二值图象平滑算法和细胞神经网络实现 [1]

杨 涛

(上海工业大学自动化系 上海 200072)

摘 要 本文利用细胞神经网络 (CNN) 的基本处理单元——细胞的分段线性饱和输出特性和相平面分析法实现了线性可分和线性不可分布尔函数。并利用这一原则实现了二值图象的多种 CNN 平滑算法。

关键词 细胞神经网络，布尔函数，二值图象，平滑算法

中图号 TN911.73, TN-052

1 引 言

细胞神经网络 (CNN) 是二维或三维的大规模、模拟、动态非线性电路。自 1988 年蔡少棠和杨林 [1,2] 提出其结构以来，不论在硬件实现、软件模拟上，还是在理论研究和实际应用上均取得了极为迅速的发展。大量的文献见两本会议论文集 [3,4] 和三本专辑 [5-7]。CNN 中每个细胞仅与邻近的细胞交换信息，这种局部互连特性使得 CNN 易于用现有的 VLSI 工艺实现。利用 $2\mu m$ 工艺，6×6 细胞的芯片已在 1991 年 7 月推出 [4]。预计 CNN 能被迅速地用于图象的并行处理等需要快速运算的领域，特别是在对于实时性要求很高的视觉机器人的控制等领域，将有极为广阔的应用前景。

由于在采样、量化、传输过程中图象信号中难免有噪声出现，所以对于图象信号进行预处理是必要的。并且在众多的图象预处理方法中，只有那些满足计算速度的低成本的方法才能满足要求。在文献 [2] 中，介绍了一种 CNN 邻域平均算法，文中给出的算例证明，在汉字图象处理中，这种简单的算法是有效的。但文献 [2] 中未给出这种算法用 CNN 实现的理论依据。并且在图象的预处理中，邻域平均算法会使边沿和清晰的细节变得模糊不清，在某些时候是得不偿失的。

本文将集中讨论二值图象的平滑问题。在二值图象中，噪声通常表现为不规则的边界、小孔、丢失的角点和孤立点。由于可以把二值图象的二值性和布尔函数的二值性对应起来，所以布尔函数已成为二值图象处理中的有力工具。在文献 [8] 中，给出了处理：(1) 填充暗区域中的小 (一个象素) 孔；(2) 填补直线段上的缺口；(3) 消除孤立点；(4) 消除直线上的小突起；(5) 恢复丢失角点的平滑算法的布尔表达式。本文将讨论如何用 CNN 来实现这些布尔表达式。

[1] 1994-02-01 收到，1994-06-22 定稿

2 CNN 基本理论

CNN 中每一个处理单元叫细胞，每个细胞与它邻近的若干个细胞组成邻域系统，邻域系统内的细胞相互作用 [1,2] 。

定义 1 设有一单层 $M \times N$ 细胞的 CNN， C_{ij}， $i \in \underline{M}$； $j \in \underline{N}$ 为其中的一个细胞，则 C_{ij} 的 r邻域系统 $N_r(i,j)$ 为

$$N_r(i,j) = \{C_{rl} | \max(|k-i|, |l-j|) \le r, 1 \le k \le M, 1 \le l \le N\}, \tag{1}$$

其中 r 为正整数。

C_{ij} 的状态方程为

$$\dot{x}_{ij}(t) = -k_x x_{ij}(t) + \sum_{C_{kl} \in N_r(i,j)} A(i,j;k,l) y_{kl}(t) + \sum_{C_{kl} \in N_r(i,j)} B(i,j;k,l) u_{kl} + I;\ i \in \underline{M},\ j \in \underline{N},\ k_x > 0, \tag{2}$$

其中 x_{ij} 为细胞 C_{ij} 的状态电压， y_{ij} 为 C_{ij} 的输出电压， $A(i,j;k,l)$ 和 $B(i,j;k,l)$ 分别为反馈和控制权重， u_{kl} 为细胞 C_{kl} 的输出电压， I 为偏置。

