CELLULAR NEURAL NETWORKS

by

Leon O. Chua and Lin Yang

Memorandum No. UCB/ERL M87/49

8 July 1987

CELLULAR NEURAL NETWORKS

by

Leon O. Chua and Lin Yang

Memorandum No. UCB/ERL M87/49

8 July 1987

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

CELLULAR NEURAL NETWORKS [†]

Leon O. Chua and Lin Yang [††]

ABSTRACT

A novel class of information-processing systems called *cellular neural networks* is proposed. Like *neural network,* it is a large-scale nonlinear analog circuit which processes signals in *real* time. Like *cellular automata,* it is made of a massive aggregate of regularly spaced circuit clones, called *cells,* which communicate with each other directly only through its nearest neighbors. Each cell is made of a *linear* capacitor, a *nonlinear* voltage-controlled current source, and a few *resistive* linear circuit elements.

Cellular neural networks shares the best features of both worlds; its continuous time feature allows *real-time* signal processing found wanting in the digital domain and its local interconnection feature makes it tailor made for VLSI implementation.

Although still in its embryonic stage, some impressive applications in such area as *image processing* will be demonstrated, albeit with only a crude circuit. In particular, examples of cellular neural networks which can be designed to recognize the key features of Chinese characters will be presented.

July 8, 1987

[†] This research is supported in part by the office of Naval Research under contract N000-14-86k-0351 and by the National Science Foundation grant No. MIP8614000.

[††] L. O. Chua and L. Yang are with the University of California, Berkeley, Department of Electrical Engineering and Computer Sciences, Berkeley, CA 94720.

1. INTRODUCTION

Analog circuits have had a very important role in the development of the modern technology. Even in our digital computer era, analog circuits still dominate such fields as communication, power, automatic control, audio and video electronics because of their *real-time* signal processing capabilities.

Conventional digital computation methods have run into a serious bottleneck problem due to its serial character. To overcome this problem, a new computation model, called "neural networks", has been proposed, which is based on some aspects of neurobiology and adapted to integrated circuits[1,2,3]. The key features of neural networks are asynchronous parallel processing, continuous-time dynamics, and global interaction of network elements. Some encouraging if not impressive applications of neural networks have been proposed for various fields such as optimization, linear and nonlinear programming, associative memory, pattern recognition and computer vision[4,5,6,7,8,9].

In this paper, we will present a new circuit model, called a *cellular neural network,* which shares some key features of neural networks and which has important potential applications in such area as image processing. In *Section 2,* we will define our cellular neural network model and discuss some related circuit problems, such as stability, dynamic range etc., with the help of circuit and system theory[10]. In *Section 3,* we will provide some examples of its applications in image processing, especially in the recognition of Chinese characters. In *Section 4,* we will discuss the mathematical foundation of cellular neural networks and its relationship with two other well-known mathematical models; namely, partial differential equation and cellular automata, We will also formulate the generalized mathematical equations governing *multi-layered* cellular neural networks.

2. CELLULAR NEURAL NETWORK MODEL AND SOME RELATED THEORETICAL PROBLEMS

2.1. Model of cellular neural networks

The basic circuit unit of cellular neural networks is called a *cell*. It is made of linear and nonlinear circuit elements, which typically are linear capacitors, linear and nonlinear resistors, linear and nonlinear controlled sources, and independent sources. The structure of cellular neural networks looks like that found in cellular automata; namely, any *cell* in a cellular neural network is connected only to its neighbor *cells*. The connected *cells* can interact directly and the far away *cells* may affect each other indirectly because of the propagation effect of the *continuous* time dynamics of cellular neural networks. An example of a two-dimensional cellular neural network is shown in Fig. 2.1. Theoretically, we can define a cellular neural network of any dimension, but in this paper, we will concentrate on the two-dimensional case because here, we will focus our attention on image processing problems.

Consider an $M \times N$ cellular neural network, having $M \times N$ *cells* arranged in M rows and N columns. We call the *cell* on the ith row and the jth column as *cell* (i, j), and denote it as $C(i, j)$ as in Fig. 2.1. Now let us define what we mean by a neighborhood of $C(i, j)$.

Definition 2.1

The neighborhood of a *cell*, $C(i, j)$, in a cellular neural network is defined as

$$N(i, j) = \left\{ C(k, l) \;\middle|\; \max\left\{ |k - i|, |l - j| \right\} \le r, \; 1 \le k \le M, \; 1 \le l \le N \right\} \quad (2.1)$$

where r is a positive integer number.

Figure 2.2 shows 3 neighborhoods of the same *cell* (located at the center) with $r = 1$, 2 and 3, respectively. Usually, we call the $r = 1$ neighborhood as a 3×3 neighborhood, the $r = 2$ neighborhood as a 5×5 neighborhood, and the $r = 3$ neighborhood as a 7×7 neighborhood. It is easy to show that the neighborhood system defined above has the *symmetry* property in the sense that if $C(i, j) \in N(k, l)$, then $C(k, l) \in N(i, j)$, for all $C(i, j)$ and $C(k, l)$ in a cellular neural network.

A typical example of a *cell* of a cellular neural network is shown in Fig. 2.3, where the first suffix u, x, and y denotes the *input, state,* and *output* respectively. The node voltage v_{xij} of $C(i, j)$ is defined as the state of the *cell* whose *initial condition* is assumed to have a magnitude less than or equal to 1; the node voltage v_{uij} is defined as the input of the *cell* and is assumed to have a magnitude less than or equal to 1 for all time t; and the node voltage v_{yij} is defined as the output. C is a linear capacitor; R_x and R_y are linear resistors; I is an independent voltage source; $I_{xu}(i,j;k,l)$ and $I_{xy}(i,j;k,l)$ are linear voltage controlled current sources with the characteristics $I_{xy}(i,j;k,l) = A(i,j;k,l)v_{ykl}$ and $I_{xu}(i,j;k,l) = B(i,j;k,l)v_{ukl}$ for all $C(k, l) \in N(i, j)$; $I_{yx} = \frac{1}{R_y} f(v_{xij})$ is a piecewise-linear voltage-controlled current source with its characteristic $f(\cdot)$ as shown in Fig. 2.4; and E_{ij} is an independent voltage source. All the linear and piecewise-linear controlled sources used in our cellular neural networks can be easily realized by operational amplifiers (op amp)[10,11].

The circuit equations of a *cell* which satisfies KCL and KVL are easily derived as follow :

state equation :

$$C\frac{d\,v_{xij}(t)}{d\,t} = -\frac{1}{R_x}v_{xij}(t) + \sum_{C(k,l)\in N(i,j)} A(i,j;k,l)v_{ykl}(t)$$

$$+ \sum_{C(k,l)\in N(i,j)} B(i,j;k,l)v_{ukl} + I \quad 1\le i\le M,\ 1\le j\le N, \qquad (2.2a)$$

output equation :

$$v_{yij}(t) = \frac{1}{2}\left(|\,v_{xij}(t)+1\,| - |\,v_{xij}(t)-1\,|\right) \quad 1\le i\le M,\ 1\le j\le N, \qquad (2.2b)$$

constraint conditions :

$$|v_{xij}(0)| \le 1 \quad 1\le i\le M,\ 1\le j\le N, \qquad (2.2c)$$

$$|v_{uij}| \le 1 \quad 1\le i\le M,\ 1\le j\le N. \qquad (2.2d)$$

Remarks :

(a) All the *inner cells* of a cellular neural network have the same circuit structures and the same circuit elements with the same values. The *inner cell* is the cell which has $(2r + 1)^2$ neighbor cells, where r is defined in (2.1). All the other cells are called *boundary cells*. A cellular neural network is completely characterized by the set of all nonlinear differential equations associated the *cells* in the circuit.

(b) All the cells of a cellular neural network have at most three nodes. (Some times we will choose $E_{ij} = 0$ if $B(i,j;k;l) = 0$ for all the cells in a cellular neural network. In this case, there are only two nodes in a *cell* circuit.) Since all the *cells* have the same datum node, and since all circuit elements are voltage controlled, our cellular neural networks are ideally suited for the nodal analysis method. Moreover, since the interconnections are *local,* the associated node equation is extremely sparse for large circuits.

(c) The dynamics of a cellular neural network has both feedback and feed-forward mechanisms. The feedback effect depends on the interactive parameter $A(i,j;k,l)$ and the feed-forward effect depends on $B(i,j;k,l)$. Consequently, we will sometimes refer to $A(i,j;k,l)$ as a *feedback operator* and $B(i,j;k,l)$ as a *feed-forward operator.*

(d) The values of the circuit elements can be chosen conveniently in practice. R_x and R_y determine the power dissipated in the circuits and are usually chosen to be $10^3 \sim 10^6\ \Omega$. CR_x is the time constant of the dynamics of the circuit and is usually chosen to be $10^{-11} \sim 10^{-8}$ seconds.

