FUZZY CELLULAR NEURAL NETWORK

by

Tao Yang, Lin-Bao Yang, and Chun-Mei Yang

Memorandum No. UCB/ERL M97/61

3 September 1997

FUZZY CELLULAR NEURAL NETWORK

by

Tao Yang, Lin-Bao Yang, and Chun-Mei Yang

Memorandum No. UCB/ERL M97/61

3 September 1997

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Fuzzy Cellular Neural Networks

Tao Yang
Department of Electrical Engineering and Computer Sciences,
University of California at Berkeley,
Berkeley, CA 94720, U.S.A.
Tel: (510)-642-5311 Fax: (510)-643-8869
Email: taoyang@fred.eecs.berkeley.edu
http://trixie.eecs.berkeley.edu/~taoyang/cv-taoyang.html

Lin-Bao Yang
University of E-Zhou,
E-Zhou, Hubei, 436000,
P.R. China

Chun-Mei Yang
China Construction Bank Songjiang Branch,
and Everbeauty Houseware (Shanghai) Co. Ltd.,
Shanghai, 201612, P.R. China

September 3, 1997

77
78
79
88
89
105
106
143
157

To:

The Great China

And

Our Families

Contents

List of Figures

Acknowledgments

First, the authors would like to thank the cultural background of the ancient China, from which they learned how to handle the poverty, lack of equipments, books and references and all the other difficulties to become excellent in the first-rate level in the world. They also want to thank the Chinese language system which is qualitatively different to almost all the other language systems in the Earth. The analog nature of Chinese makes it easy to be used as a *fuzzy* tool for simulating the complexity of the world.

We also want to thank Professor L. A. Zadeh, who is well-known as the father of fuzzy theory, of University of California at Berkeley. Although none of the authors had the chance to learn fuzzy theory in personal under the guidance of Professor Zadeh, the beauty of fuzzy theory for coping with the unavoidable complexity of substantial world is the best teacher. While the first author stay in University of California at Berkeley, he eventually have chance to know Professor Zadeh in personal and attend his seminars. In one of his seminar, he presented the computing-with-word nature of fuzzy theory impressed the first author so much.

The authors are grateful to many of their colleges in China. In particular, they would like to express the gratitude to Prof. Hong-Nian Yang of Wuhan University for his excellent guidance of the ancient Chinese and the root of the Chinese nationality. They would like to thank Prof. Xiu-Ping Yang of Huazhong Agricultural University for her invaluable discussion on the biological neural networks. The first author would like to thank Prof. Hong-Chang Zhou, Prof. Zuo-Tao Xie, Prof. Qi-Di Wu, Dr. Wei-Sheng Xu and Ms. Fei Qiao of Tongji University, Shanghai, China, Prof. Bo-Shi Chen, Prof. Kan-Yu Zhang, Prof. Zhao-Min Zhou and Mr. Xiao-Yong Huang of Shanghai University for their academic support and discussion. The first author would like to thank Mr. An-Zhuo Xu, which was a classmate of the first author, of Shanghai Golden Card Company for his generous support of computation and communication equipments when the first author did his researches unemployedly and without any founding in China in 1993.

They also grateful to many colleagues abroad. In particular, they would like to express their gratitude to Prof. L. O. Chua of University of California at Berkeley, for his invaluable guidances and for sending them lots of reprints. They would like to thank Prof. J.A. Nossek of Technical university of Munich, for invaluable discussing, encouragement and for sending them one copy of proceedings of CNNA'92. They also would like to thank Prof. T. Roska of Computer and Automation Institute of the Hungarian Academy of Science, for his comments and encouragements and for sending them lots of technical reports and reprints.

We also want to thank the editors and reviewers of *IEEE Transactions on Circuits and Systems-I*, *IEEE Transactions on Fuzzy Systems*, *International Journal of Circuit Theory and Applications*, and *Journal of Circuits, Systems and Computers* for their invaluable comments.

This book was prepared by the authors using the LaTeX document processing system.

The figures were generated using IDRAW and MATLAB and included into the book using Psfig.

Preface

The field of *cellular neural networks* (CNN) is very attractive. The CNN spreads out the ideal of local connectedness and emergent computation. Encountering with the huge body of literatures of this field one can find that CNN is mostly nourished from two main fields: One is the concept of *cellular automata* and the other is *artificial neural networks.* As an inter-discipline product, the CNN mainly comes from the tendency of finding a neural network which can be easily implemented by the state-of-the-art VLSI technique and finding a high speed parallel super-computing platform based on *ana-logic computation.*

Since its heavy color of circuit implementation and the inspiration to the first generation CNN chip and the possible applications to the image processing field, lot of successes misleaded the CNN community to focus on developing a parallel image processing chip using ana-logic paradigm. Since 1988, very little had been done to consider the inner philosophic ideal that the CNN concept contributed to the science. While the western circuit community are facing the successes of new concept, new principle, new applications, new chips and most important, new papers on *conventional CNN* whose main principles were mainly built up in the two pioneer papers of Chua and Yang. During 1988 to 1996, in the western world, the CNN is restricted in the range of low-level CNN, it seems that no one try to unified the CNN universe from the non-circuitry point of view. And all the generalizations were restricted in the low-level of *artificial intelligence*(AI). From the mathematical point of view, the development of CNN in western world during these 8 years emphasized the function instead of functional, implementation instead of generalization.

However, the case was totally different in China where instruments and computational power were too weak to support VLSI design and large scale simulation of image processing, the research of CNN was focused on the systematic levels. In early 1993, a CNN group leaded by Tao Yang began to develop a new CNN structure which could embed fuzzy logic. This structure was latter found to be a type-II *fuzzy CNN*(FCNN) , which then was embedded into a more general FCNN framework. Until early 1995, all the branches of FCNN had been set up and lots of applications of FCNN to mathematical morphology, image processing and AI had also been founded.

This monograph contained almost all the results that presented by this group during 1993 to 1996. Since almost all the early results were presented in Chinese, none of them was known in the Western World until early 1995 when Tao Yang went to University of California at Berkeley and shared his fuzzy ideas with the CNN group there.

This monograph is not only the first book of FCNN but also the first collection of all the results of FCNN so far. And most of them are brand new to western CNN community. We hope that the new ideas in this book can be shared by the whole Western CNN community in its English form.

However, FCNN had been overcome lots of unbelievable obstacles both in China and Western world. Sometimes, the only straw for us to clutch at in the ocean of criticisms is

the noble model of Prof. L. A. Zadeh who himself also experienced what we did. We would like to cite what he felt to express what we did:

"For the most part, however, what I experienced was skepticism and hostility. Even through I had a thick skin, there were occasions when I had to control my emotions." [L. A. Zadeh, "The evolution of systems analysis and control: A personal perspective", *IEEE Control Systems*, vol.16, no.3, June 1996, pp. 95-98].

However, even we have thicker skins than that of Prof. Zadeh's, sometimes we find that we can not control our emotions by using any fuzzy or non-fuzzy control strategies.

October, 1996

Chapter 1

Introduction

In this chapter we give an overview of the state-of-the-art of *cellular neural network*(CNN). This overview will give the readers who are not very familiar with CNN all the elementary concepts, expressions and symbols used in this field. For experts in this field, we will show why from the systematic point of view, the structure of discrete-time CNN(DTCNN) is more important than the other CNN structures appeared during the same period, e.g., Delay-type CNN(DCNN), CNN with nonlinear synaptic laws(NCNN), and chaotic CNN(CCNN). We also show that CNN universal machine(CNNUM) is a secondary level concept which can not be mixed with the concepts of CNN universe in the first level (the elementary level) which consists of all the existed conventional CNN and FCNN. The purpose of this chapter is not to survey all the existed CNN literatures but to show that the existed CNN map from a systematic point of view instead of only a circuitry point of view.

1.1 The State-of-the-art of CNN

CNN is a kind of locally connected network. It originally stemmed from the concept of *cellular automata*(CA)[101, 322, 83, 227, 105] and *artificial neural network* (ANN)[125, 31, 294, 68, 192, 399, 173, 157]. The local connectedness is the most significant property of CNN, which is a significant difference to other ANN. The continuous dynamics distinguishes CNN from CA. The local connectedness restricts the ability of CNN for solving lots of global problems which can not be decomposed into local components. However, the local property has its advantage such as easy implementation by using VLSI technique and efficiency for local problems. Another important aspect of this structure is that it provides a paradigm for studying emergent computation[87] and the relevant topics, e.g., artificial life(AL)[340, 177, 320], besides some other models such as CA and spin glass model[190].

CNN was first introduced in two twin papers[47, 46] in October 1988. This decade viewed a rapid growth of this field. So far, there had been published one book[271], four conference proceedings[242, 243, 244, 245] and lots of special issues in different international journals[204, 205, 207, 206, 203].

Since its first introduction in 1988 [47, 46] there had been developed lots of different branches of this field. The main branches are motivated by engineering applications and the results from biological sciences, especially those results from retina research because retina and CNN structures share lots of common properties such as layer structures and local connectedness[136]. The engineering field keeps trying to find the easy implementation of different CNN structures by different techniques including VLSI, optical component and

quantum dot techniques. So far, CNN is mainly used in image processing because the two-dimensional (2D) array of cells is directly connected with the digital image which is normally a 2D array of small units called *pixels*.

Since image processing is well-studied and is one of the most challenging field in artificial intelligence and signal processing, the relation between it and CNN is two folds. First, there exist lots of existed results from the linear and nonlinear image processing which can be directly mapped into CNN structures. Almost all the early work and lots of the recent work are concentrated into this direction[209, 311, 67, 53, 57, 321, 305, 214, 209, 316, 144, 300, 55, 220, 93, 262, 396, 195, 169, 268, 336, 44, 229, 299, 310, 251, 359]. Almost all the results along this direction employ computational CNN structures which only map the existed image processing algorithms into the weight arrays of CNN. The new tendency in the CNN research is the embedding of nonlinear synaptic laws into the linear CNN framework. This is directly motivated by nonlinear image processing techniques. Again, lots of work in this direction is focused on computational CNN[250]. On the other hand, the CNN structures also contribute to image processing with new computational arrays and learning algorithms. However, this is a brand new direction, lots of work should be done and this direction is not yet well-understood because there is no existed paradigm as image processing can be used.

Although there may also exist some other techniques such as optical computing array [25, 90, 306, 308, 89, 139] and quantum dot computing array, so far, VLSI technique is the only way to implement CNN[334, 112, 330, 84, 17, 138, 331, 113, 175, 235, 41, 217, 211, 281, 284, 76, 174, 81, 137, 154, 274, 199, 16, 253, 163, 162, 20, 237, 167, 283, 296, 160, 79, 22, 69, 338, 159, 176, 297, 298, 232, 80, 6, 82, 38]. Lots of VLSI implementation are designed for special task with fixed templates[332, 347, 254, 56, 168, 313], while the other try to design the programmable chip in which the templates are adaptable[24, 36, 62, 9, 114, 110, 111, 109, 213, 212, 164, 252, 280, 285, 70, 8, 45, 248, 288, 75, 287, 61, 37, 161, 58, 286, 210, 182, 71]. And some software and hardware accelerating board were also developed[333, 270, 261, 259, 171, 260, 178, 179, 165, 72, 236]. In the VLSI based implementation, the digital ones and the analog ones were both presented. The input and output problems can be solved by using the embedded optical sensor and optical interface in every cell[77, 78, 313, 314, 297, 309, 140, 21, 19]. However, the state-of-the-art chips are not better than the existed programmable DSP chips which had been developed for many years and widely used today. But we can not overlook the potential power in the future CNN chips. This is one of the main reasons why the study of CNN is becoming more and more attractive.

In a word, the state-of-the-art CNN structures emphasize the VLSI implementation and applications to image processing. There almost exist no work considering the global CNN universe from a systematic point of view.

1.2 Structures of the conventional CNN

1.2.1 Elementary CNN

Definition 1

The elementary processor in a CNN array is called *cell*.

Remark: A cell is the most elementary unit in a CNN array, which builds up the structure of a CNN. We denote a cell respectively by C_i, C_{ij} and C_{ijk} in a 1D, 2D and 3D CNN array.

Definition 2

Let $r \in \mathbf{N}$ be a positive integer, the *r-neighborhood system*, $N_r(i)$, $N_r(ij)$ and $N_r(ijk)$ of a *center cell* C_i, C_{ij}, C_{ijk} in a 1D, 2D and 3D CNN, is respectively defined by

$$N_r(i) \triangleq \{C_j : \max(d_1(j-i)) \leq r\} \tag{1.1}$$

where $d_1(\cdot)$ is a distance defined in $\mathbf{R}$.

$$N_r(ij) \triangleq \{C_{kl} : \max(d_2(k-i, l-j)) \leq r\} \tag{1.2}$$

where $d_2(.,.)$ is a distance defined in $\mathbf{R}^2$.

$$N_r(ijk) \triangleq \{C_{pqr} : \max(d_3(p-i, q-j, r-k)) \leq r\} \tag{1.3}$$

where $d_3(.,.,.)$ is a distance defined in $\mathbf{R}^3$.

Remark: r-neighborhood system is the core definition for describing the local connectedness of a CNN. This concept only define the possible longest synaptic weight that may affect a center cell. It does not give the structure and the connections between cells. Although we can also define the neighborhood system in $\mathbf{R}^n$, $n > 3$, in view of the near future techniques, we find $n \leq 3$ is enough.

We then define the dynamics of a cell in CNN. Here, only the 2D case is given. The 1D and 2D cases are similar.

Definition 3

An elementary CNN is an $M \times N$ array which consists of M rows and N columns cells, every cell C_{ij} is given by the following equations:

1. State equationcell, state equation

$$\begin{aligned} C\frac{dx_{ij}(t)}{dt} &= -\frac{1}{R_x}x_{ij}(t) + \sum_{C_{kl}\in N_r(ij)} A(i,j;k,l)y_{kl}(t) \\ &\quad + \sum_{C_{kl}\in N_r(ij)} B(i,j;k,l)u_{kl}(t) + I_{ij}(t) \end{aligned} \tag{1.4}$$

where x_{ij}, u_{ij} and y_{ij} denote the state variable, input and output of cell C_{ij}, respectively. $A(i,j;k,l)$ and $B(i,j;k,l)$ denote the *feedback* and *feed-forward* synaptic weights between cells C_{ij} and C_{kl}, respectively. $I_{ij}(t)$ is the *bias* (also called *threshold*) of cell C_{ij}, which may be static, time-varying, space-invariant or space-varying. $C > 0$ and $R_x > 0$ are the values of the capacitor and the resistor, respectively [1]. y_{ij} is given by

2. Output equation

$$y_{ij} = f_y(x_{ij}) = \frac{1}{2}[|x_{ij}+1| - |x_{ij}-1|] \tag{1.5}$$

u_{ij} is given by

[1] Of course, a cell is not necessary to be a first-order dynamic system, it can also be represented by a high-order ordinary differential equation or a functional differential equation.

3. Input equation

$$u_{ij} = f_u(E_{ij}) \tag{1.6}$$

where E_{ij} is the detected signal, e.g, the intensity of light detected by the embedded optical sensor in cell C_{ij}. In the elementary CNN, $f_u(\cdot)$ is used to normalize the detected signal to a proper range.

4. Initial condition $x_{ij}(0)$.

5. Boundary condition.

Remark: In a VLSI implementation of CNN, u_{ij}, x_{ij}, and y_{ij} are three voltages. $I_{ij}(t)$ is a bias current.

If the CNN is space-invariant, then Eq.(1.4) can be written into a 2D convolution form as[46]

$$C\frac{dx_{ij}(t)}{dt} = -\frac{1}{R_x}x_{ij}(t) + A * y_{ij}(t) + B * u_{ij}(t) + I(t) \tag{1.7}$$

where "*" denotes a 2D convolution. A is called *feedback template* and B is called *feed-forward template.*

1.2.2 Different CNN structures

Since its invention[47, 46], different CNN structures had been proposed for different applications and from different biological models.

In the motion related applications[266], time delays are introduced into the CNN structure and give a *delay-type CNN* (DCNN) which is defined by[263, 266]

$$\begin{aligned} C_x\frac{dx_{ij}(t)}{dt} = & -\frac{1}{R_x}x_{ij}(t) + \sum_{C_{kl}\in N_r(ij)} A(i,j;k,l)y_{kl}(t) \\ & + \sum_{C_{kl}\in N_r(ij)} A^\tau(i,j;k,l)y_{kl}(t-\tau) \\ & + \sum_{C_{kl}\in N_r(ij)} B^\tau(i,j;k,l)u_{kl}(t-\tau) \\ & + \sum_{C_{kl}\in N_r(ij)} B(i,j;k,l)u_{kl}(t) + I \end{aligned} \tag{1.8}$$

where $\tau \in \mathbf{R}^+$ is called *time-delay.* DCNN was proved to be essential in motion related CNN-applications [266]. There exists some theoretical results on the stability of DCNN in [86, 98, 388, 135, 273, 50, 272, 48]. Since DCNN is governed by a set of functional differential equation (FDE), some complex phenomena, e.g., chaos, were observed even only small number of cells were used[49, 97], and some results of predicting the chaotic sequence generated by chaotic DCNN is presented in [99].

Since only linear synaptic weights are not enough to deal with some image processing tasks where non-linear properties are embedded, the *CNN with nonlinear synaptic laws*(NCNN) were introduced[86]. A general NCNN can be given by

$$\begin{aligned} \dot{x}_{ij} = & A\Big(\mathbf{x}(N_r(ij)), \mathbf{u}(N_r(ij)), t\Big) * x_{ij} + B\Big(\mathbf{x}(N_r(ij)), \mathbf{u}(N_r(ij)), t\Big) * x_{ij} \\ & + I\Big(\mathbf{x}(N_r(ij)), \mathbf{u}(N_r(ij)), t\Big) \end{aligned} \tag{1.9}$$

where $\mathbf{x}(N_r(ij))$ and $\mathbf{u}(N_r(ij))$ denote all the state variables and inputs within $N_r(ij)$, respectively. In this case, synaptic laws are functions of time, state variables and inputs within the neighborhood system.

Since CNN is a nonlinear dynamical array, there also found some complex phenomena, e.g., chaos, even when a few cells is used[317, 404, 403, 401, 14, 400, 405, 402, 12]. The strange non-chaotic attractor is also observed in 2-cell quasi-periodically forced CNN. If an array of cells is used, the hyper-chaos also emerges[13]. Since the control of chaos is also a very active field in view of its possible applications to spread spectrum communication, secure communication and measurement improvement, some control methods are also used to control the chaos generated by CNN[94, 150].

There also exists some other kinds of CNN structures such as *chaotic CNN*(CCNN) where every cell is a chaotic dynamic system[107, 42] which can be used to model some kinds of emergent computation behaviors and be used to simulate some wave and pattern formation phenomena in active medium[228, 152, 43, 42]. In CCNN arrays some nonlinear dynamic behaviors such as synchronization[151], cluttering and cooperative phenomena are also found[201, 202]. The existed results of CCNN consist of two main branches. One is try to study how to used the elementary CNN to generate chaotic signal and directed to the relevant applications[107, 194, 42]. The other branch is to study how to use chaotic elements as elementary cells to model spatio-temporal chaotic processes[228, 13, 151, 202, 152].

Multi-layer CNN(MCNN) uses more than one layer CNN's to perform a single task[191, 47, 118].

The *CNN universal machine* (CNNUM)[265, 264, 267, 257, 258, 54, 307] is not an elementary CNN structure, it is a platform for integrating the flow of CNN operations, we do not discuss it in this book, the interested readers are referred to [265, 264, 267, 257, 258, 54, 307]. However, the CNNUM is an important tool for organizing different kinds of CNN structures to perform a more complicated task that a single CNN can not do. The CNNUM structure sometimes can also be used to solve some global problem which are difficult to be decomposed. In fact, the CNNUM had been proved to be as universal as a Turing Machine[54]. Since it is only a platform for CNN operations, any kind of CNN should be included in the core of this platform, including DTCNN[307] and FCNN[351, 352]. However, the state-of-the-art CNNUM has only the elementary CNN core[267], this platform needs further improvement.

1.2.3 Discrete-time CNN

Since analog and digital, continuous and discrete-time methods represent the state-of-the-art implementation of information processing processes, it is very natural to modify the original continuous CNN model into a discrete-time structure[120]. A *discrete-time CNN*(DTCNN) is defined by the following discrete dynamic equations:

1. State equation

$$x_{ij}(k) = \sum_{C_{kl}\in N_r(i,j)} A(i,j;k,l)y_{kl}(k) + \sum_{C_{kl}\in N_r(i,j)} B(i,j;k,l)u_{kl} + I \quad ,1 \le i \le M, 1 \le j \le N \tag{1.10}$$

2. Output equation

$$y_{ij}(k) = F(x_{ij}(k-1)), 1 \le i \le M, 1 \le j \le N \tag{1.11}$$

3. Initial condition

$$x_{ij}(0), 1 \leq i \leq M, 1 \leq j \leq N \tag{1.12}$$

4. Input equation

5. Boundary condition

$x_{ij}(t) \in \mathbf{R}$, $y_{ij}(t) \in \{0,1\}$ and $u_{ij} \in \{0,1\}$ are state, output and input of cell C_{ij}, respectively. $I \in \mathbf{R}$ is the bias. $k \in \mathbf{Z}$ is the discrete time. $F(\cdot)$ is the nonlinear output property of a cell, it should be easy to be implemented by using VLSI techniques.

Similarly, we can rewrite Eq.(1.10) into the following form:

$$x_{ij}(k) = A * y_{ij}(k) + B * u_{ij} + I \tag{1.13}$$

One can see that the structure of DTCNN is close to that of a CA. One advantage of DTCNN over the elementary CNN is that it has both binary input and output which makes the connection between two different chips very easy. And the DTCNN has a stronger robustness than elementary CNN.

Although DTCNN may be viewed as a discrete form of CNN, it is not necessary to think that CNN can do all what a DTCNN can do. For example, let we consider the following DTCNN defined by[349]

1. State equation

$$x_{ij}(k) = A * y_{ij}(k), 1 \leq i \leq M, 1 \leq j \leq N \tag{1.14}$$

2. Output equation

$$y_{ij}(k+1) = f(A * y_{ij}(k)) \oplus y_{ij}(k-1), 1 \leq i \leq M, 1 \leq j \leq N \tag{1.15}$$

where $\oplus$ is the exclusive OR(XOR).

3. Cell nonlinearity

$$f(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{1.16}$$

4. Initial condition

$$y_{ij}(-1), y_{ij}(0), 1 \leq i \leq M, 1 \leq j \leq N \tag{1.17}$$

One can see that the nonlinearity of the above cell is different from the original one in [120], we used this nonlinearity to guarantee that the outputs of this DTCNN can be represented by 0-1 logic. The output equation is also different from that in the original paper[120] and needs a 2-bits local digital memory for storing the previous outputs. Since XOR is reversible, from Eq.(1.15) we have:

$$y_{ij}(k-1) = f(A * y_{ij}(k)) \oplus y_{ij}(k+1), 1 \leq i \leq M, 1 \leq j \leq N \tag{1.18}$$

,which means that if we know the output $y_{ij}(k+1)$ at $k+1$ and the output $y_{ij}(k)$ at k then we can find the output $y_{ij}(k-1)$ at $k-1$. This kind of DTCNN is reversible(RDTCNN), which has no corresponding CNN structures.

Although it had been shown that a CNNUM program can model the behavior of any DTCNN, we should note that the invention of DTCNN played a very important role in CNN history. First, its importance comes from the introduction of discrete-time dynamics into the CNN framework. It shows us the possibility that in CNN universe, a cell may have different kinds of dynamics except for continuous dynamics. The invention of DTCNN reminds us that cell dynamics should evolve in the *event space*, which may include the *time axis* and any *event sequence*. This is the direct motivation for the unification of CNN structure as will be shown in Sec. 2.1. Second, if we break the restriction of implementation consideration, DTCNN directly enlightens the concept of "structural dynamics" of neighborhood system, which is a most important concept in CNN universe. This concept will be discussed in Sec. 2.1.

There exists some stability results of DTCNN in [120, 128]. Some other theoretical results can be found in[304, 231, 230, 92, 88, 184]. The learning algorithm based on DTCNN is presented in [122]. DTCNN can also be used in associative memories[34, 32], image coding and decoding[145, 143], image thinning[134, 132] and other applications[124, 216]. Some DTCNN were designed to have continuous output[52, 345], this kind of DTCNN emphasized the discrete nature in time instead of both in time and input/output. There also exists lots of work on hardware implementation of DTCNN[215, 234, 346, 133, 121, 119, 7]. Since DTCNN are nonlinear discrete-time dynamic systems, complex phenomena, e.g., chaos, can also be easily found[233, 345]. The multi-layer DTCNN structure can be found in[118].

1.3 Notes

Since CNN can also function as local connected learning networks, lots of literatures concerning with learning algorithms of CNN were presented[198, 33, 318, 104, 127, 197, 2, 327, 186, 185, 108, 290, 191, 18, 116, 291, 106, 183, 315, 319, 247, 328, 3, 256, 170, 326, 312, 406, 123, 329, 407, 225]. Since almost all these learning algorithms are similar to those used in ANN, we do not discuss them here. The interested readers are refereed to the above references.

Chapter 2

Fuzzy CNN

In this Chapter, we present the principles and structures of FCNN in the framework of a unified CNN structure. We first present the unified CNN structure from a systematic point of view, then FCNN is developed from this unified CNN structure by introducing fuzzy set theory into it.

Fuzzy set theory provides an inference methodology which approximates human reasoning capabilities and can be applied to knowledge-based system[393]. It provides a mathematical strength to capture the uncertainties associated with human cognitive processes, e.g., thinking and reasoning. Also, it provides a mathematical methodology to model linguistic statements and knowledge.

While fuzzy theory provides an inference mechanism under cognitive uncertainty, CNN structures offers advantages such as learning, adaptation, fault-tolerance and parallelism. However, it seems that a CNN cell which is typically of 3×3- or 5×5-cell neighborhood system is a poor model of a real neuron which is typically of thousands of synapses[4, 153, 172, 241]. There exist some uncertainties in CNN synaptic weights. On the other hand, the input information (e.g., images) may also bear some fuzziness which comes from sensing, transmitting and processing. The information flow propagating in a CNN is mostly like a fuzzy process if inputs and/or synaptic weights are fuzzy.

The conventional CNN has a poor interface to the knowledge of human expert which is represented by fuzzy IF-THEN rules and the experience of human expert which is described by linguistic statements. From above we know that it is very necessary to integrate fuzzy set theory with the CNN paradigm and give birth to a new concept called *fuzzy CNN*(FCNN).

The concepts of FCNN are reasonable extensions of CNN from classical sets to fuzzy sets. The principles of FCNN is based on the uncertainties in human cognitive processes and in modeling neural systems.

For purpose of giving the reader an overview of the position of FCNN in the CNN universe, we first give the unified CNN structure and then we discuss FCNN as an important generalization of the unified CNN structure.

2.1 A unified CNN structure

Since CNN is an infant, it is hard to predict its potential structures for different applications. Also, it is hard to predict the possible structures that the future devices and new materials may implemented. The authors of [43] presented a kind of generalization of CNN structure which is a circuit based unification. Although the concept of CNN is developed based on

the circuit theory, it is indeed a much wider concept from the systematic point of view because it reflects some aspects of life systems where some biological cells also share the local connectedness nature. In this book, we unified the CNN in a systematic framework. We present a unified CNN structure which includes all the existed CNN structures as its subclasses and make FCNN as an important generalization, which embeds fuzzy dynamics and fuzzy information flow (linguistic flow) into the unified CNN structure.

The most basic properties and principles of the unified CNN structure are local connectedness, dynamics and the concept of cell.

The *local connectedness* emphasizes not only the easy implementation but also the inner nature of lots of biological, physical and social phenomena which are reflected the new tendency in artificial intelligence and artificial life where a new concept of *decentralization* is proposed[340]. On the other hand, the local connectedness also provides us with a tool to imitate the emergent behaviors of life systems[87] besides CA.

The *dynamics* provides the unified CNN structure with the ability of self-organization, learning ability and the basic condition for emergent computation. Without dynamics, the CNN is only a computational array which does not have any significant difference from the existed parallel image processing chip or parallel signal processing chip.

The *cell* is the elementary unit of CNN, which is a trade-off between implementation and function. A simple cell structure provides a bigger cell population and larger neighborhood system but a relatively simple cell function. Sometimes, since the emergent computation is hard to deal with, one may want to embed more controllable (programmable) functions into a cell such that the control of the CNN behavior becomes easier. But a more complicated cell decreases the cell population and the size of neighborhood system or makes the implementation very difficult.

It is easy to understand that cell may be any kind of individual. Local connectedness can be measured by a "distance". The new concept we used to define a unified CNN structure is the dynamics which consists of cell dynamics, neighborhood system dynamics and synaptic weight dynamics.

1. Cell dynamics

$$x_i(\sqcup) \overset{\heartsuit}{\longleftarrow} \mathcal{D}_x\Big(x(N_i(\sqcup)), u(N_i(\sqcup)), I(\sqcup), \sqcup\Big) \circ S_i(\sqcup) \tag{2.1}$$

where $x(N_i(\sqcup))$ and $u(N_i(\sqcup))$ denote all the states and inputs within the neighborhood system $N_i(\sqcup)$. "$\sqcup$" is a generalized time, which may denote the order of an arbitrary event sequence, e.g., discrete event sequences. Symbol "$\circ$" denotes a combining relation between *information flow* ($\mathcal{D}_x(\cdot,\cdot,\cdot)$) and *structural flow* ($S_i(\cdot)$). Symbol "$\overset{\heartsuit}{\longleftarrow}$" [1] denotes an arbitrary relationship from right hand side to the left hand side. This relation may be defined by an ODE, a PDE, an FDE, a discrete-time dynamics, a linguistic dynamics, a conceptual dynamics, a functional dynamics or any other dynamics.

2. Neighborhood system dynamics

$$N_i(\sqcup) \overset{\heartsuit}{\longleftarrow} \mathcal{D}_N\Big(x(N_i(\sqcup)), u(N_i(\sqcup)), I(\sqcup), \sqcup\Big) \tag{2.2}$$

3. Synaptic weight dynamics

[1] This symbol comes from an ancient Chinese philosopher ZHUANG ZHOU who believed that all the things one could define came from one's heart(brain). In this sense, this symbol means: "from all the things one can define".

$$S_i(\sqcup) \overset{\triangledown}{\longleftarrow} \mathcal{D}_S(\text{cells within } N_i(\sqcup)) \tag{2.3}$$

where $\mathcal{D}_x(\cdot,\cdot,\cdot)$, $\mathcal{D}_N(\cdot,\cdot,\cdot,\cdot)$ and $\mathcal{D}_S(\cdot)$ are three dynamic systems/or transformations.

Remarks:

1. Cell dynamics can be any dynamic system, function or functional, continuous or discrete-time, given a variable called *state*, a variable called *input* and a variable which can be accessed by the other neighbor cells called *output*.

2. Neighborhood system dynamics consists of any fixed, time-varying, space-varying, and movable organization rules. Even a random connection is possibly used. The only elementary principle here is *local cluster* of a collection of cells with any possible manner and organization.

3. Synaptic weight dynamics only defines the connection type within a neighborhood system. It may be a relation of matter transmission, chemical exchanges or energy exchanges and any other physical or non-physical relationships.

Definition 4

A unified CNN is defined by a cell dynamics in Eq. (2.1), a neighborhood dynamics in Eq. (2.2). And given a cell C_i which is restricted by the following neighborhood restriction:

$$N_i(\sqcup) < d_r(C_i) \tag{2.4}$$

where $d_r(C_i)$ denotes a kind of distance, which is less than r, from C_i.

Fig. 2.1 show the concept of this kind of unified CNN structure. In Fig. 2.1 the general time is labeled by "1" and "2", However, it only denotes two "snapshots" of the dynamics. In a real unified CNN model "$\sqcup$" may be continuous or discrete(or some cells are continuous while the other are discrete). In Fig. 2.1, every small circle denotes a cell and the black dot denotes cell C_i. The regions within the closed solid or dashed curves denote the neighborhood systems of C_i. The thin arrowed curves denote the direction and form of information flow. From Fig.2.1 one can see that $\mathcal{D}_N$ changes the shape of neighborhood system and also changes the numbers of cells within the neighborhood system at different moments. $\mathcal{D}_S$ changes the information flow directions and the connecting relationship between a center cell and its neighbor cells.

2.2 Principles of general FCNN

An FCNN structure should maintain two main features: 1) local connectedness between cells, and 2) simple cell structures and characteristics. A general FCNN is defined by:

Definition 5

1. Cell dynamics

$$\tilde{x}_i(\sqcup) \overset{\triangledown}{\longleftarrow} \tilde{\mathcal{D}}_{\tilde{x}}(\tilde{x}(\tilde{N}_i(\sqcup)), \tilde{u}(\tilde{N}_i(\sqcup)), \tilde{I}(\sqcup), \sqcup) \circ \tilde{S}_i(\sqcup) \tag{2.5}$$

2. Neighborhood system dynamics

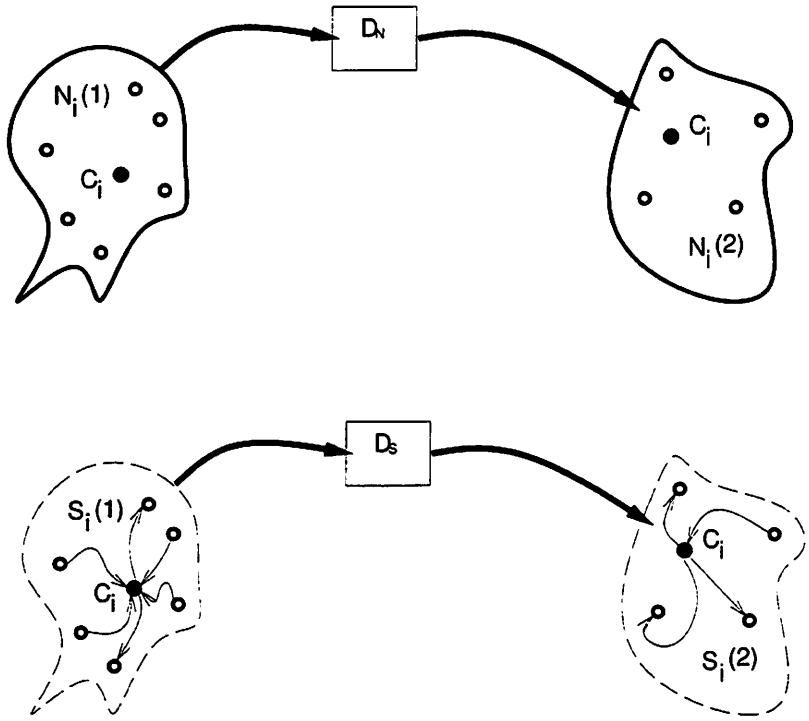


Figure 2.1: Unified CNN structure. In this figure only the neighborhood system dynamics(upper) and synaptic weight dynamics(lower) are shown.

$$\tilde{N}_i(\sqcup) \overset{\heartsuit}{\longleftarrow} \tilde{\mathcal{D}}_N(\tilde{x}(\tilde{N}_i(\sqcup)), \tilde{u}(\tilde{N}_i(\sqcup)), \tilde{I}(\sqcup), \sqcup) \tag{2.6}$$

3. Synaptic weight dynamics

$$\tilde{S}_i(\sqcup) \overset{\heartsuit}{\longleftarrow} \tilde{\mathcal{D}}_S(\text{cells within } \tilde{N}_i(\sqcup)) \tag{2.7}$$

Remark: This is a generalized case of the general CNN framework presented in Sec.2.1. In this definition, "~" over a character means that the character may represent something related to fuzzy set, e.g., fuzzy number, linguistic statement and conceptual variable. However, this definition is too general to be implemented and applied to special applications. In this book, we use the following operational definition of FCNN to unified all the existed FCNN structures.

Here, only the 2D cases are considered. For 1D and 3D cases, the definitions are similar.

Definition 6

A cell C_{ij} in an $M \times N$ general FCNN is defined by

1. State equation

$$
\begin{aligned}
& C\frac{dx_{ij}(t)}{dt} \\
= & \underbrace{-\frac{1}{R_x}x_{ij}(t) + \sum_{C_{kl}\in N_r(i,j)} A(i,j;k,l)y_{kl}(t) + \sum_{C_{kl}\in N_r(i,j)} B(i,j;k,l)u_{kl} + I}_{conventional\ part} \\
& \underbrace{\begin{array}{l} +\tilde{F}_{A_{C_{kl}\in N_r(i,j)}}\left(\mu_{A_f(i,j;k,l)}(A_f(i,j;k,l)), y_{kl}(t)\right) \\ +\tilde{F}_{B_{C_{kl}\in N_r(i,j)}}\left(\mu_{B_f(i,j;k,l)}(B_f(i,j;k,l)), \mu_{u_{kl}}(u_{kl})\right) \end{array}}_{fuzzy\ part} \\
& , 1 \le i \le M, 1 \le j \le N
\end{aligned} \tag{2.8}
$$

2. Output equation

$$y_{ij}(t) = \mu_{x_{ij}}(x_{ij}(t)), 1 \le i \le M, 1 \le j \le N \tag{2.9}$$

Remarks

1. From Eqs.(2.8) and (2.9) one can see that this structure is a type-III FCNN (see Sec.2.3). It can be degenerated to type-I, -II FCNN's and conventional CNN.

2. One can see that the fuzzifier layer is embedded in the fuzzy part. The reason that we don't insert a separate fuzzifier layer is that we can choose simple membership functions $\mu_{A_f(i,j;k,l)}(\cdot)$, $\mu_{B_f(i,j;k,l)}(\cdot)$, $\mu_{u_{kl}}(\cdot)$, and $\mu_{x_{ij}}(\cdot)$ which can be easily implemented by VLSI technique. It is easy to see that the fuzzy part consists of all the nonlinear synaptic laws while the conventional part consists of all the linear synaptic laws. However, we can not conclude that FCNN is only a kind of NCNN. Remember that the fuzzy number (i.e., a convex and normal fuzzy set on a real line) can propagate through an FCNN structure. For example, consider the following FDTCNN:

$$
\begin{aligned}
\tilde{x}_{ij}(k+1) = & F_{A_{C_{kl}\in N_r(i,j)}}\left(\tilde{A}_f(i,j;k,l), \tilde{y}_{kl}(k)\right) \\
& + F_{B_{C_{kl}\in N_r(i,j)}}\left(\tilde{B}_f(i,j;k,l), \tilde{u}_{kl}\right)
\end{aligned} \tag{2.10}
$$

$$\tilde{y}_{kl}(k) = F_x(\tilde{x}_{kl}(k)) \tag{2.11}$$

where the symbol "~" over a variable is used to denote that the variable is a fuzzy number. It is very clear that no conventional NCNN can implement this structure.

3. The membership function $\mu_{x_{ij}}(\cdot)$ in a conventional CNN may be interpreted as an "output function". Sometimes we would like to view it as a defuzzifier. For example, if we choose $\mu_{x_{ij}}(\cdot)$ as:

$$\mu_{x_{ij}}(x) = \begin{cases} 0, & x \le 0 \\ 1, & x > 0 \end{cases} \tag{2.12}$$

then it defuzzifies all states into classical logic variables.

4. $F_{A_{C_{kl}\in N_r(i,j)}}(\cdot)$ and $F_{B_{C_{kl}\in N_r(i,j)}}(\cdot)$ may be simple operations on fuzzy sets, e.g., *union, intersection, algebraic product, algebraic sum, bounded sum and bounded difference.* Also,

they may be complicated operations, e.g., *similarity between two fuzzy numbers.* By the way, $F_{A_{C_{kl}\in N_r(i,j)}}(\cdot)$ and $F_{B_{C_{kl}\in N_r(i,j)}}(\cdot)$ may also be any combination of the above operations. They can use any number of entries in A_f and B_f and use any y_{kl} and u_{kl} in $N_r(i,j)$. It is not necessary to keep the tradition of the conventional CNN where every entry in A or B can only multiply a y_{kl} or a u_{kl} because the relation between the (local) structure (synaptic weights) and the (local) information flow (inputs, state variables and outputs) is the only thing which we concern about in an FCNN.

5. It is easy to see that the general frame given by Eqs.(2.8) and (2.9) can be easily generalized to fuzzy DCNN(FDCNN) and fuzzy DTCNN(FDTCNN).

2.3 Classification of FCNN

Fortunately, there existed lots of literatures of FNN since 1974[181]. According to the method presented in [35], we can lump FNN into 3 types: 1) Type-I FNN, which has real signal and fuzzy weight[343, 344], 2) Type-II FNN, which has fuzzy signal and real weight[149, 295], and 3) Type-III FNN, which has fuzzy signal and fuzzy weight[146, 126, 26, 28, 223, 91]. We borrow the concepts of FNN to FCNN structures, so there also exist this kind of classification.

Since FCNN is a generalization of CNN from classical set to fuzzy set, it is not strange to enable a generalization of almost all the conclusions and applications of the conventional CNN. Corresponding to the classification presented in Sec.1.2.2, the FCNN structures can also be classified in that way. With the ability of interpreting linguistic statements, the fuzzy set theory should introduce the ability of processing linguistic inputs into FCNN. From this point of view, we hope that FCNN can model not only structures of neural systems, as the conventional CNN does, but also the behaviors and function of neural systems, namely, the cognitive processes.

From the above statements one can see that the classification of FCNN can be done along both the CNN direction and the FNN direction. There also exist some differences between the FNN and FCNN. In an FCNN, the cell property is not necessary space-invariant which means that FCNN has more possibility than FNN. For example, we can define the type-IV FCNN while there doesn't exist the corresponding type-IV FNN. A type-IV FCNN may contain all kinds of cells belong to any type-I, -II, and -III FCNN. This structure gives the type-IV FCNN the ability of processing both signals(real numbers) within some neighborhood systems and processing more general information flow (e.g., conceptual or linguistic variables) within some other neighborhood systems.

One can see that a type-II FCNN is the closest one to the conventional CNN because it has real weight. On the other hand, type-II FCNN has the simplest structure for VLSI implementation, this is the reason why so far almost all the results of FCNN are focused on this type.

2.4 Different structures of FCNN

Since it is impossible to study all the kinds of FCNN's in Eqs.(2.8) and (2.9) in a single book, we will focus on one of its simplest case, in which only the fuzzy logical OR ($\tilde{\vee}$ or MAX) and the fuzzy logical AND ($\tilde{\wedge}$ or MIN) are integrated. On the other hand, MAX and MIN are the simplest fuzzy union and intersection operations which can be implemented by using VLSI technique.

The structure of FCNN is a tradeoff between VLSI implementation and general function. For the purpose of VLSI implementation, the FCNN proposed in this book integrates the fuzzifier, the defuzzifier and the fuzzy inference engine into a planar structure. The nonlinear dynamics of the conventional CNN is kept in the new FCNN structure.

2.4.1 Type-I and Type-II FCNN

In this section, we give the structure of type-I and type-II FCNN's which will be used in this book. Also, only the 2D cases are given.

A cell C_{ij} in an $M \times N$ type-I FCNN is defined by

State equation

$$\begin{aligned} C\frac{dx_{ij}(t)}{dt} &= -\frac{1}{R_x}x_{ij}(t) + \sum_{C_{kl}\in N_r(i,j)} A(i,j;k,l)y_{kl}(t) \\ &+ \sum_{C_{kl}\in N_r(i,j)} B(i,j;k,l)u_{kl} + I \\ &+\tilde{F}_{A_{C_{kl}\in N_r(i,j)}}\left(\mu_{A_f(i,j;k,l)}(y_{kl}(t))\right) \\ &+\tilde{F}_{B_{C_{kl}\in N_r(i,j)}}\left(\mu_{B_f(i,j;k,l)}(u_{kl})\right) \\ &, 1 \le i \le M, 1 \le j \le N \end{aligned} \tag{2.13}$$

Output equation

$$y_{ij}(t) = f(x_{ij}(t)), 1 \le i \le M, 1 \le j \le N \tag{2.14}$$

Input equation

$$u_{ij} = E_{ij}, 1 \le i \le M, 1 \le j \le N \tag{2.15}$$

Constraint conditions

$$|u_{ij}| \le 1, 1 \le i \le M, 1 \le j \le N \tag{2.16}$$

Parameter assumptions

$$\begin{aligned} &C > 0, R_x > 0 \\ &|f(x_{ij}(t))| \le 1, 1 \le i \le M, 1 \le j \le N \end{aligned} \tag{2.17}$$

Boundary condition and initial condition

In a type-I FCNN, there exist fuzzy synaptic weights $\tilde{A}_f(i,j;k,l)$ and $\tilde{B}_f(i,j;k,l)$ (In this chapter, we use symbol "~" over a character to denote a fuzzy number. The relation between a fuzzy feedback synaptic weight and an output is defined by the membership function $\mu_{A_f(i,j;k,l)}(y_{kl})$. The relation between a fuzzy feed-forward synaptic weight and an input is defined by the membership function $\mu_{B_f(i,j;k,l)}(u_{kl})$. The inputs and outputs are crisp variables in a type-I FCNN.

Remark: One can see that fuzzy synaptic weights introduce a set of nonlinear synaptic laws into a type-I FCNN. In general case, the concept "template", which is very useful in conventional CNN[46] is not suitable for describing fuzzy synaptic laws.

A cell C_{ij} in a type-II FCNN is defined by

State equation

$$\begin{aligned} C\frac{dx_{ij}(t)}{dt} &= -\frac{1}{R_x}x_{ij}(t) + \sum_{C_{kl}\in N_r(i,j)} A(i,j;k,l)y_{kl}(t) + I \\ &+ \sum_{C_{kl}\in N_r(i,j)} B(i,j;k,l)u_{kl} + \tilde{F}_{A_{C_{kl}\in N_r(i,j)}}(A_f(i,j;k,l)y_{kl}(t)) \\ &+\tilde{F}_{B_{C_{kl}\in N_r(i,j)}}(B_f(i,j;k,l)\mu_{ukl}(u_{kl})) \\ &,1 \le i \le M, 1 \le j \le N \end{aligned} \tag{2.18}$$

Output equation

$$y_{ij}(t) = \mu_{xij}(x_{ij}(t)), 1 \le i \le M, 1 \le j \le N \tag{2.19}$$

Input equation

$$u_{ij} = E_{ij}, 1 \le i \le M, 1 \le j \le N \tag{2.20}$$

Constraint conditions

$$|u_{ij}| \le 1, 1 \le i \le M, 1 \le j \le N \tag{2.21}$$

Parameter assumptions

$$\begin{aligned} &C > 0, R_x > 0 \\ &|\mu_{ukl}(u_{kl})| \le 1, |\mu_{xij}(x_{ij}(t))| \le 1, 1 \le i \le M, 1 \le j \le N \end{aligned} \tag{2.22}$$

Boundary condition and initial condition

---

where $\mu_{ukl}(\cdot)$ and $\mu_{xij}(\cdot)$ are two membership functions. In a type-II FCNN, all the synaptic weights are crisp. Inputs and outputs are supposed to be fuzzy. They are described by membership functions $\mu_{ukl}(\cdot)$ and $\mu_{xij}(\cdot)$. One can see that $\mu_{xij}(\cdot)$ corresponds to the output function in a conventional CNN.

The above type-II FCNN is sometimes called *multiplicative* type-II FCNN. Correspondingly, there also exists the *additive* type-II FCNN whose state equation is given by

$$\begin{aligned} C\frac{dx_{ij}(t)}{dt} &= -\frac{1}{R_x}x_{ij}(t) + \sum_{C_{kl}\in N_r(i,j)} A(i,j;k,l)y_{kl}(t) + I \\ &+ \sum_{C_{kl}\in N_r(i,j)} B(i,j;k,l)u_{kl} + \tilde{F}_{A_{C_{kl}\in N_r(i,j)}}(A_f(i,j;k,l) + y_{kl}(t)) \\ &+\tilde{F}_{B_{C_{kl}\in N_r(i,j)}}(B_f(i,j;k,l) + \mu_{ukl}(u_{kl})) \\ &,1 \le i \le M, 1 \le j \le N \end{aligned} \tag{2.23}$$

A subclass of the additive type-II FCNN had been found to be a universal paradigm for implementing mathematical morphology operations[351, 352].

In type-I and type-II FCNN's, $\tilde{F}_A(\cdot)$ and $\tilde{F}_B(\cdot)$ denote two fuzzy local operators defined in $N_r(i,j)$, which may be any fuzzy logical expression combined by fuzzy OR "$\tilde{\vee}$" and fuzzy AND "$\tilde{\wedge}$". For example, suppose $\tilde{F}_A(\cdot)$ denotes the following fuzzy logical expression in a 1-neighborhood system:

$$\tilde{F}_A\left(\begin{array}{|c|c|c|}\hline x_{i-1,j-1} & x_{i-1,j} & x_{i-1,j+1} \\ \hline x_{i,j-1} & x_{i,j} & x_{i,j+1} \\ \hline x_{i+1,j-1} & x_{i+1,j} & x_{i+1,j+1} \\ \hline\end{array}\right) = \tilde{\bigwedge_{-1\le k,l\le 1}} x_{i-k,j-l} \tag{2.24}$$

where x_{ij} denotes fuzzy variable (for example, the gray value of the pixel (i,j)). Then we have

$$\tilde{F}_{A_{C_{kl}\in N_1(i,j)}}\left(\mu_{A_f(i,j;k,l)}(y_{ij})\right) = \tilde{\bigwedge_{C_{kl}\in N_1(i,j)}} \mu_{A_f(i,j;k,l)}(y_{kl}) \tag{2.25}$$

and

$$\tilde{F}_{A_{C_{kl}\in N_1(i,j)}}\left(A_f(i,j;k,l)y_{ij}\right) = \tilde{\bigwedge_{C_{kl}\in N_1(i,j)}} A_f(i,j;k,l)y_{kl} \tag{2.26}$$

A simple and most commonly used type-I FCNN is given by

$$\begin{aligned} & C\frac{dx_{ij}}{dt} \\ = & -\frac{1}{R_x}x_{ij} + \sum_{C_{kl}\in N_r(i,j)} A(i,j;k,l)y_{kl} + \sum_{C_{kl}\in N_r(i,j)} B(i,j;k,l)u_{kl} + I \\ & + \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} \mu_{A_{fmin}(i,j;k,l)}(y_{kl}) + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} \mu_{A_{fmax}(i,j;k,l)}(y_{kl}) \\ & + \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} \mu_{B_{fmin}(i,j;k,l)}(u_{kl}) + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} \mu_{B_{fmax}(i,j;k,l)}(u_{kl}) \\ & , 1\le i\le M, 1\le j\le N \end{aligned} \tag{2.27}$$

If FCNN is space-invariant, then in view of the method used in conventional CNN[46] Eq.(2.27) can be rewritten as the following 2D convolution form:

$$\begin{aligned} & C\frac{dx_{ij}}{dt} \\ = & -\frac{1}{R_x}x_{ij} + A * y_{ij} + B * u_{ij} + I \\ & + \tilde{A}_{fmin}\tilde{\odot}_{min}y_{ij} + \tilde{A}_{fmax}\tilde{\odot}_{max}y_{ij} + \tilde{B}_{fmin}\tilde{\odot}_{min}u_{ij} + \tilde{B}_{fmax}\tilde{\odot}_{max}u_{ij} \\ & , 1\le i\le M, 1\le j\le N \end{aligned} \tag{2.28}$$

where * denotes a 2D convolution. A, B are feedback and feed-forward *templates*, respectively. $\tilde{A}_{fmin}$, $\tilde{A}_{fmax}$, $\tilde{B}_{fmin}$ and $\tilde{B}_{fmax}$ are fuzzy feedback MIN template, fuzzy feedback

MAX template, fuzzy feed-forward MIN template, and fuzzy feed-forward MAX template, respectively. $\tilde{\odot}_{max}$ denotes a 2D operation as shown in following example:

$$\tilde{A}_{fmax}\tilde{\odot}_{max}y_{ij} = \tilde{\bigvee}_{C_{kl}\in N_r(i,j)} \mu_{A_{fmax}(i,j;k,l)}(y_{kl}) \tag{2.29}$$

and $\tilde{\odot}_{min}$ denotes a 2D operation as shown in following example:

$$\tilde{A}_{fmin}\tilde{\odot}_{min}y_{ij} = \tilde{\bigwedge}_{C_{kl}\in N_r(i,j)} \mu_{A_{fmin}(i,j;k,l)}(y_{kl}) \tag{2.30}$$

From the above one can see that fuzzy templates $\tilde{A}_f$ and $\tilde{B}_f$ are local fuzzy patterns which are fuzzified for purpose of fitting a more loose relation between fuzzy templates and signal patterns.

A simple and most used type-II FCNN is given by

$$\begin{aligned} & C\frac{dx_{ij}}{dt} \\ = & -\frac{1}{R_x}x_{ij} + \sum_{C_{kl}\in N_r(i,j)} A(i,j;k,l)y_{kl} + \sum_{C_{kl}\in N_r(i,j)} B(i,j;k,l)u_{kl} + I \\ & + \tilde{\bigwedge}_{C_{kl}\in N_r(i,j)} A_{fmin}(i,j;k,l)y_{kl} + \tilde{\bigvee}_{C_{kl}\in N_r(i,j)} A_{fmax}(i,j;k,l)y_{kl} \\ & + \tilde{\bigwedge}_{C_{kl}\in N_r(i,j)} B_{fmin}(i,j;k,l)\mu_u(u_{kl}) + \tilde{\bigvee}_{C_{kl}\in N_r(i,j)} B_{fmax}(i,j;k,l)\mu_u(u_{kl}) \\ & ,1 \le i \le M, 1 \le j \le N \end{aligned} \tag{2.31}$$

Also, Eq.(2.31) can be rewritten as the following 2D convolution form:

$$\begin{aligned} & C\frac{dx_{ij}}{dt} \\ = & -\frac{1}{R_x}x_{ij} + A * y_{ij} + B * u_{ij} + I \\ & + A_{fmin} \odot_{min} y_{ij} + A_{fmax} \odot_{max} y_{ij} + B_{fmin} \odot_{min} \mu_u(u_{ij}) \\ & + B_{fmax} \odot_{max} \mu_u(u_{ij}) \\ & ,1 \le i \le M, 1 \le j \le N \end{aligned} \tag{2.32}$$

where A_{fmin}, A_{fmax}, B_{fmin} and B_{fmax} are feedback MIN template, feedback MAX template, feed-forward MIN template, and feed-forward MAX template, respectively. $\odot_{max}$ denotes a 2D operation as shown in following example:

$$A_{fmax} \odot_{max} y_{ij} = \tilde{\bigvee}_{C_{kl}\in N_r(i,j)} A_{fmax}(i,j;k,l)y_{kl} \tag{2.33}$$

and $\odot_{min}$ denotes a 2D operation as shown in following example:

$$A_{fmin} \odot_{min} y_{ij} = \tilde{\bigwedge}_{C_{kl}\in N_r(i,j)} A_{fmin}(i,j;k,l)y_{kl} \tag{2.34}$$

2.5 Differences between FCNN and FNN

One should also see that FCNN structure has a significant difference from other FNN structures. In an FCNN, the fuzzifier layer and the defuzzifier layer, which always appear in a standard FNN structure, are embedded into a single layer. This planar structure is mostly suitable for 2D VLSI implementation because links between two chips are avoided. So, an FCNN universal cell should be programmable for different membership functions and allow some basic programs of fuzzy operations (relational computation).

The most significant characteristics of FCNN are the local connectedness of the cells. This spatial(or, relational) local connection can be used to derive different spatial(or, relational) structures of FCNN cells. Any FNN structure which is based on the local connectedness will fall in the range of the FCNN concept.

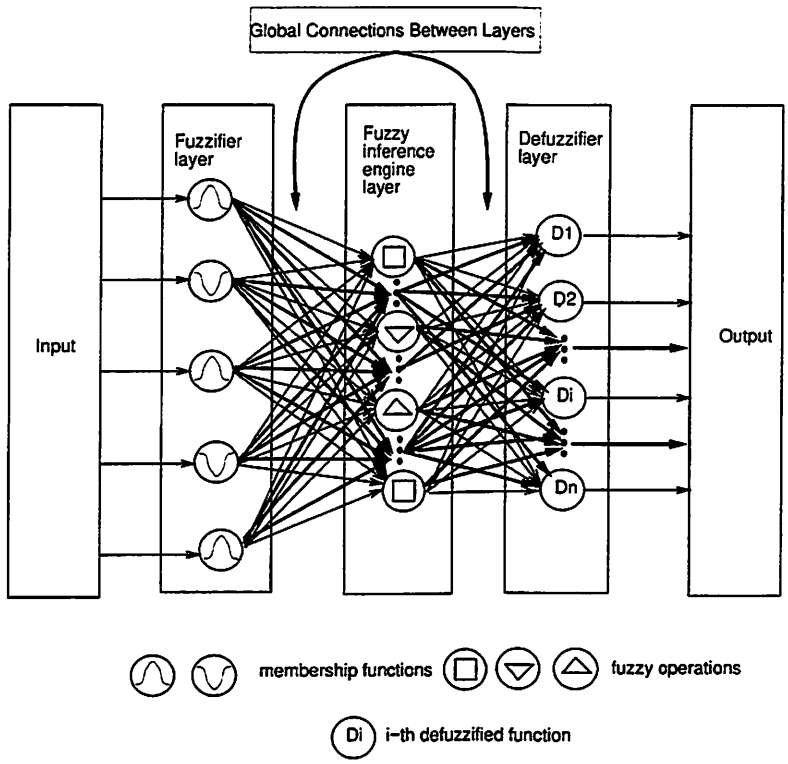
To understand the differences between FCNN and FNN, we first show how can we get the FCNN structures from the crossover of FNN and CNN. A typical FNN has three layers as shown in Fig.2.2(a). The first layer is used to give the crisp inputs some fuzzy measurements. The nonlinearities in the neurons of this layer are some membership functions. The second layer is used to calculate the relationship between different fuzzy variables from the first layer. The fuzzy computations are embedded into the nonlinearities of this layer. The third layer is used to give some crisp forms of outputs. It's easy to see that if the inputs are already fuzzy variables(e.g., the gray value of a pixels in a digital image) then the first layer can be eliminated. Also, if we need some fuzzy outputs (e.g., outputs which are used by a high-level AI system), then the third layer can be eliminated.

Fig.2.2(b) shows the typical structure of a signal layer conventional CNN. One can see that this single layer CNN contains an input function sub-layer, a cell dynamics sub-layer and output function sub-layer. In a conventional CNN, the input function is usually a linear function and the output function is a piecewise linear function. Of course, we can use more complicated nonlinearities as input and output functions.

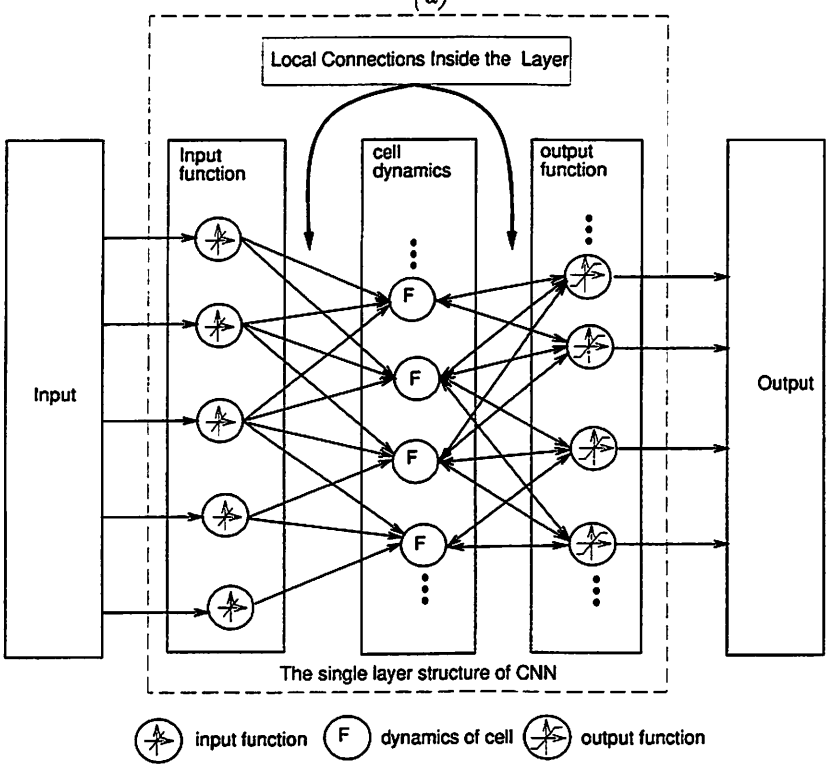
Then we are in the situation to show how can we combine the both structures in Fig.2.2(a) and (b) into FCNN, which is shown in Fig.2.2(c). We only shown a single layer FCNN structure. One can see that by using different membership functions as the input functions, we embedded the fuzzifier layer of FNN into the input function sub-layer of FCNN. We combine the cell dynamics of CNN and the fuzzy operations of FNN into a fuzzy-crisp mixed dynamical sub-layer which contributes the cell dynamics of FCNN. the fuzzy-crisp mixed dynamics makes our FCNN structure can solve both crisp and fuzzy problems. Finally, we choose the nonlinearities of the defuzzifier layer of FNN as the output functions of FCNN. The output function sub-layer is equivalent to the defuzzifier layer of FNN. Of course, the FCNN structures carry out the properties of local connectedness from CNN.

2.6 A brief history of FCNN

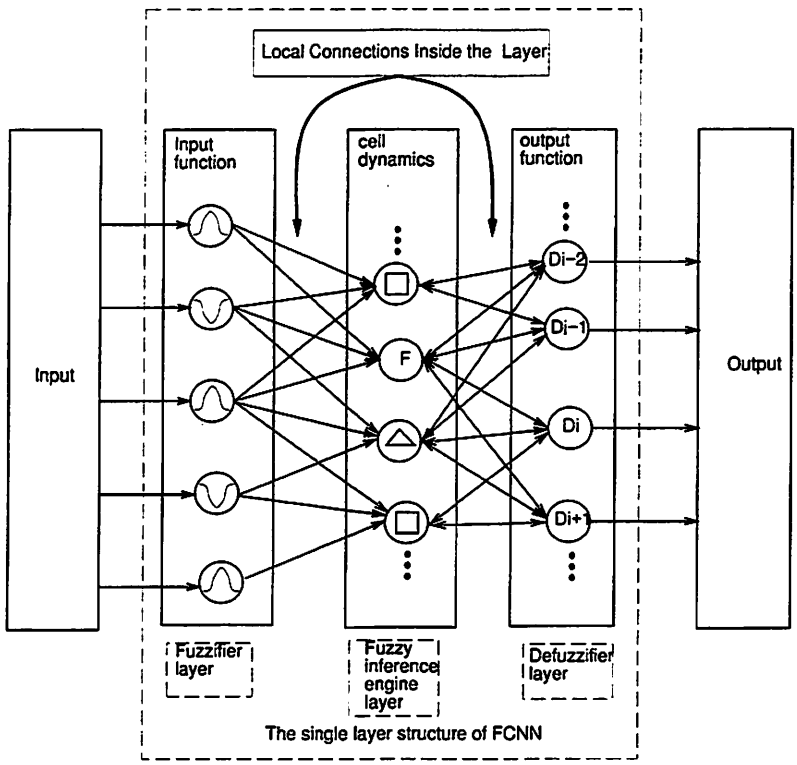
In early 1993 the FCNN structures were first presented in China by a Chinese group, which consisted of a fuzzy mathematician, a biologist, a physical professor, a control engineer professor, and more then 20 Ph.D and master students form these fields, guided by Tao Yang who is an assistant professor of electrical engineering. From early 1993 to 1996, this group presented over 20 technical papers in Chinese on FCNN both in theory and applications [374, 367, 369, 371, 375, 372, 368, 365, 370, 363, 364, 373, 366, 379, 377, 380, 378, 376, 381,


Global Connections Between Layers
Input
Fuzzifier layer
Fuzzy inference engine layer
Defuzzifier layer
Output
D1
D2
Di
Dn
membership functions
fuzzy operations
Di i-th defuzzified function

(a)


Local Connections Inside the Layer
Input
Input function
cell dynamics
output function
Output
F
F
F
F
The single layer structure of CNN
input function
F
dynamics of cell
output function

(b)



(c)

Figure 2.2: FCNN structure is a crossover of conventional CNN and the FCNN. (a) The typical structure of the FNN. (b) The typical structure of the conventional CNN. (c) The structure of the FCNN.

385, 386, 384, 382, 383]. Then, this group try to translate all the results into English and contributed them to the international academic and engineering communities. The series work of FCNN from this group have been or will be published in different international conference proceedings and journals [350, 351, 352, 354, 353, 356, 355]. This book contains almost all the results of the translated papers.

At the very beginning, lots of constructive, non-constructive, professional and non-professional criticisms from some authorities of CNN field , fuzzy field and neural network field were poured on the new concept of FCNN. But the group leaded by Tao Yang knew what they did was an inevitable tendency of this field just as the fuzzy set to conventional set theory. Two years later, some of the primary results were received by the international community. The first English paper that this group sent to *IEEE Transaction on Circuits and Systems*, which is the only journal in the world has a separate section and an associate editor for CNN, as a short Express Letter dated Oct.23, 1994, after numerical revisions, eventually published in October issue of 1996[350]. We think this may be the longest delayed Expressed Letter in the history of this journal. However, this was the first victory in the history of FCNN and announced the reinvention of this branch in Western Community. We want to thank the editors of this journal for not abandoning this orphan and the knowledgeable, high-level reviewers for their first rate comments which were invaluable to the growth of this branch. And after that, Tao Yang went to the University of California at Berkeley(UCB) as a visiting scholar. There he jointed the CNN group of UCB and published two conference papers which are something that very elementary to the type-II FCNN[356, 355].

At the time we wrote this book, since the only existed publications are these two papers in proceedings of CNNA'96[356, 355], in which only the basic structure and the elementary theoretical results of the simplest type-II FCNN were presented. The titles of these two papers were proved to be misleading because they gave the readers an impression that the type-II FCNN was all what the FCNN concept and structure meant. Based on this, it was not strange that some readers argued that "FCNN is only a kind of NCNN". This illusion is totally wrong because the results that will be published elsewhere soon have clearly shown that FCNN is much more that the simplest type-II FCNN—although this kind of FCNN is well studied, well understood, and well known. This is one of the reasons why we want to published this book. We also hope that the collection of the work which performed in the last three years can give the reader a panorama of what FCNN is.

Here, also exist lots of obstacles for the birth of FCNN just like the birth of fuzzy set theory itself. The obstacles mainly come from two fields. One field is the artificial neural network(ANN) people who found lots of applications of FCNN(or generally CNN) to image processing problems didn't need the learning ability which is a significant trademark of ANN. They can't image how a structure without learning algorithm can be called as "something-NN". This problem should be fought back together with the CNN people. Yes, some FCNN (or CNN) have fixed weights and do not have any learning ability because we use this kind of FCNN as a platform which we called *computational* FCNN. Although the weights are fixed, they are trained by examples before we can use them as computational FCNN. On the other hand, the ANN people will find that lots of weights of computational FCNN are trained by the standard learning algorithms of ANN(e.g., BP algorithms and Hebbian algorithms).

Some other arguments come from the conventional CNN people. They found that the first generation of type-II FCNN has a very similar form as NCNN, so they decided to say that FCNN was only an NCNN, there existed nothing new. But they did not see the

type-I, -III and even -IV FCNN, where the revolutionary changes were introduced from the information point of view to explain why the information—even the linguistic variables, the fuzzy numbers, the conceptual variables and some other functional things besides signals(the real numbers)—can flow through FCNN structures. The conclusion is that FCNN is a more general concept than the conventional CNN and it should not be a small subset of the conventional CNN.

Once again, those CNN people who have blind faith in the conventional CNNUM will try to figure out all the strange CNNUM programs and some complicated piece-wise linear nonlinearities with many breakpoints to embedded the fuzzy concept into the conventional CNN framework. We find that it is totally unnecessary because given the VLSI techniques for implementing fuzzy chip and CNN chip, we can easily fabricate the FCNN chip. We think that the natural tendency is to choose the new concept and build the new chip instead of being trapped into the old and narrow concept. Anyhow, time will tell us what will happen.

Chapter 3

Theory of FCNN

In this chapter, we study the elementary theory of different kinds of FCNN. This chapter is the collection of our papers[350, 354, 353, 356, 367, 369, 371, 380, 385]. In this chapter, we always present the results for type-II FCNN first and then present the corresponding results of type-I FCNN.

3.1 Elementary theory

In this section, we study the dynamical range of type-I and -II FCNN. The dynamical range is important to a physically implemented FCNN because only we know the dynamical range then we can choose the power supply, and physical structures of FCNN. Also, the existence of equilibrium point for type-I and -II FCNN is studied for guaranteeing the correct operation of FCNN.

3.1.1 Dynamical range of type-II FCNN

To guarantee that FCNN's can be realized by physical systems, we should study its dynamic range. In this section, we study the dynamic range of type-II FCNN in Eq.(2.31). First, we need the following definition:

Definition 7

Dissipative FCNN: Let Ξ be a compact set in R^{MN}, if all solutions of an FCNN finally fall into Ξ and stay in Ξ, then this FCNN is called a dissipative FCNN.

Remark: The dynamical range of a dissipative FCNN is Ξ.
Let

$$\phi_{kl} = \begin{cases} 0 & \text{, when } (k,l) = (i,j) \text{ and } A(i,j;k,l) \leq 0 \\ 1 & \text{, when } (k,l) = (i,j) \text{ and } A(i,j;k,l) > 0, \text{ or, } (k,l) \neq (i,j) \end{cases} \tag{3.1}$$

then we have the following theorem:

Theorem 1

The type-II FCNN in Eq.(2.31) is a dissipative FCNN and all its solutions with any initial conditions $x_{ij}(0)$ finally fall into the following compact set:

$$\Xi \triangleq R^{MN}/\Omega_1 \bigcap R^{MN}/\Omega_2 \tag{3.2}$$

where

$$\begin{aligned}
\Omega_1 \triangleq & \Big\{\mathbf{x}|\sum_{i,j}[|x_{ij}| - \frac{R_x}{2}(\sum_{k,l}|A(i,j;k,l)|\phi_{kl} + \sum_{k,l}|B(i,j;k,l)| + |I| \\
& + \tilde{\bigvee_{k,l}}|A_{fmax}(i,j;k,l)|\phi_{kl} + \tilde{\bigwedge_{k,l}}|A_{fmin}(i,j;k,l)|\phi_{kl} \\
& + \tilde{\bigvee_{k,l}}|B_{fmax}(i,j;k,l)| + \tilde{\bigwedge_{k,l}}|B_{fmin}(i,j;k,l)|)]^2 \\
> & \sum_{i,j}[\frac{R_x}{2}(\sum_{k,l}|A(i,j;k,l)|\phi_{kl} + \sum_{k,l}|B(i,j;k,l)| + |I| \\
& + \tilde{\bigvee_{k,l}}|A_{fmax}(i,j;k,l)|\phi_{kl} + \tilde{\bigvee_{k,l}}|A_{fmin}(i,j;k,l)|\phi_{kl} \\
& + \tilde{\bigvee_{k,l}}|B_{fmax}(i,j;k,l)| + \tilde{\bigvee_{k,l}}|B_{fmin}(i,j;k,l)|)]^2\Big\}
\end{aligned} \tag{3.3}$$

$$\begin{aligned}
\Omega_2 \triangleq & \Big\{\mathbf{x}| \ |x_{ij}| > M_{ij} \triangleq R_x(\sum_{k,l}|A(i,j;k,l)|\phi_{kl} + \sum_{k,l}|B(i,j;k,l)| + |I| \\
& + \tilde{\bigvee_{k,l}}|A_{fmax}(i,j;k,l)|\phi_{kl} + \tilde{\bigvee_{k,l}}|A_{fmin}(i,j;k,l)|\phi_{kl} \\
& + \tilde{\bigvee_{k,l}}|B_{fmax}(i,j;k,l)| + \tilde{\bigvee_{k,l}}|B_{fmin}(i,j;k,l)|)\Big\}, \\
& 1 \le i \le M, 1 \le j \le N
\end{aligned} \tag{3.4}$$

where $\mathbf{x} = col(x_{11}, ..., x_{MN})$.