C 的输出方程为

$$y_{ij} = (|x_{ij}+1| - |x_{ij}-1|)/2. \tag{3}$$

为了便于表达，以下引入模板和卷积算子的定义。

定义 2 在一个 2-D $M \times N$ CNN 中，如果 (2) 式中 $A(i,j;k,l)$ 和 $B(i,j;k,l)$ 是空间不变的，则在 C_{ij} 的 r邻域系统中控制模板 $B = \{B(g,h)\}$， 反馈模板 $A = \{A(g,h)\}; g,h$ 为整数且满足 $g \in [-r,r], h \in [-r,r]$ 。定义卷积运算 " * " 如下：

$$A * y_{ij} = \sum_{C_{kl} \in N_r(i,j)} A(i-k, j-l) \cdot y_{kl}, \tag{4}$$

$$B * u_{ij} = \sum_{C_{kl} \in N_r(i,j)} B(i-k, j-l) \cdot u_{kl}. \tag{5}$$

由定义 2， (2) 式可改写为

$$\dot{x}_{ij}(t) = -k_x x_{ij}(t) + A * y_{ij} + B * u_{ij} + I. \tag{6}$$

特别地，如果 $k_x = 1$， $A * y_{ij} = 2y_{ij}$， $B * u_{ij} = u$， $I = -1$， 则 (6) 式成为

$$\dot{x}_{ij}(t) = (-x_{ij}(t) + 2y_{ij} - 1) + u. \tag{7}$$

以下根据 u 的不同取值来讨论 (7) 式所对应的相轨迹：

(1) 若 $u = 0$，则相轨迹对应于图 1 中的折线①。①与横轴交于一个不稳定平衡点 (1,0) 和一个稳定平衡点 (-3,0) 。由于在实际电路中存在扰动，因而 (1,0) 观察不到，细胞最终稳定于 (-3,0)， 此时由 (3) 式知细胞输出 $y_{ij} = -1$ 。

(2) 若 $u>0$，则相轨迹如图 1 中折线②所示。不难看出，在 $x_{ij}(0)=1$ 的条件下，C_{ij} 将最终趋近于 (1+u,0) 这一稳定平衡点，此时，$y_{ij}=1$。

(3) 若 $u<0$，则相轨迹如图 1 中③所示。C_{ij} 将最终趋近于 (-3+u,0) 这一唯一的稳定平衡点，此时 $y_{ij}=-1$。

由以上的分析可以看出：如果将 y_{ij} 为 1 和 -1 的两种状态作为逻辑状态 T 和 F 的物理实现，那么便可以通过改变 u 来使由 (7) 式描述的细胞最终输出 T 或 F。这种改变 u 的过程就对应于对逻辑表达式求值的过程，因而 C_{ij} 在这种工作方式下是一个逻辑运算单元，在初态 $x_{ij}(0)$=1 的条件下，其逻辑运算功能满足：

$F1$ 如果 $u>0$，则 y_{ij} 实现 T；

$F2$ 如果 $u\le 0$，则 y_{ij} 实现 F。

其中 $x_{ij}(0)$=1 是由 CNN 硬件初始化来实现的 [7]。

3 CNN 二值图象平滑算法

3.1 填补小孔和缺口

将平面二值图象的每一个象素对应于一平面 CNN 的相应细胞，并称那些代表二值图象象素颜色的细胞为象素细胞。象素细胞的输入值取为相应象素的逻辑值（0 或 1），以这种方式将待处理的图象输入 CNN，那么象素细胞的输出最终决定了输出图象，这种对应关系有很明确的物理意义。

为进行逻辑运算，我们约定以下等价关系：

$R1$ 逻辑 T 等价于：数值 1，$y_{ij}=1$，$u_{ij}=1$，黑细胞，黑象素；

$R2$ 逻辑 F 等价于：数值 -1，$y_{ij}=-1$，$u_{ij}=0$，白细胞，白象素。

若平滑，要达到以下两个目的：

$G1$ 填补暗区中的小洞（一个象素）；

$G2$ 填补直线段（线宽大于 1 象素）上的缺口。那么必须考查图 2 所示的 1邻域。

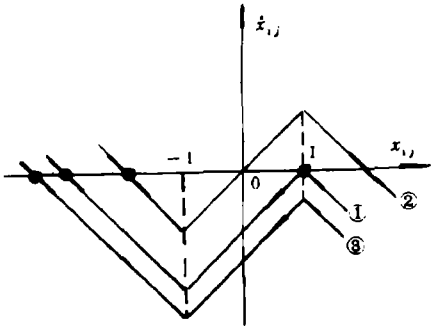

图 1　u 不同取值时 x_{ij} 的相轨迹

$C_{i-1,j-1}$	$C_{i-1,j}$	$C_{i-1,j+1}$
$C_{i,j-1}$	C_{ij}	$C_{i,j+1}$
$C_{i+1,j-1}$	$C_{i+1,j}$	$C_{i+1,j+1}$

