基於細胞神經網路的仿生型電腦視覺系統

Bionic Computer Visual System based on Cellular Neural Networks

研 究 生：黃朝暉　　Student：Chao-Hui Huang

指導教授：林進燈 博士　　Advisor：Dr. Chin-Teng Lin

國 立 交 通 大 學

電 機 與 控 制 工 程 學 系

博 士 論 文

A Dissertation
Submitted to Department of Electrical and Control Engineering
College of Electrical Engineering and Computer Science
National Chiao-Tung University
in partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in
Electrical and Control Engineering
June 2007
Hsinchu, Taiwan

中華民國九十六年六月

基於細胞神經網路的仿生型電腦視覺系統

學生：黃朝暉　　　　　　　　　　　　　　　　　　指導教授：林進燈

國立交通大學電機與控制工程學系（研究所）博士班

摘　要

自從 Dr. Ramón y Cajal 開始了對視覺神經細胞的研究以來，包含了眼球與大腦的人類視覺系統 （Human Visual System） 就成了一個有趣的研究主題。許多學者們也已提出了他們的研究成果並指出了一些新的研究議題。至目前為止，人類視覺系統與其生理結構已經大概被了解了。一些神經學家也已經開始分析人類視覺系統在神經學上的機制。然而，由於人類視覺系統相當龐大且複雜。至目前為止，我們仍然尚無法完全地了解、分析、甚至是模擬它。

視覺資訊 (Visual Information) 的旅程始於視網膜。過去數十年來人們相信視網膜的作用就如同照相機的感光底片。實際上，視網膜的功能遠超過我們已經知道的。根據 Werblin 與 Roska 在 2005 年的研究，視網膜提供了相當巨量的視覺前處理 (Preliminary Visual Processing)。在視網膜中，視覺資訊首先被光感受體 (Photo-receptor) 所接收。接下來經過一連串的視覺細胞的協同作用後，處理過的視覺資訊被分成超過 12 個不同的頻道並同時透過視神經節即時地傳送到視覺大腦皮質層作更進一步的處理。

延續這些學者的研究成果，本論文提出了一個基於細胞神經網路 (Cellular Neural Networks) 的仿生型電腦視覺系統 (Bionic Computer Visual System)。本系統包含了兩個模型，分別是視網膜中央小窩電腦計算模型 （Computer Fovea Model)，與視覺大腦皮質層電腦計算模型 (Computer Cortex Model)。這兩個模型均是基於人類視覺系統而開發。其中大部份並以

細胞神經網路加以實現。

為了將所提出的模型以細胞神經網路加以實現，本論文亦提出了一些使用細胞神經網路處理影像的新方法。為了達到與生物體結構一致，本論文提出了一個以六角形狀排列的細胞神經網路（Hexagonal-type Cellular Neural Network）。同時亦提出將在本研究中被應用的數種基於細胞神經網路的穩態中央線性系統（Stable Central Linear System of Cellular Neural Networks）的演算法；包括了 CNN-based Laplace-type Filtering、CNN-based Gaussian-type Filtering、與 CNN-based Gabor-type Filtering。這些演算法將成為所提出的視網膜中央小窩電腦計算模型與視覺大腦皮質層電腦計算模型的基本元件。

視網膜中央小窩電腦計算模型可用於模擬某些視網膜的生理結構。透過研究以及模擬這些生理結構，一些人類視覺系統的特性可以被模仿出來。這些特性形成了天然的影像強化機制。因此可以使用所提出的架構來重現這些影像強化機制。這些機制包括了色彩統一性處理（Color Constancy）與影像銳化處理（Image Sharpness）。

視覺大腦皮質層電腦計算模型可用於模擬某些處理視覺的腦皮質區。其提供了簡單的影像紋理辨識及聯想的機制。透過研究腦部視覺皮質區的作用，部份人類視覺系統處理影像紋理的機制也可以被模擬出來。這些機制包含了影像紋理隔離（Segregration）、分類（Classification）、以及辨識（Identification）。

透過了解這兩個電腦計算模型，部份人類視覺系統中有意義的功能可以被研究、學習、甚至是被模仿。因此，本研究可以提供人機界面領域（Human-Computer-Interaction）的研究一個新的方向。

Bionic Computer Visual System based on Cellular Neural Networks

Student：Chao-Hui Huang Advisors：Dr. Chin-Teng Lin

Department of Electrical and Control Engineering

National Chiao Tung University

ABSTRACT

A Human Visual System, including the eyes and the brain, is always an interesting research topic. Since Dr. Ramón y Cajal initialed the study of the visual neurons, many researchers introduced their research results and indicated more and more issues. Currently, the Human Visual System and its biological structure has been roughly understood. Some neurologists also started to analyze the neurological mechanisms of the Human Visual System. However, the Human Visual System is an extremely huge and complex system; we are not able to fully analyze, to understand, or even to simulate it, yet.