2.2. The dynamical range of cellular neural networks

Before we design a physical cellular neural network, we have to know its dynamical range in order to guarantee that it will satisfy our assumptions on the dynamical equations stipulated in the preceding section. The following theorem provides therefore the foundation for our design.

Theorem 2.1

All the states of the *cells* in cellular neural networks are bounded and the bound v_{max} can be computed by the following formula for *any* cellular neural network :

$$v_{max} = 1 + R_x |I| + R_x \max_{1\le i\le M, 1\le j\le N} \left[\sum_{C(k,l)\in N(i,j)} \left(| A(i,j;k,l) | + | B(i,j;k,l) | \right) \right]. \tag{2.3}$$

Proof:

First, we rewrite the *cell* dynamical equation (2.2) as

$$\frac{d\, v_{xij}(t)}{d\, t} = -\frac{1}{R_x C} v_{xij}(t) + f_{ij}(t) + g_{ij}(u) + \hat{I}$$

$$\text{for } 1\le i\le M,\ 1\le j\le N, \tag{2.4a}$$

where

$$f_{ij}(t) = \frac{1}{C} \sum_{C(k,l)\in N(i,j)} A(i,j;k,l) v_{ykl}(t) \quad \text{for } 1\le i\le M,\ 1\le j\le N, \tag{2.4b}$$

$$g_{ij}(u) = \frac{1}{C} \sum_{C(k,l)\in N(i,j)} B(i,j;k,l) v_{ukl} \quad \text{for } 1\le i\le M,\ 1\le j\le N, \tag{2.4c}$$

and

$$\hat{I} = \frac{I}{C}. \tag{2.4d}$$

Equation (2.4a) is a first-order ordinary differential equation and its solution is given by

$$v_{xij}(t) = v_{xij}(0) e^{\frac{-t}{R_x C}} + \int_0^t e^{\frac{-(t-\tau)}{R_x C}} \left[f_{ij}(\tau) + g_{ij}(u) + \hat{I} \right] d\tau. \tag{2.5}$$

It follows that

$$|v_{xij}(t)| \le |v_{xij}(0)e^{\frac{-t}{R_xC}}| + |\int_0^t e^{\frac{-(t-\tau)}{R_xC}} \left[f_{ij}(\tau) + g_{ij}(u) + \hat{I} \right] d\tau|$$

$$\le |v_{xij}(0)|e^{\frac{-t}{R_xC}} + \int_0^t e^{\frac{-(t-\tau)}{R_xC}} \left[|f_{ij}(\tau)| + |g_{ij}(u)| + |\hat{I}| \right] d\tau$$

$$\le |v_{xij}(0)|e^{\frac{-t}{R_xC}} + \left[F_{ij} + G_{ij} + |\hat{I}| \right] \int_0^t e^{\frac{-(t-\tau)}{R_xC}} d\tau$$

$$\le |v_{xij}(0)| + R_xC \left[F_{ij} + G_{ij} + |\hat{I}| \right], \tag{2.6}$$

where

$$F_{ij} \equiv \max_t |f_{ij}(t)| \le \frac{1}{C} \sum_{C(k,l)\in N(i,j)} |A(i,j;k,l)| \, |v_{ykl}(t)| \tag{2.7a}$$

and

$$G_{ij} \equiv \max_u |g_{ij}(u)| \le \frac{1}{C} \sum_{C(k,l)\in N(i,j)} |B(i,j;k,l)| \, |v_{ukl}|, \tag{2.7b}$$

Since $|v_{xij}(t)|$ and $|v_{uij}|$ satisfy the conditions in equations (2.2c) and (2.2d), and since $|v_{yij}(t)|$ satisfies the condition

$$|v_{yij}(t)| \le 1 \qquad \text{for } all \ \ t \tag{2.8}$$

in view of its characteristic function (2.2b).

It follows from (2.6) and (2.7) that

$$|v_{xij}(t)| \le |v_{xij}(0)| + R_x \left[\sum_{C(k,l)\in N(i,j)} \left(|A(i,j;k,l)| \, |v_{ykl}(t)| \right. \right.$$

$$\left. \left. + |B(i,j;k,l)| \, |v_{ukl}| \right) + |I| \right]$$

$$\le 1 + R_x \left[\sum_{C(k,l)\in N(i,j)} \left(|A(i,j;k,l)| + |B(i,j;k,l)| \right) + |I| \right]$$

$$\text{for} \quad 1 \le i \le M, \ 1 \le j \le N. \tag{2.9}$$

Now let

$$v_{\max} = \max_{(i,j)} \Big\{ 1 + R_x |I| + R_x \sum_{C(k,l)\in N(i,j)} \Big[|A(i,j;k,l)| + |B(i,j;k,l)| \Big] \Big\}, \tag{2.10}$$

then since $v_{\max}$ is independent of the time t and the *cell* $C(i, j)$ for all i and j, we have

$$\max_t |v_{xij}| \le v_{\max} \qquad \text{for } \textit{all} \quad 1 \le i \le M,\ 1 \le j \le N. \tag{2.11}$$

For any cellular neural network, the parameters R_x, C, I, $A(i,j;k,l)$ and $B(i,j;k,l)$ are finite constants, therefore the bound of the states of the *cells*, $v_{\max}$, is finite and can be computed via formula (2.3). □

Remark :

In actual circuit design, we always choose the scale of the circuit parameters such that $R_x |I| = 1$, $R_x |A(i,j;k,l)| = 1$ and $R_x |B(i,j;k,l)| = 1$, for all i, j, k and l. Hence, we can easily estimate the upper bound of the dynamic range of our cellular neural networks. For example, if a neighborhood of the cellular neural network is 3×3, then we can have $v_{\max} = 20\ V$, which is within the ball park for IC circuits.

2.3. The steady states of cellular neural networks

One of the applications of cellular neural networks is image processing, which will be discussed in the later sections. The basic function of a cellular neural network for image processing is to map or transform an input image into another image; namely, the output image. Here, we restrict our output images to binary images with -1 and 1 as the pixel values. However, the input images can have multiple gray levels, provided their corresponding voltages satisfy (2.2d). This means that our image processing cellular neural network must always converge to a constant steady state after

any transient initialized and/or driven by a given input image. How can we guarantee the above convergence requirement of cellular neural networks and what are the conditions or restrictions for such convergence to be possible? In this section, we will discuss the convergence property and its related problems for cellular neural networks.

For dynamic nonlinear circuits, one of the most effective approaches to study their convergent behaviors is Lyapunov's method. Hence, let us first define a Lyapunov function for cellular neural networks.

Definition 2.2

We define the Lyapunov function, $E(t)$, of a cellular neural network by the scalar function

$$E(t) = -\frac{1}{2}\sum_{(i,j)(k,l)}\sum A(i,j;k,l)v_{yij}(t)v_{ykl}(t) + \frac{1}{2R_x}\sum_{(i,j)} v_{yij}(t)^2$$

$$- \sum_{(i,j)(k,l)}\sum B(i,j;k,l)v_{yij}(t)v_{ukl} - \sum_{(i,j)} I v_{yij}(t). \qquad (2.12)$$

Remark :

Observe that the above Lyapunov function, $E(t)$, is a function of only the *output* voltages v_{yij} and the input v_{yij} of the circuit. Although it does not possess the complete information contained in the state variables v_{xij}, we can nevertheless derive the steady state properties of the state variables from the properties of $E(t)$.

The Lyapunov function, $E(t)$, defined above, can be interpreted as the "generalized energy" of a cellular neural network. In the following theorem, we will prove that $E(t)$ is bounded.

Theorem 2.2

The function $E(t)$ defined in (2.12) is bounded by

$$\max_t |E(t)| \leq E_{\max} \tag{2.13a}$$

where

$$E_{\max} = \frac{1}{2} \sum_{(i,j)(k,l)} \sum |A(i,j;k,l)| + \sum_{(i,j)(k,l)} \sum |B(i,j;k,l)|$$

$$+ MN\left(\frac{1}{2R_x} + |I|\right), \tag{2.13b}$$

for an $M \times N$ cellular neural network.

Proof :

From the definition of $E(t)$, we have

$$|E(t)| \leq \frac{1}{2} \sum_{(i,j)(k,l)} \sum |A(i,j;k,l)|\,|v_{yij}(t)|\,|v_{ykl}(t)| + \frac{1}{2R_x} \sum_{(i,j)} v_{yij}^2$$

$$+ \sum_{(i,j)(k,l)} \sum |B(i,j;k,l)|\,|v_{yij}(t)|\,|v_{ukl}| + \sum_{(i,j)} |I|\,|v_{yij}|. \tag{2.14}$$

Since $v_{yij}(t)$ and v_{uij} are bounded as stipulated in (2.2 c,d), we have

$$|E(t)| \leq \frac{1}{2} \sum_{(i,j)(k,l)} \sum |A(i,j;k,l)| + MN\frac{1}{2R_x}$$

$$+ \sum_{(i,j)(k,l)} \sum |B(i,j;k,l)| + MN\,|I|. \tag{2.15}$$

It follows from (2.13b) and (2.15) that $E(t)$ is bounded via (2.13a). □

In addition to the above bounded property of the function $E(t)$, we can prove another important property.