图 2　C_{ij} 的 1邻域系统

$G1$ 和 $G2$ 可用以下布尔表达式表示：

$$Y_{ij}=u_{ij}\vee u_{i-1,j}\wedge u_{i+1,j}\wedge(u_{i,j-1}\vee u_{i,j+1})\vee u_{i,j-1}\wedge u_{i,j+1}\wedge(u_{i-1,j}\vee u_{i+1,j}),\qquad(8)$$

其中 $\vee$， $\wedge$ 分别表示逻辑“或”和逻辑“与”， Y_{ij} 表示 C_{ij} 的最终逻辑输出，由输出 y_{ij} 实现，并满足 $R1$ 和 $R2$。为便于 CNN 实现，(8) 式改写为

$$\begin{aligned} Y_{ij} &= u_{ij} \vee (u_{i-1,j} \wedge u_{i+1,j} \wedge u_{i,j-1}) \vee (u_{i-1,j} \wedge u_{i+1,j} \wedge u_{i,j+1}) \\ &\quad \vee (u_{i,j-1} \wedge u_{i,j+1} \wedge u_{i-1,j}) \vee (u_{i,j-1} \wedge u_{i,j+1} \wedge u_{i+1,j}) \\ &\triangleq u_{ij} \vee u_1 \vee u_2 \vee u_3 \vee u_4, \end{aligned} \tag{9}$$

其中

$$u_1 = u_{i-1,j} \wedge u_{i+1,j} \wedge u_{i,j-1}; \quad u_2 = u_{i-1,j} \wedge u_{i+1,j} \wedge u_{i,j+1};$$
$$u_3 = u_{i,j-1} \wedge u_{i,j+1} \wedge u_{i-1,j}; \quad u_4 = u_{i,j-1} \wedge u_{i,j+1} \wedge u_{i+1,j}.$$

令

$$s_1 = 3(u_{i,j-1} + u_{i,j+1} + u_{i-1,j} + u_{i+1,j}). \tag{10}$$

不难看出，若 u_1， u_2， u_3， u_4 中至少有一个为 T，则 $s_1 = 9$ 或 12；否则， $s_1 = 0$， 3 或 6。设 a， b 是两个待定常数，令

$$u = a \cdot u_{ij} + s_1 + b, \tag{11}$$

由 $F1$ 和 $F2$ 易知，若 $Y_{ij} = T$，且 $u_{ij} = 1$，则对于 s_1 的一切可能值下式成立：

$$u = a + s_1 + b > 0. \tag{12}$$

若 $Y_{ij} = T$，且 $u_{ij} = 0$，则有

$$u = s_1 + b > 0, \quad s_1 = 9\text{或}12. \tag{13}$$

若 $Y_{ij} = F$，那么必有 $u_{ij} = 0$， $s_1 = 0$， 3 或 6，且有

$$u = s_1 + b \leq 0, \quad s_1 = 0, 3\text{或}6. \tag{14}$$

综合 (12)， (13)， (14) 三式，我们得到 $-9 < b \leq -6$， $a > -b$。为简便起见，取 $b = -6, a = 7$。因此我们可以得到满足 (9) 式的 CNN 算法，由 (7) 和 (11) 两式知其状态方程为

$$\begin{aligned} \dot{x}_{ij} &= (-x_{ij} + 2y_{ij} - 1) + 7u_{ij} + 3(u_{i,j-1} + u_{i,j+1} + u_{i-1,j} + u_{i+1,j}) - 6 \\ &= (-x_{ij} + 2y_{ij} - 7) + B * u_{ij}, \quad x_{ij}(0) = 1, \end{aligned} \tag{15}$$

其中

$$B = \begin{array}{|c|c|c|} \hline & 3 & \\ \hline 3 & 7 & 3 \\ \hline & 3 & \\ \hline \end{array}$$

3.2 恢复丢失的角点

恢复右上角点可用下列布尔函数实现：

$$Y_{ij1} = \overline{u}_{ij} \wedge (u_{i,j-1} \wedge u_{i+1,j-1} \wedge u_{i+1,j}) \wedge \overline{(u_{i-1,j-1} \vee u_{i-1,j} \vee u_{i-1,j+1} \vee u_{i,j+1} \vee u_{i+1,j+1})} \vee u_{ij}, \tag{16}$$