The journey of visual information is started at a retina. For dacades, people believe that the retina is similar to the silver pieces in the film of a camera that react to light. In fact, the functions of the retina are much more than what we already knew. According to Werblin and Roska (2005), the retina provides a huge number of preliminary visual processings. In the retina, first, the visual information has been accepted by the photo-receptors. After the co-operations of some visual neurons have been finished, the visual information is separated into more than 12 channels and are transmitted to the visual cortex of the brain via some neural axises for further processing in real-time.

Following the works of those researchers, in this thesis, a Bionic Computer Visual System based on Cellular Neural Networks (CNNs) has been introduced. This system includes two major parts: a Computer Fovea Model and a Computer Cortex Model. Those two models are based on the biological structures of the Human Visual System and they are implemented based on the Cellular Neural Networks.

For the sake of implementing the proposed models on the Cellular Neural Networks, some advanced approaches for CNN-based image processing are also introducted. In order to consist with the biological structures, a hexagonal-type Cellular Neural Network is introduced. Meanwhile, some operators, which are based on the Stable Central Linear System of the Cellular Neural Networks, are also presented and are used in this thesis. Those operators include CNN-based Gaussian-type filtering, CNN-based Laplace-type filtering and CNN-based Gabor-type filtering. They became the fumdental elements of the proposed Computer Fovea Model and the Computer Cortex Model.

The Computer Fovea Model can be used to simulate the preliminary processing of a retina. Through investigating this model, various properties of the Human Visual System can be simulated. The Human Visual System possesses numerous interesting properties, which provide the natural methods of enhancing visual information. Various visual information enhancing algorithms can be developed using these properties and this model. The proposed algorithms include color constancy, image sharpness.

The Computer Cortex Model provides simple texture recognition and association for the textures on an image. Through investigating the behaviors of this model, some properties of the texture analysis of the Human Visual System also can be simulated, including texture segregation, classification, and identification.

Based on the studies of those two models, some significant functions of the Human Visual

System can be investigated, understood, even simulated. Thus, this study can provide a new method for the field of Human-Computer-Interaction.

ACKNOWLEDGEMENTS

首先感謝指導教授林進燈多年來的指導。對於本論文的完成，除了林教授的指導以外，也非常感謝諸位口試委員寶貴的意見，使得本論文更加完備。此外，感謝 Professor Leon Chua 在學生赴美期間的指導，以及 Professor Harri Reddy、Professor Tamas Roska、Professor Hyongsuk Kim、Dr. Casaba Rekeczky、Dr. Heinz Koeppl、Dr. Sook Yoon、Dr. Heather Levien 等前輩在學生赴美期間給與許多生活與課業上的協助。

在學校方面，感謝蒲博士鶴章學長、劉博士得正學長、鄭博士文昌學長、張博士俊隆學長、世安、政宏等同學在學業上與生活上的幫忙與照顧。

最感謝的，是家人的支持。雙親黃連山與黃林淑香多年來無悔的付出。大姐黃慧寧、大姐夫黃一哲、二姐黃慧雯在經濟上的全力支援，以及未來的牽手周惠美長年以來的支持與鼓勵，使得我無後顧之憂的專心於學業。

謹以本論文獻給我的家人與關心我的師長與朋友們。

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

For decades, numerous scientists have examined the following questions: "How do humans see the world?" and "How do humans experience vision?" To answer these questions, this study proposes a Computer Visual System (CVS) based on the biological structure of Human Visual System (HVS). In this study, we focus on two major biological mechanisms, namely retina and cortex. Certain biological mechanisms of a retina can be simulated using an in-state-of-art architecture named Cellular Neural Network (CNN) since the simality between the biological neurons and the CNNs, and used some soft computing methods to simulate and evaluate the recognition methods for image textures in a HVS. The biological mechanisms include the behaviors of photoreceptors, horizontal cells, ganglions, and their co-operations.