Theorem 2.3

The scalar function $E(t)$ defined in (2.12) is a monotone-decreasing function, that is

$$\frac{d\,E(t)}{d\,t} \leq 0. \tag{2.16}$$

Proof :

To differentiate $E(t)$ in (2.12) with respect to time t, take the derivative of $v_{yij}(t)$ on the right side of (2.12) with respect to $v_{xij}(t)$, and then differentiate $v_{xij}(t)$ with respect to time t :

$$\begin{aligned}\frac{d\,E(t)}{d\,t} = & -\sum_{(i,j)}\sum_{(k,l)} A(i,j;k,l)\frac{d\,v_{yij}}{d\,v_{xij}}\frac{d\,v_{xij}(t)}{d\,t}v_{ykl}(t) \\ & + \frac{1}{R_x}\sum_{(i,j)}\frac{d\,v_{yij}}{d\,v_{xij}}\frac{d\,v_{xij}(t)}{d\,t}v_{yij}(t) \\ & - \sum_{(i,j)}\sum_{(k,l)} B(i,j;k,l)\frac{d\,v_{yij}}{d\,v_{xij}}\frac{d\,v_{xij}(t)}{d\,t}v_{ukl} \\ & - \sum_{(i,j)} I\frac{d\,v_{yij}}{d\,v_{xij}}\frac{d\,v_{xij}(t)}{d\,t}\end{aligned} \tag{2.17}$$

From the output functions in (2.2b), we obtain the following relations

$$\frac{d\,v_{yij}}{d\,v_{xij}} = \begin{cases} 1 & |v_{xij}| < 1 \\ 0 & |v_{xij}| \geq 1 \end{cases} \tag{2.18a}$$

and

$$v_{xij} = v_{yij} \qquad |v_{xij}| < 1\,. \tag{2.18b}$$

(Here, we define $\frac{d\ v_{yij}}{d\ v_{xij}} = 0$, for $|v_{xij}| = 1$.) And according to our definition of cellular neural networks, we have

$$A(i,j;k,l) = 0,\ \ B(i,j;k,l) = 0 \qquad \text{for } C(k,l) \in N(i,j). \tag{2.18c}$$

It follows from (2.17) and (2.18) that

$$\begin{aligned}\frac{d\ E(t)}{d\ t} &= -\sum_{(i,j)} \frac{d\ v_{yij}}{d\ v_{xij}} \frac{d\ v_{xij}(t)}{d\ t} \left[\sum_{C(k,l)\in N(i,j)} A(i,j;k,l) v_{ykl}(t) \right. \\ &\quad \left. - \frac{1}{R_x} v_{xij}(t) + \sum_{C(k,l)\in N(i,j)} B(i,j;k,l) v_{ukl} + I \right] \\ &= -\sum_{|v_{xij}|<1} \frac{d\ v_{xij}(t)}{d\ t} \left[\sum_{C(k,l)\in N(i,j)} A(i,j;k,l) v_{ykl}(t) \right. \\ &\quad \left. - \frac{1}{R_x} v_{xij}(t) + \sum_{C(k,l)\in N(i,j)} B(i,j;k,l) v_{ukl} + I \right]\end{aligned} \tag{2.19}$$

Substituting the *cell* circuit equation (2.2) into (2.19), and assuming $C > 0$, we obtain

$$\begin{aligned}\frac{d\ E(t)}{d\ t} &= -\sum_{|v_{xij}|<1} C \left[\frac{d\ v_{xij}(t)}{d\ t} \right]^2 \\ &= -\sum_{|v_{yij}|<1} C \left[\frac{d\ v_{yij}(t)}{d\ t} \right]^2 \\ &= -\sum_{(i,j)} C \left[\frac{d\ v_{yij}(t)}{d\ t} \right]^2 \\ &\le 0. \qquad \square\end{aligned} \tag{2.20}$$

From Theorems 2.2 and 2.3, we can easily prove the following important result :

Theorem 2.4

For any given input $\mathbf{v}_u$ and any initial state $\mathbf{v}_x$ of a cellular neural network, we have

$$\lim_{t\to\infty} E(t) = constant \tag{2.21a}$$

and

$$\lim_{t\to\infty} \frac{d\ E(t)}{d\ t} = 0. \tag{2.21b}$$

proof :

From theorem 2.2 and Theorem 2.3, $E(t)$ is a bounded monotone-decreasing function of time t. Hence $E(t)$ must converge to a limit and its derivative must converge to 0. □

Corollary

After the transient of a cellular neural network has decayed to zero, we always get constant dc outputs. In other words, we have

$$\lim_{t\to\infty} v_{yij}(t) = constant \qquad 1\le i\le M,\ 1\le j\le N. \tag{2.21c}$$

Let us investigate next the steady state behavior of cellular neural networks. It follows from the proof of Theorem 2.3 that under the condition $\frac{d\ E(t)}{d\ t} = 0$, there are three cases for the state of a *cell* :

$$(1) \qquad \frac{d\ v_{xij}(t)}{d\ t} = 0 \quad \text{and} \quad |v_{xij}| < 1; \tag{2.22a}$$

$$(2) \qquad \frac{d\ v_{xij}(t)}{d\ t} = 0 \quad \text{and} \quad |v_{xij}| > 1; \tag{2.22b}$$

$$(3) \qquad \frac{d\ v_{xij}(t)}{d\ t} \neq 0 \quad \text{and} \quad |v_{xij}| > 1; \tag{2.22c}$$

Since Theorem 2.4 says nothing when $|v_{xij}(t)| > 1$, is it possible then for all three cases to exist when a cellular neural network is in its steady state, or can only one or two of them exist? We claim that only case (2) can exist in the *steady state*.

To prove this claim, let us rewrite cell equation (2.2) as follow :

$$C \frac{d\, v_{xij}(t)}{d\, t} = -f(\, v_{xij}(t)\,) + g(t) \tag{2.23a}$$

where

$$f(\, v_{xij}(t)\,) = -0.5A(i,j;i,j)\,(|\, v_{xij}(t)+1\,| - |\, v_{xij}(t)-1\,|) + \frac{1}{R_x} v_{xij}(t) \tag{2.23b}$$

and

$$g(t) = \sum_{C(k,l)\in N(i,j) \text{and} \neq C(i,j)} \left[A(i,j;k,l) v_{ykl}(t) + B(i,j;k,l) v_{ukl} \right] + I. \tag{2.23c}$$

Let us first make some restrictions on the function $f(\cdot)$ in (2.23b). Suppose $A(i,j;i,j) > \frac{1}{R}$; for convenience and without loss of generality, let $A(i,j;i,j) = 2$ and $R_x = 1$ in the following analysis. Then $f(v_{xij})$ has the characteristic shown in Fig. 2.5.

Consider next the equivalent circuit of a *cell* in a cellular neural network as shown in Fig. 2.6. There are only three circuit elements, a linear capacitor with a positive capacitance C; a piecewise-linear voltage controlled resistor with its driving-point characteristic $i = f(v)$ ($f(\cdot)$ is the same function as in Fig. 2.5); and a time-varying independent current source having a time-dependent function given by $g(t)$. The two circuits in Fig. 2.3 and Fig. 2.6 are equivalent because they are both described by (2.23a), which we rewrite for simplicity as follow :

$$C \frac{d\, v(t)}{d\, t} = -f(\, v(t)\,) + g(t). \tag{2.24}$$

For $g(t) = 0$, the *equilibrium points* and the *dynamic route* [10] of the equivalent circuit are shown in Fig. 2.7a. There are three *equilibrium points* in this circuit, one of them, $v = 0$, denoted by a circle is *unstable;* the other two, $v = -2$ and $v = 2$, are *stable,* and are denoted by solid points. The unstable equilibrium point is never

observed in physical electronic circuits, because of the unavoidable thermal noise. So, after the transient, and depending on the initial state, the circuit will always approach one of its *stable equilibrium points* and stay there forever. For example, if the initial state of the circuit is $v = 0.5$, then the steady state will be at the *stable equilibrium point* $v = 2$; but if the initial state of the circuit is $v = -0.5$, then the steady state will be at the *stable equilibrium point* $v = -2$.