其中上横线表示逻辑反。同样，右下、左上和左下角点可用以下布尔函数恢复：

$$Y_{ij2} = \overline{u}_{ij} \wedge (u_{i-1,j-1} \wedge u_{i-1,j} \wedge u_{i,j-1}) \wedge \overline{(u_{i-1,j+1} \vee u_{i,j+1} \vee u_{i+1,j-1} \vee u_{i+1,j} \vee u_{i+1,j+1})} \vee u_{ij}, \tag{17}$$

$$Y_{ij3} = \overline{u}_{ij} \wedge (u_{i,j+1} \wedge u_{i+1,j} \wedge u_{i+1,j+1}) \wedge \overline{(u_{i-1,j-1} \vee u_{i-1,j} \vee u_{i-1,j+1} \vee u_{i,j-1} \vee u_{i+1,j-1})} \vee u_{ij}, \tag{18}$$

$$Y_{ij4} = \overline{u}_{ij} \wedge (u_{i-1,j} \wedge u_{i-1,j+1} \wedge u_{i,j+1}) \wedge \overline{(u_{i-1,j-1} \vee u_{i,j-1} \vee u_{i+1,j-1} \vee u_{i+1,j} \vee u_{i+1,j+1})} \vee u_{ij}. \tag{19}$$

在满足 $R1$ 和 $R2$ 的情况下，我们有 $\overline{u}_{ij} = 1 - u_{ij}$，其余类推，则由与第 3.1 节相同的讨论可知 (16) 式可由以下状态方程所对应的运算过程实现：

$$\begin{aligned}\dot{x}_{ij} &= (-x_{ij} + 2y_{ij} - 1) + (\overline{u}_{ij} + u_{i,j-1} + u_{i+1,j-1} + u_{i+1,j} + \overline{u}_{i-1,j-1} + \overline{u}_{i-1,j} \\ &\quad + \overline{u}_{i-1,j+1} + \overline{u}_{i,j+1} + \overline{u}_{i+1,j+1}) - 8 + 9u_{ij} \\ &= (-x_{ij} + 2y_{ij} - 3) + B * u_{ij}, \quad x_{ij}(0) = 1,\end{aligned} \tag{20}$$

其中

$$B_1 = \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline 1 & 8 & -1 \\ \hline 1 & 1 & -1 \\ \hline \end{array}$$

同样地，(17)，(18) 和 (19) 式可由与 (20) 式相同形式的状态方程所对应的运算过程来实现，只是控制模板分别取为

$$B_2 = \begin{array}{|c|c|c|}\hline 1 & 1 & -1 \\ \hline 1 & 8 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array}; \quad B_3 = \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline -1 & 8 & 1 \\ \hline -1 & 1 & 1 \\ \hline \end{array}; \quad B_4 = \begin{array}{|c|c|c|}\hline -1 & 1 & 1 \\ \hline -1 & 8 & 1 \\ \hline -1 & -1 & -1 \\ \hline \end{array}.$$

3.3 消除孤立点和直线上的小突起

在第 3.1 和 3.2 节我们用 CNN 实现了线性可分布尔函数的运算。在本节我们讨论如何用 CNN 实现线性不可分布尔函数。

对于二值图象，若要求消除孤立的黑点或直线上的小突起，则以下布尔函数可达到目的：

$$\begin{aligned}Y_{ij} &= u_{ij} \wedge [(u_{i-1,j-1} \vee u_{i-1,j} \vee u_{i,j-1}) \wedge (u_{i,j+1} \vee u_{i+1,j} \vee u_{i+1,j+1}) \\ &\quad \vee (u_{i-1,j} \vee u_{i-1,j+1} \vee u_{i,j+1}) \wedge (u_{i,j-1} \vee u_{i+1,j-1} \vee u_{i+1,j})] \\ &= u_{ij} \wedge [(U_1 \wedge U_2) \vee (U_3 \wedge U_4)],\end{aligned} \tag{21}$$

其中

$$U_1 = u_{i-1,j-1} \vee u_{i-1,j} \vee u_{i,j-1}; \qquad U_2 = u_{i,j+1} \vee u_{i+1,j} \vee u_{i+1,j+1};$$
$$U_3 = u_{i-1,j} \vee u_{i-1,j+1} \vee u_{i,j+1}; \quad U_4 = u_{i,j-1} \vee u_{i+1,j-1} \vee u_{i+1,j}.$$