Proof: (1) We construct a radially unbounded positive definite Lyapunov function:

$$V_1 = \frac{1}{2C}\sum_{i=1}^{M}\sum_{j=1}^{N}x_{ij}^2 \tag{3.5}$$

Differentiate V_1 along the solution of Eq.(2.31), and since $|y_{ij}| \le 1$, $|u_{ij}| \le 1$ and $|\mu_u(u_{ij})| \le 1$, we have:

$$\begin{aligned}
& \frac{dV_1}{dt}|_{Eq.(2.31)} \\
= & \sum_{i,j}\Big[-\frac{1}{R_x}|x_{ij}|^2 + \sum_{k,l}A(i,j;k,l)y_{kl}x_{ij} + \sum_{k,l}B(i,j;k,l)u_{kl}x_{ij} + Ix_{ij} \\
& + x_{ij}\tilde{\bigvee_{k,l}}A_{fmax}(i,j;k,l)y_{kl} + x_{ij}\tilde{\bigwedge_{k,l}}A_{fmin}(i,j;k,l)y_{kl}
\end{aligned}$$

$$
\begin{aligned}
&+x_{ij}\tilde{\bigvee_{k,l}}B_{fmax}(i,j;k,l)\mu_u(u_{kl})+x_{ij}\tilde{\bigwedge_{k,l}}B_{fmin}(i,j;k,l)\mu_u(u_{kl})\Big]\\
\leq\ &\sum_{i,j}-\frac{1}{R_x}\Big[|x_{ij}|^2-R_x\Big(\sum_{k,l}|A(i,j;k,l)\phi_{kl}||x_{ij}|+\sum_{k,l}|B(i,j;k,l)||x_{ij}|\\
&+|I||x_{ij}|+|x_{ij}|\tilde{\bigvee_{k,l}}|A_{fmax}(i,j;k,l)\phi_{kl}|+|x_{ij}|\tilde{\bigvee_{k,l}}|A_{fmin}(i,j;k,l)\phi_{kl}|\\
&+|x_{ij}|\tilde{\bigvee_{k,l}}|B_{fmax}(i,j;k,l)|+|x_{ij}|\tilde{\bigvee_{k,l}}|B_{fmin}(i,j;k,l)|\Big)\Big]\\
\leq\ &\sum_{i,j}-\frac{1}{R_x}\Big[|x_{ij}|-\frac{R_x}{2}\Big(\sum_{k,l}|A(i,j;k,l)|\phi_{kl}+\sum_{k,l}|B(i,j;k,l)|+|I|\\
&+\tilde{\bigvee_{k,l}}|A_{fmax}(i,j;k,l)|\phi_{kl}+\tilde{\bigvee_{k,l}}|A_{fmin}(i,j;k,l)|\phi_{kl}\\
&+\tilde{\bigvee_{k,l}}|B_{fmax}(i,j;k,l)|+\tilde{\bigvee_{k,l}}|B_{fmin}(i,j;k,l)|\Big)\Big]^2\\
&+\sum_{i,j}\frac{R_x}{4}\Big(\sum_{k,l}|A(i,j;k,l)|\phi_{kl}+\sum_{k,l}|B(i,j;k,l)|+|I|\\
&+\tilde{\bigvee_{k,l}}|A_{fmax}(i,j;k,l)|\phi_{kl}+\tilde{\bigvee_{k,l}}|A_{fmin}(i,j;k,l)|\phi_{kl}\\
&+\tilde{\bigvee_{k,l}}|B_{fmax}(i,j;k,l)|+\tilde{\bigvee_{k,l}}|B_{fmin}(i,j;k,l)|\Big)^2\\
<\ &0
\end{aligned}
\tag{3.6}
$$

The last inequality is satisfied when $\mathbf{x}\in\Omega_1$.

(2) Then we construct MN radially unbounded Lyapunov functions with respect to the state variable x_{ij}:

$$V_{ij}=\frac{1}{C}x_{ij}sgn(x_{ij}),1\leq i\leq M,1\leq j\leq N \tag{3.7}$$

where $sgn(\cdot)$ is the signum function.

Along the solution of Eq.(2.31), we calculate the Dini upper-right differential as:

$$
\begin{aligned}
D^+V_{ij}|_{Eq.(2.31)}\ \leq\ &\Big(-\frac{1}{R_x}|x_{ij}|+\sum_{k,l}|A(i,j;k,l)|\phi_{kl}+\sum_{k,l}|B(i,j;k,l)|+|I| \qquad (3.8)\\
&+\tilde{\bigvee_{k,l}}|A_{fmax}(i,j;k,l)|\phi_{kl}+\tilde{\bigvee_{k,l}}|A_{fmin}(i,j;k,l)|\phi_{kl}\\
&+\tilde{\bigvee_{k,l}}|B_{fmax}(i,j;k,l)|+\tilde{\bigvee_{k,l}}|B_{fmin}(i,j;k,l)|\Big)\\
<\ &0 \qquad (3.9)
\end{aligned}
$$

the last inequality is satisfied when $\mathbf{x}\in\Omega_2$. So, the solution of Eq.(2.31) will fall in R^{MN}/Ω_2, and fall in Ξ and stay in Ξ. If $x(0)\in\Xi$, then $x(t)\in\Xi,\forall t>0$. So, Ξ is an ω-invariant set, in R^{MN}/Ξ there exists no stable equilibrium point of the FCNN in Eq.(2.31). □

Remark: This theorem gives the dynamical range of an FCNN, in practical circuit design, we can choose proper parameters to guarantee that the FCNN can work in the typical power supply voltage range of IC circuits.

Our conclusion is more accurate than that given by Theorem 1 in [47]. For comparison, we list the conclusion in [47] as follows: the bound on the state of a cell C_{ij} is:

$$\begin{aligned} |x_{ij}(t,0,\mathbf{x}(0))| &\leq 1+\max_{ij} R_x(\sum_{k,l}|A(i,j;k,l)|+\sum_{k,l}|B(i,j;k,l)|+|I|) \\ &= 1+\overline{M}, |x_{ij}(0)|\leq 1, 1\leq i\leq M, 1\leq j\leq N \end{aligned} \tag{3.10}$$

Let $A_{fmin}(i,j;k,l)=0$, $A_{fmax}(i,j;k,l)=0$, $B_{fmin}(i,j;k,l)=0$ and $B_{fmax}(i,j;k,l)=0$, $1\leq i,k\leq M$, $1\leq j,l\leq N$. Since the solution falling in R^{MN}/Ω_2, we have the corresponding bound given by Theorem 1 with respect to the state of C_{ij} as:

(1) If $\max_{ij} M_{ij}\geq 1$, then

$$\begin{aligned} &|x_{ij}(t,0,\mathbf{x}(0))|\leq \max_{ij} M_{ij}=\overline{M}, \\ &|x_{ij}(0)|\leq 1, 1\leq i\leq M, 1\leq j\leq N \end{aligned} \tag{3.11}$$

(2) If $\max_{ij} M_{ij}<1$, then

$$|x_{ij}(t,0,\mathbf{x}(0))|\leq 1, |x_{ij}(0)|\leq 1, 1\leq i\leq M, 1\leq j\leq N \tag{3.12}$$

(3) $\forall \mathbf{x}(0)\in R^{MN}$, there exists a $T>0$, such that when $t\geq T$ we have

$$|x_{ij}(t,0,\mathbf{x}(0))|\leq \max_{ij} M_{ij}=\overline{M} \tag{3.13}$$

The bounds in Eqs.(3.11) and (3.12) are more accurate than that in Eq.(3.10). And the Theorem 1 in [47] has no conclusion corresponding to the conclusion in Eq.(3.13).

The following theorem guarantees that the FCNN in Eq.(2.31) has at least one equilibrium point.

Theorem 2

The FCNN in Eq.(2.31) has at least one equilibrium point.

Proof: Let the right hand side of Eq.(2.31) be 0, then we have:

$$\begin{aligned} x_{ij} = R_x\Big[&\sum_{C_{kl}\in N_r(i,j)} A(i,j;k,l)y_{kl}+\sum_{C_{kl}\in N_r(i,j)} B(i,j;k,l)u_{kl}+I \\ &+\tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} A_{fmin}(i,j;k,l)y_{kl}+\tilde{\bigvee_{C_{kl}\in N_r(i,j)}} A_{fmax}(i,j;k,l)y_{kl} \\ &+\tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} B_{fmin}(i,j;k,l)\mu_u(u_{kl}) \\ &+\tilde{\bigvee_{C_{kl}\in N_r(i,j)}} B_{fmax}(i,j;k,l)\mu_u(u_{kl})\Big] \\ &,1\leq i\leq M, 1\leq j\leq N \end{aligned} \tag{3.14}$$

Consider the following vector operator:

$$\Phi=(\phi_{ij}(\mathbf{x}))_{MN} \tag{3.15}$$

where

$$\begin{aligned}\phi_{ij}(\mathbf{x}) &= R_x\Big[\sum_{C_{kl}\in N_r(i,j)} A(i,j;k,l)y_{kl} + \sum_{C_{kl}\in N_r(i,j)} B(i,j;k,l)u_{kl} + I \\ &+ \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} A_{fmin}(i,j;k,l)y_{kl} + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} A_{fmax}(i,j;k,l)y_{kl} \\ &+ \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} B_{fmin}(i,j;k,l)\mu_u(u_{kl}) \\ &+ \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} B_{fmax}(i,j;k,l)\mu_u(u_{kl})\Big] \\ &, 1 \le i \le M, 1 \le j \le N \end{aligned} \tag{3.16}$$

and $\mathbf{x} = col(x_{11}, ..., x_{MN})$.

Let

$$\begin{aligned}M &= max\Big\{R_x\Big[\sum_{C_{kl}\in N_r(i,j)} |A(i,j;k,l)| + \sum_{C_{kl}\in N_r(i,j)} |B(i,j;k,l)| + |I| \\ &+ \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} |A_{fmin}(i,j;k,l)| + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} |A_{fmax}(i,j;k,l)| \\ &+ \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} |B_{fmin}(i,j;k,l)| + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} |B_{fmax}(i,j;k,l)|\Big]\Big\}\end{aligned} \tag{3.17}$$

Then the vector operator Φ maps the following set

$$S = \{\mathbf{x} | |x_{ij}| \le M, 1 \le i \le M, 1 \le j \le N\} \tag{3.18}$$

into itself. Since S is a convex compact set, from Brown's fix-point theorem, we know $\Phi : S \longmapsto S$ has at least a fix-point $\mathbf{x} = \mathbf{x}^*$. And $\mathbf{x}^*$ is an equilibrium point of FCNN in Eq.(2.31). □

3.1.2 Dynamical range of type-I FCNN

Theorem 3

The type-I FCNN in Eq.(2.27) is a dissipative FCNN and any of its solution with any initial condition $x_{ij}(0)$ will fall into the following compact set:

$$\Xi \triangleq R^{MN}/\Omega_1 \bigcap R^{MN}/\Omega_2 \tag{3.19}$$

where

$$\begin{aligned}\Omega_1 \triangleq & \Big\{\mathbf{x}|\sum_{i,j}[|x_{ij}| - \frac{R_x}{2}(\sum_{k,l}|A(i,j;k,l)|\phi_{kl} + \sum_{k,l}|B(i,j;k,l)| + |I| + 4)]^2 \\ & > \sum_{i,j}[\frac{R_x}{2}(\sum_{k,l}|A(i,j;k,l)|\phi_{kl} + \sum_{k,l}|B(i,j;k,l)| + |I| + 4)]^2\Big\}\end{aligned} \tag{3.20}$$

$$\begin{aligned}\Omega_2 \triangleq & \Big\{\mathbf{x}|\ |x_{ij}| > M_{ij} \triangleq R_x(\sum_{k,l}|A(i,j;k,l)|\phi_{kl} \\ & + \sum_{k,l}|B(i,j;k,l)| + |I| + 4)\Big\}, \\ & 1 \le i \le M, 1 \le j \le N\end{aligned} \tag{3.21}$$

where $\mathbf{x} = col(x_{11}, ..., x_{MN})$.

Proof: (1) We construct a radially unbounded positive definite Lyapunov function:

$$V_1 = \frac{1}{2C}\sum_{i=1}^{M}\sum_{j=1}^{N}x_{ij}^2 \tag{3.22}$$

Differentiate V_1 along the solution of Eq.(2.27), and since $|y_{ij}| \le 1$, $|u_{ij}| \le 1$ and $|\mu(\cdot)| \in [0,1]$ ($\mu(\cdot)$ is any of $\mu_{A_{fmin}(i,j;k,l)}(\cdot)$, $\mu_{A_{fmax}(i,j;k,l)}(\cdot)$,$\mu_{B_{fmin}(i,j;k,l)}(\cdot)$ and $\mu_{B_{fmax}(i,j;k,l)}(\cdot)$), we have:

$$\begin{aligned}& \frac{dV_1}{dt}|_{Eq.(2.27)} \\ = & \sum_{i,j}\Big[-\frac{1}{R_x}|x_{ij}|^2 + \sum_{k,l}A(i,j;k,l)y_{kl}x_{ij} + \sum_{k,l}B(i,j;k,l)u_{kl}x_{ij} + Ix_{ij} \\ & + x_{ij}\tilde{\bigvee_{k,l}}\mu_{A_{fmax}(i,j;k,l)}(y_{kl}) + x_{ij}\tilde{\bigwedge_{k,l}}\mu_{A_{fmin}(i,j;k,l)}(y_{kl}) \\ & + x_{ij}\tilde{\bigvee_{k,l}}\mu_{B_{fmax}(i,j;k,l)}(u_{kl}) + x_{ij}\tilde{\bigwedge_{k,l}}\mu_{B_{fmin}(i,j;k,l)}(u_{kl})\Big] \\ \le & \sum_{i,j} -\frac{1}{R_x}\Big[|x_{ij}|^2 - R_x\Big(\sum_{k,l}|A(i,j;k,l)\phi_{kl}||x_{ij}| \\ & + \sum_{k,l}|B(i,j;k,l)||x_{ij}| + |I||x_{ij}| \\ & + |x_{ij}|\tilde{\bigvee_{k,l}}|\mu_{A_{fmax}(i,j;k,l)}(y_{kl})| + |x_{ij}|\tilde{\bigwedge_{k,l}}|\mu_{A_{fmin}(i,j;k,l)}(y_{kl})|\end{aligned}$$

$$
\begin{aligned}
&+|x_{ij}|\tilde{\bigvee_{k,l}}|\mu_{B_{fmax}(i,j;k,l)}(u_{kl})| + |x_{ij}|\tilde{\bigwedge_{k,l}}|\mu_{B_{fmin}(i,j;k,l)}(u_{kl})|\Big)\Big] \\
\leq\ & \sum_{i,j} -\frac{1}{R_x}\Big[|x_{ij}|^2 - R_x\Big(\sum_{k,l}|A(i,j;k,l)\phi_{kl}||x_{ij}| \\
&+\sum_{k,l}|B(i,j;k,l)||x_{ij}| + |I||x_{ij}| + 4|x_{ij}|\Big)\Big] \\
\leq\ & \sum_{i,j} -\frac{1}{R_x}\Big[|x_{ij}| - \frac{R_x}{2}\Big(\sum_{k,l}|A(i,j;k,l)|\phi_{kl} + \sum_{k,l}|B(i,j;k,l)| + |I| + 4\Big)\Big]^2 \\
&+\sum_{i,j}\frac{R_x}{4}\Big(\sum_{k,l}|A(i,j;k,l)|\phi_{kl} + \sum_{k,l}|B(i,j;k,l)| + |I| + 4\Big)^2 \\
<\ & 0
\end{aligned}
\tag{3.23}
$$

The last inequality is satisfied when $\mathbf{x} \in \Omega_1$.

(2) Then we construct MN radially unbounded Lyapunov functions with respect to the state variable x_{ij}:

$$
V_{ij} = \frac{1}{C}x_{ij}sgn(x_{ij}), 1 \leq i \leq M, 1 \leq j \leq N \tag{3.24}
$$

where $sgn(\cdot)$ is the signum function.

Along the solution of Eq.(2.27), we calculate the Dini upper-right differential as:

$$
\begin{aligned}
D^+V_{ij}|_{Eq.(2.27)} \leq\ & \Big(-\frac{1}{R_x}|x_{ij}| + \sum_{k,l}|A(i,j;k,l)|\phi_{kl} + \sum_{k,l}|B(i,j;k,l)| + |I| \\
&+\tilde{\bigvee_{k,l}}|\mu_{A_{fmax}(i,j;k,l)}(y_{kl})| + \tilde{\bigwedge_{k,l}}|\mu_{A_{fmin}(i,j;k,l)}(y_{kl})| \\
&+\tilde{\bigvee_{k,l}}|\mu_{B_{fmax}(i,j;k,l)}(u_{kl})| + \tilde{\bigwedge_{k,l}}|\mu_{B_{fmin}(i,j;k,l)}(u_{kl})|\Big) \\
\leq\ & \Big(-\frac{1}{R_x}|x_{ij}| + \sum_{k,l}|A(i,j;k,l)|\phi_{kl} \\
&+\sum_{k,l}|B(i,j;k,l)| + |I| + 4\Big) \\
<\ & 0
\end{aligned}
\tag{3.25}
$$

the last inequality is satisfied when $\mathbf{x} \in \Omega_2$. So, the solution of Eq.(2.27) will fall in R^{MN}/Ω_2, and fall in Ξ and stay in Ξ. If $x(0) \in \Xi$, then $x(t) \in \Xi, \forall t > 0$. So, Ξ is an ω-invariant set, in R^{MN}/Ξ there exists no stable equilibrium point of the FCNN in Eq.(2.27). □

The following theorem guarantees that the type-I FCNN has at least one equilibrium point.

Theorem 4

The FCNN in Eq.(2.27) has at least one equilibrium point.

Proof: Let the right hand side of Eq.(2.27) be 0, then we have:

$$
\begin{aligned}
x_{ij} &= R_x[\sum_{C_{kl}\in N_r(i,j)} A(i,j;k,l)y_{kl} + \sum_{C_{kl}\in N_r(i,j)} B(i,j;k,l)u_{kl} + I \\
&+ \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} \mu_{A_{fmin}(i,j;k,l)}(y_{kl}) + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} \mu_{A_{fmax}(i,j;k,l)}(y_{kl}) \\
&+ \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} \mu_{B_{fmin}(i,j;k,l)}(u_{kl}) + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} \mu_{B_{fmax}(i,j;k,l)}(u_{kl})] \\
&, 1 \le i \le M, 1 \le j \le N
\end{aligned} \tag{3.26}
$$

Consider the following vector operator:

$$
\Phi = (\phi_{ij}(\mathbf{x}))_{MN} \tag{3.27}
$$

where

$$
\begin{aligned}
\phi_{ij}(\mathbf{x}) &= R_x[\sum_{C_{kl}\in N_r(i,j)} A(i,j;k,l)y_{kl} + \sum_{C_{kl}\in N_r(i,j)} B(i,j;k,l)u_{kl} + I \\
&+ \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} \mu_{A_{fmin}(i,j;k,l)}(y_{kl}) + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} \mu_{A_{fmax}(i,j;k,l)}(y_{kl}) \\
&+ \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} \mu_{B_{fmin}(i,j;k,l)}(u_{kl}) + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} \mu_{B_{fmax}(i,j;k,l)}(u_{kl})] \\
&, 1 \le i \le M, 1 \le j \le N
\end{aligned} \tag{3.28}
$$

and $\mathbf{x} = col(x_{11}, ..., x_{MN})$.

Let

$$
\begin{aligned}
M &= max\Big\{R_x\Big[\sum_{C_{kl}\in N_r(i,j)} |A(i,j;k,l)| \\
&\quad + \sum_{C_{kl}\in N_r(i,j)} |B(i,j;k,l)| + |I| + 4\Big]\Big\}
\end{aligned} \tag{3.29}
$$

Then the vector operator Φ maps the following set

$$
S = \{\mathbf{x} | |x_{ij}| \le M, 1 \le i \le M, 1 \le j \le N\} \tag{3.30}
$$

into itself. Since S is a convex compact set, from Brown's fix-point theorem, we know $\Phi : S \longmapsto S$ has at least one fix-point $\mathbf{x} = \mathbf{x}^*$. And $\mathbf{x}^*$ is an equilibrium point of FCNN in Eq.(2.27). □

3.2 Global stability

3.2.1 Results for type-II FCNN

Since every pixel of input image can be viewed as a *fuzzy singleton*, we can choose $\mu_u(\cdot)$ as $\mu_u(x) = x$, then the type-II FCNN in Eq.(2.31) can be rewritten as:

$$
C\frac{dx_{ij}}{dt} = -\frac{1}{R_x}x_{ij} + \sum_{C_{kl}\in N_r(i,j)} A(i,j;k,l)y_{kl} + \sum_{C_{kl}\in N_r(i,j)} B(i,j;k,l)u_{kl} + I
$$

$$+ \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} A_{fmin}(i,j;k,l)y_{kl} + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} A_{fmax}(i,j;k,l)y_{kl}$$
$$+ \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} B_{fmin}(i,j;k,l)u_{kl} + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} B_{fmax}(i,j;k,l)u_{kl}$$
$$, 1 \le i \le M, 1 \le j \le N \tag{3.31}$$

Output equation of C_{ij} is given by:

$$y_{ij}(t) = f\big(x_{ij}(t)\big) = \frac{1}{2}\big(|x_{ij}(t)+1| - |x_{ij}(t)-1|\big)$$
$$, 1 \le i \le M, 1 \le j \le N \tag{3.32}$$

Constraint conditions are given by:

$$\big|x_{ij}(0)\big| \le 1, \quad \big|u_{ij}\big| \le 1, \quad 1 \le i \le M, 1 \le j \le N \tag{3.33}$$

Parameter assumptions:

$$C > 0,\ R_x > 0$$
$$A(i,j;k,l) = A(k,l;i,j),\ A_{fmin}(i,j;k,l) = A_{fmin}(k,l;i,j),$$
$$A_{fmax}(i,j;k,l) = A_{fmax}(k,l;i,j)$$
$$1 \le i \le M, 1 \le j \le N \tag{3.34}$$

In state equation Eq.(3.31), if there exits no fuzzy logical relation between two cells C_{ij} and C_{kl}, then we say that the fuzzy connections between them are *non-existed*, else we say that the fuzzy connections between them are *existed*. We only study the FCNN with flat fuzzy feedback MIN templates and flat fuzzy feedback Max templates. A *flat* fuzzy feedback MIN template is defined by:

$$A_{fmin}(i,j;k,l) = \alpha, \forall C_{kl} \in N_r(i,j) \text{ and } A_{fmin}(i,j;k,l) \text{ is existed.} \tag{3.35}$$

where α is a constant. A flat fuzzy feedback Max template is defined by:

$$A_{fmax}(i,j;k,l) = \beta, \forall C_{kl} \in N_r(i,j) \text{ and } A_{fmax}(i,j;k,l) \text{ is existed.} \tag{3.36}$$

where β is a constant.

Then Eq.(3.31) can be rewritten as:

$$C\frac{dx_i}{dt} = -\frac{1}{R_x}x_i + \sum_{j=1}^{MN} a_{ij}f(x_j) + \sum_{j=1}^{MN} b_{ij}u_j + I + \tilde{\bigwedge}_{j=1}^{MN} \alpha_{ij}f(x_j)$$
$$+\tilde{\bigvee}_{j=1}^{MN} \beta_{ij}f(x_j) + \tilde{\bigwedge}_{j=1}^{MN} b_{fmin}(ij)u_j + \tilde{\bigvee}_{j=1}^{MN} b_{fmax}(ij)u_j$$
$$, i = 1, 2, ..., MN \tag{3.37}$$

Where

$$\alpha_{ij} = \begin{cases} \alpha, & \text{if corresponding } A_{fmin}(i,j;k,l) \text{ is existed.} \\ \text{undefined}, & \text{if corresponding } A_{fmin}(i,j;k,l) \text{ is nonexisted.} \end{cases} \tag{3.38}$$

$$\beta_{ij} = \begin{cases} \beta, & \text{if corresponding } A_{fmax}(i,j;k,l) \text{ is existed.} \\ \text{undefined}, & \text{if corresponding } A_{fmax}(i,j;k,l) \text{ is nonexisted.} \end{cases} \tag{3.39}$$

From parameter assumptions in Eq.(3.34), one can see that

$$a_{ij} = a_{ji}, \alpha_{ij} = \alpha_{ji}, \beta_{ij} = \beta_{ji} \tag{3.40}$$

We then have the following corollary:

Proposition 1

Suppose that $\mathbf{x}$ and $\mathbf{x}'$ are two states of FCNN in Eq.(3.37), then we have:

(1)

$$\left| \tilde{\bigwedge}_{j=1}^{MN} \alpha_{ij} f(x_j) - \tilde{\bigwedge}_{j=1}^{MN} \alpha_{ij} f(x'_j) \right| \leq \sum_{j=1}^{MN} |\alpha_{ij}| \left| f(x_j) - f(x'_j) \right| \tag{3.41}$$

(2)

$$\left| \tilde{\bigvee}_{j=1}^{MN} \beta_{ij} f(x_j) - \tilde{\bigvee}_{j=1}^{MN} \beta_{ij} f(x'_j) \right| \leq \sum_{j=1}^{MN} |\beta_{ij}| \left| f(x_j) - f(x'_j) \right| \tag{3.42}$$

Proof:

(1) Suppose that there exist k and l such that

$$\tilde{\bigwedge}_{j=1}^{MN} \alpha_{ij} f(x_j) = \alpha_{ik} f(x_k) \tag{3.43}$$

$$\tilde{\bigwedge}_{j=1}^{MN} \alpha_{ij} f(x'_j) = \alpha_{il} f(x'_l) \tag{3.44}$$

then we have

$$\begin{aligned} & \left| \tilde{\bigwedge}_{j=1}^{MN} \alpha_{ij} f(x_j) - \tilde{\bigwedge}_{j=1}^{MN} \alpha_{ij} f(x'_j) \right| \\ \leq & \max\left(\left| \alpha_{ik} f(x_k) - \alpha_{ik} f(x'_k) \right|, \left| \alpha_{il} f(x_l) - \alpha_{il} f(x'_l) \right| \right) \\ \leq & \sum_{j=1}^{MN} |\alpha_{ij}| \left| f(x_j) - f(x'_j) \right| \end{aligned} \tag{3.45}$$

(2) Suppose that there exist k and l such that

$$\tilde{\bigvee}_{j=1}^{MN} \beta_{ij} f(x_j) = \beta_{ik} f(x_k) \tag{3.46}$$

$$\tilde{\bigvee}_{j=1}^{MN} \beta_{ij} f(x'_j) = \beta_{il} f(x'_l) \tag{3.47}$$

then we have

$$\begin{aligned}
&\left|\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x_j)-\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x'_j)\right| \\
\leq\ & \max\left(\left|\beta_{ik}f(x_k)-\beta_{ik}f(x'_k)\right|,\left|\beta_{il}f(x_l)-\beta_{il}f(x'_l)\right|\right) \\
\leq\ & \sum_{j=1}^{MN}|\beta_{ij}|\left|f(x_j)-f(x'_j)\right| \qquad (3.48)
\end{aligned}$$

□

Let

$$|A|=\left(|a_{ij}|+|\alpha_{ij}|+|\beta_{ij}|\right)_{MN\times MN} \qquad (3.49)$$

then we have the following theorem.

Theorem 5

Suppose that the spectral radius of matrix $R_x|A|$, $\rho\left(R_x|A|\right)<1$, then the type-II FCNN in Eq.(3.37) has only one equilibrium point, and this equilibrium point is globally stable.

Proof: The existence of equilibrium point of FCNN in Eq.(3.37) is guaranteed by Theorem 2. Now, we only need to prove that the FCNN has less than two equilibrium points. Let the right hand side of Eq.(3.37) be 0, we have:

$$\begin{aligned}
x_i \ =\ & R_x\sum_{j=1}^{MN}a_{ij}f(x_j)+R_x\sum_{j=1}^{MN}b_{ij}u_j+R_xI+R_x\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}f(x_j) \\
& +R_x\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x_j)+R_x\tilde{\bigwedge}_{j=1}^{MN}b_{fmin}(ij)u_j+R_x\tilde{\bigvee}_{j=1}^{MN}b_{fmax}(ij)u_j \\
& ,i=1,2,...,MN \qquad (3.50)
\end{aligned}$$

Let $\mathbf{x}^{(1)}=col\left(x_1^{(1)},...,x_{MN}^{(1)}\right)$ and $\mathbf{x}^{(2)}=col\left(x_1^{(2)},...,x_{MN}^{(2)}\right)$ be two solutions of Eq.(3.37), then we have:

$$\begin{aligned}
&\left|x_i^{(1)}-x_i^{(2)}\right| \\
\leq\ & R_x\sum_{j=1}^{MN}|a_{ij}|\left|f(x_j^{(1)})-f(x_j^{(2)})\right|+\left|R_x\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}f(x_j^{(1)})\right. \\
& \left.-R_x\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}f(x_j^{(2)})\right|+\left|R_x\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x_j^{(1)})-R_x\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x_j^{(2)})\right| \\
\leq\ & R_x\sum_{j=1}^{MN}|a_{ij}|\left|f(x_j^{(1)})-f(x_j^{(2)})\right|+R_x\sum_{j=1}^{MN}|\alpha_{ij}|\left|f(x_j^{(1)})-f(x_j^{(2)})\right| \\
& +R_x\sum_{j=1}^{MN}|\beta_{ij}|\left|f(x_j^{(1)})-f(x_j^{(2)})\right| \\
\leq\ & R_x\sum_{j=1}^{MN}\left(|a_{ij}|+|\alpha_{ij}|+|\beta_{ij}|\right)|x_j^{(1)}-x_j^{(2)}| \qquad (3.51)
\end{aligned}$$

The second inequality is in view of Proposition 1.

We then write Eq.(3.51) into a vector form as follows:

$$\begin{aligned} & col\Big(|x_1^{(1)}-x_1^{(2)}|,...,|x_{MN}^{(1)}-x_{MN}^{(2)}|\Big) \\ \leq\ & R_x|A|col\Big(|x_1^{(1)}-x_1^{(2)}|,...,|x_{MN}^{(1)}-x_{MN}^{(2)}|\Big) \\ \leq\ & ... \leq \Big(R_x|A|\Big)^m col\Big(|x_1^{(1)}-x_1^{(2)}|,...,|x_{MN}^{(1)}-x_{MN}^{(2)}|\Big) \end{aligned} \tag{3.52}$$

Since $\rho\Big(R_x|A|\Big) < 1$, we have

$$\lim_{m\to\infty}\Big(R_x|A|\Big)^m = 0 \tag{3.53}$$

Then we have:

$$\mathbf{x}^{(1)} = \mathbf{x}^{(2)} \tag{3.54}$$

which yields that the FCNN in Eq.(3.37) has only one equilibrium point, $\mathbf{x}^*$.

Since $\mathbf{x}^*$ is the only equilibrium point, followed Eq.(3.37) we have:

$$\begin{aligned} C\frac{d(x_i-x_i^*)}{dt} &= -\frac{1}{R_x}(x_i-x_i^*)+\sum_{j=1}^{MN}a_{ij}(f(x_j)-f(x_j^*)) \\ &+\left[\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}f(x_j)-\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}f(x_j^*)\right] \\ &+\left[\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x_j)-\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x_j^*)\right] \\ &, i=1,2,...,MN \end{aligned} \tag{3.55}$$

Since $\rho\Big(R_x|A|\Big) < 1$, $\Big(E-R_x|A|\Big)$ is an M-matrix, where E is the unity matrix. So, there exits a group of positive constants, $p_i > 0, i = 1,2,...,MN$, such that:

$$\begin{aligned} &-\frac{p_j}{R_x}+\sum_{i=1}^{MN}p_i\Big(|a_{ij}|+|\alpha_{ij}|+|\beta_{ij}|\Big) < 0 \\ &, j=1,2,...,MN \end{aligned} \tag{3.56}$$

We construct the following Lyapunov function:

$$V(\mathbf{x}) = \frac{1}{C}\sum_{j=1}^{MN}p_j|x_j-x_j^*| > 0 \tag{3.57}$$

When $\mathbf{x}=\mathbf{x}^*$, we have $V(\mathbf{x})=0$. When $|x_j-x_j^*|\to+\infty$, we have $V(\mathbf{x})\to+\infty$.

Along the solution of Eq.(3.55), we calculate the Dini upper-right differential of $V(\mathbf{x})$ as:

$$\begin{aligned} & D^+V(\mathbf{x})\Big|_{Eq.(3.55)} \\ \leq\ & \sum_{j=1}^{MN}p_j\left(-\frac{1}{R_x}|x_j-x_j^*|+\sum_{i=1}^{MN}|a_{ji}|\Big|f(x_i)-f(x_i^*)\Big|+\Big|\tilde{\bigwedge}_{i=1}^{MN}\alpha_{ji}f(x_i)\right. \end{aligned}$$

$$
\begin{aligned}
& -\tilde{\bigwedge}_{i=1}^{MN}\alpha_{ji}f(x_i^*)\Big| + \Big|\tilde{\bigvee}_{i=1}^{MN}\beta_{ji}f(x_i) - \tilde{\bigvee}_{i=1}^{MN}\beta_{ji}f(x_i^*)\Big|\Big) \\
\leq & \sum_{j=1}^{MN} p_j\Big(-\frac{1}{R_x}|x_j - x_j^*| + \sum_{i=1}^{MN}|a_{ji}|\Big|f(x_i) - f(x_i^*)\Big| \\
& + \sum_{i=1}^{MN}|\alpha_{ji}|\Big|f(x_i) - f(x_i^*)\Big| + \sum_{i=1}^{MN}|\beta_{ji}|\Big|f(x_i) - f(x_i^*)\Big|\Big) \\
\leq & \sum_{j=1}^{MN} -\frac{p_j}{R_x}\Big|x_j - x_j^*\Big| + \sum_{j=1}^{MN} p_j \sum_{i=1}^{MN}\Big(|a_{ji}| + |\alpha_{ji}| + |\beta_{ji}|\Big)|x_i - x_i^*| \\
= & \sum_{j=1}^{MN} -\frac{p_j}{R_x}\Big|x_j - x_j^*\Big| + \sum_{i=1}^{MN} p_i \sum_{j=1}^{MN}\Big(|a_{ij}| + |\alpha_{ij}| + |\beta_{ij}|\Big)|x_j - x_j^*| \\
= & \sum_{j=1}^{MN}\Big[-\frac{p_j}{R_x} + \sum_{i=1}^{MN} p_i\Big(|a_{ij}| + |\alpha_{ij}| + |\beta_{ij}|\Big)\Big]|x_j - x_j^*| \\
< & \ 0
\end{aligned}
\tag{3.58}
$$

The second inequality is in view of Proposition 1. The second equality is in view of parameter assumption in Eq.(3.40). The last inequality is satisfied when $\mathbf{x} \neq \mathbf{x}^*$. □

Similarly, we can have the following theorem:

Theorem 6

Suppose that the following matrix

$$
\begin{aligned}
& diag\Big(-\frac{1}{R_x} + a_{ii} + |\alpha_{ii}| + |\beta_{ii}|\Big)_{MN\times MN} \\
& + \Big((1-\delta_{ij})(|a_{ij}| + |\alpha_{ij}| + |\beta_{ij}|)\Big)_{MN\times MN}
\end{aligned}
$$

is a Hurwitz matrix, then the equilibrium point $\mathbf{x} = \mathbf{x}^*$ is globally stable. Here

$$
\delta_{ij} = \begin{cases} 1 & , i = j \\ 0 & , i \neq j \end{cases} \tag{3.59}
$$

Proof: Since

$$
\begin{aligned}
& diag\Big(-\frac{1}{R_x} + a_{ii} + |\alpha_{ii}| + |\beta_{ii}|\Big)_{MN\times MN} \\
& + \Big((1-\delta_{ij})(|a_{ij}| + |\alpha_{ij}| + |\beta_{ij}|)\Big)_{MN\times MN}
\end{aligned}
$$

is a Hurwitz matrix, we have

$$
\begin{aligned}
& -diag\Big(-\frac{1}{R_x} + a_{ii} + |\alpha_{ii}| + |\beta_{ii}|\Big)_{MN\times MN} \\
& -\Big((1-\delta_{ij})(|a_{ij}| + |\alpha_{ij}| + |\beta_{ij}|)\Big)_{MN\times MN}
\end{aligned}
$$

is an M-matrix. Then, from properties of M-matrix, there exits a group of positive constants, $p_i > 0, i = 1, 2, ..., MN$, such that:

$$
p_j(-\frac{1}{R_x} + a_{jj} + |\alpha_{jj}| + |\beta_{jj}|)
$$

$$+\sum_{i=1}^{MN} p_i(1-\delta_{ij})(|a_{ij}|+|\alpha_{ij}|+|\beta_{ij}|) < 0$$
$$, j = 1, 2, ..., MN \tag{3.60}$$

We construct the following Lyapunov function:

$$V(\mathbf{x}) = \frac{1}{C}\sum_{j=1}^{MN} p_j(x_j - x_j^*)sgn(x_j - x_j^*) > 0 \tag{3.61}$$

where $sgn(\cdot)$ is the signum function. When $\mathbf{x} = \mathbf{x}^*$, we have $V(\mathbf{x}) = 0$. When $|x_j - x_j^*| \to +\infty$, we have $V(\mathbf{x}) \to +\infty$.

Along the solution of Eq.(3.55), we calculate the Dini upper-right differential of $V(\mathbf{x})$ as:

$$\begin{aligned}
& D^+V(\mathbf{x})\Big|_{Eq.(3.55)} \\
= & \sum_{j=1}^{MN} p_j\Big[-\frac{1}{R_x}(x_j - x_j^*) + \sum_{i=1}^{MN} a_{ji}(f(x_i) - f(x_i^*)) + (\tilde{\bigwedge}_{i=1}^{MN} \alpha_{ji}f(x_i) \\
& -\tilde{\bigwedge}_{i=1}^{MN} \alpha_{ji}f(x_i^*)) + (\tilde{\bigvee}_{i=1}^{MN} \beta_{ji}f(x_i) - \tilde{\bigvee}_{i=1}^{MN} \beta_{ji}f(x_i^*))\Big]sgn(x_j - x_j^*) \\
\leq & \sum_{j=1}^{MN} p_j\Big[-\frac{1}{R_x}|x_j - x_j^*| + a_{jj}|f(x_j) - f(x_j^*)| \\
& + \sum_{i=1}^{MN}(1-\delta_{ji})|a_{ji}||f(x_i) - f(x_i^*)| \\
& + |\tilde{\bigwedge}_{i=1}^{MN} \alpha_{ji}f(x_i) - \tilde{\bigwedge}_{i=1}^{MN} \alpha_{ji}f(x_i^*)| \\
& + |\tilde{\bigvee}_{i=1}^{MN} \beta_{ji}f(x_i) - \tilde{\bigvee}_{i=1}^{MN} \beta_{ji}f(x_i^*)|\Big] \\
\leq & \sum_{j=1}^{MN} p_j\Big[-\frac{1}{R_x}|x_j - x_j^*| + a_{jj}|f(x_j) - f(x_j^*)| \\
& + \sum_{i=1}^{MN}(1-\delta_{ji})|a_{ji}||f(x_i) - f(x_i^*)| \\
& + \sum_{i=1}^{MN}|\alpha_{ji}||f(x_i) - f(x_i^*)| + \sum_{i=1}^{MN}|\beta_{ji}||f(x_i) - f(x_i^*)|\Big] \\
\leq & \sum_{j=1}^{MN} p_j(-\frac{1}{R_x} + a_{jj} + |\alpha_{jj}| + |\beta_{jj}|)|f(x_j) - f(x_j^*)| \\
& + \sum_{j=1}^{MN} p_j \sum_{i=1}^{MN}(1-\delta_{ji})(|a_{ji}| + |\alpha_{ji}| + |\beta_{ji}|)|f(x_i) - f(x_i^*)| \\
= & \sum_{j=1}^{MN} p_j(-\frac{1}{R_x} + a_{jj} + |\alpha_{jj}| + |\beta_{jj}|)|f(x_j) - f(x_j^*)| \\
& + \sum_{i=1}^{MN} p_i \sum_{j=1}^{MN}(1-\delta_{ij})(|a_{ij}| + |\alpha_{ij}| + |\beta_{ij}|)|f(x_j) - f(x_j^*)|
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{j=1}^{MN}\Big[p_j(-\frac{1}{R_x}+a_{jj}+|\alpha_{jj}|+|\beta_{jj}|) \\
&\quad +\sum_{i=1}^{MN} p_i(1-\delta_{ij})(|a_{ij}|+|\alpha_{ij}|+|\beta_{ij}|)\Big]|f(x_j)-f(x_j^*)| \\
&< 0
\end{aligned} \tag{3.62}
$$

To get the first inequality, one should note that

$$a_{jj}(f(x_j)-f(x_j^*))sgn(x_j-x_j^*) = a_{jj}|f(x_j)-f(x_j^*)| \tag{3.63}$$

The second inequality is in view of Proposition 1. To get the third inequality, one should note that

$$-|x_j-x_j^*| \le -|f(x_j)-f(x_j^*)| \tag{3.64}$$

The second equality is from parameter assumption in Eq.(3.40). The last inequality is satisfied when $\mathbf{x} \neq \mathbf{x}^*$. □

3.2.2 Results for type-I FCNN

We recast the type-I FCNN in Eq.(2.27) into:

$$
\begin{aligned}
C\frac{dx_i}{dt} &= -\frac{1}{R_x}x_i+\sum_{j=1}^{MN} a_{ij}f(x_j)+\sum_{j=1}^{MN} b_{ij}u_j+I \\
&\quad +\tilde{\bigwedge}_{j=1}^{MN}\mu_{A_{fmin}(ij)}(y_j) \\
&\quad +\tilde{\bigvee}_{j=1}^{MN}\mu_{A_{fmax}(ij)}(y_j) \\
&\quad +\tilde{\bigwedge}_{j=1}^{MN}\mu_{B_{fmin}(ij)}(u_j) \\
&\quad +\tilde{\bigvee}_{j=1}^{MN}\mu_{B_{fmax}(ij)}(u_j) \\
&\quad , i=1,2,...,MN
\end{aligned} \tag{3.65}
$$

We need the following proposition:

Proposition 2

Assume that for two points $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ there exists a k such that:

$$\tilde{\bigwedge}_{j=1}^{MN}\mu_{A_{fmin}(ij)}(f(x_i^{(2)})) = \mu_{A_{fmin}(ik)}(f(x_k^{(2)})) \tag{3.66}$$

then

$$
\begin{aligned}
&|\tilde{\bigwedge}_{j=1}^{MN}\mu_{A_{fmin}(ij)}(f(x_i^{(1)})) - \tilde{\bigwedge}_{j=1}^{MN}\mu_{A_{fmin}(ij)}(f(x_i^{(2)}))| \\
\le\ & |\mu_{A_{fmin}(ik)}(f(x_k^{(1)})) - \mu_{A_{fmin}(ik)}(f(x_k^{(2)}))|
\end{aligned} \tag{3.67}
$$

Similarly, let there exists an l such that:

$$\tilde{\bigvee}_{j=1}^{MN}\mu_{A_{fmax}(ij)}(f(x_i^{(1)})) = \mu_{A_{fmax}(ik)}(f(x_k^{(1)})) \tag{3.68}$$

then we have:

$$\begin{aligned} & |\tilde{\bigvee}_{j=1}^{MN} \mu_{A_{fmax(ij)}}(f(x_i^{(1)})) - \tilde{\bigvee}_{j=1}^{MN} \mu_{A_{fmin(ij)}}(f(x_i^{(2)}))| \\ \leq\ & |\mu_{A_{fmax(il)}}(f(x_l^{(1)})) - \mu_{A_{fmax(il)}}(f(x_l^{(2)}))| \end{aligned} \tag{3.69}$$

Proof: 1. Assume that there exists an h such that

$$\tilde{\bigwedge}_{j=1}^{MN} \mu_{A_{fmin(ij)}}(f(x_i^{(1)})) = \mu_{A_{fmin(ih)}}(f(x_h^{(1)})) \tag{3.70}$$

We have

$$\begin{aligned} & |\tilde{\bigwedge}_{j=1}^{MN} \mu_{A_{fmin(ij)}}(f(x_i^{(1)})) - \tilde{\bigwedge}_{j=1}^{MN} \mu_{A_{fmin(ij)}}(f(x_i^{(2)}))| \\ =\ & |\mu_{A_{fmin(ih)}}(f(x_h^{(1)})) - \mu_{A_{fmin(ik)}}(f(x_k^{(2)}))| \\ \leq\ & |\mu_{A_{fmin(ik)}}(f(x_k^{(1)})) - \mu_{A_{fmin(ik)}}(f(x_k^{(2)}))| \end{aligned} \tag{3.71}$$

2. Assume that there exists an h such that

$$\tilde{\bigvee}_{j=1}^{MN} \mu_{A_{fmax(ij)}}(f(x_i^{(2)})) = \mu_{A_{fmax(ih)}}(f(x_h^{(2)})) \tag{3.72}$$

We have

$$\begin{aligned} & |\tilde{\bigvee}_{j=1}^{MN} \mu_{A_{fmax(ij)}}(f(x_i^{(1)})) - \tilde{\bigvee}_{j=1}^{MN} \mu_{A_{fmin(ij)}}(f(x_i^{(2)}))| \\ =\ & |\mu_{A_{fmax(il)}}(f(x_l^{(1)})) - \mu_{A_{fmax(ih)}}(f(x_h^{(2)}))| \\ \leq\ & |\mu_{A_{fmax(il)}}(f(x_l^{(1)})) - \mu_{A_{fmax(il)}}(f(x_l^{(2)}))| \end{aligned} \tag{3.73}$$

□

Assume that all the $\mu_{A_{fmin(ij)}}(\cdot)$ and $\mu_{A_{fmax(ij)}}(\cdot)$ are globally Lipschitzian, i.e., for any y_j and y_j' we have:

$$|\mu_{A_{fmin(ij)}}(y_j) - \mu_{A_{fmin(ij)}}(y_j')| \leq \alpha_{ij}^M |y_j - y_j'| \tag{3.74}$$

and

$$|\mu_{A_{fmax(ij)}}(y_j) - \mu_{A_{fmax(ij)}}(y_j')| \leq \beta_{ij}^M |y_j - y_j'| \tag{3.75}$$

And we assume that

$$a_{ij} = a_{ji}, \alpha_{ij}^M = \alpha_{ji}^M, \beta_{ij}^M = \beta_{ji}^M \tag{3.76}$$

In this case, we let

$$|A| = \left(|a_{ij}| + |\alpha_{ij}^M| + |\beta_{ij}^M|\right)_{MN \times MN} \tag{3.77}$$

then we have the following theorem.

Theorem 7

Suppose that the spectral radius of matrix $R_x|A|$, $\rho\Big(R_x|A|\Big) < 1$, then the type-I FCNN in Eq.(3.65) has only one equilibrium point, and this equilibrium point is globally stable.

Proof: The existence of equilibrium point of FCNN in Eq.(3.65) is guaranteed by Theorem 4. Now, we only need to prove that the FCNN has less than two equilibrium points. Let the right hand side of Eq.(3.65) be 0, we have:

$$\begin{aligned} x_i &= R_x \sum_{j=1}^{MN} a_{ij} f(x_j) + R_x \sum_{j=1}^{MN} b_{ij} u_j + R_x I + R_x \tilde{\bigwedge}_{j=1}^{MN} \mu_{A_{fmin(ij)}}\Big(f(x_j)\Big) \\ &\quad + R_x \tilde{\bigvee}_{j=1}^{MN} \mu_{A_{fmax(ij)}}\Big(f(x_j)\Big) + R_x \tilde{\bigwedge}_{j=1}^{MN} \mu_{B_{fmin(ij)}}(u_j) \\ &\quad + R_x \tilde{\bigvee}_{j=1}^{MN} \mu_{B_{fmax(ij)}}(u_j) \\ &\quad , i = 1, 2, ..., MN \end{aligned} \tag{3.78}$$

Let $\mathbf{x}^{(1)} = col\Big(x_1^{(1)}, ..., x_{MN}^{(1)}\Big)$ and $\mathbf{x}^{(2)} = col\Big(x_1^{(2)}, ..., x_{MN}^{(2)}\Big)$ be two solutions of Eq.(3.65), then we have:

$$\begin{aligned} & \Big|x_i^{(1)} - x_i^{(2)}\Big| \\ \leq\ & R_x \sum_{j=1}^{MN} |a_{ij}| \Big|f(x_j^{(1)}) - f(x_j^{(2)})\Big| \\ & + \Big| R_x \tilde{\bigwedge}_{j=1}^{MN} \mu_{A_{fmin(ij)}}(f(x_j^{(1)})) - R_x \tilde{\bigwedge}_{j=1}^{MN} \mu_{A_{fmin(ij)}}(f(x_j^{(2)})) \Big| \\ & + \Big| R_x \tilde{\bigvee}_{j=1}^{MN} \mu_{A_{fmax(ij)}}(f(x_j^{(1)})) - R_x \tilde{\bigvee}_{j=1}^{MN} \mu_{A_{fmax(ij)}}(f(x_j^{(2)})) \Big| \\ \leq\ & R_x \sum_{j=1}^{MN} |a_{ij}| \Big|f(x_j^{(1)}) - f(x_j^{(2)})\Big| \\ & + R_x |\mu_{A_{fmin(ik)}}\Big(f(x_k^{(1)})\Big) - \mu_{A_{fmin(ik)}}\Big(f(x_k^{(2)})\Big)| \\ & + R_x |\mu_{A_{fmin(il)}}\Big(f(x_l^{(1)})\Big) - \mu_{A_{fmin(il)}}\Big(f(x_l^{(2)})\Big)| \\ \leq\ & R_x \sum_{j=1}^{MN} \Big(|a_{ij}| + |\alpha_{ij}^M| + |\beta_{ij}^M|\Big) |x_j^{(1)} - x_j^{(2)}| \end{aligned} \tag{3.79}$$

The second inequality is in view of Proposition 2. The rest of the proof is similar to that of Theorem 5. □

Similar to Theorem 6, we have the following theorem:

Theorem 8

Suppose that the following matrix

$$diag\Big(-\frac{1}{R_x}+a_{ii}+\alpha_{ii}^M+\beta_{ii}^M\Big)_{MN\times MN}$$
$$+\Big((1-\delta_{ij})(|a_{ij}|+\alpha_{ij}^M+\beta_{ij}^M)\Big)_{MN\times MN}$$

is a Hurwitz matrix, then the equilibrium point $\mathbf{x}=\mathbf{x}^*$ is globally stable. Here

$$\delta_{ij}=\begin{cases}1 & ,i=j\\ 0 & ,i\neq j\end{cases} \tag{3.80}$$

3.3 Local stability

3.3.1 Results for type-II FCNN

In the FCNN as in Eq.(3.37), the state variable of every cell, $x_i(i=1,...,MN)$, can stay in three different intervals: $(-\infty,-1]$, $(-1,1)$ and $[1,\infty)$, which correspond to three different cell outputs: -1, x_i and 1. So, the state space of the FCNN can be divided into 3^{MN} separated regions, $D_i(i=1,...,3^{MN})$. Every D_i is an MN-dimensional hypercube.

Suppose that $\mathbf{x}^*$ is an equilibrium point of the FCNN in Eq.(3.37), and it is an inner point of D_k. Then there exists a neighborhood of $\mathbf{x}^*$, which is in the interior of D_k. Let

$$G\triangleq\{x:\sum_{i=1}^{MN}(x_i-x_i^*)^2<\delta^2\}\subset D_k \tag{3.81}$$

be the biggest hyper-ball in D_k. We can rewrite Eq.(3.37) in G as follows:

$$\begin{aligned}
&C\frac{d(x_i-x_i^*)}{dt}\\
=\;&-\frac{1}{R_x}(x_i-x_i^*)+\sum_{j=1}^{MN}a_{ij}(f(x_j)-f(x_j^*))\\
&+[\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}f(x_j)-\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}f(x_j^*)]+[\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x_j)-\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x_j^*)]\\
=\;&-\frac{1}{R_x}(x_i-x_i^*)+\sum_{j=1}^{MN}a_{ij}\sigma(x_j^*)(x_j-x_j^*)\\
&+[\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}f(x_j)-\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}f(x_j^*)]+[\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x_j)-\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x_j^*)]\\
&,i=1,2,...,MN
\end{aligned} \tag{3.82}$$

then we have the following theorem.

Theorem 9

Suppose that the following matrix

$$diag\Big(-\frac{1}{R_x}+\sigma(x_i^*)(a_{ii}+|\alpha_{ii}|+|\beta_{ii}|)\Big)_{MN\times MN}$$
$$+\Big((1-\delta_{ij})\sigma(x_j^*)(|a_{ij}|+|\alpha_{ij}|+|\beta_{ij}|)\Big)_{MN\times MN}$$

is a Hurwitz matrix, then the equilibrium point $\mathbf{x}=\mathbf{x}^*$ is asymptotically stable in the basin of attraction G. Here

$$\delta_{ij}=\begin{cases}1 & ,i=j\\ 0 & ,i\neq j\end{cases} \tag{3.83}$$

and

$$\sigma(x)=\begin{cases}1 & ,if\ \ |x|<1\\ 0 & ,if\ \ |x|\geq 1\end{cases} \tag{3.84}$$

Proof: Since

$$diag\Big(-\frac{1}{R_x}+\sigma(x_i^*)(a_{ii}+|\alpha_{ii}|+|\beta_{ii}|)\Big)_{MN\times MN}$$
$$+\Big((1-\delta_{ij})\sigma(x_j^*)(|a_{ij}|+|\alpha_{ij}|+|\beta_{ij}|)\Big)_{MN\times MN}$$

is a Hurwitz matrix, we know that

$$-diag(-\frac{1}{R_x}+\sigma(x_i^*)(a_{ii}+|\alpha_{ii}|+|\beta_{ii}|))_{MN\times MN}$$
$$-((1-\delta_{ij})\sigma(x_j^*)(|a_{ij}|+|\alpha_{ij}|+|\beta_{ij}|))_{MN\times MN}$$

is an M-matrix. Then, from properties of M-matrix, there exits a group of positive constants, $p_i>0, i=1,2,...,MN$, such that:

$$p_j(-\frac{1}{R_x}+\sigma(x_j^*)(a_{jj}+|\alpha_{jj}|+|\beta_{jj}|))$$
$$+\sum_{i=1}^{MN}p_i(1-\delta_{ij})\sigma(x_j^*)(|a_{ij}|+|\alpha_{ij}|+|\beta_{ij}|)<0$$
$$,j=1,2,...,MN \tag{3.85}$$

We construct the following Lyapunov function:

$$V(\mathbf{x})=\frac{1}{C}\sum_{j=1}^{MN}p_j(x_j-x_j^*)sgn(x_j-x_j^*)>0 \tag{3.86}$$

where $sgn(\cdot)$ is the signum function.

Along the solution of Eq.(3.82), we calculate the Dini upper-right differential of $V(\mathbf{x})$ as:

$$
\begin{aligned}
& D^+V(\mathbf{x})\Big|_{eq.(3.82)} \\
= & \sum_{j=1}^{MN} p_j\Big[-\frac{1}{R_x}(x_j - x_j^*) + \sum_{i=1}^{MN} a_{ji}\sigma(x_i^*)(x_i - x_i^*) + (\tilde{\bigwedge}_{i=1}^{MN} \alpha_{ji} f(x_i) \\
& -\tilde{\bigwedge}_{i=1}^{MN} \alpha_{ji} f(x_i^*)) + (\tilde{\bigvee}_{i=1}^{MN} \beta_{ji} f(x_i) - \tilde{\bigvee}_{i=1}^{MN} \beta_{ji} f(x_i^*))\Big] sgn(x_j - x_j^*) \\
\leq & \sum_{j=1}^{MN} p_j\Big[-\frac{1}{R_x}|x_j - x_j^*| + a_{jj}\sigma(x_j^*)|x_j - x_j^*| + \sum_{i=1}^{MN}(1-\delta_{ji})\sigma(x_j^*)|a_{ji}||x_i - x_i^*| \\
& +|\tilde{\bigwedge}_{i=1}^{MN} \alpha_{ji} f(x_i) - \tilde{\bigwedge}_{i=1}^{MN} \alpha_{ji} f(x_i^*)| + |\tilde{\bigvee}_{i=1}^{MN} \beta_{ji} f(x_i) - \tilde{\bigvee}_{i=1}^{MN} \beta_{ji} f(x_i^*)|\Big] \\
\leq & \sum_{j=1}^{MN} p_j\Big[-\frac{1}{R_x}|x_j - x_j^*| + a_{jj}\sigma(x_j^*)|x_j - x_j^*| + \sum_{i=1}^{MN}(1-\delta_{ji})\sigma(x_i^*)|a_{ji}||x_i - x_i^*| \\
& + \sum_{i=1}^{MN} |\alpha_{ji}||f(x_i) - f(x_i^*)| + \sum_{i=1}^{MN} |\beta_{ji}||f(x_i) - f(x_i^*)|\Big] \\
\leq & \sum_{j=1}^{MN} p_j(-\frac{1}{R_x} + (a_{jj} + |\alpha_{jj}| + |\beta_{jj}|)\sigma(x_j^*))|x_j - x_j^*| \\
& + \sum_{j=1}^{MN} p_j \sum_{i=1}^{MN}(1-\delta_{ji})\sigma(x_i^*)(|a_{ji}| + |\alpha_{ji}| + |\beta_{ji}|)|x_i - x_i^*| \\
= & \sum_{j=1}^{MN} p_j(-\frac{1}{R_x} + (a_{jj} + |\alpha_{jj}| + |\beta_{jj}|)\sigma(x_j^*))|x_j - x_j^*| \\
& + \sum_{i=1}^{MN} p_i \sum_{j=1}^{MN}(1-\delta_{ij})\sigma(x_j^*)(|a_{ij}| + |\alpha_{ij}| + |\beta_{ij}|)|x_j - x_j^*| \\
= & \sum_{j=1}^{MN}\Big[p_j(-\frac{1}{R_x} + (a_{jj} + |\alpha_{jj}| + |\beta_{jj}|)\sigma(x_j^*)) \\
& + \sum_{i=1}^{MN} p_i(1-\delta_{ij})\sigma(x_j^*)(|a_{ij}| + |\alpha_{ij}| + |\beta_{ij}|)\Big]|x_j - x_j^*| \\
< & \ 0
\end{aligned}
\tag{3.87}
$$

The second inequality is in view of Proposition 1. The last equality is from parameter assumption in Eq.(3.40). The last inequality is satisfied when $\mathbf{x} \neq \mathbf{x}^*$ and $\mathbf{x}$ in hyper-ball G. So, G is the basin of attraction of the equilibrium point $\mathbf{x}^*$. □

3.3.2 Results for type-I FCNN

Similar to Theorem 9, we can get the following theorem for type-I FCNN:

Theorem 10

Suppose that the following matrix

$$diag\Big(-\frac{1}{R_x}+\sigma(x_i^*)(a_{ii}+\alpha_{ii}^M+\beta_{ii}^M)\Big)_{MN\times MN} \tag{3.88}$$

$$+\Big((1-\delta_{ij})\sigma(x_j^*)(|a_{ij}|+\alpha_{ij}^M+\beta_{ij})\Big)_{MN\times MN} \tag{3.89}$$

is a Hurwitz matrix, then the equilibrium point $\mathbf{x}=\mathbf{x}^*$ is asymptotically stable in the basin of attraction G. Here

$$\delta_{ij}=\begin{cases}1 & ,i=j\\ 0 & ,i\neq j\end{cases} \tag{3.90}$$

and

$$\sigma(x)=\begin{cases}1 & ,if\ \ |x|<1\\ 0 & ,if\ \ |x|\geq 1\end{cases} \tag{3.91}$$

3.4 Type-II Fuzzy DCNN

In [266], conventional DCNN was introduced. In [380], we presented the corresponding results of fuzzy DCNN(FDCNN). This section contains the main results of [380, 385].

An $M\times N$ type-II FDCNN is described by the following state equation:

$$\begin{aligned}
&C\frac{dx_{ij}(t)}{dt}\\
=\ &-\frac{1}{R_x}x_{ij}(t)+\sum_{C_{kl}\in N_r(ij)}A(i,j;k,l)y_{kl}(t)\\
&+\sum_{C_{kl}\in N_r(ij)}A^\tau(i,j;k,l)y_{kl}(t-\tau)+\sum_{C_{kl}\in N_r(ij)}B(i,j;k,l)u_{kl}(t)+I\\
&+\tilde{\bigwedge_{C_{kl}\in N_r(i,j)}}A_{fmin}(i,j;k,l)y_{kl}(t)+\tilde{\bigvee_{C_{kl}\in N_r(i,j)}}A_{fmax}(i,j;k,l)y_{kl}(t)\\
&+\tilde{\bigwedge_{C_{kl}\in N_r(i,j)}}A^\tau_{fmin}(i,j;k,l)y_{kl}(t-\tau)\\
&+\tilde{\bigvee_{C_{kl}\in N_r(i,j)}}A^\tau_{fmax}(i,j;k,l)y_{kl}(t-\tau)\\
&+\tilde{\bigwedge_{C_{kl}\in N_r(i,j)}}B_{fmin}(i,j;k,l)u_{kl}(t)+\tilde{\bigvee_{C_{kl}\in N_r(i,j)}}B_{fmax}(i,j;k,l)u_{kl}(t)\\
&,1\leq i\leq M,1\leq j\leq N
\end{aligned} \tag{3.92}$$

We repack the state variables x_{ij} into a vector $\mathbf{x}$ of size $n=MN$. Similarly, the input and output variables u_{ij} and y_{ij} are repacked into $\mathbf{u}$ and $\mathbf{y}$ using the same labeling order. The initial conditions for an FDCNN is given by:

$$x_{ij}(t)=x_{0ij}(t),\qquad t\in[-\tau,0] \tag{3.93}$$

We will assume that $x_{0ij}(t)$ is a continuous function.

Then we recast the state equations Eq.(3.92) into following functional differential equations (FDE):

$$\begin{aligned} C\dot{\mathbf{x}} &= CF(t,\mathbf{x}_t) = -\frac{1}{R_x}\mathbf{x}(t) + \tilde{A}(f(\mathbf{x}(t))) + \tilde{A}^\tau(f(\mathbf{x}(t-\tau))) \\ &\quad +\tilde{B}(\mathbf{u}(t)) + \mathbf{I} \\ &\quad +\tilde{A}_{min} \odot_{min} (f(\mathbf{x}(t))) + \tilde{A}^\tau_{min} \odot_{min} (f(\mathbf{x}(t-\tau))) \\ &\quad +\tilde{A}_{max} \odot_{max} (f(\mathbf{x}(t))) + \tilde{A}^\tau_{max} \odot_{max} (f(\mathbf{x}(t-\tau))) \\ &\quad +\tilde{B}_{min} \odot_{min} (\mathbf{u}(t)) + \tilde{B}_{max} \odot_{max} (\mathbf{u}(t)) \end{aligned} \tag{3.94}$$

$\mathbf{x}_t \in C_\tau$ is defined as

$$\mathbf{x}_t(\theta)_i = \mathbf{x}(t+\theta)_i, \qquad \theta \in [-\tau, 0] \tag{3.95}$$

3.4.1 Existence and Uniqueness of solutions

Proposition 3

Given the initial condition

$$\mathbf{x}_0(t) = \phi(t), \qquad \phi(t) \in C_\tau \tag{3.96}$$

then the FDCNN in Eq.(3.94) has a unique continuous solution for $t \in [0, \infty)$.

Proof: we need to show that Eq.(3.94) has a unique solution. First we show that $F(t, \mathbf{x}_t)$ is globally Lipschitzian, i.e.

$$|F(t,\psi) - F(t,\phi)| \leq L\,|\psi - \phi| \qquad \text{for all} \quad \psi, \phi \in C_\tau \quad \text{and all} \quad t \tag{3.97}$$

for some constant L. If we define

$$\begin{aligned} L &= \frac{1}{C}\Big(\sum_{i,j}\sum_{k,l}|A^\tau(i,j;k,l)| + \sum_{i,j}\sum_{k,l}|A(i,j;k,l)| \\ &+ \max_{i,j,k,l}\{|A_{min}(i,j;k,l)|\} + \max_{i,j,k,l}\{|A^\tau_{min}(i,j;k,l)|\} \\ &+ \max_{i,j,k,l}\{|A_{max}(i,j;k,l)|\} + \max_{i,j,k,l}\{|A^\tau_{max}(i,j;k,l)|\} + \frac{1}{R_x}\Big) \end{aligned} \tag{3.98}$$

, then L qualifies as our Lipschitz constant.

Since the input is continuous, $F(t,\psi)$ is continuous with respect to t for all ψ. The conclusion then follows from [73, page 308–309]. □

Proposition 4

If the initial condition are bounded by $K > 0$, then all states x_{ij} of a type-II FCNN in Eq.(3.92) are bounded for all time in absolute value by the sum:

$$\begin{aligned} M &= K + R_x|I| \\ &\quad +R_x \max_{i,j}\{\sum_{k,l}(|A^\tau(i,j;k,l)| + |A(i,j;k,l)| + |B(i,j;k,l)|)\} \\ &\quad +R_x \max_{i,j}\{\bigvee_{kl}|A_{fmax}(i,j,k,l)| + \bigvee_{kl}|A^\tau_{fmax}(i,j,k,l)| \\ &\quad +\bigvee_{kl}|A_{fmin}(i,j,k,l)| + \bigvee_{kl}|A^\tau_{fmin}(i,j,k,l)| \\ &\quad +\bigvee_{kl}|B_{fmin}(i,j,k,l)| + \bigvee_{kl}|B_{fmax}(i,j,k,l)|\} \end{aligned} \tag{3.99}$$

and the ω-limit points of $x_{ij}(t)$ are bounded in absolute value by $(M - K)$.

Proof: It is sufficient to follow the proof of theorem 1 of [47] to see that also in this case it is possible to recast the equations of the network in the same form of Eq. (4a) of [47]:

$$C\frac{dx_{ij}}{dt}(t) = -\frac{1}{R_x}x_{ij} + f_{ij}(t) + g_{ij}(t) \tag{3.100}$$

where f_{ij} depends only on $y_{kl}(t)$ and $y_{kl}(t-\tau)$ and g_{ij} depends only on the inputs and the bias, and for both it is possible to compute an upper bound in the same way as in [47]. □

3.4.2 Stability Results

Given two points $\mathbf{x}, \mathbf{x}^* \in R^{MN}$, and the following function, $\Sigma : R^{MN} \mapsto R^{MN}$, such that for $\Sigma(\mathbf{x}-\mathbf{x}^*) \triangleq (\sigma(x_i - x_i^*))_{MN\times 1}$ which is defined by:

$$\begin{aligned}\sigma(x_i - x_i^*) &\triangleq f(x_i) - f(x_i^*) \\ &= \begin{cases} x_i - x_i^*, & x_i, x_i^* \in [-1,1] \\ sgn(x_i) - x_i^*, & x_i \notin [-1,1], x_i^* \in [-1,1] \\ x_i - sgn(x_i^*), & x_i \in [-1,1], x_i^* \notin [-1,1] \end{cases}\end{aligned} \tag{3.101}$$

Suppose $\mathbf{x}^*$ is an equilibrium point and let $\mathbf{w} = \{w_i\}_{MN\times 1} \triangleq \mathbf{x} - \mathbf{x}^*$, then Eq.(3.92) can be rewritten into:

$$\begin{aligned} C\dot{w}_i &= -\frac{1}{R_x}w_i + \sum_j a_{ij}\sigma(w_j(t)) + \sum_j a_{ij}^\tau \sigma(w_j(t-\tau)) \\ &\quad + \tilde{\bigwedge_j}\alpha_{ij} f(x_j(t)) - \tilde{\bigwedge_j}\alpha_{ij} f(x_j^*(t)) \\ &\quad + \tilde{\bigwedge_j}\alpha_{ij}^\tau f(x_j(t-\tau)) - \tilde{\bigwedge_j}\alpha_{ij}^\tau f(x_j^*(t-\tau)) \\ &\quad + \tilde{\bigvee_j}\beta_{ij} f(x_j(t)) - \tilde{\bigvee_j}\beta_{ij} f(x_j^*(t)) \\ &\quad + \tilde{\bigvee_j}\beta_{ij}^\tau f(x_j(t-\tau)) - \tilde{\bigvee_j}\beta_{ij}^\tau f(x_j^*(t-\tau)) \\ &\quad i = 1, 2, ..., MN. \end{aligned} \tag{3.102}$$

Also, we study the stability of the type-II FDCNN with flat fuzzy templates. Similar to that in Proposition 1, we have the following proposition:

Proposition 5

Suppose $\mathbf{x}$ and $\mathbf{x}'$ are two solutions of type-II FDCNN in Eq.(3.102), then we have:

(1)

$$\begin{aligned} &\left|\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}^\tau f(x_j(t-\tau)) - \tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}^\tau f(x_j'(t-\tau))\right| \\ \leq &\sum_{j=1}^{MN}|\alpha_{ij}^\tau|\left|f(x_j(t-\tau)) - f(x_j'(t-\tau))\right| \end{aligned} \tag{3.103}$$

(2)

$$\begin{aligned}&\left|\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}^{\tau}f(x_j(t-\tau))-\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}^{\tau}f(x_j'(t-\tau))\right|\\ \leq\ &\sum_{j=1}^{MN}|\beta_{ij}^{\tau}|\Big|f(x_j(t-\tau))-f(x_j'(t-\tau))\Big|\end{aligned}\tag{3.104}$$

Let

$$|D|=\Big(|a_{ij}+a_{ij}^{\tau}|+|\alpha_{ij}|+|\beta_{ij}|+|\alpha_{ij}^{\tau}|+|\beta_{ij}^{\tau}|\Big)_{MN\times MN}\tag{3.105}$$

then we have the following theorem.

Theorem 11

Suppose that the spectral radius of matrix $R_x|D|$, $\rho\Big(R_x|D|\Big)<1$, then the FCNN in Eq.(3.102) has only one equilibrium point.