If $g(t) = constant \neq 0$, there are six different cases of the dynamic behavior of the equivalent circuit as shown in Fig. 2.7b-g. For the cases in Fig. 2.7 (b) and (c), there are also three *equilibrium points;* one of them is *unstable,* while the other two are *stable.* For the cases in Fig. 2.7 (d) and (e), there are two *equilibrium points;* one is *unstable* and the other is *stable.* For the dynamic route in Fig. 2.7f and Fig. 2.7g, there is only one *equilibrium point* for the circuit, and it is *stable.* It is very important to notice that all the *stable equilibrium points* for the seven *dynamic route* cases of the equivalent circuit of a *cell* in cellular neural networks share the common property $|v| > 1$.

Let us return now to the basic *cell* circuit of our cellular neural networks. Since $g(t)$ is a function of only the outputs, $v_{ykl}(t)$, and the inputs, v_{ukl}, of the neighborhood of the *cell,* it follows from the results of Theorem 2.4 that all the outputs of our cellular neural network are constants. Hence, after the initial transients our assumption $g(t) = constant$ is valid for the study of the steady state behavior of cellular neural networks. Let us summarize our above observations as follow :

Theorem 2.5

Assuming the circuit parameters satisfy

$$A(i,j;i,j) > \frac{1}{R}, \tag{2.25}$$

then each *cell* of our cellular neural network must settle at a *stable equilibrium point* after the *transient* has decayed to zero. Moreover, the magnitude of all *stable equilibrium points* is greater than 1. In other words, we have the following properties

$$\lim_{t\to\infty} |v_{xij}(t)| > 1 \qquad 1\le i\le M,\ 1\le j\le N, \tag{2.26a}$$

and

$$\lim_{t\to\infty} v_{yij}(t) = \pm 1 \qquad 1\le i\le M,\ 1\le j\le N. \tag{2.26b}$$

Remarks :

(a) The above theorem is significant for cellular neural networks, because it implies that the circuit will not oscillate or become chaotic.

(b) Theorem 2.5 guarantees our cellular neural networks have binary-value outputs. This property is very important for the *classification* problems in image processing applications.

3. APPLICATIONS OF CELLULAR NEURAL NETWORKS IN IMAGE PROCESSING

In the following, we will use the cellular neural network model analyzed in the previous sections to solve some image processing problems. So far we have stressed only the steady-state behavior of cellular neural networks. However, for applications in image processing, the transient behavior is equally important. In fact, it is the transient behavior which makes it possible to extract different features from the same picture or to deal with various image processing problems. The role played by the cellular neural network's transient behavior will be cleared from the following examples.

3.1. A simple example

Before we consider the real image processing problems, it is instructive to look at a very simple example. Although it is a much simplified image processing problem, this example will help us understand some of the dynamic behaviors of cellular neural networks and derive some intuitive ideas on how to design cellular neural networks for a specific practical image processing problem.

One of the important problems in image processing is *pixel classification* [12]. To illustrate this concept, consider a small image as shown in Fig. 3.1a. This image is a 4×4 pixel array with each pixel value $P_{ij} \in [-1, 1]$, for $1 \le i \le 4$ and $1 \le j \le 4$. Suppose that the pixel value, -1, corresponds to the brightest gray level or the white background, and the pixel value, 1, corresponds to the darkest gray level or the black object point value. The *pixel classification* problem is to classify each pixel of an image into two or more classes.

From the mathematical point of view, *pixel classification* can be considered as a map, F, which maps a continuous vector space to a discrete vector space as defined below

$$F : \quad [a, b]^{M\times N} \rightarrow \left\{ A, B, C, \cdots \right\}^{M\times N} \tag{3.1}$$

where $M \times N$ is the number of pixels in the image. To use our cellular neural network for *pixel classification,* assume the output images belong to the two-class case. For this simple example, we wish to assign to each pixel in the array one of the two values, -1 and 1, based on some classification rules. For this case, the pixel classification map, F, is

$$F : \quad [-1.0, 1.0]^{M\times N} \rightarrow \left\{ -1, 1 \right\}^{M\times N} . \tag{3.2}$$

Suppose that we want to design a horizontal line detector by using a cellular neural network to check whether or not there are horizontal lines in the input image in Fig. 3.1a. To simplify our analysis, we have chosen a very simple dynamic rule for this "horizontal line detector" circuit. We chose the circuit element parameters of the *cell* $C(i,j)$ as follows:

$$C = 10^{-9}F \; ; \quad R_x = 10^3\Omega \; ; \quad I = 0 \; ;$$

$$A(i,j;i-1,j-1) = A(i,j;i-1,j) = A(i,j;i-1,j+1) = 0 \; ;$$

$$A(i,j;i,j) = 2\times10^{-3}\Omega^{-1}; \quad A(i,j;i,j-1) = A(i,j;i,j+1) = 10^{-3}\Omega^{-1} \; ;$$

$$A(i,j;i+1,j-1) = A(i,j;i+1,j) = A(i,j;i+1,j+1) = 0;$$

for the 3×3 neighborhood system. Since the interactive rules, which is determined by the feedback operator $A(i,j;k,l)$ shown above, are independent of the absolute position of a *cell*, that is each *cell* has the same interactive relation with its neighbors as the other *cells* in the same cellular neural network, we can simplify the expressions of the parameters $A(i,j;k,l)$ for the *cell* $C(i,j)$ by coding them as follow :

$$A(-1,-1) = A(-1,0) = A(-1,1) = 0 ;$$

$$A(0,0) = 2\times10^{-3}\Omega;\ A(0,-1) = A(0,1) = 10^{-3}\Omega ;$$

$$A(1,-1) = A(1,0) = A(1,1) = 0.$$

The indices in the above interactive parameters indicate the relative positions with respect to $C(i,j)$. Therefore we can express the above interactive voltage-controlled current source parameters in the neighborhood of a *cell* by a two-dimensional operator as shown in Fig. 3.2, and call it the templet of the interactive operator of the *cell* with its neighbor cells. This templet is constructed as follow : the center entry of the templet corresponds to $A(0,0)$; the upper left corner entry of the templet corresponds to $A(-1,-1)$; the lower right corner entry of the templet corresponds to $A(1,1)$; and so forth. Since this templet expression is a very convenient way to characterize the interactions of a *cell* with its neighbors, we will use the templet expression to express the neighborhood controlled sources in the following examples.

The *cell* dynamical equations of the cellular neural network corresponding to the above parameters are given by :

$$\frac{d\ v_{xij}(t)}{d\ t} = 10^6\left[-v_{xij}(t) + v_{yij-1}(t) + 2v_{yij}(t) + v_{yij+1}(t)\right] \qquad (3.3a)$$

and

$$v_{yij}(t) = 0.5\ (|\ v_{xij}(t) + 1\ |\ -\ |\ v_{xij}(t) - 1\ |) \qquad (3.3b)$$

$$\text{for}\quad 1 \le i \le 4,\ \ 1 \le j \le 4.$$

Note that we have chosen the feed-forward operator $B(i,j;k,l) = 0$ for all i,j,k,l in this circuit.

Suppose that the initial state of the cellular neural network is the pixel array in Fig. 3.1a. From above analysis, we know that the circuit equations in (3.3) are first order nonlinear ordinary differential equations. In system theory, they are also called a piecewise-linear autonomous system. So, the cellular neural network in this example is a piecewise linear autonomous circuit. Now, let us first analyze the steady state of this cellular neural network. The equilibrium points of the circuit can be found by solving the equivalent DC equations

$$v_{xij}(t) = v_{yij-1}(t) + 2v_{yij}(t) + v_{yij+1}(t) \tag{3.4a}$$

$$v_{yij}(t) = 0.5\ (|\ v_{xij}(t) + 1\ |\ -\ |\ v_{xij}(t) - 1\ |), \tag{3.4b}$$

$$1 \le i \le 4, \quad 1 \le j \le 4.$$

In this case, there are 32 unknown variables and 32 equations (16 linear equations and 16 piecewise linear equations). In general, for piecewise linear circuits, we can find all the solutions of the DC circuit equations either by the brute force algorithm [13] or by some other improved efficient algorithms [14, 15]. However, even for this very simple example, it is time consuming to find all of the equilibrium points of the cellular neural network by using the algorithms mentioned above because of the large size of its circuit equations. (Note that if the circuit is not piecewise linear, there is no general method to find all its equilibrium points.)

To simplify our problem, we will take advantage of various features of cellular neural networks in our analysis. As mentioned before, every *cell* in a cellular neural network, has the same connections with its neighbors. Therefore, each *cell* circuit equation is the same as that of the other *cells* in the same circuit. (Here we ignored the difference of the *inner cells* and the *boundary cells* again.) Hence, we can understand the global properties of a cellular neural network by studying the local properties of its *cell*. This approach is a very important and typical method for cellular neural networks analysis and design.