As a highly structured neuron network, a retina extracts and processes the stimulation from an image projected upon it by the optical system of the eye [1-3]. This process is extremely complex. Consequently, analyzing, modeling, and even simulating the retina have been considered highly challenging tasks. To meet these challenges, numerous studies have been published during the past decade: for example, Shah *et al.* studied the information processing procedure of the retina in both the space and time domains [4]. Based on the study of Shah *et al.*, Thiem later proposed a bio-inspired retina model capable of implementing some visual properties in the HVS [5]. Roska and Bálya *et al.* even discussed the parallel structure of retina and proposed an implementation on Cellular Neural Networks (CNNs) [6]. Their study demonstrated that certain visual information enhancing mechanisms exist in the HVS.

A visual cortex is the visual information processor of a human brain. The major parts exist on the upper and lower lips of calcarine sulcus on medical surface of cerebral hemisphere and provide simple texture recognition and association for the textures on an image.

The visual cortex is also able to effortlessly integrate the local features to construct our rich perception of patterns, despite the fact that the visual information is discretely sampled by the retina and the cortex. According to some biological and computational evidences, some kinds of data compression procedure occur at a very early stage in the HVS [2]. Moreover, many physiological evidences imply that some form of this compression mechanism is involved in the mechanism locating the boundaries in the image [7]. In the early 1960s, there was a research result which implied that the majority of neurons in the primary visual cortex respond to a line or a boundary of a certain orientation in a given position of the visual field [2]. In 1981s, Hubel and Wiesel found two types of orientation-selective neuron, one of that is sensitive to the areas of lines and boundaries, called simple cells, and the other is not, called complex cells [8]. The receptive field of the simple cells can be modeled by the Gabor function, and it has been widely used for information extraction, which is the so-called second order feature [7]. The mechanism of second order feature extraction is more commonly known as the filter-rectify-filter cascade. This consists of the early linear filtering subunits, a nonlinearity (*e.g.*, rectification), and a late linear filter.

In this thesis, first the fundamental theories and the applications of the CNN are introduced and some new templates are developed. Next, the computer retina model is discussed. The relationships between the proposed model and a biological structure are presented. Even more, some applications are shown. Third, the computer cortex model is discussed. In this study, several methods are used to implement the proposed model and are followed by several applications. Finally, the conclusions are drawn.

2 FUNDAMENTAL THEORIES

Cellular Neural Network, also known as CNN, has already been proved as a very powerful image processing kernel with artificial intelligent abilities [9, 10]. Since Chua and Yang first introduced CNN, it has been widely applied on many areas. CNN is an alternative fully connected neural network and has evolved into a paradigm for this type of array. As shown in Figure 2-1(a), the dynamic equation of CNN can be represented by the following equation

$$\begin{aligned}\dot{x}_t(i,j) &= -x_t(i,j) + \sum_{k,l\in N_r(i,j)} A(i,j;k,l)\,y_t(k,l) \\ &+ \sum_{k,l\in N_r(i,j)} B(i,j;k,l)\,u(k,l) + Z,\end{aligned} \tag{2-1}$$

where

$$y_t = f(x_t) = \frac{1}{2}\left(|x_t+1| - |x_t-1|\right), \tag{2-2}$$

where Eq.(2-2) can be represented as Figure 2-1(b).

2.1 Stable Central Linear Systems and their Inverse Systems

If the CNN is considered as an image processor then numerous linear properties must be analyzed; for example, the states located in the non-saturated region. Crounse and Chua mentioned how to analyze the CNN-based image processing in the frequency domain [11]. A similar mechanism can be applied to the hexagonal-type CNN [12]. Assuming all of the cells operate in the linear region, that is, $|x_{\mathbf{i}}| < 1$, then $y_{\mathbf{i}} = x_{\mathbf{i}}$. Thus, Eq.(2-1) can be reformulated as

$$\begin{aligned}\frac{d}{dt}x_t(\mathbf{i}) &= -x_t(\mathbf{i}) + \sum_{\mathbf{j}\in N(\mathbf{i})} A(\mathbf{i}-\mathbf{j})\,x_t(\mathbf{j}) \\ &+ \sum_{\mathbf{j}\in N(\mathbf{i})} B(\mathbf{i}-\mathbf{j})\,u(\mathbf{j}) + Z.\end{aligned} \tag{2-3}$$