Proof: Let the right hand side of Eq.(3.102) be zero, then we have the corresponding equilibrium equation as:

$$\begin{aligned}&-\frac{1}{R_x}w_i^*+\sum_j a_{ij}\sigma(w_j^*(t))+\sum_j a_{ij}^{\tau}\sigma(w_j^*(t-\tau))\\ &+\tilde{\bigwedge_j}\alpha_{ij}f(x_j(t))-\tilde{\bigwedge_j}\alpha_{ij}f(x_j^*(t))\\ &+\tilde{\bigwedge_j}\alpha_{ij}^{\tau}f(x_j(t-\tau))-\tilde{\bigwedge_j}\alpha_{ij}^{\tau}f(x_j^*(t-\tau))\\ &+\tilde{\bigvee_j}\beta_{ij}f(x_j(t))-\tilde{\bigvee_j}\beta_{ij}f(x_j^*(t))\\ &+\tilde{\bigvee_j}\beta_{ij}^{\tau}f(x_j(t-\tau))-\tilde{\bigvee_j}\beta_{ij}^{\tau}f(x_j^*(t-\tau))=0,\\ &i=1,2,...,MN.\end{aligned}\tag{3.106}$$

At the equilibrium point, we have:

$$\begin{aligned}&w_i^*(t-\tau)=w_i^*(t),\sigma(w_i^*(t-\tau))=\sigma(w_i^*(t)),\\ &f(x_i(t-\tau))=f(x_i(t)),f(x_i^*(t-\tau))=f(x_i^*(t)),\ \forall i\end{aligned}\tag{3.107}$$

then we can recast Eq.(3.106) into

$$\begin{aligned}&-\frac{1}{R_x}w_i^*+\sum_j a_{ij}\sigma(w_j^*(t))+\sum_j a_{ij}^{\tau}\sigma(w_j^*(t))\\ &+\tilde{\bigwedge_j}\alpha_{ij}f(x_j(t))-\tilde{\bigwedge_j}\alpha_{ij}f(x_j^*(t))\\ &+\tilde{\bigwedge_j}\alpha_{ij}^{\tau}f(x_j(t))-\tilde{\bigwedge_j}\alpha_{ij}^{\tau}f(x_j^*(t))\end{aligned}$$

$$+\tilde{\bigvee_j}\beta_{ij}f(x_j(t)) - \tilde{\bigvee_j}\beta_{ij}f(x_j^*(t))$$
$$+\tilde{\bigvee_j}\beta_{ij}^\tau f(x_j(t)) - \tilde{\bigvee_j}\beta_{ij}^\tau f(x_j^*(t)) = 0,$$
$$i = 1, 2, ..., MN. \tag{3.108}$$

then we have:

$$\begin{aligned} w_i^* &= R_x\{\sum_j a_{ij}\sigma(w_j^*(t)) + \sum_j a_{ij}^\tau\sigma(w_j^*(t)) \\ &+\tilde{\bigwedge_j}\alpha_{ij}f(x_j(t)) - \tilde{\bigwedge_j}\alpha_{ij}f(x_j^*(t)) \\ &+\tilde{\bigwedge_j}\alpha_{ij}^\tau f(x_j(t)) - \tilde{\bigwedge_j}\alpha_{ij}^\tau f(x_j^*(t)) \\ &+\tilde{\bigvee_j}\beta_{ij}f(x_j(t)) - \tilde{\bigvee_j}\beta_{ij}f(x_j^*(t)) \\ &+\tilde{\bigvee_j}\beta_{ij}^\tau f(x_j(t)) - \tilde{\bigvee_j}\beta_{ij}^\tau f(x_j^*(t))\}, \\ &i = 1, 2, ..., MN. \end{aligned} \tag{3.109}$$

In view of $|\sigma(x) - \sigma(y)| \leq |x - y|$, and using the similar process of the proof of Theorem 5 we complete the proof. □

Let $H = (h_{ij})_{MN\times MN}$ satisfies:

$$h_{ij} = \begin{cases} \frac{1}{R_x} - a_{ii} - |a_{ii}^\tau| - |\alpha_{ii}| - |\alpha_{ii}^\tau| - |\beta_{ii}| - |\beta_{ii}^\tau|, & i = j \\ -|a_{ij}| - |a_{ij}^\tau| - |\alpha_{ij}| - |\alpha_{ij}^\tau| - |\beta_{ij}| - |\beta_{ij}^\tau|, & i \neq j \end{cases} \tag{3.110}$$

then we have the following theorem:

Theorem 12

The origin of Eq.(3.102) is global asymptotically stable if H is a nonsingular M-matrix.

Proof: We construct the following Lyapunov function:

$$\begin{aligned} V(\mathbf{w}(t)) = &\sum_{i=1}^{MN} p_i\Big(Csgn(w_i(t))w_i(t) \\ &+\sum_{j=1}^{MN}\int_{t-\tau}^{t}(|a_{ij}^\tau| + |\alpha_{ij}^\tau| + |\beta_{ij}^\tau|)|\sigma(w_j(s))|ds\Big) \end{aligned} \tag{3.111}$$

where $p_i > 0, i = 1, 2, ..., MN$, are constants.

Along the solution of Eq.(3.102), we calculate the Dini upper-right differential of $V(\mathbf{w}(t))$ as:

$$\begin{aligned} &D^+V(\mathbf{w}(t))\Big|_{Eq.(3.102)} \\ = &\sum_{i=1}^{MN} p_i\{sgn(w_i(t))C\dot{w}_i(t) \end{aligned}$$

$$
\begin{aligned}
&+\sum_{j=1}^{MN}(|a_{ij}^{\tau}|+|\alpha_{ij}^{\tau}|+|\beta_{ij}^{\tau}|)[|\sigma(w_j(t))|-|\sigma(w_j(t-\tau))|]\}\\
=\ &\sum_{i=1}^{MN} p_i\Big\{-\frac{1}{R_x}sgn(w_i(t))w_i(t)+sgn(w_i(t))\sum_{j=1}^{MN}a_{ij}\sigma(w_j(t))\\
&+sgn(w_i(t))\sum_{j=1}^{MN}a_{ij}^{\tau}\sigma(w_j(t-\tau))\\
&+sgn(w_i(t))\left(\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}f(x_j(t))-\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}f(x_j^*(t))\right)\\
&+sgn(w_i(t))\left(\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}^{\tau}f(x_j(t-\tau))-\tilde{\bigwedge}_{j=1}^{MN}\alpha_{ij}^{\tau}f(x_j^*(t-\tau))\right)\\
&+sgn(w_i(t))\left(\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x_j(t))-\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}f(x_j^*(t))\right)\\
&+sgn(w_i(t))\left(\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}^{\tau}f(x_j(t-\tau))-\tilde{\bigvee}_{j=1}^{MN}\beta_{ij}^{\tau}f(x_j^*(t-\tau))\right)\\
&+\sum_{j=1}^{MN}(|a_{ij}^{\tau}|+|\alpha_{ij}^{\tau}|+|\beta_{ij}^{\tau}|)[|\sigma(w_j(t))|-|\sigma(w_j(t-\tau))|]\Big\} \qquad (3.112)
\end{aligned}
$$

In view of $|\sigma(w_i)| \le |w_i|$ and from the similar process of proof of Theorem 5 and in view of parameter assumptions in Eq.(3.76), we have:

$$
\begin{aligned}
&D^+V(\mathbf{w}(t))\Big|_{Eq.(3.102)}\\
\le\ &-\sum_{i=1}^{MN}\Big\{p_i(\frac{1}{R_x}-a_{ii}-|a_{ii}^{\tau}|-|\alpha_{ii}|-|\alpha_{ii}^{\tau}|-|\beta_{ii}|-|\beta_{ii}^{\tau}|)\\
&-\sum_{j\neq i}^{MN}p_j(|a_{ji}|+|a_{ji}^{\tau}|-|\alpha_{ji}|+|\alpha_{ji}^{\tau}|+|\beta_{ji}|+|\beta_{ji}^{\tau}|)\Big\}|\sigma(w_i(t))|\\
=\ &-\sum_{i=1}^{MN}\left(p_ih_{ii}+\sum_{j\neq i}^{n}p_jh_{ji}\right)|\sigma(w_i(t))|<0 \qquad (3.113)
\end{aligned}
$$

The last inequality is satisfied when $\mathbf{w} \neq 0$. □

3.5 Type-I FDCNN

An $M \times N$ type-I FDCNN is described by the following state equation:

$$
\begin{aligned}
&C\frac{dx_{ij}(t)}{dt}\\
=\ &-\frac{1}{R_x}x_{ij}(t)+\sum_{C_{kl}\in N_r(ij)}A(i,j;k,l)y_{kl}(t)\\
&+\sum_{C_{kl}\in N_r(ij)}A^{\tau}(i,j;k,l)y_{kl}(t-\tau)+\sum_{C_{kl}\in N_r(ij)}B(i,j;k,l)u_{kl}(t)+I
\end{aligned}
$$

$$
\begin{aligned}
&+ \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} \mu_{A_{fmin}(i,j;k,l)}\Big(y_{kl}(t)\Big) + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} \mu_{A_{fmax}(i,j;k,l)}\Big(y_{kl}(t)\Big) \\
&+ \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} \mu_{A^{\tau}_{fmin}(i,j;k,l)}\Big(y_{kl}(t-\tau)\Big) \\
&+ \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} \mu_{A^{\tau}_{fmax}(i,j;k,l)}\Big(y_{kl}(t-\tau)\Big) \\
&+ \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} \mu_{B_{fmin}(i,j;k,l)}\Big(u_{kl}(t)\Big) + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} \mu_{B_{fmax}(i,j;k,l)}\Big(u_{kl}(t)\Big) \\
&, 1 \le i \le M, 1 \le j \le N
\end{aligned} \tag{3.114}
$$

Similarly we have:

Proposition 6

Given the initial condition

$$\mathbf{x}_0(t) = \phi(t), \qquad \phi(t) \in C_\tau \tag{3.115}$$

then the FDCNN in Eq.(3.114) has a unique continuous solution for $t \in [0, \infty)$.

Proposition 7

If the initial condition are bounded by $K > 0$, then all states x_{ij} of a delay-type cellular neural network are bounded for all time in absolute value by the sum:

$$
\begin{aligned}
&M = K + R_x|I| \\
&+R_x \max_{i,j}\Big\{\sum_{k,l}(|A^{\tau}(i,j;k,l)| + |A(i,j;k,l)| + |B(i,j;k,l)|\Big\} \\
&+6R_x
\end{aligned} \tag{3.116}
$$

and the ω-limit points of $x_{ij}(t)$ are bounded in absolute value by $(M - K)$.

By repacking $\mathbf{x} = \{x_{ij}\}_{M\times N}$ in to an 1D vector $\mathbf{x} = \{x_i\}_{MN\times 1}$, we can rewrite Eq.(3.114) as:

$$
\begin{aligned}
&C\frac{dx_j(t)}{dt} \\
= \; &-\frac{1}{R_x}x_j(t) + \sum_{C_j\in N_r(i)} a_{ij}y_j(t) \\
&+ \sum_{C_j\in N_r(i)} a^{\tau}_{ij}y_j(t-\tau) + \sum_{C_j\in N_r(i)} b_{ij}u_j(t) + I \\
&+ \tilde{\bigwedge_{C_j\in N_r(i)}} \mu_{A_{fmin}(ij)}\Big(y_j(t)\Big) + \tilde{\bigvee_{C_j\in N_r(i)}} \mu_{A_{fmax}(ij)}\Big(y_j(t)\Big) \\
&+ \tilde{\bigwedge_{C_j\in N_r(i)}} \mu_{A^{\tau}_{fmin}(ij)}\Big(y_j(t-\tau)\Big) \\
&+ \tilde{\bigvee_{C_j\in N_r(i)}} \mu_{A^{\tau}_{fmax}(ij)}\Big(y_j(t-\tau)\Big) \\
&+ \tilde{\bigwedge_{C_j\in N_r(i)}} \mu_{B_{fmin}(ij)}\Big(u_j(t)\Big) + \tilde{\bigvee_{C_j\in N_r(i)}} \mu_{B_{fmax}(ij)}\Big(u_j(t)\Big) \\
&, 1 \le i \le M, 1 \le j \le N
\end{aligned} \tag{3.117}
$$

Suppose $\mathbf{x}^*$ is an equilibrium point and let $\mathbf{w} = \{w_i\}_{MN\times 1} \triangleq \mathbf{x} - \mathbf{x}^*$, then Eq.(3.117) can be rewritten into:

$$\begin{aligned} C\dot{w}_i &= -\frac{1}{R_x}w_i + \sum_j a_{ij}\sigma(w_j(t)) + \sum_j a_{ij}^\tau \sigma(w_j(t-\tau)) \\ &+\tilde{\bigwedge_j}\mu_{A_{fmin}(ij)}\big(y_j(t)\big) - \tilde{\bigwedge_j}\mu_{A_{fmin}(ij)}\big(y_j^*(t)\big) \\ &+\tilde{\bigwedge_j}\mu_{A_{fmin}^\tau(ij)}\big(y_j(t-\tau)\big) - \tilde{\bigwedge_j}\mu_{A_{fmin}^\tau(ij)}\big(y_j^*(t-\tau)\big) \\ &+\tilde{\bigvee_j}\mu_{A_{fmax}(ij)}\big(y_j(t)\big) - \tilde{\bigvee_j}\mu_{A_{fmax}(ij)}\big(y_j^*(t)\big) \\ &+\tilde{\bigvee_j}\mu_{A_{fmax}^\tau(ij)}\big(y_j(t-\tau)\big) - \tilde{\bigvee_j}\mu_{A_{fmax}^\tau(ij)}\big(y_j^*(t-\tau)\big) \\ &i = 1, 2, ..., MN. \end{aligned} \tag{3.118}$$

Similar to that in Proposition 2, we have the following proposition:

Proposition 8

Assume that for two points $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ there exists a k such that:

$$\tilde{\bigwedge}_{j=1}^{MN}\mu_{A_{fmin}^\tau(ij)}(f(x_i^{(2)}(t-\tau))) = \mu_{A_{fmin}^\tau(ik)}(f(x_k^{(2)}(t-\tau))) \tag{3.119}$$

then

$$\begin{aligned} &|\tilde{\bigwedge}_{j=1}^{MN}\mu_{A_{fmin}^\tau(ij)}(f(x_i^{(1)}(t-\tau))) - \tilde{\bigwedge}_{j=1}^{MN}\mu_{A_{fmin}^\tau(ij)}(f(x_i^{(2)}(t-\tau)))| \\ \leq\ &|\mu_{A_{fmin}^\tau(ik)}(f(x_k^{(1)}(t-\tau))) - \mu_{A_{fmin}^\tau(ik)}(f(x_k^{(2)}(t-\tau)))| \end{aligned} \tag{3.120}$$

Similarly, let there exists an l such that:

$$\tilde{\bigvee}_{j=1}^{MN}\mu_{A_{fmax}^\tau(ij)}(f(x_i^{(1)}(t-\tau))) = \mu_{A_{fmax}^\tau(ik)}(f(x_k^{(1)}(t-\tau))) \tag{3.121}$$

then we have:

$$\begin{aligned} &|\tilde{\bigvee}_{j=1}^{MN}\mu_{A_{fmax}^\tau(ij)}(f(x_i^{(1)}(t-\tau))) - \tilde{\bigvee}_{j=1}^{MN}\mu_{A_{fmin}(ij)}^\tau(f(x_i^{(2)}(t-\tau)))| \\ \leq\ &|\mu_{A_{fmax}(il)}^\tau(f(x_l^{(1)}(t-\tau))) - \mu_{A_{fmax}(il)}^\tau(f(x_l^{(2)}(t-\tau)))| \end{aligned} \tag{3.122}$$

Assume that all the $\mu_{A_{fmin}(ij)}(\cdot)$, $\mu_{A_{fmax}(ij)}(\cdot)$, $\mu_{A_{fmin}^\tau(ij)}(\cdot)$ and $\mu_{A_{fmax}^\tau(ij)}^\tau(\cdot)$ are globally Lipschitzian, i.e., for any y_j and y_j' we have:

$$|\mu_{A_{fmin}(ij)}(y_j) - \mu_{A_{fmin}(ij)}(y_j')| \leq \alpha_{ij}^M \tag{3.123}$$

$$|\mu_{A_{fmax}(ij)}(y_j) - \mu_{A_{fmax}(ij)}(y_j')| \leq \beta_{ij}^M \tag{3.124}$$

$$|\mu_{A_{fmin}^\tau(ij)}(y_j) - \mu_{A_{fmin}^\tau(ij)}(y_j')| \leq \alpha_{ij}^{\tau M} \tag{3.125}$$

$$|\mu_{A^\tau_{fmax}(ij)}(y_j) - \mu_{A^\tau_{fmax}(ij)}(y'_j)| \leq \beta^{\tau M}_{ij} \tag{3.126}$$

And we assume that

$$\begin{aligned} a_{ij} = a_{ji}, \alpha^M_{ij} = \alpha^M_{ji}, \beta^M_{ij} = \beta^M_{ji} \\ \alpha^{\tau M}_{ij} = \alpha^{\tau M}_{ji}, \beta^{\tau M}_{ij} = \beta^{\tau M}_{ji} \end{aligned} \tag{3.127}$$

Let

$$|D^\tau| = \left(|a_{ij} + a^\tau_{ij}| + |\alpha^M_{ij}| + |\beta^M_{ij}| + |\alpha^{\tau M}_{ij}| + |\beta^{\tau M}_{ij}|\right)_{MN \times MN} \tag{3.128}$$

then we have the following theorem.

Theorem 13

Suppose that the spectral radius of matrix $R_x|D^\tau|$, $\rho\left(R_x|D^\tau|\right) < 1$, then the FCNN in Eq.(3.118) has only one equilibrium point.

Proof: It the same as those in Theorem 7 and Theorem 11. □

Let $H^M = (h_{ij})_{MN \times MN}$ satisfies:

$$h^M_{ij} = \begin{cases} \frac{1}{R_x} - a_{ii} - |a^\tau_{ii}| - |\alpha^M_{ii}| - |\alpha^{\tau M}_{ii}| - |\beta^M_{ii}| - |\beta^{\tau M}_{ii}|, & i = j \\ -|a_{ij}| - |a^\tau_{ij}| - |\alpha^M_{ij}| - |\alpha^{\tau M}_{ij}| - |\beta^M_{ij}| - |\beta^{\tau M}_{ij}|, & i \neq j \end{cases} \tag{3.129}$$

then we have the following theorem:

Theorem 14

The origin of Eq.(3.118) is global asymptotically stable if H^M is a nonsingular M-matrix.

Proof: Similar to that of Theorem 12. □

3.6 Stability of Discrete-Time FCNN

Fuzzy discrete-time cellular neural networks(FDTCNN) is a very important branch of fuzzy cellular neural networks(FCNN). FDTCNN is governed by a set of difference equations. We presented the structures of FDTCNN and provide some stability criteria for them.

The dynamics of a cell C_{ij} in an $M \times N$ FDTCNN is given by

State equation

$$\begin{aligned} x_{ij}(t+1) = & \bigvee_{C_{kl} \in N_r(ij)} A_1(i,j;k,l) y_{kl}(t) + \bigwedge_{C_{kl} \in N_r(ij)} A_2(i,j;k,l) y_{kl}(t) \\ & + \bigvee_{C_{kl} \in N_r(ij)} B_1(i,j;k,l) u_{kl} \\ & + \bigwedge_{C_{kl} \in N_r(ij)} B_2(i,j;k,l) u_{kl} + I \end{aligned} \tag{3.130}$$

where $t \in \mathbf{N}$ is the discrete-time. u_{ij}, x_{ij} and y_{ij} are input, state and output, respectively. $\bigvee$ and $\bigwedge$ denote fuzzy OR and fuzzy AND, respectively. In this paper, we let $\bigvee = \max$ and $\bigwedge = \min$. $A_1(i,j;k,l)$ and $A_2(i,j;k,l)$ denote the *fuzzy Max feedback synaptic weight* and *fuzzy Min feedback synaptic weight*, respectively. $B_1(i,j;k,l)$ and $B_2(i,j;k,l)$ denote

the *fuzzy Max feed-forward synaptic weight* and *fuzzy Min feed-forward synaptic weight*, respectively.

Output equation

$$y_{ij}(t) = f(x_{ij}(t)) \tag{3.131}$$

Parameter Assumptions

$$A_1(i,j;k,l) = A_1(k,l;i,j), A_2(i,j;k,l) = A_2(k,l;i,j), \tag{3.132}$$

Let x_{ij}^* be an equilibrium point of the FDTCNN and let $e_{ij}(t) = x_{ij}(t) - x_{ij}^*$, we can recast Eq.(3.130) into

$$\begin{aligned} e_{ij}(t+1) &= [\bigvee_{C_{kl}\in N_r(ij)} A_1(i,j;k,l)y_{kl}(t) - \bigvee_{C_{kl}\in N_r(ij)} A_1(i,j;k,l)y_{kl}^*] \\ &+[\bigwedge_{C_{kl}\in N_r(ij)} A_2(i,j;k,l)y_{kl}(t) \\ &- \bigwedge_{C_{kl}\in N_r(ij)} A_2(i,j;k,l)y_{kl}^*] \end{aligned} \tag{3.133}$$

Let

$$\sigma(e_{kl}(t)) \triangleq f(x_{kl}(t)) - f(x_{kl}^*) \tag{3.134}$$

Similar to the Corollary 1 of [350], we need the following proposition.

Proposition 9

Let $\{x_{ij1}\}$ and $\{x_{ij2}\}$ be two states of FDTCNN in Eq.(3.130), then we have

$$\begin{aligned} &\bigvee_{C_{kl}\in N_r(ij)} A_1(i,j;k,l)y_{kl1} - \bigvee_{C_{kl}\in N_r(ij)} A_1(i,j;k,l)y_{kl2} \\ &\leq \sum_{C_{kl}\in N_r(ij)} |A_1(i,j;k,l)||y_{kl1} - y_{kl2}| \end{aligned} \tag{3.135}$$

and

$$\begin{aligned} &\bigwedge_{C_{kl}\in N_r(ij)} A_2(i,j;k,l)y_{kl1} - \bigwedge_{C_{kl}\in N_r(ij)} A_2(i,j;k,l)y_{kl2} \\ &\leq \sum_{C_{kl}\in N_r(ij)} |A_2(i,j;k,l)||y_{kl1} - y_{kl2}| \end{aligned} \tag{3.136}$$

Proof:

Similar to that of Corollary 1 of [350]. □

Theorem 15

The equilibrium point of the FDTCNN in Eq.(3.130), $\{x_{ij}^*\}$, is asymptotically stable if

$$L^2(\sum_{C_{kl}\in N_r(ij)} |A_1(i,j;k,l)| + |A_2(i,j;k,l)|)^2 < 1 \tag{3.137}$$

and

$$|\sigma(e_{ij}(t))| \leq L|e_{ij}(t)| \tag{3.138}$$

where $L > 0$ is a constant.

Proof:

We define a Lyapunov function as

$$V = \sum_{i=1}^{M}\sum_{j=1}^{N} e_{ij}^2(t) \tag{3.139}$$

Taking the forward difference of V along the solution of Eq.(3.133) we get

$$\begin{aligned}\Delta V &= \sum_{i=1}^{M}\sum_{j=1}^{N}[(\bigvee_{C_{kl}\in N_r(ij)} A_1(i,j;k,l)y_{kl}(t) - \bigvee_{C_{kl}\in N_r(ij)} A_1(i,j;k,l)y_{kl}^*) \\ &\quad +(\bigwedge_{C_{kl}\in N_r(ij)} A_2(i,j;k,l)y_{kl}(t) - \bigwedge_{C_{kl}\in N_r(ij)} A_2(i,j;k,l)y_{kl}^*)]^2 \\ &\quad -e_{ij}^2(t)\end{aligned} \tag{3.140}$$

In view of Proposition 9, we have

$$\begin{aligned}\Delta V &\le \sum_{i=1}^{M}\sum_{j=1}^{N}[\sum_{C_{kl}\in N_r(ij)} |A_1(i,j;k,l)||\sigma(e_{kl}(t))| \\ &\quad + \sum_{C_{kl}\in N_r(ij)} |A_2(i,j;k,l)||\sigma(e_{kl}(t))|]^2 - e_{ij}^2(t) \\ &\le \sum_{i=1}^{M}\sum_{j=1}^{N}(\sum_{C_{kl}\in N_r(ij)} |A_1(i,j;k,l)| + |A_2(i,j;k,l)|)^2\sigma^2(e_{kl}(t)) - e_{ij}^2(t) \\ &\le \sum_{i=1}^{M}\sum_{j=1}^{N}(\sum_{C_{kl}\in N_r(ij)} |A_1(i,j;k,l)| + |A_2(i,j;k,l)|)^2L^2e_{kl}^2(t) - e_{ij}^2(t) \\ &= \sum_{i=1}^{M}\sum_{j=1}^{N}(\sum_{C_{ij}\in N_r(kl)} |A_1(k,l;i,j)| + |A_2(k,l;i,j)|)^2L^2e_{ij}^2(t) - e_{ij}^2(t) \\ &= \sum_{i=1}^{M}\sum_{j=1}^{N} e_{ij}^2(t)[(\sum_{C_{ij}\in N_r(kl)} |A_1(k,l;i,j)| + |A_2(k,l;i,j)|)^2L^2 - 1]\end{aligned} \tag{3.141}$$

We can see that if

$$L^2(\sum_{C_{ij}\in N_r(kl)} |A_1(k,l;i,j)| + |A_2(k,l;i,j)|)^2 < 1 \tag{3.142}$$

then ΔV is negative, which implies the asymptotic stability of the equilibrium point $\{x_{ij}^*\}$.

Since the symmetric property of the neighborhood system and the parameter assumptions, we can recast Eq.(3.142) into

$$L^2(\sum_{C_{kl}\in N_r(ij)} |A_1(i,j;k,l)| + |A_2(i,j;k,l)|)^2 < 1 \tag{3.143}$$

□

We give a structure of FDTCNN which is governed by a set of nonlinear difference equations. Based on a discrete Lyapunov function, we present a stability criterion for FDTCNN. FDTCNN can also be used in some typical applications of FCNN as we have presented before[351, 352]. Furthermore, FDTCNN can also be used to model some discrete-time and spatial-distributed phenomena such as highway traffic flow.

Chapter 4

FCNN as Computational Arrays

In this chapter, we present the FCNN structures which function as computational arrays. Since the most used interpretation of fuzzy AND and fuzzy OR are minimum and maximum calculations and since mathematical morphology[292, 293, 129, 117] is closely connected with fuzzy logic because it is essentially concerned with *min* and *max* operations[103], FCNN is a paradigm for implementing morphological image operators. The applications of FCNN as a computational min-max network are also presented in this chapter. So, we show that FCNN functions as a low-level computational structure just as the conventional CNN does. The advantage of the applications of FCNN to the image processing problems presented in this chapter is that type-II FCNN can implement *max* and *min* operations in a very natural and efficient way than the conventional CNN does. To show this advantage of FCNN over conventional CNN, the comparison between FCNN-based and conventional CNN-based mathematical morphological operations is also presented. This chapter is the collection of the results in our papers: [351, 352, 354, 353, 355, 360, 375, 372, 368, 365, 379, 377].

4.1 Basic knowledge of mathematical morphology

Mathematical morphology[292, 293, 129] is a theory which is concern with processing and analysis of image, using operators and functionals based on topological and geometrical concepts. During the last decade, it has become a cornerstone of image processing problems. Morphological operations have been widely used for object recognition[302, 301], edge detection[180], shape analysis[239], thinning[148], image coding[102, 188], and smoothing[147].

Four basic transformations in mathematical morphology are: dilation, erosion, opening and closing. These basic transformations permit to extract contours, skeletons, separate close objects, compute geodesic distances, etc.[292, 293, 129]. The basic idea of mathematical morphology is to probe an image with a *structuring element* and to quantify the manner in which the structuring element fits(or does not fit) within the image. In general, the structuring element has a simple shape and is very small compared to the image being investigated.

Let $f : X \mapsto E$ and $s : S \mapsto E$ be maps for image and structuring element, respectively, where E is the range of gray values. X is a gray-scale image, S is a weighted structuring element. Then the basic morphological operations of erosion and dilation for gray-scale images are given by[117]:

Gray-scale Erosion:

$$X \ominus S = \min\{f(x+z) - s(z)\} \tag{4.1}$$

for all $z \in S$ and $x + z \in X$.

Gray-scale Dilation:

$$X \oplus S = \max\{f(x-z) + s(z)\} \tag{4.2}$$

for all $z \in S$ and $x - z \in X$.

With the definition of gray-scale erosion and gray-scale dilation, gray-scale opening and gray-scale closing are given by:

Gray-scale Opening:

$$X \circ S = (X \ominus S) \oplus S \tag{4.3}$$

Gray-scale Closing:

$$X \bullet S = (X \oplus S) \ominus S \tag{4.4}$$

For the purpose of implementation of gray-scale morphological operations by FCNN, E is normalized within $[0, 1]$.

For example, let S be within a 3×3 square with the origin located at its center as follows:

$$S = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \tag{4.5}$$

where $h_1 \sim h_9$ denote gray values of corresponding entries of structuring element. An entry in a structuring element is *defined* if there exists an operation, is *undefined* if there exists no operation(or simply set an undefined entry as $+\infty$ in erosion and as $-\infty$ in dilation). A structuring element whose defined entries have the same value is called a *flat structuring element.*

4.2 Implementation of morphological operations

Since type-II FCNN is a combination of *min* and *max* operations with parallel dynamics of CNN, it is very convenient to implement morphological operations in its structure. Another reason for using FCNN in morphology is that given a relatively small and simple shaped structuring element the morphological operations have strong local property. And a big size structuring elements can be decomposed into a set of smaller size structuring elements. This makes the possible applications of 3×3- or 5×5-neighborhood FCNN's to image processing problem where large structuring element is needed. FCNN is found to be a universal parallel array to implement morphological operations for processing both binary and gray-scale images[351, 352]. In this section, we use different FCNN structures to implement the basic morphological operations. Although the results presented in this section is based on FCNN, one can very easy to find the corresponding FDTCNN's.

4.2.1 Using multiplicative FCNN

The following multiplicative FCNN[1][350] is used to implement a morphological operator with a flat structuring element:

$$\begin{aligned} C\frac{dx_{ij}}{dt} &= -\frac{1}{R_x}x_{ij} + I + \tilde{\bigwedge}_{C_{kl}\in N_r(i,j)} B_{fmin}(i,j;k,l)u_{kl} \\ &\quad + \tilde{\bigvee}_{C_{kl}\in N_r(i,j)} B_{fmax}(i,j;k,l)u_{kl} \\ &\quad , 1 \le i \le M, 1 \le j \le N \end{aligned} \tag{4.6}$$

The parameters for implementation of erosion with a flat structuring element are given by:

$$I = -h, R_x = 1, B_{fmax} = 0, B_{fmin} = \frac{1}{h}S \tag{4.7}$$

where h is the height of the flat structuring element S.

The parameters for implementation of dilation with a flat structuring element are given by:

$$I = h, R_x = 1, B_{fmax} = \frac{1}{h}S_D, B_{fmin} = 0 \tag{4.8}$$

where $S_D = \{-x : x \in S\}$

For example, let S as that in eq.(4.5), then S_D is given by:

$$S_D = \begin{bmatrix} h_9 & h_8 & h7 \\ h_4 & h_5 & h_6 \\ h_3 & h_2 & h_1 \end{bmatrix} = B_{fmax} \tag{4.9}$$

4.2.2 Using additive FCNN

The following additive FCNN is used to implement erosion and dilation with any structuring element besides flat ones[351, 352, 355]:

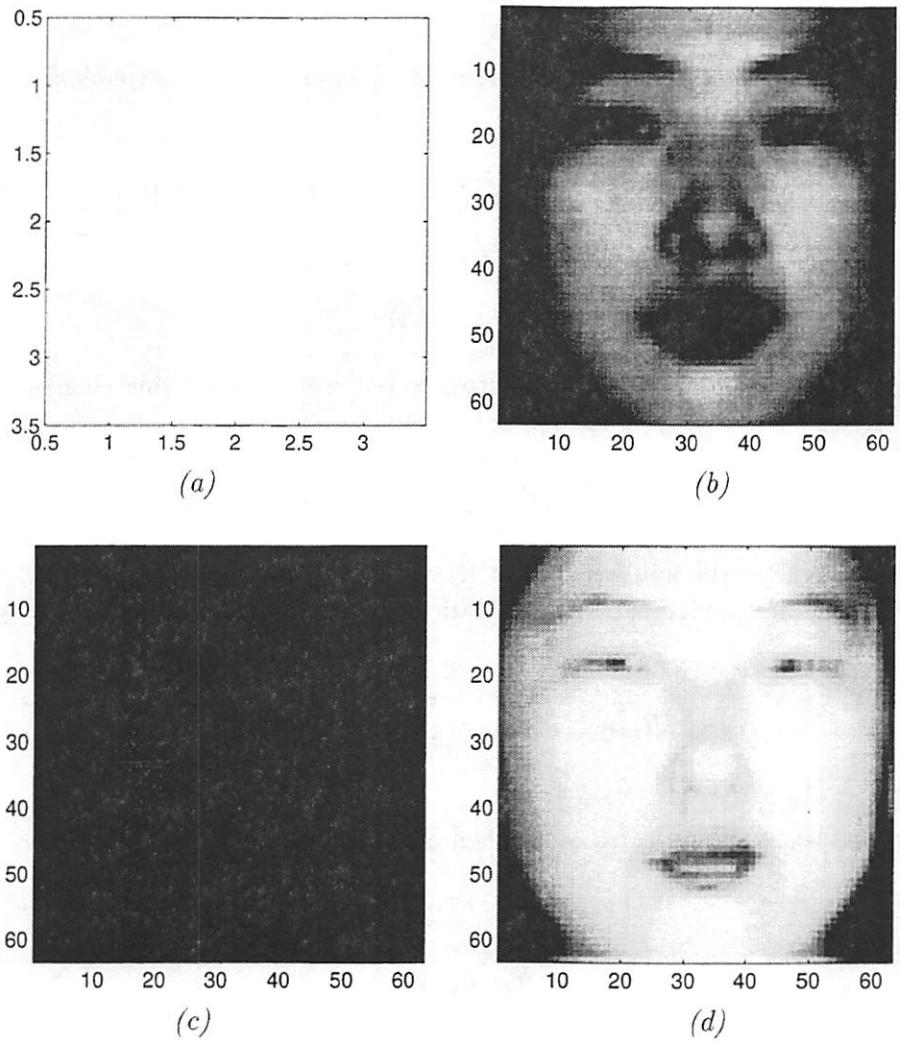
$$\begin{aligned} C\frac{dx_{ij}}{dt} &= -\frac{1}{R_x}x_{ij} + \tilde{\bigwedge}_{C_{kl}\in N_r(i,j)} (B_{fmin}(i,j;k,l) + u_{kl}) \\ &\quad + \tilde{\bigvee}_{C_{kl}\in N_r(i,j)} (B_{fmax}(i,j;k,l) + u_{kl}) \\ &\quad , 1 \le i \le M, 1 \le j \le N \end{aligned} \tag{4.10}$$

The additive FCNN for implementing erosion has the following parameters:

$$R_x = 1, B_{fmax} = \text{undefined}, B_{fmin} = -S \tag{4.11}$$

We call the above FCNN as an *erosion FCNN*. Image X is its input and its initial state is arbitrary. When we say a template is "undefined", it means that the template is not used by the CNN.

[1] An FCNN is called *multiplicative* if it has multiplicative fuzzy synaptic laws. An FCNN is called *additive* if it has additive fuzzy synaptic laws.



The additive FCNN for implementing dilation is given by

$$R_x = 1, B_{fmin} = \text{undefined}, B_{fmax} = S_D \tag{4.12}$$

We call the above FCNN as a *dilation FCNN*. Image X is its input and its initial state is arbitrary.

Then opening can be implemented by using an erosion FCNN followed by a dilation FCNN. Closing can be implemented by using a dilation FCNN followed by an erosion FCNN.

Fig. 4.1 shows examples of implementation of basic morphological operations to a gray-scale image using FCNN. The structuring element is given by

$$S = \begin{bmatrix} \frac{2}{25} & \frac{1}{25} & \frac{2}{25} \\ \frac{1}{25} & 0 & \frac{1}{25} \\ \frac{2}{25} & \frac{1}{25} & \frac{2}{25} \end{bmatrix} \tag{4.13}$$

as shown in Fig. 4.1(a). Fig. 4.1(b) shows a gray image X of size 63×63 and 256 gray levels. Fig. 4.1(c) shows the output of the dilation FCNN. Input is X. Fig. 4.1(d) shows the output of the erosion FCNN. Input is X. Fig. 4.1(e) shows the output of dilation FCNN

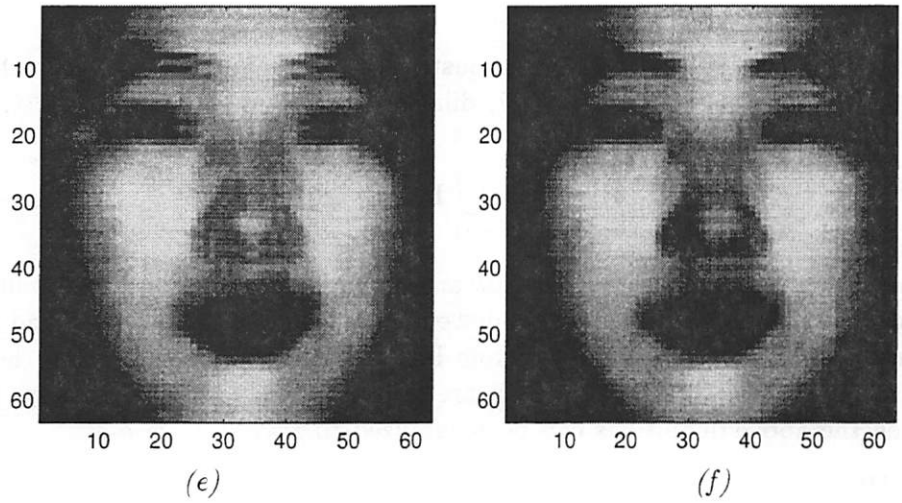


Figure 4.1: Implementation of gray scale morphological operations to gray scale images using additive FCNN. (a) A non-flat gray-scale structuring element. (b) Input image of a Chinese girl. (c) Output of dilation FCNN. (d) Output of erosion FCNN. (e) Output of opening. (f) Output of closing.

when the image in Fig. 4.1(d) is input. So, it's the opening of X. Fig. 4.1(f) shows the output of erosion FCNN when the image in Fig. 4.1(c) is input. So, it's the closing of X.

4.3 Applications to image processing

4.3.1 Grey-scale reconstruction

We first give the concept of morphological reconstruction [337, 293, 292] briefly. Let an image I be a map from a finite rectangular subset D_I of the discrete plane $\mathbf{Z}^2$ into a discrete set $E = \{0, 1, ..., N-1\}$ of gray levels. A binary image I can only take values 0 or 1 and is often regarded as the set of its pixels with value 1. Given a set X and two pixels $p, q \in X$, the *geodesic distance* between p and q, $d_X(p, q)$, is the length of the shortest paths joining p and q which are included in X.

Definition 8

Binary Geodesic Dilation: Let $X \subset \mathbf{Z}^2$ be a discrete set and $Y \subseteq X$. The binary geodesic dilation of size $n \geq 0$ of Y within X is the set of the pixels of X whose geodesic distance to Y is smaller or equal to n:

$$\tilde{\mathbf{D}}_X^{(n)}(Y) = \{p \in X | d_X(p, Y) \leq n\} \tag{4.14}$$

Geodesic dilation of size n can be obtained by iterating n elementary geodesic dilation, $\tilde{\mathbf{D}}_X^{(1)}(Y)$:

$$\tilde{\mathbf{D}}_X^{(n)}(Y) = \underbrace{\tilde{\mathbf{D}}_X^{(1)} \circ \tilde{\mathbf{D}}_X^{(1)} \circ ... \circ \tilde{\mathbf{D}}_X^{(1)}}_{n}(Y) \tag{4.15}$$

The elementary geodesic dilation can be obtained via a standard dilation of size one followed by an intersection

$$\tilde{\mathbf{D}}_X^{(1)}(Y) = (Y \oplus S) \cap X \tag{4.16}$$

Definition 9

Reconstruction for Binary Images: The reconstruction of binary image X from $Y \subseteq X$ is obtained by iterating elementary geodesic dilation of Y inside X until stability, denoted by

$$\tilde{\mathbf{R}}_X(Y) = \bigcup_{n \geq 1} \tilde{\mathbf{D}}_X^{(n)}(Y) \tag{4.17}$$

In Definition 9, the set X is called *mask* and Y is called *marker*. Fig. 4.2 illustrated this definition. In Fig. 4.2(a), the light shadowed regions denote the mask X, and the dark shadowed regions denote the marker Y. From Fig. 4.2(b) one can see that all the regions in X which are marked by the marker set Y are recovered while the other regions in X are deleted. Then the above definitions can be generalized to gray-scale cases.

Definition 10

Gray-scale Geodesic Dilation: The elementary gray-scale geodesic dilation of gray-scale image $J \leq I$ "under" I is defined by

$$\mathbf{D}_I^{(1)}(J) = (J \oplus S) \wedge I \tag{4.18}$$

where "$\wedge$" stands for the point-wise minimum and $J \oplus S$ is the dilation of J by flat structuring element S.

Gray-scale geodesic dilation of size $n \geq 0$ is then defined by

$$\mathbf{D}_I^{(n)}(J) = \underbrace{\mathbf{D}_I^{(1)} \circ \mathbf{D}_I^{(1)} \circ \ldots \circ \mathbf{D}_I^{(1)}}_{n}(J) \tag{4.19}$$

Definition 11

Reconstruction for Gray-scale Images: The gray-scale reconstruction $\mathbf{R}_I(J)$ of gray image I from $J \leq I$ is obtained by iterating gray-scale geodesic dilation of J "under" I until stability is reached, i.e.,

$$\mathbf{R}_I(J) = \bigvee_{n \geq 1} \mathbf{D}_I^{(n)}(J) \tag{4.20}$$

Fig. 4.3 illustrates this definition. In Fig. 4.3(a), the marker J is denoted by shadowed region and the mask is outlined by a curve which has two peaks. Note that J is totally under I and only the left peak of I is marked by J. Fig. 4.3(b) shows the result of gray-scale reconstruction. Just like binary reconstruction extracts those connected components of the mask which are marked, gray-scale reconstruction extracts the peaks of the mask which are marked by the marker image.

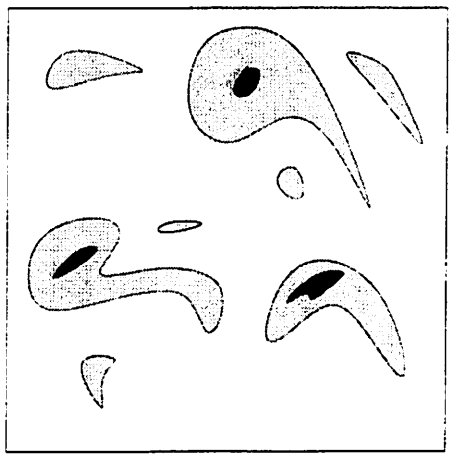
Definition 12

Gray-scale Geodesic Erosion: The elementary geodesic erosion $\mathbf{E}_I^{(1)}(J)$ of gray-scale image $J \geq I$ "above" I is given by

$$\mathbf{E}_I^{(1)}(J) = (J \ominus S) \vee I \tag{4.21}$$

where "$\vee$" stands for point-wise maximum and $J \ominus S$ is the erosion of J by flat structuring element S. The gray-scale geodesic erosion of size $n \geq 0$ is then given by

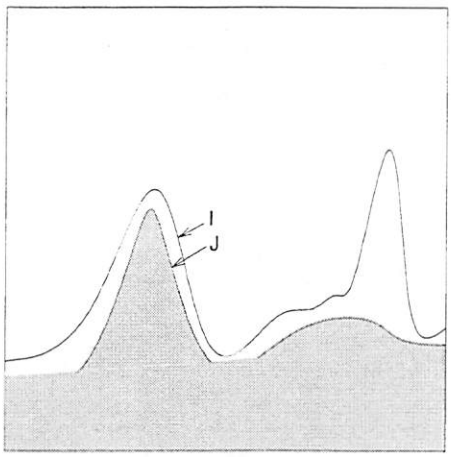
$$\mathbf{E}_I^{(n)}(J) = \underbrace{\mathbf{E}_I^{(1)} \circ \mathbf{E}_I^{(1)} \circ \ldots \circ \mathbf{E}_I^{(1)}}_{n}(J) \tag{4.22}$$



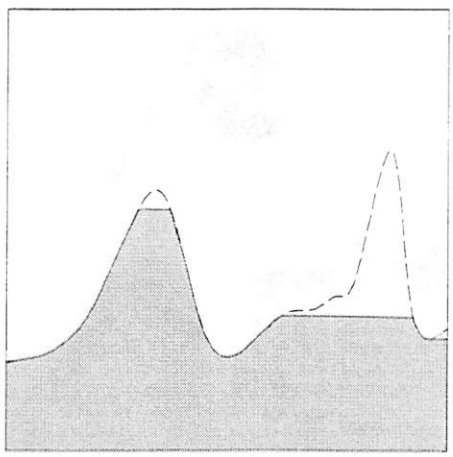
(a)

(b)

Figure 4.2: Binary reconstruction of mask X from marker Y. (a) Mask X (light shadowed regions) and marker Y(dark shadowed regions). (b) Result of reconstruction.



(a)



(b)

Figure 4.3: Gray-scale reconstruction of mask I from marker J. (a) Mask I and marker J (shadowed region). (b) result of reconstruction (shadowed region).

Definition 13

Dual Reconstruction for Gray-scale Images: The dual gray-scale reconstruction $\mathbf{R}_I^*(J)$ of mask I from marker J is obtained by iterating gray-scale geodesic erosion of J "above" I until stability is reached:

$$\mathbf{R}_I^*(J) = \bigwedge_{n\geq 1} \mathbf{E}_I^{(n)}(J) \tag{4.23}$$

Then we use the flow-chart of CNN series as shown in Fig. 4.4 to implement gray-scale reconstruction. In Fig. 4.4, the FCNN with two input layers implements *min* operation between its two inputs, $\{u_{ij1}\}$ and $\{u_{ij2}\}$. Its output is image #3. This FCNN is given by:

$$\dot{x}_{ij} = -x_{ij} + \min(u_{ij1}, u_{ij2}) \tag{4.24}$$

The minus CNN is given by:

$$\dot{x}_{ij} = -x_{ij} + (u_{ij3} - u_{ij4}) \tag{4.25}$$

Image #4 is initialized using marker J.

Similarly, we can also implement the dual gray-scale reconstruction by replacing dilation FCNN with erosion FCNN in Fig. 4.4, and give the FCNN with two input layers in Fig. 4.4 the following state equation:

$$\dot{x}_{ij} = -x_{ij} + \max(u_{ij1}, u_{ij2}) \tag{4.26}$$

, which implements *max* operation in this case.

To illustrate the usefulness of FCNN based gray-scale reconstruction algorithm, we used this algorithm to remove black dot noises as shown in Fig. 4.5. Fig. 4.5(a) shows the image in Fig. 4.1(b) and some black dot noises. The noises can be removed by applying opening operation as shown in Fig. 4.5(b). The structuring element is given by:

$$S = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4.27}$$

It is a flat structuring element with height 0. Compare the image in Fig. 4.5(b) with the one in Fig. 4.1(b), one can see that some details of the original image is also removed. To overcome this problem, we use the image in Fig. 4.5(a) as mask I and that in Fig. 4.5(b) as marker J and apply the gray-scale reconstruction algorithm. The output is shown in Fig. 4.5(c). One can see that some details are recovered.

Another application of FCNN based gray-scale reconstruction algorithm is scratch removal. Although the traditional CNN can be used to remove scratches in an image[335], the locations of the scratches should be known in advance. FCNN can remove scratches in an image without knowing their locations. Fig. 4.6(a) shows the scratched version of the image in Fig. 4.1(b). To remove scratches, we can use FCNN based opening operation. The output is shown in Fig. 4.6(b). Then the FCNN based gray-scale reconstruction algorithm is used to recover some details removed by opening operation. The mask I is the image shown in Fig. 4.6(a) and the marker J is that in Fig. 4.6(b). Fig. 4.6(c) shows the final result. One can see that some details are recovered. The structuring element is as that in Eq.(4.27).

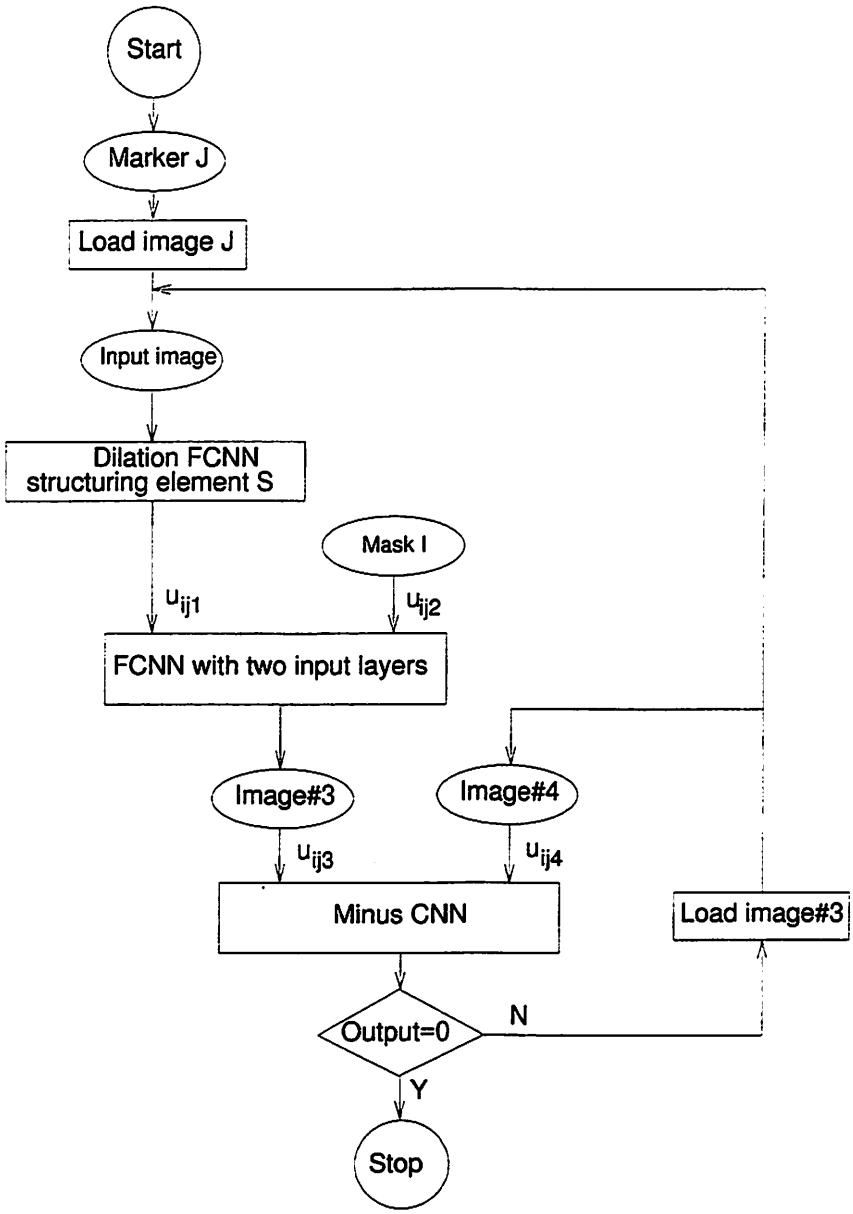


Figure 4.4: The flow-chart of CNN series for implementation of gray-scale reconstruction using FCNN series.

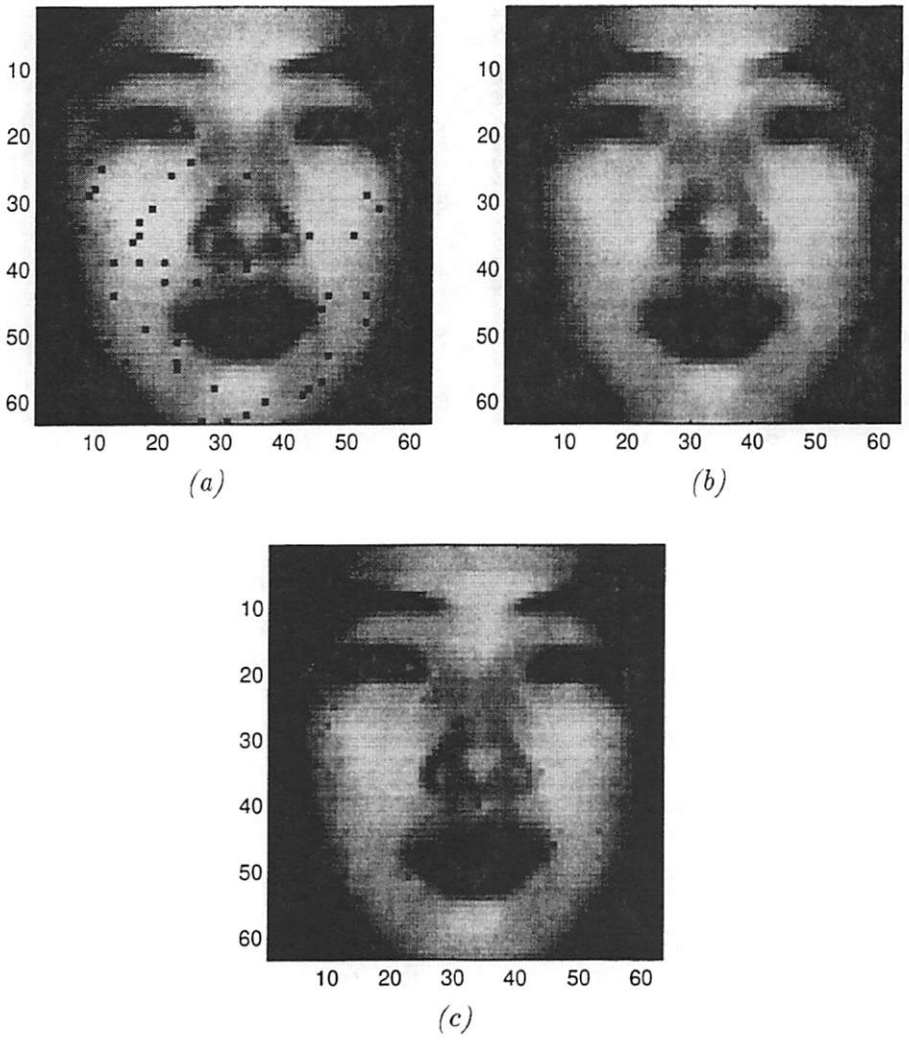


Figure 4.5: Remove black dot noises using FCNN based gray-scale reconstruction algorithm. (a) Image with black dot noises. (b) Output of FCNN based opening operation. (c) Output of FCNN based gray-scale reconstruction algorithm.

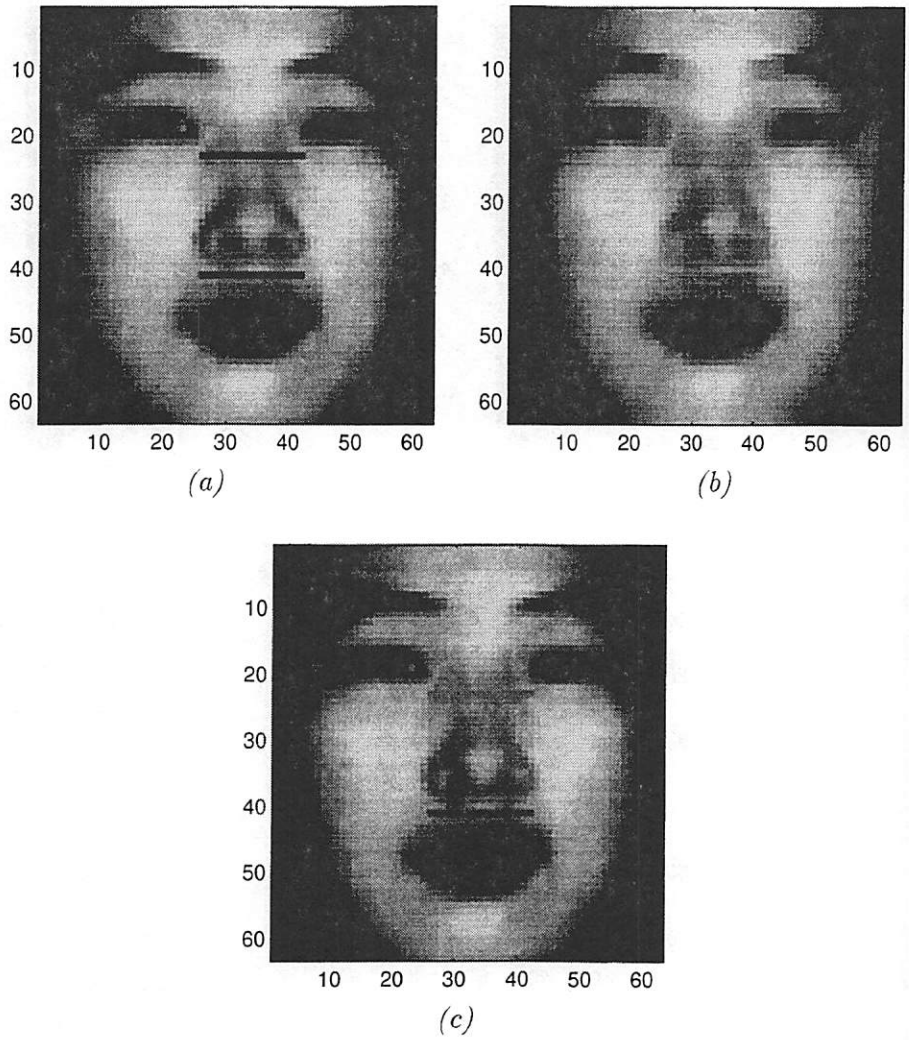


Figure 4.6: Remove scratches using FCNN based gray-scale reconstruction algorithm. (a) Scratched image. (b) Output of FCNN based opening operation. (c) Output of FCNN based gray-scale reconstruction algorithm.

The number of templates used by the FCNN based gray-scale reconstruction algorithm is $3k$, where k is the number of loops till the "Minus CNN" outputs 0's. k is very sensitive to the structure of the mask image, in the worst case, k may be very close to an half of the size of the mask image. k is not sensitive to the size and the shape of the structuring element. In the above two simulations, $k = 11$.

4.3.2 Euclidean distance transformation

The distance transformation based on Euclidean distance is not sensitive to the orientation of the object. The mathematical morphology distance transformation uses a predefined structuring element. By defining the center of the structuring element as the origin point O, we can denote all the points by their distances from O (thereby O is denoted by 0). The weights(entries) of an Euclidean distance structuring element are:

$$S_E = \lambda \begin{bmatrix} . & . & . & . & 4.0 & . & . & . & . \\ . & . & 3.6 & 3.2 & 3.0 & 3.2 & 3.6 & . & . \\ . & 3.6 & 2.8 & 2.2 & 2.0 & 2.2 & 2.8 & 3.6 & . \\ . & 3.2 & 2.2 & 1.4 & 1.0 & 1.4 & 2.2 & 3.2 & . \\ 4.0 & 3.0 & 2.0 & 1.0 & 0.0 & 1.0 & 2.0 & 3.0 & 4.0 \\ . & 3.2 & 2.2 & 1.4 & 1.0 & 1.4 & 2.2 & 3.2 & . \\ . & 3.6 & 2.8 & 2.2 & 2.0 & 2.2 & 2.8 & 3.6 & . \\ . & . & 3.6 & 3.2 & 3.0 & 3.2 & 3.6 & . & . \\ . & . & . & . & 4.0 & . & . & . & . \end{bmatrix} \tag{4.28}$$

where $\lambda > 0$ is used to normalize the value of the biggest entry into the interval [0,1]. Only those distances of 4λ or smaller are shown.

Suppose a binary image X consists of two classes: object pixels, which have values 1's, and background pixels, which have values 0's. Then the distance transformation is implemented by the following gray-scale erosion:

$$X \ominus (-S_E) \tag{4.29}$$

where the minus "-" before S_E is used to give a positive distance measure in the output.

The size of the distance structuring element must be at least as large as the largest object in the image, otherwise the central pixels of the object will remain 1's. However, the big neighborhood operation is very difficult to be implemented using VLSI techniques in FCNN. It is necessary to decompose a big structuring element into smaller ones.

From the properties of gray-scale mathematical morphology operators, when an image X is eroded by a large size structuring element S' which can be decomposed into dilation of several small structuring components S'_i as follows:

$$S' = S'_1 \oplus S'_2 \oplus ... \oplus S'_n \tag{4.30}$$

, then we can obtain the same results by sequential erosion with these small structuring elements[292] as follows:

$$X \ominus S' = (...(X \ominus S'_1) \ominus S'_2...) \ominus S'_n \tag{4.31}$$

A Euclidean distance structuring element S_E can be segmented into the point-wise maximum selection of multiple linearly sloped structuring components[2] S_i[189] as follows:

$$-S_E = \max(S_1, S_2, ..., S_n) \tag{4.32}$$

[2] A structuring element is said to be linearly sloped when it is contained in a piecewise linear hyper-plane.

where S_i is of size $(2i+1) \times (2i+1)$, and the entries outside the window are regarded as $-\infty$[292, 129]. Let $S_i(k,l)$ denote the *(k,l)-th* entry of S_i, then Eq.(4.32) can be recast as:

$$-S_E(k,l) = \max(S_1(k,l), S_2(k,l), ..., S_n(k,l)) \tag{4.33}$$

Then, the gray-scale erosion of X with a Euclidean distance structuring element is equivalent to the minimum of the outputs when the image is individually eroded with these structuring components, which can be expressed as follows:

$$X \ominus (-S_E) = \min(X \ominus S_1, X \ominus S_2, ..., X \ominus S_n) \tag{4.34}$$

Since each structuring component $S_i, i = 1, ..., n$, in Eq.(4.32) has linear slopes, it can be further decomposed by using the method in Eqs.(4.30) and (4.31) into dilation of its structuring subcomponents $S_{ij}, i = 1, ..., n; j = 1, ..., i$; such that:

$$S_i = S_{i1} \oplus S_{i2} \oplus ... \oplus S_{ii}, \quad i = 1, ..., n \tag{4.35}$$

Every $S_{ij}, i = 1, ..., n, j = 1, ..., i$, is a 3×3 structuring element, which is given by:

$$S_{i1} = -\lambda \begin{bmatrix} i\sqrt{2} & \sqrt{i^2+(i-1)^2} & i\sqrt{2} \\ \sqrt{i^2+(i-1)^2} & x & \sqrt{i^2+(i-1)^2} \\ i\sqrt{2} & \sqrt{i^2+(i-1)^2} & i\sqrt{2} \end{bmatrix} \tag{4.36}$$

where

$$x = \begin{cases} 0, & i = 1 \\ \text{don't care}, & i > 1 \end{cases} \tag{4.37}$$

and

$$S_{ij} = \lambda \begin{bmatrix} 0 & a & 0 \\ a & y & a \\ 0 & a & 0 \end{bmatrix}, \quad j = 2, 3, ..., i \tag{4.38}$$

where $a = \sqrt{i^2+(i-j+1)^2} - \sqrt{i^2+(i-j)^2}$ and y denotes a *don't care* element. In this case, λ can be chosen as $\lambda \leq \frac{1}{\sqrt{2}n}$ such that all entries in S_E are not bigger than 1. An example of this kind of decomposition can be found in [303]. The block diagram of Euclidean distance transformation based on this decomposition is shown in Fig. 4.7. One can see that all the structuring elements are of size 3×3.

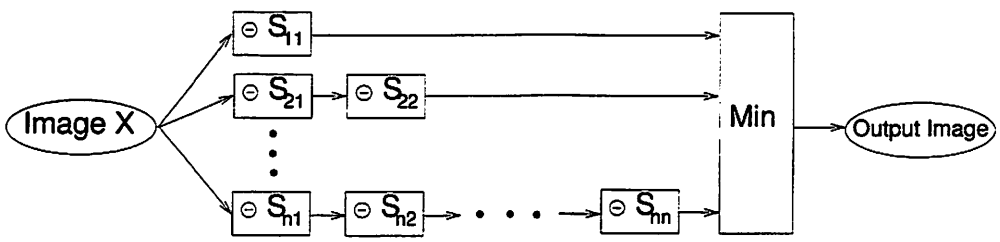


Figure 4.7: The block diagram of Euclidean distance transformation using decomposed structuring element algorithm.

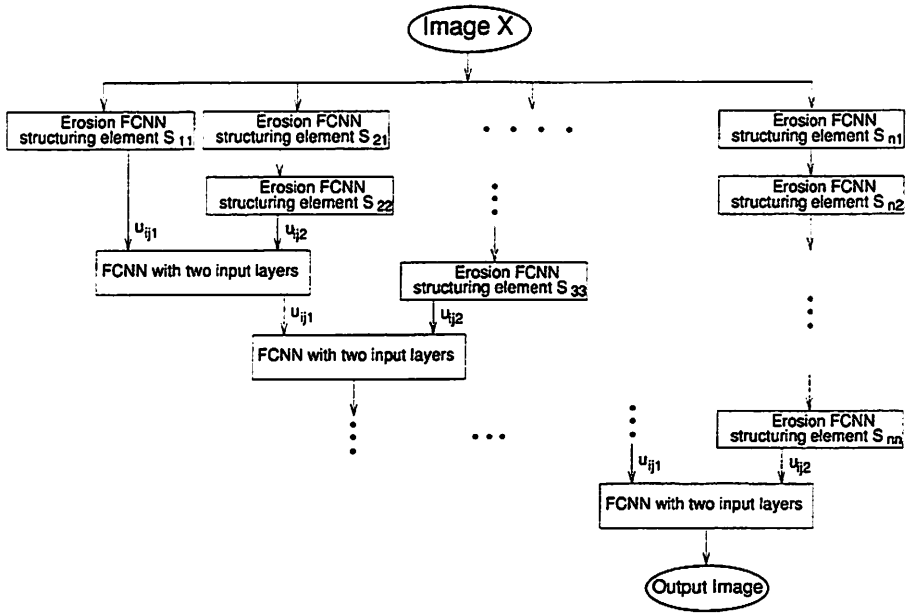


Figure 4.8: The flow-chart of CNN series for implementation of Euclidean distance transformation using FCNN's.

We use the flow-chart of CNN series as shown in Fig. 4.8 to implement this algorithm. One can see that this flow-chart has the same structure as the block diagram shown in Fig. 4.7 except that the "MIN" block with n inputs in Fig. 4.7 is replaced by $n-1$ FCNN's each with two input layers. In Fig. 4.8, the FCNN with two input layers implements *min* operation between its two inputs. This FCNN is given by:

$$\dot{x}_{ij} = -x_{ij} + \min(u_{ij1}, u_{ij2}) \tag{4.39}$$

where $\{u_{ij1}\}$ and $\{u_{ij2}\}$ denote its two inputs. The output image is the final result when all the FCNN's settle down.

Since FCNN is a parallel, dynamic, computational array, no matter what the size of the input image is, it will finish its operation within a characteristic time, τ_{CNN}. In Fig. 4.8 $\frac{n(n+1)}{2}$ erosion FCNN's and other $n-1$ FCNN's are used, the time complexity of our algorithm is $(0.5n^2 + 1.5n - 1)\tau_{CNN}$. τ_{CNN} has the order of 10^{-9}s, one can see that the time complexity is very small.

Fig. 4.9 shows computer simulation results. Fig. 4.9(a) shows a binary image of a Chinese character "YANG" of size 64×64. Fig. 4.9(b) shows the output image of the flow-chart shown in Fig. 4.8 with $n = 5, \lambda = \frac{1}{5\sqrt{2}}$. The result is represented by 256 gray levels. In this case, the time complexity is 19 τ_{CNN}.

In this section, FCNN is used to implement Euclidean distance transformation, which is a global problem. So, the results in this paper demonstrate that FCNN can also solve global problems using a flow-chart of FCNN's. Since distance transformation has found applications to skeleton extraction and shape factor extraction, the FCNN's can be used to offset the computational loads in these applications.

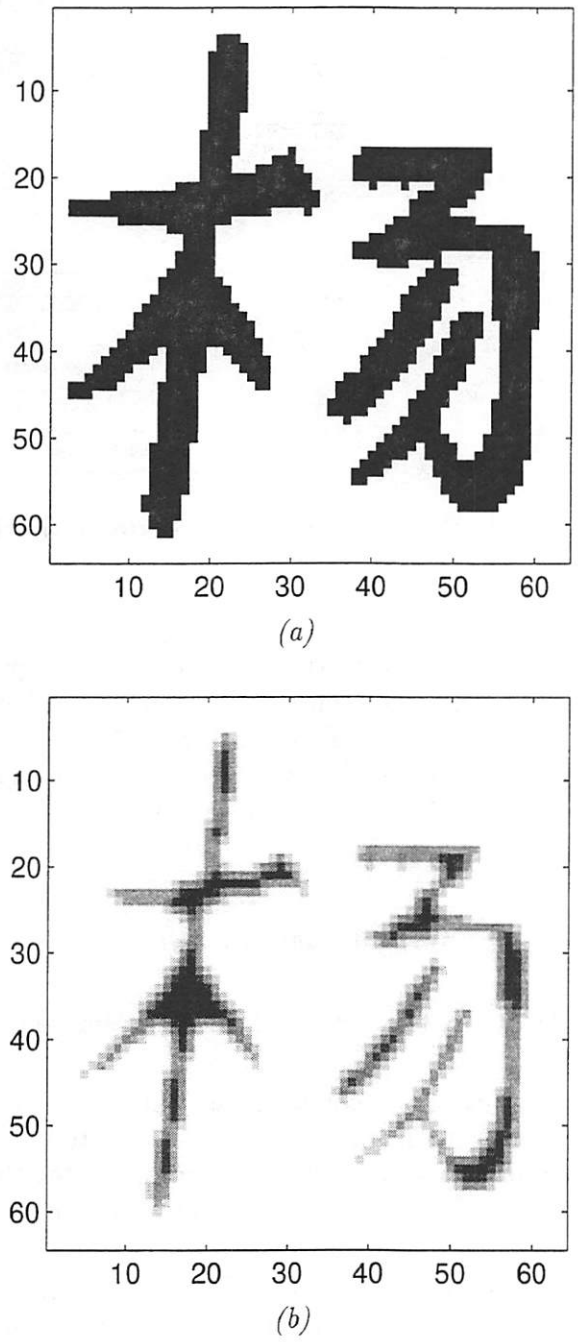


Figure 4.9: Implementation of Euclidean distance transformation using FCNN. (a) A binary image of a Chinese character "YANG". (b) Output image of the FCNN based Euclidean distance transformation algorithm.

4.4 Compare with the conventional CNN

In this section, the differences between CNN based and FCNN based mathematical morphology operations are presented. The performances of the CNN-based mathematical morphology operations are analyzed. The stability and the basin of attraction of CNN-based erosion and dilation are studied. The CNN-based erosion and dilation may introduce some time-varying errors in their outputs. For comparison, the performances of the FCNN based gray-scale mathematical morphology operations is also presented.

When binary mathematical morphology operations is concerned about, there are three kinds of CNN structures can be used. That a DTCNN[120] can be used to implement morphological operations is based on the decompose of the basic erosion and dilation with a threshold in (0, 1)[358]. That a CNN can be used to implement morphological operations is based on the saturation operation regions in the standard cell[395]. That a FCNN[350, 351, 352, 354, 353, 356, 355] can be used to implement morphological operations is based on the correspondence of OR and AND to FUZZY OR and FUZZY AND when only boolean set is encountered [351, 352]. Of course, using FCNN in this case is something funny like a gun being used to hit a fly.

The difference between the DTCNN-based and the CNN-based methods lies in the fact that in a DTCNN structure, a single template can be used to implement both the erosion and dilation operations while the input function and the output function are switched between two configurations. This structure is most suitable for VLSI implementation with fixed parameters. On the other hand, the CNN structure needs two sets of templates to implement erosion and dilation, respectively. Since an analog CNN chip with the ability of dynamically programming templates is far from practical applications, the results may remain a long time as a computer simulation which is much slower than a specially designed digital binary morphology chip. So, the DTCNN maybe the only CNN implementation of binary mathematical morphology in the near future.

When we concern about the gray-scale mathematical morphology, both CNN and FCNN can be used. When the CNN-based method in [395] is used, every synaptic law is a threshold type nonlinear function of difference between output and inputs with a programmable threshold, which corresponds to the gray value of the entries of the structuring element. For example, in a 3×3 structuring element, a cell has 9 threshold type nonlinear synaptic laws and other 9 minus operations. When an FCNN [351, 352] is used, only a *min* (or *max*) operation is needed. So, the FCNN implementation is much simpler than that of CNN. Since the CNN structure proposed in [395] is structurally unstable, there exist some problems in its VLSI implementation. On the other hand, this structure is very sensitive to initial conditions and in some case it will provide unstable and error results. However, the FCNN implementation is globally and asymptotically stable and its output is error free. In this section, we show that the FCNN-based mathematical morphology is superior to the CNN-based one.