Before analyzing this example, it is helpful to introduce the following definitions.

Definition 3.1

The *local equilibrium states* of a *cell* circuit are the solutions v_{xij} of the DC *cell* circuit equations under the condition that all outputs v_{yij} of its neighbor *cells* are in a stable steady state, that is either -1 or 1;

Definition 3.2

The *local stable equilibrium states* of a *cell* circuit are its *local equilibrium states* with their corresponding outputs equal to ±1.

Definition 3.3

A *global stable equilibrium point* of a cellular neural network is the circuit state vector with all its state components consisting of *local stable equilibrium states.*

Now, let us compute the *local stable equilibrium states* of an *inner cell* of the preceding circuit example. Considering the condition in the above definitions, the *local stable equilibrium states* can be solved by setting (3.3a) to zero and solving for v_{xij} with v_{yij} taking on either the value +1 or -1 :

$$v_{xij} = sign\left[v_{yij-1}\right] + 2sign\left[v_{yij}\right] + sign\left[v_{yij+1}\right] \tag{3.5a}$$

$$|v_{xij}| \geq 1 \tag{3.5b}$$

$$1 \leq i \leq M,\ 1 \leq j \leq N.$$

Substituting v_{yij-1} and v_{yij+1} in the above equations by ±1, we obtain the following four cases:

(a) For $v_{yij-1} = -1$ and $v_{yij+1} = -1$, we have $v_{xij} = -2 + 2sign\left[v_{xij}\right]$, and hence $v_{xij} = -4$.

(b) For $v_{yij-1} = +1$ and $v_{yij+1} = -1$, we have $v_{xij} = 2sign\left[v_{xij}\right]$, and hence $v_{xij} = -2,\ or\ 2$.

(c) For $v_{yij-1} = -1$ and $v_{yij+1} = +1$, we have $v_{xij} = 2sign\left[v_{xij}\right]$, and hence $v_{xij} = -2$, *or* 2.

(d) For $v_{yij-1} = 1$ and $v_{yij+1} = 1$, we have $v_{xij} = 2 + 2sign\left[v_{xij}\right]$, and hence $v_{xij} = 4$.

It follows from the above analysis that the *local stable equilibrium states* of any *inner cell* circuit for our present example are -4, -2, 2 and 4. We can also compute the dynamic range of this circuit by using (2.3). The result is $v_{\max} = 5\ V$.

The *local stable equilibrium state* of each *cell* depends on the *local stable equilibrium states* of its neighbor *cells*. Of course, if the input of the cellular neural network is not zero, then the *local stable equilibrium states* of the *cell* circuit will depend also on the input. So, the *global stable equilibrium states* of a cellular neural network depend on the initial conditions, the inputs and the dynamic rule of the circuit. The properties of the *global stable equilibrium states* can be determined by those of the *local stable equilibrium states*.

From the above analysis, we can see that any input image will be mapped to a specific output image by a cellular neural network. For a given cellular neural network, the output images have some spatial structures resulting from the dynamic rule of the circuit. For instance, it is impossible to have a row like [1, -1, 1, -1],which is a confused pattern, in the output image of the cellular neural network in the above simple example. Hence, an appropriately chosen dynamic rule could imbued a cellular neural network with the ability to recognize and extract some special patterns in the input image. Hence, different input images may be mapped to the same output images if they have the same patterns, and the same input image may be mapped to different output images by different cellular neural networks for different image processing or pattern recognition purposes.

The dynamic behavior of the cellular neural network with zero feed-forward operators and nonzero feedback operators in this example is reminiscent of the two-dimensional *cellular automata*. The theory and application of cellular automata is presently an active research area [16]. N.H.Packard and S.Wolfram, in a recent

paper[17], have provided some basic results for two-dimensional *cellular automata*. A *cellular automata machine*, called CAM-6, has been built at the department of artificial intelligence at MIT [18]. The only difference between a cellular neural network and a cellular automata machine is their dynamic behavior. The former is a *continuous* dynamical system while the latter is a *discrete* dynamical system. Because the two systems have many similarities, we can use the cellular automata theory to study the steady state behaviors of cellular neural networks. Another remarkable distinction between them is that while the cellular neural networks will always reach their equilibrium points, cellular automaton usually has a much richer dynamical behaviors, such as periodic, chaotic and even more complex phenomena. This is because we have chosen a particular nonlinearity for the nonlinear circuit elements in our cellular neural networks. If we choose some other nonlinearity for the nonlinear elements, there will be many complex phenomena in the cellular neural networks. In Section 4, we will compare these two models again.

3.2. Simulation of cellular neural networks

Since cellular neural networks are nonlinear dynamical systems, there are presently no analytical methods for studying their transient behaviors. Consequently, it is necessary to use computer simulation. The circuit simulator we used is PWLSPICE, which is a modified version of SPICE.3 [19] for piecewise-linear circuit analysis developed by You-lin Liao from our laboratory. In this section, we will introduce a *preprocessor* of PWLSPICE, called CELL, which automatically generates the input circuit files for PWLSPICE according to an easily understandable circuit description file and a data file. We will also describe an two *postprocessors*, called BINF and PLOT, which transfer the standard outputs from PWLSPICE into a binary data file and then feeding them into a color graphics terminal for display.

Our computer simulation procedure consists of the following steps :

Step 1: write a cellular neural network description file for CELL. As an example, the circuit description file for the simple example in the preceding section is shown in Fig. 3.3. The first word of each line in the description file is a *key* word: it tells CELL the meaning of the following parameters.

Step 2: write a data file for CELL, which consists of the initial condition and/or the input of the cellular neural network. The data file for the horizontal line detector is shown in Fig. 3.4. Note that, we have quantized the pixel values in this example into 11 discrete levels to simplify the display.

Step 3: after running CELL with the circuit description and data files, we obtain the input file for the circuit simulator PWLSPICE. The input PWLSPICE input file of the horizontal line detector is shown in Table 3.1 in the Appendix.

Step 4: obtain the transient simulation output of PWLSPICE. The transient simulation of the horizontal line detector is shown in Table 3.2 in the Appendix.

Step 5: transfer the PWLSPICE output file to a binary data file by using BINF.

Step 6: display the output image on a color graphics terminal (we use a MASSCOMP terminal) by using PLOT with the binary output data file.

Our simulation result of the simple example in section 3.1 is shown in Fig. 3.1b. Note that row 3 stands out as a black horizontal line and flanked by a white background. Hence, even such a crude horizontal line detector circuit is capable of extracting the horizontal line structures in the given image in Fig. 3.1a. (Of course, this simple dynamic rule cannot handle more sophisticated line detection problems.)

If we change the dynamic rule of our circuit, say, instead of (3.1a), we use

$$\frac{d\ v_{xij}(t)}{d\ t} = 10^6 \left[-v_{xij}(t) + v_{yi-1j}(t) + 2v_{yij}(t) + v_{yi+1j}(t) \right] \qquad (3.3a')$$

then this cellular neural network becomes a simple vertical line detector. Its simulation result for the pixel array in Fig. 3.1a is shown in Fig. 3.1c. Here, all pixels are of uniformly white color. From the simulation results of the above two detector circuits, we know that the pixel array in Fig. 3.1a has horizontal lines but no vertical lines.

3.3. Cellular neural networks for noise removal

Now, let us consider the most important problem in image processing. Since the input pictures usually come from the real world by a camera or some other optical equipment , there are always some noise on the images of the objects. In this paper, we will concentrate on the text processing problems, specifically, on the Chinese character recognition problems. Suppose that the characters in the input images are smeared in some way such as the picture shown in Fig. 3.6a. (We use colors to indicate the different gray levels of the pixels on the image.) For this case, the smeared character in Fig. 3.6a is generated from a perfect binary image by adding a Gaussian white noise with $\sigma = 0.2$, $m = 0$. The size of the image in Fig. 3.6a is 16×16, so the noise removing cellular neural network should have 16×16 *cells*. In image processing, the simplest way to delete noise from the image is to use an *averaging operator*. We choose therefore the averaging operator as the dynamic rule for our "noise removing" cellular neural network. This averaging operator is shown in Fig. 3.5a, and we use it as the feedback operator. Assuming the other circuit parameters are the same as those in the simple example in Section 3.1, the resulting *cell* circuit equation is given by :

$$\frac{d\ v_{xij}(t)}{d\ t} = 10^6 \Big[-v_{xij}(t) + v_{yi-1j}(t) + v_{yij-1}(t) + 2v_{yij}(t) + v_{yij+1}(t) + v_{yi+1j}(t) \Big] \tag{3.6a}$$

and

$$v_{yij}(t) = 0.5\ (|\ v_{xij}(t) + 1\ |\ -\ |\ v_{xij}(t) - 1\ |). \tag{3.6b}$$

Note that the rate change of the state of *cell* $C(i, j)$ is proportional to the average of the outputs of the neighborhood $N(i, j)$. Hence, the steady state of $C(i, j)$ depends on the average of those of its neighbor *cells*.