(21) 式是线性不可分的，所以要用多层 CNN 来实现。但多层 CNN 的层间通信可能会成为 CNN 计算速度的瓶颈，所以本文采用以下方法来避免层间通信，以保证 CNN 的并行处理的优越性能得到充分发挥。

为图 2 所示的 1- 邻域系统中的每一个象素细胞配备若干计算细胞。计算细胞专门从事中间计算，其输入端和输出端均在芯片内，因而不会增加 CNN 芯片外接引脚数和通信的负担。这种扩展了的 1- 邻域系统称为象素 1- 邻域。图 3 是象素细胞 C_{ij} 的象素 1- 邻域，其中每个象素细胞配备有 $C_1 \sim C_4$ 共 4 个计算细胞。我们将用 CNN 的这种结构来计算 (21) 式。

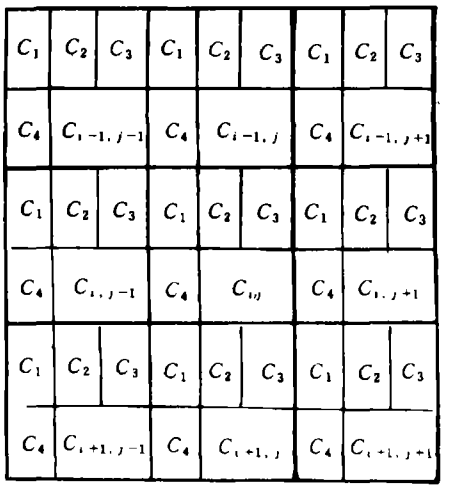

图 3 象素细胞 C_{ij} 的象素 1-邻域

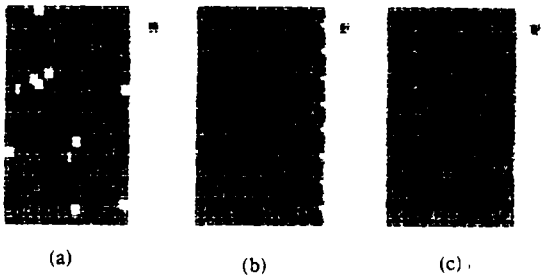

图 4 填补小孔和缺口 (a) 原始图象；(b) 0.25μs 的结果；(c) 0.5μs 的结果。

计算 (21) 式中的 Y_{i3} 要分 5 个部分进行：(1)C_1 计算 U_1 ；(2) C_2 计算 $U_1 \wedge U_2$ ；(3)C_3 计算 U_3 ；(4)C_4 计算 $U_3 \wedge U_4$ ；(5)C_{ij} 计算 Y_{ij} 。以下给出在象素细胞 $C_1 \sim C_4$ 和计算细胞 C_{ij} 在初态均为 1 的情况下完成上述任务的状态方程：

$$\dot{x}_1 = (-x_1 + 2y_1 - 1) + (u_{i-1,j-1} + u_{i-1,j} + u_{i,j-1}), \tag{22}$$
$$\dot{x}_2 = (-x_2 + 2y_2 - 4) + (u_{i,j+1} + u_{i+1,j} + u_{i+1,j+1}) + 3y_1, \tag{23}$$
$$\dot{x}_3 = (-x_3 + 2y_3 - 1) + (u_{i-1,j} + u_{i-1,j+1} + u_{i,j+1}), \tag{24}$$
$$\dot{x}_4 = (-x_4 + 2y_4 - 4) + (u_{i,j-1} + u_{i+1,j-1} + u_{i+1,j}) + 3y_3, \tag{25}$$
$$\dot{x}_{ij} = (-x_{ij} + 2y_{ij} - 1) + (y_2 + y_4 - 2) + 3u_{ij}. \tag{26}$$

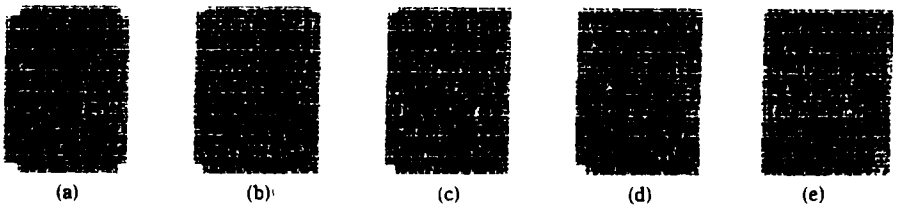

图 5　恢复丢失的角点
(a) 原始图象；(b) 恢复右下角（0.5μs）；(c) 恢复右上角（0.5μs）；
(d) 恢复左上角（0.5μs）；(e) 恢复左下角（0.5μs）。