In [395], the authors proposed two CNN structures to implement the basic gray-scale morphological operations. The CNN-based erosion is given by:
State equation:

$$\dot{x}_{ij} = -x_{ij} + y_{ij} + \sum_{C_{kl} \in N_r(ij} D^E_{kl}(u_{kl} - x_{ij}) + 1, \quad x_{ij}(0) = -1 \tag{4.40}$$

where the function $D^E_{kl}(.)$ is given by:

$$D_{kl}^{E}(x) = \begin{cases} 0, & x > S_{ij}(kl) \\ -1, & x \leq S_{ij}(kl) \\ 0, & S_{ij}(kl) \text{ undefined} \end{cases} \tag{4.41}$$

where $S_{ij}(kl)$ is defined by:

$$S_{ij}(kl) = s(k-i, l-j), \quad (k-i, l-j) \in S \tag{4.42}$$

The output equation is given by:

$$y_{ij} = 1/2(|x_{ij}+1| - |x_{ij}-1|) \tag{4.43}$$

And the CNN-based dilation is given by the following state equation

$$\dot{x}_{ij} = -x_{ij} + y_{ij} + \sum_{C_{kl} \in N_r(ij} D_{kl}^{D}(u_{kl} - x_{ij}) - 1, \quad x_{ij}(0) = 1 \tag{4.44}$$

where the function $D_{kl}^{D}(.)$ is given by:

$$D_{kl}^{D}(x) = \begin{cases} 1, & x \geq S_{ij}^{*}(kl) \\ 0, & x < S_{ij}^{*}(kl) \\ 0, & S_{ij}^{*}(kl) \text{ undefined} \end{cases} \tag{4.45}$$

where $S_{ij}^{*}(kl)$ is defined by:

$$S_{ij}^{*}(kl) = s(i-k, j-l), \quad (i-k, j-l) \in S \tag{4.46}$$

The output equation is the same as that in Eq.(4.43). Without loss of generality, in this paper we only study the 0 height flat structuring element of size 3×3, i.e.,

$$S = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4.47}$$

then the erosion CNN in Eq.(4.40) can be written as:

$$\dot{x}_{ij} = -x_{ij} + y_{ij} + \sum_{C_{kl} \in N_1(ij)} d_E(u_{kl} - x_{ij}) + 1 \tag{4.48}$$

where the function $d_E(.)$ is given by:

$$d_E(x) = \begin{cases} 0, & x > 0 \\ -1, & x \leq 0 \end{cases} \tag{4.49}$$

Assume that the input u_{ij} is normalized to $(-1, 1)$, then the equilibrium point is given by:

$$\sum_{C_{kl} \in N_1(ij)} d_E(u_{kl} - x_{ij}) = -1 \tag{4.50}$$

We sort u_{kl} in $N_1(i, j)$ by a non-increasing order as $\{u_1, u_2, ..., u_8, u_9\}$, and assume that $u_1 \neq u_2$. We then study the stability of this equilibrium point. We have the following theorem:

Theorem 16

Given $u_1 \neq u_2$, u_1 is an asymptotically stable equilibrium point in the basin of attraction $(-\infty, u_1)$.

Proof: If $u_1 \neq u_2$, then it is easy to see that $x_{ij}^* = u_1$ is a solution of Eq.(4.50). We construct the following Lyapunov function:

$$V_{ij}(t) = \frac{1}{2}(x_{ij} - u_1)^2 > 0 \tag{4.51}$$

where $x_{ij} \in (-\infty, u_1)$. Differential $V_{ij}(t)$ along solutions of Eq.(4.48), we have:

$$\begin{aligned} \frac{dV_{ij}(t)}{dt}\Big|_{Eq.(4.48)} &= (x_{ij} - u_1)\dot{x}_{ij} \\ &= (x_{ij} - u_1)(-x_{ij} + y_{ij} + \sum_{C_{kl} \in N_1(ij)} d_E(u_{kl} - x_{ij}) + 1) \\ &= (x_{ij} - u_1)(-x_{ij} + y_{ij} + 1) < 0 \end{aligned} \tag{4.52}$$

The last inequality is in view of $x_{ij} < u_1$. It is easy to see that since $x_{ij} < u_1 < 1$, we have $(-x_{ij} + y_{ij} + 1) > 0$. □

When $u_1 \neq u_2$, it is very easy to see that any $x_{ij} \in [u_1, u_2)$ is an equilibrium point of Eq.(4.48). In case when $u_2 - u_1$ is very big, any noise in the circuit will make the cell output any value in $[u_1, u_2)$. So, the performance of this circuit is not very good. On the other hand, this circuit is structurally unstable. In a VLSI implementation, a very small positive error in A template will eventually blow up the state.

When $u_1 = u_2 = ... = u_i, i \leq 9$, the cell in Eq.(4.48) has no equilibrium point. In fact, the cell will fluctuate around an *m-equilibrium point*[348], which will stay at u_1 with a probability of zero measurement[357].

Also, when the structuring element in Eq.(4.47) is used, the dilation CNN proposed in Eq.(4.44) can be written as:

$$\dot{x}_{ij} = -x_{ij} + y_{ij} + \sum_{C_{kl} \in N_1(ij)} d_D(u_{kl} - x_{ij}) - 1 \tag{4.53}$$

where the function $d_D(.)$ is given by:

$$d_D(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{4.54}$$

Assume that the input u_{ij} is normalized in $(-1, 1)$, then the equilibrium point is given by:

$$\sum_{C_{kl} \in N_1(ij)} d_D(u_{kl} - x_{ij}) = 1 \tag{4.55}$$

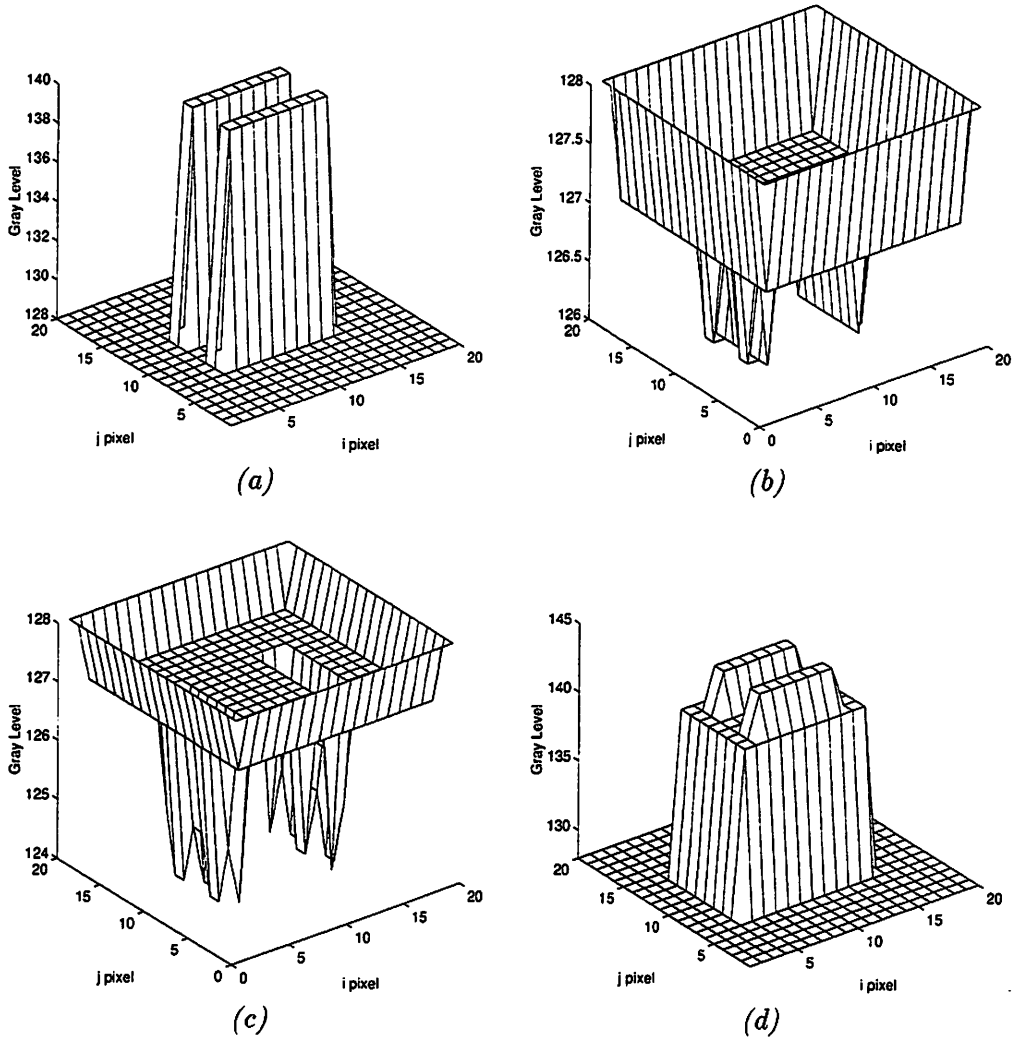
We then have the following theorem.

Theorem 17

Given $u_8 \neq u_9$, u_9 is an asymptotically stable equilibrium point in the basin of attraction $(u_9, +\infty)$.

Proof: If $u_8 \neq u_9$, then it is easy to see that $x_{ij}^* = u_9$ is a solution of Eq.(4.55). We construct the following Lyapunov function:

$$V_{ij}(t) = \frac{1}{2}(x_{ij} - u_9)^2 > 0 \tag{4.56}$$



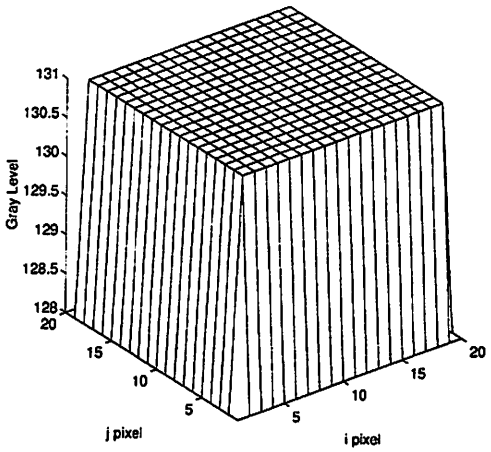
where $x_{ij} \in (u_9, +\infty)$. Differential $V_{ij}(t)$ along solutions of Eq.(4.53), we have:

$$\begin{aligned} \frac{dV_{ij}(t)}{dt}\Big|_{Eq.(4.53)} &= (x_{ij} - u_9)\dot{x}_{ij} \\ &= (x_{ij} - u_9)(-x_{ij} + y_{ij} + \sum_{C_{kl} \in N_1(ij)} d_D(u_{kl} - x_{ij}) - 1) \\ &= (x_{ij} - u_9)(-x_{ij} + y_{ij} - 1) < 0 \end{aligned} \tag{4.57}$$

The last inequality is in view of $x_{ij} > u_9$. It is easy to see that since $x_{ij} > u_9 > -1$, we have $(-x_{ij} + y_{ij} - 1) < 0$. □

If $u_8 \neq u_9$, then any $x_{ij} \in (u_8, u_9]$ is an equilibrium point of Eq.(4.53). Also, when $u_9 = u_8 = ... = u_j$, $j \geq 1$, the cell in Eq.(4.53) has no equilibrium point.

To show the output errors of the above CNN-based erosion and dilation, the following simulations shown in Fig. 4.10 are used. Fig. 4.10 (a) shows an artificial image of size 20×20 with 256 gray-level. In this simulation, we normalize the input image to $(-1, 1)$. So, the voltage $0v$ corresponding to the 128th gray level. And a gray-level corresponding to about $8mv$. It is easy to see that with the structuring element in Eq.(4.47), the erosion will make all the pixels to the 128th gray level, and the dilation will make all the pixels



(e)

Figure 4.10: The output results of CNN-based mathematical morphology operations proposed in [395] (a) An artificial image. (b) The output of erosion CNN in Eq.(4.40) at $t = 4s$. (c) The output of erosion CNN in Eq.(4.40) at $t = 4.3s$. (d) The output of dilation CNN in Eq.(4.44) at $t = 4s$. (e) The output of dilation CNN in Eq.(4.44) at $t = 4.5s$.

in the center region to the 140th gray level and all the pixels in the outer region to the 128th gray level. Fig. 4.10(b) shows the output result of CNN in Eq.(4.40) at $t = 4s$. The initial state is -1. One can see that this output is totally different from the standard erosion result which should be a flat surface with height 128. However, since in this case the CNN in Eq.(4.40) is not stable, it will oscillate forever. To demonstrate this, we show the snapshot at $t = 4.3s$ in Fig. 4.10(c). One can see that Fig. 4.10(c) is totally different from Fig. 4.10(b) and there exist more errors in Fig. 4.10(c). From above we know that the erosion results given by the CNN in Eq.(4.40) are "random" images with uncontrollable and unpredictable errors which are decided by the input images and the stop moments.

The same can be addressed to the dilation CNN in Eq.(4.44). Fig. 4.10(d) shows the output of CNN in Eq.(4.44) at $t = 4s$. And Fig. 4.10(e) shows the output result at $t = 4.5s$. One can find that it is hard to believe the output results of the CNN in Eq.(4.44) because the results in Fig. 4.10(d) and Fig. 4.10(e) share no similarity. In the above simulations, a fourth order Roung-Kuta method with step size 0.01 is used.

When the input image is a real image, the errors in the output results of the CNN's in Eqs.(4.40) and (4.44) are very hard to be predicted and controlled. The simulation results are shown in Fig. 4.11. Fig.4.11(a) shows a face image of a Chinese girl of size of 63×63 with 256 gray-levels. Also, the structuring element as in Eq.(4.47) is used. The difference of gray value of the output of the CNN in Eq.(4.40) and the standard erosion at $t = 4s$ is shown in Fig.4.11(b). One can see that there exist lots of errors. The largest error is equal to -5 gray levels. The initial state is -1. The difference of gray value of the output of the CNN in Eq.(4.44) and the standard dilation at $t = 4s$ is shown in Fig.4.11(c). Also, one can see that there exists lots of errors. The largest error is equal to 5 gray levels. The initial state is 1. In the above simulations, a fourth order Roung-Kuta method with step size of 0.01 is used.

We then use the FCNN in Eq. (4.10) to implement erosion and dilation. It's very easy to see that the above two FCNN's are globally and asymptotically stable. On the other hand, a local max/min operation is much easier to be implemented than 9 programmable

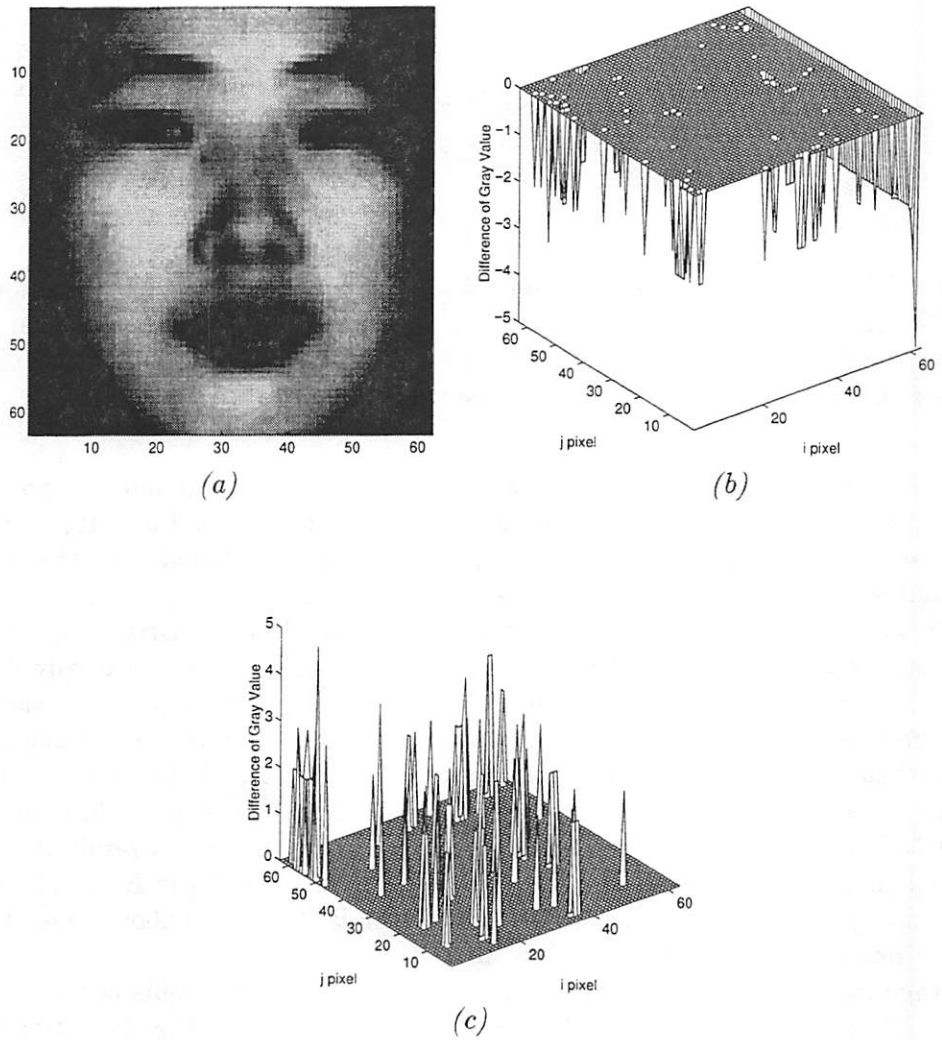


Figure 4.11: The output results of CNN-based mathematical morphology operations proposed in [395]. (a) The gray-scale image of a Chinese girl. (b) The difference of gray level between output of CNN in Eq.(6.60) and the standard erosion at $t = 4s$. (c) The difference of gray level between output of CNN in Eq.(4.44) and the standard dilation at $t = 4s$.

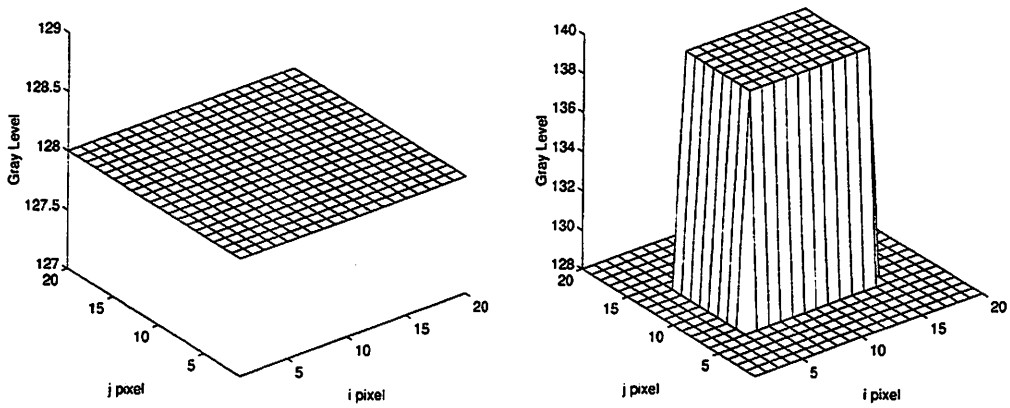


Figure 4.12: The output results of FCNN-based mathematical morphology operations proposed in [351, 352]. (a) The output of erosion FCNN in Eq.(4.10) at $t = 3s$. (b) The output of dilation FCNN in Eq.(4.10) at $t = 3s$.

thresholding nonlinear synaptic laws. The output results of the FCNN-based erosion and dilation are error free.

Fig. 4.12(a) shows the output of the erosion FCNN at $t = 3s$. The structuring element as that in Eq.(4.47). Fig. 4.12(b) shows the output of the dilation FCNN at $t = 3s$. In both simulations, the input is that image shown in Fig. 4.10(a). The initial conditions are arbitrary. One can see that the FCNN-based erosion and dilation give the error free results.

In this chapter, the performances of the existed CNN and FCNN implementations of mathematical morphology operations are studied. We find that the CNN-based mathematical morphology operations proposed in [395] output results with time-varying errors which are input image dependent. And the CNN structures in [395] is structurally unstable. The error free and globally asymptotically stable implementations of morphological operations are based on FCNN[351, 352]. Although we don't argue that the FCNN-based mathematical morphology operations are the only choices of the future CNN-based morphology engine, if one want an error free and totally stable CNN-based morphology engine, so far the FCNN is the best choice.

4.5 Other applications to image processing

In this section, we present some applications of type-II FCNN as $min - max$ network and some applications of type-II FCNN. Although $min - max$ network is a special case of mathematical morphology network, they give us examples to show how natural an FCNN can be translated to this kind of applications. Since the applications of type-I and even type-III FCNN's are still an unexplored field, the other examples presented here are used to show what kind of potentials a type-I FCNN may have.

4.5.1 Fuzzy shrinking and expanding using type-II FCNN

"Shrinking" and "expanding" operations on two-valued digital image are useful for noise removal and segmentation[255]. The authors of [193] had generalized these operations to

process gray-scale images. For simplicity, we use two-valued images to demonstrate how the shrinking and expanding operations work. Let X be a two-valued image which suffers from salt-and-pepper noise, so, X contains regions consisting primarily of 0's with a sprinkling of isolated 1's and vice versa. Any set of 1's that is no more than two pixels wide will removal by using a shrinking operation, in which 1's are changed to 0's if they have 0's as neighbors, followed by an expanding operation, in which 0's are changed to 1's if they have 1's as neighbors. Similarly, expanding following by shrinking destroys sets of 0's that are at most two pixels wide.

More generally, we can use k repetitions of shrinking followed by K repetitions of expanding to eliminate sets of 1's that are at most $2k$ pixels wide. Shrinking and expanding operations can be used to detect elongated parts of objects in an image. They can also be used to detect the clusters or dense regions in an image composed of isolated 1's on a background of 0's (or, vice versa).

If we perform k repeated expansions where k is at least half the distance between the 1's in a cluster, the cluster will "fuss" into a solid region; when we subsequently perform k repeated shrinks, this region will remain large. On the other hand, 1's that don't belong to cluster will expand but not fuse with other 1's, so that when they are shrunk, they shrink back to single 1's.

In two-valued image, shrinking the 1's is equivalent to computing the logical AND of each pixel with its neighbors, and expanding the 1's is equivalent to logically ORing each pixel with its neighbors. So, we can generalize shrinking and expanding operations in the case of gray-scale image by using fuzzy AND, $\tilde{\wedge}$, and fuzzy OR, $\tilde{\vee}$, to replace the logical AND and logical OR. We then present the FCNN's for implementing fuzzy shrinking and fuzzy expanding.

We use the type-II FCNN in Eq.(2.32) to implement fuzzy shrinking and fuzzy expanding. The template for implementing fuzzy shrinking in a 1-neighborhood system is given by:

$$A = 0, B = 0, I = 0, A_{fmin} = 0, A_{fmax} = 0, R_x = 1,$$
$$B_{fmax} = 0, B_{fmin} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{4.58}$$

The template for implementing fuzzy expanding in a 1-neighborhood system is given by:

$$A = 0, B = 0, I = 0, A_{fmin} = 0, A_{fmax} = 0, R_x = 1,$$
$$B_{fmin} = 0, B_{fmax} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{4.59}$$

The inputs of the two FCNN's in Eqs.(4.58) and (4.59) are gray-scale images, initial states are arbitrary.

Fig. 4.13 shows the results of noise removal by repeatedly applying shrinking FCNN in Eq.(4.58) and then repeatedly applying expanding FCNN in Eq.(4.59) to an artificial image. Fig. 4.13(a) shows an artificial image of size 64×64 and 256 gray levels consisting of a noisy white rectangle on a noisy background. The mean gray levels of the rectangular and the background are 220 and 140, respectively, with the deviation 30 for each. Fig. 4.13(b)

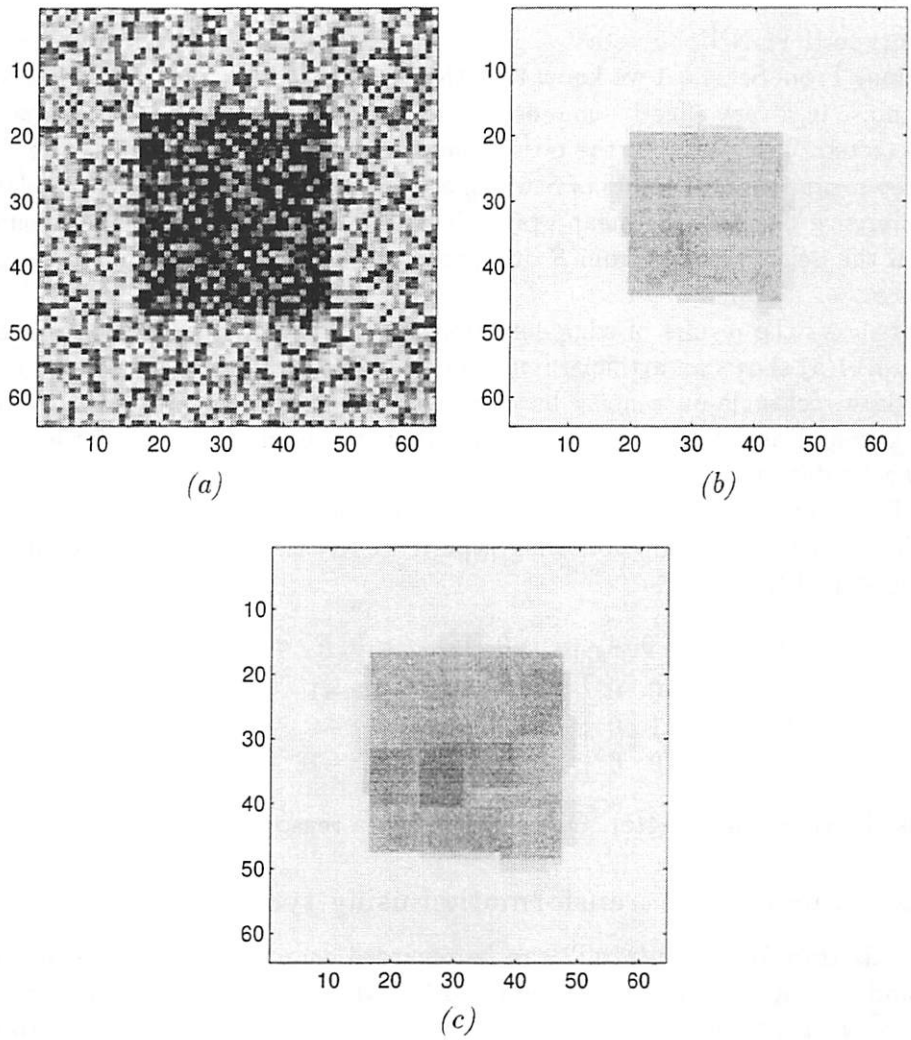


Figure 4.13: Using shrinking FCNN and expanding FCNN to remove noise in a gray-scale image. (a) The noisy image containing a noisy bright square in a noisy dim background. (b) The result of applying the shrinking FCNN 3 times to the image shown in (a). (c) The result of applying the expanding FCNN 3 times to the image shown in (b).

shows the result of repeatedly applying shrinking FCNN for three times. Fig. 4.13(c) shows the result of repeatedly applying expanding FCNN for three times to the image in Fig. 4.13(b). One can see that the noises are removed.

It is easy to see that shrinking and expanding is closely related to the erosion and the dilation operations which we have presented in Sec.4.2.

4.5.2 Edge detection using type-II FCNN

Now we use type-II FCNN's to detect edges of an object from its background under a low SNR condition. From Sec.4.5.1 we know that the fuzzy shrinking and expanding operators can remove noise in a very effective manner. They are used to pre-process the image before edges are detected. Let S denote the object and $\overline{S}$ denote its complement. Then, a single shrinking step means that all points of S which are neighbors of points in $\overline{S}$ are deleted from S. If the difference between the mean gray values of the object and the background is big enough, then the deleted points from S during a single shrinking operation is a reasonable boundary of S.

Fig. 4.14 shows the results of edge detection under a low SNR condition using type-II FCNN. Fig. 4.14(a) shows an artificial image of size 64×64 and 256 gray levels consisting of a noisy white rectangle on a noisy background. The mean gray levels of the rectangle and the background are 220 and 190, respectively, with the deviation 30 for each. Before the edges can be detected, the fuzzy shrinking CNN in Eq.(4.58) and the fuzzy expanding CNN as in Eq.(4.59) are repeatedly applied 7 times each to remove noise. The result is shown in Fig. 4.14(b). Then the following type-II FCNN is used to get edges from image shown in Fig. 4.14(b).

$$A = 0, I = 0, A_{fmin} = 0, A_{fmax} = 0, R_x = 1, B_{fmin} = 0$$
$$B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B_{fmax} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & -1 & -1 \end{bmatrix} \tag{4.60}$$

The result is shown in Fig. 4.14(c). One can see that a reasonable edge is detected.

4.5.3 Fuzzy medial axis transformation using type-II FCNN

The medial axis transformation(MAT) can be regarded as a generalized axis of symmetry of a figure, and constitutes a kind of "skeleton"[226]. MAT can be defined by a propagation process toward the inside of a figure. The contour is the initial wavefront of the propagation process, and the propagation velocity is fixed. Wavefront superposition is not allowed, and wavefront intersection points are the points of the MAT. In the case of gray-scale image, the propagation of a wavefront can be modeled by a sequence of fuzzy shrink operation, and the MAT can be constructed by a simple process of iterated fuzzy shrinking and fuzzy expanding.

Let X be a gray-scale image and X^k denotes the result of expanding X k times. Similarly, let X^{-k} denote the result of shrinking X k times. Then it's easy to see that

$$(X^{-i})^j \leq X^{j-i} \leq (X^j)^{-i} \tag{4.61}$$

where i and j are nonnegative integers. The difference result

$$D_k = X^{-k+1} - (X^{-k})^1 \tag{4.62}$$

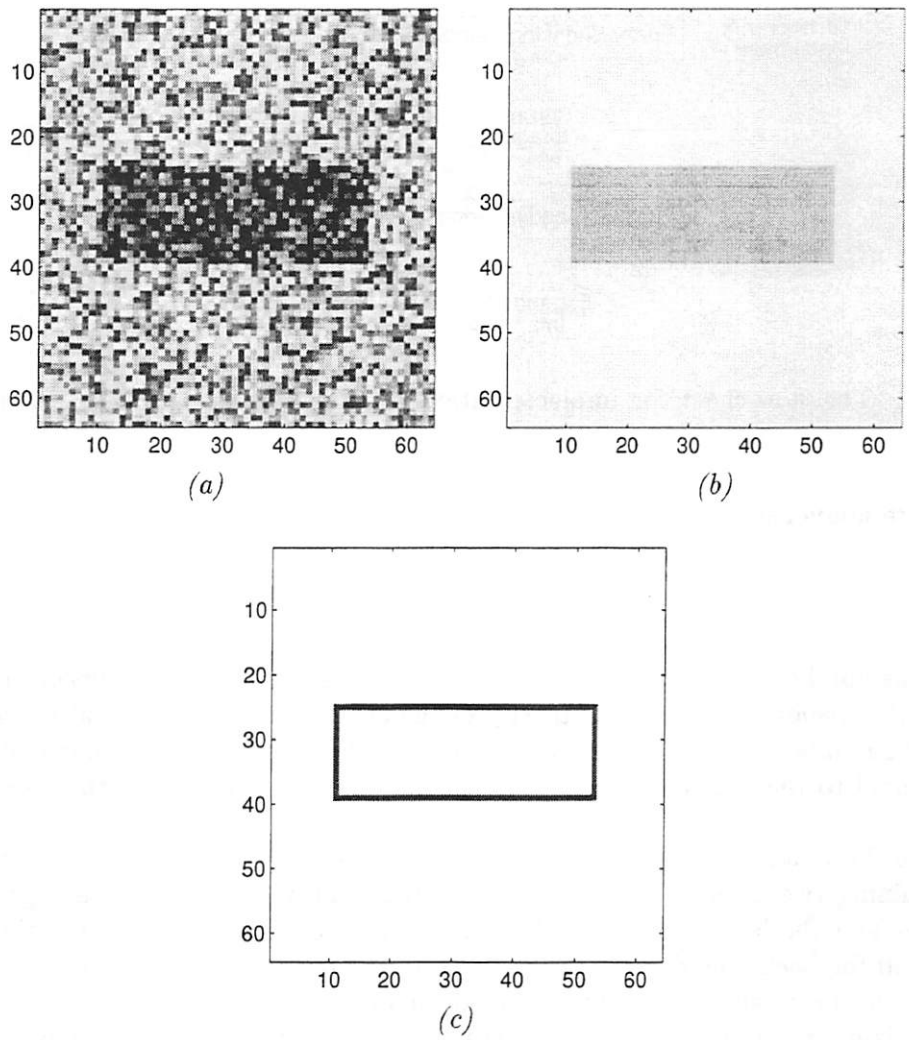


Figure 4.14: Edge detection under a low SNR condition using type-II FCNN. (a)An artificial image containing a noisy white rectangle on a noisy background. (b) Result of noise removal using shrinking FCNN and expanding FCNN 7 times each. (c) Result of edge detection using FCNN.

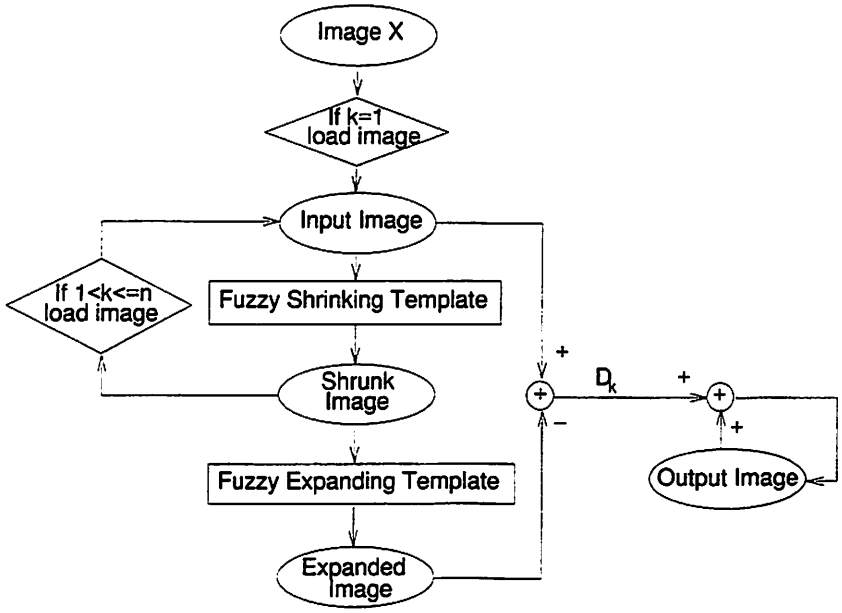


Figure 4.15: The flow-chart for implementation of fuzzy MAT algorithm using type-II FCNN.

is everywhere nonnegative. Then the fuzzy MAT is given by

$$\sum_{k=1}^{n} D_k \tag{4.63}$$

where n is the number of iterations, which is somewhat greater than the radius of the largest high gray value region. The time complexity of the algorithm is proportional to the image size times the number of iterations. However, by using FCNN's, the time complexity would be proportional to the number of iterations. The flow-chart of this algorithm is shown in Fig. 4.15.

Fig. 4.16 shows a gray-scale image and its fuzzy MAT. Fig. 4.16(a) shows a gray-scale image containing two circles of size 128×128 and 256 gray levels. The mean gray levels of the circles and the background are 230 and 120, respectively, with the deviation 10 for circles and 30 for background. Fig. 4.16(b) shows the output image of the flow-chart in Fig. 4.15. The fuzzy shrinking template is given by Eq.(4.58) and the fuzzy expanding template is given by Eq.(4.59) One can see that the high fuzzy MAT values constitute very reasonable "skeletons" of the circles.

4.5.4 Face image processing using type-I FCNN

For the purpose of face recognition[39], facial expression animation[358] and synthesis[289], and compression of face image[51, 325], we need to model face images. Two of important steps of modeling face images are face image segmentation and feature extraction. The structural features of a face image include eyebrows, eyes, nose and mouth. For facial expression animation, the deformation of cheeks is of interest.

Although the authors of [358] used a 2-layer conventional CNN to animate facial expressions, they didn't provide the method to locate the key-points, which specify the locations of eyes (eyebrows) and mouth. Of cause, we can locate eyebrows, eyes, nose and mouth

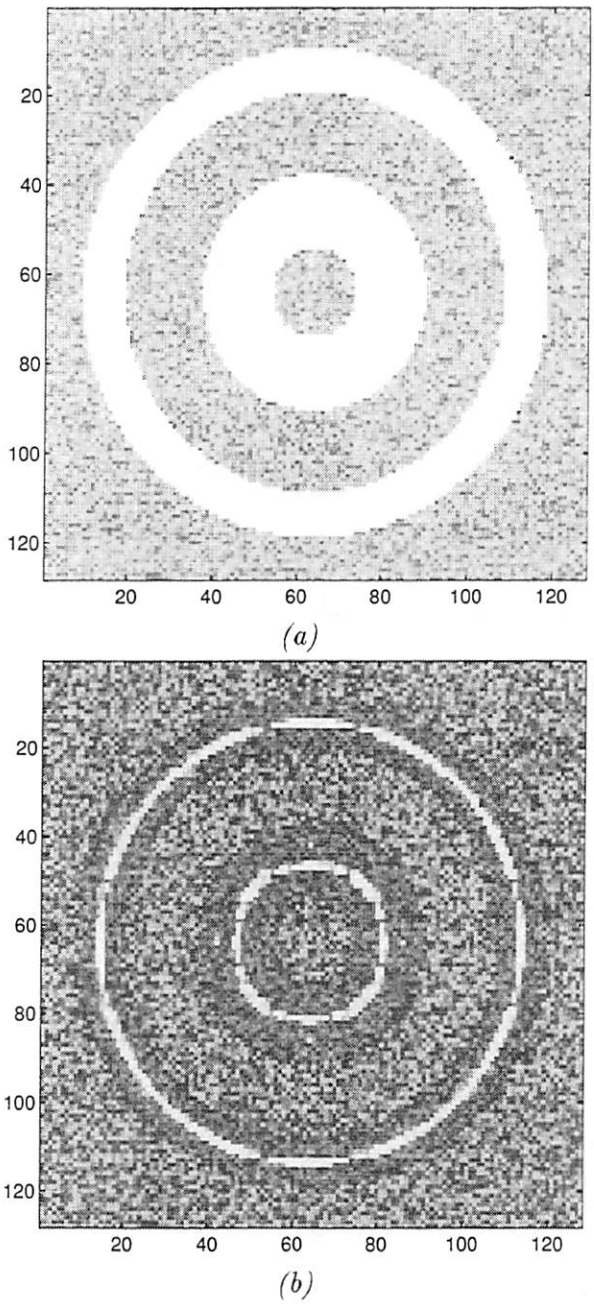


Figure 4.16: Fuzzy medial axis transformation using type-II FCNN. (a) The original image containing two noisy white circles with a dim noisy background. (b) The Fuzzy medial axis transformation of (a).

by using a series of templates which perform the corresponding digital image algorithms as surveyed in [39]. In this subsection, we show how the following two simple type-I FCNN's can be used to solve this problem.

One of them is given by:

State equation

$$\dot{x}_{ij} = -x_{ij} + \tilde{B}_{fmin} \tilde{\odot}_{min} u_{ij} \tag{4.64}$$

Output equation

$$y_{ij} = f(x_{ij}) = \begin{cases} 0, & x_{ij} \leq 0 \\ x_{ij}, & 0 < x_{ij} \leq 1 \\ 1, & x_{ij} > 1 \end{cases} \tag{4.65}$$

The other is given by:

State equation

$$\dot{x}_{ij} = -x_{ij} + \tilde{B}_{fmax} \tilde{\odot}_{max} u_{ij} \tag{4.66}$$

The output equation is the same as in Eq.(4.65).

Example 1: Locating low boundaries of eyebrows, eyes, nose and mouth

In this example, the templates $\tilde{B}_{fmin}$ and $\tilde{B}_{fmax}$ in Eqs.(4.64) and (4.66) are given by:

$$\tilde{B}_{fmin} = \begin{bmatrix} \overline{\mathcal{A}} & \overline{-\mathcal{C}} & \overline{\mathcal{A}} \\ \overline{-\mathcal{C}} & \overline{\mathcal{B}} & \overline{-\mathcal{C}} \\ \overline{\mathcal{A}} & \overline{-\mathcal{C}} & \overline{\mathcal{A}} \end{bmatrix} \tag{4.67}$$

where "-" is an algebraic negative and "$\overline{\mathcal{A}}$" denotes the fuzzy complement of $\mathcal{A}$.

$$\tilde{B}_{fmax} = \begin{bmatrix} \mathcal{A} & -\mathcal{C} & \mathcal{A} \\ -\mathcal{C} & \mathcal{B} & -\mathcal{C} \\ \mathcal{A} & -\mathcal{C} & \mathcal{A} \end{bmatrix} \tag{4.68}$$

where the fuzzy number $\mathcal{A}$ is given by:

$$\mu_{\mathcal{A}}(x) = \begin{cases} 0, & x \leq 0.5 \\ max(0, k(x-1)+1), & 0.5 < x \leq 1 \end{cases} \tag{4.69}$$

the fuzzy number $\mathcal{B}$ is given by:

$$\mu_{\mathcal{B}}(x) = \begin{cases} max(0, -kx+1), & x \leq 0.5 \\ 0, & 0.5 < x \leq 1 \end{cases} \tag{4.70}$$

the fuzzy number $\mathcal{C}$ is given by:

$$\mu_{\mathcal{C}}(x) = \begin{cases} max(0, -mx+1), & x \leq 0.5 \\ max(0, m(x-1)+1), & 0.5 < x \leq 1 \end{cases} \tag{4.71}$$

From above, one can see that the fuzzy numbers $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ are piecewise linear for the purpose of easy VLSI implementation. To show how the definition "$\tilde{\odot}_{min}$" works, we rewrite Eq.(4.64) in an explicit form as:

$$\dot{x}_{ij} = -x_{ij} + \tilde{\bigwedge} \begin{pmatrix} 1-\mu_{\mathcal{A}}(u_{i-1,j-1}), & 1+\mu_{\mathcal{C}}(u_{i-1,j}), & 1-\mu_{\mathcal{A}}(u_{i-1,j+1}), \\ 1+\mu_{\mathcal{C}}(u_{i,j-1}), & 1-\mu_{\mathcal{B}}(u_{i,j}), & 1+\mu_{\mathcal{C}}(u_{i,j+1}), \\ 1-\mu_{\mathcal{A}}(u_{i+1,j-1}), & 1+\mu_{\mathcal{C}}(u_{i+1,j}), & 1-\mu_{\mathcal{A}}(u_{i+1,j+1}) \end{pmatrix} \tag{4.72}$$

Also, to show how the definition "$\tilde{\odot}_{max}$" works, we rewrite Eq.(4.66) in an explicit form as:

$$\dot{x}_{ij} = -x_{ij} + \tilde{\bigvee} \begin{pmatrix} \mu_{\mathcal{A}}(u_{i-1,j-1}), & -\mu_{\mathcal{C}}(u_{i-1,j}), & \mu_{\mathcal{A}}(u_{i-1,j+1}), \\ -\mu_{\mathcal{C}}(u_{i,j-1}), & \mu_{\mathcal{B}}(u_{i,j}), & -\mu_{\mathcal{C}}(u_{i,j+1}), \\ \mu_{\mathcal{A}}(u_{i+1,j-1}), & -\mu_{\mathcal{C}}(u_{i+1,j}), & \mu_{\mathcal{A}}(u_{i+1,j+1}) \end{pmatrix} \tag{4.73}$$

The simulation results are shown in Fig. 4.17. Fig. 4.17(a) shows a face image of a Chinese girl of size 63×63 and 256 gray levels (normalized within [0,1]). This image is fed into the input of FCNN in Eq.(4.66), the output image is shown in Fig. 4.17(b). Then the image in Fig. 4.17(b) is fed into the input of FCNN in Eq.(4.64), the output image is shown in Fig. 4.17(c). From Fig. 4.17(c) one can see that the low boundaries of eyebrows, eyes, nose and mouth are marked by black lines. In this simulation, we choose $k = 2$ and $m = 4$. However, different face images should choose different k's and m's.

Example 2: Segmentation of face

In this example, we show the segmentation of "flat" parts of a face image, which contain cheeks, chin and forehead. The two type-I FCNN's in Eqs.(4.64) and (4.66) are used. The templates $\tilde{B}_{fmin}$ and $\tilde{B}_{fmax}$ are given by:

$$\tilde{B}_{fmin} = \begin{bmatrix} \overline{\mathcal{A}} & \overline{\mathcal{B}} & \overline{\mathcal{A}} \\ \overline{\mathcal{B}} & \overline{\mathcal{B}} & \overline{\mathcal{B}} \\ \overline{\mathcal{A}} & \overline{\mathcal{B}} & \overline{\mathcal{A}} \end{bmatrix} \tag{4.74}$$

$$\tilde{B}_{fmax} = \begin{bmatrix} \mathcal{A} & \mathcal{B} & \mathcal{A} \\ \mathcal{B} & \mathcal{B} & \mathcal{B} \\ \mathcal{A} & \mathcal{B} & \mathcal{A} \end{bmatrix} \tag{4.75}$$

The fuzzy number $\mathcal{A}$ and $\mathcal{B}$ are the same as those in Eqs.(4.69) and (4.70).

The simulation results are shown in Fig. 4.18. Fig. 4.18(a) shows the output results of the FCNN in Eq.(4.64). The input image is that shown in Fig. 4.17(a). Then the image in Fig. 4.18(a) is fed into the input of FCNN in Eq.(4.66), the output image is shown in Fig. 4.18(b). From Fig. 4.18(b) one can see that all the flat regions of face (e.g., forehead, cheeks and chin) are marked by light regions. We choose $k = 2$.

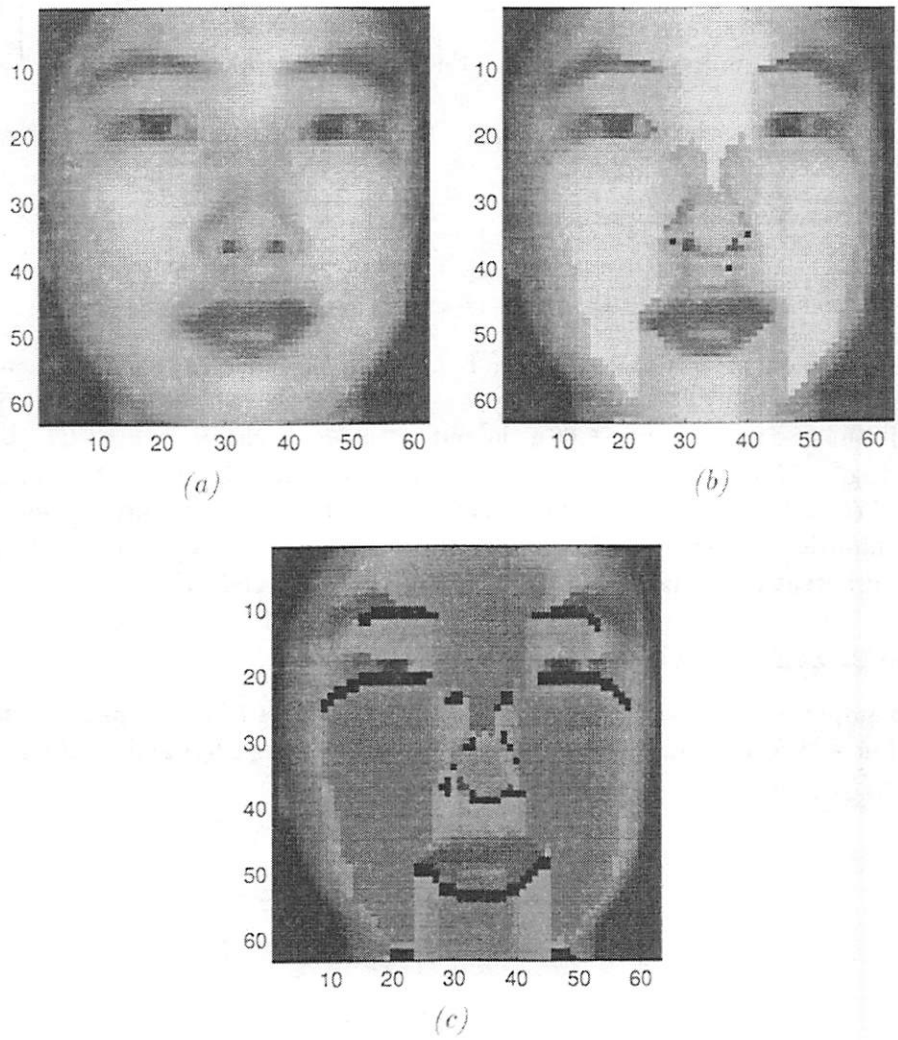


Figure 4.17: Locating structural features of a face image using type-I FCNN. (a) A gray-scale face image of a Chinese girl. (b)The output image of the FCNN in Eq.(4.66). Input is the image in (a). (c) The output image of the FCNN in Eq.(4.64). Input is the image in (b).

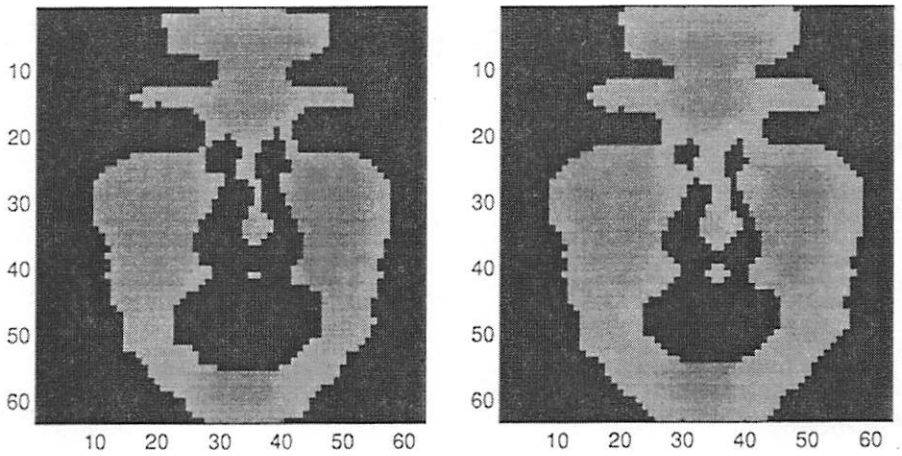


Figure 4.18: Segmentation of a face image. (a) The output image of the FCNN in Eq.(4.64). Input is the image in 4.17(a). (b) The output image of the FCNN in Eq.(4.66). Input is the image in (a).

Chapter 5

Embed Linguistic Statements into FCNN

In this chapter we presented some inner properties of FCNN which functions as an intelligent component in the CNN universe as an interpreter between the high-level knowledge expression and the low level hardware implementation. Comparing the results provided in Chap. 4, we can see that this chapter emphasizes the high-level ability of FCNN structures. And the high-level ability is the significant difference between conventional CNN and FCNN. This chapter is a collection of the following papers [354, 355, 362, 361, 364, 376, 384, 383].

5.1 FCNN: Interfaces between human expert and CNN

In an artificial intelligent system, the motivation is to make a brain model as the core of the system. A conventional CNNUM can't function as this core even though it had been proved to be a Turing machine[54] (Turing machines cannot answer a simple question: "How are you feeling?") and even though it can be used to explain lots of visual phenomena [358, 59, 339](Seeing is not thinking.). On the other hand, the FCNN structure can be used as an interface between the human expert and the conventional CNN. In this sense, the input of an FCNN is the knowledge of human expert which is described by linguistic statements, and the outputs are sets of "templates". In other words, the FCNN is used to translate linguistic or higher level statements which are expressed as fuzzy rules into CNN templates.

5.1.1 Fuzzy set theory and fuzzy property of images

In every phase of image processing, there exist lots of uncertainty[342], e.g., additive and non-additive noise in the sensing and transmission processes; the lose of information while 3D shape or scene is projected into 2D image; lack of the quantitative measurement of image quality and imprecision in computations; ambiguity and vagueness in representations, definitions and interpretation of complex scenes. The fuzzy set theory[393] provides a mathematical strength to capture these uncertainties[155, 142]. It has found wide applications in image processing such as[342, 394, 240, 155, 142, 226, 193]: image modeling, preprocessing, segmentation, object/region recognition and reasoning aspects of image processing problems.

While fuzzy set theory provides an inference mechanism under cognitive uncertainty, the CNN[47, 46] offers advantages such as learning, adaptation, fault-tolerance, parallelism

and generalization. Although fuzzy logic is a natural mechanism for modeling cognitive uncertainty, it may involve an increase in the amount of computation required (compared with a system using digital logic). This can be readily offset by using FCNN, which has the potential for parallel computation with high flexibility.

A fuzzy set A with its finite number of supports $x_i, i = 1, ..., n$, is defined as an ordered pair

$$A = \{(x_i, \mu_A(x_i))\} \tag{5.1}$$

or, in a union form,

$$A = \bigcup_i \mu_i / x_i, i = 1, ..., n \tag{5.2}$$

where the membership function $\mu_A(x_i)$ in the interval [0,1] denotes the degree to which an event x_i may be a member of A. $\mu_A = 0$ represents no membership and $\mu_A = 1$ represents full membership. This characteristic function can be viewed as a weighting coefficient which reflects the ambiguity in A. A *fuzzy singleton* is a fuzzy set which has only one supporting point. In digital image this concept is very useful because a pixel can be viewed as a fuzzy singleton.

The operations on fuzzy sets are extensions of those used for traditional sets. Some of the common operations include comparison, containment, intersection, union and complement. Assuming U to be the universe of discourse, $A \in U$ and $B \in U$, these operations are defined as follows:

Comparison: is $A = B$?

$$A = B \text{ iff } \mu_A(x) = \mu_B(x), \forall x \in U \tag{5.3}$$

Containment: is $A \subset B$?

$$A \subset B \text{ iff } \mu_A(x) < \mu_B(x), \forall x \in U \tag{5.4}$$

Union: The union of two fuzzy sets A and B, $A \vee B$, is given by combining the membership functions of A and B. Although there have been several different union operations defined[341], the most common, and so far the simplest, union is defined as

$$\mu_{A \vee B} = max\{\mu_A(x), \mu_B(x)\}, \forall x \in U \tag{5.5}$$

Intersection: Like the union, the intersection of two fuzzy sets A and B, $A \wedge B$, is given by combining the membership functions of A and B and is defined as

$$\mu_{A \wedge B} = min\{\mu_A(x), \mu_B(x)\}, \forall x \in U \tag{5.6}$$

Complement: The complement of the fuzzy set A , $\overline{A}$, is defined as

$$\mu_{\overline{A}}(x) = 1 - \mu_A(x), \forall x \in U \tag{5.7}$$

In addition to these operations, De Morgan's law, the distributive laws, algebraic operation such as addition and multiplications, and the notion of convexity have fuzzy set equivalents[393].

Remark:

The traditional CNN can implement the following fuzzy operations:

Comparison can be implemented by using a minus operation template and then checking the output is zero or not. **Containment** can be implemented by using a minus operation template and then checking the sign of the output. **Complement** can also be implemented by using a minus operation template. However, **intersection** and **union** can't be implemented by using the traditional CNN, and they can be implemented by using FCNN.

A gray level image possesses ambiguity with each pixel because of the possible multi-valued levels of brightness. If the gray levels are scaled to lie in the region [0,1], we can regard the gray levels of a pixel as its degree of membership in the set of high-valued "bright" pixels—thus a gray image can be viewed as a fuzzy set. Regions, features, primitives, properties, and relations among pixels that are not crisply defined can similarly be regard as fuzzy subsets of images[394, 240].

With the concept of fuzzy set, a gray-scale image X of $M \times N$ pixels and gray levels belong to [0,1] can be considered as an array of fuzzy singletons, each with a value of membership function denoting the degree of having brightness relative to some brightness level in [0,1]. On the other hand, the fuzzy property of an image also comes from the uncertainty of relationship between different pixels. This is the basis for application of FCNN to image processing.

5.1.2 FCNN as an interpreter

The human experts are usually use the linguistic statements to evaluate and describe the image. When we take a picture we may mostly say "get a little blur" instead of "filtering by a low pass filter". Or we most likely to say "there exist some black dots !" instead of "there exist impulsive noises at pixel (3,4), (44,94) and (123,321)".

Based on the fuzzy description of images, lots of fuzzy methods are developed to deal with image processing problems. One can find that a conventional CNN chip is very hard to embed the linguistic if-then rule based fuzzy image processing techniques into the templates. To overcome this problem, we need an interpreter between human experts and the conventional CNN. From lots of previous work[350, 351, 352, 354, 353, 356, 355, 362, 361, 374, 370, 364, 376, 384, 383], we found that FCNN can efficiently embed the fuzzy rules into its structures. This capability of FCNN is found very useful to interpret the linguistic knowledge accumulated these year from the field of fuzzy image processing.

On the other hand, the study ability of FCNN also provides us with a possibility of "teaching" FCNN using linguistic statements and make the design of CNN structures for special image processing tasks much easier. If we think that the only way a human expert communicated with a conventional CNN is the template design or collecting a huge body of date to train the CNN structure, we can find that by teaching an FCNN using our language may be a more direct and easy way than the method we used so far. In this chapter we only presented the method for embedding linguistic statements into FCNN structure. We also found FCNN can learn its structure from linguistic input in Chap. 6.

5.2 Embedding fuzzy inference into FCNN

In [355, 364], the FCNN structure for implementing the following kind of fuzzy IF-THEN rule was presented. The rule is given by:

Rule 1:

IF $K^{(1)}_{c_{kl}\in N_r(i,j)} \circ u_{kl}$ is A_1 and $K^{(2)}_{c_{kl}\in N_r(i,j)} \circ u_{kl}$ is A_2 ... and $K^{(M)}_{c_{kl}\in N_r(i,j)} \circ u_{kl}$ is A_M, THEN y_{ij} is B.

where $K^{(i)}_{c_{kl}\in N_r(i,j)} \circ x_{kl}, 1 \leq i \leq M$, is an algebraic or fuzzy local operation defined in $N_r(i,j)$. $A_i, 1 \leq i \leq M$, is a fuzzy variable. B is a fuzzy variable and the consequent. The corresponding FCNN structure for implementing the above IF-THEN rule is given by:

State equation

$$\begin{aligned} \dot{x}_{ij} &= -x_{ij} + \tilde{\bigwedge}\Big(\mu_{A_1}(K^{(1)}_{c_{kl}\in N_r(i,j)} \circ u_{kl}), \mu_{A_2}(K^{(2)}_{c_{kl}\in N_r(i,j)} \circ u_{kl}) \\ &\quad , ..., \mu_{A_M}(K^{(M)}_{c_{kl}\in N_r(i,j)} \circ u_{kl})\Big) \end{aligned} \tag{5.8}$$

Output equation

$$y_{ij} = \mu_B(x_{ij}) \tag{5.9}$$

where $\mu_{A_i}(.), i = 1, 2, ..., M$, and $\mu_B(.)$ are membership functions of A_i and B, respectively.

In [352, 354], we presented the FCNN structure for embedding the fuzzy inference ruled by else action (FIRE) operators which are a recently proposed family of fuzzy operators for image processing[277, 275]. The FIRE operators are based on a fuzzy IF-THEN-ELSE architecture to perform many important image processing tasks, e.g., image enhancement[279, 276] and edge detecting[277, 275].

First, we introduce the FIRE operators briefly. Consider an L-level gray-scale image U. Suppose u_{ij} is a pixel in U. And u_{kl} is a pixel in the neighborhood of u_{ij}, then we define $x_{kl} = |u_{kl} - u_{ij}|$ as "gray value difference". We also need the membership function of linguistic variable ZERO(ZE), $\mu_{ZE}(x)$, the membership function of linguistic variable WHITE(WH), $\mu_{WH}(x)$, and that of linguistic variable BLACK(BL), $\mu_{BL}(x)$.

In general, an FIRE operator consists of a group of N IF-THEN-rules and one ELSE-rule as:

Rule 2:

IF x_1 is A_{11} and ..... and x_M is A_{1M} THEN y is B_T
.....
IF x_1 is A_{N1} and ..... and x_M is A_{NM} THEN y is B_T
ELSE y is B_E

where M is the number of input variables. A_{ij}, $i = 1, ..., N; j = 1, ..., M$, is the fuzzy set corresponds to the j-th input variable in the i-th THEN-rule. B_T is the common consequent set of the group of THEN-rules, B_E is the consequent set of the ELSE-rule. One can see that every THEN-rule in Rule 2 has the same structure as Rule 1. If B_T is different for every THEN-rule, we can use N layers of FCNN in Eq.(5.8) to implement Rule 2. Since all the THEN-rules have the same consequent, we can use a simpler FCNN structure to implement Rule 2.

Let λ_i be the strength of the i-th THEN-rule in Rule 2, we have

$$\lambda_i = \tilde{\bigwedge_{j=1,...,M}} \mu_{A_{ij}}(x_j) \tag{5.10}$$

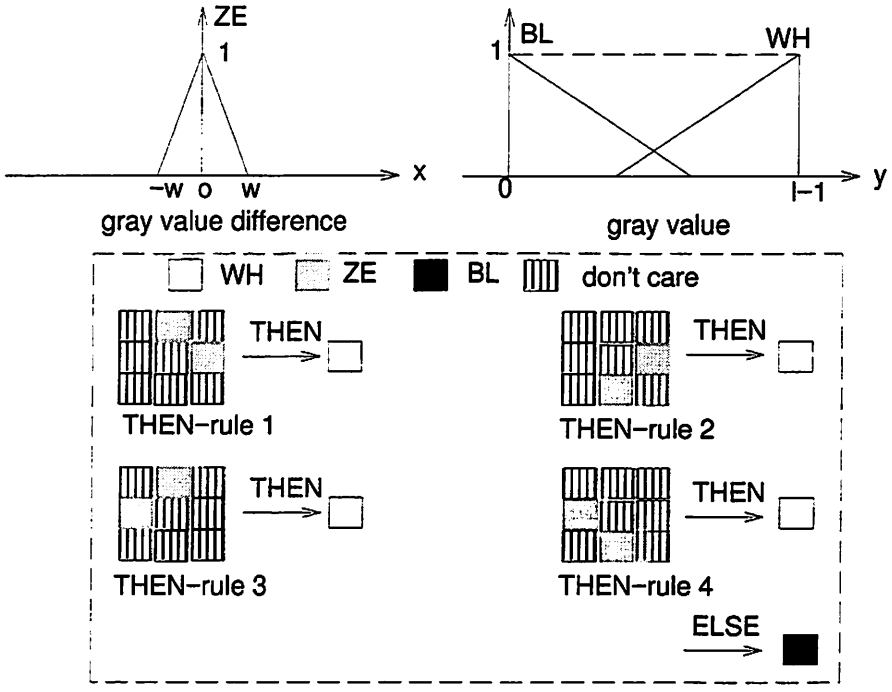


Figure 5.1: Rules for FIRE edge extractor.

where $\mu_{A_{ij}}(.)$ is the membership function of A_{ij}.

Let λ_T be the strength of the THEN-sub-rule in Rule 2, we have

$$\lambda_T = \tilde{\bigvee_{j=1,\ldots,N}} \lambda_i \tag{5.11}$$

Let λ_E be the strength of the ELSE-rule in Rule 2, we have

$$\lambda_E = 1 - \lambda_T \tag{5.12}$$

Finally, the output y is given by a trade off between λ_T and λ_E by using a proper defuzzifier. There is no unique way to perform the defuzzification. And there exist some considerations for choosing defuzzifiers. Several existed methods for defuzzification take into consideration the shape of the clipped fuzzy numbers[30]. Also, the complexity of computations and the possibility of VLSI implementation are taken into account. Usually, according the principles given in [354], the defuzzifier is implemented by the so called output function in the conventional CNN (In our FCNN, it's called an *output membership function* or *defuzzifier function*). So, the simplest defuzzifier is the threshold function when only the binary output is needed.

5.3 Application to image processing

5.3.1 Fuzzy inference edge detector

We then give the FCNN structure which can be used to embed the FIRE edge extractor[277]. An FIRE edge extractor is illustrated in Fig.5.1. In Fig.5.1, the membership functions of ZE, BL and WH are defined as trapezoidal shape. To simplify the structure of FCNN, we

only use λ_T to defuzzify the output. We use only one layer of FCNN to implement all the THEN-rules shown in Fig. 5.1. The state equation is given by:

$$\begin{aligned}
\dot{x}_{ij} &= -x_{ij} + \tilde{\bigvee}[\underset{THEN-rule1}{\tilde{\bigwedge}} \mu_{ZE}(u_{kl}-u_{ij}), \underset{THEN-rule2}{\tilde{\bigwedge}} \mu_{ZE}(u_{kl}-u_{ij}), \\
&\quad \underset{THEN-rule3}{\tilde{\bigwedge}} \mu_{ZE}(u_{kl}-u_{ij}), \underset{THEN-rule4}{\tilde{\bigwedge}} \mu_{ZE}(u_{kl}-u_{ij})] \\
&= -x_{ij} + \tilde{\bigvee}[\underset{kl\in\{(i-1,j),(i,j+1)\}}{\tilde{\bigwedge}} \mu_{ZE}(u_{kl}-u_{ij}), \\
&\quad \underset{kl\in\{(i,j+1),(i+1,j)\}}{\tilde{\bigwedge}} \mu_{ZE}(u_{kl}-u_{ij}), \\
&\quad \underset{kl\in\{(i,j-1),(i-1,j)\}}{\tilde{\bigwedge}} \mu_{ZE}(u_{kl}-u_{ij}), \\
&\quad \underset{kl\in\{(i,j-1),(i+1,j)\}}{\tilde{\bigwedge}} \mu_{ZE}(u_{kl}-u_{ij})]
\end{aligned} \tag{5.13}$$

It should be noticed that a *don't care* pixel in Fig. 5.1 introduces no template relation. One can see that the equilibrium point of the state variable gives the strength of the THEN-rules, λ_T. The output equation of the above FCNN functions as a defuzzifier and is given by:

$$y_{ij} = \begin{cases} 0, & \mu_{WH}(x_{ij}) > h \\ 1, & \mu_{WH}(x_{ij}) \leq h \end{cases} \tag{5.14}$$

where y_{ij} denotes the classical truth value of the pixel to be an edge pixel and $h \geq 0$ is the threshold.

The simulation results are shown in Fig. 5.2. Fig.5.2(a) shows the original gray-scale image of size 63×63 with 256 gray levels. Fig.5.2(b) shows the state of FCNN in Eq.(5.13) which is the fuzzy inference result. Fig.5.2(c) is the corresponding output which is the thresholded(defuzzified) result.

5.3.2 Impulsive noise removing via fuzzy inference

In this section, we consider the impulsive noise removal problem. Median filters are usually used to remove impulsive noise [10, 391]. Unfortunately, a median filter may blur fine structures of an image and cause edge jitter, streaking. To overcome this problem, an efficient method is that the median filter only filter those pixels where impulsive noises are existed and keep the other pixels unchanged[10]. To do this, the first step is to identify impulsive noises. From our experience, an impulsive noise always seems to introduce a significant gray value difference to its neighbors. This experience can be expressed by the following fuzzy IF-THEN rule:

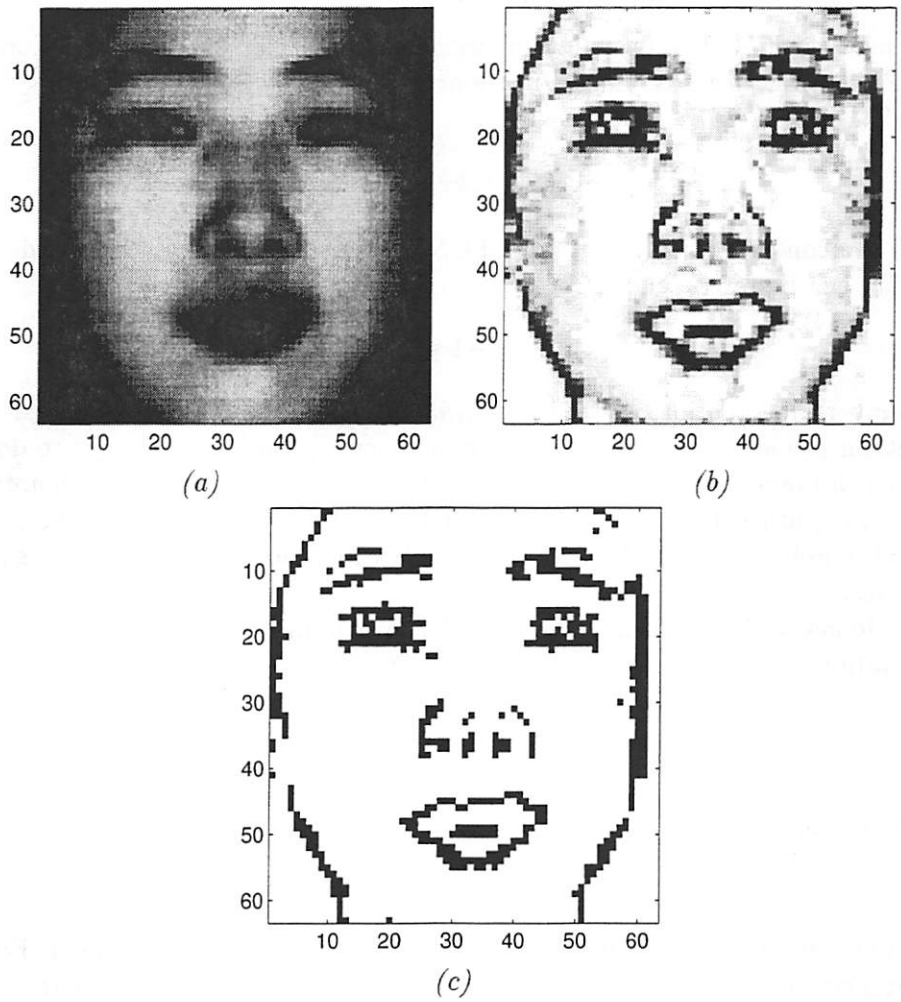


Figure 5.2: FCNN-based fuzzy inference for edge detection. (a) Input image of a Chinese girl. (b) Output of FCNN-based fuzzy inference. (c) Thresholding results of the image in (b).

IF $\mu_F(u_{i,j-1} - u_{ij})$ is *big* and $\mu_F(u_{i,j+1} - u_{ij})$ is *big*
and
$\mu_F(u_{i-1,j} - u_{ij})$ is *big* and $\mu_F(u_{i+1,j} - u_{ij})$ is *big*
and
$\mu_F(u_{i-1,j-1} - u_{ij})$ is *big* and $\mu_F(u_{i+1,j+1} - u_{ij})$ is *big*
and
$\mu_F(u_{i-1,j+1} - u_{ij})$ is *big* and $\mu_F(u_{i+1,j-1} - u_{ij})$ is *big*,
THEN u_{ij} is an impulsive noise.

Where the membership function $\mu_F(.)$ functions as a fuzzifier. In this paper, we choose $\mu_F(.)$ as the following piecewise linear function:

$$\mu_F(x) = \begin{cases} \min(1, -kx), & x \leq 0 \\ \min(1, kx), & x > 0 \end{cases} \tag{5.15}$$

where $k > 0$ is a constant. If the input of a FCNN is normalized within [0,1] and let $k = 1$, then Eq.(5.15) can be rewritten as:

$$\mu_F(x) = |x|, -1 \leq x \leq 1 \tag{5.16}$$

And the membership function of linguistic variable *big*, $\mu_{big}(.)$, can be obtained by using training method proposed in [10]. We use the membership function $\mu_{noise}(.)$ to denote the degree of truth of the sentence "there is an impulsive noise". So, $\mu_{noise}(.) = 1$ denotes "there is (exactly) an impulsive noise"; $\mu_{noise}(.) = 0$ denotes "there isn't an impulsive noise". To ease the VLSI complementation of FCNN, we usually choose $\mu_{big}(.)$ and $\mu_{noise}(.)$ as piecewise linear functions.

