CELLULAR NEURAL NETWORKS OPERATING IN OSCILLATORY MODES

by

Chai Wah Wu, Tamás Roska, and Leon O. Chua

Memorandum No. UCB/ERL M94/5

3 February 1994

CELLULAR NEURAL NETWORKS OPERATING IN OSCILLATORY MODES

by

Chai Wah Wu, Tamás Roska, and Leon O. Chua

Memorandum No. UCB/ERL M94/5

3 February 1994

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

CELLULAR NEURAL NETWORKS OPERATING IN OSCILLATORY MODES

by

Chai Wah Wu, Tamás Roska, and Leon O. Chua

Memorandum No. UCB/ERL M94/5

3 February 1994

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Cellular Neural Networks Operating in Oscillatory Modes

Chai Wah Wu Tamás Roska* Leon O. Chua

June 1992, Last Revised: February 3, 1994

Electronics Research Laboratory and
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
CA 94720, U. S. A.

Abstract

Several applications of cellular neural networks (CNN) operating in oscillatory mode are presented. In particular, we will use oscillation to indicate the presence of a vertical or horizontal edge in the image, and to perform a spatial-temporal halftoning operation.

1 Introduction

Cellular neural networks (CNN) are analog parallel processing arrays capable of high-speed computation while amenable to VLSI implementation [1, 2]. Many applications of CNN's has been found [3, 4]. Most applications to date use the steady state behavior of the CNN's. We present here several applications that uses the transitory behavior of the network. In particular, we will present in section 2 a CNN which detects edges through oscillations and in section 3 we propose CNNs which perform spatial-temporal halftoning via error diffusion. In section 4 several applications using these spatial-temporal halftoning CNNs are discussed and future research directions are indicated.

2 Oscillating Edge Detector

In this section we present a CNN that detects edges by making the cells which corresponds to edges in the image oscillate.

*Also with the Analogical and Neural Computing Systems Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, Kende-u 13, Budapest, H-1111, Hungary.

2.1 Nonlinear Oscillator

The basic component of the oscillating CNN template is a two-state oscillator which is derived from a Connected Component Detector [5]. The oscillator has the following state equations:

$$\begin{aligned} \dot{x}_1 &= -x_1 + 2f(x_1) + 1.5f(x_2) \\ \dot{x}_2 &= -x_2 + 2f(x_2) - 1.5f(x_1) \end{aligned} \tag{1}$$

where $f(x) = \frac{1}{2}[|x+1| - |x-1|]$.

We will primarily use the following results:

Theorem 1 *Consider the equation*

$$\dot{x} = -x + pf(x) + s_1 f(g_1(t)) - s_2 f(g_2(t)) + c \tag{2}$$

where $p > 1$, $s_1 > 0$, and $s_2 > 0$ and g_1 and g_2 are two arbitrary functions which depends on time t. If $c > s_1 + s_2 + p - 1$ then $\exists t_1 \forall t > t_1 (x(t) > 1)$. If $c < -(s_1 + s_2 + p - 1)$ then $\exists t_1 \forall t > t_1 (x(t) < -1)$. Thus if $|c| > s_1 + s_2 + p - 1$ then $x(t)$ will reach the saturation region ($|x(t)| > 1$) after some time.

Proof: Suppose that $c > s_1 + s_2 + p - 1$. Consider the differential equation,

$$\dot{z} = -z + pf(z) + d(p-1) \tag{3}$$

for $d > 1$. Looking at the graph of $\dot{z}$ versus z (Fig. 1), it is clear that $z(t) > 1$ after some time. Then since for sufficiently small $d > 1$, $s_1 f(g_1(t)) - s_2 f(g_2(t)) + c > d(p-1)$, it follows from a comparison theorem [6] that $x(t) > 1$ after some time. Similarly when $d < -1$, $z(t) < -1$ and thus $x(t) < -1$ after some time. ■

Theorem 2 *Consider the equations*

$$\begin{aligned} \dot{x}_1 &= -x_1 + p_1 f(x_1) + s_{12} f(x_2) + c_1 \\ \dot{x}_2 &= -x_2 + p_2 f(x_2) - s_{21} f(x_1) + c_2 \end{aligned} \tag{4}$$

where $s_{12} + 1 > p_1 > 1$ and $s_{21} + 1 > p_2 > 1$. If $|c_1| < s_{12} + 1 - p_1$ and $|c_2| < s_{21} + 1 - p_2$ then for almost all initial conditions, the solutions of (4) will oscillate (i.e. the solutions will approach a periodic solution).