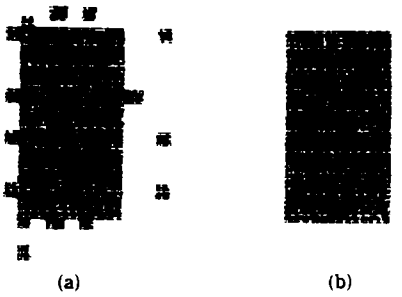

图 6　消除孤立点和直线上的小突起
(a) 原始图象；(b) 1μs 的结果。

4 平滑的例子

在 CNN 的电路实现中，其状态变量、输入、输出均为一特定的电压值。故 (6) 式可改写为

$$C \cdot \frac{dv_{x_{i,j}}(t)}{dt} = -\frac{1}{R_x} v_{x_{i,j}}(t) + \tilde{A} * v_{y_{i,j}}(t) + \tilde{B} * v_{u_{i,j}} + I;\ i \in \underline{M},\ j \in \underline{N}. \tag{27}$$

取 $C = 10^{-9}\mathrm{F}$，$R_x = 10^3\Omega$，$\tilde{A} = 10^{-3}\Omega^{-1}A$，$\tilde{B} = 10^{-3}\Omega^{-1}B$。由于 $\tilde{A}$，$\tilde{B}$ 分别是 A，B 乘了一个小于 1 的数得到的，故原收敛域和收敛条件均没变。图 4，图 5，图 6 分别给出了填补小孔和缺口。恢复丢失的角点以及消除孤立点和直线上的小突起的仿真结果。所用时间分别为 0.5 μs，$4 \times 0.5\mu$s，1μs。在用硬件实现时，增多象素的数目并不延长处理时间，只是要增加细胞的数目。效果是令人满意的。

5 结　论

本文利用 CNN 实现了线性可分和线性不可分的逻辑函数，为 CNN 的广泛应用提供了一种新的权重设计方法。同逻辑门阵列相比，CNN 逻辑运算最诱人的潜力是可以方便地推广到多值逻辑和连续逻辑（即模糊逻辑）运算中去，而不需要扩展更多的字节 [9]。

参 考 文 献

[1] Chua L O, Yang L. IEEE Trans. on CAS, 1988, CAS-35(10): 1257–1272.

[2] Chua L O, Yang L. IEEE Trans. on CAS, 1988, CAS-35(10): 1273–1290.

[3] Proceedings of the First IEEE International Workshop on Cellular Neural Networks and Their Applications. Budapest: 1990.

[4] Proceeding of the Second IEEE International Workshop on Cellular Neural Networks and Their Applications. Munich: 1992.

[5] Special Issue on Cellular Neural Networks. Int. J Circuit Theory and Appl., 1992, 20(8):

[6] Special Issue on Cellular Neural Networks: Theory. IEEE Trans. on CAS, 1993, CAS-I, 40(3):

[7] Special Issue on Cellular Neural networks: Applications. IEEE Trans. on CAS, 1993, CAS-II, 40(3):

[8] Fu K S, *et al.* Robotics: Control, Sensing, Vision and Intelligence. New York, North-Holland: 1985.

[9] Yager R R, Zadeh L A. An Introduction to Fuzzy Logic Applications in Intelligent Systems, Boston: Kluwer Academic Publishers, 1992, 1–25.

SMOOTHING ALGORITHMS FOR BINARY IMAGE USING CELLULAR NEURAL NETWORKS

Yang Tao

(*Shanghai University, Shanghai* 200072)

Abstract The piecewise linear saturation characteristics of cell in a cellular neural network(CNN) and phase plane analysis method are used to realize linear separable and nonseparable Boolean expressions. And the principle is also used to achieve some CNN smoothing algorithms for binary images.

Key words Cellular neural network, Boolean expression, Binary image, Smoothing algorithm

杨 涛： 男， 1970 年生，助教，现从事 CNN 和 Chua 氏电路的基本理论及其应用研究。