Then the following FCNN can be used to identify impulsive noises:

State equation

$$\dot{x}_{ij} = -x_{ij} + \tilde{\bigwedge_{C_{kl} \in N_1(i,j)/C_{ij}}} \mu_{big}(|u_{kl} - u_{ij}|) \tag{5.17}$$

Output equation

$$y_{ij} = \mu_{noise}(x_{ij}) \tag{5.18}$$

Generally, to fine the corresponding template of an FCNN structure as in Eq.(5.8) is very difficult and unnecessary. Fortunately, since the relation between synaptic weights and inputs in Eq.(5.17) is not very complicated, the FCNN in Eq.(5.17) has the following space-varying nonlinear B_{fmin} template:

$$B_{fmin}(ij) \odot u_{ij} = \begin{bmatrix} \mu_{big}(|u_{i-1,j-1} - u_{ij}|) & \mu_{big}(|u_{i-1,j} - u_{ij}|) & \mu_{big}(|u_{i-1,j+1} - u_{ij}|) \\ \mu_{big}(|u_{i,j-1} - u_{ij}|) & & \mu_{big}(|u_{i,j+1} - u_{ij}|) \\ \mu_{big}(|u_{i+1,j-1} - u_{ij}|) & \mu_{big}(|u_{i+1,j} - u_{ij}|) & \mu_{big}(|u_{i+1,j+1} - u_{ij}|) \end{bmatrix} \tag{5.19}$$

However, even in this simple case, the usage of template seems to complicate the expression.

It is easy to see that the above FCNN will identify impulsive noise in a parallel way over the whole image. Here, $\mu_{big}(.)$ and $\mu_{noise}(.)$ are usually difficult to be chosen and depend on

the statistical properties of impulsive noise. To overcome this problem, we can train these membership functions by a 3×3 FCNN. Some efficient learning algorithms for this kind of FCNN had been developed for learning $\mu_{big}(.)$ and $\mu_{noise}(.)$ from examples, these results will be presented in Chap.6.

The simulation results are shown in Fig.5.3. Fig.5.3(a) shows a real face image of size 63×63 and 256 gray levels. In Fig.5.3(b), some impulsive noises of mean value $\frac{125.6}{256}$ and deviation $\frac{39.8}{256}$ are added. This image is denoted by $\{u^{ij}\}$. Fig.5.3(c) shows the output result of a median filter. This image is denoted by $\{u_m^{ij}\}$. One can see that the fine structures of the image in Fig.5.3(a) is totally destroyed because Fig.5.3(a) has a low resolution and lots of details have characteristic width of 1 pixel. Fig. 5.3(d) shows the output of the FCNN in Eqs.(5.17) and (5.18). In this image, the gray value of every pixel denotes the degree of being an impulsive noise. A black pixel means that it is an impulsive noise (or, $\mu_{noise} = 1$). A white pixel means that it isn't an impulsive noise (or, $\mu_{noise} = 0$). A gray pixel, which occurs at boundaries or edges, means that it is suspected to be an impulsive noise. This image is denoted by $\{\mu_{noise}^{ij}\}$. In this simulation, we normalize the gray-level within interval [0,1], then $\mu_{big}(.)$ is given by:

$$\mu_{big} = x, 0 \leq x \leq 1 \tag{5.20}$$

and $\mu_{noise}(.)$ is given by:

$$\mu_{noise}(x) = \begin{cases} 0, & x \leq s \\ \frac{x-s}{p-s}, & s < x < p \\ 1, & x \geq p \end{cases} \tag{5.21}$$

where $p > s > 0$ are two constants, $p = \frac{40}{256}$, $s = \frac{20}{256}$.

Then we are ready to give the output result of our FCNN-based median filter, $\{u_{fm}^{ij}\}$. We have

$$u_{fm}^{ij} = (1 - \mu_{noise}^{ij})u^{ij} + \mu_{noise}^{ij} u_m^{ij} \tag{5.22}$$

The output result is shown in Fig. 5.3(e). One can see that this result is much better than that in Fig. 5.3(c).

The median filter may be implemented by some conventional CNN's. So far, there exist three kinds of CNN-based median filters. The first one[218][219] needs n cells to sort n samples. The second one[23] reduces the cell number to one and needs a neighborhood of odd number of cells. The third one[249][269], which is supposed to be an improved version of the second one, needs a neighborhood of even number of cells. In this section, some analysis of the second and the third CNN-based median filters are proposed. For simplicity, we only consider the CNN's with neighborhood size of 3×3 (i.e., $N_1(i,j)$), the analysis of the general case is similar.

The median filter given in[23] has the following state equation:

$$\dot{x}_{ij}(t) = \sum_{C_{kl} \in N_1(i,j)} sgn(u_{kl} - x_{ij}(t)) \tag{5.23}$$

where the function $sgn()$ should be defined by:

$$sgn(x) = \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1, & x > 0 \end{cases} \tag{5.24}$$

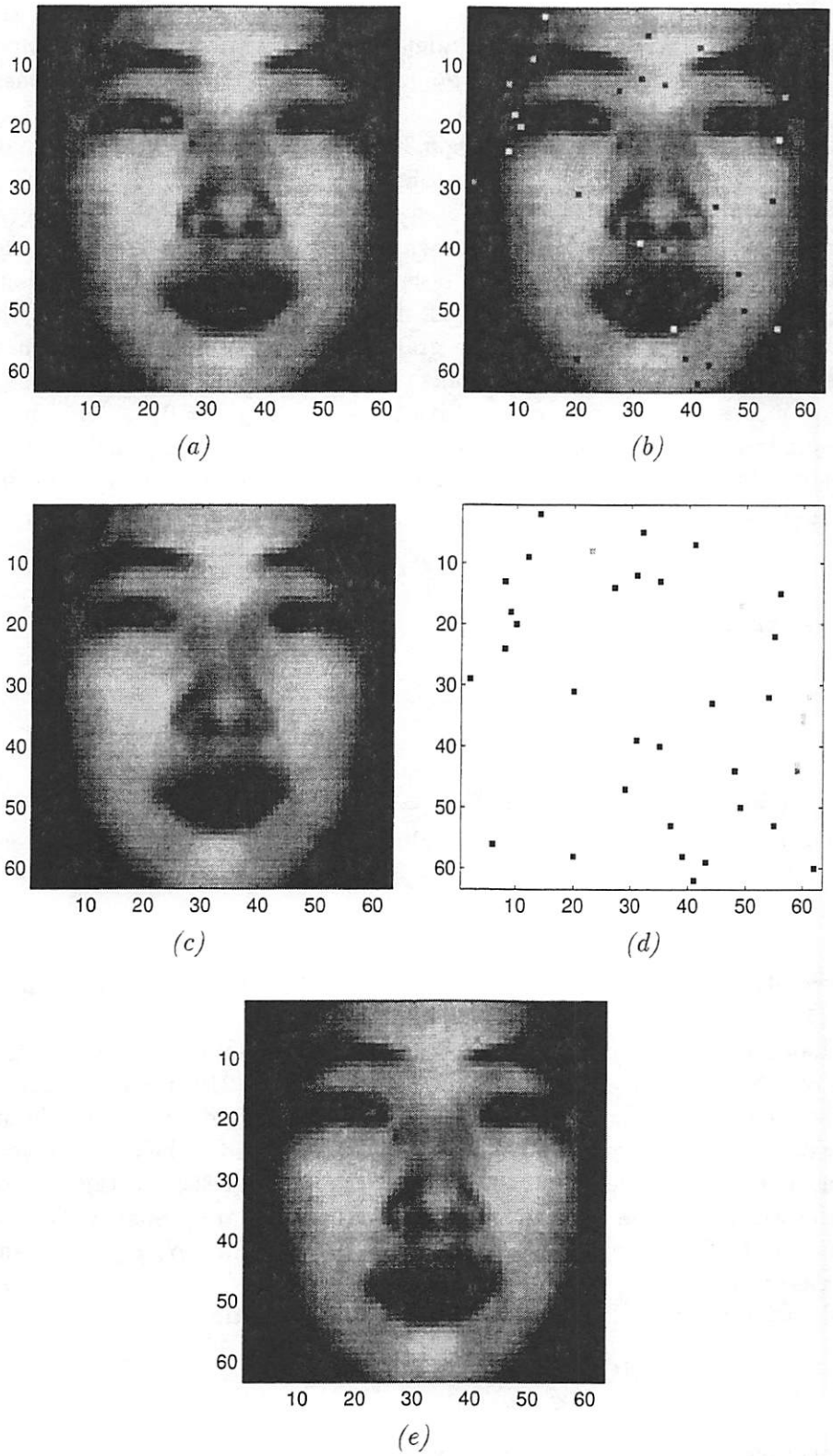


Figure 5.3: Using FCNN to remove impulsive noise. (a) The image of the face of a Chinese girl. (b) Impulsive noises are added. (c) Output of a median filter. (d) Output of FCNN shows the degree of being impulsive noise. (e) Output of FCNN-based median filter.

If the input set $\{u_{kl}|C_{kl} \in N_1(i,j)\}$ is sorted into a nondecreasing order as: $\{u_1, u_2, ..., u_5, ..., u_8, u_9\}$, then the median value should be u_5. Let n^+ denote the number of elements in the set $\{u_k|u_k = u_5, k > 5\}$ and n^- denote the number of elements in the set $\{u_k|u_k = u_5, k < 5\}$, then it is easy to see that if $n^+ = n^-$ then u_5 is the only equilibrium point of the cell in Eq.(5.23). In this case, we study the global stability of this equilibrium point. We have the following theorem:

Theorem 18

Given $n^+ = n^-$, then u_5 is asymptotically stable in the basin of attraction $(-\infty, \infty)$.

Proof: We construct the following Lyapunov function:

$$V_{ij}(t) = 1/2(x_{ij}(t) - u_5)^2 \geq 0 \tag{5.25}$$

$V_{ij}(t) > 0$ for any $x_{ij}(t) \neq u_5$. Differential $V_{ij}(t)$ along the solutions of Eq.(5.23), we have:

$$\begin{aligned}
\frac{dV_{ij}}{dt}|_{Eq.(5.23)} &= (x_{ij}(t) - u_5)\dot{x}_{ij}(t) \\
&= (x_{ij}(t) - u_5) \sum_{C_{kl} \in N_1(i,j)} sgn(u_{kl} - x_{ij}(t)) \\
&= (n^+ + n^- + 1)(x_{ij}(t) - u_5)sgn(u_5 - x_{ij}(t)) \\
&\quad + (x_{ij}(t) - u_5) \sum_{C_{kl} \in N_1(i,j), u_{kl} \neq u_5} sgn(u_{kl} - x_{ij}(t)) \\
&= -(2n^+ + 1)|x_{ij}(t) - u_5| + (x_{ij}(t) - u_5)\alpha \\
&\leq (x_{ij}(t) - u_5)\alpha
\end{aligned} \tag{5.26}$$

where

$$\alpha = \sum_{C_{kl} \in N_1(i,j), u_{kl} \neq u_5} sgn(u_{kl} - x_{ij}(t)) \tag{5.27}$$

It is easy to see that when $x_{ij} < u_5$, $\alpha > 0$ and when $x_{ij} > u_5$, $\alpha < 0$. So, we have: $(x_{ij}(t) - u_5)\alpha < 0$ for any $x_{ij} \neq u_5$. Then we have:

$$\frac{dV_{ij}}{dt}|_{Eq.(5.23)} \leq 0 \tag{5.28}$$

The equality is satisfied only when $x_{ij} = u_5$. So, the median value is asymptotically stable in the basin of attraction $(-\infty, \infty)$. □

However, if $n^+ \neq n^-$ then the cell in Eq.(5.23) has no equilibrium point. In this case, this median filter has no stable output in the common sense. In the simulations we found that the output of the cell will fluctuate around a certain value with a very small deviation as time becomes sufficiently large. To describe this fact, we need the following definition[348]:

Definition 14

Equilibrium point for the mean (m-equilibrium point): x_{ij}^* is said to be an m-equilibrium point of Eq.(5.23) if

$$\lim_{t \to \infty} E\{x_{ij}(t)\} = x_{ij}^* \tag{5.29}$$

and

$$\lim_{t \to \infty} E\{\dot{x}_{ij}(t)\} = 0 \tag{5.30}$$

Then we have the following theorem:

Theorem 19

Let $x_{ij} = u_5 - d^*(t)$ and $d^*(t) \in (-\delta, \delta)$, where $\delta = \min_{u_i \neq u_5} \delta_i$, $\delta_i = inf(|u_i - u_5|)$, $i \in \underline{9}$, $u_i \neq u_5$. And suppose

$$E\{sgn(d^*(t))\} = \frac{n^- - n^+}{n^+ + n^- + 1} \tag{5.31}$$

then u_5 is an m-equilibrium point of Eq.(5.23) when $n^+ \neq n^-$.

Proof:

$$\begin{aligned} E\{\dot{x}(t)\} &= \sum_{C_{kl} \in N_1(i,j)} E\{sgn(u_{kl} - x_{ij})\} \\ &= \sum_{C_{kl} \in N_1(i,j)} E\{sgn(u_{kl} - u_5 + d^*(t))\} \\ &= \sum_{C_{kl} \in N_1(i,j), u_{kl} \neq u_5} E\{sgn(u_{kl} - u_5 + d^*(t))\} \\ &\quad + \sum_{C_{kl} \in N_1(i,j), u_{kl} = u_5} E\{sgn(u_{kl} - u_5 + d^*(t))\} \\ &= (n^+ - n^-) + (n^+ + n^- + 1) E\{sgn(d^*(t))\} \\ &= 0 \end{aligned} \tag{5.32}$$

The fourth equality is in view of $d^*(t) \in (-\delta, \delta)$. □

Remark: We don't argue that u_5 is the only m-equilibrium point. In fact, there are infinite m-equilibrium points, x_{ij}^*'s, given different $E\{sgn(x_{ij}(t) - x_{ij}^*)\}$. This is possible if there exists noise in the circuit. Fig.5.4 shows the effect of noise on m-equilibrium points. In this simulation, we let $\{u_1, u_2, ..., u_5, ..., u_8, u_9\}$=$\{0.4, 0.5, 0.5, 0.5, 0.5, 0.6, 0.7, 0.8, 0.9\}$, i.e., $u_5 = 0.5$, $n^+ = 0, n^- = 3$. $v_n(t)$ denotes the additive noise in $x_{ij}(t)$. Fig.5.4(a) shows the cases when $v_n(t) = Kv$ is a DC bias. One can see that the m-equilibrium points are changed by different DC biases. For purpose of comparison, Fig.5.4(b) shows the cases when $v_n(t)$ uniformly distributes between $-Kv \sim Kv$. The change of m-equilibrium point is similar to that in Fig.5.4(a). Since these results are initial state independent, we find the CNN-based median filter in Eq.(5.23) has a promising robustness. Given a low level of additive noise, the m-equilibrium point of Eq.(5.23) is very closed to the median value when $n^+ \neq n^-$. In most cases, this CNN-based median filter can output satisfied result. When this median filter is used to process a 256 gray-scale image, it can always output the correct median value because the offset of an m-equilibrium point from a real median value is 3 times less than the value corresponding to 1 bit (We normalized 256 gray-scale in [-1, 1]).

The authors of [249] proposed a CNN-based median filter as follow:

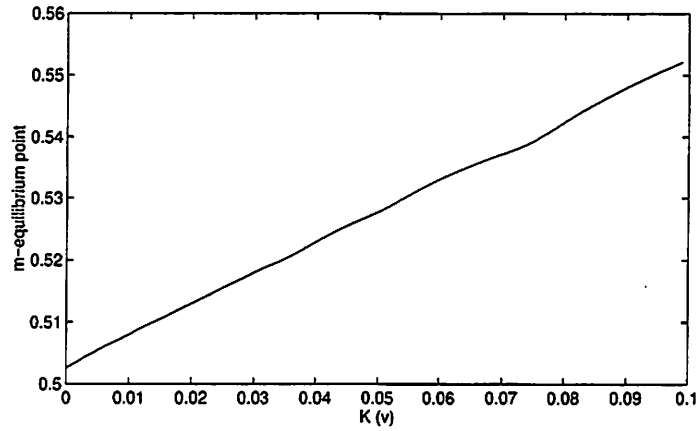
State equation:

$$\dot{x}_{ij}(t) = -x_{ij} + f(x_{ij}) + \sum_{C_{kl} \in N_1(i,j)/C_{ij}} sgn(x_{ij}(t) - u_{kl}) \tag{5.33}$$

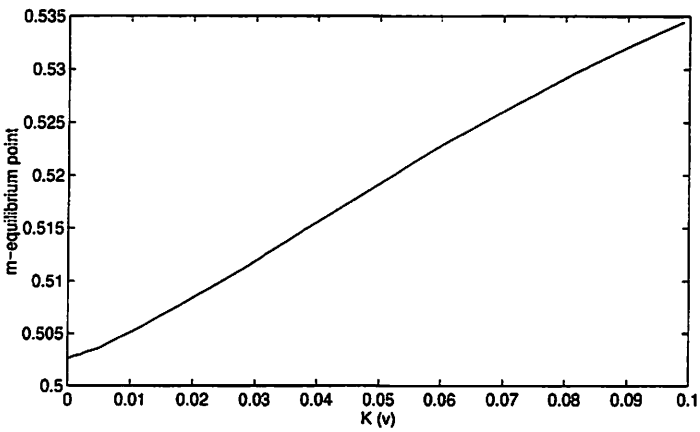
Output equation:[1]

$$f(x_{ij}(t)) = 1/2(|x_{ij}(t) + 1| - |x_{ij}(t) - 1|) \tag{5.34}$$

[1]The authors of [249] and [269] didn't give the explicit expression of the output function but described it as([249], p684) "a sigmoid-type piecewise linear function".



(a)



(b)

Figure 5.4: (a) m-equilibrium points with different DC biases in $x_{ij}(t)$. (b) m-equilibrium point with different uniformly distributed noise in $x_{ij}(t)$.

Then the authors of [249] argued that "General rank order filters can be implemented simply changing the bias value of the template(e.g., MIN filter: I=8; MAX filter: I=-8)"([249], p684).

However, [390] gives the following description of a median filter: "To computer the output of a median filter, an odd number of sample values are sorted, and the middle or median value is used as the filter output." Since the CNN-based median filter in Eq.(5.33) uses only eight sample values in $N_1(i,j)$, it is not a median filter and its result is very sensitive to initial conditions. For example, suppose this eight values are sorted into a nondecreasing order as $\{u_1, ..., u_4, u_5, ..., u_8\}$ and $u_4 \neq u_5$. If $x_{ij}(0) < u_4$, then $x_{ij}(\infty) = u_4 + \delta, \delta \to 0^+$. If $x_{ij}(0) > u_5$, then $x_{ij}(\infty) = u_5 + \delta, \delta \to 0^-$. If $x_{ij}(0)$ is a random number such that $u_4 < x_{ij}(0) < u_5$, then $x_{ij}(\infty) = x_{ij}(0)$ is also a random result. When $(u_5 - u_4)$ is big, (e.g., there exists an edge in $N_1(i,j)$), this CNN-based median filter seems to output a random result when noise exists in $x_{ij}(0)$. So, this CNN-based median filter is much worst than that in Eq.(5.23). Also, it can't be a rank-order filter with different biases.

To show this kind of randomness in a real image processing problem, we use the CNN-based median filter in Eq.(5.33) to process the real gray-scale image as in Fig.4.11(a). Fig.5.5(a) shows the difference between the median filtering result and the output of CNN in Eq.(5.33) with initial condition $x_{ij}(0) = -1$. Fig.5.5(b) shows the difference between the median filtering result and the output of CNN in Eq.(5.33) with initial condition $x_{ij}(0) = 1$. Fig.5.5(c) shows the difference between the median filtering result and the output of CNN in Eq.(5.33) with initial condition $x_{ij}(0) = u_{ij}$.

5.3.3 Other applications

As indicated in [278], the FIRE smoother, the FIRE sharpener, the FIRE high-pass filter and the FIRE image enhancement can also be easily implemented by FCNN using the same methods presented in this chapter.

For example, we use an FDTCNN to implement the the FIRE sharpener. The rule is shown in Fig.9.1 which is applied to a 256-greylevel digital image. It should be noted that all the inputs in the rules are gray-value difference between each pixel at the neighborhood system and the center pixel. This is the so called "relative in the antecedents" approach[278].

To implement this fuzzy sharpener, we let E_{ij} denote the grey-value of pixel (i,j), then we have the following multi-layer FDTCNN structure:

The State equation of FDTCNN #1 which is used to implement the 1st rule in Fig.9.1 is given by:

$$x_{ij}^1(k) = \min_{C_{kl} \in N_1(ij)/C_{ij}} \mu_P(E_{kl} - E_{ij}) \tag{5.35}$$

where $\mu_P(.)$ is the membership function of fuzzy set P as shown in Fig.9.1.

The State equation of FDTCNN #2 which is used to implement the 2nd rule in Fig.9.1 is given by:

$$x_{ij}^2(k) = \min_{C_{kl} \in N_1(ij)/C_{ij}} \mu_N(E_{kl} - E_{ij}) \tag{5.36}$$

where $\mu_N(.)$ is the membership function of fuzzy set N as shown in Fig.9.1.

The State equation of FDTCNN #0 which is used to implement the ELSE rule in Fig.9.1 is given by:

$$x_{ij}^0(k) = \min(1 - x_{ij}^1(k-1), 1 - x_{ij}^2(k-1)) \tag{5.37}$$

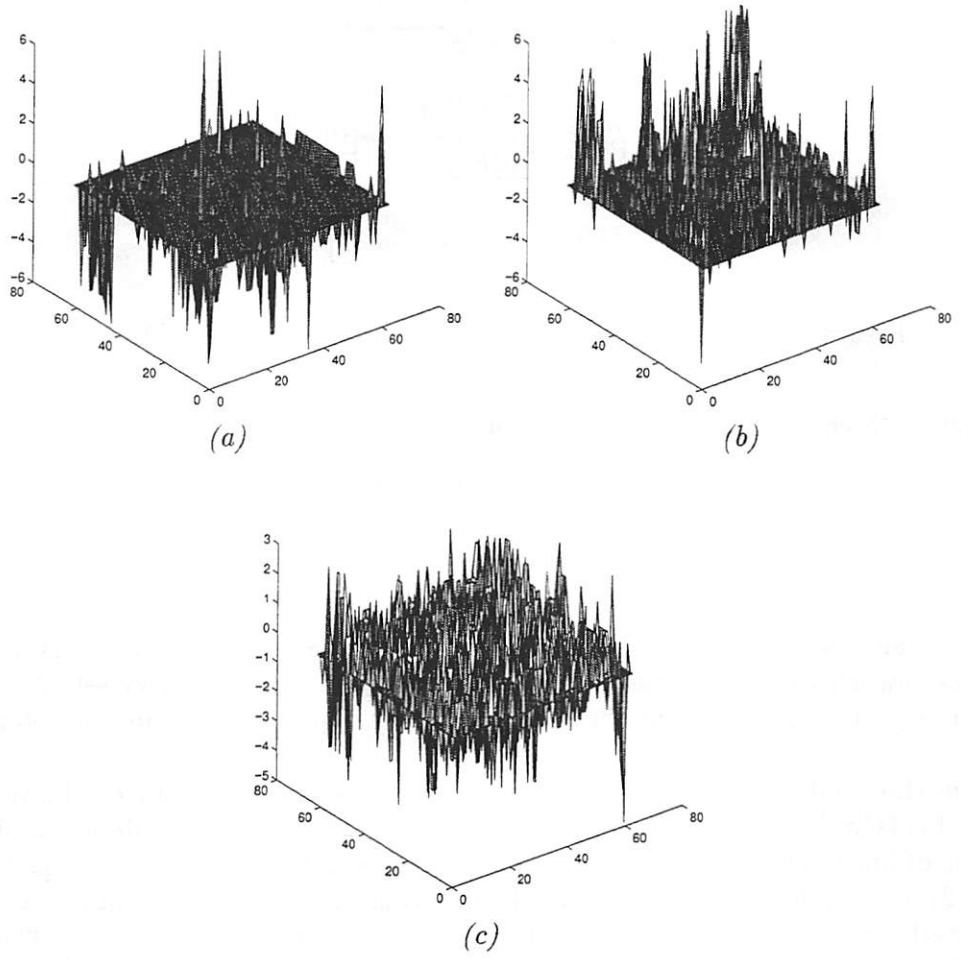


Figure 5.5: (a) The difference of gray value between a median filter and the CNN in Eq.(5.33) with initial condition $x_{ij}(0) = -1$. (b) The difference of gray value between a median filter and the CNN in Eq.(5.33) with initial condition $x_{ij}(0) = 1$. (c) The difference of gray value between a median filter and the CNN in Eq.(5.33) with initial condition $x_{ij}(0) = u_{ij}$.

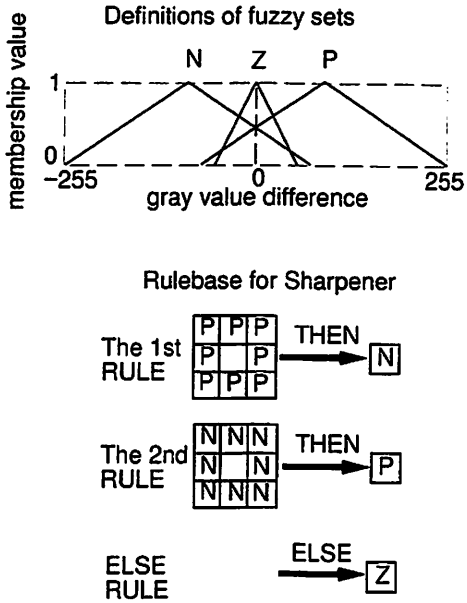


Figure 5.6: The rule-base for fuzzy sharpener presented in[278].

The above three layers share a single output layer as:

$$\begin{aligned} y_{ij}(k) &= f(x_{ij}^0(k-1), x_{ij}^1(k-1), x_{ij}^2(k-1)) \\ &= \frac{\sum_{I=0}^{2} x_{ij}^I(k-1) c_I w_I}{\sum_{I=0}^{2} x_{ij}^I(k-1) w_I} \end{aligned} \tag{5.38}$$

where c_0, c_1 and c_2 represent the centers of triangular-shaped fuzzy sets Z, N and P, respectively. w_0, w_1 and w_2 represent the widths of triangular shaped fuzzy sets Z, N and P, respectively. One can see that the output result in Eq.(5.38) is a kind of defuzzified result.

Since in this section we only want to show how can we map the nonlinear fuzzy operators into FDTCNN, we do not want to discuss the advantages and the disadvantages of these kinds of image processing techniques, the interested reader are referred to [276, 277, 275, 279, 278] and references therein. All the disadvantages and the advantages of these kinds of methods are due to themselves but not FDTCNN(or general FCNN). The only thing FDTCNN can do here is to provide a computational platform to offset the possible computational complexity.

Chapter 6

Learning algorithms of FCNN

In this chapter we present the results which distinguish a FCNN from a computational array. It means that the FCNN can learn it weights from examples or from the existed knowledge and the experience of human experts.

There had been existed lots of references on learning algorithms of different CNN structures[122, 32, 198, 33, 318, 104, 127, 197, 2, 327, 186, 185, 108, 290, 191, 18, 116, 291, 106, 183, 315, 319, 247, 328, 3, 256, 170, 326, 312, 406, 123, 329, 407, 225]. Learning is one of the promising properties of CNN which distinguishes a CNN structure from a parallel computational array. On the other hand, the FNN literatures also provide us with lots of special learning algorithm concerning the high nonlinearity of FNN[344, 146, 26]. Nourished by these two fields, the learning algorithms of FCNN were also developed. One difference between the learning algorithms for FCNN and those for conventional CNN is that the learning algorithm of FCNN may have the linguistic variables as its examples(input-output pairs). In some cases, when the experience of a human expert is easy to be obtained and the measuring data is hard to be obtained, this learning ability may be very useful.

In Chap. 4, we have shown that additive FCNN is a universal framework of implementing different kinds of mathematical morphology operators. Although mathematical morphology is very useful in signal processing as shown in Chap. 4, one key problem is the choice of structuring elements for different tasks. Normally, the structuring elements are chosen by trial-and-error method. Recently, some morphological (neural) networks with learning ability was presented[63, 64, 11, 65]. But the structure of a morphology(neural) network is too complicated to be implemented by using the state-of-the-art VLSI techniques. On the other hand, we find that when FCNN are used as computational arrays, some of them(see examples in Sec. 4.2) are in fact morphological networks. So, we can train FCNN with examples and find the structuring elements from the results.

Since FCNN is a combination of two mature fields: fuzzy set theory and CNN, lots of regions are waiting for exploring. At the very beginning when we try to set up the framework of this brand new field, there exist two basic motivations. One comes from mathematical morphology[292, 293, 129] which is a very elegant framework for signal processing from the geometrical point of view. We found both FCNN and mathematical morphology operators share two elementary features: local connectedness and max/min operations. The other motivation comes from the necessity of development of an interface between the human experts (users) and the low level conventional CNN structures. In [351, 352, 355, 375, 372, 368, 365, 363, 378], we presented the results which were prompted by the first motivation. In [354, 353, 355, 364, 376, 384], we presented results which were prompted by the second

motivation.

An FCNN structure can be used as either a computational array or a learning array. In Chaps. 4 and 5, the FCNN's were used as computational arrays. However, the learning ability of FCNN is also a very important aspect because only when an FCNN can learn its parameters from both real number examples and linguistic statements, it can actually perform as an "intelligent" interface between human experts and the low-level CNN's(e.g., the conventional CNN). This chapter is the collection of our following papers:[362, 387, 370].

6.1 Learning structuring elements

In this section, we presented some learning algorithms for additive FCNN's. The learning algorithms are based on the FDTCNN structure. Although FDTCNN can be viewed as a corresponding concept of DTCNN[120], it is not necessary to obey the tradition of standard DTCNN in which the output should be binary. Then these learning algorithms are used to learn structuring elements from examples. In this view, these FDTCNN structures are mathematical morphology networks with learning ability.

A general framework of type-II FDTCNN is given by:

$$\begin{aligned} x_{ij}(k+1) &= F^A_{C_{kl}\in N_r(i,j)}(A_f(i,j;k,l), y_{kl}(k)) \\ &\quad + F^B_{C_{kl}\in N_r(i,j)}(B_f(i,j;k,l), u_{kl}) \end{aligned} \tag{6.1}$$

where $F^A(\cdot)$ and $F^B(\cdot)$ are two local fuzzy operations defined in $N_r(i,j)$. $A_f(i,j;k,l)$ and $B_f(i,j;k,l)$ are fuzzy feedback synaptic weight and fuzzy feed-forward synaptic weight, respectively.

In this section, we study the learning algorithm of the following type-II FDTCNN:

$$x_{ij}(k+1) = F^B_{C_{kl}\in N_r(i,j)}(B_f(i,j;k,l), u_{kl}) \tag{6.2}$$

This FDTCNN is a kind of *uncoupled* DTCNN. It maps the input to the output by a single iteration. This computational structure is very useful in implementation of mathematical morphology operators[129].

6.1.1 Learning algorithm of additive type-II FDTCNN

In [351, 352] we have shown that the following FDTCNN is very useful to implement gray-scale mathematical morphology transformations:

State equation:

$$\begin{aligned} x_{ij}(k+1) &= \tilde{\bigwedge_{C_{kl}\in N_r(i,j)}} (B_{fmin}(i,j;k,l) + u_{kl}) \\ &\quad + \tilde{\bigvee_{C_{kl}\in N_r(i,j)}} (B_{fmax}(i,j;k,l) + u_{kl}) \\ &\quad , 1 \le i \le M, 1 \le j \le N \end{aligned} \tag{6.3}$$

Since the operations between fuzzy feed-forward synaptic weights and the inputs are additions, the above FDTCNN is called *additive FDTCNN*. The output equation is given by:

$$\begin{aligned} y_{ij}(k) &= f(x_{ij}(k)) = \frac{1}{2}(|x_{ij}(k)+1| - |x_{ij}(k)-1|), \\ &\quad 1 \le i \le M, 1 \le j \le N \end{aligned} \tag{6.4}$$

The parameters of DTCNN in Eq.(6.3) for implementation of erosion are given by:

$$B_{fmax} = \text{undefined}, B_{fmin} = -S \tag{6.5}$$

where S is the structuring element.

And the parameters of DTCNN in Eq.(6.3) for implementation of dilation are given by:

$$B_{fmax} = S_D, B_{fmin} = \text{undefined} \tag{6.6}$$

where S_D is given by:

$$S_D = \{-x : x \in S\} \tag{6.7}$$

In this section, we will study how can an additive FDTCNN learn the structuring element when only a set of examples $\{(u_{ij}, O_{ij})\}$ is available. $\{u_{ij}\}$ is input set and $\{O_{ij}\}$ is output set. Since the structuring element are embedded in the feed-forward templates of the FDTCNN, the objective of training the network is to adjust the weights so that a set of inputs produces the desired set of outputs. This is driven by minimizing the square of the difference between the desired output $\{O_{ij}\}$ and actual output $\{y_{ij}\}$, for all the samples to be learned:

$$E = \frac{1}{2}\sum_{ij}(O_{ij} - y_{ij})^2 \tag{6.8}$$

It is well known that:

$$\frac{\partial E}{\partial B_{fmin}(i,j;p,q)} = \frac{\partial E}{\partial y_{ij}}\frac{\partial y_{ij}}{\partial x_{ij}}\frac{\partial x_{ij}}{\partial B_{fmin}(i,j;p,q)} \tag{6.9}$$

$$\frac{\partial E}{\partial B_{fmax}(i,j;p,q)} = \frac{\partial E}{\partial y_{ij}}\frac{\partial y_{ij}}{\partial x_{ij}}\frac{\partial x_{ij}}{\partial B_{fmax}(i,j;p,q)} \tag{6.10}$$

Let us expand the first terms in the right hand side of Eqs.(6.9) and (6.10) as:

$$\frac{\partial E}{\partial y_{ij}} = -(O_{ij} - y_{ij}) \tag{6.11}$$

and expand the second terms in the right hand side of Eqs.(6.9) and (6.10) as:

$$\frac{\partial y_{ij}}{\partial x_{ij}} = \theta_{ij} = \begin{cases} 1, & |x_{ij}| \le 1 \\ 0, & |x_{ij}| > 1 \end{cases} \tag{6.12}$$

Then let us expand the third term in the right hand side of Eq.(6.9) as:

$$\begin{aligned} & \frac{\partial x_{ij}}{\partial B_{fmin}(i,j;p,q)} \\ = & \frac{\partial \tilde{\bigwedge}_{C_{kl}\in N_r(i,j)}(B_{fmin}(i,j;k,l) + u_{kl})}{\partial B_{fmin}(i,j;p,q)} \\ = & \frac{\partial \min(B_{fmin}(i,j;p,q) + u_{pq}, \tilde{\bigwedge}_{C_{kl}\in N_r(i,j),(k,l)\neq(p,q)}(B_{fmin}(i,j;k,l) + u_{kl}))}{\partial B_{fmin}(i,j;p,q)} \\ = & \frac{\partial \min(B_{fmin}(i,j;p,q) + u_{pq}, \tilde{\bigwedge}_{C_{kl}\in N_r(i,j),(k,l)\neq(p,q)}(B_{fmin}(i,j;k,l) + u_{kl}))}{\partial (B_{fmin}(i,j;p,q) + u_{pq})} \\ = & \frac{\partial \min(y,\chi)}{\partial y} \end{aligned} \tag{6.13}$$

where

$$y \triangleq B_{fmin}(i,j;p,q) + u_{pq} \tag{6.14}$$

and

$$\chi \triangleq \tilde{\bigwedge_{C_{kl} \in N_r(i,j),(k,l)\neq(p,q)}} (B_{fmin}(i,j;k,l) + u_{kl}) \tag{6.15}$$

Then we consider the so called "smooth derivative"[26] of $min(y,\chi)$. In the classical sense, $max(y,\chi)$ is derivable into the open intervals $y < \chi$ and $y > \chi$ but the derivative is not defined at $y = \chi$, i.e.,

$$\frac{\partial \min(y,\chi)}{\partial y} = \begin{cases} 0, & if\ y > \chi \\ 1, & if\ y < \chi \end{cases} \tag{6.16}$$

From Eqs.(6.9) and (6.16) we know that the FDTCNN will stop learning when $y > \chi$. This makes the learning process very slow. In the worst case, this can even make the learning process impossible. To overcome this problem, we notice that Eq.(6.16) only gives the crisp truth value of statement: "y is less than χ". In this view, we can fuzzify Eq.(6.16) using different schemes to make the FDTCNN learning in a fuzzified way. One example can be found in [26]. But the method used in [26] can not be used here, we fuzzifier our "smooth derivative" as:

$$\frac{\partial \min(y,\chi)}{\partial y} = \begin{cases} 1, & y \leq \chi \\ \frac{\chi+1}{y+1}, & y > \chi \end{cases} \tag{6.17}$$

where $y \in [-1,1], \chi \in [-1,1]$.

Similarly, the third term in the right hand side of Eq.(6.10) can be expanded as:

$$\begin{aligned} & \frac{\partial x_{ij}}{\partial B_{fmax}(i,j;p,q)} \\ = & \frac{\partial \max(B_{fmax}(i,j;p,q) + u_{pq}, \tilde{\bigvee}_{C_{kl} \in N_r(i,j),(k,l)\neq(p,q)} (B_{fmax}(i,j;k,l) + u_{kl}))}{\partial (B_{fmax}(i,j;p,q) + u_{pq})} \\ = & \frac{\partial \max(z,\psi)}{\partial z} \end{aligned} \tag{6.18}$$

where

$$z \triangleq B_{fmax}(i,j;p,q) + u_{pq} \tag{6.19}$$

and

$$\psi \triangleq \tilde{\bigvee_{C_{kl} \in N_r(i,j),(k,l)\neq(p,q)}} (B_{fmax}(i,j;k,l) + u_{kl}) \tag{6.20}$$

Similarly, we use the following " smooth derivative" to guarantee the learning process of FDTCNN in Eq.(6.10):

$$\frac{\partial \max(z,\psi)}{\partial z} = \begin{cases} 1, & z \geq \psi \\ \frac{z+1}{\psi+1}, & z < \psi \end{cases} \tag{6.21}$$

We denote $-\frac{\partial E}{\partial y_{ij}}$ by δ_{ij}, therefore:

$$-\frac{\partial E}{\partial w(i,j;k,l)} = \delta_{ij}\theta_{ij}\frac{\partial x_{ij}}{\partial w(i,j;k,l)} \tag{6.22}$$

where $w(i,j;k,l)$ denotes $B_{fmin}(i,j;k,l)$ or $B_{fmax}(i,j;k,l)$. Finally, the changes for the weights will be obtained from a δ-rule with expression:

$$\Delta w(i,j;k,l) = \mu\delta_{ij}\theta_{ij}\frac{\partial x_{ij}}{\partial w(i,j;k,l)} \tag{6.23}$$

where $w(i,j;k,l)$ as that in the Eq.(6.37). μ is a positive constant.

6.1.2 Examples

In this section, the learning algorithms of FDTCNN are used to learning structuring elements from examples. We use two examples to show the usefulness of the learning algorithms to structuring element learning. Let the structuring element S be:

$$S = \begin{bmatrix} 0.11 & 0.15 & 0.13 \\ 0.16 & 0.19 & 0.18 \\ 0.12 & 0.17 & 0.14 \end{bmatrix} \tag{6.24}$$

then we have

$$B_{fmax} = S_D = \begin{bmatrix} 0.14 & 0.17 & 0.12 \\ 0.18 & 0.19 & 0.16 \\ 0.13 & 0.15 & 0.11 \end{bmatrix} \tag{6.25}$$

and

$$B_{fmin} = -S = \begin{bmatrix} -0.11 & -0.15 & -0.13 \\ -0.16 & -0.19 & -0.18 \\ -0.12 & -0.17 & -0.14 \end{bmatrix} \tag{6.26}$$

Then we use the dilation operator to generate 2000 samples $\{(u_{ij}, O_{ij})\}$ as the training data to train a dilation FDTCNN. The learning process of the B_{fmax} template is shown in Fig. 6.1. One can see that the elements of B_{fmax} approached to correct values(see Eq.(6.25)). The initial conditions for B_{fmax} template is 0. $\mu = 1$. Since the B_{fmax} template is of size 3×3, we only need a 3×3 FDTCNN to learn the structuring element.

Next we use the erosion operator to generate 2000 samples $\{(u_{ij}, O_{ij})\}$ as the training data to train an erosion FDTCNN. The learning process of the B_{fmin} template is shown in Fig. 6.2 . One can see that the elements of B_{fmin} approached to correct values(see Eq.(6.26)). The initial conditions for B_{fmin} template is 0. $\mu = 1$. The above examples show that our FDTCNN learning algorithm works well.

6.2 Advanced learning algorithms of additive discrete-time FCNN

The breakpoints of min and max operators pose a big problem on their derivatives. In practice there exist two methods to overcome this problem. The first one is to use bounded-addition and multiplication to replace the min and max operators. Although this method

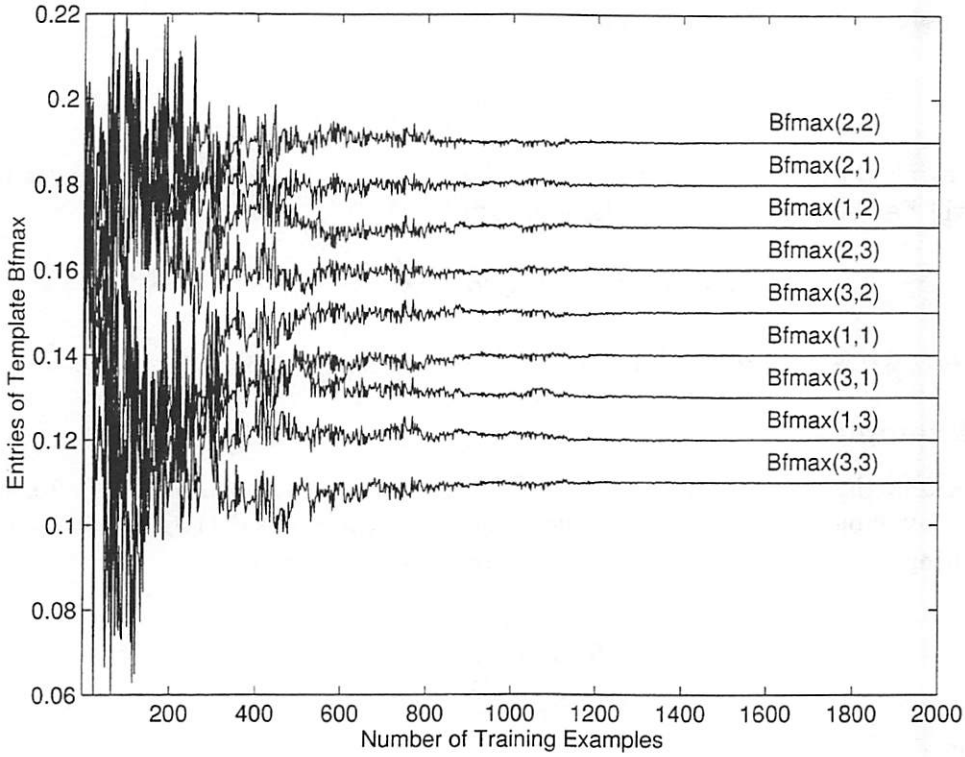


Figure 6.1: Learning process of dilation FDTCNN.

bypasses the problem of derivatives, the trained network may be functionally very different from the original one. The second one is to develop a rigorous and systematic theory for the differentiation of min and max functions by means of step function[141], functional analysis[398] and some special functions[397].

For purpose of deriving the Δ-learning law for FDTCNN, we have to cope with the partial differentiation of E with respect to $B_{fmin}(i,j;k,l)$ and $B_{fmax}(i,j;k,l)$, such a differentiation can not be given in a conventional sense.

By Theorem 8 of [397] we know that the following two expressions are satisfied almost everywhere in real field $\mathbf{R}$.

$$\frac{\tilde{\partial} E}{\tilde{\partial} B_{fmin}(i,j;p,q)} = \frac{\partial E}{\partial y_{ij}} \frac{\partial y_{ij}}{\partial x_{ij}} \frac{\tilde{\partial} x_{ij}}{\tilde{\partial} B_{fmin}(i,j;p,q)} \tag{6.27}$$

$$\frac{\tilde{\partial} E}{\tilde{\partial} B_{fmax}(i,j;p,q)} = \frac{\partial E}{\partial y_{ij}} \frac{\partial y_{ij}}{\partial x_{ij}} \frac{\tilde{\partial} x_{ij}}{\tilde{\partial} B_{fmax}(i,j;p,q)} \tag{6.28}$$

where ∂ denotes a partial derivative in a conventional sense and $\tilde{\partial}$ denotes the partial derivative presented in [397].

Let us expand the first term in the right hand sides of Eqs.(6.27) and (6.28) as

$$\frac{\partial E}{\partial y_{ij}} = -(O_{ij} - y_{ij}) \tag{6.29}$$

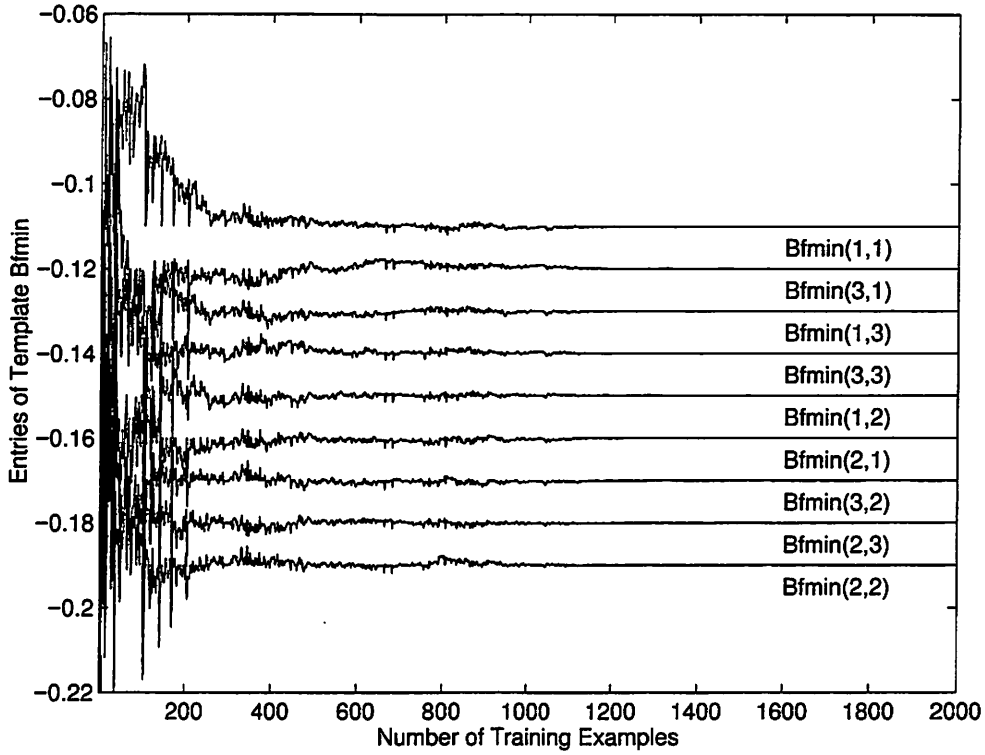


Figure 6.2: Learning process of erosion FDTCNN.

and expand the second terms in the right hand sides of Eqs.(6.27) and (6.28) as

$$\frac{\partial y_{ij}}{\partial x_{ij}} = \theta_{ij} = \begin{cases} 1, & |x_{ij}| \leq 1 \\ 0, & |x_{ij}| > 1 \end{cases} \tag{6.30}$$

Then let us expand the third term in the right hand side of Eq.(6.27) as

$$\begin{aligned} & \frac{\tilde{\partial} x_{ij}}{\tilde{\partial} B_{fmin}(i,j;p,q)} \\ = & \frac{\tilde{\partial} \tilde{\bigwedge}_{C_{kl} \in N_r(i,j)} (B_{fmin}(i,j;k,l) + u_{kl})}{\tilde{\partial} B_{fmin}(i,j;p,q)} \\ = & \frac{\tilde{\partial} \min(B_{fmin}(i,j;p,q) + u_{pq}, \tilde{\bigwedge}_{C_{kl} \in N_r(i,j),(k,l) \neq (p,q)} (B_{fmin}(i,j;k,l) + u_{kl}))}{\tilde{\partial} B_{fmin}(i,j;p,q)} \\ = & \frac{\tilde{\partial} \min(B_{fmin}(i,j;p,q) + u_{pq}, \tilde{\bigwedge}_{C_{kl} \in N_r(i,j),(k,l) \neq (p,q)} (B_{fmin}(i,j;k,l) + u_{kl}))}{\tilde{\partial} (B_{fmin}(i,j;p,q) + u_{pq})} \\ = & \frac{\tilde{\partial} \min(y, \chi)}{\tilde{\partial} y} \end{aligned} \tag{6.31}$$

where $y \triangleq B_{fmin}(i,j;p,q) + u_{pq}$ and $\chi \triangleq \tilde{\bigwedge}_{C_{kl} \in N_r(i,j),(k,l) \neq (p,q)} (B_{fmin}(i,j;k,l) + u_{kl})$.

Since $min(\cdot,\cdot)$ and $max(\cdot,\cdot)$ are not differentiable functions in the conventional sense, we need to show that under certain conditions all min-max functions are continuously differentiable almost everywhere in the real number field **R**. Fortunately, a rigorous theory

on this problem had been presented in [397]. To make this paper self-contained, we need the following definition and lemma:

Definition 15

(Definition 1, p.1141, [397]) A function $f_{lor} : \mathbf{R} \mapsto \mathbf{R}$ on the real number field $\mathbf{R}$ is defined as

$$f_{lor}(x) = \begin{cases} 1, & x > 0 \\ \frac{1}{2}, & x = 0 \\ 0, & x < 0 \end{cases} \tag{6.32}$$

Proposition 10

(Corollary 1, p.1143, [397]) Suppose a is a real number and $f(x)$, $h_1(x) = a\tilde{\vee}f(x)$, and $h_2(x) = a\tilde{\wedge}f(x)$ are real variable functions. If they are all differentiable at point x, then

$$\frac{\tilde{d}h_1(x)}{\tilde{d}x} = \frac{\tilde{d}[a\tilde{\vee}f(x)]}{\tilde{d}x} = f_{lor}[f(x) - a]\frac{df(x)}{dx} \tag{6.33}$$

$$\frac{\tilde{d}h_2(x)}{\tilde{d}x} = \frac{\tilde{d}[a\tilde{\wedge}f(x)]}{\tilde{d}x} = f_{lor}[a - f(x)]\frac{df(x)}{dx} \tag{6.34}$$

It follows from Proposition 10 that

$$\frac{\tilde{\partial}\min(y, \chi)}{\tilde{\partial}y} = f_{lor}(\chi - y) \tag{6.35}$$

Similarly, the third term in the right hand side of Eq.(6.28) can be expanded as

$$\begin{aligned} & \frac{\tilde{\partial}x_{ij}}{\tilde{\partial}B_{fmax}(i,j;p,q)} \\ = & \frac{\tilde{\partial}\max(B_{fmax}(i,j;p,q) + u_{pq}, \tilde{\bigvee}_{C_{kl}\in N_r(i,j),(k,l)\neq(p,q)}(B_{fmax}(i,j;k,l) + u_{kl}))}{\tilde{\partial}(B_{fmax}(i,j;p,q) + u_{pq})} \\ = & \frac{\tilde{\partial}\max(z,\psi)}{\tilde{\partial}z} = f_{lor}(z - \psi) \end{aligned} \tag{6.36}$$

where $z \triangleq B_{fmax}(i,j;p,q) + u_{pq}$ and $\psi \triangleq \tilde{\bigvee}_{C_{kl}\in N_r(i,j),(k,l)\neq(p,q)}(B_{fmax}(i,j;k,l) + u_{kl})$.

We denote $-\frac{\partial E}{\partial y_{ij}}$ by δ_{ij}, therefore

$$-\frac{\tilde{\partial}E}{\tilde{\partial}w(i,j;k,l)} = \delta_{ij}\theta_{ij}\frac{\tilde{\partial}x_{ij}}{\tilde{\partial}w(i,j;k,l)} \tag{6.37}$$

where $w(i,j;k,l)$ denotes $B_{fmin}(i,j;k,l)$ or $B_{fmax}(i,j;k,l)$. Finally, the changes of weights will be obtained from a δ-rule with expression

$$\Delta w(i,j;k,l) = \mu\delta_{ij}\theta_{ij}\frac{\tilde{\partial}x_{ij}}{\tilde{\partial}w(i,j;k,l)} \tag{6.38}$$

where $w(i,j;k,l)$ as that in Eq.(6.37). μ is a positive constant.

The following two theorems guarantee that the learning algorithm in Eq.(6.38) make sense almost everywhere in $\mathbf{R}$ and the learning results will be a local minimum of the cost function E.

Theorem 20

For the erosion FDTCNN in Eq.(6.5) and the dilation FDTCNN in Eq.(6.6), and the cost function in Eq.(6.8), the partial differentials in Eqs. (6.31) and (6.36) exist almost everywhere in $\mathbf{R}$.

Proof: Since x_{ij}'s in both erosion FDTCNN and dilation FDTCNN are $(\tilde{\vee}, \tilde{\wedge})$-functions, i.e., functions containing $\tilde{\vee}$ and/or $\tilde{\wedge}$ operations, it follows from Corollary 4 of [397] that the partial differentials in Eqs. (6.31) and (6.36) exist almost everywhere in $\mathbf{R}$. □

Theorem 21

The δ-rule given in Eq.(6.38) guarantees the erosion FDTCNN in Eq.(6.5) and the dilation FDTCNN in Eq.(6.6) to converge to a local minimum of E in Eq.(6.8) with Probability 1 with increasing iteration index.

Proof: Similar to the proof of Theorem 10 of [397], let us prove the theorem in two steps. First, using the similar process in the proof of Theorem 10 of [397], we immediately know that E in Eq.(6.8) is differentiable with respect to discrete time with Probability 1.

Then, as the second part of the proof, we show that E always decreasing whenever it is differentiable. Suppose E is differentiable at time t, then,

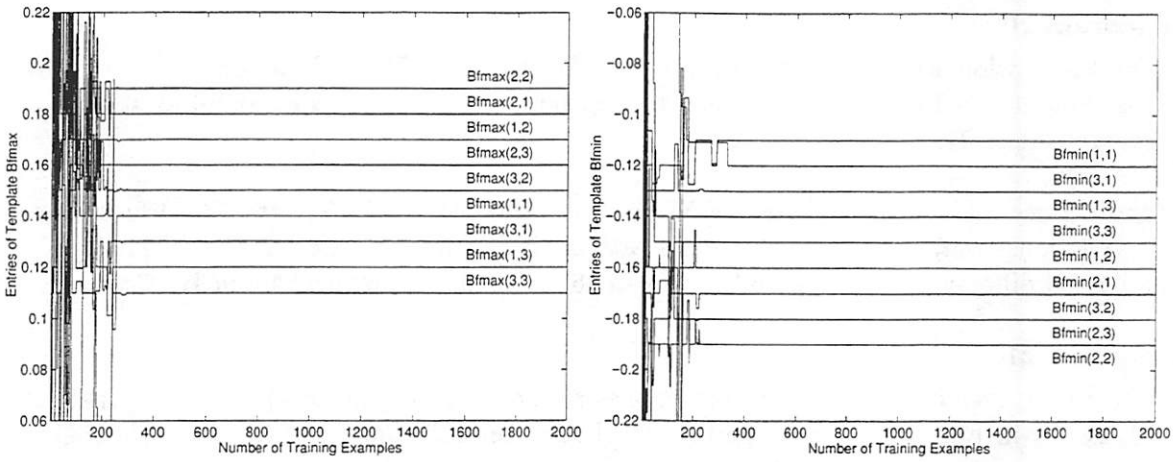
$$\begin{aligned}
\frac{dE}{dt} &= \sum_{ij} \sum_{C_{kl} \in N_r(ij)} \frac{\tilde{\partial} E}{\tilde{\partial} w(i,j;k,l)} \frac{dw(i,j;k,l)}{dt} \\
&= \sum_{ij} \sum_{C_{kl} \in N_r(ij)} \frac{\tilde{\partial} E}{\tilde{\partial} w(i,j;k,l)} \Delta w(i,j;k,l) \\
&= -\mu \sum_{ij} \sum_{C_{kl} \in N_r(ij)} \frac{\tilde{\partial} E}{\tilde{\partial} w(i,j;k,l)} \frac{\tilde{\partial} E}{\tilde{\partial} w(i,j;k,l)} \\
&\leq 0
\end{aligned} \tag{6.39}$$

□

6.2.1 Examples

In this section, the advanced learning algorithms of FDTCNN are used to learning structuring elements from examples. In this sense, the FDTCNN is a kind of morphological network with learning ability. We use two examples to show the usefulness of the learning algorithms to structuring element learning. Let the structuring element S be the same in Eq.(6.24), we then use the dilation operator to generate 2000 samples $\{(u_{ij}, O_{ij})\}$ as training data to train a dilation FDTCNN. The learning process of B_{fmax} template is shown in Fig.6.3(a). One can see that elements of B_{fmax} approach correct values(see Eq.(6.25)) within 300 iterations. The initial conditions for B_{fmax} template is 0. $\mu = 1$. Since the B_{fmax} template is of size 3×3, we only need a 3×3 FDTCNN to learn the structuring element.

Next we use the erosion operator to generate 2000 samples $\{(u_{ij}, O_{ij})\}$ as training data to train an erosion FDTCNN. The learning process of B_{fmin} template is shown in Fig.6.3(b). One can see that elements of B_{fmin} approach correct values(see Eq.(6.26)) within 400 iterations. The initial conditions for B_{fmin} template is 0. $\mu = 1$. The above examples show that our FDTCNN learning algorithms work well.



(a) (b)

Figure 6.3: Learning process of dilation FDTCNN and erosion FDTCNN by using the advanced learning algorithms. (a) Training dilation FDTCNN. (b) Training erosion FDTCNN.

Comparing the results in Fig.6.3 and those in Figs.6.1 and 6.2 we find that the learning time of the learning algorithms presented in this section is much shorter than that presented in previous section.

To show the fact that the learning algorithm can get correct learning results, we also show the learning results of different types of structuring elements.

The next one is the so called *flat structuring element*, which has its elements all the same grey-scale values.

$$S = \begin{bmatrix} 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 \end{bmatrix} \tag{6.40}$$

The learning process of B_{fmax} template is shown in Fig.6.4(a). One can see that entries of B_{fmax} approach correct values within 900 iterations. The initial conditions for B_{fmax} template is 0. The learning process of B_{fmin} template is shown in Fig.6.4(b). One can see that elements of B_{fmin} approach correct values within 1400 iterations. The initial conditions for B_{fmin} template is 0. Also for comparison, we present the results of the learning algorithms presented in the previous section. The learning process of B_{fmax} template and B_{fmin} template are shown in Fig.6.4(c) and Fig.6.4(d), respectively. The results in Fig.6.4(b) and Fig.6.4(d) are something misleading due to the low resolution of the printer. After 1400 iteration, the learning errors in Fig.6.4(b) are much smaller than those in Fig.6.4(d).

We have performed extensive simulations using different templates, it seems very hard to find the cases when the learning algorithms does not converge to the correct results(global minimum).

We present learning algorithms which make additive DTCNNs learn their templates from examples. The applications to learning structuring elements used by grey-scale erosion and grey-scale dilation from examples are presented. The FDTCNN structure used in this paper is a type-II uncoupled FDTCNN. We present theoretical results to guarantee that our learning algorithms to converge to a local minimum of the cost function. Since the surface of E is very complex, the choice of initial conditions and μ is very important to

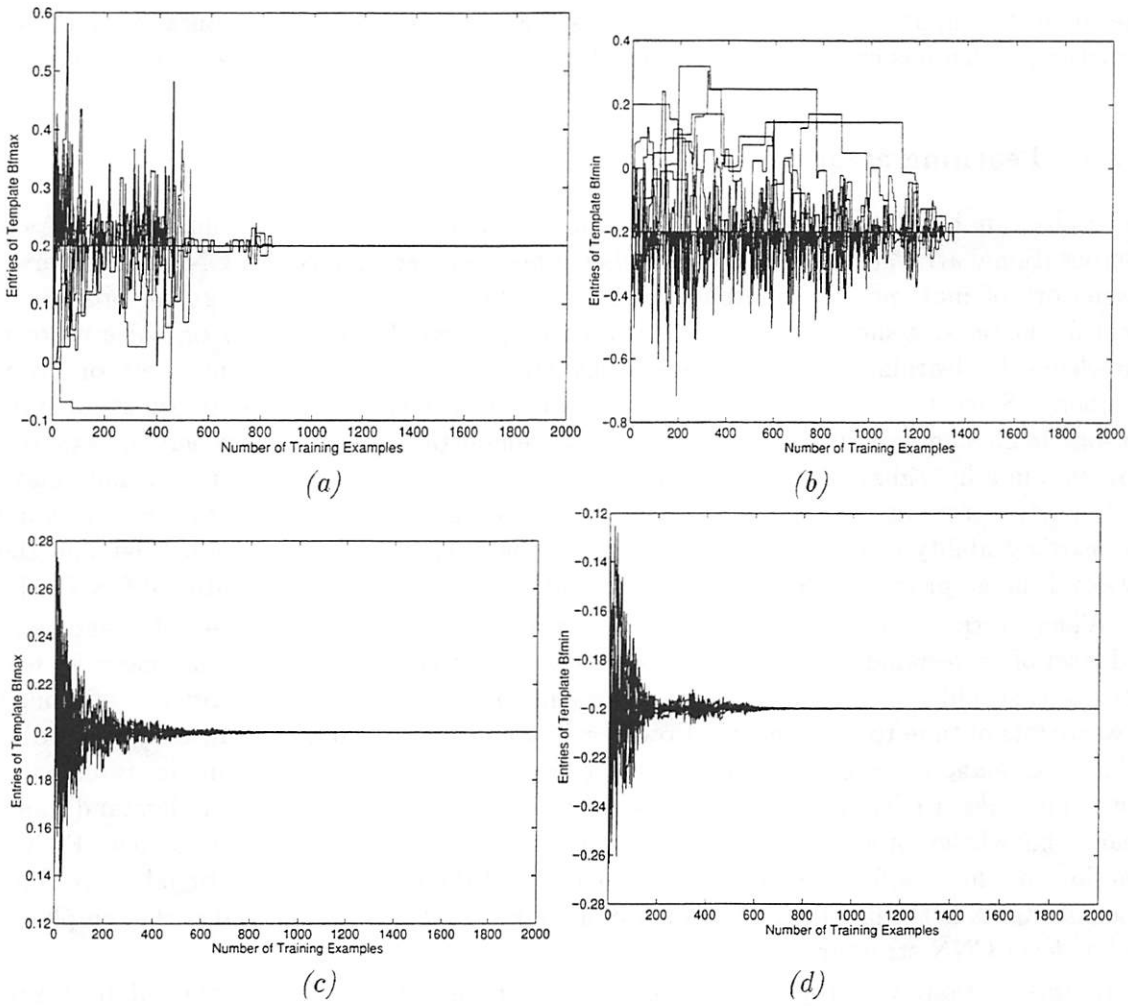


Figure 6.4: Learning process of dilation FCNN and erosion FCNN for flat structuring element. (a) Training dilation FCNN using advanced learning algorithm. (b) Training erosion FCNN using advanced learning algorithm. (c) Training dilation FCNN using the old learning algorithm of the previous section. (d) Training erosion FCNN using the old learning algorithm of the previous section.

get global minimum. The authors of [397] proposed a method to get global minimum by randomly choosing many groups of initial conditions and then choosing the best one from these training results. However, the local minimum problem is still an open problem for almost all the existed learning algorithms.

6.3 Learning from linguistic inputs

In this section, a learning algorithm for an FDTCNN is presented. Unlike the fuzzy cellular neural networks we proposed before[350, 351, 352, 354, 353, 356, 355], this FDTCNN can process fuzzy number inputs besides real number inputs. Its learning algorithm is also based on *fuzzy number inputs*. One application to impulsive noise identification is given to demonstrate the usefulness of this FDTCNN and its learning algorithm. First, a learning FDTCNN, which has fuzzy number input and state, is used to learn its crisp templates from

the linguistic inputs. Then, these templates are embedded into a computational FCNN structure, which has crisp input and state, for purpose of impulsive noise identification.

6.3.1 Learning algorithm

Generally, an FCNN can be used as a computational array or a learning array. As a computational array, the synaptic weights being pre-designed and fixed, FCNN is a universal framework of mathematical morphology network[351, 352] and a paradigm of processing local linguistic statements[354, 355]. As a learning array, FCNN should organize its own knowledge by learning from examples which may be related to crisp numbers or fuzzy numbers. Since the conventional CNN can only process the numerical information from sensors (e.g., camera), it can't learn from the linguistic information of human experts. However, in a hybrid system, the knowledge represented by fuzzy if-then rules usually plays an important role in the high-level of image processing and understanding. So, we hope the learning ability of FCNN can bridge the gap between the linguistic knowledge and the low-level image processing ability of the conventional CNN(or, the conventional CNNUM).

When we train a CNN, we need a set of examples which consists of a set of phenomena and a set of corresponding results. So, we have to collect enough data. In some cases we feel CNN is so stupid that it can only learn knowledge from data which may cost lots of money or waste lots of time to be collected. However, human experts had accumulated a huge body of knowledge and experiences which can not be expressed by data but linguistic statements. If we can make a CNN-based hybrid system which is smart enough to "understand" and "learn" knowledge of human experts', we may save money and time. In this view, FCNN functions as the interface between human expert and the low-level conventional CNN. The input of FCNN is the linguistic statement of human expert, and the output is the templates for low-level CNN structures.

In this section, we proposed an FDTCNN structure which can be trained by fuzzy number (i.e., a convex and normal fuzzy set on a real line[156]). This FDTCNN has a crisp structure which allows fuzzy number information flow through it. So, the synaptic weights in this FDTCNN are crisp set while the inputs, states and outputs are fuzzy numbers. This FDTCNN structure can process the knowledge of human expert's.

In particular, we teach this FDTCNN how to remove impulsive noise in an image using linguistic variables. To remove impulsive noises in images, median filters are usually used[10, 187, 238]. Although median filters have some edge-preserving capabilities, they distort the fine structures of images (thin lines in the image may disappear and the image becomes a little blurred). One can use weighted median filters[392] or conditional median filters[10, 187] to improve the performance. However, setting weights of a weighted median filter is very difficult, so we don't discuss this kind of median filter henceforth. A conditional median filter outputs the median value if an impulsive noise is identified and keeps the input value unchanged if no impulsive noise is identified. So, identification of impulsive noise plays the most important role in conditional median filter. In [10, 187], the fuzzy rule based methods are used to identify impulsive noise and have high performances. However, to design the fuzzy rules and choose the membership functions are difficult. To overcome this problem, the FDTCNN learning algorithm is used to learn these fuzzy rules from the linguistic examples which are based on our experience.

In this section we use the symbol " ~ " over a character to denote a fuzzy number. To reduce the computational complex, the LR-type fuzzy number[74] is used. A fuzzy number

$\tilde{x}$ is said to be LR-type if and only if

$$\mu_{\tilde{x}} = \begin{cases} g_L\left(\frac{c-y}{a}\right), & for\ y < c, a > 0 \\ g_R\left(\frac{y-c}{b}\right), & for\ y \geq c, b > 0 \end{cases} \tag{6.41}$$

where $\mu_{\tilde{x}}$ is the membership function of $\tilde{x}$. $g_L(\cdot)$ and $g_R(\cdot)$ are the reference functions for left and right references. c denotes the mean values of $\tilde{x}$. a and b denote left and right references, respectively. If a and b are both zero then $\tilde{x}$ is degraded to a crisp number.

We define addition of two fuzzy numbers $\tilde{x}$ and $\tilde{y}$ as:

$$\mu_{\tilde{x}+\tilde{y}}(z) = \max\{\mu_{\tilde{x}}(x) \wedge \mu_{\tilde{y}}(y) | z = x + y\} \tag{6.42}$$

and define multiplication of a real number k and a fuzzy number $\tilde{x}$ as:

$$\mu_{k\tilde{x}}(y) = \max\{\mu_{\tilde{x}}(x) | y = kx\} \tag{6.43}$$

For a monotonically increasing function $f(\cdot)$, we define $f(\tilde{x})$ as

$$\mu_{f(\tilde{x})}(y) = \max\{\mu_{\tilde{x}}(x) | y = f(x)\} \tag{6.44}$$

The h-level set, $\tilde{x}^h$, of $\tilde{x}$ is defined by:

$$\tilde{x}^h = \{x | \mu_{\tilde{x}}(x) \geq h, h \in (0, 1]\} \tag{6.45}$$

So, $\tilde{x}^h$ is a closed interval denoted by:

$$\tilde{x}^h = [\tilde{x}_L^h, \tilde{x}_U^h] \tag{6.46}$$

where the subscripts "L" and "U" denote the lower limit and the upper limit, respectively. We define addition of two intervals $[x_L, x_U]$ and $[y_L, y_U]$ as[5]:

$$[x_L, x_U] + [y_L, y_U] = [x_L + y_L, x_U + y_U] \tag{6.47}$$

and define multiplication between a real number k and an interval $[x_L, x_U]$ as[5]:

$$k[x_L, x_U] = \begin{cases} [\mathrm{k}x_L,\ \mathrm{k}\ x_U], & k \geq 0 \\ [\mathrm{k}x_U,\ \mathrm{k}\ x_L], & k < 0 \end{cases} \tag{6.48}$$

$max(\cdot)$ and $min(\cdot)$ operations are defined by:

$$\max\{[x_L, x_U], [y_L, y_U]\} = [\max(x_L, y_L), \max(x_U, y_U)] \tag{6.49}$$

$$\min\{[x_L, x_U], [y_L, y_U]\} = [\min(x_L, y_L), \min(x_U, y_U)] \tag{6.50}$$

For a monotonically increasing function $f(\cdot)$ we define $f([x_L, x_U])$ as

$$f([x_L, x_U]) = [f(x_L), f(x_U)] \tag{6.51}$$

Then the following relations can be easily found:

$$(\tilde{x} + \tilde{y})^h = \tilde{x}^h + \tilde{y}^h \tag{6.52}$$

$$(k\tilde{x})^h = k\tilde{x}^h \tag{6.53}$$

$$(f(\tilde{x}))^h = f(\tilde{x}^h) \tag{6.54}$$

$$(\max\{\tilde{x}, \tilde{y}\})^h = \max\{\tilde{x}^h, \tilde{y}^h\} \tag{6.55}$$

$$(\min\{\tilde{x}, \tilde{y}\})^h = \min\{\tilde{x}^h, \tilde{y}^h\} \tag{6.56}$$

A cell C_{ij} in an $M \times N$ FDTCNN used in this section can be defined by:
State equation:

$$\tilde{x}_{ij}(t+1) = \tilde{F}_{C_{kl} \in N_r(i,j)}\Big(B(i,j;k,l)\tilde{u}_{kl}\Big), 1 \le i \le M, 1 \le j \le N \tag{6.57}$$

where $\tilde{x}_{ij}(t+1)$ is the state of C_{ij} at discrete-time $t+1$. $\tilde{x}_{ij}(t+1)$ is a fuzzy number. $\tilde{F}(\cdot)$ denotes a fuzzy local operator defined in r-neighborhood $N_r(i,j)$. $\tilde{u}_{kl}$ is the input of C_{kl} and a fuzzy number.

Since the above FDTCNN doesn't have feedback synaptic weight, its output equation is given by:

$$\tilde{y}_{ij}(t) = f(\tilde{x}_{ij}(t)), 1 \le i \le M, 1 \le j \le N \tag{6.58}$$

where $f(\cdot)$ is a monotonically increasing nonlinear function given by:

$$f(x) = \frac{1}{1 - e^{-2(x-0.5)}} \tag{6.59}$$

The conventional DTCNN has an $f(\cdot)$ as a $sgn(\cdot)$ function[120]. However, when an FDTCNN is subjected to a learning process, a continuous first order derivative of $f(\cdot)$ should be used. It is easy to see that $f(\cdot)$ defined in Eq.(6.59) satisfies this condition.