Proof: First we show that there are no equilibrium points in the regions $|x_1| \geq 1$ or $|x_2| \geq 1$. A fixed point (x_1^*, x_2^*) of system (4) must satisfy:

$$-x_1^* + p_1 f(x_1^*) + s_{12} f(x_2^*) + c_1 = 0 \tag{5}$$

$$-x_2^* + p_2 f(x_2^*) - s_{21} f(x_1^*) + c_2 = 0 \tag{6}$$

Suppose $x_1^* \geq 1$. Then $-x_2^* + p_2 f(x_2^*) = s_{21} - c_2 \geq s_{21} - |c_2| > p_2 - 1$. If $x_2^* > -1$, then $-x_2^* + p_2 f(x_2^*) \leq p_2 - 1$, therefore $x_2^* \leq -1$ and we have from (5) $x_1^* = p_1 - s_{12} + c_1 \leq |c_1| + p_1 - s_{12} < 1$ which is a contradiction.

Suppose $x_1^* \leq -1$. Then $-x_2^* + p_2 f(x_2^*) = -s_{21} - c_2 \leq -s_{21} + |c_2| < 1 - p_2$. If $x_2^* < 1$, then $-x_2^* + p_2 f(x_2^*) \geqq 1 - p_2$, therefore $x_2^* \geq 1$ and we have from (5) $x_1^* = -p_1 + s_{12} + c_1 \geq -|c_1| - p_1 + s_{12} > -1$ which is a contradiction.

Suppose $x_2^* \geq 1$. Then $-x_1^* + p_1 f(x_1^*) = -s_{12} - c_1 \leq -s_{12} + |c_1| < 1 - p_1$. If $x_1^* < 1$, then $-x_1^* + p_1 f(x_1^*) \geq 1 - p_1$, therefore $x_1^* \geq 1$ and we have from (6) $x_2^* = p_2 - s_{21} + c_2 \leq |c_2| + p_2 - s_{21} < 1$ which is a contradiction.

Suppose $x_2^* \leq -1$. Then $-x_1^* + p_1 f(x_1^*) = s_{12} - c_1 \geq s_{12} - |c_1| > p_1 - 1$. If $x_1^* > -1$, then $-x_1^* + p_1 f(x_1^*) \leq p_1 - 1$, therefore $x_1^* \leq -1$ and we have from (6) $x_2^* = -p_2 + s_{21} + c_2 \geq -|c_2| - p_2 + s_{21} > -1$ which is a contradiction.

Thus an equilibrium point must satisfy $|x_1^*| < 1$, $|x_2^*| < 1$ and:

$$(p_1 - 1)x_1^* + s_{12}x_2^* + c_1 = 0 \quad (7)$$

$$(p_2 - 1)x_2^* - s_{21}x_1^* + c_2 = 0 \quad (8)$$

Since $(p_1 - 1)(p_2 - 1) + s_{12}s_{21} > 0$, there is a unique solution to the equations above, which is

$$\begin{aligned} x_1^* &= \tfrac{1}{(p_1-1)(p_2-1)+s_{12}s_{21}}(s_{12}c_2 - (p_2 - 1)c_1) \\ x_2^* &= \tfrac{1}{(p_1-1)(p_2-1)+s_{12}s_{21}}(-s_{21}c_1 - (p_1 - 1)c_2) \end{aligned} \quad (9)$$

If for x_1^* and x_2^* as defined by (9) satisfy $|x_1^*| < 1$ and $|x_2^*| < 1$, then that is the unique equilibrium point of the system, otherwise the system has no equilibrium point. The trajectories are bounded [1], so if there are no equilibrium points, then by the Poincaré-Bendixon theorem [7], there exists a closed orbit, and by index theory [8], the closed orbit must enclose at least one equilibrium point, which is a contradiction.