Fig. 3.6b-d are the outputs of the cellular neural network at time step 10, 20 and 30 respectively. This cellular neural network has the same properties as two-dimensional low-pass filters. It retains the low frequency components while eliminating the high frequency components. Consequently, the corners of the objects in the images suffer the same problem as they do in the two-dimensional low-pass filter cases. The problem is that, at the corners of the objects, there are high frequency

components, and these high frequency components are removed along with the high frequency noise because of the low-pass filter effect. Therefore, the *pixel classification* will not be correct at the corners of object. To see this point, consider the picture in Fig. 3.7a, the only difference between Fig. 3.7a and Fig. 3.6a is that the variance of the Gaussian white noise in Fig. 3.7a is 0.4 ($\sigma = 0.4$). We use the same circuit to process the image in Fig. 3.7a. The results are showed in Fig. 3.7b-d, for the time step 10, 20 and 30. For the corner points in the images, their steady states depend mainly on their initial states. This can be seen from the the dynamic rule we used for the above noise removing cellular neural networks. If we change our *cell* circuit dynamic rule as shown in Fig. 3.5b, that is we add more weight for the *cell* itself, then we have the results as shown in Fig. 3.8 and Fig. 3.9 for $\sigma = 0.2$ and $\sigma = 0.4$ respectively. We can see that the results in Fig. 3.8 and Fig. 3.9 are better than those in Fig. 3.6 and Fig. 3.7. The image in Fig. 3.8d, which is the output image of the cellular neural network with its interactive operator as shown in Fig. 3.5b, is exactly the original undisturbed character.

The above *cell* circuit dynamic rules all involve only four nearest neighbors. If we change the *cell* circuit dynamic rule to one which relate all eight neighbors, such as the averaging operator shown in Fig. 3.5c, then the results will be as shown in Fig. 3.10 for $\sigma = 0.2$, and in Fig. 3.11 for $\sigma = 0.4$.

The above examples all involved images corrupted by a Gaussian white noise. Now, let us consider the non-Gaussian noise case, since the Gaussian noise is usually not a good model for disturbed images. Fig. 3.12 shows a result for a non-Gaussian noise image using a cellular neural network under the dynamic rule shown in Fig. 3.5a.

To see the effects of the interactions of the *cells* in cellular neural networks, Fig. 3.13 gives the results of a non-Gaussian noise image being processed byunder the circuit using the dynamic rule shown in Fig. 3.5d. (Note that, the pixel values of the yellow points in the output image of Fig. 3.13f are zeros. They are the *unstable local equilibrium states*. Unlike those in a physical electronic circuit, the *unstable local equilibrium states* can be obtained by computer simulation.)

From the above results, it can be seen that cellular neural networks are effective for removing noise in image processing, especially for images with large objects and

few corners. Fig. 3.14 and Fig. 3.15 are the simulation results for $\sigma = 0.6$ and $\sigma = 1.0$ respectively by using the operator in Fig. 3.5a.

3.4. Cellular neural networks for feature extraction

Feature extraction is another important problem in image processing. As we have seen in the previous simple example in Section 3.1, the cellular neural network can extract the horizontal lines in the input image in a very simple case. In this section, we will give some other examples of cellular neural networks for *feature extraction* in image processing.

3.4.1. Extract the edges of a diamond

Consider the upper left image shown in Fig. 3.17; namely, the picture of a diamond. What we want here is to extract the edges of this diamond, since they contained most of the information regarding the shape of the diamond. This time we will use another two-dimensional filter, called the *Laplacian operator,* as the *cell* circuit dynamic rule for our cellular neural network. The Laplacian is a well-known operator which is good for edge detections [20,21]. The dynamic rule of the Laplacian operator is shown in Fig. 3.16, and is chosen as the feedback operator. We still use the same parameters C, R_x and $B(i,j;k,l)$ as those in the circuit in Section 3.1. However, we choose $I = -1.75\times10^{-3}\ A$ for our diamond edge-extraction cellular neural network.

The *cell* circuit equation of this cellular neural network is given by :

$$\frac{d\ v_{xij}(t)}{d\ t} = 10^6 \Big[-v_{xij}(t) + -0.5v_{yi-1j}(t) + -0.5v_{yij-1}(t) + 2v_{yij}(t) + -0.5v_{yij+1}(t) + -0.5v_{yi+1j}(t)\ - 1.75\Big] \qquad (3.7a)$$

and

$$v_{yij}(t) = 0.5\ (|\ v_{xij}(t) + 1\ |\ -\ |\ v_{xij}(t) - 1\ |) \qquad (3.7b)$$

$$1 \le i \le 16,\ \ 1 \le j \le 16.$$

The result of our circuit simulation is shown in Fig. 3.17. This result is just what we expected. But if we choose the circuit parameter $I = -1.5\times10^{-3}\ A$ or

$I = -2.0\times10^{-3}\ A$ and keep the other circuit parameters the same as those above, then the results of the circuit simulations are shown in Fig. 3.18 and Fig. 3.19 respectively. From the above circuit simulations we can see the effects of the parameter I for feature extraction in image processing.

3.4.2. Extract corners of a square

If we use the cellular neural network designed in the preceding section with the equation (3.5) to process the upper left image shown in Fig. 3.20, we would obtain the circuit simulation results shown in Fig. 3.20. The output image, or the output of the cellular neural network at the steady state, has not extracted the edges of the square, but its corners in this case. Why is that? Let us take a closer look at the dynamic rule of the cellular neural network circuit. At the initial time of the circuit transient, $t = 0$, there are six relationships for the *cells* in the circuit with their neighbors as shown in Fig. 3.21. (here we have ignored the *cells* on the boundary of the image.) The derivatives of the state voltage with respect time, t, for the *inner cells* are :

(a) -1.75; (b) -1.75; (c) -2.75; (d) -0.75; (e) -1.75; (f) 0.25.

We can see that only the *cells* from case (f) have positive state voltage derivatives and the other *cells* all have negative state voltage derivatives at the initial time of the circuit transient. The state voltage derivatives of the *cells* of the cellular neural network during the transient period can be computed by using the same method presented above. However, circuit simulation with graphics output, just like the image shown in the Fig. 3.20, can help us analyze the transient behavior of the cellular neural network in a very convenient manna.

3.4.3. Extract the edges of the square

How can we extract the edges of a square by using the cellular neural network? After the analysis of the transient behavior of the above circuit, We choose the Laplacian operator as shown in Fig. 3.22, and the circuit parameters as the same as those in Section 3.1 with $I = -2.0\times10^{-3}\ A$. The circuit simulation result is shown in Fig. 3.23. Unfortunately, in addition to extracting the edges of the square, we have also extracted four other meaningless points. This is because during circuit transient, the derivatives of the state variables of the cells keep changing. Although some of them are the same

at the initial time, they can be very different after a long period of time. For instance, the cell $C(6,7)$ in Fig. 3.23a has a negative derivative, but in Fig. 3.23c, after 15 time steps of transient simulation, we found its derivative changes to positive.

The reason for this problem is that we did not keep the original information of the image in the circuit transient. If we choose a nonzero feed-forward operators for a cellular neural network, and use the input image both as the input and as the initial condition of the circuit, then we will be able to over the above problem, since the input (input image) will maintain the same value during circuit transients.

3.4.4. An edge detecting cellular neural network

Using our above experience in designing cellular neural networks for structural feature extractions, let us now design a more practical edge detector. In this edge detecting cellular neural network, both the feedback and feed-forward operators are nonzero, whose templets are shown in Fig. 3.24a and Fig. 3.24b, respectively. The other circuit parameters are chosen as follow: $C = 10^{-9}\,F$; $R_x = 10^3\,\Omega$; and $I = -1.5\times10^{-3}\,A$. The circuit simulation results of the above designed cellular neural network for detecting edges of a diamond and a square are shown in Fig. 3.25 and Fig. 3.26, respectively, where the input images are chosen for both the inputs and the initial conditions of the edge detector. From the pictures in Fig. 3.25 and Fig. 3.26, we can see the perfect performance of this cellular neural network. Note that, although we can use a digital computer to do the same job as the above edge detector, here the processing speed of the cellular neural network is much faster than that of digital computers. For the circuit parameters chosen in this example, the processing speed is about 10^{-6} seconds. Moreover, the processing speed of cellular neural networks are independent of the circuit size, this means that we can process a 16×16 and a 512×512 images using the same time.