Remarks:

On can see that the FDTCNN structure in Eq.(6.57) is completely different from the structures we proposed before[350, 351, 352, 354, 353, 356, 355]. In our previous FCNN structures the membership values were mapped to either crisp values or other membership values, which are real numbers, i.e., only the real numbers are propagated through these FCNN structures. The FDTCNN structure in Eq.(6.57) can map fuzzy numbers to crisp values or fuzzy numbers and allows fuzzy numbers propagate through it. Although in [350, 351, 352, 354, 353, 356, 355] we have demonstrated that the general FCNN structure is not a kind of conventional NCNN, we didn't present examples of FCNN structures which cannot be included into the classical CNN with nonlinear synaptic laws. However, it is easy to find that the FDTCNN in Eq.(6.57) is totally different from any kind of conventional NCNN because the fuzzy number can flow through this structure.

This FDTCNN structure is very useful to classification problems where input patterns are fuzzy numbers. Since the structure in Eq.(6.57) is very general, we would like to study the learning algorithm of one of its simple form as follow:

$$\begin{aligned} \tilde{x}_{ij}(t+1) \;=\; & \tilde{\bigvee_{C_{kl} \in N_r(i,j)}} B_1(i,j;k,l)\tilde{u}_{kl} + \tilde{\bigwedge_{C_{kl} \in N_r(i,j)}} B_2(i,j;k,l)\tilde{u}_{kl}, \\ & 1 \le i \le M, 1 \le j \le N \end{aligned} \tag{6.60}$$

This FDTCNN shares the same mathematical form of a simple min/max FCNN we proposed in[350, 351, 352, 354, 353, 356, 355]. However, since its inputs and states are totally different from those we proposed before, it is a new FCNN structure.

In this section we propose the learning algorithm of DTCNN for two-class classification problems. Assume that we have the following example set:

$$\left\{(\{\tilde{u}_{ij}\}, O_{ij})\right\} \tag{6.61}$$

where $\{\tilde{u}_{ij}\}$ is a set of fuzzy number given by:

$$\{\tilde{u}_{ij}\} = \{\tilde{u}_{kl} | C_{kl} \in N_r(i,j)\} \tag{6.62}$$

O_{ij} is a classification result given by:

$$O_{ij} = \begin{cases} 1, & \text{if } C_{ij} \text{ belongs to class 1} \\ 0, & \text{if } C_{ij} \text{ belongs to class 2} \end{cases} \tag{6.63}$$

If we use the output of C_{ij} to denote the classification result of C_{ij}, we have

$$\tilde{y}_{ij}(t) = \begin{cases} 1, & \text{if } C_{ij} \text{ belongs to class 1} \\ 0, & \text{if } C_{ij} \text{ belongs to class 2} \end{cases} \tag{6.64}$$

From above one can see that $\tilde{y}_{ij}(t)$ degenerates to a crisp number. It is because the nonlinear output function $f(\cdot)$ functions as a defuzzifier. An explicit expression of this kind of defuzzifier can be given by:

$$\tilde{y}_{ij}^h(t) = f(\tilde{x}_{ij}^h(t)) \approx \begin{cases} 1, & for\ \tilde{x}_{ijL}^h(t) \geq 1 \\ 0, & for\ \tilde{x}_{ijU}^h(t) \leq 0 \end{cases} \tag{6.65}$$

where $h \in (0, 1]$.

Then, given an h-level set of $\tilde{y}_{ij}$, our training objective is to minimize the following cost function

$$E = \frac{1}{2} \sum_{ij} \max(O_{ij} - \tilde{y}_{ij}^h)^2 \tag{6.66}$$

where

$$\max(O_{ij} - \tilde{y}_{ij}^h)^2 = \begin{cases} (O_{ij} - f(\tilde{x}_{ijL}^h))^2, & if\ O_{ij} = 1 \\ (O_{ij} - f(\tilde{x}_{ijU}^h))^2, & if\ O_{ij} = 0 \end{cases} \tag{6.67}$$

From above one can see that we should train the FDTCNN using different h-level sets. An increase of the number of h-level sets improves the training results but increases the training time too. So, there may be a trade-off between the number of h-level sets and the performance of the training results. To train the FDTCNN, we use the following learning rules to update two kinds of feed-forward synaptic weights $B_1(i,j;p,q)$ and $B_2(i,j;p,q)$, respectively:

$$\Delta B_1(i,j;p,q)(t+1) = \alpha\Big(-\frac{\partial E}{\partial B_1(i,j;p,q)}\Big) + \beta\Delta B_1(i,j;p,q)(t) \tag{6.68}$$

$$\Delta B_2(i,j;p,q)(t+1) = \alpha\Big(-\frac{\partial E}{\partial B_2(i,j;p,q)}\Big) + \beta\Delta B_2(i,j;p,q)(t) \tag{6.69}$$

where t is the learning iteration. $C_{pq} \in N_r(i,j)$. α and β are learning rate and momentum rate, respectively. In the right hand side of Eq.(6.68) the $\frac{\partial E}{\partial B_1(i,j;p,q)}$ is given by:

$$\begin{aligned}\frac{\partial E}{\partial B_1(i,j;p,q)} &= \frac{\partial E}{\partial \tilde{y}_{ij}}\frac{\partial \tilde{y}_{ij}}{\partial \tilde{x}^h_{ij}}\frac{\partial \tilde{x}^h_{ij}}{\partial B_1(i,j;p,q)} \\ &= -\delta\theta\frac{\partial \tilde{x}^h_{ij}}{\partial B_1(i,j;p,q)}\end{aligned} \tag{6.70}$$

where

$$\delta = -\frac{\partial E}{\partial \tilde{y}_{ij}} = \begin{cases} (O_{ij} - f(\tilde{x}^h_{ijL})), & if\ O_{ij} = 1 \\ (O_{ij} - f(\tilde{x}^h_{ijU})), & if\ O_{ij} = 0 \end{cases} \tag{6.71}$$

and

$$\theta = \frac{\partial \tilde{y}_{ij}}{\partial \tilde{x}_{ij}} = \begin{cases} f'(\tilde{x}^h_{ijL}), & if\ O_{ij} = 1 \\ f'(\tilde{x}^h_{ijU}), & if\ O_{ij} = 0 \end{cases} \tag{6.72}$$

Since $\tilde{x}_{ij}$ is a fuzzy number, then we can train the FDTCNN using the h-level interval numbers as:

$$\begin{aligned}&\frac{\partial \tilde{x}^h_{ij}}{\partial B_1(i,j;p,q)} \\ =\ &\frac{\partial \tilde{\bigvee}_{C_{kl}\in N_r(i,j)} B_1(i,j;k,l)\tilde{u}^h_{kl}}{\partial B_1(i,j;p,q)} \\ =\ &\frac{\partial \max\Big(B_1(i,j;p,q)\tilde{u}^h_{pq}, \tilde{\bigvee}_{C_{kl}\in N_r(i,j),kl\neq pq} B_1(i,j;k,l)\tilde{u}^h_{kl}\Big)}{\partial B_1(i,j;p,q)} \\ =\ &\frac{\partial \max\Big(\overbrace{B_1(i,j;p,q)\tilde{u}^h_{pq*}}^{u_1}, \overbrace{\tilde{\bigvee}_{C_{kl}\in N_r(i,j),kl\neq pq} B_1(i,j;k,l)\tilde{u}^h_{kl}}^{\Xi}\Big)}{\partial \underbrace{B_1(i,j;p,q)\tilde{u}^h_{pq*}}_{u_1}}\tilde{u}^h_{pq*} \\ =\ &\frac{\partial \max(u_1,\Xi)}{\partial u_1}\tilde{u}^h_{pq*}\end{aligned} \tag{6.73}$$

where

$$\tilde{u}^h_{pq*} = \begin{cases} \tilde{u}^h_{pqU}, & if\ B_1(i,j;p,q) \geq 0 \\ \tilde{u}^h_{pqL}, & if\ B_1(i,j;p,q) < 0 \end{cases} \tag{6.74}$$

Then we consider the so called "smooth derivative"[26] of $max(u_1, \Xi)$. In the classical sense, $max(u_1, \Xi)$ is derivable into the open intervals $u_1 < \Xi$ and $u_1 > \Xi$ but the derivative is not defined at $u_1 = \Xi$, i.e.,

$$\frac{\partial \max(u_1, \Xi)}{\partial u_1} = \begin{cases} 1, & if\ u_1 > \Xi \\ 0, & if\ u_1 < \Xi \end{cases} \tag{6.75}$$

From Eq.(6.75) we know that the FDTCNN will stop learning when $u_1 < \Xi$. This makes the learning process of FDTCNN very slow. In the worst case, this can even make the learning process impossible. To overcome this problem, we notice that Eq.(6.75) only gives the crisp truth value of statement: "u_1 is greater than Ξ". In this sense, we can fuzzify Eq.(6.75) using different methods. One example can be found in [26]. But the method used in [26] can not be used here, we fuzzifier our "smooth derivative" as:

$$\frac{\partial \max(u_1, \Xi)}{\partial u_1} = \begin{cases} 1, & if\ u_1 > \Xi \\ \min\{\frac{|u_1|}{|\Xi|}, \frac{|\Xi|}{|u_1|}\}, & if\ u_1 \leq \Xi \end{cases} \tag{6.76}$$

Then we have

$$\frac{\partial \tilde{x}_{ij}^h}{\partial B_1(i,j;p,q)} = \begin{cases} \tilde{u}_{pq*}^h, & if\ u_1 > \Xi \\ \tilde{u}_{pq*}^h \min\{\frac{|u_1|}{|\Xi|}, \frac{|\Xi|}{|u_1|}\}, & if\ u_1 \leq \Xi \end{cases} \tag{6.77}$$

Similarly, in the right hand side of Eq.(6.69) the $\frac{\partial E}{\partial B_2(i,j;p,q)}$ is given by:

$$\frac{\partial E}{\partial B_2(i,j;p,q)} = -\delta\theta \frac{\partial \tilde{x}_{ij}^h}{\partial B_2(i,j;p,q)} \tag{6.78}$$

where δ and θ are given by Eqs. (6.71) and (6.72), respectively. Then we have

$$\begin{aligned} & \frac{\partial \tilde{x}_{ij}^h}{\partial B_2(i,j;p,q)} \\ = & \frac{\partial \tilde{\bigwedge}_{C_{kl} \in N_r(i,j)} B_2(i,j;k,l)\tilde{u}_{kl}^h}{\partial B_2(i,j;p,q)} \\ = & \frac{\partial \min\left(B_2(i,j;p,q)\tilde{u}_{pq}^h, \tilde{\bigwedge}_{C_{kl} \in N_r(i,j), kl \neq pq} B_2(i,j;k,l)\tilde{u}_{kl}^h\right)}{\partial B_2(i,j;p,q)} \\ = & \frac{\partial \min\left(\overbrace{B_2(i,j;p,q)\tilde{u}_{pq*}^h}^{u_2}, \overbrace{\tilde{\bigwedge}_{C_{kl} \in N_r(i,j), kl \neq pq} B_2(i,j;k,l)\tilde{u}_{kl}^h}^{\Upsilon}\right)}{\partial \underbrace{B_2(i,j;p,q)\tilde{u}_{pq*}^h}_{u_2}} \tilde{u}_{pq*}^h \\ = & \frac{\partial \min(u_2, \Upsilon)}{\partial u_2} \tilde{u}_{pq*}^h \end{aligned} \tag{6.79}$$

where

$$\tilde{u}_{pq*}^h = \begin{cases} \tilde{u}_{pqL}^h, & if\ B_2(i,j;p,q) \geq 0 \\ \tilde{u}_{pqU}^h, & if\ B_2(i,j;p,q) < 0 \end{cases} \tag{6.80}$$

Similarly, the "smooth derivative" of $\min(u_2, \Upsilon)$ can be given by:

$$\frac{\partial \min(u_2, \Upsilon)}{\partial u_2} = \begin{cases} 1, & if\ u_2 < \Upsilon \\ \min\{\frac{|u_2|}{|\Upsilon|}, \frac{|\Upsilon|}{|u_2|}\}, & if\ u_2 \geq \Upsilon \end{cases} \tag{6.81}$$

Then we have

$$\frac{\partial \tilde{x}_{ij}^h}{\partial B_2(i,j;p,q)} = \begin{cases} \tilde{u}_{pq*}^h, & if\ u_2 < \Upsilon \\ \tilde{u}_{pq*}^h \min\{\frac{|u_2|}{|\Upsilon|}, \frac{|\Upsilon|}{|u_2|}\}, & if\ u_2 \geq \Upsilon \end{cases} \tag{6.82}$$

6.3.2 Application to impulsive noise identification

Impulsive noises in an image can be removed by using nonlinear filter such as medium filter, rank-order filter or using mathematical morphology operator. However, almost all the above filtering methods blur the fine structures of the parts of the image where impulsive noises don't exist. So, there exists a kind of expert knowledge based method to remove impulsive noises while keep the region without impulsive noise unchanged[187, 10]. The first step of this kind of method is to identify the locations of impulsive noises based on linguistic statements of knowledge of impulsive noises.

If we assume that a real image are smooth enough, then an impulsive noise will introduce a significant difference of gray value from its neighbors. Our visual system has an experience that if a pixel has a gray value which is significantly different from all its neighbors should be an impulsive noise. To make this experience understandable to an FDTCNN, we first translate it into a set of fuzzy if-then rules[187].

Consider a 3×3 neighborhood ($N_1(i,j)$) and use u_{ij} to denote the gray value of pixel (i,j) in the image, we have:

If $\lvert u_{i,j-1} - u_{ij}\rvert$ is *big* and $\lvert u_{i,j+1} - u_{ij}\rvert$ is *big*
and
If $\lvert u_{i-1,j} - u_{ij}\rvert$ is *big* and $\lvert u_{i+1,j} - u_{ij}\rvert$ is *big*
and
If $\lvert u_{i-1,j-1} - u_{ij}\rvert$ is *big* and $\lvert u_{i+1,j+1} - u_{ij}\rvert$ is *big*
and
If $\lvert u_{i-1,j+1} - u_{ij}\rvert$ is *big* and $\lvert u_{i+1,j-1} - u_{ij}\rvert$ is *big*
THEN u_{ij} is an impulsive noise.

Where "*big*" is a fuzzy number. Since the characteristics of impulsive noises are changed from one image to another, the human experts will have different qualitative statements for "*big*". The FDTCNN can do a trade-off between the judgments of human experts by learning from different linguistic examples (knowledge from different human experts).

To train the FDTCNN, we define the fuzzy number *big*, *middle* and *small* as shown in Fig. 6.5. From Fig. 6.5 one can see that membership functions of fuzzy numbers *small*, *middle* and *big* can be expressed by:

$$\mu_S(x) = \begin{cases} 0, & x < 0 \\ \max\{0, 1 - 3x\}, & x \geq 0 \end{cases} \tag{6.83}$$

$$\mu_M(x) = \max\{0, 1 - 4|\frac{1}{2} - x|\} \tag{6.84}$$

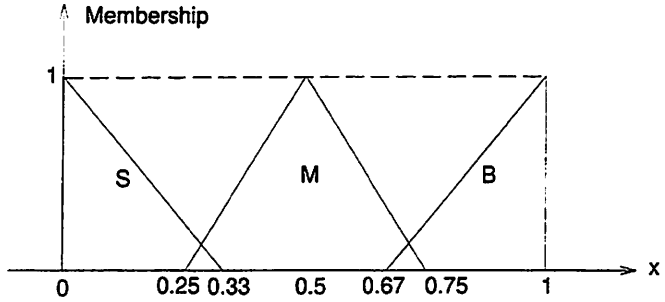


Figure 6.5: Membership function of three fuzzy numbers: $small(S)$, $middle(M)$ and $big(B)$.

$$\mu_B(x) = \begin{cases} \max\{0, 1-3|1-x|\}, & x \leq 1 \\ 0, & x > 1 \end{cases} \tag{6.85}$$

From above we know that we can used the following FDTCNN to identify impulsive noise:

$$\begin{aligned} \tilde{x}_{ij}(t+1) = & \tilde{\bigvee_{k\in\{-1,0,1\},l\in\{-1,0,1\}}} B_{max}(k,l)(t)\tilde{u}^*_{i+k,j+l} \\ & + \tilde{\bigwedge_{k\in\{-1,0,1\},l\in\{-1,0,1\}}} B_{min}(k,l)(t)\tilde{u}^*_{i+k,j+l} \\ & , 1 \leq i \leq M, 1 \leq j \leq N \end{aligned} \tag{6.86}$$

where $B_{max} = \{B_{max}(k,l)\}_{3\times3}$ and $B_{min} = \{B_{min}(k,l)\}_{3\times3}$ are two feed-forward templates. $\tilde{u}^*_{i+k,j+l}$ denotes the fuzzy number which is used to describe the uncertainty of $|u_{i+k,j+l}-u_{ij}|$. Since $|u_{ij}-u_{ij}| = 0$ is always true, $B_{min}(0,0)$ and $B_{max}(0,0)$ are *don't care* entries. In this paper, we let $B_{min}(0,0) = 0$ and $B_{max}(0,0) = 0$.

Since the FDTCNN in Eq.(6.86) is space-invariant, to learn a 3×3 template, we only need a 3×3 FDTCNN. And since the training process only needs knowledge form a human expert, we can generate the training examples as shown in Fig. 6.6. In Fig. 6.6, every input pattern denotes a possible configuration of $\tilde{u}^*_{i+k,j+l}$'s in $N_1(i,j)$. There are two classes of examples are illustrated. There is only one pattern in class 1 which has output 1(impulsive noise) while all the 8 patterns in class 2 have outputs 0's(not an impulsive noise).

We train the FDTCNN using examples choosing from class 1 and class 2 randomly. During the first 2000 examples, we chose 80% of examples from class 1 and the rest from class 2. This makes the learning precess faster. After that, we choose only 8% of examples from class 1. This makes the learning process slower and smoother.

Fig. 6.7 shows the learning curves of $B_{min}(1,1)$ and $B_{max}(1,1)$ with parameters: $\alpha = 0.5$, $\beta(t) = 0.5 \times (0.999)^t$. The initial values of the entries of templates B_{min} and B_{max} are chosen randomly in interval (0,1). Since the ranges of inputs and outputs are in $[0,1]$, in the learning process we restrict the dynamical ranges of $B_{min}(k,l)$ and $B_{max}(k,l)$ in interval $[-1,1]$. From Fig. 6.7 one can see that $B_{min}(1,1)$ approaches 1 while $B_{max}(1,1)$ approaches 0. In this simulation, we use 3 level sets ($h = \frac{1}{3}, \frac{2}{3}$, and 1) of every fuzzy number input pattern to train FDTCNN. After being trained by 40000 examples, the FDTCNN

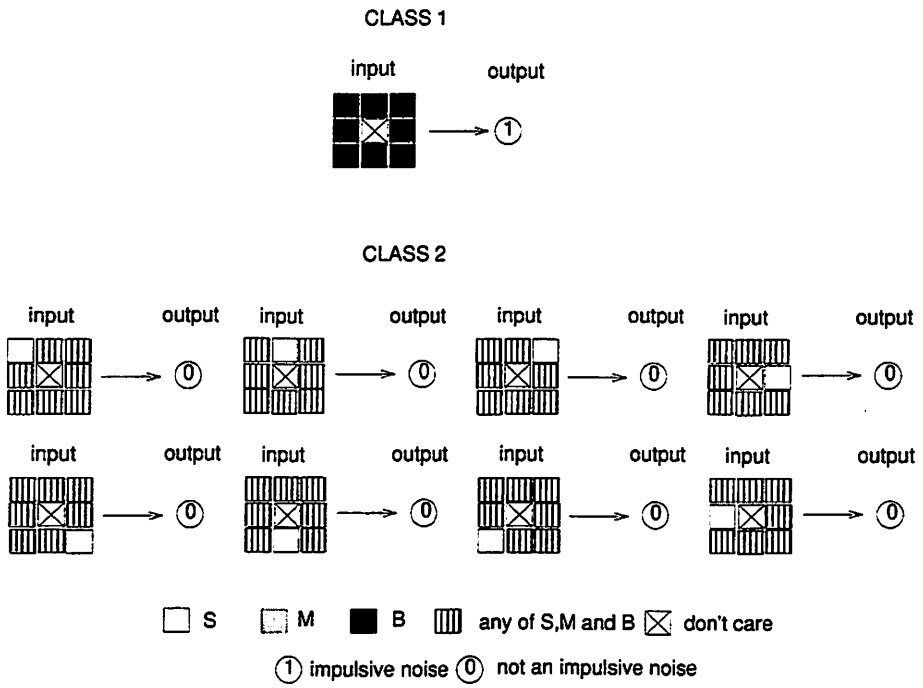


Figure 6.6: Illustrations of patterns of training examples in two classes. Class 1 denotes that there exists an impulsive noise. Class 2 denotes that there doesn't exist an impulsive noise.

learns the following templates:

$$B_{min} = \begin{pmatrix} 0.996746 & 0.996926 & 0.994743 \\ 0.996385 & 0.000000 & 0.997409 \\ 0.997262 & 0.997369 & 0.996796 \end{pmatrix},$$
$$B_{max} = \begin{pmatrix} -0.000003 & -0.000001 & -0.000002 \\ -0.000001 & 0.000000 & -0.000002 \\ -0.000002 & -0.000002 & -0.000003 \end{pmatrix} \quad (6.87)$$

One can see that every entry in B_{min} is very close to 1 and every entry in B_{max} is very close to 0 (one should notice that the central entries of both templates are *don't case* entries). We then use the templates in Eq.(6.87) to process a 63×63 gray-scale image of 256 gray levels, which consists of impulsive noises of mean value 220 and deviation 35, shown in Fig. 6.8(a). The image in Fig. 6.8(a) is used as $u_{ij}, 1 \leq i, j \leq 63$. u_{ij} is normalized such that condition

$$\max_{ij} \max_{kl \in N_1(i,j)} |u_{kl} - u_{ij}| \leq 1 \quad (6.88)$$

is satisfied. After training, synaptic weights of the FDTCNN can be fixed and the FDTCNN is also degenerated into a computational array whose inputs and outputs are crisp values. So, the crisp form of the trained FDTCNN used in this simulation can be written as:

$$\begin{aligned} x_{ij}(t+1) = & \max_{C_{kl} \in N_1(i,j)} B_{max}(k-i, l-j)|u_{kl} - u_{ij}| \\ & + \min_{C_{kl} \in N_1(i,j)} B_{min}(k-i, l-j)|u_{kl} - u_{ij}| \\ & , 1 \leq i \leq M, 1 \leq j \leq N \end{aligned} \quad (6.89)$$

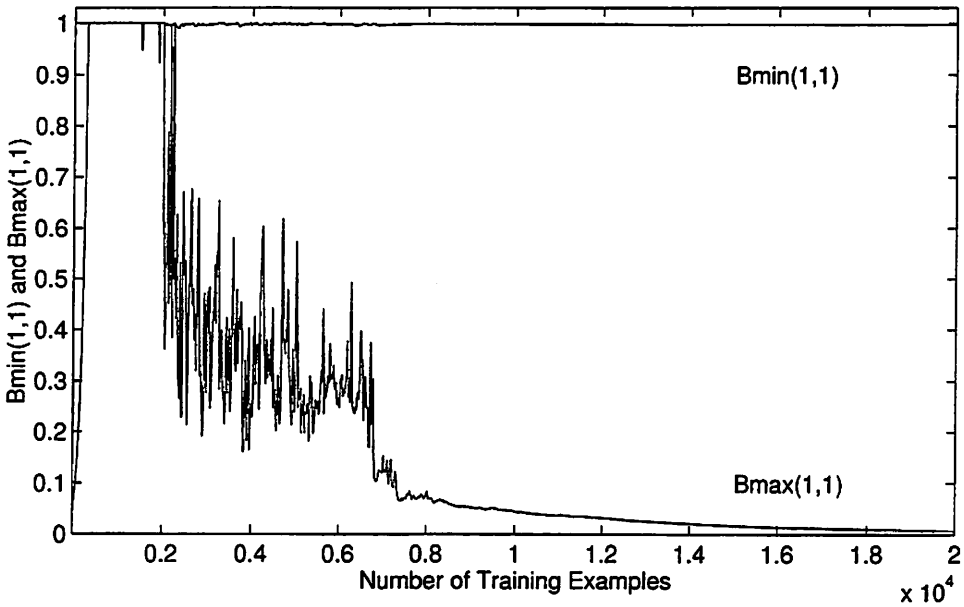


Figure 6.7: Learning curves of $B_{min}(1,1)$ and $B_{max}(1,1)$.

Fig. 6.8(b) shows the output of the above FDTCNN, from which one can see that every impulsive noise is identified except those in the first and the last rows and those in the first and last columns. It is because we used these cells as dumb cells (boundary cells) for 3×3 templates in our simulation. Fig. 6.8(c) shows the thresholded result of Fig. 6.8(b), from which one can see that all impulsive noises are identified. One can see that we never use the crisp examples to train the FDTCNN but it works well when it processes crisp inputs.

We have proposed an FDTCNN structure which can learn from the fuzzy number examples. We also developed a learning algorithm for this FDTCNN structure. Since the templates are space-invariant, we can only use an FDTCNN of the same size of the biggest template to train the synaptic weights. After training, the templates can be used as a standard template for low-level computational FCNN to solve the corresponding image processing problems.

Since fuzzy numbers can be propagated through this FDTCNN structure, this FDTCNN can learn its templates from linguistic knowledge. In some cases where training examples are difficult to be collected and the knowledge of human expert is available, this structure should be useful. On the other hand, this structure can also be used as an interface between the conventional CNN and human experts, designers and users in a CNN-based hybrid image processing system. So, this structure will extend the CNN concept from low-level image processing to high-level image processing and from structure-base image processing to knowledge-based image processing.

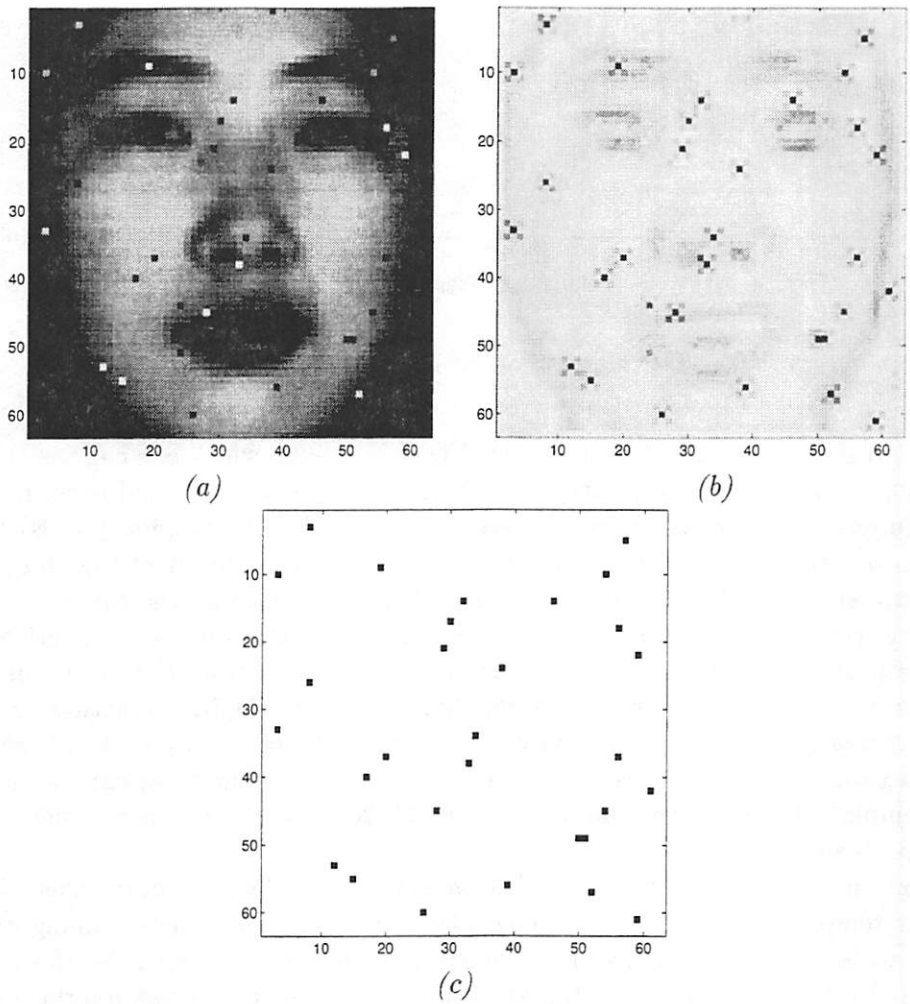


Figure 6.8: The computer simulation results of impulsive noises identification using the trained FDTCNN. (a) The image containing impulsive noises. (b) The output of the trained FDTCNN. (c) The thresholded result of (b).

Chapter 7

Generic algorithm for FCNN

FCNN structures can be used effectively to solve fuzzy IF-THEN-ELSE rules based image processing problems. Given a set of local fuzzy rules, a systematic method is presented for selecting the corresponding FCNN structures in Chap.5. The membership functions of the linguistic variables used in the fuzzy rules should be chosen according to different rules. In a fuzzy IF-THEN-ELSE rule the membership functions, whose choice is usually a very difficult and time-consuming process, play very important roles. In this chapter, a real coded *genetic algorithm*(GA) is used to optimize the membership functions of the chosen FCNN structure. The corresponding crossover and mutation operations are presented. The crossover operation consists of three schemes which are the trade off between the evolution of the best individual and that of the other population. The mutation operation consists of a local one and a global one. The local one makes the evolution search the local basin of the best individual while the global one makes the evolution search the global problem space to overcome the trap of a local optimization. Then the GA is used to optimize the membership functions for solving the edge extraction problem with ill-conditioned examples. This chapter is a collection of our papers: [361, 383].

7.1 Genetic algorithm for optimizing FCNN

GA's are optimization approaches motivated by creature evolution. They combine robustness with the ability to explore huge search space quickly. The basic knowledge of GA can be found in [66, 100]. GA exploits the collective learning process within a population of individuals, and each individual represents a search point in the space of potential solutions to a given problem. The applications of GA to fuzzy logic [115, 15, 85, 130] can roughly lumped into two categories: 1) optimization of the membership functions of fuzzy sets, and 2) automatic learning of fuzzy rules.

We use GA to optimize the membership functions of FCNN. The correct choice of the membership functions plays an important role in the design of FCNN. There exists some applications [115, 15, 85, 130] show that GA are capable of optimizing the membership functions. The basic idea is to represent the complete set of membership functions by an individual and to evolve shapes of the membership functions. We only use the GA to optimize the normalized trapezoidal membership functions which can be represented by a

4-tuple $(a^{(1)}, a^{(2)}, a^{(3)}, a^{(4)})$ as follows[30]:

$$\mu_A(x) = \begin{cases} \frac{x-a^{(1)}}{a^{(2)}-a^{(1)}}, & for\ a^{(1)} \le x \le a^{(2)}, \\ 1, & for\ a^{(2)} \le x \le a^{(3)}, \\ \frac{x-a^{(4)}}{a^{(3)}-a^{(4)}}, & for\ a^{(3)} \le x \le a^{(4)}, \\ 0, & otherwise. \end{cases} \tag{7.1}$$

where A is a trapezoidal fuzzy variable.

A typical GA starts with a randomly chosen population of individuals. Then this population undergoes evolution in a form of natural selection. In each generation, relatively good individuals are reproduced to give offsprings that replace the relatively bad individuals which are eliminated. An evaluation or fitness function is used to distinguish good and bad individuals. A typical GA consists of three basic operations: 1) evaluation of individual fitness, 2) formation of a gene pool, and 3) recombination using two basic genetic operators: crossover and mutation. The GA used in this paper is shown as follows:

```
/* initialize */
Generation t = 0;
Initialize the gene pool=GP(0);
while (not termination-condition) do
        generation t=t+1;
        select individual C(t+1) = {c_i} ∈ GP(t-1);
        crossover c_i, c_j ∈ C(t-1) and get C(t);
        evaluation and selection C(t) and get GP(t);
        mutation GP(t);
end
```

Since in an FCNN-based FIRE edge extractor, only two fuzzy variable ZERO and WHITE are used, the k^{th} individual in the gene pool, c_k, can be represented by:

$$\begin{aligned} c_k &= \{(a_k^{(1)}, a_k^{(2)}, a_k^{(3)}, a_k^{(4)}), (b_k^{(1)}, b_k^{(2)}, b_k^{(3)}, b_k^{(4)})\} \\ &= \{c_k(1), c_k(2), c_k(3), c_k(4), c_k(5), c_k(6), c_k(7), c_k(8)\} \end{aligned} \tag{7.2}$$

where $(a_k^{(1)}, a_k^{(2)}, a_k^{(3)}, a_k^{(4)})$ is a 4-tuple for determining the trapezoidal membership function of ZERO and $(b_k^{(1)}, b_k^{(2)}, b_k^{(3)}, b_k^{(4)})$ is that for WHITE.

Then the initialization of the gene pool is given by the following process. Suppose ξ is a pseudo-random number distributed in $(0, 1)$. It is easy to see that the only choice of $a_k^{(1)}$ of $\mu_{ZE}(x)$ is $a_k^{(1)} = 0$. Then $a_k^{(4)}$ is chosen by

$$a_k^{(4)} = \frac{\xi}{2} \tag{7.3}$$

and then $a_k^{(2)}$ and $a_k^{(3)}$ are respectively chosen by:

$$a_k^{(2)} = a_k^{(1)} + \frac{\xi}{2}(a_k^{(4)} - a_k^{(1)}) \tag{7.4}$$

$$a_k^{(3)} = a_k^{(2)} + \frac{\xi}{2}(a_k^{(4)} - a_k^{(2)}) \tag{7.5}$$

It is easy to see that $b_k^{(4)}$ for $\mu_{WH}(x)$ has a best choice of $b_k^{(4)} = 1$. Then $b_k^{(1)}$ is given by:

$$b_k^{(1)} = 0.5 + \frac{\xi}{2} \tag{7.6}$$

and then $b_k^{(2)}$ and $b_k^{(3)}$ are respectively chosen by:

$$b_k^{(2)} = b_k^{(1)} + \frac{\xi}{2}(b_k^{(4)} - b_k^{(1)}) \tag{7.7}$$

$$b_k^{(3)} = b_k^{(2)} + \frac{\xi}{2}(b_k^{(4)} - b_k^{(2)}) \tag{7.8}$$

Crossover is given by the max-min-arithmetical algorithm presented in [85]. Assume that $c_1 = \{c_1(1), c_1(2), ..., c_1(8)\}$ and $c_2 = \{c_2(1), c_2(2), ..., c_2(8)\}$ are two individuals to be crossed, then the four offsprings are given by:

$$c_1^* = \{c_1^*(i) | c_1^*(i) = \alpha c_1(i) + (1-\alpha)c_2(i), i = 1, 2, ..., 8\} \tag{7.9}$$

$$c_2^* = \{c_2^*(i) | c_2^*(i) = (1-\alpha)c_1(i) + \alpha c_2(i), i = 1, 2, ..., 8\} \tag{7.10}$$

$$c_3^* = \{c_3^*(i) | c_3^*(i) = \max(c_1(i) + c_2(i)), i = 1, 2, ..., 8\} \tag{7.11}$$

$$c_4^* = \{c_4^*(i) | c_4^*(i) = \min(c_1(i) + c_2(i)), i = 1, 2, ..., 8\} \tag{7.12}$$

where $\alpha \in (0, 1)$ is a constant. And then the best ones are selected. The are three crossover schemes used in our GA. The first one, which occurs with a probability p_{c1}, is the crossover between the best individual and the worst one. And then the worst one is substituted by the best offsprings. The second one, which occurs with a probability p_{c2}, is the crossover between the best individual and any of the sub-worst one. And then the sub-worst one is substituted by the best offsprings. The third one, which occurs with a probability p_{c3}, is the crossover between any two of the sub-best ones. And then the two sub-best one is substitute by the two best offsprings.

Mutations consist of a local mutation scheme and a global mutation scheme. The local mutation, which occurs with a probability p_{m1}, is given by following process. Assume that an element of an individual $c_k = \{c_k(1), ..., c_k(i), ..., c_k(8)\}$, $c_k(i), i = 1, ..., 8$, is chosen for local mutation and assume that the domain of $c_k(i)$ is $[d^l, d^r]$, then the result is a new individual $c_k = \{c_k(1), ..., c_k^*(i), ..., c_k(8)\}$, where $c_k^*(i)$ is given by:

$$c_k^*(i) = \begin{cases} c_k^*(i) - \xi^*(c_k^*(i) - d^l), & for\ \xi > 0.5 \\ c_k^*(i) + \xi^*(d^r - c_k^*(i)), & else \end{cases} \tag{7.13}$$

where ξ^* is a pseudo-random number distributed in $(0, 1)$. The $[d^l, d^r]$ for each element of individual c_k is given by: $\{([0, 0], [0, a^{(3)}], [a^{(2)}, a^{(4)}], [a^{(3)}, 1]), ([0, b^{(2)}], [b^{(1)}, b^{(3)}], [b^{(2)}, b^{(4)}], [1, 1])\}$

The global mutation, which occurs with a probability p_{m2}, is the same as initialization. So, the local mutation can be used to improve the existed individual while the global

mutation continuously added new types of individuals into the gene pool during the evolving process.

To evaluate the performance of an individual c_k, the output of FCNN in Eqs.(5.13) and (5.14), y_{ij}, is compared with the ideal output $\{o_{ij}\}$, by using an error function:

$$E_k = \sum_i \sum_j \phi_{ij}(o_{ij} - y_{ij})^2 \tag{7.14}$$

where

$$\phi_{ij}(t) = \begin{cases} \alpha(t), & for\ o_{ij} = 0 \\ \beta(t), & for\ o_{ij} = 1 \end{cases} \tag{7.15}$$

is the evaluation weight. In our simulations, we let $\alpha(t) = 1$ and $\beta(t) = 0.5 + 0.5\xi$. And we define the global fitness of the t^{th} generation gene pool $GP(t)$, E_{min}, as

$$E_{min}(t) = \min_{k=1}^{n} E_k(t) \tag{7.16}$$

where n is the number of population in the gene pool.

7.2 Application to image processing

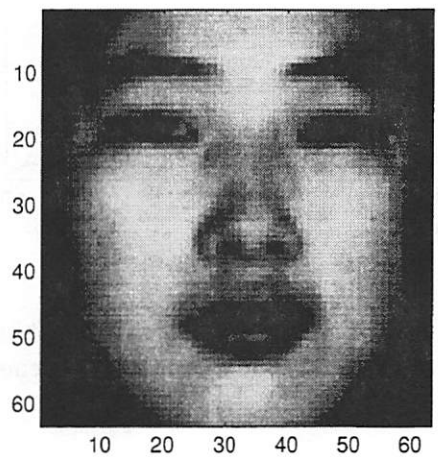
In this section, the computer simulation results are provided. We used the GA to choose $\mu_{ZE}(.)$ and $\mu_{WH}(.)$. Fig. 7.1(a) shows the original gray-scale image of size 63×63 with 256 gray levels. The image is normalized to $[0, 1]$. Fig. 7.1(b) shows a bad version of edge detected result. One can see that lots of noises existed in this result and the edge is almost indistinguishable. Then we use GA to learn $\mu_{ZE}(.)$ and $\mu_{WH}(.)$ from this ill-conditioned example. The parameters of the GA used in this simulation are chosen as:

population size:
$n = 10$.
probability of crossover:
$p_{c1} = 0.1$, $p_{c2} = 0.1$, $p_{c3} = 0.2$.
Max-Min-Arithmetical crossover parameter:
$\alpha = 0.618$.
probability of mutation:
$p_{m1} = 0.2$, $p_{m2} = 0.5$.
stop conditions:
If E_{min} trapped into a local minimum more that 45 generations.

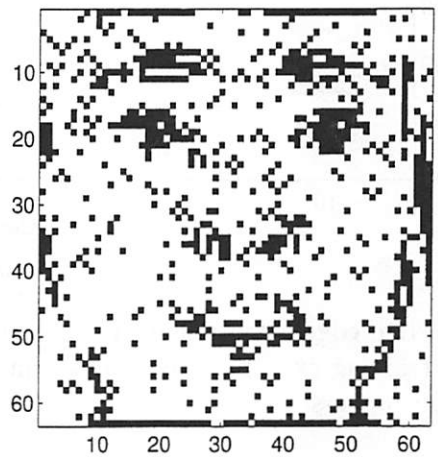
After 100 generations, we get the following individual:

$$\begin{aligned} (a^{(1)}, a^{(2)}, a^{(3)}, a^{(4)}) &= (0.000000, 0.004611, 0.007106, 0.043000) \\ (b^{(1)}, b^{(2)}, b^{(3)}, b^{(4)}) &= (0.543300, 0.675663, 0.689058, 1.000000) \end{aligned} \tag{7.17}$$

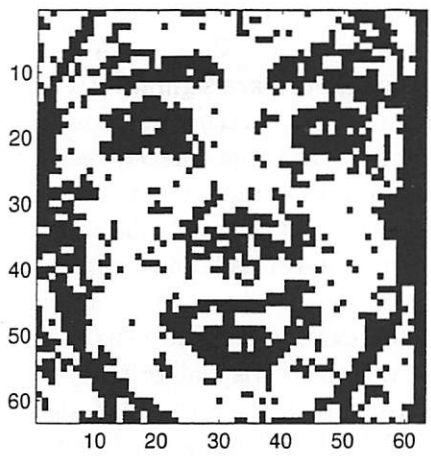
the corresponding output is shown in Fig. 7.1(c). One can see that the edge characteristics are enhanced while the noise is suppressed. In this simulation, h in Eq.(5.14) is chosen as $h = 0$.


10
20
30
40
50
60
10 20 30 40 50 60

(a)


10
20
30
40
50
60
10 20 30 40 50 60

(b)


10
20
30
40
50
60
10 20 30 40 50 60

(c)

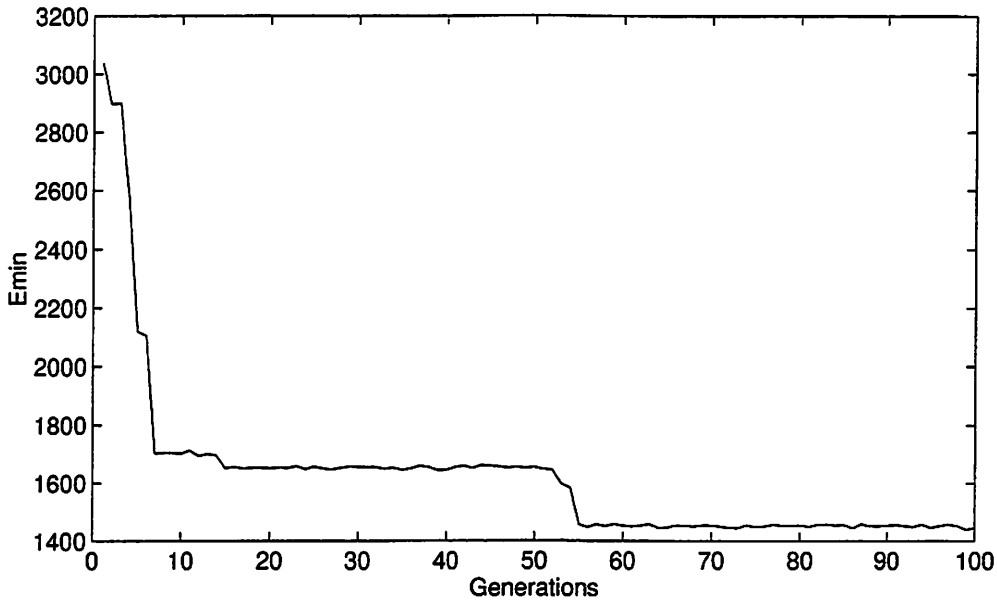


Figure 7.1: FIRE edge detection using GA and FCNN. (a) Input face image of a Chinese girl. (b) A noisy training example of edge detecting. (c) Output of FCNN after 100 generations. (d) Evolution process.

One should notice that the contour of the whole face which is almost diffused by noises in Fig. 7.1(b) is perfectly recovered by the FCNN and GA as shown in Fig. 7.1(c). And also the contours of the two eyes and the mouth are significantly recovered and filled into closed curves while those in the example in Fig. 7.1(b) are broken lines. Maybe from the crisp or classical image processing point of view, the learned result in Fig. 7.1(c) is a "terrible" one since there exits a huge error from the original examples. But from the human expert's point of view (i.e., from our cognitive point of view), the result in Fig. 7.1(c) is much better than its original example. In the first view, this kind of improvement is unbelievable because our common sense is that any learning algorithm of an artificial neural network (ANN) is an approximation to its supervisor examples. The usual example that an ANN learning simulation can give is that use a perfect crisp algorithm to generate some input-output examples and then use an ANN to learn the known crisp algorithm from the input-output pairs. In this view, the trained ANN should be not better than the crisp algorithm. Why the FCNN in the simulation performs better than its supervisor examples? In this case, the improvement come from the *structure* of the FCNN. When we come back to Sec.5.3 and Fig.5.1, we can find that the human expert's intuition (or experience, knowledge) for the concept of "edge" has been embedded into the FCNN structure as shown in Eq.(5.13). So, by using this example, we remind the ANN people that structure is also a kind of very important aspect of ANN besides learning.

While the learning ability of our brain is over emphasized these days, we should also remember that our brain has a unique structure which cost the nature billions of years to evolve. So, the structure of our brain is also a kind of knowledge. How this kind of structural knowledge become useful? The answer is learning. Come back to the example shown in Fig. 7.1, one can see that in the evolution of learning, the edge extracting knowledge embedded

in FCNN structure is gradually correct the errors existed in the supervisor examples. Our simulation had been shown that any distortion existed in the knowledge structure as shown in Eq.(5.13) or Fig.5.1 gave a much worse result.

From this example, one can also find the evidence that FCNN is a high-level CNN structure which can embedded the human expert's knowledge in a very efficient way and performs some intelligent behaviors.

Of course, if we can get the perfect examples, the conventional CNN may be a better choice to learn from these examples. In fact, an excellent example of DTCNN learning algorithm of edge detection had been proposed in [122] long time ago. But the question is how the conventional CNN performs when only ill-conditioned examples are available. Without the prior knowledge of the task embedded in its structure, the CNN will be puzzled and wander in the problem space and be settled down to an arbitrary local minimum in the vicinity of its initial condition.

Why don't use both the ill-conditioned examples and the prior knowledge (usually represented by a set of rules or linguistic statements) to train our model? If we can use this method, we can let our model use its "knowledge" to judge whether an example is good or bad. And then big weights are automatically assigned to the good examples and small weights are automatically assigned to the bad ones. Structural representation of knowledge plays a very important role in human intelligence. One can see this by thinking that a monkey who is supposed to be trained for a couple of thousand years will still not wiser than a normal human individual. It's because the structure evolution of human brain from the monkey brain cost billions of years. On the other hand, the learning skills of a monkey and a human individual may be the same in the nature.

We keep emphasizing that there are two motivation of inventing FCNN: one is mathematical morphology and the other is embedding human knowledge into CNN structure. So, this chapter provides another result motivated by the latter one.

Chapter 8

Linguistic Flow in Discrete-Time FCNN and Its Stability

8.1 Introduction

In this chapter, a type-III fuzzy discrete-time cellular neural network(FDTCNN) structure is presented to implement local linguistic dynamic systems. The theoretical results of equilibrium points and stability of this FDTCNN are presented. The state transient process of this FDTCNN is analyzed by using fuzzy mathematics, Markov chain and graph theory. An efficient algorithm is presented to check the existence of globally stable equilibrium point. Examples in a two-cell FDTCNN are used to demonstrate theoretical results. Finally, designing examples of a 1-D FDTCNN are presented to simulate the traffic flow of a highway system. Computer simulation results are given.

Type-II FCNN's are mostly studied and understood[350, 351, 352, 353, 356, 355] because it has crisp synaptic weights. In the near future, we believe that the first generation of VLSI implementation of FCNN will be type-II FCNN because type-II FCNN is the simplest structure in the FCNN universe. The type-II FCNN is a universal paradigm for implementing mathematical morphology operators[351, 352] while the conventional CNN structures can only give some wrong results with uncontrollable errors[360].

The most parts of type-I, -III and -IV FCNN still remain unexploited. We study architectures and theory of type-III FCNN in this chapter. Unlike our previous works where FCNN's were studied in the framework of VLSI implementation, we study type-III FCNN in a systematic framework. In a VLSI implementing framework, we define the information flow as a "signal". However, in a type-III FCNN, we define the information flow as a "linguistic variable(or, flow)". It is very necessary to exchange linguistic variables between cells when each cell represents a complex system (e.g., a human individual). In a type-III FCNN, synaptic weights are used to model structures and characteristics of linguistic flows between cells, which exist in an out world called *circumstance*. In a word, synaptic weights are fuzzy models of the relationship between cells and that between cells and circumstance.

8.2 Structures of fuzzy discrete-time cellular neural networks

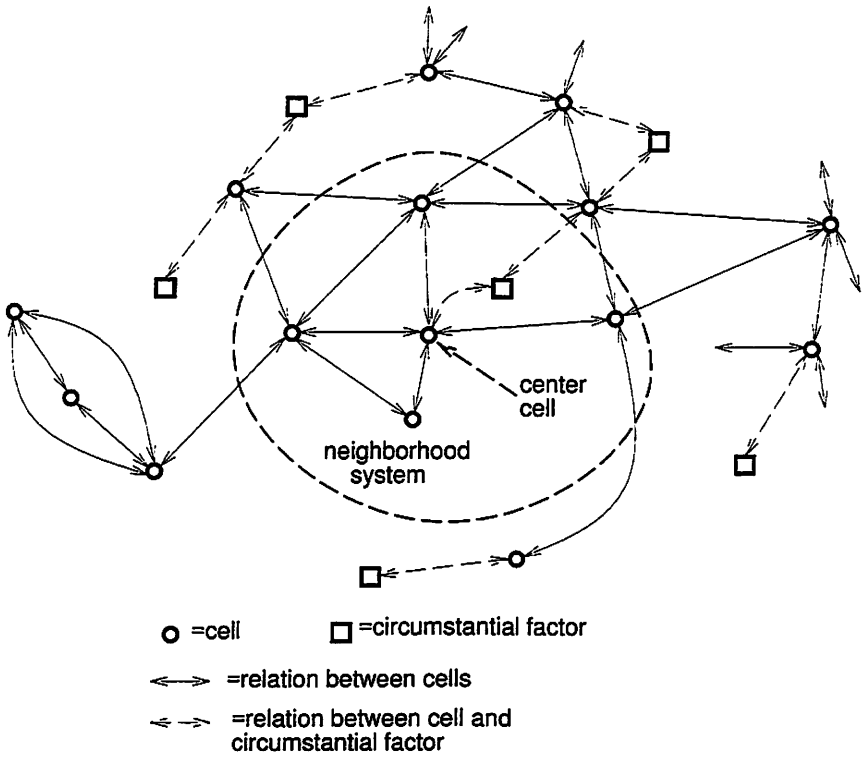


Figure 8.1: A type-III FCNN is used to model the linguistic relationship between cells and the relationship between cells and circumstance.

Fig.8.1 shows the relationship between cells and that between cells and circumstance. Every cell exchanges linguistic flows with all the cells within its neighborhood system. A cell can also exchange linguistic flows with its circumstance. We use a *circumstantial factor* to denote a local circumstance unit which can exchange linguistic flows with cells. For example, if a cell denotes a driver, then circumstantial factors may be signs along a highway. Since a driver can only see signs that immediately before it or behind it, we find that circumstantial factors also share local properties. By using linguistic dynamics to model the system shown in Fig.8.1, we need to consider cell dynamics and circumstance dynamics. We use "inputs" to denote all the things related to linguistic flows between cells and circumstantial factors. We use "state-variables" to denote all the things related to linguistic flows between cells. Then an $M \times N$ 2-dimensional type-III FDTCNN is given by

$$x_{ij}(k+1) = \overset{C_{kl} \in N_r(ij)}{F} \Big(x_{kl}(k), u_{kl}(k) \Big), 1 \le i \le M, 1 \le j \le N \tag{8.1}$$

where $\overset{C_{kl} \in N_r(ij)}{F}(\cdot)$ denotes a linguistic dynamic system defined in *neighborhood system* $N_r(ij)$. $x_{ij}(k)$ denotes the *linguistic state* of cell C_{ij} at discrete moment k. $u_{ij}(k)$ denotes the *linguistic input* from a circumstantial factor (i,j) at discrete moment k. We should note that both $x_{ij}(k)$ and $u_{ij}(k)$ are linguistic variables. Although the type-III FCNN in Eq.(8.1) has the same form as that of a general conventional DTCNN, the information flows and structures are totally different. A simple form of Eq.(8.1) can be written as the following fuzzy relational equation:

$$x_{ij}(k+1) = X_{ij}(k) \circ R_{ij}, 1 \le i \le M, 1 \le j \le N \tag{8.2}$$

where $X_{ij}(k)$ is a $1 \times (2r+1)^2$ row vector which consists of all the possible linguistic states within $N_r(ij)$. R_{ij} is a fuzzy relation. This fuzzy equation is borrowed from models of fuzzy control systems[60, 389]. We can see that R_{ij} is something similar to feedback template used in conventional DTCNN. Since R_{ij} is a *fuzzy matrix*, the FDTCNN in Eq.(8.2) has fuzzy synaptic laws. If the FDTCNN in Eq.(8.2) is space-invariant, then it can be recast into

$$x_{ij}(k+1) = X_{ij}(k) \circ R', 1 \leq i \leq M, 1 \leq j \leq N \tag{8.3}$$

If we repack all state variables of an $M \times N$ FDTCNN into a $1 \times MN$ row vector, $X(k)$, then Eq.(8.3) can be recast into

$$X(k+1) = X(k) \circ R \tag{8.4}$$

In this case, we view an $M \times N$ FDTCNN as an MN order dynamic linguistic system. In this chapter, we call the matrix R a *relation matrix.*

8.3 Different ways to check global stability of type-III FDTCNN

Since a phenomenon can be interpreted and modeled under different frameworks, it is not surprising that usually the same problem can be analyzed using different mathematical tools. To study the global stability of type-III FDTCNN, we can use tools from fuzzy mathematics (Type-III FDTCNN is modeled as a fuzzy relational system.), Markov chain (The behavior of FDTCNN is modeled as a transient process from one linguistic state to others.) and graph theory(The special properties of relation matrices make corresponding directed graphs have some promising properties). In this section we present different theoretical results and a practical algorithm for checking the global stability of FDTCNN. Since an FDTCNN may have different limit cycles as indicated in [323] and we should design our fuzzy systems by underlying theory rather than by trial and errors[324], the different methods presented here may benefit to the future work.

8.3.1 Methods from fuzzy mathematics

If only finite number of linguistic sets are needed to denote all possible state-variables, then we can use finite number of unit row vectors to encode all of the state-variables. Suppose that there exist m *referential fuzzy sets*[222, 246, 95] X_1, X_2, ..., X_m defined in the space X. We use a $1 \times m$ unit row vector, $\tilde{X}_i$, to encode any a fuzzy set X_i by making the i^{th} element one and the others zeros, e.g., $\tilde{X}_1 = (1, 0, 0, ..., 0)$ denotes X_1, $\tilde{X}_2 = (0, 1, 0, ..., 0)$ denotes X_2, etc.. Then Eq.(8.4) can be rewritten as

$$\tilde{X}(k+1) = \tilde{X}(k) \circ \tilde{R} \tag{8.5}$$

where $\tilde{R}$ is called as a *referential relation matrix.*

Definition 16

(Linguistic equilibrium point) If a linguistic state $\tilde{X}_e$ satisfies

$$\tilde{X}_e = \tilde{X}_e \circ \tilde{R} \tag{8.6}$$

where $\tilde{R}$ is an $m \times m$ matrix whose rows are $1 \times m$ unit vectors, then $\tilde{X}_e$ is a linguistic equilibrium point of FDTCNN in Eq.(8.5).

Proposition 11

(Corollary, p.31, [95]) The fuzzy system

$$\tilde{X}(k+1) = \tilde{X}(k) \circ \tilde{R} \tag{8.7}$$

is stable at the linguistic equilibrium point $\tilde{X}_e$ for an arbitrary initial linguistic condition $\tilde{X}_0$ if there exists a positive integer N, when $n \geq N$, the row vectors of $\tilde{R}^n$ are unit vectors and

$$\tilde{R}^n \circ \tilde{X}_e^T = (1, 1, ..., 1)^T \tag{8.8}$$

where " ^{T}" denotes the transpose.

Theorem 22

(Global Equilibrium Point) The FDTCNN defined by Eq.(8.5) has a globally stable linguistic equilibrium point $\tilde{X}_e$ if there exists a positive integer N, when $n \geq N$

$$\tilde{R}^n = \begin{pmatrix} \tilde{X}_e \\ \tilde{X}_e \\ \cdot \\ \cdot \\ \cdot \\ \tilde{X}_e \end{pmatrix} \tag{8.9}$$

is satisfied.

Proof:

It is easy to see that

$$\tilde{R}^n \circ \tilde{X}_e^T = (1, 1, ..., 1)^T \tag{8.10}$$

Followed Proposition 11, we get the conclusion. □

Remark: Although this theorem gives a criterion for checking whether an FDTCNN has a global equilibrium point or not, it fails to point out what kind of structure of $\tilde{R}$ implies a global equilibrium point. In next subsections, we give constructive methods for choosing a proper R such that an excepted linguistic state can be a global equilibrium point. However, when m is not too big, this theorem gives a very straightforward and simple method to check whether an FDTCNN has a globally stable equilibrium point or not.

8.3.2 Methods from Markov chains

In this section, we study the existence of the longest periodic evolving process(limit cycle) of FDTCNN's. Without loss of generality, in this chapter, we always suppose that a finite FDTCNN only has $m < \infty$ possible linguistic states. This means that we view FDTCNN as a discrete state system[324] in a finite linguistic space. The longest period of this FDTCNN is m. This means that no matter what the initial linguistic state $\tilde{X}_0$ is, the FDTCNN will go through all possible linguistic states each for once and then come back to $\tilde{X}_0$ at the m^{th} iteration. Since this is the only way to go through all possible linguistic states, a minor revision can derive a criterion of existence of globally stable equilibrium point.

Since $\tilde{R}$ is an $m \times m$ matrix whose row vectors are unit row vector, Eq.(8.5) can be rewritten as a classical multiplication form

$$\tilde{X}(k+1) = \tilde{X}(k)\tilde{R} \tag{8.11}$$

Since

$$\begin{aligned} \tilde{r}_{ij} &\geq 0 \\ \sum_{j=1}^{m} \tilde{r}_{ij} &= 1, i = 1, 2, ..., m \end{aligned} \tag{8.12}$$

$\tilde{R}$ is a *Markov matrix*. Comparing the form of Eq.(8.11) with that of a Markov chain[158], we find that the linguistic evolving process of an FDTCNN can be modeled by a Markov chain. In this sense, the referential relation matrix can also be called as a *transition matrix*.

Definition 17

(Definition 2, page 50, [96]) An $n \times n$ matrix $A = \{a_{ik}\}$ is called *reducible* if the index set $1, 2, ..., n$ can be split into two complementary sets (without common indices) $i_1, i_2, ..., i_\mu$; $k_1, k_2, ..., k_\nu$ $(\mu + \nu = n)$ such that

$$a_{i_\alpha k_\beta} = 0 \quad (\alpha = 1, 2, ..., \mu; \beta = 1, 2, ..., \nu) \tag{8.13}$$

otherwise the matrix is called *irreducible*.

Definition 18

A matrix is *regular* if some power of it is positive. A matrix is not regular is *non-regular*.

Proposition 12

(Theorem 3.4.7, page 305, [221]) An irreducible non-regular transition matrix is periodic.

Remark: If the transition matrix of a linguistic evolution process with m linguistic states is periodic, then for any initial linguistic state $\tilde{X}_0$ the FDTCNN will come back to $\tilde{X}_0$ after m iterations.

Theorem 23

The FDTCNN is periodic if $\tilde{R}$ is irreducible.

Proof: The FDTCNN can only stay in a distinct linguistic state in each iteration. This means that each row vector of $\tilde{R}^n$, $n \geq 1$, can only have one nonzero entry which is 1. Hence, the referential relation matrix $\tilde{R}$ is non-regular. If $\tilde{R}$ is irreducible, it follows from Proposition 12 that the FDTCNN is periodic. □

Remark: Although Theorem 23 only gives conditions for an FDTCNN being periodic, this theorem can be easily used to construct an FDTCNN which has a desired global equilibrium point as will be presented in next subsection.

8.3.3 Methods from graph theory

The basic knowledge of graph theory can be found in a famous book by Prof. W.K. Chen[40].

Definition 19

The *directed graph* of an $m \times m$ referential relation matrix $\tilde{R} = (\tilde{r}_{ij})_{m\times m}$, Γ, is obtained by connecting m *nodes* $P_1, P_2, ..., P_m$ by a *directed edge* directed away P_i and directed toward P_j if $\tilde{r}_{ij} \neq 0$.

Definition 20

A directed graph Γ is *strongly connected* if for any two nodes P_i and P_j, there is a *directed path* from P_i to P_j.

Proposition 13

(Theorem 6.1.3, page 218, [208]) An $m \times m$ matrix $\tilde{R}$ is irreducible if and only if the directed graph of $\tilde{R}$ is strongly connected.

Theorem 24

If an $m \times m$ matrix $R^* = [r_i^*]$ is an irreducible matrix, and $r_i^*, i = 1, 2, ..., m$, are $1 \times m$ unit row vectors, the referential relation matrix $\tilde{R} = (\tilde{r}_i)$ is given by

$$\begin{aligned} \tilde{r}_i &= \tilde{r}_i^*, i \neq e, 1 \leq i, e \leq m \\ \tilde{r}_i &= \tilde{X}_e, i = e \end{aligned} \tag{8.14}$$

where $\tilde{X}_e$ denotes the unit row vector whose e^{th} entry is 1, then the FDTCNN has a global equilibrium point $\tilde{X}_e$. And the FDTCNN will get this global equilibrium point within m iterations.

Proof: The proof is straightforward. We use $P_i, 1 \leq i \leq m$, to denote the m linguistic states of the FDTCNN. Since R^* is an irreducible matrix, from Proposition 13 we know that its directed graph Γ^* is strongly connected. And since every row in R^* is a unit row vector, we know that for any two nodes P_i^* and P_j^* in Γ^*, they can not be directed toward the same node. Otherwise, there will exist no directed path between P_i^* and P_j^* which is contradicted to the fact that Γ^* is strongly connected. Without loss of generality Γ^* is illustrated in Fig.8.2(a). We denote the directed graph of R by Γ, then by Eq.(8.14) all the edges of Γ are the same as those in Γ^* except for the one from node P_e directed towards node P_{e+1}. This edge is replaced by a self-loop of node P_e as shown in Fig.8.2(b). Then all the initial states will be absorbed by node P_e within m iterations. □

Of course, the directed graph does not need to be strong connected. Since a tree does not contain any loop, we want to find how can a tree imply a globally stable equilibrium point.

Proposition 14

In the directed graph of the referential relation matrix $\tilde{R}$, there exists one and only one directed edge directed away a node $P_i, 1 \leq i \leq m$.

Proof: In matrix $\tilde{R} = [\tilde{r}_i]$, each row vector $\tilde{r}_i$ is a unit vector, i.e., only one element of $\tilde{r}_i$, say, $\tilde{r}_{ij}, 1 \leq j \leq m$, is nonzero ($\tilde{r}_{ij} = 1$). This means that for each $1 \leq i \leq m$, there exists one and only one directed edge directed away the node P_i. □

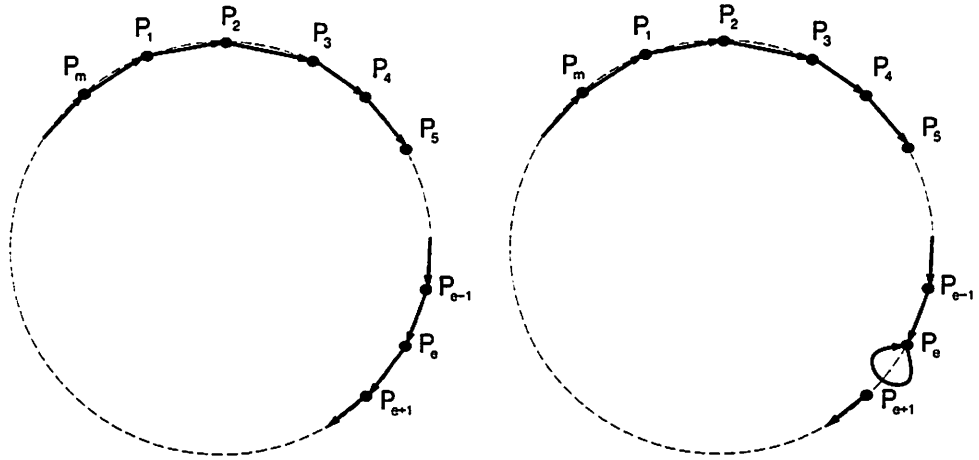


Figure 8.2: Illustrations of the directed graphs used in the proof of Theorem 24.

Theorem 25

If in the directed graph of a referential relation matrix $\tilde{R}$, there exists one and only one self-loop at node P_e, and by redirecting this self-loop to a datum node P_{00} (the **0** vector), we get a modified directed graph, if the modified directed graph is a tree, then $\tilde{X}_e$ is a global equilibrium point.

Proof: First, as shown in Fig.8.3, we label the datum node as P_{00}. Without loss of generality, we assume that the node P_{11} has a self-loop. Then we can break this self-loop by replacing it with a directed edge from P_{11} to P_{00}. Let set $\{P_{2i}\}$ denote nodes that connected to P_{11} by an edge. It follows from Proposition 14 that all the edges which connect node P_{11} and nodes in set $\{P_{2i}\}$ directed towards node P_{11}. This is illustrated in Fig.8.3. Similarly, let set $\{P_{3i}\}$ denote nodes that connect to a node in $\{P_{2i}\}$ by an edge, all the edges which connect nodes in set $\{P_{2i}\}$ and nodes in set $\{P_{3i}\}$ directed towards the nodes in $\{P_{2i}\}$. By using the same process we know that for a given k all the edges which connect nodes in set $\{P_{ki}\}$ and nodes in set $\{P_{(k-1)i}\}$ directed towards the nodes in $\{P_{(k-1)i}\}$. Since a finite FDTCNN can only have finite number of linguistic states, we can find a K such that when $k > K$, the set $\{P_{ki}\}$ is empty.

Since the modified directed graph is a tree, there exists no node which does not in the node set $P = \bigcup_{k=0}^{K}\{P_{ki}\}$. Hence, for any node $P_{ij} \in P$, there exists a directed path from P_{ij} to P_{00} containing less than $K+1$ branches. At the end, if we delete the datum node and redirect the edge from P_{11} to P_{00} as a self-loop of P_{11}, then any initial state will get node P_{11} within K iterations and stay there forever, i.e., P_{11} is the globally stable equilibrium point. □

Given a big number of cells, to draw the directed graph of a referential relation matrix $\tilde{R}$ is a tedious task. To try to figure out whether the modified directed graph is a tree is an even time consuming task. But the importance of Theorem 25 is that it gives a very clear guidance to construct a practical algorithm for checking if a given linguistic state is a globally stable equilibrium point or not.

Algorithm 1: Criterion for checking global equilibrium point

Assumption: In the referential relation matrix $\tilde{R} = (\tilde{r}_{ij})_{m\times m}$, there exists one and only one index e, $1 \le e \le m$, such that $\tilde{r}_{ee} = 1$.

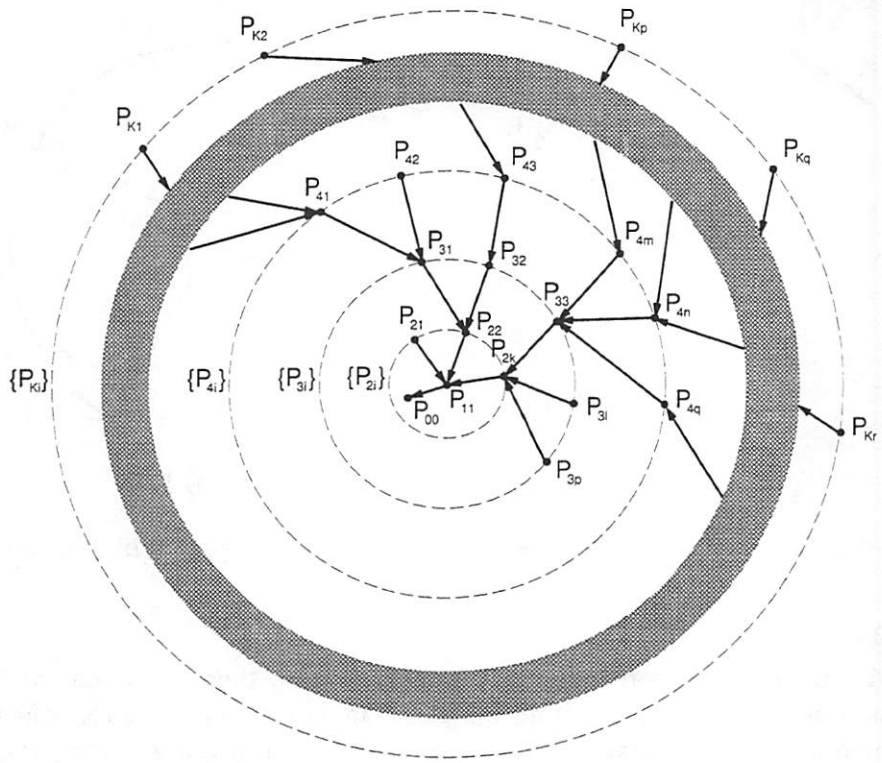


Figure 8.3: Illustration of the tree structure of the modified directed graph of referential relation matrix $\tilde{R}$. Only a part of the whole graph is shown in details.

Step 1: let $\tilde{r}_{ee} = 0$.

Step 2: scanning through all index i, $1 \leq i \leq m$. If $\tilde{r}_{ki} = 0$ for all $1 \leq k \leq m$, then set $\tilde{r}_{ik} = 0$ for all $1 \leq k \leq m$.

Step 3: scanning though all index i, $1 \leq i \leq m$. If $\tilde{r}_{ki} = 0$ for all $1 \leq k \leq m$, and there exists a $\tilde{r}_{ij} = 1$ for $1 \leq j \leq m$, then goto step 2. Else goto step 4.

Step 4: if $\tilde{R}$ is a $\mathbf{0}$ matrix, then $\tilde{X}_e$ is the global equilibrium point.

Remarks: The Step 1 is used to redirect the only self-loop to the datum node P_{00}. Step 1 transfers the directed graph of $\tilde{R}$ into the modified graph of $\tilde{R}$. From Theorem 25 we know that if the modified graph is a tree then $\tilde{X}_e$ is the global equilibrium point. To check if the modified graph is a tree or not, we can keep deleting "leaves" of the graph till no more leaves left. We define a leaf as an edge which has a *free node*. A free node is a node towards which no edge is directed. The leaf deleting is performed in Step 2. Since this process does not destroy any loops, what left after Step 3 are edges which are belong to some loops. If there is nothing left, then the modified directed graph is a tree which implies that $\tilde{X}_e$ is the global equilibrium point.

8.4 Examples of a two-cell FDTCNN

In this section we present some examples to show how the theoretical results in the previous section can be used to analysis the behavior of a two-cell FDTCNN.

8.4.1 Stable Cases

Here, we study an infinite 1-D type-III FDTCNN. We suppose that a cell C_i, $i \in Z$ determines its next state by checking its state and those of cells C_{i+1} and C_{i-1}. This is a simple linguistic model of traffic on an endless high way. We suppose that a driver changes its speed according to the speeds of the car before it, the car after it and its car. In this chapter, we suppose that a cell(car) can stay in three linguistic states S(small), M(meddle) and L(large).

Figure 8.4: A 2-cell 1-D FDTCNN.