Thus the system has a unique equilibrium point, given by equation (9). The Jacobian at the equilibrium point is $\begin{bmatrix} p_1 - 1 & s_{12} \\ -s_{21} & p_2 - 1 \end{bmatrix}$. Since the trace of the Jacobian is positive, it has eigenvalues on the right half plane[1], so the equilibrium point is unstable, and by the Poincaré-Bendixon Theorem [7], this system contains closed orbits and almost all initial conditions generate a closed orbit or approach a limit cycle. In other words, the system will oscillate. By calculating the index [8], we can show that all periodic solutions of this system must enclose the equilibrium point. ■

[1]In fact, by using index theory, one can show that both eigenvalues are in the right half plane.

2.2 Oscillating Edge Detector

Now consider the CNN with the following cloning templates:

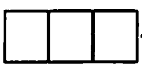
$$A = \begin{bmatrix} -1.5 & 2 & 1.5 \end{bmatrix}, B = \begin{bmatrix} 3 & 0 & 3 \end{bmatrix}, I = -1.5 \tag{10}$$

The input image is placed as input to the CNN and takes on values 1 for black pixels and -1 for white pixels. We assume that the relevant images has enough resolution such that the features are at least 3 pixels wide (i.e. white lines and black lines are at least 3 pixels thick). Then over each 3-cell window in each row of the input. we have the following 4 cases:

Case 1: The input is :  or 

If we denote the three cells in the window as x_{i-1}, x_i and x_{i+1} respectively, then the differential equation governing x_i will be

$$\frac{dx_i}{dt} = -x_i + 2f(x_i) - 1.5f(x_{i-1}) + 1.5f(x_{i+1}) - 1.5 \pm 6 \tag{11}$$

By theorem 1 $f(x_i) \to 1$ as $t \to \infty$ for the case  and $f(x_i) \to -1$ as $t \to \infty$ for the case .

Case 2: The input is : 

Let's denote the three cells in the window as x_{i-1}, x_i and x_{i+1} respectively. By case 1 and the fact that all lines are at least 3 pixels wide, $f(x_{i-1}) \to 1$, so after some time the differential equation governing x_i will be

$$\frac{dx_i}{dt} = -x_i + 2f(x_i) + 1.5f(x_{i+1}) - 3.0 \tag{12}$$

So by theorem 1 $f(x_i) \to -1$ as $t \to \infty$.

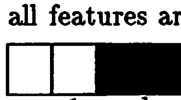
Case 3: The input is : 

Let's denote the three cells in the window as x_{i-1}, x_i and x_{i+1} respectively. By case 1, $f(x_{i+1}) \to -1$ and by case 2, $f(x_{i-1}) \to -1$. So after some time the differential equation governing x_i will be

$$\frac{dx_i}{dt} = -x_i + 2f(x_i) - 1.5 \tag{13}$$

So by theorem 1 $f(x_i) \to -1$ as $t \to \infty$.

Case 4: The input is :  or 

Because we assume all features are three pixels wide, the two inputs occurs together, i.e. the cells form the pattern  . If we label these four cells as x_{i-1}, x_i, x_{i+1} and x_{i+2} respectively, then from case 1, we know that $f(x_{i-1})$ converges to -1 and $f(x_{i+2})$ converges to 1. Thus after some time the differential equations governing x_i and x_{i+1} looks like:

$$\frac{dx_i}{dt} = -x_i + 2f(x_i) + 1.5f(x_{i+1}) \tag{14}$$

$$\frac{dx_{i+1}}{dt} = -x_{i+1} + 2f(x_{i+1}) - 1.5f(x_i) \tag{15}$$

This has the form of equation (4) and by theorem 2 it will oscillate. Thus we see that the cells immediate to the left and right of a left sided edge will oscillate while the other cells will move to the saturation regions (output = ± 1).

Scolium: We can of course also construct templates for detecting horizontal edges and right side edges by rotating the templates.

By adding external periodic forcing or by making each cell a second order cell, we can drive the system into chaos [9].