Let $a_{\mathbf{n}} \in A$, the linearized templates then can be represented by [11]

$$a_{\mathbf{n}} = \begin{cases} A_{\mathbf{n}} - 1, & \mathbf{n} = 0 \\ A_{\mathbf{n}}, & \mathbf{n} \in N_r(\mathbf{i}) \\ 0, & \text{otherwise} \end{cases}, \text{ and } b_{\mathbf{n}} = \begin{cases} B_{\mathbf{n}}, & (\mathbf{n}) \in N_r(\mathbf{i}) \\ 0, & \text{otherwise} \end{cases}. \tag{2-4}$$

The dynamics can then be represented in a convoluted form as

$$\frac{d}{dt} x_t(\mathbf{n}) = a(\mathbf{n}) \otimes x_t(\mathbf{n}) + b(\mathbf{n}) \otimes u(\mathbf{n}) + Z, \tag{2-5}$$

where $\otimes$ represents a hexagonal convolution.

The dynamics can be written into [13]

$$\frac{d}{dt} \tilde{X}_t(\mathbf{w}) = \tilde{A}(\mathbf{w}) \tilde{X}(\mathbf{w}) + \tilde{B}(\mathbf{w}) \tilde{U}(\mathbf{w}) + Z\delta(\mathbf{w}), \tag{2-6}$$

if and only if all of the discrete hexagonal Fourier transforms exist. Regarding the dynamics, assume that infinite time is available and that a discrete hexagonal Fourier transform exists for all $\mathbf{w}$, then Eq.(2-6) can be represented as follows [11]

$$\tilde{X}_\infty(\mathbf{w}) = \tilde{H}(\mathbf{w}) \tilde{U}(\mathbf{w}), \tag{2-7}$$

where

$$\tilde{H}(\mathbf{n}) = \frac{-\tilde{B}(\mathbf{n})}{\tilde{A}(\mathbf{n})}. \tag{2-8}$$

It is already known that if $\tilde{A}(\mathbf{w}) < 0$ for all $\mathbf{w}$, then the central linear system will be stable. Thus, the equilibrium state can be thought of as a version of the input that has been spatially filtered by the hexagonal transfer function $\tilde{H}(\mathbf{w})$ [14].

2.2 Implementing Existed Filters on the Cellular Neural Networks

This study realizes some operators that are required by the proposed model. These operators include CNN-based Laplace-like operators, CNN-based Gaussian-like operators and corresponding inverse operators. These operators are introduced in the following sections.

2.2.1 CNN-based Laplace-like Filtering

A Laplace operator can be represented as

$$l_r(\mathbf{n}) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}. \tag{2-9}$$

The zero-frequency component of the Fourier transform of Eq.(2-9) is zero. This fact implies the inverse operation of Eq.(2-9) is difficult in the discrete environment. Thus, a small positive value ε^2 is added to Eq.(2-9). Accordingly, the Laplace-like operator is modified as follows

$$l_r(\mathbf{n}) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4+\varepsilon^2 & -1 \\ 0 & -1 & 0 \end{bmatrix}. \tag{2-10}$$

Notably, the value of ε approaches zero as $l_r(\mathbf{n})$ approaches a Laplace operator. If and only if the Fourier transform of the Laplace-like operator $l(\mathbf{n})$ exists and is denoted as

$$\mathcal{F}_{\mathrm{w}}\{l(\mathbf{n})\} = \tilde{L}(\mathbf{w}). \tag{2-11}$$

Next, map Eq.(2-11) into Eq.(2-8). Since $\tilde{A}(\mathbf{w}) < 0$ for all $\mathbf{w}$ is required, we choose $\tilde{A}(\mathbf{w}) = -1$. Thus

$$\frac{-\tilde{B}(\mathbf{w})}{\tilde{A}(\mathbf{w})} = \tilde{L}(\mathbf{w}) = \frac{-\tilde{L}(\mathbf{w})}{-1}. \tag{2-12}$$

According to Eq.(2-4), the coefficients of the system that has been described in Eq.(2-12) can be represented as

$$a(\mathbf{n}) = -\delta(\mathbf{n}), \text{ and } b(\mathbf{n}) = l(\mathbf{n}). \tag{2-13}$$

Thus, the templates are given by $A = a(\mathbf{n}) + \delta(\mathbf{n})$, and $B = b(\mathbf{n})$, where $\delta(\cdot)$ represents *Dirac Delta function*. Finally, for the CNN-based Laplace-like operator, the templates are as follows

$$A_r = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \text{ and } B_r = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4+\varepsilon^2 & -1 \\ 0 & -1 & 0 \end{bmatrix}. \tag{2-14}$$

For convenience, this study denotes the CNN-based Laplace-like operator is denoted as follows

$$CNN_{\text{Laplace},\varepsilon}\left(u(\mathbf{k})\right), \tag{2-15}$$

where $u(\mathbf{k})$ is the input, ε is a corresponding parameter, and the initial state of all cells is zero.

