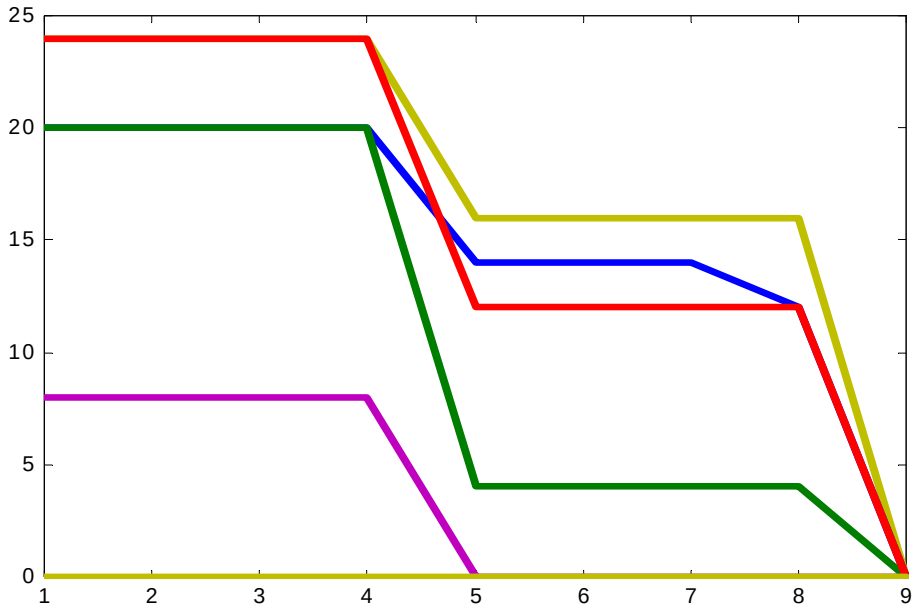


Fig. 4.6 The error function during elapsed time factor

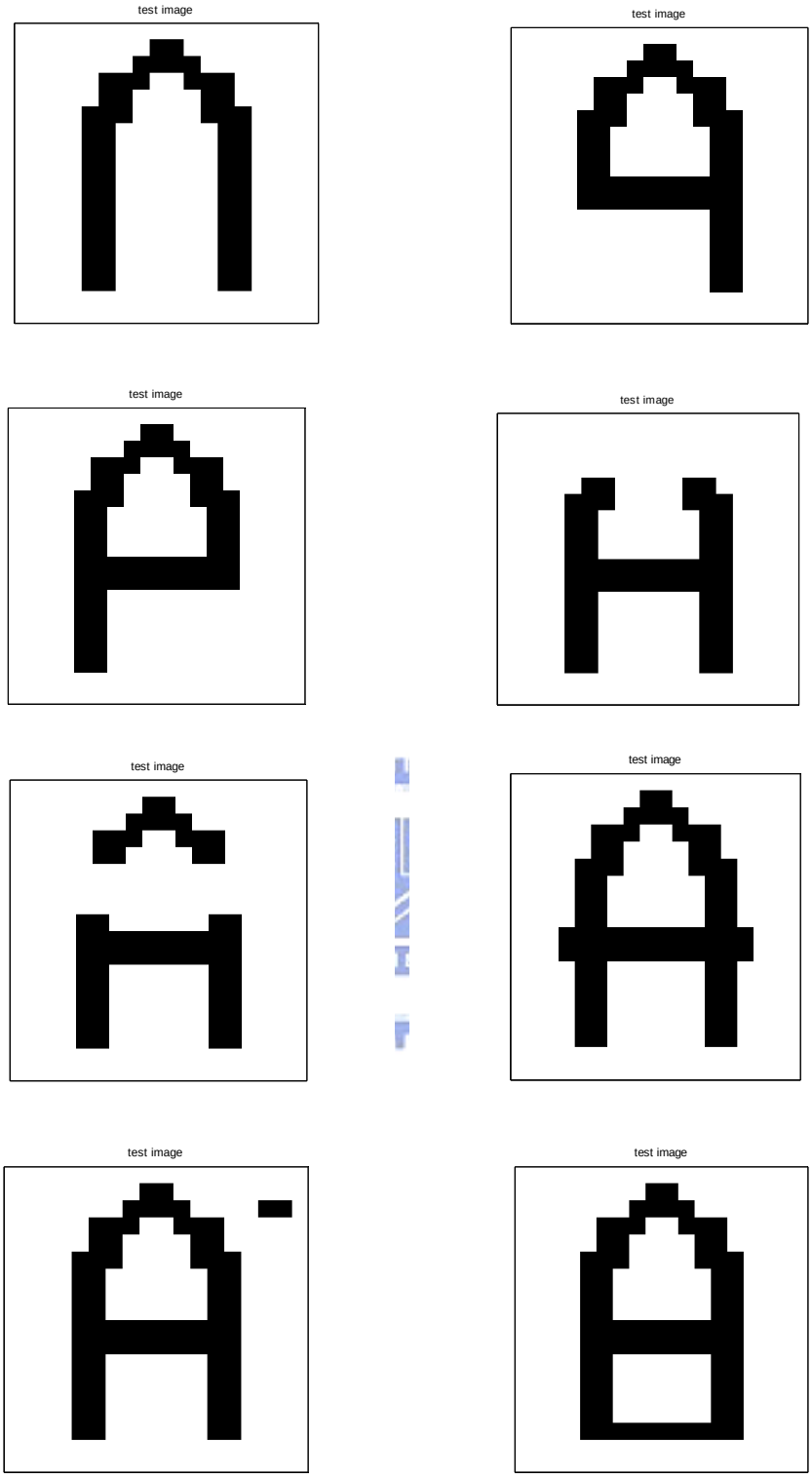
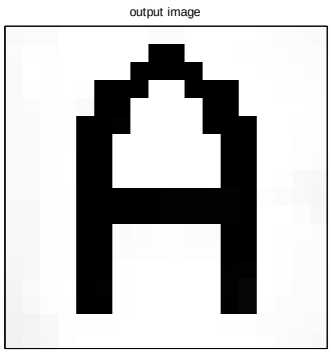
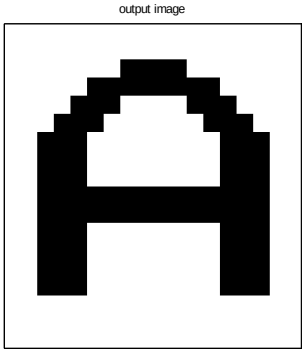


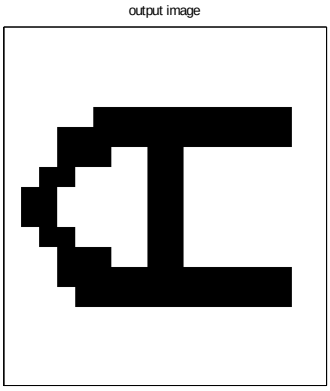
Fig. 4.7 The test patterns of character A with white-black noise



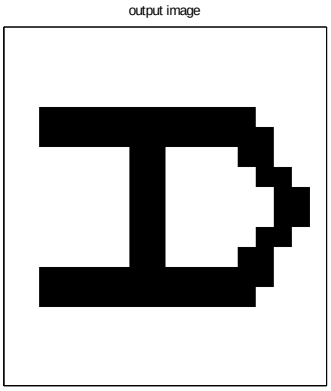
Normal A character



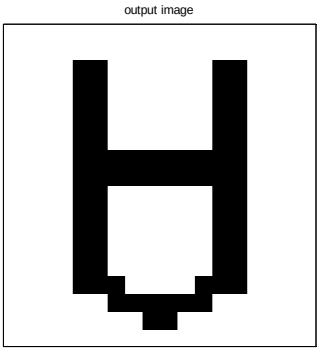
Expanded A character



Left-rotate A character



Right-rotate A character



Reversal A character

Fig. 4.8 The test patterns of character A with normal, expanded, left-rotate, right-rotate, and reversal types