3 Error Diffusion for Spatial-temporal Halftoning

Halftoning is the operation where a grayscale image is transformed into a binary image such that when viewed from a distance the two images appear the same. Error diffusion is a popular method of performing halftoning of grayscale images [10]. Consider the following scheme of performing error diffusion:

$$\begin{aligned} x_{ij}(n) &= \textstyle\sum_{k,l,m} h(k-i, l-j, m-n)u_{kl}(m) - \sum_{k,l,m} w(k-i, l-j, m-n)(g(x_{kl}(m))) \\ y_{ij}(n) &= \operatorname{sgn}(x_{ij}(n)) \end{aligned} \tag{16}$$

where $g(x) \triangleq \operatorname{sgn}(x) - x$ and h and w are two spatio-temporal filters which are recursive with respect to a particular ordering. We will think of h and w as three-dimensional filters where the first two dimensions correspond to a two-dimensional image space and the third dimension to time[2]. We then have the following interpretation of the error diffusion scheme:

Theorem 3 *The output y_{ij} in equation (16) are equal to the y_{ij} in the following minimization procedure :*

$$y_{ij}(n) = \operatorname{Arg} \min_{y_{ij}(n) \in \{-1,1\}} |A * y - B * u|_{ijn} \tag{17}$$

where $A = Z^{-1}\{1 + \frac{\tilde{w}}{1-\tilde{w}}\}$ and $B = Z^{-1}\{\frac{\tilde{h}}{1-\tilde{w}}\}$.

Proof: the proof is essentially Proposition 1 in [10]. Taking the 3-dimensional Z-transform,

$$\tilde{x} = \tilde{h}\tilde{u} - \tilde{w}\tilde{y} + \tilde{w}\tilde{x} \tag{18}$$

[2] As the third coordinate is time, we will sometimes use the notation $h(i,j)[n]$ to denote $h(i,j,n)$.

After some manipulation we get

$$x_{ij}(n) = (B * u - A * y)_{ijn} + y_{ij}(n) \tag{19}$$

$$y_{ij}(n) = \text{sgn}(B * u - A * y)_{ijn} + y_{ij}(n) \tag{20}$$

Since $A(0,0,0) = 1$, the conclusion follows from Lemma 1 in [10]. ■

In other words, the output will minimize the difference between a filtered version of the output and a filtered version of the input. A halftoned image can be considered as a spatial oscillation. In spatial-temporal error diffusion, time is added as another dimension, and thus we will also have temporal oscillations. We will now propose several architectures which approximate this error diffusion scheme.

3.1 Recursive Nonlinear Difference Equations

This scheme merely implements equation (16) by truncating the filters h and w to a finite length. The network consists of a three-dimensional array of cells $C_{i,j,k}$ whose state corresponds to $x_{i,j}(n-k)$ and whose output corresponds to $y_{ij}(n-k)$ for a finite number of k's, where the top layer corresponds to $k = 0$, i.e. the most recent states. The states are calculated in each time step in the topmost layer according to equation (16) and the current states are shifted to the next layer. The network can be thought of as a layered network with each layer consisting of the cells $C_{i,j,k}$ for a fixed k. To make this realizable, there must exist an ordering which make the filter h and w causal and recursive.

3.2 Parallel Processors

One drawback of the first scheme is that in the computation layer, each cell must wait until the cell before it in the ordering of h and w has generated an output. By making $h(i,j,n) = w(i,j,n) = 0$ for $i \neq j$, which removes the spatial dependency, all the cells in the computation layer can process the data in parallel.

3.3 Continuous Time Neural Networks with Buffers

By using an architecture such as the CNN Universal Machine [11], continuous time neural networks can perform a computation, the result can be stored, and used as the input to another continuous-time neural network. The minimization in each layer can be done by a CNN similar to that used in [10], while the computation between layers is done in a discrete-time fashion. We start with the following CNN for each time iterate n:

$$\begin{aligned}\frac{d\hat{x}_{ij}}{dt} &= -\hat{x}_{ij} + \sum_{k,l}(A(k-i,l-j)[0]\hat{y}_{kl} + B(k-i,l-j)[0]u_{kl}[n]) \\ &\quad + \sum_{m>0,k,l}(B(k-i,l-j)[m]u_{kl}[n-m] - A(k-i,l-j)[m]y_{ij}[n-m]) \quad (21)\\ \hat{y}_{ij} &= f(\hat{x}_{ij}) \quad (22)\end{aligned}$$

At each iteration n, after a steady state solution of equation (21) is reached, we set $y_{ij}[n] = f(\hat{x}_{ij})$. The output of the system will be $y_{ij}[n]$. In the case when $\hat{x}_{ij}$ (almost) always converges to the saturation regions (such as in halftoning [10]), the buffer $y_{ij}[n]$ needed to store the result can be logical bits.