Referential Number at k	$x_1(k)$	$x_2(k)$	$x_1(k+1)$	$x_2(k+1)$	Referential Number at $k+1$
100000000	S	S	S	M	010000000
010000000	S	M	S	M	010000000
001000000	S	L	S	M	010000000
000100000	M	S	S	M	010000000
000010000	M	M	M	S	000100000
000001000	M	L	S	L	001000000
000000100	L	S	M	S	000100000
000000010	L	M	S	S	100000000
000000001	L	L	S	L	001000000

Table 1: Dynamics of the FDTCNN in Fig.8.4 with globally stable output.

First, we study the case when only two cells with wrap-up boundary condition is used as shown in Fig.8.4. There exist $3^2 = 9$ referential fuzzy sets defined in $\{C_1, C_2\}$, the linguistic dynamics of the FDTCNN is given in Table 1. In Table 1, the first column gives the referential number of the linguistic state $(x_1(k), x_2(k))$, e.g., $(x_1(k), x_2(k)) = (S, S)$ is encoded by (100000000). Similarly, the last column gives the referential number of the linguistic state $(x_1(k+1), x_2(k+1))$. Let $X(k) = (x_1(k), x_2(k))$ where $x_1(k)$ and $x_2(k)$ are linguistic state of C_1 and C_2, respectively. Then we have

$$\tilde{R} = \begin{pmatrix} 010000000 \\ 010000000 \\ 010000000 \\ 010000000 \\ 000100000 \\ 001000000 \\ 000100000 \\ 100000000 \\ 001000000 \end{pmatrix}, \tilde{R}^2 = \begin{pmatrix} 010000000 \\ 010000000 \\ 010000000 \\ 010000000 \\ 010000000 \\ 010000000 \\ 010000000 \\ 010000000 \\ 010000000 \end{pmatrix} \tag{8.15}$$

Followed from Theorem 22 we know that $\tilde{X}_2 = (010000000)$, i.e., $x_1 = S$, $x_2 = M$, is

a globally stable equilibrium point of the DTCNN in Fig.8.4. Since (010000000) is the only entry in $\tilde{R}$ which satisfies $\tilde{r}_{ee} = 1$, by using Algorithm 1 we can also draw the same conclusion. After setting $\tilde{r}_{22} = 0$, we do the follows. Firstly, since columns $5 \sim 9$ of $\tilde{R}$ are zeros, we set the rows $5 \sim 9$ into zeros and get $\tilde{R}_1$. Secondly, in $\tilde{R}_1$ the columns 1, 3 and 4 are zeros, we can set the rows 1, 3 and 4 into zeros and get $\tilde{R}_2$ which is **0**. Thus $\tilde{X}_2$ is the global equilibrium point.

$$\tilde{R}_1 = \begin{pmatrix} 010000000 \\ 000000000 \\ 010000000 \\ 010000000 \\ 000000000 \\ 000000000 \\ 000000000 \\ 000000000 \\ 000000000 \end{pmatrix}, \tilde{R}_2 = \begin{pmatrix} 000000000 \\ 000000000 \\ 000000000 \\ 000000000 \\ 000000000 \\ 000000000 \\ 000000000 \\ 000000000 \\ 000000000 \end{pmatrix} \tag{8.16}$$

We also present the directed graph in Fig.8.5. From Fig.8.5 we can find that $P_{11} = P_2$, $\{P_{2i}\} = \{P_1, P_3, P_4\}$ and $\{P_{3i}\} = \{P_5, P_6, P_7, P_8, P_9\}$. We use these examples to show the inner connectedness between the three different methods presented in Section 3.

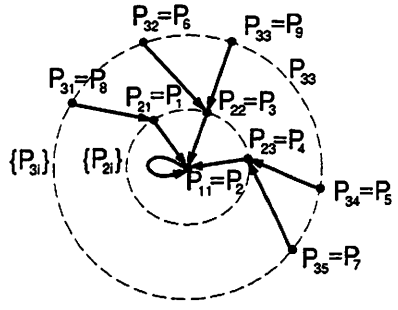


Figure 8.5: The directed graph of the referential relation matrix $\tilde{R}$ in Eq.(15).

The simulation results are shown in Fig.8.6 with different initial conditions. In Fig.8.6, the vertical axis denotes cell number while the horizontal axis denotes the iteration number (discrete time). One can see that the FDTCNN gets its global equilibrium point within 3 iterations with any initial condition.

8.4.2 Periodic cases and their stable revisions

We study the cases when the FDTCNN has a periodic output pattern. A local rule of periodic behavior is given in Table 2.

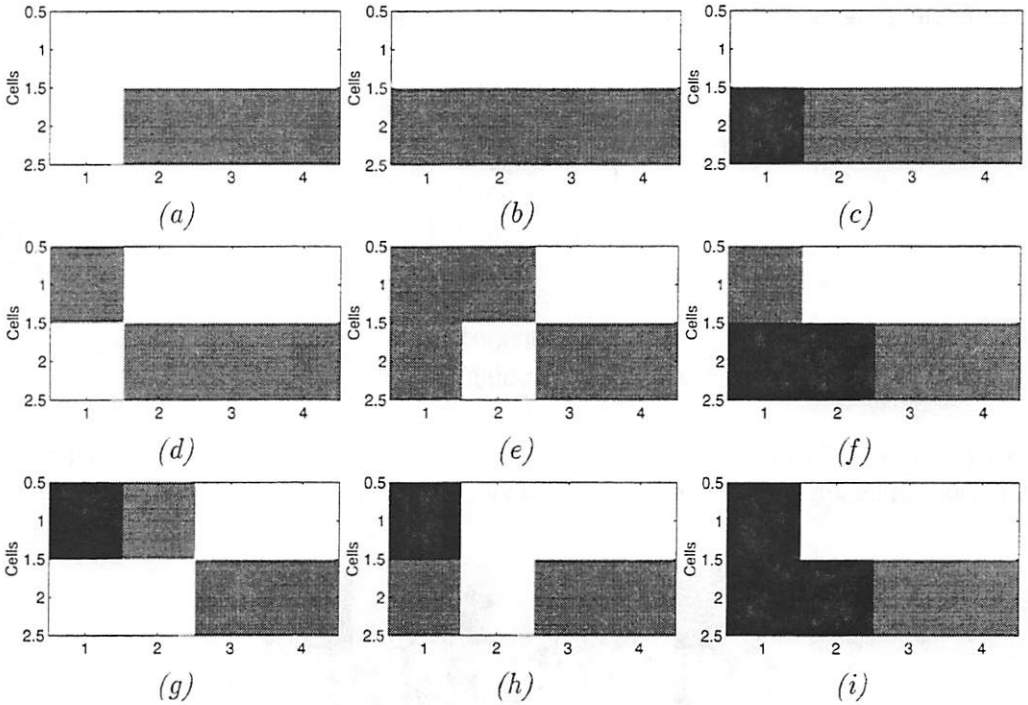


Figure 8.6: Simulation results of the two-cell FDTCNN with different initial conditions. Here, S, M and L are encoded by white, grey and black, respectively. (a) Initial condition $(x_1(1), x_2(1)) = (S, S)$. (b) Initial condition $(x_1(1), x_2(1)) = (S, M)$. (c) Initial condition $(x_1(1), x_2(1)) = (S, L)$.(d) Initial condition $(x_1(1), x_2(1)) = (M, S)$. (e) Initial condition $(x_1(1), x_2(1)) = (M, M)$. (f) Initial condition $(x_1(1), x_2(1)) = (M, L)$. (g) Initial condition $(x_1(1), x_2(1)) = (L, S)$. (h) Initial condition $(x_1(1), x_2(1)) = (L, M)$. (i) Initial condition $(x_1(1), x_2(1)) = (L, L)$.

Referential Number at k	$x_1(k)$	$x_2(k)$	$x_1(k+1)$	$x_2(k+1)$	Referential Number at $k+1$
100000000	S	S	S	M	010000000
010000000	S	M	S	L	001000000
001000000	S	L	M	S	000100000
000100000	M	S	M	M	000010000
000010000	M	M	M	L	000001000
000001000	M	L	L	S	000000100
000000100	L	S	L	M	000000010
000000010	L	M	L	L	000000001
000000001	L	L	S	S	100000000

Table 2: Dynamics of the FDTCNN in Fig.8.4 with periodic output patterns.

From Table 2 we can construct the referential relation matrix $\tilde{R}$ as

$$\tilde{R} = \begin{pmatrix} 010000000 \\ 001000000 \\ 000100000 \\ 000010000 \\ 000001000 \\ 000000100 \\ 000000010 \\ 000000001 \\ 100000000 \end{pmatrix} \tag{8.17}$$

which is irreducible. From Theorem 23 we know that the output will be oscillated with the longest period. The simulation results are shown in Fig.8.7. It is a period-9 pattern.

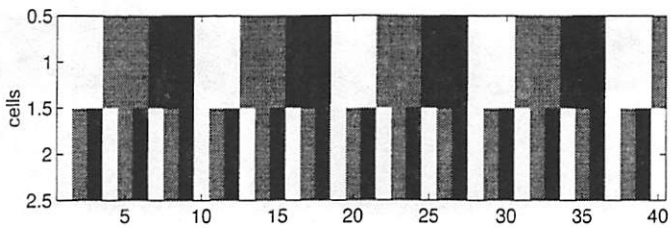


Figure 8.7: Periodic output of the two-cell FDTCNN. Here, S, M and L are encoded by white, grey and black, respectively.

Since an $m \times m$ matrix $\tilde{R}$ is irreducible if and only if the directed graph of $\tilde{R}$ is strongly connected, and in the directed graph there exists one and only one edge directed away a node, there exist 9! different period-9 output patterns in the FDTCNN shown in Fig.8.4.

We also show another stable example which is modified from the periodic result given by the referential relation matrix in Eq.(8.17). Suppose we want the linguistic state (MM) be the global stable equilibrium point, according to Theorem 24, we can change the fifth row of $\tilde{R}$ from (000001000) into (000010000) such that the new $\tilde{R}$ is given by

$$\tilde{R} = \begin{pmatrix} 010000000 \\ 001000000 \\ 000100000 \\ 000010000 \\ 000010000 \\ 000000100 \\ 000000010 \\ 000000001 \\ 100000000 \end{pmatrix} \tag{8.18}$$

Then all the initial linguistic patterns should get (MM) within 8 iterations. The simulation results is shown in Fig.8.8. Since the iteration 1 is the initial condition (ML), we can see that the FDTCNN get the designed global stable point in 8 iterations which is the longest evolving process before being absorbed by linguistic pattern(MM).

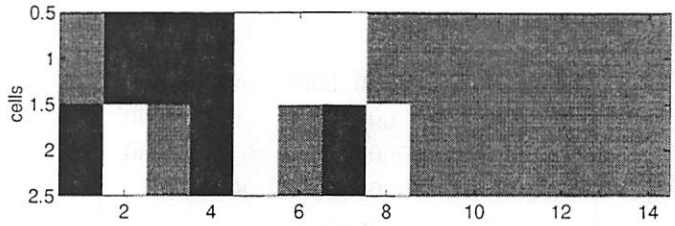


Figure 8.8: The simulation result of a globally stable output linguistic pattern. Here, S, M and L are encoded by white, grey and black, respectively.

8.5 Examples of a 1-D FDTCNN

In this section we will use an example to show how can we design local linguistic rules for achieving a certain global performance. The system we consider is a highway system. Every cell is a driver. The goal is to design a set of local rules such that a stable, safe and efficient traffic flow, say, all drivers have speed M, is the global equilibrium under any initial condition.

We suppose that a driver changes its speech according to the speeds of both the drivers who are immediately before and after him. We have to study a 1×3 neighborhood system which can be described by $3^3 = 27$ referential fuzzy sets defined on $\{C_{i-1}, C_i, C_{i+1}\}$, $-\infty < i < +\infty$. First, we design the fuzzy relation matrix R such that the linguistic pattern (MMM) is the global equilibrium point of all the 27 linguistic patterns.

One of this fuzzy relation is given by the following referential relation matrix

$$\tilde{R} = \begin{pmatrix}
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000000100 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000000000 & 010000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000000000 & 001000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000000000 & 000000100 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000001000 & 000000000 & 000000000 \\
000000000 & 000010000 & 000000000 \\
000000000 & 000010000 & 000000000
\end{pmatrix} \tag{8.19}$$

By using Algorithm 1, it is very easy to find that the vector (000000000000010000000000000), which corresponds to linguistic pattern (MMM), is the global equilibrium point of a 3-cell FDTCNN with wrap-up boundary condition. The corresponding linguistic local rules are given in Table 3.

x_{i-1}^k	x_i^k	x_{i+1}^k	x_i^{k+1}	x_{i-1}^k	x_i^k	x_{i+1}^k	x_i^{k+1}	x_{i-1}^k	x_i^k	x_{i+1}^k	x_i^{k+1}
S	S	S	M	M	S	S	M	L	S	S	M
S	S	M	M	M	S	M	M	L	S	M	M
S	S	L	L	M	S	L	M	L	S	L	L
S	M	S	M	M	M	S	M	L	M	S	M
S	M	M	M	M	M	M	M	L	M	M	M
S	M	L	M	M	M	L	M	L	M	L	M
S	L	S	S	M	L	S	M	L	L	S	M
S	L	M	M	M	L	M	M	L	L	M	M
S	L	L	S	M	L	L	M	L	L	L	M

Table 3: Local rules of the 1-D FDTCNN used to model a high way traffic flow.

In the above, we only give the conditions under which an arbitrary 1×3 pattern with wrap-up boundary approaches (MMM). A wrap-up 1×3 pattern, $(x_{i-1}(k), x_i(k), x_{i+1}(k))$

can be represented by

$$(\underbrace{x_{i-2}(k) = x_{i+1}(k)}_{\text{left boundary}}, \underbrace{x_{i-1}(k), x_i(k), x_{i+1}(k)}_{\text{center pattern}}, \underbrace{x_{i+2}(k) = x_{i-1}(k)}_{\text{right boundary}})$$

We then study the cases with the following boundary conditions:

$$(\underbrace{x_{i-2}(k) \neq x_{i+1}(k)}_{\text{left boundary}}, \underbrace{x_{i-1}(k), x_i(k), x_{i+1}(k)}_{\text{center pattern}}, \underbrace{x_{i+2}(k) \neq x_{i-1}(k)}_{\text{right boundary}})$$

In Table 3 we find that a 3-cell pattern which consists of at least one M approaches (MMM) pattern. Similar, those 3-cell patterns which change their center cells to $M's$ will also approach (MMM) patterns. Only the following four patterns, SSL, SLS, SLL and LSL may not result in (MMM). Since there doesn't exist periodic structure in these four patterns by using the local rules in Table 3, we conclude that all the cells in an infinite chain approaches M state with arbitrary initial conditions.

The simulation result is shown in Fig.8.9(a). We use 100 cells with a wrap-up boundary condition. One can see that all cells go to M states at the second iteration. However, for most of local rules, the FDTCNN approaches an unstable state. If we only change one local rule in Table 3 such that the local rules are given by Table 4, then the FDTCNN will output some periodic solutions. The simulation results are shown in Fig.8.9(b).

x_{i-1}^k	x_i^k	x_{i+1}^k	x_i^{k+1}	x_{i-1}^k	x_i^k	x_{i+1}^k	x_i^{k+1}	x_{i-1}^k	x_i^k	x_{i+1}^k	x_i^{k+1}
S	S	S	M	M	S	S	M	L	S	S	M
S	S	M	M	M	S	M	M	L	S	M	M
S	S	L	L	M	S	L	M	L	S	L	L
S	M	S	M	M	M	S	M	L	M	S	M
S	M	M	M	M	M	M	S	L	M	M	M
S	M	L	M	M	M	L	M	L	M	L	M
S	L	S	S	M	L	S	M	L	L	S	M
S	L	M	M	M	L	M	M	L	L	M	M
S	L	L	S	M	L	L	M	L	L	L	M

Table 4: Local rules of the 1-D FDTCNN which generate periodic output pattern.

8.6 Conclusions

Since restrictions from the state-of-the-art technology, fuzzy systems are always implemented in traditional platforms. This really makes some distortions of the cream of fuzzy mathematics. If conventional mathematics are used to model quantitative phenomena, then fuzzy mathematics are used to model qualitative ones. The similar relationship exists between conventional CNN and FCNN. Although someone may find that the type-II FCNN structures presented so far are similar to some conventional CNN structures with nonlinear synaptic laws, FCNN is not a subset of conventional CNN. Since the conventional CNN is developed in the conventional mathematic framework, it can not model qualitative phenomena. The nature of FCNN is *qualitative*. The conventional CNN and the FCNN view the same gray-scale input image in two different ways. For conventional CNN the gray value of a pixel is an isolated value, i.e., a point in an interval. On the other hand, FCNN views the

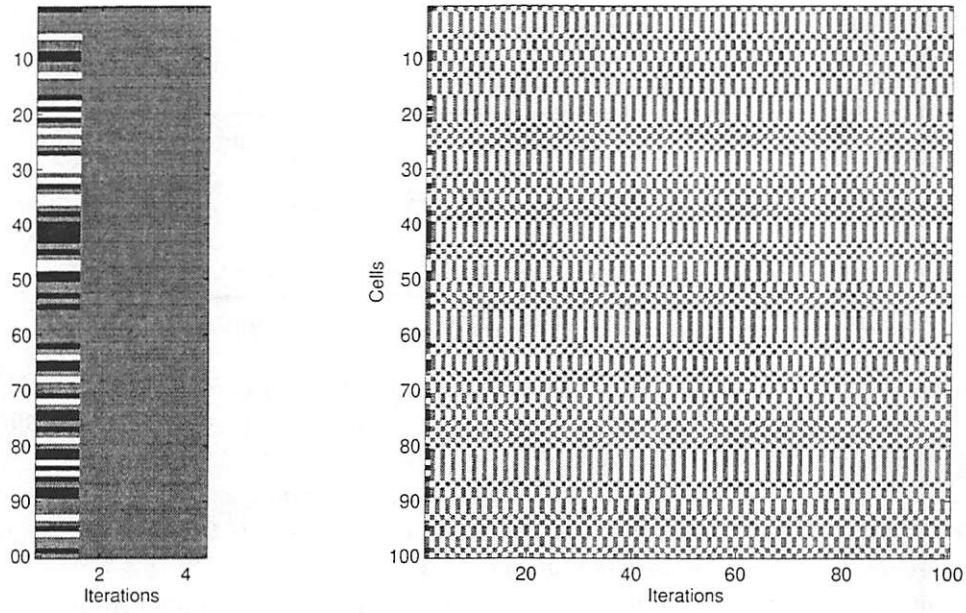


Figure 8.9: Simulation results with random initial condition and different linguistic local rules. Here, S, M and L are encoded by white, grey and black, respectively. (a) The local rules in Table 3 is used. (b) The local rules in Table 4 is used.

gray value of a pixel as a qualitative relation between the pixel and some datum characteristics. The type-III FDTCNN structure presented in this chapter shows this difference very clear because each cell in a type-III FDTCNN can process linguistic variables.

In this chapter, we presented some theoretical results of equilibrium points and stability of type-III FDTCNN's. We study these problems under three different mathematical frameworks, i.e., fuzzy mathematics, Markov chain and graph theory. An efficient algorithm for checking the global stability of type-III FDTCNN is presented. Some examples are given to show how to use the theoretic results. As applications, we used type-III FDTCNN to model the dynamics of the traffic flow in a highway.

Chapter 9

Applications of Discrete-Time FCNN

9.1 Implementing Nonlinear Fuzzy Operators for Image Processing

We present the structures of fuzzy discrete-time cellular neural networks(FDTCNN) by combining fuzzy neural network and conventional cellular neural network, and then FDTCNN are used to embed the nonlinear fuzzy operators for image processing. The fuzzy operators we discussed in this chapter are based on fuzzy IF-THEN-ELSE rule bases. The parallel computation mechanics of FDTCNN are possibly used to offset the computational complexity of fuzzy image processing problems.

The discrete-time dynamics in some cases may be superior to the continuous one as indicated in the first paper of discrete-time cellular neural networks(DTCNN)[120]. Therefore, we present applications of FDTCNN to image processing problems in this chapter. Since not all the people in the fuzzy field know the advantage of FCNN structures over the serial and conventional fuzzy image processing platform(e.g., some DSP chip or a serial fuzzy chip), we have to remind them that the future FCNN is supposed to be a parallel fuzzy processing chip. We try to persuade the people from fuzzy field who may be not familiar with the VLSI implementation that the purpose of the invention of FCNN is to speed up the existed fuzzy algorithms and give those fuzzy algorithms used in signal processing, in particular, in image processing a computational platform with parallel structures.

9.1.1 The structure of FDTCNN

Here we show how an FDTCNN structure can be derived from the framework of FNN[35]. A typical FNN structure consists of three layers. The first layer called *fuzzifier layer* is used to give the crisp inputs some fuzzy measurements. The nonlinearities in the neurons of this layer are some membership functions. The second layer called *fuzzy inference engine layer* is used to calculate the relationship between different fuzzy variables and (fuzzy/non-fuzzy) weights. The fuzzy computations are embedded into the nonlinearities of this layer. The third layer called *defuzzifier layer* is used to give some crisp forms of outputs.

Since the consideration of easy VLSI implementation and the local connectedness of CNN, a typical structure of a conventional CNN[47, 46] consists of three sub-layers: an *input function sub-layer*, a *cell dynamics sub-layer* and an *output function sub-layer*. In

a conventional CNN, the input function sublayer is usually used to normalize the sensed signal into a suitable range for VLSI based calculation. The cell dynamics sub-layer can be used to calculate operations defined by inputs, weights and outputs within a neighborhood system. The output function can be used to define the range of output and the dynamics of cell, etc.

Then it is very easy to find the FCNN structure by combining those of FNN and CNN. We embed the fuzzifier layer of FNN into the input function sub-layer of FCNN such that we use different membership functions as the input functions. We embed the fuzzy inference engine layer of FNN into the cell dynamics of FCNN. Finally, we embed the nonlinearities of the defuzzifier layer of FNN into the output function sub-layer of FCNN.

We then focus on FDTCNN. To give this kind of combination a quantitative description, we list all the equations of FNN and FDTCNN. By comparing these equations, one can easily see that FDTCNN is a kind of locally connected FNN with a planar structure. The corresponding equations describing an FNN are given by:

1. Fuzzifier layer (Layer 1)

$$y_i^1 = \mu_i^1(u_i), i = 1, 2, ..., k. \tag{9.1}$$

where u_i and y_i^1 are the i-th input and its corresponding fuzzifier value, respectively. $\mu_i^1(\cdot)$ is a membership function of the fuzzy variable represented by the i-th neuron in this layer.

2. Fuzzy inference engine layer (Layer 2)

$$\tilde{y}_i^2 = F_i(y_1^1, ..., y_k^1), i = 1, 2, ..., l. \tag{9.2}$$

where $F_i(\cdot)$ may be a fuzzy inference process, e.g., a set of fuzzy IF-THEN rules. It can also be some simple fuzzy operations, e.g., fuzzy logical operations. $\tilde{y}_i^2$ is the output of the i-th neuron in Layer 2. Here, we use "~" over a character to denote something related to fuzzy variable, e.g., BIG and $SMALL$.

3. Defuzzifier layer (Layer 3)

$$y_i^3 = D_i(\tilde{y}_1^2, ..., \tilde{y}_l^2), i = 1, 2, ..., m. \tag{9.3}$$

where y_i^3 is the output of the i-th neuron in layer 3. $D_i(\cdot)$ may be any defuzzifying operations, e.g., center of gravity and mean of maximum.

The corresponding FDTCNN can be given by the following equations:

1. Input function sub-layer (=fuzzifier layer)

$$u_{ij}(t) = \overset{\mu_{ij}^u}{C_{kl} \in N_r(ij)} \left(\{E_{kl}(t)\} \right) \tag{9.4}$$

where $E_{kl}(t)$ is the detected signal, e.g., the output of a camera. t denotes discrete-time iteration. $\overset{\mu_{ij}^u}{C_{kl} \in N_r(ij)}(\cdot)$ is the membership function of the fuzzy variable embedded in cell C_{ij}, it is used by feed-forward synaptic law. $\{\}$ denotes a set.

2. Cell dynamics sub-layer (= fuzzy inference engine layer)

$$\tilde{x}_{ij}(t) = \overset{F_{ij}}{C_{kl} \in N_r(ij)} \left(\{u_{kl}(t)\}, \{\mu_{kl}^y(y_{kl}(t))\} \right) \tag{9.5}$$

where $\overset{F_{ij}}{C_{kl} \in N_r(ij)}(\cdot)$ denotes a fuzzy inference process in $N_r(ij)$. $\mu_{kl}^y(\cdot)$ is the membership function of the fuzzy variable embedded in cell C_{ij}, it's used by feedback synaptic law. $y_{kl}(t)$ is output which is given by:

3. Output function sub-layer (=defuzzifier layer)

$$y_{ij}(t) = \overset{D_{ij}}{C_{kl}\in N_r(ij)} \left(\{\tilde{x}_{kl}(t-1)\}\right) \tag{9.6}$$

where $\overset{D_{ij}}{C_{kl}\in N_r(ij)}(\cdot)$ denotes a defuzzifier function defined in $N_r(ij)$.

9.1.2 Embedded fuzzy IF-THEN-ELSE rules into FDTCNN

Fuzzy local image operations had been developed in these years as a new branch of image processing[276, 277, 275, 279, 278] because there exist different kinds of uncertainties in image processing and image understanding. Some simple local fuzzy operators such as fuzzy shrinking and fuzzy expanding may be considered as a kind of mathematical morphological operations and can be readily implemented by type-II FCNN's[350, 351, 352, 355]. Here, we give the fuzzy IF-THEN rule based image operator a type-II FDTCNN implementation.

We then in the situation to show how a fuzzy IF-THEN-ELSE rule can be embedded into an FDTCNN structure. One simple fuzzy operator is given by the following fuzzy rule:

Rule One
IF $(\{E_{kl}\}^1$ is $A^1)$ AND $(\{E_{kl}\}^2$ is $A^2)$ ... and $(\{E_{kl}\}^M$ is $A^M)$, THEN $(y_{ij}$ is $B_1(x; c_1, w_1))$,
ELSE $(y_{ij}$ is $B_0(x; c_0, w_0))$
$C_{kl} \in N_r(ij)$

where $B_0(x; c_0, w_0)$ and $B_1(x; c_1, w_1)$ are two triangular-shaped fuzzy sets, which are defined by[30]

$$B(x; c, w) = \begin{cases} \frac{w-|x-c|}{w}, & c-w < x \le c+w \\ 0, & \text{else} \end{cases} \tag{9.7}$$

$A^p, p = 1, 2, ..., M$ are M fuzzy variables. y_{kl} is a crisp output of Rule One. Then we can use the following FDTCNN to implement Rule One.

1. Input sub-layer for implementing $(\{E_{kl}\}^p$ is $A^p)$, $p = 1, 2, ...M$:

$$u_{ij}^p(t) = \overset{\mu_{A^p}}{C_{kl}\in N_r(ij)} \left(\{E_{kl}\}^p\right), p = 1, 2, ...M \tag{9.8}$$

2. Cell dynamics sub-layer for implementing IF parts:

$$x_{ij}(t) = \min_{p=1,2,...,M}\{u_{ij}^p\} \tag{9.9}$$

By adopting correlation-product inference[166], we get the third sub-layer:

3. Output sub-layer for implementing THEN-ELSE part:

$$y_{ij}(t+1) = \frac{(1-x_{ij}(t))c_0w_0 + x_{ij}(t)c_1w_1}{(1-x_{ij}(t))w_0 + x_{ij}(t)w_1} \tag{9.10}$$

Then for purpose of comparison, the above FDTCNN structure can be summarized into a form which is similar to that of a conventional CNN as:

1. State-equation

$$x_{ij}(t) = \min_{C_{kl}\in N_r(ij)}\{\mu_{A^1}(\{E_{kl}(t)\}^1), \mu_{A^2}(\{E_{kl}(t)\}^2)..., \mu_{A^M}(\{E_{kl}(t)\}^M)\} \tag{9.11}$$

2. Output-equation

$$y_{ij}(t) = f(x_{ij}(t-1)) = \frac{(1-x_{ij}(t-1))c_0w_0 + x_{ij}(t-1)c_1w_1}{(1-x_{ij}(t-1))w_0 + x_{ij}(t-1)w_1} \tag{9.12}$$

Of course, a simple set of fuzzy IF-THEN rules may be too simple to solve a practical problem. We usually need more that one set of IF-THEN rules in fuzzy image operator. For example, in [279] the authors use a fuzzy rule set which contains 32 IF-THEN rules and one ELSE rule. Usually we have to consider the following rule:

Rule Two
IF $(\{E_{kl}\}^{11}$ is $A^{11})$ AND $(\{E_{kl}\}^{12}$ is $A^{12})$ ... and $(\{E_{kl}\}^{1M_1}$ is $A^{1M_1})$, THEN $(y_{ij}$ is $B_1(x; c_1, w_1))$,
.
.
.
IF $(\{E_{kl}\}^{i1}$ is $A^{i1})$ AND $(\{E_{kl}\}^{i2}$ is $A^{i2})$ ... and $(\{E_{kl}\}^{iM_i}$ is $A^{iM_i})$, THEN $(y_{ij}$ is $B_i(x; c_i, w_i))$,
.
.
.
IF $(\{E_{kl}\}^{N1}$ is $A^{N1})$ AND $(\{E_{kl}\}^{N2}$ is $A^{N2})$ ... and $(\{E_{kl}\}^{NM_N}$ is $A^{NM_N})$, THEN $(y_{ij}$ is $B_N(x; c_N, w_N))$,
ELSE $(y_{ij}$ is $B_0(x; c_0, w_0))$,
$C_{kl} \in N_r(ij)$

where $B_p(x; c_p, w_p), p = 1, 2, ..., N$ are N triangular-shaped fuzzy sets as defined in Eq.(9.7). $\{E_{kl}\}^{pq}, p = 1, 2, ..., N$, $q = 1, 2, ..., M_p$ are fuzzy variables. This rule base consists of N IF-THEN rules and one ELSE rule. Similar to the implementation of Rule One, the IF part of the p-th IF-THEN rule can be implemented by the following FDTCNN:

$$\begin{aligned} x_{ij}^p(t) &= \min_{C_{kl} \in N_r(ij)} \{\mu_{A^{p1}}(\{E_{kl}(t)\}^{p1}), \mu_{A^{p2}}(\{E_{kl}(t)\}^{p2}), ..., \mu_{A^{pM_p}}(\{E_{kl}(t)\}^{pM_p})\}, \\ & p = 1, 2, ..., N \end{aligned} \tag{9.13}$$

Therefore, we need p layers of FDTCNN to implement p IF-parts. The ELSE-rule is implemented by a special layer

$$x_{ij}^0(t) = \min_{p=1,2,...,N} \{1 - x_{ij}^p(t)\} \tag{9.14}$$

Then the whole rule base is finished by a common output function sub-layer

$$y_{ij}(t) = \frac{\sum_{p=0}^{N} x_{ij}^p(t-1)c_pw_p}{\sum_{p=0}^{N} x_{ij}^p(t-1)w_p} \tag{9.15}$$

In conclusion, to implement Rule Two, we need $(N+1)$ layer FDTCNN and a common output function layer.

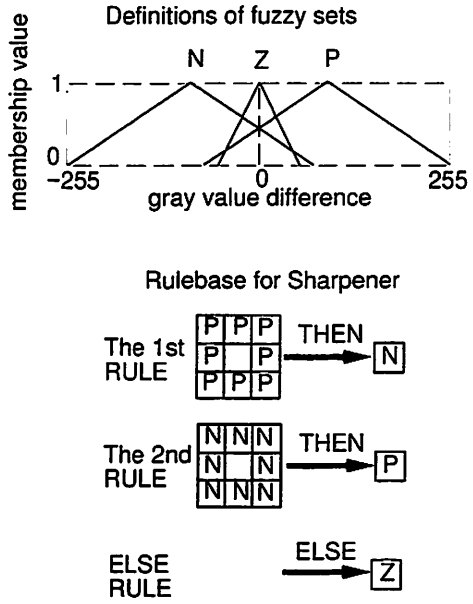


Figure 9.1: The rule-base for fuzzy sharpener presented in[278].

9.1.3 An example

We then show how a fuzzy IF-THEN-ELSE rule base for image processing can be embedded into an FDTCNN structure. Consider a basic fuzzy sharpener presented in [278]. The rule is shown in Fig.9.1 which is applied to a 256-greylevel digital image. It should be noted that all the inputs in the rules are gray-value difference between each pixel in the neighborhood system and the center pixel. This is the so called "relative in the antecedents" approach[278].

The rule base in Fig.9.1 consists of two IF-THEN rules and one ELSE rule. We can express the rule base by the following equivalent statements

IF ($E_{i-1,j-1}$ is P) AND ($E_{i-1,j}$ is P) AND ($E_{i-1,j+1}$ is P) AND ($E_{i,j-1}$ is P) AND ($E_{i,j+1}$ is P) AND ($E_{i+1,j-1}$ is P) AND ($E_{i+1,j}$ is P) AND ($E_{i+1,j+1}$ is P) THEN (y_{ij} is N),
IF ($E_{i-1,j-1}$ is N) AND ($E_{i-1,j}$ is N) AND ($E_{i-1,j+1}$ is N) AND ($E_{i,j-1}$ is N) AND ($E_{i,j+1}$ is N) AND ($E_{i+1,j-1}$ is N) AND ($E_{i+1,j}$ is N) AND ($E_{i+1,j+1}$ is N) THEN (y_{ij} is P),
ELSE (y_{ij} is Z)

To implement this fuzzy sharpener, we let E_{ij} denote the grey-value of pixel (i,j), then we have the following multi-layer FDTCNN structure. The State equation of FDTCNN #1 which is used to implement the 1st rule in Fig.9.1 is given by:

$$x_{ij}^1(t) = \min_{C_{kl} \in N_1(ij)/C_{ij}} \mu_P(E_{kl} - E_{ij}) \tag{9.16}$$

where $\mu_P(\cdot)$ is the membership function of fuzzy set P as shown in Fig.9.1.

The State equation of FDTCNN #2 which is used to implement the 2nd rule in Fig.9.1 is given by:

$$x_{ij}^2(t) = \min_{C_{kl} \in N_1(ij)/C_{ij}} \mu_N(E_{kl} - E_{ij}) \tag{9.17}$$

where $\mu_N(\cdot)$ is the membership function of fuzzy set N as shown in Fig.9.1.

The State equation of FDTCNN #0 which is used to implement the ELSE rule in Fig.9.1 is given by:

$$x_{ij}^0(t) = \min(1 - x_{ij}^1(t-1), 1 - x_{ij}^2(t-1)) \tag{9.18}$$

The above three layers share a single output layer as

$$y_{ij}(t) = f(x_{ij}^0(t-1), x_{ij}^1(t-1), x_{ij}^2(t-1)) = \frac{\sum_{p=0}^{2} x_{ij}^p(t-1)c_p w_p}{\sum_{p=0}^{2} x_{ij}^p(t-1)w_p} \tag{9.19}$$

where c_p and $w_p, p = 0, 1, 2$, are centers and widths of triangular-shaped fuzzy variables Z, N and P as shown in Fig.9.1, respectively.

Since in this section we only want to show how can we map the nonlinear fuzzy operators into FDTCNN, we do not want to discuss the advantages and the disadvantages of these kinds of image processing techniques, the interested reader are referred to [276, 277, 275, 279, 278] and references therein. All the disadvantages and the advantages of these kinds of methods are due to themselves but not FDTCNN(or general FCNN). The only thing FDTCNN can do here is to provide a computational platform to offset the possible computational complexity.

We presented the connections between the concepts of FCNN, FNN and CNN. In particular, we demonstrated by using FDTCNN structures that fuzzy properties were really embedded in FCNN structures. To demonstrate the potential applications of FCNN as computational array to image processing problems, we embedded a nonlinear fuzzy operator into a multi-layer FDTCNN structure.

9.2 Embedding Local Fuzzy Relation Equations

In this section we embed local fuzzy relation equations into fuzzy discrete-time cellular neural networks(FDTCNN). First, we introduce the FDTCNN from the VLSI implementation point of view. Then the multi-layer FDTCNN structure is used to embed local fuzzy relation equations which are defined in the neighborhood systems. An example is given to show how our method works. Computer simulation results are also given.

9.2.1 Introduction

It had been found that FCNN was the only existed high level CNN structure in the CNN universe. Here, the word "high level" means the ability of processing conceptual variables, e.g., linguistic variables. We have shown in different papers[351, 355] that FCNN can embedded the local fuzzy IF-THEN-ELSE rules in their structures. On the other hand, a subclass of type-II FCNN is found to be a universal and error free mathematical morphology paradigm for implementing grey scale morphological operation[351, 352, 355] while the conventional CNN based grey-scale morphological operation[395] always gives error results[360].

Here, we present a new fuzzy discrete-time CNN (FDTCNN) structure to embed local fuzzy relation equations. Fuzzy relation equations were first recognized and studied by Sanchez[282]. Fuzzy relation equations play an important role in areas such as fuzzy system analysis, design of fuzzy controller, decision-making processes, and fuzzy pattern recognition. Fuzzy relation equations are associated with the concept of composition of binary fuzzy relations, which includes both the set-relation composition and the relation-relation

composition. We only use the max-min composition because it has been studied extensively and has been utilized in numerous applications.

Embedding fuzzy relation equations in artificial neural networks(ANN) is not a new work, there exist lots of references[27, 131, 29, 196, 224]. Thanks to all these references, we can combine the concepts of FDTCNN and fuzzy relational neurocomputations in a very easy way.

9.2.2 Local fuzzy relation equation and its implementation

Let A_{ij} be a fuzzy set in $N_r(ij)$, where $N_r(ij)$ denotes the r-neighborhood system of cell C_{ij}, and $R_{ij}(N_r(ij), \{\phi_{ij}\})$ be a binary fuzzy relation in $N_r(ij) \times \{\phi_{ij}\}$ where set $\{\phi_{ij}\} = \{\phi_{ij}^1, ..., \phi_{ij}^m\}$. The set-relation composition of A_{ij} and R_{ij}, $A_{ij} \circ R_{ij}$, results in a fuzzy set in $\{y_{ij}\}$. Let us denote the resulting fuzzy set as B_{ij}. Then we have:

$$A_{ij} \circ R_{ij} = B_{ij} \tag{9.20}$$

The above equation is as a fuzzy relation equation. The membership function of B_{ij} is given by:

$$\begin{aligned} \mu_{B_{ij}}(\phi_{ij}^p) &= \mu_{A_{ij} \circ R_{ij}}(\phi_{ij}^p) \triangleq \max_{C_{kl} \in N_r(ij)} \min[\mu_{A_{ij}}(E_{kl}), \mu_R(E_{kl}, \phi_{ij}^p)] \\ & p = 1, ..., m. \end{aligned} \tag{9.21}$$

If we view R_{ij} as a local fuzzy system, A_{ij} as a local fuzzy input, and B_{ij} as a fuzzy output, then we can consider Eq.(9.20) as describing the characteristics of a fuzzy system via its fuzzy input-output relation.

From Eq.(9.21) one can see that we can use an m-layer FDTCNN to implement the fuzzy relation equation as shown in Eq.(9.20). The p-th layer FDTCNN is given by:

1. Input equation:

$$u_{kl} = \mu_{A_{ij}}(E_{kl}), C_{kl} \in N_r(ij) \tag{9.22}$$

2. Cell dynamics:

$$x_{ij}^p(t) = \max_{C_{kl} \in N_r(ij)} \min[u_{kl}(t), B^p(i, j; k, l)] \tag{9.23}$$

where

$$B^p(i, j; k, l) = \mu_R(E_{kl}, \phi_{ij}^p) \tag{9.24}$$

is synaptic weight.

3. Output equation

$$y_{ij}^p(t) = x_{ij}^p(t - 1) \tag{9.25}$$

One can see that

$$y_{ij}^p(t) = \mu_{B_{ij}}(\phi_{ij}^p) \tag{9.26}$$

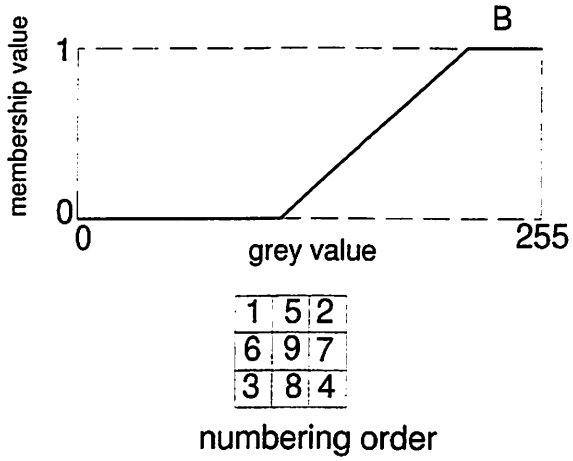


Figure 9.2: The fuzzy set B and the numbering order of cells in $N_1(ij)$.

9.2.3 An example

To show how the fuzzy relation equations can be embedded into a multi-layer FDTCNN we define the following fuzzy relation equation:

$$U \circ R = C \tag{9.27}$$

where $U = \{u_1, u_2, ..., u_8, u_9\}$ is numbered as that shown in Fig.(9.2)(lower part labeled "numbering order"), i.e., the following pattern

$$U = \begin{array}{c} \\ i-1 \\ i \\ i+1 \end{array} \begin{array}{c} \begin{array}{ccc} j-1 & j & j+1 \end{array} \\ \begin{pmatrix} u_1 & u_5 & u_2 \\ u_6 & u_9 & u_7 \\ u_3 & u_8 & u_4 \end{pmatrix} \end{array} \tag{9.28}$$

$C = \{c_1, c_2\}$. R is given by:

$$R = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \end{array} \begin{array}{c} \begin{array}{cc} c_1 & c_2 \end{array} \\ \begin{pmatrix} 0.5 & 0.75 \\ 0.5 & 0.75 \\ 0.5 & 0.75 \\ 0.5 & 0.75 \\ 1 & 0.5 \\ 1 & 0.5 \\ 1 & 0.5 \\ 1 & 0.5 \\ 0.2 & 1 \end{pmatrix} \end{array} \tag{9.29}$$

The first layer FDTCNN is used to implement the first column of R in Eq.(10.1), which is given by:

1. Input equation

$$u_{kl} = \mu_B(E_{kl}), C_{kl} \in N_1(ij) \tag{9.30}$$

2. Cell dynamics

$$x_{ij}^{(1)}(t) = \max_{C_{kl} \in N_1(ij)} \min[u_{kl}, B^{(1)}(i,j;k,l)] \tag{9.31}$$

where $B^{(1)}(i,j;k,l)$ is given by the following fuzzy set defined in $\mathbf{Z}^2(N_1(ij))$ grid:

$$\{B^{(1)}(i,j;k,l)\} = \begin{array}{c} \\ i-1 \\ i \\ i+1 \end{array} \begin{array}{c} \begin{array}{ccc} j-1 & j & j+1 \end{array} \\ \begin{pmatrix} 0.5 & 1 & 0.5 \\ 1 & 0.2 & 1 \\ 0.5 & 1 & 0.5 \end{pmatrix} \end{array} \tag{9.32}$$

which corresponds to the first column of R in Eq.(10.1).

3. Output equation:

$$y_{ij}^{(1)}(t) = x_{ij}^{(1)}(t-1) \tag{9.33}$$

The second layer is used to implement the second column in Eq.(10.1), which is given by:

1. Input equation

$$u_{kl} = \mu_B(E_{kl}), C_{kl} \in N_1(ij) \tag{9.34}$$

2. Cell dynamics

$$x_{ij}^{(2)}(t) = \max_{C_{kl} \in N_1(ij)} \min[u_{kl}, B^{(2)}(i,j;k,l)] \tag{9.35}$$

where $B^{(2)}(i,j;k,l)$ is given by the following fuzzy set defined in $\mathbf{Z}^2(N_1(ij))$ grid:

$$\{B^{(2)}(i,j;k,l)\} = \begin{array}{c} \\ i-1 \\ i \\ i+1 \end{array} \begin{array}{c} \begin{array}{ccc} j-1 & j & j+1 \end{array} \\ \begin{pmatrix} 0.75 & 0.5 & 0.75 \\ 0.5 & 1 & 0.5 \\ 0.75 & 0.5 & 0.75 \end{pmatrix} \end{array} \tag{9.36}$$

3. Output equation:

$$y_{ij}^{(2)}(t) = x_{ij}^{(2)}(t-1) \tag{9.37}$$

Fig.9.3 shows the simulation results. Fig.9.3(a) shows a grey-scale image of a Chinese girl of size 63×63 with 256 grey levels. In this simulation, we choose the fuzzy set B in Fig.9.2 as:

$$\mu_B(x) = \begin{cases} 0, & x < 50 \\ \frac{x-50}{150}, & 50 \le x < 200 \\ 1, & else \end{cases} \tag{9.38}$$

Fig.9.3(b) shows the output result of the first layer. The grey value of every pixel corresponds to a membership value. Fig.9.3(c) shows the output result of the second layer. Also, the grey value of every pixel corresponds to a membership value.

9.2.4 Conclusion

Since fuzzy relation equations can be viewed as a description of fuzzy systems which have fuzzy input and fuzzy output, we find that FDTCNN functions as a parallel implementation of this kind of fuzzy system. The immediate applications of this kind of fuzzy relational FDTCNN structure are image processing and pattern recognition.

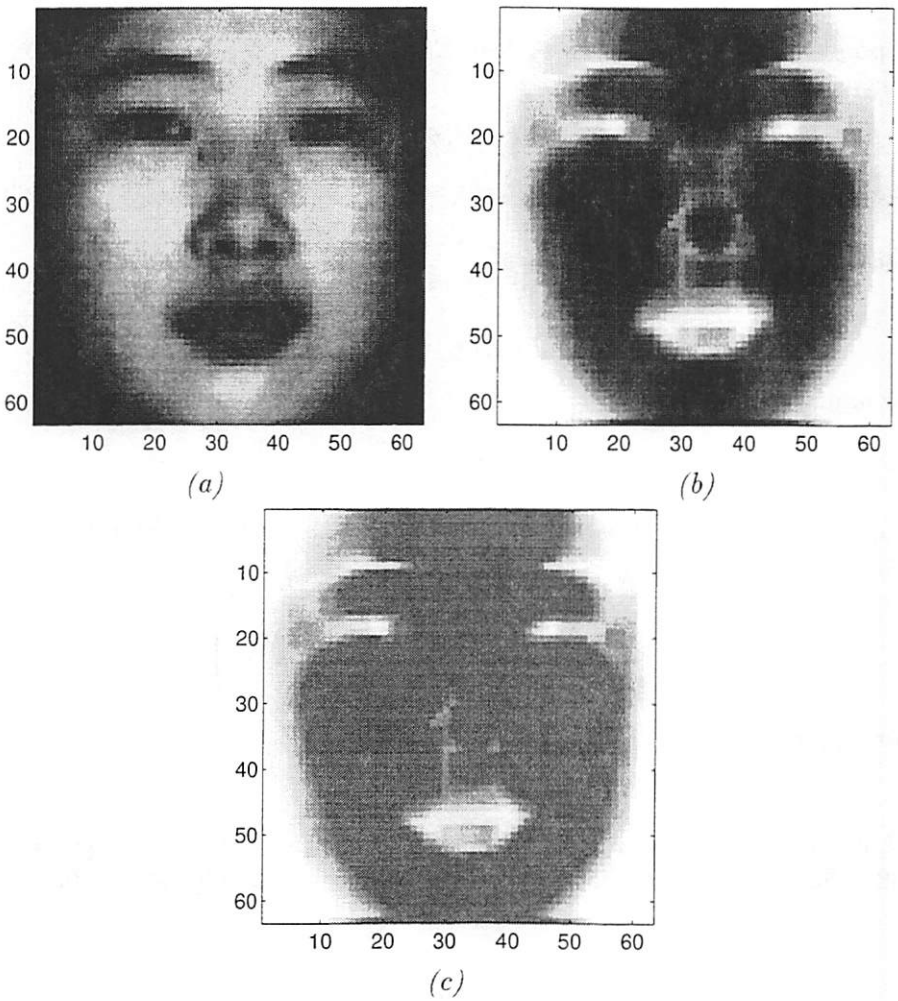


Figure 9.3: The simulation results. (a) The original image. (b) The output of the first FDTCNN layer. (b) The output of the second FDTCNN layer.

Chapter 10

Complexity in Discrete-Time FCNN—Fuzzy Spatial Dilemmas

We present a fuzzy discrete-time cellular neural network(FDTCNN) to implement fuzzy spatial Dilemmas. First, fuzzy spatial Dilemmas is presented. Then an FDTCNN structure is proposed to embed the fuzzy spatial Dilemmas using local fuzzy synaptic laws and fuzzy inputs. Some computer simulation results are given.

10.1 Introduction

Since the invention of cellular neural networks(CNN)[47, 46], the CNN concept has been widened for twice. The first one is the invention of discrete-time CNN(DTCNN)[120] which introduced the discrete-time dynamics into the conventional CNN framework. The second one is the invention of fuzzy CNN(FCNN)[350, 374] which introduced the fuzzy set theory into the CNN universe and split the history into two stages: the conventional stage and the systematic stage.

By embedded the max and min operations into the conventional CNN structure we can get the simplest FCNN structure(which is a kind of type-II FCNN[350, 351, 352, 356, 355]). It is very interested that the boundary between the conventional CNN and the FCNN is a fuzzy boundary. Sometimes we can lump some type-II FCNN which only using min/max operations into the conventional CNN framework because the image itself can be viewed either as 2D real signals or as a set of fuzzy singletons. Anyhow, it seems that the tendency of clearly defining the boundaries between conventional CNN and FCNN is a very trivial thing. On the other hand, since type-I, -II and -IV FCNN are totally different from type-II FCNN, they have very clear boundaries to conventional CNN. We should remind the reader that the type-II FCNN—although it is well known as FCNN—is only a very small part of FCNN and min/max operations are only the simplest fuzzy operations which can be used in FCNN. Someone may argue that only min/max is not enough to be fuzzy but no one can deny the existence of FCNN.

Here, we present a fuzzy DTCNN(FDTCNN) structure to implement fuzzy spatial Dilemmas. By the way, the fuzzy spatial Dilemma is also a new concept we generated from the conventional spatial Dilemma[200].

10.2 Fuzzy spatial Dilemmas

The conventional spatial Dilemma[200] is defined by a game played between two types of players: the defector(denoted by D) and the cooperator(denoted by C). The interaction between a cooperator and a defector is described by the following payoff matrix:

$$\begin{array}{c} \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cc} C & D \end{array} \\ \begin{pmatrix} 1 & 0 \\ 2 & 0 \end{pmatrix} \end{array} \tag{10.1}$$

In above matrix, we only show the payoff of a player. If two cooperators interact both receive 1 point. If a defector meet a cooperator, the defector receives the payoff 2 points and the cooperator 0. If two cooperators interact both receive 0 point.

The fuzzy generalization of the spatial Dilemmas are along two directions. One is fuzzifies the payoff. Since there may existed uncertainties in the payoff that a player may gain, we may describe the payoff as "high" or "low". The other fuzzifies the property of a player. Since a player may be a very complex system such as an animal or even a human individual, we cannot absolutely define which one is a defector or cooperator. A better way to describe the property of a player may assign a degree of being defector(or cooperator). By using fuzzy property of a player, we can say a play is low defection, meddle defection or high defection.

Here, we use two fuzzy descriptions for describing the uncertainties in both the payoff and the property of a player. We use the membership function $\mu_D(\cdot)$ to denote the degree of the player x to be a defector. We use a fuzzy set P to denote the payoff that a player obtained. Here, only the triangular-shaped fuzzy sets are used. We denote a triangular shaped fuzzy set by $A(x; c, w)$ which is given by:

$$A(x; c, w) = \begin{cases} \frac{w-|x-c|}{w}, & c - w < x \leq c + w \\ 0, & \text{else} \end{cases} \tag{10.2}$$

Then the mathematical operations between two triangular-shaped fuzzy set $A(x; c_a, w_a)$ and $B(x; c_b, w_b)$ are defined by[30]:

$$A(x; c_a, w_a) + B(x; c_b, w_b) = H(x; c_a + c_b, w_a + w_b) \tag{10.3}$$

$$\alpha A(x; c_a, w_a) = H(x; \alpha c_a, \alpha w_a) \tag{10.4}$$

where $\alpha > 0$ is a scalar.

In this case, the payoff map $T(x, y)$ for player x when it plays with another player y is a 2D fuzzy set $T : R \times R \mapsto [0, 1]$ defined by:

$$\mu_T(x, y) = 0.5\mu_D(x) - 0.5\mu_D(y) + 0.5 \tag{10.5}$$

We use $\mu_T(x, y)$ to measure the degree of player x to obtain a "high" payoff. However, since the payoff is also described by a fuzzy set, the payoff when player x play with player y should be a fuzzy set in respect with the fuzzy properties of x and y and the fuzzy representation of the payoff. There exist different fuzzy functions to combine the above two kinds of fuzziness, here we use following fuzzy function to denote the payoff, $P(x, y)$, for x when it plays with y:

$$P(x, y) = \mu_T(x, y)S(x; c_s, w_s) \tag{10.6}$$

where $S(x; c_s, w_s)$ is the standard fuzzy set which corresponds to the payoff for a defector when it meets a cooperator.

10.3 FDTCNN for implementing fuzzy spatial dilemmas

Then an FDTCNN is used to model the spatial fuzzy dilemmas. The dynamics of this FDTCNN is given by

1. Fuzzy state equation:

$$\tilde{x}_{ij}(t) = \sum_{C_{kl} \in N_1(ij)} \mu_T(y_{ij}(t), y_{kl}(t)) S(x; c_s, w_s) \tag{10.7}$$

Notice that the state-variable of the FDTCNN is a fuzzy set(usually a fuzzy number) instead of a signal(real number).

2. Output equation:

$$y_{ij}(t) = f_d(\tilde{x}_{ij}(t-1)) \tag{10.8}$$

where $f_d(\cdot)$ is a defuzzified function. Here, we choose $f_d(\cdot)$ as[282]:

$$f_d(\tilde{x}) = \frac{\int_{-\infty}^{\infty} y \mu_{\tilde{x}}(y) dy}{\int_{-\infty}^{\infty} \mu_{\tilde{x}}(y) dy} \tag{10.9}$$

If $\tilde{x}$ is a triangular-shaped fuzzy set, then

$$f_d(\tilde{x}(x; c, w)) = c + \frac{1}{3} w \tag{10.10}$$

At every iteration t, a center cell C_{ij} will be replaced by a cell $C_{kl} \in N_r(ij)$ whose output is the maximum in $N_r(ij)$.

In the following simulations, the wrap-up boundary condition is used. The gray-scale images with 256 gray levels are used to denote the evolving of spatial patterns. The degree of whiteness denote the membership value of a cell being a cooperator.

In Fig.10.1 we show the case when $S(x; c_s, w_s) = S(x; 5, 4)$. Fig.10.1(a) shows the initial conditions which is of size 63×63 cells. Fig.10.1(b) shows the snapshot of the 20 iterations. Fig.10.1(c) shows the snapshot of the 150 iterations. Finally, the pattern goes to a periodic 3 solution as shown in Figs.10.1(d), (e) and (f) of the 316, 317 and 318 iterations, respectively.

In Fig.10.2 we show the case when $S(x; c_s, w_s) = S(x; 10, 10)$. The initial condition is the same as that in Fig.10.1(a). Figs.10.2(a), (b), (c) and (d) show the snapshots the 455, 630, 845 and 1000 iterations, respectively. In this case, the evolution becomes chaotic. In this case since the payoff of cooperation becomes higher, the cells of high degree of cooperation increase. One very interested phenomenon is that a large collective of defector can not exist for a long time because the high payoff of cooperation will turn some of them into cooperator soon. This can not observe in Fig.10.1 where the payoff of cooperation is relatively low.

10.4 Conclusion

The FDTCNN we presented here functions as a kind of fuzzy cellular automata[1]. Although at present, conventional cellular automata (CA) are widely used in the simulation of such natural phenomena as fluid dynamic, diffusion, reaction-diffusion systems, populations, epidemics, etc., the applications of fuzzy cellular neural automata are very few. The lack of proper platform for studying FCA is a main reason for the silence of applications of FCA. FDTCNN is a very natural platform for FCA. Here we also present a new application of FCA to model of fuzzy spatial dilemmas.

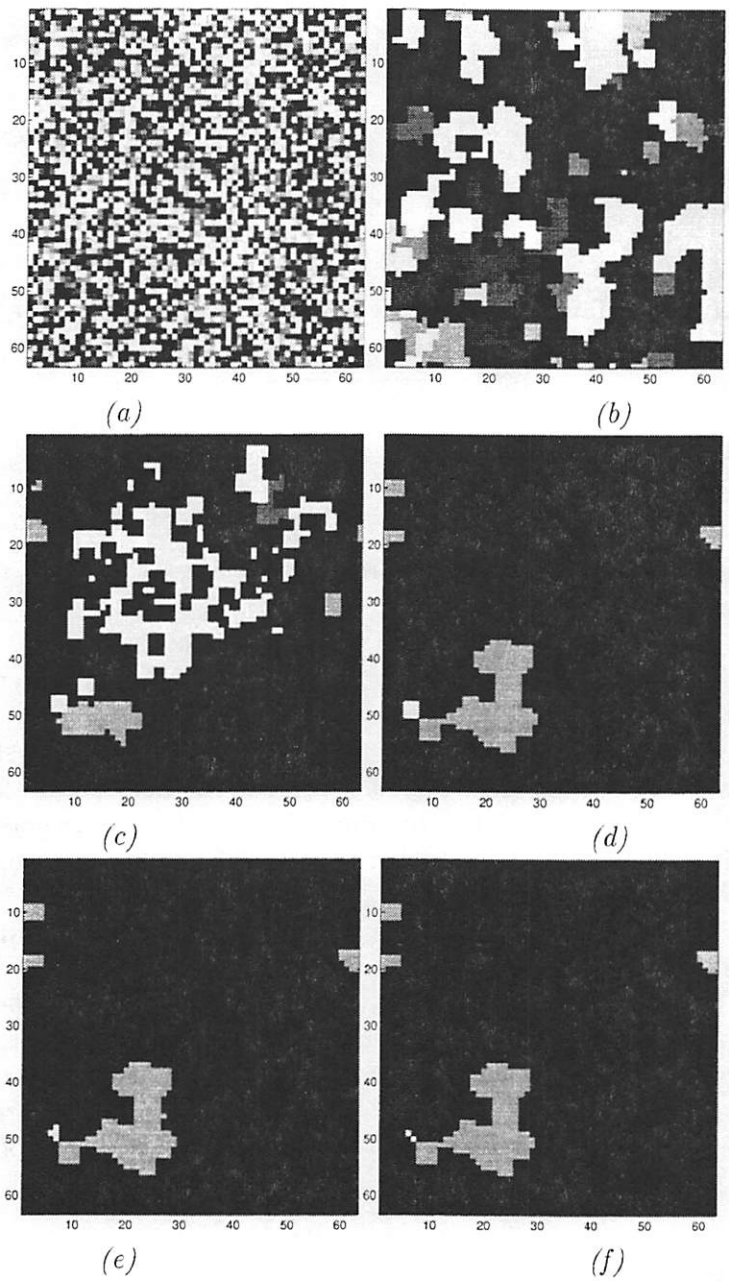


Figure 10.1: The evolving process of fuzzy spatial dilemmas with $S(x; c_s, w_s) = S(x; 5, 4)$. (a) Initial condition. (b) Output at $t = 20$. (c) Output at $t = 150$. (d) Output at $t = 316$. (e) Output at $t = 317$. (f) Output at $t = 318$.

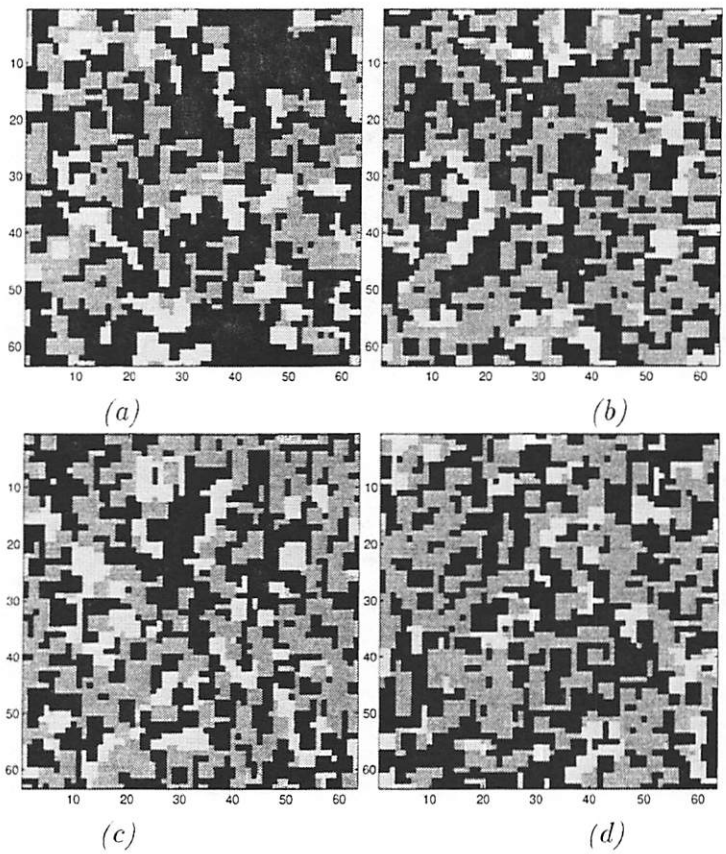


Figure 10.2: The evolving process of fuzzy spatial dilemmas with $S(x; c_s, w_s) = S(x; 10, 10)$. (a) Output at $t = 455$. (b) Output at $t = 630$. (c) Output at $t = 845$. (d) Output at $t = 1000$.

Chapter 11

Conclusions and the Future Work

Little or more, the invention of FCNN comes from AI. The idea of embedding fuzzy set theory into CNN framework is partially motivated by the inner connection of mathematical morphology and fuzzy logic. Since the substantial basis of FCNN is FCA, which is a model for modeling and coping with complexity, the invention of FCNN can also used to model the complexity from high-level local activity. Another motivative comes from the demand of employing the huge body of the knowledge from human experts. Encounter with the complexity of the outer and inner world, a human individual uses a systematic but general description to model and handle its behavior for the purpose of survival. If we only model the complexity itself, it is not very helpful to survive it. What fuzzy theory contributes to the science is to model the survival strategy of human individual directly. In FCNN, we model the systematic behavior of handling the complexity due to local activity but not the complexity itself.

However, FCNN is only a new branch in the CNN universe. When we encountered the excellent idea of CNN in early 1990's, its elegant structure attracted us immediately not only because it can be easily implemented by using VLSI but also because it denotes a high-level idea which is coincident with the development of modern AI, i.e., the *decentralization*. This field will grow up together with the growing up of the modern AI. From the stand point of AI, the existed CNN universe is something like that shown in Fig. 11.1. In this figure, we do not included the concept of CNNUM because it is not a CNN class, it is a platform of CNN. The CNNUM emphasizes the implementation and integration of different CNN structures. Although there have accumulated over 400 papers in this field since 1988, CNN is far from mature because we only find the tip of an iceberg of the CNN universe.

The implementation of CNN using different techniques deserves a further investigation because the simple structure of CNN provides us with lots of possibility of implementing it. Applications of CNN to signal processing, in particular, image processing need also further study because it seems that CNN is a very promising candidate for next generation of parallel image processing engine. On the other hand, the CNN paradigm can be used to animate lots of biologic, chemical and physical processes where the dynamics are governed by local coupling of simple units.

However, from Fig.11.1 one can see that the most parts of high-level CNN are unknown. In fact, the only high-level CNN we know so far is FCNN. We can image that the high-level CNN should include some paradigm which can be used to model the dynamics of human society when the non-physical factors such as: emotions, feelings and intuitions are used as the local couplings besides the physical factors such as: food, money, house, and work

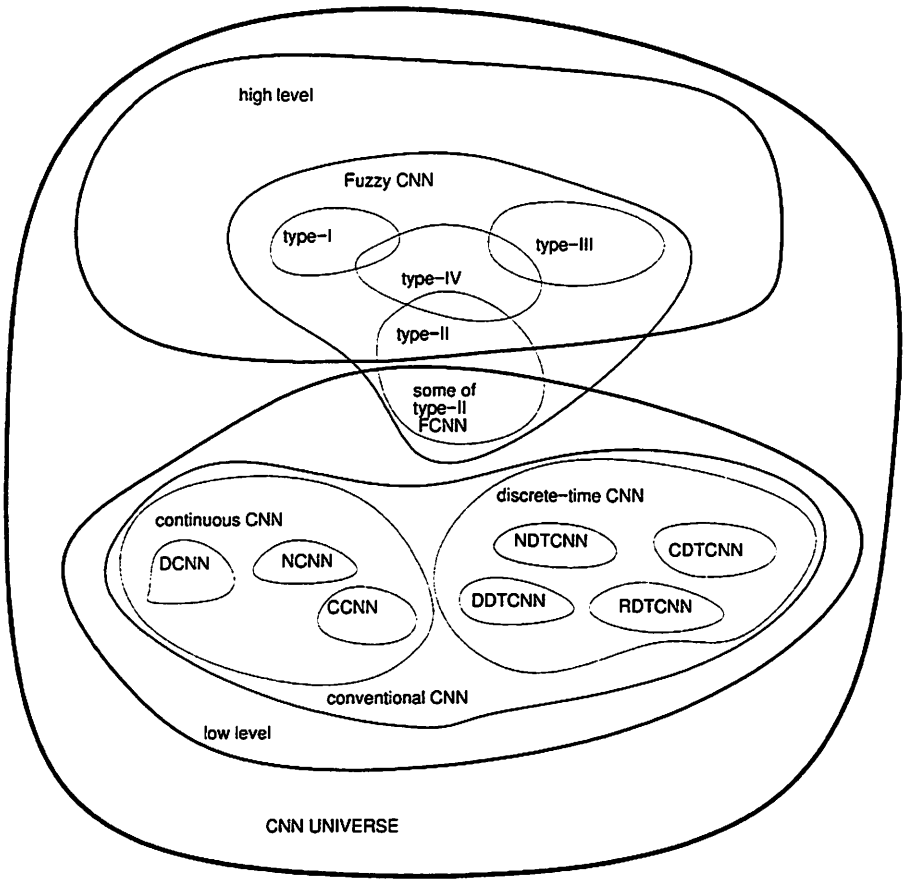


Figure 11.1: The map of the CNN universe.

opportunity. Although today we can not imagine how to embed the non-physical things into the structure of CNN, we know our human society uses them efficiently to organize itself for thousands of years. We believe that the future CNN model should be something like those we have predicted in Sec.2.1.

Of cause, we are always very carefully to avoid to give our readers the impression that we try to say CNN can do everything. On the contrary, we restrict the range of CNN in a class of problems which can be decomposed into local components. Since the top-down process, i.e., decomposing a global problem into local components, sometimes are very difficult, the bottom-up processes, i.e., by using relatively simple elements and local rules to generate some global behaviors, are also employed. So far, almost all the applications of CNN to signal processing and biological modeling employ top-down method. And some applications of CNN to pattern formation and spatio-temporal process modeling employ bottom-up method. However, when the bottom-up method is used, the emergent behavior of CNN may be very difficult to be interpreted. In general, it is not the problem of CNN itself, but the elementary problem of emergent computation.

Then, we should come back to FCNN. From this book one can see that the type-II FCNN is most studied and well-understood. In particular, we have presented a whole set of methods to exploit the world of some type-II FCNN which are used as computational low-level CNN. The overlap of this kind of FCNN to NCNN is not important, we do not want to address this trivial problem any more.

However, when fuzzy numbers flow though the type-II FCNN structure, the problem becomes very complicated. And this makes the FCNN totally different to conventional CNN. The research of this field is just opened. The potential applications are very attractive. One possible application may introduce the CNN into the high-level of AI, i.e., modeling and organizing the knowledge of human experts.

Type-I and -III FCNN give us more possibility and flexibility of modeling local coupling processes. Sometimes, type-I FCNN is more complicated than type-II FCNN because its fuzzy structure may introduce more complexity. On the other hand, from the examples in Sec.4.5.4 one can see that the potential application of type-I FCNN is out the range of linear signal processing. It is most possible to provide new methods to nonlinear signal processing. So far, we know very few of type-III FCNN because it is too complicated to analyze. The results in Chap. 8 shows that type-III FCNN can be used to model very complex systems where linguistic flows are used as state variables. The whole world of type-III and even type-IV FCNN is still waiting for being explored.

Bibliography

[1] A.I. Adamatzky. Hierarchy of fuzzy cellular automata. *Fuzzy Sets and Systems*, 62:167–174, 1994.

[2] N.N. Aizenberg and I.N. Aizenberg. Cnn-like networks based on multi-valued and universal binary neurons: learning and application to image processing. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 153–8, 1994.

[3] N.N. Aizenberg, I.N. Aizenberg, and G.A. Krivosheev. Cnn based on universal binary neurons: Learning algorithm with error-corection and application to impulsive-noise filtering on gray-scale images. In *Proceedings of the Fourth IEEE International Workshop on Cellular Neural Networks and Their Aplications, Seville, Spain, June 24-26*, pages 309–314, 1996.

[4] Taketoshi Ono[et al.](eds.). *Brain mechanisms of perception and memory : from neuron to behavior*. New York : Oxford University Press, 1993.

[5] G. Alefeld and J. Herzberger. *Introduction to Interval Computations*. Academic Press, New York, 1983.

[6] M. Anguita, F.J. Pelayo, F.J. Fernandez, and A. Prieto. A low-power analog implementation of cellular neural networks. In *From Natural to Artificial Neural Computation. International Workshop on Artificial Neural Networks. Proceedings.*, pages 736–43, 7-9 June 1995.

[7] M. Anguita, F.J. Pelayo, A. Prieto, and J. Ortega. Analog cmos implementation of a discrete time cnn with programmable cloning templates. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 40(3):215–18, March 1993.

[8] M. Anguita, F.J. Pelayo, E. Ros, D. Palomar, and A. Prieto. Vlsi implementations od cnns for image processing and vision tasks: Single and multiple chip approach. In *Proceedings of the Fourth IEEE International Workshop on Cellular Neural Networks and Their Aplications, Seville, Spain, June 24-26*, pages 479–484, 1996.

[9] M. Anguita, A. Prieto, F.J. Pelayo, J. Ortega, , et al. Cmos implementation of a cellular neural network with dynamically alterable cloning templates. In *Artificial Neural Networks. International Workshop IWANN '91 Proceedings*, pages 260–7, 1991.

[10] K. Arakawa. Median filter based on fuzzy rules and its application to image restoration. *Fuzzy Sets and Systems*, 77:3–13, 1996.

[11] C.P.S. Araujo and G. Ritter. Morphological neural networks and image algebra in artificial perception systems. *Image Algebra and Morphological Image Processing III, Proceedings of the SPIE - The International Society for Optical Engineering, San Diego, CA, USA,*, 1769(20-22):128–42, July 1992.

[12] P. Arena, S. Baglio, L. Fortuna, and G. Manganaro. Chua's circuit can be generated by cnn cells. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 42(2):123–5, Feb. 1995.

[13] P. Arena, S. Baglio, L. Fortuna, and G. Manganaro. Hyperchaos from cellular neural networks. *Electronics Letters*, 31(4):250–1, Feb. 1995.

[14] P. Arena, S. Baglio, L. Fortuna, and G. Manganaro. Generation of n-double scrolls via cellular neural networks. *International Journal of Circuit Theory and Applications*, 24(3):241–52, May-June 1996.

[15] T. Back and F. Kursawe. *Evolutionary algorithms for fuzzy logic: A brief overview*. pages 21–28. World Scientific, River Edge, NJ, 1995.

[16] I.A. Baktir and M.A. Tan. Analog cmos implementation of cellular neural networks. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 40(3):200–6, March 1993.

[17] I.A. Baktir and M.A. Tan. Analog cmos implementation of cellular neural networks. In *Computer and Information Sciences VI. Proceedings of the 1991 International Symposium*, pages 825–34, 30 Oct.-2 Nov. 1991.