In fact, for any CNN, if the initial condition x_0, the template A, and the bias I are zero, then the system is simply a FIR system and the template B can be considered as the FIR operator.

For the inverse Laplace-like operator, the inverse system of Eq.(2-12) can be used. That is

$$\frac{-\tilde{B}(\mathbf{w})}{\tilde{A}(\mathbf{w})} = \tilde{L}^{-1}(\mathbf{w}) = \frac{-1}{-\tilde{L}(\mathbf{w})}. \tag{2-16}$$

Notably, since the condition, $\tilde{A}(\mathbf{w}) < 0$ for all $\mathbf{w}$ must be satisfied, this study sets $\tilde{A}(\mathbf{w}) = -\tilde{L}(\mathbf{w})$. Thus, the templates can be obtained as following

$$A_r = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -\left(3+\varepsilon^2\right) & 1 \\ 0 & 1 & 0 \end{bmatrix}, \text{ and } B_r = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{2-17}$$

For convenience, the CNN-based Inverse Laplace-like Operator is denoted here as follows

$$CNN^{-1}_{\text{Laplace},\varepsilon}\left(u\left(\mathbf{k}\right)\right), \tag{2-18}$$

where $u(\mathbf{k})$ denotes the input, and ε represents the corresponding parameter in Eq.(2-14). The initial condition x_0 and the bias I are zero.

2.2.2 CNN-based Gaussian-like Filtering

Kobayashi *et al.* suggested an active resister network for Gaussian-like filtering for image [11]. The proposed system can be described as follows [14, 15]

$$h\left(\mathbf{n}\right) = \frac{v\left(\mathbf{n}\right)}{u\left(\mathbf{n}\right)}, \tag{2-19}$$

where $$u\left(\mathbf{n}\right) = \begin{bmatrix} 1 & -\left(2+\lambda^2\right) & 1 \end{bmatrix},\ v\left(\mathbf{n}\right) = \begin{bmatrix} 0 & -\lambda^2 & 0 \end{bmatrix}. \tag{2-20}$$

The Fourier transforms are denoted as $\mathrm{F}_{\mathbf{w}}\{u(\mathbf{n})\} \triangleq \tilde{U}(\mathbf{w})$, and $\mathrm{F}_{\mathbf{w}}\{v(\mathbf{n})\} \triangleq \tilde{V}(\mathbf{w})$, then the system can be described as

$$\tilde{H}\left(\mathbf{w}\right) = \frac{-\tilde{B}\left(\mathbf{w}\right)}{\tilde{A}\left(\mathbf{w}\right)} = \frac{\tilde{V}\left(\mathbf{w}\right)}{\tilde{U}\left(\mathbf{w}\right)}. \tag{2-21}$$

Thus, the templates can be obtained by

$$A = u(\mathbf{n}) + \delta(\mathbf{n}), \text{ and } B = -v(\mathbf{n}). \tag{2-22}$$

The templates for different architectures are listed below

1-D CNN:

$$A = \begin{bmatrix} 1 & -(1+\lambda^2) & 1 \end{bmatrix}, \text{ and } B = \begin{bmatrix} 0 & \lambda^2 & 0 \end{bmatrix}, \tag{2-23}$$

2-D Rectangular-type CNN:

$$A_r = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -(3+\lambda^2) & 1 \\ 0 & 1 & 0 \end{bmatrix}, \text{ and } B_r = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \lambda^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{2-24}$$

Shi proposed a CNN-based Gabor-type filtering model based on the study of Kobayashi [14]. His model is based on a Gaussian-like operator similar to Eq.(2-24) However, Shi modified the dynamic equation of the CNN to reduce the complexity of the hardware implementation. The proposed model requires no such modification.

For convenience, we denotes the CNN-based Gaussian-like operator as follows

$$CNN_{\text{Gaussian},\lambda}(u(\mathbf{k})), \tag{2-25}$$

where $u(\mathbf{k})$ is the input, λ is the corresponding parameter, and the initial condition x_0 and the bias I are zero.