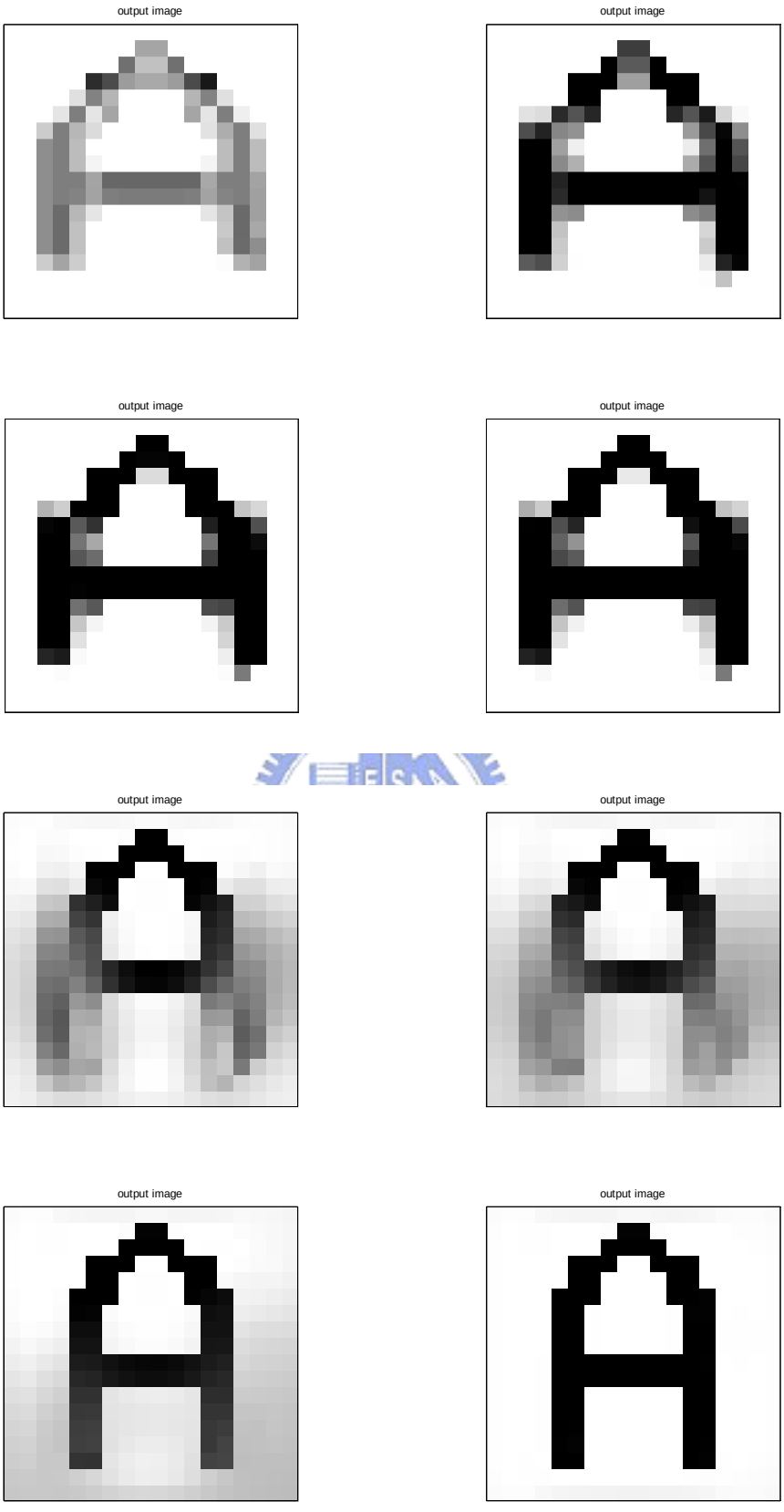


Fig. 4.9 The recognition sequence of expanded A character during recognition period