3.4.5. A corner detecting cellular neural network

To obtain a corner detecting cellular neural network, we chang only the circuit parameter I into $-1.5\times10^{-3}\,A$ and keep the other circuit parameters the same as those in the edge detector designed in the preceding section. The circuit simulation results of the this corner detector for detecting the corners of a diamond and a square are

shown in Fig. 2.27 and Fig. 2.28, respectively.

3.5. Cellular neural networks for Chinese character recognition

There are about 60,000 Chinese characters and approximately 6,000 of them are used in the daily life. The prohibitively large number of the Chinese characters makes the Chinese character recognition problem much more difficult than other character recognition problems. For the past twenty years, research on the Chinese character recognition problem has focused exclusively on algorithms using digital computers.

Theoretically, the Chinese character recognition problem has been solved for the 6,000 most commonly used characters [22]. But the slow recognition speed remains the main bottle-neck in practice. To the best of our knowledge, the current average speed for the Chinese character recognition using ordinary digital computers is 2 characters/ second. This recognition speed is much too slow for practical needs. The reason for this low recognition speed is that the nature of the algorithm for Chinese character recognition involves mainly parallel processing, because of the two-dimensional structure of the characters, but the conventional digital computers are sequential processing machines.

From the above feature extraction examples, we have seen that the cellular neural networks can extract certain features of images using appropriate dynamic rules. Consequently, the cellular neural network could be an efficient tool for solving the Chinese character recognition problem. First, we can design various cellular neural networks for extracting different features from Chinese characters. Then we can pass the character image simultaneously to all distinct feature extraction circuits in parallel. After the transient has settled down, (the time constants of cellular neural networks are generally less than 10^{-6} seconds) we would have extracted the different features of the original input character, which can then be used for a higher level character recognition by using computers or any other kind of processing machines. In the following, we will present a feature extraction circuit and use it to process a few simple Chinese characters.

The feature we want to extract is the convex corners of the strokes of the Chinese characters. So, we will use the corner detector designed in Section 3.4.5 as our feature extracting cellular neural network. Fig. 3.29 to Fig. 3.36 are the circuit simulation

results of 8 Chinese characters obtained by using our above feature extraction circuit. From the simulation results, it can be seen that the convex corners of the strokes of the characters have been extracted. These corners contain most structural information of the characters, and can be used for coding the characters. The purpose of these examples is to demonstrate the feature-extraction capability of cellular neural networks in image processing. What kind of structural features are useful for Chinese character recognition and how to extract them using cellular neural networks represent two important future research problems.

The resolution of the character images in this figures is poor because of the small size of our pixel array. In Chinese character recognition, the typical pixel array for the character images is 48×48 or 64×64. Our experience shows the larger the size of the cellular neural network, the better is the feature extraction capability for the characters. The VLSI technique will make it possible to implement large-size cellular neural networks.

4. MATHEMATICAL FOUNDATION

In the above sections, we have presented the cellular neural network model and some examples of its applications in image processing. In this section, we will discuss some related research topics on cellular neural networks.

4.1. The mathematical model of cellular neural networks

In general, cellular neural networks can be characterized by a *large* system of ordinary differential equations from the mathematical point of view. Since all the *cells* are arranged in a regular array, we can exploit many spatial properties, such as regularity, sparsity and symmetry in studying the dynamics of cellular neural networks.

There are two mathematical models which can characterize dynamical systems having these spatial properties. One is *partial differential equations,* and the other is *cellular automata.* Our objective in this section is to find the relationships between our cellular neural networks and these two mathematical models.

Consider the partial differential equations first. The well known *heat equation* from physics is

$$\frac{\partial^2 u(x,y,t)}{\partial x^2} + \frac{\partial^2 u(x,y,t)}{\partial y^2} = \frac{1}{\kappa}\frac{d\ u(x,y,t)}{d\ t} \tag{4.1}$$

where κ is a constant, called the *thermal conductivity*. The solution, $u(x,y,t)$, of the *heat equation* is a continuous function of the time, t, and the space variables, x and y. Suppose we use a set of functions $u_{ij}(t)$ to approximate the function $u(x,y,t)$ by discretizing the continuous two-dimensional space into a two-dimensional grid, that is

$$u_{ij}(t) = u(ih_x,\ jh_y,\ t) \tag{4.2}$$

where h_x and h_y are the space interval in the x and y coordinates. Then, the partial derivatives of the $u(x,y,t)$ respect to x and y can be replaced approximately by

$$\frac{\partial^2 u(x,y,t)}{\partial x^2} + \frac{\partial^2 u(x,y,t)}{\partial y^2} \approx \frac{1}{4}\left[\ u_{ij-1}(t) + u_{ij+1}(t) + u_{i-1j}(t) + u_{i+1j}(t)\ \right] - u_{ij}(t)$$

$$\text{for } \textit{all}\ \ i,\ j. \tag{4.3}$$

So, the *heat equation* can be approximated by a set of equations

$$\frac{1}{\kappa}\frac{d\ u_{ij}(t)}{d\ t} = \frac{1}{4}\left[\ u_{ij-1}(t) + u_{ij+1}(t) + u_{i-1j}(t) + u_{i+1j}(t)\ \right] - u_{ij}(t)$$

$$\text{for}\quad \textit{all}\quad i,\ j. \tag{4.4}$$

Comparing equation (4.4) with equation (2.2), we can see a remarkable similarity between these two equations. They are both the ordinary differential equations with the nearest neighbor variables involved in the dynamic rules. The important difference between these two equations is that our cell equation (2.2) is a nonlinear (piecewise linear) ordinary differential equation whereas equation (4.4) is a linear ordinary differential equation.

Consider next the relationship between our cellular neural network cellular circuit model and the cellular automata model. The two-dimensional cellular automaton is defined by [14]

$$a_{ij}(n+1) = \phi\left[\ a_{kl}(n)\quad \text{for } \textit{all}\quad C(k,l) \in N(i,\ j)\ \right] \tag{4.5}$$

If we discretize the time, t, in the *cell* equation (2.2) and let $B(i,j;k,l) = 0$ for all

i,j,k and l, we would obtain

$$\frac{C}{h}\left[v_{xij}((n+1)h) - v_{xij}(nh) \right] = -\frac{1}{R_x}v_{xij}(nh) + \sum_{(k,l)\in N(i,j)} A(i,j;k,l)v_{yij}(nh) + I$$

$$1 \le i \le M,\ 1 \le j \le N \tag{4.6a}$$

and

$$v_{yij}(nh) = 0.5R_y\ (|v_{xij}(nh) + 1| - |v_{xij}(nh) - 1|) \quad 1\le i\le M, 1\le j\le N. \tag{4.6b}$$

After rearranging equation (4.6a) and substituting the resulting expression for $v_{xij}((n+1)h)$ into equation (4.6b), we obtain

$$v_{yij}(n+1) = \phi'\left[v_{xij}(n), v_{ykl}(n) \quad \text{for all } C(k,l)\in N(i,j) \right] \tag{4.7}$$

where

$$v_{yij}(n) = v_{yij}(nh), \quad v_{xij}(n) = v_{xij}(nh), \tag{4.8a}$$

$$\phi'\left[v_{xij}(n), v_{ykl} \quad \text{for all } C(k,l)\in N(i,j) \right] = g\left[\left(\frac{h}{CR_x} + 1 \right) v_{xij}(n) + \frac{h}{C}\sum_{(k,l)\in N(i,j)} A(i,j;k,l)v_{yij}(n) + \frac{hI}{C} \right] \tag{4.8b}$$

and

$$g[z] = \frac{1}{2}\ (|z + 1| - |z - 1|). \tag{4.8c}$$

Comparing equation (4.5) and equation (4.7), we can once again see a remarkable similarity between them. The main difference is that for cellular automata, the state variables are binary value variables and the dynamic function is a logic function of the previous states of the neighbor cells, whereas for cellular neural networks, the state variables are real-valued variables and the dynamic function is a nonlinear real function of the previous states of the neighbor cells.

Our comparisons of the above three mathematical models are summarized in Table 4.1.

Table 4.1. Comparison of three mathematical models having spatial regularities

Model	Cellular neural network	Partial differential equation	2-D cellular automata
time	continuous	continuous	discrete
space	discrete	continuous	discrete
state value	real	real	binary number
dynamics	nonlinear	linear (for (4.1))	nonlinear

4.2. Multilayer cellular neural networks

A direct generalization of the cellular neural network introduced in Section 2.1 is the multilayer cellular neural network model. The generalization is that instead only one state variable there may be several state variables in a *cell* of the multilayer cellular neural network. To avoid clutter, it is convenient to define an operator * as follows. (It is similar to the two-dimensional convolution operator in image processing.)