[18] M. Balsi. Hardware supervised learning for cellular and hopfield neural networks. In *World Congress on Neural Networks-San Diego. 1994 International Neural Network Society Annual Meeting*, pages III/451–6, 1994.

[19] M. Balsi, I. Ciancaglioni, V. Cimagalli, and F. Galluzzi. Optoelectronic cellular neural networks based on amorphous silicon thin film technology. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 399–403, 18-21 Dec. 1994.

[20] S.H. Bang, B.J. Sheu, and T.H.-Y. Wu. Paralleled hardware annealing of cellular neural networks for optimal solutions. In *1994 IEEE International Conference on Neural Networks. IEEE World Congress on Computational Intelligence*, pages 2046–51 vol.4, 27 June-2 July 1994.

[21] R. Beccherelli, G. de Cesare, and F. Palma. Towards an hydrogenated amorphous silicon phototransistor cellular neural network. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 357–62, 18-21 Dec. 1994.

[22] T.W. Berger, B.J. Sheu, and R.H.-J. Tsai. Analog vlsi implementation of a nonlinear systems model of the hippocampal brain region. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 47–51, 18-21 Dec. 1994.

[23] B.E.Shi. Order statistic filtering with cellular neural networks. In *Proc. 3rd IEEE Int. Workshop on Cellular Neural Networks and Their Applications.*

[24] G.F.D. Betta, S. Graffi, Z.M. Kovacs, and G. Masetti. Cmos implementation of an analogically programmable cellular neural network. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 40(3):206–15, March 1993.

[25] Jr. Bey, P.P., D.J. Yonce, and R.W. Newcomb. Investigation of contrast enhancement by numerical methods for an optical cellular neural network. In *Proceedings of the 36th Midwest Symposium on Circuits and Systems*, pages 582–3, 1993.

[26] A. Blanco, M. Delgado, and I. Requena. Identification of fuzzy relational equations by fuzzy neural networks. *Fuzzy Sets and Systems*, 71:215–226, 1995.

[27] A. Blanco, M. Delgado, and I. Requena. Identification of fuzzy relational equations by fuzzy neural networks. *Fuzzy Sets and Systems*, 71:215–226, 1995.

[28] A. Blanco, M. Delgado, and I. Requena. Improved fuzzy neural networks for solving relational equations. *Fuzzy Sets and Systems*, 71:311–322, 1995.

[29] A. Blanco, M. Delgado, and I. Requena. Improved fuzzy neural networks for solving relational equations. *Fuzzy Sets and Systems*, 72:311–322, 1995.

[30] G. Bojadziev and M. Bojadziev. *Fuzzy Sets, Fuzzy Logic, Applications.* World Scientific, Singapore, 1995.

[31] P.J. Braspenning, F. Thuijsman, and A.J.M.M. Weijters(eds.). *Artificial neural networks : an introduction to ANN theory and practice.* New York : Springer, 1995.

[32] M. Brucoli, L. Carnimeo, and G. Grassi. Discrete-time cellular neural networks for associative memories: a new design method via iterative learning and forgetting algorithms. In *38th Midwest Symposium on Circuits and Systems. Proceedings*, pages 542–5, 1995.

[33] M. Brucoli, L. Carnimeo, and G. Grassi. Discrete-time cellular neural networks for associative memories with learning and forgetting capabilities. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 42(7):396–9, July 1995.

[34] M. Brucoli, L. Carnimeo, and G. Grassi. A global approach to the design of discrete-time cellular neural networks for associative memories. *International Journal of Circuit Theory and Applications*, 24(4):489–510, July-Aug. 1996.

[35] J.J. Buckley and Y. Hayashi. *Fuzzy neural networks*, pages Chapter 11, pp.233–249. McGraw-Hall, New York, 1996.

[36] G.C. Cardarilli, R. Lojacono, M. Salerno, and F. Sargeni. Vlsi implementation of a cellular neural network with programmable control operator. In *Proceedings of the 36th Midwest Symposium on Circuits and Systems*, pages 1089–92, 1993.

[37] G.C. Cardarilli, R. Lojacono, M. Salerno, and F. Sargeni. A vlsi implementation of programmable cellular neural networks. In *Artificial Neural Networks, 2. Proceedings of the 1992 International Conference (ICANN-92)*, pages 1491–4 vol.2, 4-7 Sept. 1992.

[38] G.C. Cardarilli and F. Sargeni. Very efficient vlsi implementation of cnn with discrete templates. *Electronics Letters*, 29(14):1286–7, 8 July 1995.

[39] R. Chellappa, C.L. Wilson, and S. Sirohey. Human and machine recognition of faces: A survey. *Proceedings of The IEEE*, 83:705–740, 1995.

[40] W.K. Chen. *Applied graph theory*, volume 13 of *North-Holland series in applied mathematics and mechanics.* North-Holland Pub. Co., Amsterdam, 1972.

[41] L.O. Chua. Cnn. ii. applications and vlsi circuit realizations. In *Proceedings of the 35th Midwest Symposium on Circuits and Systems*, pages 146–9 vol.1, 9-12 Aug.1992.

[42] L.O. Chua and L. Goras. Turing patterns in cellular neural networks. *International Journal of Electronics*, 79(6):719–36, Dec. 1995.

[43] L.O. Chua, M. Hasler, G.S. Moschytz, and J. Neirynck. Autonomous cellular neural networks: a unified paradigm for pattern formation and active wave propagation. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 42(10):559–77, Oct. 1995.

[44] L.O. Chua and T. Roska. The cnn paradigm. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(3):147–56, March 1993.

[45] L.O. Chua, T. Roska, T. Kozek, and A. Zarandy. Cnn universal chips crank up the computing power. *IEEE Circuits and Devices Magazine*, 12(4):18–28, July 1996.

[46] L.O. Chua and L. Yang. Cellular neural networks: Applications. *IEEE Transactions on Circuits and Systems*, 35(10):1273–1290, 10 1988.

[47] L.O. Chua and L. Yang. Cellular neural networks: Theory. *IEEE Transactions on Circuits and Systems*, 35(10):1257–1272, 10 1988.

[48] P.P. Civalleri and M. Gilli. Some stability properties of cnn's with delay. In *CNNA'92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 94–9, 1992.

[49] P.P. Civalleri and M. Gilli. Some dynamic phenomena in delayed cellular neural networks. *International Journal of Circuit Theory and Applications*, 22(2):77–105, Mar.-Apr. 1994.

[50] P.P. Civalleri, M. Gilli, and L. Pandolfi. On stability of cellular neural networks with delay. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(3):157–65, Mar. 1993.

[51] R.J. Clarke. *Digital Compression of Still Image and Video.* Academic Press, 1995.

[52] M. Coli, P. Palazzari, and R. Rughi. Design of dynamic evolution of discrete-time continuous-output cellular neural networks. In *ICANN '95. International Conference on Artificial Neural Networks. Neuronimes '95 Scientific Conference.*, pages 419–24 vol.2, 9-13 Oct. 1995.

[53] K.R. Crounse and L.O. Chua. Methods for image processing and pattern formation in cellular neural networks: a tutorial. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 42(10):583–601, Oct. 1995.

[54] K.R. Crounse and L.O. Chua. The cnn universal machine is as universal as a turing machine. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 43(4):353–5, Apr. 1996.

[55] K.R. Crounse, T. Roska, and L.O. Chua. Image halftoning with cellular neural networks. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 40(4):267–83, April 1993.

[56] J.M. Cruz and L.O. Chua. A cnn chip for connected component detection. *IEEE Transactions on Circuits and Systems*, 38(7):812–17, July 1991.

[57] J.M. Cruz and L.O. Chua. Application of cellular neural networks to model population dynamics. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 42(10):715–20, Oct. 1995.

[58] J.M. Cruz, L.O. Chua, and T. Roska. A fast, complex and efficient test implementation of the cnn universal machine. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 61–6, 18-21 Dec. 1994.

[59] M. Csapodi and T. Roska. Dynamic analogic cnn algorithms for a complex recognition task-a first step towards a bionic eyeglass. *International Journal of Circuit Theory and Applications*, 24(1):127–44, Jan.-Feb. 1996.

[60] E. Czogala and W. Pedrycz. Control problems in fyzzy systems. *Fuzzy Sets and Systems*, 7:257–273, 1982.

[61] G.F. Dalla Betta, S. Graffi, G. Masetti, and Z.M. Kovacs. Design of a cmos analog programmable cellular neural network. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 151–6, 14-16 Oct.1992.

[62] G.F. Dalla Betta, S. Graffi, G. Masetti, and Z.M. Kovacs. Design of a cmos analog programmable cellular neural network. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 151–6, 1992.

[63] J.L. Davidson. Simulated annealing and morphology neural networks. *Image Algebra and Morphological Image Processing III, Proceedings of the SPIE - The International Society for Optical Engineering, San Diego, CA, USA,*, 1769(20-22):119–27, July 1992.

[64] J.L. Davidson and F. Hummer. Morphology neural networks: an introduction with applications. *Circuits, Systems, and Signal Processing*, 12(2):177–210, July 1993.

[65] J.L. Davidson and G.X. Ritter. A theory of morphological neural networks. *Digital Optical Computing II, Proceedings of the SPIE - The International Society for Optical Engineering*, 1215(17-19):378–88, Jan. 1990.

[66] L. Davis. *Handbook of Genetic Algorithms.* Van Nostrand Reinhold, New York, 1991.

[67] G. Destri and P. Marenzoni. Cellular neural networks as a general massively parallel computational paradigm. *International Journal of Circuit Theory and Applications*, 24(3):397–407, May-June 1996.

[68] Joachim Diederich(Ed.). *Artificial neural networks : concept learning.* Los Alamitos, CA : IEEE Computer Society Press, 1990.

[69] M.-D. Doan, M. Glesner, R. Chakrabaty, M. Heidenreich. , and others. Realisation of a digital cellular neural network for image processing. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 85–90, 18-21 Dec. 1994.

[70] R. Dominguez-Castro, S. Espejo, A. Rodriguez-Vazquez, and R. Carmona. A cnn universal chip in cmos technology. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, 18-21 Dec. 1994.

[71] R. Dominguez-Castro, S. Espejo, A. Rodriguez-Vazquez, and R. Carmona. A cnn universal chip in cmos technology. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 91–6, 18-21 Dec. 1994.

[72] R. Dominguez-Castro. S. Espejo, A. Rodriguez-Vazquez, I. Garcia-Vargas, , and others. Sirena: a simulation environment for cnns. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 417–22, 18-21 Dec. 1994.

[73] R.D. Driver. *Ordinary and Delay Differential Equationa.* Springer-Verlag, 1977.

[74] D. Dubois and H. Prade. *Fuzzy Sets and Systems: Theory and Applications.* Academic Press, New York, 1980.

[75] S. Espeio, R. Carmona, R. Dominguez-Castro, and A. Rodriguez-Vazquez. A cnn universal chip in cmos technology. *International Journal of Circuit Theory and Applications*, 24(1):93–109, Jan.-Feb. 1996.

[76] S. Espeio, R. Carmona, R. Dominguez-Castro, and A. Rodriguez-Vazquez. A vlsi-oriented continuous-time cnn model. *International Journal of Circuit Theory and Applications*, 24(3):341–56, May-June 1996.

[77] S. Espejo, R. Dominguez-Castro, R. Carmona, and A. Rodriguez-Vazquez. A continuous-time cellular neural network chip for direction-selectable connected component detection with optical image acquisition. In *Proceedings of the Fourth International Conference on Microelectronics for Neural Networks and Fuzzy Systems*, pages 383–91, 26-28 Sept. 1994.

[78] S. Espejo, R. Dominguez-Castro, R. Carmona, and A. Rodriguez-Vazquez. Cellular neural network chips with optical image acquisition. In *1994 IEEE International Conference on Neural Networks. IEEE World Congress on Computational Intelligence*, pages 1877–82, 27 June-2 July 1994.

[79] S. Espejo, R. Dominguez-Castro, A. Rodriguez-Vazquez, and R. Carmona. Weight-control strategy for programmable cnn chips. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 405–10, 18-21 Dec. 1994.

[80] S. Espejo, A. Rodriguez-Vazquez, R. Dominguez-Castro, J.L. Huertas, , and others. Smart-pixel cellular neural networks in analog current-mode cmos technology. *IEEE Journal of Solid-State Circuits*, 29(8):895–905, Aug. 1994.

[81] S. Espejo, A. Rodriguez-Vazquez, and J.L. Huertas. Design and testing issues in current-mode cellular neural networks. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 169–74, 14-16 Oct.1992.

[82] S. Espejo, A. RodriguezVazquez, R. DominguezCastro. B. Linares. , and others. A model for vlsi implementation of cnn image processing chips using current-mode techniques. In *(Proceedings) 1993 IEEE International Symposium on Circuits and Systems.*. pages 970–3 vol.2, 3-6 May 1993.

[83] Doyne Farmer, Tommaso Toffoli, and Stephen Wolfram(Eds). *Cellular automata : proceedings of an interdisciplinary workshop.* New York : North-Holland Physics Pub., 1984.

[84] B. Faure and G. Mazare. A vlsi cellular array for the processing of back-propagation neural networks. In *Algorithms and Parallel VLSI Architectures. Lectures and Tutorials Presented at the International Workshop.*, pages 193–202, 10-16 June 1990.

[85] M.Lozano F.Herrera and J.L.Verdegay. *Generating fuzzy rules from examples using genetic algorithms*, pages 21–28. World Scientific, River Edge, NJ, 1995.

[86] M. Finocchiaro and R. Perfetti. Relation between template spectrum and stability of cellular neural networks with delay. *Electronics Letters*, 31(23):2024–6, Nov. 1995.

[87] Stephanie Forrest(ed.). *Emergent Computation.* Elesvier Science Publisher B.V., 1991.

[88] N. Fruehauf, L.O. Chua, and E. Lueder. Convergence of reciprocal time-discrete cellular neural networks with continuous nonlinearities. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 106–11, 14-16 Oct.1992.

[89] N. Fruehauf, E. Lueder, and G. Bader. Fourier optical realization of cellular neural networks. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 40(3):156–62, March 1993.

[90] N. Fruhauf and E. Luder. Realization of cnns by optical parallel processing with spatial light valves. In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 281–90, 16-19 Dec. 1990.

[91] M. Furukawa and T. Yamakawa. The design algorithms of membership functions for a fuzzy neuron. *Fuzzy Sets and Systems*, 71:329–343, 1995.

[92] Z. Galias. Designing discrete-time cellular neural networks for the evaluation of local boolean functions. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 23–8, 14-16 Oct.1992.

[93] Z. Galias. Designing cellular neural networks for the evaluation of local boolean functions. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 40(4):267–83, April 1993.

[94] Z. Galias and J.A. Nossek. Control of a real chaotic cellular neural network. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, page 345, 1994.

[95] Ju Gang and Chen Laijiu. Linguistic stability analysis of fuzzy closed loop control systems. *Fuzzy Sets and Systems*, 82(1):27–34, 26 Aug. 1996.

[96] F. R. Gantmacher. *The Theory of Matrices—Volume Two.* Chelsea Publishing Company, New York, 1960.

[97] M. Gilli. Strange attractors in delayed cellular neural networks. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(11):849–53, Nov. 1993.

[98] M. Gilli. Stability of cellular neural networks and delayed cellular neural networks with nonpositive templates and nonmonotonic output functions. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 41(8):518–28, Aug. 1994.

[99] M. Gilli. A spectral approach for chaos prediction in delayed cellular neural networks. *International Journal of Bifurcation and Chaos in Applied Sciences and Engineering*, 5(3):869–75, Jun. 1995.

[100] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, 1989.

[101] Eric Goles and Servet Martinez(Eds.). *Cellular automata, dynamical systems, and neural networks.* Kluwer Academic Publishers, 1994.

[102] J. Goutsias and D. Schonfeld. Image coding via morphological transformation: A general theorey. In *Proc. of IEEE Conf. Computer Vision and Pattern Recognition*, June, 1989.

[103] M. Grabish and M. Schmitt. Mathematical morphology, order filters and fuzzy logical. In *Proc. of 1995 IEEE Int. Conf. on Fuzzy Systs.*, pages 2103–2108, 1995.

[104] B. Gunsel and C. Guzelis. Supervised learning of smoothing parameters in image restoration by regularization under cellular neural networks framework. In *Proceedings. International Conference on Image Processing*, pages 470–3, 1995.

[105] Howard Gutowitz(Ed.). *Cellular automata : theory and experiment.* Cambridge, Mass. : MIT Press, 1991.

[106] C. Guzelis. Supervised learning of the steady-state outputs in generalized cellular networks. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 74–9, 1992.

[107] C. Guzelis. Chaotic cellular neural networks made of chua's circuits. *Journal of Circuits, Systems and Computers*, 3(2):603–12, June 1993.

[108] C. Guzelis and S. Karamahmut. Recurrent perceptron learning algorithm for completely stable cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 177–82, 1994.

[109] K. Halonen, V. Porra, T. Roska, and L. Chua. Programmable analog vlsi cnn chip with local digital logic. In *1991 IEEE International Symposium on Circuits and Systems*, pages 1291–4, 11-14 June 1991.

[110] K. Halonen, V. Porra, T. Roska, and L. Chua. Vlsi implementation of a reconfigurable cellular neural network containing local logic (cnnl). In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 206–15, 16-19 Dec. 1990.

[111] K. Halonen, V. Porra, T. Roska, and L. Chua. Vlsi implementation of a reconfigurable cellular neural network containing local logic (cnnl). In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 206–15, 16-19 Dec. 1990.

[112] K. Halonen, A. Radvany, and T. Roska. The control strategy of a dual (programmable analog-logical) cellular neural network chip. In *Proceedings of the 2nd International Conference on Microelectronics for Neural Networks*, page 251, 1991.

[113] K. Halonen and J. Vaananen. The non-idealities of the ic-realization and the stability of cnn-networks. In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 226–34, 16-19 Dec. 1990.

[114] K. Halonen, J. Vaananen, V. Porra, T. Roska, , et al. Vlsi-implementation of a programmable dual computing cellular neural network processor. In *Artificial Neural Networks. Proceedings of the 1991 International Conference.*, pages 1581–4, 1991.

[115] U.D. Hanebeck and G.K. Schmidt. Genetic optimization of fuzzy networks. *Fuzzy Sets and Systems*, 79:59–68, 1996.

[116] L.K. Hansen. Boltzmann learning of parameters in cellular neural networks. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 62–7, 1992.

[117] R.M. Haralick, S.R. Sternberg, and X. Zhuang. Image analysis using mathematical morphology. *IEEE Trans. Pattern Anal. Mach. Intell.*, PAMI-9:532–550, 1987.

[118] H. Harrer. Multiple layer discrete-time cellular neural networks using time-variant templates. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 40(3):191–9, March 1993.

[119] H. Harrer and J.A. Nossek. New test results of a 4 by 4 discrete-time cellular neural network chip. In *CNNA'92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 163–8, 14-16 Oct.1992.

[120] H. Harrer and J.A. Nossek. Discrete-time cellular neural networks. *International Journal of Circuit Theory and Applications*, 20(5):453–467, 10 1992.

[121] H. Harrer, J.A. Nossek, and R. Stelzl. An analog implementation of discrete-time cellular neural networks. *IEEE Transactions on Neural Networks*, 3(3):466–76, May 1992.

[122] H. Harrer, J.A. Nossek, and F. Zou. A learning algorithm for discrete-time cellular neural networks. In *Proc. of IJCNN'91,Singapore*, pages 717–722, 1991.

[123] H. Harrer, J.A. Nossek, and F. Zou. A learning algorithm for time-discrete cellular neural networks. In *1991 IEEE International Joint Conference on Neural Networks*, pages 717–22, 1991.

[124] H. Harrer, P.L. Venetianer, J.A. Nossek, T. Roska, , and others. Some examples of preprocessing analog images with discrete-time cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 201–6, 18-21 Dec.1994.

[125] Mohamad H. Hassoun. *Fundamentals of artificial neural networks*. Cambridge, Mass. : MIT Press, 1995.

[126] Y. Hayashi, J.J. Buckley, and E. Czogala. Fuzzy neural network with fuzzy signals and weights. *Int. J. Intelligent Syst.*, 8:527–537, 1993.

[127] C. He and A. Ushida. Iterative middle mapping learning algorithm for cellular neural networks. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E77-A(4):706–15, April 1994.

[128] Chen He and A. Ushida. A non-uniform discrete-time cellular neural network and its stability analysis. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E79-A(2):252–7, Feb. 1996.

[129] H.J.A.M. Heijmans. *Morphological Image Operators*. New York: Academic, 1992.

[130] V.Tryba H.Heider and E.Muhlenfeld. *Automatic design of fuzzy systems by genetic algorithms*, pages 21–28. World Scientific, River Edge, NJ, 1995.

[131] K. Hirota and W. Pedrycz. Solving fuzzy relational equations through logical filtering. *Fuzzy Sets and Systems*, 81:355–363, 1996.

[132] C.-Y. Ho, T.-K. Chu, and S. Mori. Discrete-time cellular neural network for thinning: A compound synthesis. In *(Proceedings) 1993 IEEE International Symposium on Circuits and Systems*, pages 2379–82 vol.4, 3-6 May 1993.

[133] C.-Y. Ho, D.-H. Yu, and S. Mori. Synthesis of discrete-time cellular neural networks for binary image processing. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E76-A(5):735–41, May 1993.

[134] Chun-Ying Ho and S. Mori. An optimized synthesis of a discrete-time cellular neural network for parallel thinning. *International Journal of Circuit Theory and Applications*, 22(5):363–75, Sept.-Oct. 1994.

[135] Xiang-Zhu Huang, Tao Yang, and Lin-Bao Yang. On stability of the time-variant delayed cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 18–21, 1994.

[136] David H. Hubel. *Eye. brain, and vision.* New York : Scientific American Library, 1988.

[137] J.L. Huertas, A. Rodriguez-Vasquez, and S. Espejo. Analog vlsi implementation of cellular neural networks. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 141–50, 14-16 Oct.1992.

[138] J.L. Huertas and A. Rueda. Testability issues in analog cellular neural networks. In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 172–6, 16-19 Dec.1990.

[139] Keng-Shen Hung, K.M. Curtis, and J.W. Orton. A programmable multi-function optoelectronic cellular neural network. In *IEE Colloquium on 'Integrated Imaging Sensors and Processing'*, pages 3/1–6, 5 Dec. 1994.

[140] K.S. Hung, K.M. Curtis, and J.W. Orton. Optoelectronic implementation of a multifunction cellular neural network. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 43(8):601–8, Aug. 1996.

[141] R.J. Marks II, S. Oh, P. Arabashahi, T. P. Caudell, J. J. Choi, and B. G. Song. Steepest descent adaption of min-max fuzzy if-then rules. In *Proc. IJCNN*, volume 3, pages 471–477. Beijing, China, 1992.

[142] R.J. Marks II(Ed.). *Fuzzy Logic Technology and applications.* IEEE, Piscataway, NJ, 1994.

[143] M. Ikegami and M. Tanaka. Moving image coding and decoding by dtcnn with 3-d templates. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 475–9, 18-21 Dec.1994.

[144] M. Ikegami and M. Tanaka. [image coding and decoding by discrete time cellular neural networks and its error evaluation]. *Transactions of the Institute of Electronics, Information and Communication Engineers A*, J77-A(7):954–64, July 1994.

[145] M. Ikegami and M. Tanaka. Image coding and decoding by discrete-time cellular neural networks. *Electronics and Communications in Japan, Part 3 (Fundamental Electronic Science)*, 78(3):32–43, March 1995.

[146] H. Ishibuchi. Neural networks that learn from fuzzy if-then rules. *IEEE Trans. on Fuzzy Systems*, 1(2):85–97, May 1993.

[147] B.K. Jang and R.T. Chin. A multiscaling approach based on morphological filtering. *IEEE Trans. Pattern. Anal. Machine Intell.*, 11(7), July 1989.

[148] B.K. Jang and R.T. Chin. Analysis of thinning algorithms using mathematical morphology. *IEEE Trans. Pattern. Anal. Machine Intell.*, 12(6), June 1990.

[149] J.R. Jang and C.T. Sun. Neuro-fuzzy modeling and control. *Proceedings of the IEEE*, 83(3):378–406, March 1995.

[150] S. Jankowski, A. Londei, A. Lozowski, and C. Mazur. Synchronization and control in a cellular neural network of chaotic units by local pinnings. *International Journal of Circuit Theory and Applications*, 24(3):275–81, May-June 1996.

[151] S. Jankowski, A. Londei, C. Mazur, and A. Lozowski. Synchronization phenomena in 2d chaotic cnn. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 339–44, 1994.

[152] S. Jankowski and R. Wanczuk. Cnn models of complex pattern formation in excitable media. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 333–8, 1994.

[153] Rhawn Joseph. *The naked neuron : evolution and the languages of the body and brain.* New York : Plenum, 1993.

[154] T. Kacprzak and K. Slot. Multiple-input ota based circuit for cellular neural network-implementation in vlsi cmos technology. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 157–62, 14-16 Oct.1992.

[155] A. Kandel. *Fuzzy Techniques in Pattern Recognition.* Wiley, New York, 1982.

[156] A. Kanfmann and M.M. Gupta. *Introduction to Fuzzy Arithmetic.* Van Nostrand Reinhold. New York, 1995.

[157] N.B. Karayiannis and A.N. Venetsanopoulos. *Artificial neural networks : learning algorithms, performance evaluation, and applications.* Boston : Kluwer Academic, 1993.

[158] J.G. Kemeny and J.L. Snell. *Finite Markov Chains.* D. Van Nostrand Co., Princeton, NJ, 1965.

[159] P. Kinget and M. Steyaert. A programmable analogue cmos chip for high speed image processing based on cellular neural networks. In *Proceedings of the IEEE 1994 Custom Integrated Circuits Conference*, pages 570–3, 1-4 May 1994.

[160] P. Kinget and M. Steyaert. Evaluation of cnn template robustness towards vlsi implementation. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 381–6, 18-21 Dec. 1994.

[161] P. Kinget and M. Steyaert. Impact of system specifications on analogue cmos implementations of continuously programmable cellular neural networks. In *1994 IEEE International Conference on Neural Networks. IEEE World Congress on Computational Intelligence*, pages 1949–54 vol.3, 27 June-2 July 1994.

[162] P. Kinget and M. Steyaert. Input/output hardware strategies for cellular neural networks. In *1994 IEEE International Conference on Neural Networks. IEEE World Congress on Computational Intelligence*, pages 1899–902 vol.3, 27 June-2 July 1994.

[163] P. Kinget and M. Steyaert. Analogue cmos vlsi implementation of cellular neural networks with continuously programmable templates. In *1994 IEEE International Symposium on Circuits and Systems*, pages 367–70 vol.6, 30 May-2 June 1994.

[164] P. Kinget and M.S.J. Steyaert. A programmable analog cellular neural network cmos chip for high speed image processing. *IEEE Journal of Solid-State Circuits*, 30(3):235–43, March 1995.

[165] I. Kohn, B. Smith, H. Harrer, and J.A. Nossek. A flexible pc-controlled analog/digital test set for cnns. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 228–33, 14-16 Oct.1992.

[166] Bart Kosko. *Neural networks and fuzzy systems : a dynamical systems approach to machine intelligence.* Prentice Hall, Englewood Ciffs, NJ, 1992. Engineering QA76.76.E95.K67 1991.

[167] J. Kowalski, K. Slot, and T. Kacprzak. A cmos current-mode vlsi implementation of cellular neural network for an image objects area estimation. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, page 351, 18-21 Dec. 1994.

[168] J. Kowalski, K. Slot, and T. Kacprzak. A cmos current-mode vlsi implementation of cellular neural network for an image objects area estimation. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, page 351, 18-21 Dec. 1994.

[169] T. Kozek and T. Roska. A double time-scale cnn for solving two-dimensional navier-stokes equations. *International Journal of Circuit Theory and Applications*, 1(24):49–55, Jan.-Feb. 1996.

[170] T. Kozek, T. Roska, and L.O. Chua. Genetic algorithm for cnn template learning. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(6):392–402, June 1993.

[171] K.R. Krieg and L.O. Chua. Hardware and algorithms for the functional evaluation of cellular neural networks and analog arrays. In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 169–71, 16-19 Dec. 1990.

[172] Stephen W. Kuffler, John G. Nicholls, and A. Robert Martin. *From neuron to brain : a cellular approach to the function of the nervous system(2nd ed.)*. Sunderland, Mass. : Sinauer Associates, 1984.

[173] Arun D. Kulkarni. *Artificial neural networks for image understanding*. New York : Van Nostrand Reinhold, 1994.

[174] K.K. Lai and P.H.W. Leong. Implementation of time-multiplexed cnn building block cell. In *Proceedings of the Fifth International Conference on Microelectronics for Neural Networks and Fuzzy Systems. MicroNeuro'96.*, pages 80–5, 12-14 Feb. 1996.

[175] K.K. Lai and P.H.W. Leong. An area efficient implementation of a cellular neural network. In *Proceedings. 1995. Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems.*, pages 51–4, 20-23 Nov. 1995.

[176] K.K. Lai, P.H.W. Leong, and M.A. Jabri. Analogue cmos vlsi implementation of cellular neural networks. In *Proceedings of the Sixth Australian Conference on Neural Networks (ACNN'95).*, pages 17–20, 6-8 Feb. 1995.

[177] Christopher G. Langton. *Artificial life : an overview*. Cambridge, Mass. : MIT Press, 1995.

[178] C.-C. Lee and J.P. de Gyvez. Single-layer cnn simulator. In *1994 IEEE International Symposium on Circuits and Systems*, pages 217–20 vol.6, 30 May-2 June 1994.

[179] Chi-Chien Lee and J. Pineda de Gyvez. Time-multiplexing cnn simulator. In *1994 IEEE International Symposium on Circuits and Systems*, pages 407–10 vol.6, 30 May-2 June 1994.

[180] J.S.J. Lee, R.M. Haralick, and L. G. Shapior. Morphological edge detection. *IEEE J. Robotics and Automat.*, 3(2), Apr. 1987.

[181] S.C. Lee and E.T. Lee. Fuzzy sets and neural networks. *J. Cybernet.*, 4:83–103, 1974.

[182] D. Lim and G.S. Moschytz. A programmable, modular cnn cell. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 79–84, 18-21 Dec. 1994.

[183] H. Magnussen and J.A. Nossek. Towards a learning algorithm for discrete-time cellular neural networks. In *CNNA'92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 80–5, 1992.

[184] H. Magnussen and J.A. Nossek. A geometric approach to properties of the discrete-time cellular neural network. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 41(10):625–34, Oct. 1994.

[185] H. Magnussen and J.A. Nossek. Global learning algorithms for discrete-time cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 165–70, 1994.

[186] H. Magnussen, G. Papoutsis, and J.A. Nossek. Continuation-based learning algorithm for discrete-time cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 171–6, 1994.

[187] M. Mancuso, R.D. Luca, R. Poluzzi, and G.G. Rizzotto. A fuzzy decision directed filter for impulsive noise reduction. *Fuzzy Sets and Systems*, 77:111–116, 1996.

[188] P. A. Maragos and R.W. Schafer. Morphological skeleton representation and coding of binary images. *IEEE Trans. Acoust. Speech signal Processing*, 34(5), Oct. 1986.

[189] P.A. Maragos and R.W. Schafer. Morphological gray-scale reconstruction in image analysis: Applications and efficient algorithms. *IEEE Trans. Acoust., Speech, Signal Processing.* 34(10):1228–1244, 10 1986.

[190] M. Mezard, G. Parisi, and M. A. Virasoro(eds.). *Spin Glass Theorey and Beyond.* World Scientif. Singapore, 1987.

[191] H. Mizutani. A new learning method for multilayered cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 195–200, 1994.

[192] Nelson Morgan(Ed.). *Artificial neural networks : electronic implementations.* Los Alamitos, CA : IEEE Computer Society Press, 1990.

[193] Y. Nakagawa and A. Rosenfeld. A note on the use of local min and max operations in digital picture processing. *IEEE Trans., Syst., Man, Cybern.*, SMC-8:632–635, 1978.

[194] L. Nemes and T. Roska. A cnn model of oscillation and chaos in ant colonies: a case study. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 42(10):741–5, Oct. 1995.

[195] L. Nemes, G. Toth, T. Roska, and A. Radvanyi. Analogic cnn algorithms for 3d interpolation-approximation and object rotation using controlled switched templates. *International Journal of Circuit Theory and Applications*, 24(3):283–300, May-June 1996.

[196] A. D. Nola, W. Pedrycz, and S. Sessa. Fuzzy relational structures: The state-of-art. *Fuzzy Sets and Systems*, 75:241–262, 1995.

[197] J.A. Nossek. Design and learning with cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 137–46, 1994.

[198] J.A. Nossek. Design and learning with cellular neural networks. *International Journal of Circuit Theory and Applications*, 24(1):15–24, Jan.-Feb. 1996.

[199] J.A. Nossek, G. Seiler, T. Roska, and L.O. Chua. Cellular neural networks: theory and circuit design. *International Journal of Circuit Theory and Applications*, 20(5):533–53, Sept.-Oct. 1992.

[200] M.A. Nowak and R.M. May. The spatial dilemmas of evolution. *International Journal of Bifurcation and Chaos*, 3(1):35–78, Feb. 1993.

[201] M.J. Ogorzalek, A. Dabrowski, and W. Dabrowski. Hyperchaos, clustering and cooperative phenomena in cnn arrays composed of chaotic circuits. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 315–20, 1994.

[202] M.J. Ogorzalek, Z. Galias, W. Dabrowski, and A. Dabrowski. Spatio-temporal co-operative phenomena in cnn arrays composed of chaotic circuits-simulation experiments. *International Journal of Circuit Theory and Applications*, 24(3):261–8, May-June 1996.

[203] Special Issue on Cellular Neural Networks. *Int. J. Circuit Theory and Appls.* vol.24, No.1, Jan.-Feb.1996.

[204] Special Issue on Cellular Neural Networks. *IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications.* vol.40, No.3, Mar.1993.

[205] Special Issue on Cellular Neural Networks. *IEEE Transactions on Circuits and Systems–II: Analog and Digital Signal Processing.* vol.40, No.3, Mar.1993.

[206] Special Issue on Cellular Neural Networks. *Int. J. Circuit Theory and Appls.* vol.24, No.3, May.-Jun.1995.

[207] Special Issue on Cellular Neural Networks. *Int. J. Circuit Theory and Appls.* vol.20, No.5, Sep.-Oct.1992.

[208] J.M. Ortega. *Matrix Theory—A Second Course.* Plenum Press, New York, 1987.

[209] J.A. Osuna and G.S. Moschytz. On the separating capability of cellular neural networks. *International Journal of Circuit Theory and Applications*, 24(3):253–9, May-June 1996.

[210] A. Paasio, A. Dawidziuk, K. Halonen, and V. Porra. Digitally controllable weights in current mode cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 73–7, 18-21 Dec. 1994.

[211] A. Paasio, A. Dawidziuk, and V. Porra. Pulse stream current mode cmos cnn chip. In *Proceedings of the Fourth IEEE International Workshop on Cellular Neural Networks and Their Aplications. Seville, Spain, June 24-26*, pages 457–460, 1996.

[212] A. Paasio, K. Halonen, and V. Porra. Cmos implementation of associative memory using cellular neural network having adjustable template coefficients. In *1994 IEEE International Symposium on Circuits and Systems*, pages 487–90 vol.6, 30 May-2 June 1994.

[213] A. Paasio, K. Halonen, V. Porra, and A. Dawidziuk. Current mode cellular neural network with digitally adjustable template coefficients. In *Proceedings of the Fourth International Conference on Microelectronics for Neural Networks and Fuzzy Systems*, pages 268–72, 26-28 Sept. 1994.

[214] Sungjun Park and Soo-Ik Chae. [stereopsis with cellular neural networks]. *Journal of the Korean Institute of Telematics and Electronics*, 31B(12):124–31, Dec. 1994.

[215] Sungjun Park, Joonho Lim, and Soo-Ik Chae. [hardware implementation of discrete-time cellular neural networks using distributed arithmetic]. *Journal of the Korean Institute of Telematics and Electronics*, 33B(1):153–60, Jan. 1996. (in Korean).

[216] Sungjun Park, Seung-Jai Min, and Soo-Ik Chae. Stereo correspondence with discrete-time cellular neural networks. In *1994 IEEE International Symposium on Circuits and Systems*, pages 225–8 vol.6, 30 May-2 June 1994.

[217] A. Passio, A. Dawidziuk, K. Halonen, and V. Porra. Robust cmos cnn implementation with respect to manufaturing inacuracies. In *Proceedings of the Fourth IEEE International Workshop on Cellular Neural Networks and Their Aplications, Seville, Spain, June 24-26*, pages 381–386, 1996.

[218] S. Paul, K. Huper, and J.A. Nossek. Analog media filtering. In *1992 IEEE Int. Conf. on Acoustics, Speech and Signal Processing.*

[219] S. Paul, K. Huper, and J.A. Nossek. Mapping nonlinear lattice equations onto cellular neural networks. *IEEE Trans. Circuits and Systems-I*, 1993.

[220] S. Paul, K. Huper, J.A. Nossek, and L.O. Chua. Mapping nonlinear lattice equations onto cellular neural networks. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(3):196–203, March 1993.

[221] Martin Pearl. *Matrix Theory and Finite Mathematics*, volume 2 of *International Series in Pure and Applied Mathematics.* McGraw-Hill, New York, 1973.

[222] W. Pedrycz. An identification algorithm in fuzzy relational systems. *Fuzzy Sets and Systems*, 13:153–167, 1984.

[223] W. Pedrycz. Neuro computations in relational systems. *IEEE Trans. Pattern, Anal. Machine Intell.*, 13(3):289–297, March 1991.

[224] W. Pedrycz. Neurocomputations in relational systems. *IEEE Trans. Pattern Anal. Machine Intell.*, PAMI-13(3):289–297, March 1991.

[225] F.J. Pelayo, A. Prieto, B. Pino, and P. Martin-Smith. Analog vlsi implementation of a neural network with competitive learning. In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 197–205, 1990.

[226] S. Peleg and A. Rosenfeld. A min-max medial axis transformation. *IEEE Trans. Patt. Ana. and Machine Intell.*, PAMI-3:189–192, 1989.

[227] J.M. Perdang and A. Lejeune(Eds). *Cellular automata : prospects in astrophysical applications.* River Edge, NJ : World Scientific, 1993.

[228] V. Perez-Munuzuri, A.P. Munuzuri, M. Gomez-Gesteira, V. Perez-Villar, , and others. Nonlinear waves, patterns and spatio-temporal chaos in cellular neural networks. *Philosophical Transactions of the Royal Society, Series A (Physical Sciences and Engineering)*, 353(1701):101–13, Oct. 1995.

[229] V. Perez-Munuzuri, V. Perez-Villar, and L.O. Chua. Autowaves for image processing on a two-dimensional cnn array of excitable nonlinear circuits: flat and wrinkled labyrinths. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(3):174–81, March 1993.

[230] R. Perfetti. On the convergence of reciprocal discrete-time cellular neural networks. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(4):286–7, April 1993.

[231] R. Perfetti. Relation between template spectrum and convergence of discrete-time cellular neural networks. *Electronics Letters*, 29(25):2208–9, 9 Dec. 1993.

[232] R. Perfetti. On the op-amp based circuit design of cellular neural networks. *International Journal of Circuit Theory and Applications*, 22(5):425–30, Sept.-Oct. 1994.

[233] R. Perfetti. Some properties of the attractors of discrete-time cellular neural networks. *International Journal of Circuit Theory and Applications*, 23(5):485–99, Sept.-Oct. 1995.

[234] C.-K. Pham, M. Ikegami, and M. Tanaka. Discrete time cellular neural networks with two types of neuron circuits for image coding and their vlsi implementations. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E78-A(8):978–88, Aug. 1995.

[235] C.-K. Pham, T. Kimura, M. Ikegarni, and M. Tanaka. Pulse coded cellular neural network and it's hardware implementation. In *1995 IEEE International Conference on Neural Networks Proceedings*, pages 1590–4 vol.4., 27 Nov.-1 Dec. 1995.

[236] J. Pineda de Gyvez. Xcnn: a software package for color image processing. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 219–24, 18-21 Dec. 1994.

[237] A. Piovaccari and G. Setti. A versatile cmos building block for fully analogically-programmable vlsi cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, page 347, 18-21 Dec. 1994.

[238] I. Pitas and A.N. Vanetsanopoulos. *Nonlinear Digital Filters: Principles and Applications.* Kluwer Academic Publisher, Dordrecht, 1991.

[239] I. Pitas and A. N. Venetsanpoulos. Morphological shape decomposition. *IEEE Trans. Pattern. Anal. Machine Intell.*, 12(1), Jan. 1990.

[240] S.A. Pol and R.A. King. Image enhancement using smoothing with fuzzy sets. *IEEE Trans., Syst., Man, Cybern.*, SMC-11:494–501, 1981.

[241] G. I. Poliakov. *Neuron structure of the brain.* Cambridge, Mass., Harvard University Press, 1972.

[242] Proceedings. *1990 IEEE International Workshop on Cellular Neural Networks and Their applications, CNNA-90 : proceedings.* Piscataway, NJ : IEEE Service Center [distributor], 1990.

[243] Proceedings. *CNNA '92 : proceedings, second International Workshop on Cellular Neural Networks and their Applications.* Piscataway, NJ : IEEE Service Centre, 1992.

[244] Proceedings. *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and Their Applications (CNNA-94).* Piscataway, NJ : IEEE Service Centre, 1994.

[245] Proceedings. *Proceedings of the Fourth IEEE International Workshop on Cellular Neural Networks and Their Applications (CNNA-96).* Piscataway, NJ : IEEE Service Centre, 1996.

[246] T.J. Procyk and E.H. Mandani. A liguistic self-organizing process controller. *Automatica*, 15(1):15–30, 1979.

[247] F. Puffer, R. Tetzlaff, and D. Wolf. A learning algorithm for cellular neural networks (cnn) solving nonlinear partial differential equations. In *Proceedings 1995 URSI International Symposium on Signals, Systems, and Electronics. ISSSE '95*, pages 501–4, 1995.

[248] L. Raffo, S.P. Sabataini, and G.M. Bisio. A programmable vlsi architecture based on multilayer cnn paradigms for real-time visual processing. *International Journal of Circuit Theory and Applications*, 24(3):357–67, May-June 1996.

[249] C. Rekeczky, Y. Nishio, A. Ushida, and T. Roska. Cnn based adaptive smoothing and some novel types of nonlinear operators for grey-scale image processing. In *Proc. 1995 International Symposium on Nonlinear Theory and Its Applications (NOLTA'95).*

[250] C. Rekeczky, Y. nishio. A. Ushida, and T. Roska. Cnn based adaptive smoothing and some novel types of nonlinear operators for grey-scale image processing. In *1995 Int. Sym. on Nonlinear Theorey and Its Applications(NOLTA'95)*, pages 683–688, 1995.

[251] C. Rekeczky, A. Ushida, and T. Roska. Rotation invariant detection of moving and standing objects using analogic cellular neural network algorithms based on ring-codes. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E78-A(10):1316–30, Oct. 1995.

[252] B. Reljin, T. Serdar, P. Kostic, and A. Pavasovic. Cmos vlsi realization of voltage-mode programmable analog cellular neural network. In *1995 20th International Conference on Microelectronics. Proceedings*, pages 497–502 vol.2, 12-14 Sept.1995.

[253] A. Rodriguez-Vazquez, S. Espejo, R. Dominguez-Castron, J.L. Huertas, , et al. Current-mode techniques for the implementation of continuous- and discrete-time cellular neural networks. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 40(3):132–46, March 1993.

[254] A. Rodriquez-Vazques, R. Dominguez-Castro, and J.L. Huertas. Accurate design of analog cnn in cmos digital technologies. In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 273–80, 16-19 Dec. 1990.

[255] A. Rosenfeld and A.C. Kak. *Digital Picture Processing.* New York: Academic, 1976.

[256] T. Roska. Programming cnn: a hardware accelerator for simulation, learning, and real-time applications. In *Proceedings of the 35th Midwest Symposium on Circuits and Systems*, pages 437–40, 1992.

[257] T. Roska. Analogic algorithms running on the cnn universal machine. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 3–8, 1994.

[258] T. Roska. The cnn universal machine-a summary of an analogic supercomputer chip architecture for very high speed visual processing. In *1994 CERN School of Computing Proceedings (CERN 95-01). (1994 CERN School of Computing Proceedings*, pages 295–7, 1994.

[259] T. Roska, G. Bartfai, P. Szolgay, T. Sziranyi, , and others. A hardware accelerator board for cellular neural networks: Cnn-hac. In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 160–8, 16-19 Dec. 1990.

[260] T. Roska, G. Bartfai, P. Szolgay, T. Sziranyi, , and others. A digital multiprocessor hardware accelerator board for cellular neural networks: Cnn-hac. *International Journal of Circuit Theory and Applications*, 20(5):589–99, Sept.-Oct 1992.

[261] T. Roska and G. Bartfay. Cnn-hac: a digital multiprocessor hardware accelerator for general cellular neural networks. In *Hungarian Acad. Sci., Budapest, Hungary*, page (technical report), 1990.

[262] T. Roska, T. Boros, A. Radvanyi, P. Thiran, , and others. Detecting moving and standing objects using cellular neural networks. *International Journal of Circuit Theory and Applications*, 20(5):613–28, Sept.-Oct. 1992.

[263] T. Roska and L.O. Chua. Cellular neural networks with nonlinear and delay-type template elements. In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 12–25, 1990.

[264] T. Roska and L.O Chua. The cnn universal machine. i. the architecture. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 1–10, 1992.

[265] T. Roska and L.O Chua. The cnn universal machine. ii. programmability and applications. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 181–90, 1992.

[266] T. Roska and L.O. Chua. Cellular neural networks with nonlinear and delay-type template elements and nonuniform grids. *International Journal of Circuit Theory and Applications*, 20(5):469–81, Sept.-Oct. 1993.

[267] T. Roska and L.O. Chua. The cnn universal machine: an analogic array computer. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 40(3):163–73, Mar. 1993.

[268] T. Roska, L.O. Chua, D. Wolf, T. Kozek, , and others. Simulating nonlinear waves and partial differential equations via cnn. i. basic techniques. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 42(10):807–15, Oct. 1995.

[269] T. Roska and L. Kek (Eds.). Median—removes impulsive noise from a grey-scale image, p.34 of csl-cnn software library: Templates and algorithms (version 6.4). Technical report, Computer and Automaton Institute(MTA SzTAKI) of The Hungarian Academy of Sciences, Budapest, 1995.

[270] T. Roska and A. Radvanyi. Cnnd simulator. cellular neural network embedded in a simple dual computing structure. user's guide version 3.01. In *Hungarian Acad. Sci., Budapest, Hungary.*, Aug. 1990.

[271] T. Roska and J. Vandewalle(Eds.). *Cellular neural networks*. New York : Wiley, 1993.

[272] T. Roska, C.W. Wu, M. Balsi, and L.O. Chua. Stability and dynamics of delay-type general and cellular neural networks. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 39(6):487–90, June 1992.

[273] T. Roska, C.W. Wu, and L.O. Chua. Stability of cellular neural networks with dominant nonlinear and delay-type templates. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(4):270–2, Apr. 1993.

[274] A. Rueda and J.L. Huertas. Testability in analogue cellular neural networks. *International Journal of Circuit Theory and Applications*, 20(5):583–7, Sept.-Oct. 1992.

[275] F. Russo. A user-friendly research tool for image processing with fuzzy rules. In *Proc. First IEEE Int. Conf. on Fuzzy System, Fuzzy-IEEE'92*, pages 561–568, 1992.

[276] F. Russo and G. Ramponi. Combined fire filter for image enhancement. In *Proceedings of the Third IEEE Conference on Fuzzy Systems. IEEE World Congress on Computational Intelligence*, pages 249–253, 1994.

[277] F. Russo and G. Ramponi. Edge extraction by fire operators. In *Proceedings of the Third IEEE Conference on Fuzzy Systems. IEEE World Congress on Computational Intelligence*, pages 249–253, 1994.

[278] F. Russo and G. Ramponi. Nonlinear fuzzy operators for image processing. *Signal Processing*, pages 429–440, 1994.

[279] F. Russo and G. Ramponi. A fuzzy operator for the enhancement of blurred and noisy images. *IEEE Trans. on Image Processing*, 4(8):1169–1174, Aug. 1995.

[280] M. Salerno, F. Sargeni, and V. Bonaiuto. 6x6 dpcnn: a programmable mixed analogue-digital chip for cellular neural networks. In *Proceedings of the Fourth IEEE International Workshop on Cellular Neural Networks and Their Aplications, Seville, Spain, June 24-26*, pages 451–456, 1996.

[281] M. Salerno, F. Sargeni, and V. Bonaiuto. Dpcnn: a modular chip for large cnn arrays. In *1995 IEEE Symposium on Circuits and Systems (Cat. No.95CH35771). (1995 IEEE Symposium on Circuits and Systems*, pages 417–20 vol.1, 28 April-3 May 1995.

[282] E. Sanchez. Resolution of composite fuzzy relation equations. *Inf. Contl.*, 30:38–48, 1976.

[283] A. Sani, S. Graffi, G. Masetti, and G. Setti. Design of cmos cellular neural networks operating at several supply voltages. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 363–8, 18-21 Dec. 1994.

[284] F. Sargeni. Digitally programmable transconductance amplifier for cnn applications. *Electronics Letters*, 30(11):870–2, 26 May 1994.

[285] F. Sargeni and V. Bonaiuto. High performance digitally programmable cnn chip with discrete templates. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 67–72, 18-21 Dec. 1994.

[286] F. Sargeni and V. Bonaiuto. High performance digitally programmable cnn chip with discrete templates. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 67–72, 18-21 Dec. 1994.

[287] F. Sargeni and V. Bonaiuto. A fully digitally programmable cnn chip. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 42(11):741–5, Nov. 1995.

[288] F. Sargeni and V. Bonaiuto. A 3*3 digitally programmable cnn chip. *International Journal of Circuit Theory and Applications*, 24(3):369–79, May-June 1996.

[289] T. Sato, H. Ushida, and T. Yamaguchi. Retrieval system to generate facial expressions using chaos. In *Proceedings of 1995 IEEE International Conference on Fuzzy Systems*, pages 1489–1494, 1995.

[290] A.J. Schuler, M. Brabec, D. Schubel, and J.A. Nossek. Hardware-oriented learning for cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 183–8, 1994.

[291] A.J. Schuler, P. Nachbar, J.A. Nossek, and L.O. Chua. Learning state space trajectories in cellular neural networks. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 68–73, 1992.

[292] J. Serra. *Image Analysis and Mathematical Morphology*. New York: Academic, 1982.

[293] J. Serra(Ed.). *Image Analysis and Mathematical Morphology (Vol. 2): Theory Advances.* New York: Academic, 1988.

[294] Ishwar K. Sethi and Anil K. Jain(eds.). *Artificial neural networks and statistical pattern recognition : old and new connections.* New York : North-Holland, 1991.

[295] J.J. Shann and H.C. Fu. A fuzzy neural network for rule acquiring on fuzzy control systems. *Fuzzy Sets and Systems*, 71:345–357, 1995.

[296] B.J. Sheu, S.H. Bang, and Wai-Chi Fang. Analog vlsi design of cellular neural networks with annealing agility. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 387–92, 18-21 Dec. 1994.

[297] B.J. Sheu, S.H. Bang, and Wai-Chi Fang. Vlsi design of cellular neural networks with annealing and optical input capabilities. In *1995 IEEE Symposium on Circuits and Systems (Cat. No.95CH35771). (1995 IEEE Symposium on Circuits and Systems*, pages 653–6 vol.1, 28 April-3 May 1995.

[298] B.J. Sheu, R.C. Chang, T.H. Wu, and S.H. Bang. Vlsi-compatible cellular neural networks with optimal solution capability for optimization. In *1995 IEEE Symposium on Circuits and Systems (Cat. No.95CH35771). (1995 IEEE Symposium on Circuits and Systems*, pages 1165–8 vol.2, 28 April-3 May 1995.

[299] B.E. Shi and L.O. Chua. Resistive grid image filtering: input/output analysis via the cnn framework. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 39(7):531–48, July 1992.

[300] B.E. Shi, T. Roska, and L.O. Chua. Design of linear cellular neural networks for motion sensitive filtering. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 40(5):320–31, May 1993.

[301] F.Y. Shih and O.R. Mitchell. Automated fast recognition and location of arbitary shaped objects by image morphology. In *Proc. of IEEE Conf. Computer Vision and Pattern Recognition*, June, 1988.

[302] F.Y. Shih and O.R. Mitchell. Industrial parts recognition and inspection by image morphology. In *Proc. of IEEE Conf. Computer Vision and Pattern Recognition*, June, 1988.

[303] F.Y.C. Shih and O.R. Mitchell. A mathematical morphology approach to euclidean distance transformation. *IEEE Trans. on Image Processing*, 1(2):197–204, 2 1992.

[304] N. Shimizu, Gui-Xin Cheng, M. Ikegami, Y. Nakamura, , and others. Pipelining gauss seidel method for analysis of discrete time cellular neural networks. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E77-A(8):1396–403, Aug. 1994.

[305] A. Slavova. Cellular neural networks with nonlinear dynamics. *Neural, Parallel and Scientific Computations*, 3(3):369–77, Sept. 1995.

[306] K. Slot. Optically realized feedback cellular neural networks. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 175–80, 14-16 Oct.1992.

[307] K. Slot. Large-neighborhood templates implementation in discrete-time cnn universal machine with a nearest-neighbor connection pattern. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 213–18, 1994.

[308] K. Slot, T. Roska, and L.O. Chua. Optically realized feedforward-only cellular neural networks. *Archiv fur Elektronik und Uebertragungstechnik*, 46(3):158–67, May 1992.

[309] G.O. Sullivan, P. Horan, J. Hegarty, S. Kakizaki, B. Kelly, and E. McCabe. A fully optically addressable connected component detector in cmos. In *Proceedings of the Fourth IEEE International Workshop on Cellular Neural Networks and Their Aplications, Seville, Spain, June 24-26*, pages 439–444, 1996.

[310] H. Suzuki, T. Matsumoto, and L.O. Chua. A cnn handwritten character recognizer. *International Journal of Circuit Theory and Applications*, 20(5):601–12, Sept.-Oct. 1992.

[311] T. Sziranyi. Robustness of cellular neural networks in image deblurring and texture segmentation. *International Journal of Circuit Theory and Applications*, 24(3):381–96, May-June 1996.

[312] T. Sziranyi and M. Csapodi. Texture classification by cellular neural network and genetic learning. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, pages 381–3, 1994.

[313] P. Szolgay, A. Katona, G. Eross, and A. Kiss. An experimental system for path tracking of a robot using a 16*16 connected component detector cnn chip with direct optical input. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, 18-21 Dec. 1994.

[314] P. Szolgay, A. Katona, G. Eross, and A. Kiss. An experimental system for path tracking of a robot using a 16*16 connected component detector cnn chip with direct optical input. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 261–6, 18-21 Dec. 1994.

[315] P. Szolgay, I. Kispal, and T. Kozek. An experimental system for optical detection of layout errors of printed circuit boards using learned cnn templates. In *CNNA '92 Proceedings. Second International Workshop on Cellular Neural Networks and their Applications*, pages 203–9, 1992.

[316] M. Tanaka, K.R. Crounse, and T. Roska. Parallel analog image coding and decoding by using cellular neural networks. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E77-A(8):1387–95, Aug. 1994.

[317] Lu Hong Tao, Yang Lu Xi, Wang Bao Yun, and He Zhen Ya. A new type of chaotic attractor with cellular neural networks. In *Proceedings of ISCAS'95 - International Symposium on Circuits and Systems*. pages 997–1000, 1995.

[318] R. Tetzlaff and D. Wolf. A learning algorithm for the dynamics of cnn with nonlinear templates-part i: Discrete-time case. In *Proceedings of the Fourth IEEE International Workshop on Cellular Neural Networks and Their Aplications, Seville, Spain, June 24-26*, pages 461–466, 1996.

[319] R. Tetzlaff and D. Wolf. A learning algorithm for the dynamics of cnn with nonlinear templates. part ii: Continuous-time case. In *Proceedings of the Fourth IEEE International Workshop on Cellular Neural Networks and Their Aplications, Seville, Spain, June 24-26*, pages 467–472, 1996.

[320] Nadia Magnenat Thalmann and Daniel Thalmann. *Artificial life and virtual reality.* New York : Wiley, 1994.

[321] P. Thiran, K.R. Crounse, L.O. Chua, and M. Hasler. Pattern formation properties of autonomous cellular neural networks. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 42(10):757–74, Oct. 1995.

[322] Tommaso Toffoli. *Cellular automata machines : a new environment for modeling.* MIT Press series in scientific computation, 1987.

[323] R.M. Tong. Analysis and control of fuzzy systems using finite discrete relations. *Int. J. Control*, 27(3):431–440, 1978.

[324] R.M. Tong. Some properties of fuzzy feedback systems. *IEEE Trans. on Systems, Man, and Cybernetics*, SMC-10(6):327–330, 1980.

[325] L. Torres and M. Kunt. *Video Coding: The Second Generation Approach*. Kluwer Academic Publishers, Boston, 1996.

[326] P. Tzionas. A cellular neural network learning the pseudorandom behaviour of a complex system. *International Journal of Electronics*, 80(3):405–13, March 1996.

[327] W. Utschick and J.A. Nossek. Computational learning theory applied to discrete-time cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 159–64, 1994.

[328] J.W.M. Van Dam, B.J.A. Krose, and F.C.A. Groen. Cnn: a neural architecture that learns multiple transformations of spatial representations. In *ICANN '94. Proceedings of the International Conference on Artificial Neural Networks*, pages 1420–3, 1994.

[329] L. Vandenberghe, S. Tan, and J. Vandewalle. Cellular neural networks: dynamic properties and adaptive learning algorithm. In *Neural Networks. EURASIP Workshop 1990 Proceedings. (Neural Networks. EURASIP Workshop 1990 Proceedings*, pages 141–50, 1990.

[330] J.E. Varrientos, J. Ramirez-Angulo, and E. Sanchez-Sinencio. Cellular neural network implementations: a current mode approach. In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 216–25, 16-19 Dec. 1990.

[331] J.E. Varrientos, J. Ramirez-Angulo, and E. Sanchez-Sinencio. Cellular neural network implementations: a current mode approach. In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 216–25, 16-19 Dec. 1990.

[332] J.E. Varrientos, J. Ramirez-Angulo, and E. Sanchez-Sinencio. A current-mode cmos cellular neural network. In *Proceedings of the 33rd Midwest Symposium on Circuits and Systems*, pages 12–14, Aug. 1990.

[333] J.E. Varrientos and E. Sanchez-Sinencio. Cellsim: a cellular neural network simulator for the personal computer. In *Proceedings of the 35th Midwest Symposium on Circuits and Systems*, pages 1384–7, 1992.

[334] J.E. Varrientos, E. Sanchez-Sinencio, and J. Ramirez-Angulo. A current-mode cellular neural network implementation. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 40(3):147–55, March 1993.

[335] P.L. Venetianer, F. Werblin, T. Roska, and L.O. Chua. Analogic cnn algorithm for some image compression and restoration tasks. *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, CASI-42:278–284, 1995.

[336] P.L. Venetianer, F. Werblin, T. Roska, and L.O. Chua. Analogic cnn algorithms for some image compression and restoration tasks. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 42(5):278–84, May 1995.

[337] L. Vincent. Morphological gray-scale reconstruction in image analysis: Applications and efficient algorithms. *IEEE Trans. on Image Processing*, IP-2:176–201, 1994.

[338] Kuei-Ann Wen, Jeong-Yuan Su, and Chung-Yen Lu. Vlsi design of digital cellular neural networks for image processing. *Journal of Visual Communication and Image Representation*, 5(2):117–26, June 1994.

[339] F. Werblin, T. Roska, and L.O. Chua. The analogic cellular neural network as a bionic eye. *International Journal of Circuit Theory and Applications*, 23(6):541–69, Nov.-Dec. 1995.

[340] Artificial Life Workshop. *Artificial life III : proceedings of the Workshop on Artificial Life*. Reading, Mass. : Addison-Wesley, 1994.

[341] R. Yager. A measurement informational discussion on fuzzy union and intersection. *Int.J. Man-Machine Studies*, 11:189–200. 1979.

[342] R.R. Yager and L.A. Zadeh(eds.). *An Introduction to Fuzzy Logic Application in Intelligent Systems.* Kluwer Academic Publishers, Boston, 1992.

[343] T. Yamakawa. Pattern recognition hardware system employing a fuzzy neuron. In *Proc. Int. Conf. Fuzzy Logic*, pages 934–938, 1990.

[344] T. Yamakawa and M. Furukawa. A design algorithm of membership function for a fuzzy neuron using example-based learning. In *Proc. IEEE Int. Conf. Fuzzy Syst. (FUZZY-IEEE'92)*, pages 943–948, 1992.

[345] C.M. Yang, T. Yang, and K.Y. Zhang. Chaos in the discrete time cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 297–302, 1994.

[346] Hong-Kui Yang, M.A. Yakout, and E.I. El-Masry. Current-mode implementation of discrete-time cellular neural networks using the pulse width modulation technique. In *Proceedings of the 37th Midwest Symposium on Circuits and Systems*, pages 457–60 vol.1, 3-5 Aug. 1994.

[347] L. Yang, L.O. Chua, and K.R. Krieg. Vlsi implementation of cellular neural networks. In *1990 IEEE International Symposium on Circuits and Systems*, pages 2425–7, 1990.

[348] T. Yang. Blind signal separation using cellular neural networks. *International Journal of Circuit Theory and Applications*, 22(5):399–408, 10 1994.

[349] T. Yang, L.O. Chua, and K.R. Crounse. Application of discrete-time cellular neural networks to image copyright labeling. In *Proceedings of the Forth IEEE International Workshop on Cellular Neural Networks and Their Aplications(CNNA-96)*, pages 19–24, 1996.

[350] T. Yang and L.B. Yang. The global stability of fuzzy cellular neural network. *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, 43(10):880–883, Oct. 1996.

[351] T. Yang and L.B. Yang. Application of fuzzy cellular neural networks to euclidean distance transformation. *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, 44(3):242–6, Mar. 1997.

[352] T. Yang and L.B. Yang. Application of fuzzy cellular neural networks to morphological gray-scale reconstruction. *International Journal of Circuit Theory and Applications*, 25(3):153–165, May-June 1997.

[353] T. Yang and L.B. Yang. Fuzzy cellular neural network: A new paradigm for image processing. *International Journal of Circuit Theory and Applications*, 25:in press, 1997.

[354] T. Yang and L.B. Yang. Fuzzy cellular neural network and its applications. *Chinese Journal of Electronics (English Version)*, 6:in press, 1997.

[355] T. Yang, L.B. Yang, C.W. Wu, and L.O. Chua. Fuzzy cellular neural networks: Applications. In *Proc. 4th IEEE Int. Workshop on Cellular Neural Networks and Their Applications (CNNA'96)*, pages 225–230, 1996.

[356] T. Yang, L.B. Yang, C.W. Wu, and L.O. Chua. Fuzzy cellular neural networks: Theory. In *Proc. 4th IEEE Int. Workshop on Cellular Neural Networks and Their Applications (CNNA'96)*, pages 181–186, 1996.

[357] T. Yang, L.B. Yang, and C.M. Yang. Analisis of cellular neural network based rank-order filters. *International Journal of Circuit Theory and Applications*, submitted.

[358] T. Yang, L.B. Yang, and X.P. Yang. Application of cellular neural network to facial expressions animation and high-level image processing. *International Journal of Circuit Theory and Applications*, 25(3):May-Jun., 6 1996.

[359] Tao Yang. [application of cellular neural network to map recognition]. *Journal of Tongji University*, 23(1):107–12, Feb. 1995.

[360] Tao Yang, C.M. Yang, and L.B. Yang. The differences between cellular neural network based and fuzzy cellular neural network based mathematical morphology operations. *International Journal of Circuit Theory and Applications*, 25:In Press, 1997.

[361] Tao Yang, C.M. Yang, and Lin-Bao Yang. Genetic optimization of fuzzy cellular neural networks—get knowledge from both learning and structures. *International Journal of Circuit Theory and Applications*, page submitted.

[362] Tao Yang, C.M. Yang, and Lin Bao Yang. Learning algorithm of fuzzy discrete-time cellular neural networks. *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, page submitted.

[363] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Application of adaptive mathematical morphology to range image. *Inn. Memo. E-Zhou University*, 9:20–25, 1993. (in Chinese).

[364] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Application to fire operators. *Inn. Memo. E-Zhou University*, 9:12–17, 1993. (in Chinese).

[365] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Applications as min-max networks. *Inn. Memo. E-Zhou University*, 7:1–9, 1993. (in Chinese).

[366] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Chaos in fcnn. *Inn. Memo. E-Zhou University*, 12:1–4, 1993. (in Chinese).

[367] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Elementary theory. *Inn. Memo. E-Zhou University*, 2:23–25, 1993. (in Chinese).

[368] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Euclidean distance transformation. *Inn. Memo. E-Zhou University*, 6:24–29, 1993. (in Chinese).

[369] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Global stability. *Inn. Memo. E-Zhou University*, 3:2–9, 1993. (in Chinese).

[370] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Learning algorithm. *Inn. Memo. E-Zhou University*, 8:14–20, 1993. (in Chinese).

[371] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Local stability. *Inn. Memo. E-Zhou University*, 3:10–16, 1993. (in Chinese).

[372] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Morphological gray-scale reconstruction and applications. *Inn. Memo. E-Zhou University*, 5:7–12, 1993. (in Chinese).

[373] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Relational fuzzy cellular neural networks. *Inn. Memo. E-Zhou University*, 10:11–17, 1993. (in Chinese).

[374] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Structure. *Inn. Memo. E-Zhou University*, 1:4–10, 1993. (in Chinese).

[375] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Universal paradigm for implementation of mathematical morphology transformations. *Inn. Memo. E-Zhou University*, 4:12–18, 1993. (in Chinese).

[376] Tao Yang and Lin-Bao Yang. Application of fuzzy cellular neural networks to new fire filters. *Inn. Memo. E-Zhou University*, 6:18–24, 1994. (in Chinese).

[377] Tao Yang and Lin-Bao Yang. Discrete-time fuzzy cellular neural network. *Inn. Memo. E-Zhou University*, 4:7–13, 1994. (in Chinese).

[378] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network and soft mathematical morphology. *Inn. Memo. E-Zhou University*, 5:1–5, 1994. (in Chinese).

[379] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural network: Difference between fcnn and cnn. *Inn. Memo. E-Zhou University*, 2:6–12, 1994. (in Chinese).

[380] Tao Yang and Lin-Bao Yang. Fuzzy delay-type cellular neural network. *Inn. Memo. E-Zhou University*, 4:20–25, 1994. (in Chinese).

[381] Tao Yang and Lin-Bao Yang. Why fuzzy cellular neural networks are not nonlinear conventional cnn. *Inn. Memo. E-Zhou University*, 12:8–14, 1994. (in Chinese).

[382] Tao Yang and Lin-Bao Yang. Fuzzy cellular neural networks and artificial life. *Inn. Memo. E-Zhou University*, 7:20–25, 1995. (in Chinese).

[383] Tao Yang and Lin-Bao Yang. Genetic algorithm for fuzzy cellular neural networks. *Inn. Memo. E-Zhou University*, 8:1–5, 1995. (in Chinese).

[384] Tao Yang and Lin-Bao Yang. Lingustic flow in fuzzy cellular neural networks. *Inn. Memo. E-Zhou University*, 3:20–25, 1995. (in Chinese).

[385] Tao Yang and Lin-Bao Yang. Stability of fuzzy delay-type cellular neural networks. *Inn. Memo. E-Zhou University*, 2:1–5, 1995. (in Chinese).

[386] Tao Yang and Lin-Bao Yang. The universe of fuzzy cellular neural networks. *Inn. Memo. E-Zhou University*, 2:6–12, 1995. (in Chinese).

[387] Tao Yang, Lin-Bao Yang, and C.M. Yang. Training fuzzy discrete-time cellular neural networks using fuzzy number examples and application. *International Journal of Circuit Theory and Applications*, page submitted.

[388] Tao Yang, Lin-Bao Yang, and Guang Yang. On unconditional stability of the general delayed cellular neural networks. In *Proceedings of the Third IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-94)*, pages 9–14, 1994.

[389] S.Y. Yi and M.J. Chung. Identification of fuzzy relational model and its application to control. *Fuzzy Sets and Systems*, pages 25–33, 1993.

[390] L. Yin, R. Yang, and M.Gabbouj. Weighted median filters: A tutorial. *IEEE Trans. Circuits and Systems-II*, 1996.

[391] L. Yin, R. Yang, and Y. Neuvo. Weighted median filters: A tutorial. *IEEE Trans. Circuits Syst.-II*, 43(3):157–192, 3 1996.

[392] O. Yli-Harja, J. Astala, and Y. Neuvo. Analysis of the properties of median and weighted median filters using threshold logic and stack filter representation. *IEEE Trans. Signal Processing*, 39:395–410, 1991.

[393] L. Zadeh. Fuzzy sets. *Inform. and Control*, 8:338–353, 1965.

[394] L.A. Zadeh, K.S. Fu, K. Tanaka, and M. Shimura(Eds.). *Fuzzy Sets and Their Applications to Cognitive and Decision Processes.* Academic, London, 1975.

[395] A. Zarandy, A. Stoffels, T. Roska, and L.O. Chua. Implementation of binary and gray-scale mathematical morphology on the cnn universal machine. In *Proc. 4th IEEE Int. Workshop on Cellular Neural Networks and Their Applications (CNNA '96)*, pages 151–156, 1996.

[396] A. Zarandy, F. Werblin, T. Roska, and L.O. Chua. Spatial logic algorithms using basic morphological analogic cnn operations. *International Journal of Circuit Theory and Applications*, 24(3):283–300, May-June 1996.

[397] X. Zhang, C.C. Hang, S. Tan, and P.Z. Wang. The min-max function differentiation and training of fuzzy neural networks. *IEEE Trans. on Neural Networks*, 7(5):1139–1150, Sept.1996 1996.

[398] X.H. Zhang, C.C. Hang, S. Tan, and P.Z. Wang. The delta rule and learning for min-max neural networks. In *Proc. IEEE-ICNN'94*, volume 1, pages 38–43, Orlando, FL, 1994.

[399] Yi-Tong Zhou. *Artificial neural networks for computer vision.* New York : Springer-Verlag, 1992.

[400] F. Zou, A. Katerle, and J.A. Nossek. Homoclinic and heteroclinic orbits of the three-cell cellular neural networks. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(11):843–8, Nov. 1993.

[401] F. Zou and J.A. Nossek. A chaotic attractor with cellular neural networks. *IEEE Transactions on Circuits and Systems*, 38(7):811–12, July 1991.

[402] F. Zou and J.A. Nossek. Double scroll and cellular neural networks. In *1992 IEEE International Symposium on Circuits and Systems*, pages 320–3 vol.1, 1992.

[403] F. Zou and J.A. Nossek. An autonomous chaotic cellular neural network and chua's circuit. *Journal of Circuits, Systems and Computers*, 3(2):591–601, June 1993.

[404] F. Zou and J.A. Nossek. Bifurcation and chaos in cellular neural networks. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(3):166–73, March 1993.

[405] F. Zou and J.A. Nossek. Hopf-like bifurcation in cellular neural networks. In *(Proceedings) 1993 IEEE International Symposium on Circuits and Systems.*, pages 2391–4 vol.4., 1993.

[406] F. Zou, S. Schwarz, and J.A. Nossek. Cellular neural network design using a learning algorithm. In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 73–81, 1990.

[407] F. Zou, S. Schwarz, and J.A. Nossek. Cellular neural network design using a learning algorithm. In *1990 IEEE International Workshop on Cellular Neural Networks and their Applications, CNNA-90*, pages 73–81, 1990.