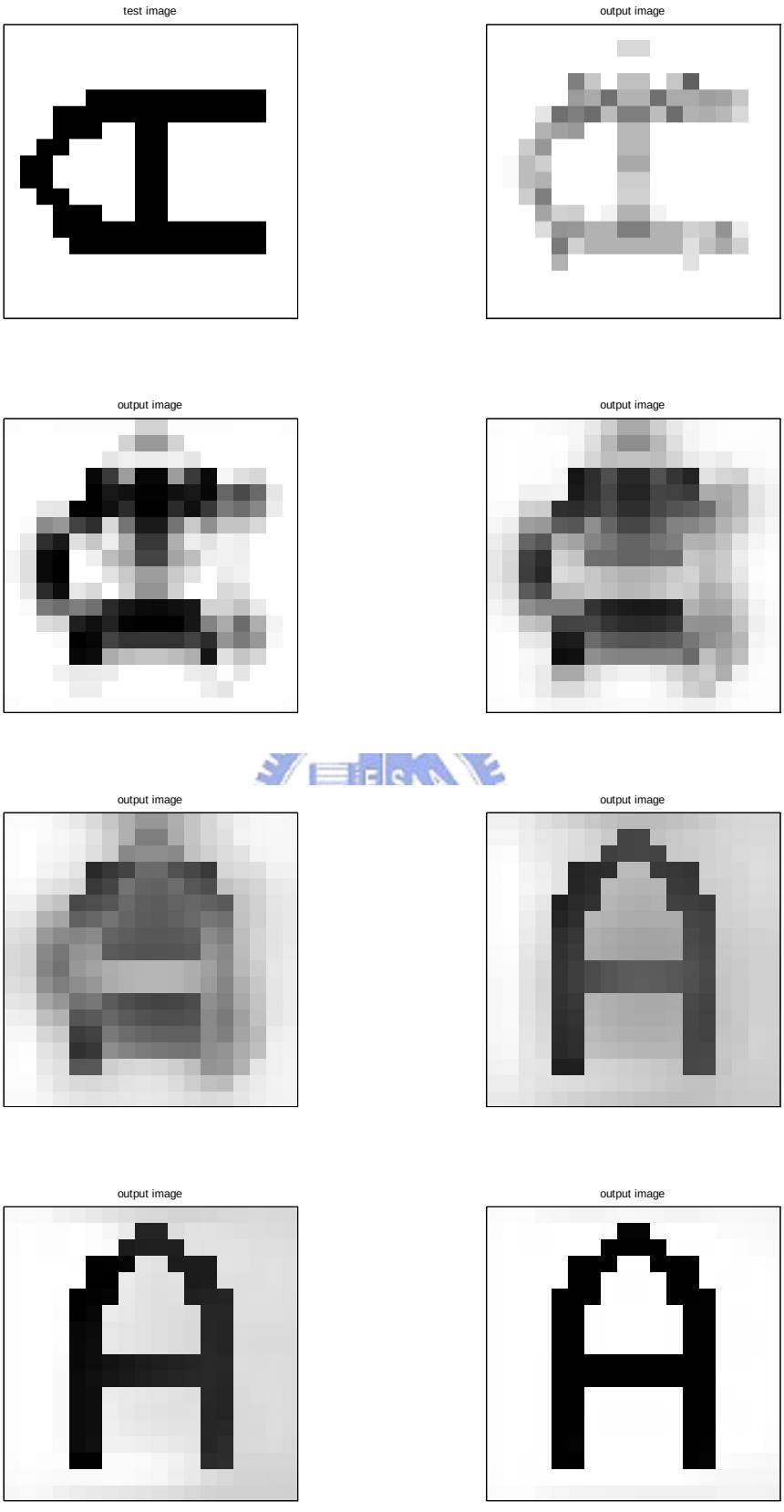


Fig. 4.10 The recognition sequence of left-rotate A character during recognition period