When A and B are properly chosen, from proposition 2 in [10], the output $y_{ij}[n]$ will be "close" to the value $y^*_{ij}[n]$:

$$y^*_{ij}[n] = \text{Arg} \min_{y^*_{ij}[n] \in \{-1,1\}} |A * y^* - B * u|_{ijn} \quad (23)$$

4 Applications and Areas of Future Research

We now suggest three applications of spatial-temporal halftoning where the input data (1) varies in time and space, (2) varies only in space (3) varies only in time and give a brief discussion of future research problems.

4.1 Compression of Moving Images

The output of the error-diffusion algorithm can be thought of as compressing the input data. At the receiving channel, an appropriate filter can be used to recover the time-varying data.

When architecture 3 is used in this application, we get a setup similar to that in [12].

4.2 Halftoning of Subsampled Images

One of the exciting application of CNN halftoning is that a VLSI chip can be fabricated where each cell is directly connected to an input sensor. As was noted in [10], it may not be possible to build a CNN of the size required. Halftoning is used by for example FAX machines to transmit images. When a grayscale image is halftoned to a binary image of the same size, some information is lost. Therefore a halftone larger than then original is desirable. In other words, the resolution of the scanner in the FAX machine should be lower than the resolution of the printer in the FAX machine. In these cases it is desirable to generate a halftoned image larger than the inputsize/cellarraysize. Furthermore, in halftoning, we want to oversample the image spatially to move the quantization noise to higher frequencies which can be filtered out without degrading the output image too much. Therefore it is desirable to generate a halftoned image of a larger size than the input image. One manner in which this can be done is to use interpolation to obtain a larger input images and perform halftoning on this image. But this requires either a larger array of cells or partitioning of the image. An alternative way of achieving this is to use spatial-temporal halftoning, and merge the output in different time steps to obtain a larger output image. For example, we can merge four images into an image four times as big by taking one pixel from each of the four images and put them into 2×2 blocks.

4.3 Oversampled Sigma-Delta Modulation

Oversampled sigma-delta modulation is a technique which halftones a scalar input in time to obtain a binary representation of an analog value. To obtain an accurate representation of the analog input value, the input data needs to be oversampled, i.e. the system needs to operating at high speed. To lower the oversampling rate, a finer quantizer can be used, or more stages in the sigma-delta modulator. We propose to use spatial-temporal halftoning to lower the oversampling rate, by constructing the decimation filter to decimate not only in time, but also in space as well. The input to the system is spatially constant (each cell receive the same input) and varies only in time. This will reduce the oversampling rate that is needed. How this technique performs in comparison to other methods of increasing the performance of the basic single-loop, single-bit quantizer sigma-delta modulator that have been proposed such as adding multiple stages and/or using finer quantizers requires further research.

It has been shown that by driving sigma-delta modulators into chaos, one can eliminate the problem of tones that occurs in the output [13]. Such techniques should also be applicable in performing spatio-temporal halftoning.

5 Conclusions

We have shown how CNN's operating in oscillatory modes can be useful in performing certain tasks such as edge detection and spatio-temporal halftoning. Such CNN's can also be driven into chaos which is desirable in certain applications.

Acknowledgement

This work is supported in part by the Office of Naval Research under grant N00014-89-J-1402 and by the National Science Foundation under grant MIP 86-14000.

References

[1] L. O. Chua and L. Yang, "Cellular neural networks: Theory," *IEEE Transactions on Circuits*

and Systems, vol. 35, pp. 1257–1272, 1988.

[2] L. O. Chua and T. Roska, "The CNN paradigm," *IEEE Transaction on Circuits and Systems-I*, vol. 40, pp. 147–156, Mar. 1993.

[3] *Proceedings of the 1990 IEEE International Workshop on Cellular Neural Networks and Their Applications*, (Budapest, Hungary), 1990. IEEE Catalog No. 90TH0312-9.