Definition 4.1

For any templet, T, of the dynamic rule for the *cell* circuit shown in Fig. 3.2, we define the convolution operator * by

$$T * v_{ij} = \sum_{(k,l)\in N(i,j)} T(k-i,l-j)v_{ij.} \tag{4.9}$$

Observe that $A(i,j;k,l)$ and $B(i,j;k,l)$ are always independent of i and j in cellular neural networks. This property is said to be *space invariant* in image processing, which implies the following equivalent forms :

$$A(i,j;k,l) = A(k-i,l-j) \text{ and } B(i,j;k,l) = B(k-i,l-j)$$

$$\text{for } all \ i,j,k \text{ and } l \tag{4.10}$$

Using the above notation, we can rewrite (2.2a) as

$$C\frac{d\ v_{xij}(t)}{d\ t}=\frac{-1}{R_x}v_{xij}(t)+A\ *\ v_{yij}(t)+B\ *\ v_{uij}+I$$

$$1\le i\le M\ ,\ 1\le j\le N. \tag{4.11}$$

For multilayer cellular neural networks, the *cell* dynamic equations can be expressed in the following compact vector form :

$$\mathbf{C}\frac{d\ \mathbf{v}_{xij}(t)}{d\ t}=-\mathbf{R}^{-1}\mathbf{v}_{xij}(t)+\mathbf{A}\ *\ \mathbf{v}_{yij}(t)+\mathbf{B}\ *\ \mathbf{v}_{uij}+\mathbf{I}$$

$$1\le i\le M\ ,\ 1\le j\le N, \tag{4.12}$$

where

$$\mathbf{C}=\begin{bmatrix}C_1 & & \\ & \ddots & \\ & & C_m\end{bmatrix};\qquad \mathbf{R}=\begin{bmatrix}R_{1x} & & \\ & \ddots & \\ & & R_{mx}\end{bmatrix}; \tag{4.13a}$$

$$\mathbf{A}=\begin{bmatrix}A_{11} & \cdots & A_{1m}\\ \vdots & & \vdots\\ A_{m1} & \cdots & A_{mm}\end{bmatrix};\quad \mathbf{B}=\begin{bmatrix}B_{11} & \cdots & B_{1m}\\ \vdots & & \vdots\\ B_{m1} & \cdots & B_{mm}\end{bmatrix}; \tag{4.13b}$$

$$\mathbf{v}_{xij}=\begin{bmatrix}v_{1xij}\\ \vdots\\ v_{mxij}\end{bmatrix};\qquad \mathbf{v}_{yij}=\begin{bmatrix}v_{1yij}\\ \vdots\\ v_{myij}\end{bmatrix};\qquad \mathbf{v}_{uij}=\begin{bmatrix}v_{1uij}\\ \vdots\\ v_{muij}\end{bmatrix}; \tag{4.13c}$$

$$\mathbf{I}=\begin{bmatrix}I_1\\ \vdots\\ I_m\end{bmatrix} \tag{4.13d}$$

and m denotes the number of the variables in the *multilayer cell* circuit. Here, the

convolution operator * between a matrix and a vector is to be decoded like matrix multiplication but with the operator * inserted between each entry of the matrix and of the vector.

Observe that **C** and **R** are diagonal matrices, whereas **A** and **B** are square matrices whose elements are entries of the templets of the dynamic rules for each state variables in the *cells*.

Remarks :

(a) For multilayer cellular neural networks, all the results presented in Section 2 still hold except for some minor modifications.

(b) Since there are several state variables in a *cell* circuit, we can choose multiple dynamic rules concurrently for the different state variables. This property makes it extremely flexible for us to deal with more complicated feature extraction problems, or other related image processing problems.

(c) In addition to using multiple dynamic rules as mentioned in (b), we can choose different *time constants* for the different state variables of the *cell* circuits. As a limiting case, we can choose $C_q = 0$ for some state variable v_{qxij}, thereby obtaining a set of differential and algebraic equations. This property give us even more flexibility in the design of the cellular neural networks for practical problems.

4.3. Improving the efficiency of computer simulation of cellular neural networks

All computer-generated results presented in this paper are produced by the circuit simulator, PWLSPICE, which is a general purpose circuit simulator for nonlinear circuits. It does not take advantage of many special properties of cellular neural networks. The average speed of circuit simulation for a 16×16 one-layer cellular neural network with a 3×3 *cell* neighborhood is about 10 minutes running on a MICROVAX-II. In practice, the required circuit size is much large than 16×16, for instance, in the Chinese character recognition, the circuit size needed is 48×48 or 64×64. Therefore, it is necessary to develop a more efficient special circuit simulator for the design of the cellular neural network circuits. The properties of *parallelism, regularity, sparsity* and the *piecewise-linear* nonlinearity can all be exploited to

improve the simulation speed.

5. CONCLUDING REMARKS

In this paper, we have proposed a *new* circuit model, called a *cellular neural network,* and discussed both the theoretical and practical problems associated with this model. We have proved some theorems concerning the dynamic range and the steady-states of cellular neural networks. We have presented some examples of the application of cellular neural networks in image processing and Chinese character recognition. In view of the *nearest neighbor* interactive property of cellular neural networks, they are much more amenable fore VLSI implementation than general neural networks. But since this is the first study of cellular neural networks, there are many theoretical and practical problems yet to be solved in future research on this subject.

REFERENCES

[1] J. J. Hopfield, "Neural networks and physical systems with emergent computational abilities," *Proc. Natl. Acad. Sci. USA.*, vol. 79, pp 2554-2558, 1982.

[2] J. J. Hopfield and D. W. Tank, "'Neural' computation of decisions in optimization problems," *Biological Cybernetics,* vol. 52, pp.141-152, 1985.

[3] J. J. Hopfield and D. W. Tank, "Computing with neural circuits: a model," *Science (USA),* vol. 233, no.4764, pp.625-633, 1986.

[4] D. W. Tank and J. J. Hopfield, "Simple 'neuron' optimization networks : an A/D converter, signal decision circuit, and a linear programming circuit," *IEEE Trans.,* CAS-33, no.5, pp.533-541, 1986.

[5] M. P. Kennedy and L. O. Chua, "Unifying the Tank and Hopfield linear programming network and the canonical nonlinear programming of Chua and Lin," *IEEE Trans.,* CAS-34, no.2, pp.210-214, 1987.

[6] T. Kohonen, *Self-organization and Associative memory.* New York: Springer-Verlag, 1984.

[7] C. Koch et al, "Analog 'neuronal' networks in early vision," *Proc. Natl. Acad. Sci. USA,* vol. 83, p. 4263, 1986.

[8] I. Kanter and H. Sompolinsky, "Associative recall of memory without errors," *Physical Review A,* vol. 35, no.1, pp.380-392, 1987.

[9] J. Buhmann and K. Schulten, "Associative Recognition and Storage in a Model network of Physiological Neurons," *Biol. Cybern.,* vol. 54, pp.319-335, 1986.

[10] L. O. Chua, C. A. Desoer and E. S. Kuh, *Linear and Nonlinear circuits.* New York: McGraw-Hill, 1987.

[11] Leon. O. Chua, *Introduction to Nonlinear Network Theory* New York: McGraw-Hill, 1969.

[12] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*. New York: Academic Press, 1982.

[13] L. O. Chua and P. M. Lin, *Computer Aided Analysis of Electronic Circuits: Algorithms and Computational Techniques*. Prentice-hall, Englewood Cliffs, NJ, 1975.

[14] L. O. Chua and R. L. P. Ying, "Finding all solutions of piecewise-linear circuits", *J. Circuit theory and Applications*, vol. 10, pp. 201-229, 1982.

[15] L. O. Chua and A. C. Deng, "Canonical piecewise-linear analysis : generalized breakpoint hopping algorithm," *Int. J. Circuit Theory and Applications*, vol. 14, pp.35-52, Jan. 1986.

[16] S. Wolfram (eds), *Theory and applications of cellular automata*. World Scientific Publishing Co., 1986.

[17] N. Packard and S. Wolfram, "Two-dimensional cellular automata", *J. Stat. Phys.*, vol. 38, Nos. 5/6, pp. 901-946, 1985.

[18] T. Toffoli and N. Margolus, *Cellular Automata Machines -- a new environment for modeling*. Cambridge, MA: M.I.T. Press, 1986.

[19] T. Quarles, A. R. Newton, D. O. Pederson and A. Sanjgiovanni-Vincentelli, "SPICE.3A7 User's Guide," Department of Electrical Engineering and Computer Sciences University of California, Berkeley, June 13, 1986.

[20] B. K. P. Horn, *Robot vision*. Cambridge, MA: M.I.T. Press, 1986.

[21] D. Marr and E. Hidreth, "Theory of edge detection," *Proc. Roy. Soc. London*, vol. B 207, pp. 187-217, 1980.

[22] X. Zhu, Y. Wu and X. Ding, "The recognition of 6763 printed Chinese characters," *J. Tsinghua Uni.*, vol. 27, no. 1, pp. 39-49, 1987.