[4] *Proceedings of the 1992 IEEE Second International Workshop on Cellular Neural Networks and Their Applications*, (Munich, Germany), 1992. IEEE Catalog No. 92TH0498-6.

[5] T. Matsumoto, L. O. Chua, and H. Suzuki, "CNN cloning template: Connected component detector," *IEEE Transactions on Circuits and Systems*, vol. 37, pp. 633–635, 1990.

[6] G. Birkhoff and G.-C. Rota, *Ordinary Differential Equations.* John Wiley & Sons, 4 ed., 1989.

[7] R. K. Miller and A. N. Michel, *Ordinary Differential Equations.* Academic Press, 1982.

[8] J. Guckenheimer and P. Holmes, *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields.* No. 42 in Applied Mathematical Sciences, Springer-Verlag, 1983.

[9] F. Zou and J. A. Nossek, "A chaotic attractor with cellular neural networks," *IEEE Transactions on Circuits and Systems*, vol. 38, pp. 811–812, 1991.

[10] K. R. Crounse, T. Roska, and L. O. Chua, "Image halftoning with cellular neural networks," ERL Memorandum UCB/ERL M91/106, University of California, Berkeley, 1991.

[11] T. Roska and L. O. Chua, "The CNN universal machine: An analogic array computer," *IEEE Transaction on Circuits and Systems-II*, vol. 40, pp. 163–173, Mar. 1993.

[12] M. Tanaka, K. R. Crounse, and T. Roska, "Template synthesis of cellular neural networks for information coding and decoding," in *Proceedings of the Second International Workshop on Cellular Neural Networks and their Applications*, pp. 29–35, 1992.

[13] S. Hein, "Exploiting chaos to suppress spurious tones in general double-loop $\Sigma\Delta$ modulators," *IEEE Transactions on circuits and systems-Part II: Analog and Digital Signal Processing*, vol. 40, pp. 651–659, Oct. 1993.

Figure captions

Figure 1 Graph of $\dot{z}$ versus z, where $\dot{z} = -z + pf(z) + d(p-1)$.

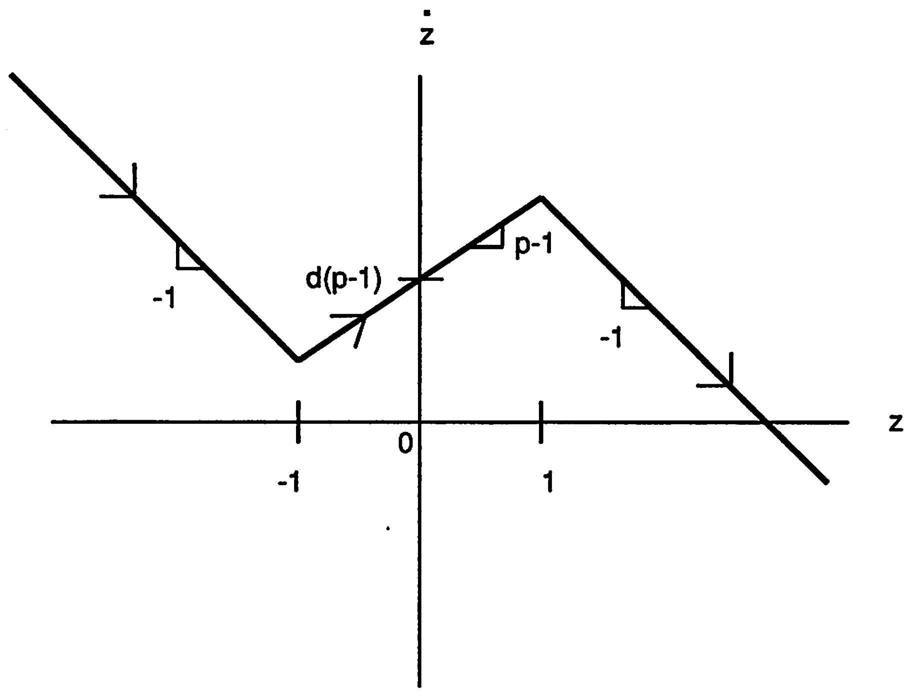


Fig. 1