For the inverse Gaussian-like operator, we have

$$\tilde{H}(\mathbf{w}) = \frac{-\tilde{B}(\mathbf{w})}{\tilde{A}(\mathbf{w})} = \frac{\tilde{U}(\mathbf{w})}{\tilde{V}(\mathbf{w})}. \tag{2-26}$$

Since the condition $\tilde{V}(\mathbf{w}) < 0$ for all $\mathbf{w}$ must be met, it is concluded that $\tilde{B}(\mathbf{w}) = -\tilde{U}(\mathbf{w})$. The templates can be obtained from $A = v(\mathbf{n}) + \delta(\mathbf{n})$ and $B = -u(\mathbf{n})$.

The CNN-based inverse Gaussian-like operators for different architectures are listed as follows

1-D CNN:

$$A' = \begin{bmatrix} 0 & -\lambda^2+1 & 0 \end{bmatrix}, \text{ and } B' = -\begin{bmatrix} 1 & -\left(2+\lambda^2\right) & 1 \end{bmatrix}, \tag{2-27}$$

2-D Rectangular-type CNN:

$$A'_r = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -\lambda^2+1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \text{ and } B'_r = -\begin{bmatrix} 0 & 1 & 0 \\ 1 & -\left(4+\lambda^2\right) & 1 \\ 0 & 1 & 0 \end{bmatrix}, \tag{2-28}$$

Finally, a CNN-based inverse Gaussian-like operator can be denoted as

$$CNN^{-1}_{\text{Gaussian},\lambda}\left(u(\mathbf{k})\right), \tag{2-29}$$

where $u(\mathbf{k})$ denotes the input, λ represents the degree of Gaussian function and the initial condition x_0 and the bias I are zero.

2.2.3 CNN-based Gabor-like Filtering

Gabor proposed an adaptive band-pass filtering method [16], which is a complex exponential function modulated by a Gaussian filter:

$$g(x,y) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\left(x^2+y^2\right)/2\sigma^2} e^{j\left(\omega_{x0} x \omega_{y0} y\right)}, \tag{2-30}$$

where ω_{x_0} and ω_{x_0} represent the angular frequency in x and y directions respectively, and σ is the standard deviation of the Gaussian. The Gabor filter can be used as an adjustable band-pass filter tuned to frequencies near ω_{x_0} and ω_{x_0}.

In 1998, Shi implemented Gabor-type filtering in space and time with CNNs [14]. According to Shi, a low-pass filtering modulating a complex exponential function will result in Gabor-type filtering. In Shi's research, the following templates can be used to represent low-pass filtering in CNN:

$$A=\begin{bmatrix}0 & 1 & 0\\ 1 & -\left(4-\lambda^2\right) & 1\\ 0 & 1 & 0\end{bmatrix},\ B=\begin{bmatrix}0 & 0 & 0\\ 0 & \lambda^2 & 0\\ 0 & 0 & 0\end{bmatrix}, \tag{2-31}$$

and the following equation can represent a low-pass filtering modulating a complex exponential function:

$$A=\begin{bmatrix}0 & e^{-j\omega_{y_0}} & 0\\ e^{j\omega_{x_0}} & -\left(4-\lambda^2\right) & e^{-j\omega_{x_0}}\\ 0 & e^{j\omega_{y_0}} & 0\end{bmatrix},\ B=\begin{bmatrix}0 & 0 & 0\\ 0 & \lambda^2 & 0\\ 0 & 0 & 0\end{bmatrix}. \tag{2-32}$$

Clearly, Shi's CNN-based Gabor-type filter is orthogonal in the domain. This fact implies a problem since in hexagonal-type image processing, the sampled signals present non-orthogonal structure. Thus, the parameters, ω_{x_0} and ω_{y_0}, cannot be referred in hCNN-based Gabor-type filtering.

Here, a similar method can be used. The templates for different architectures are listed below

1-D CNN:

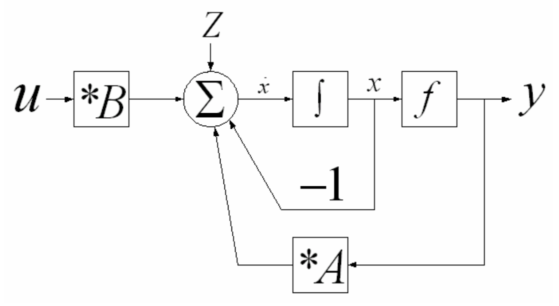
$$A'=\begin{bmatrix}e^{j\omega_0} & -\left(1+\lambda^2\right) & e^{-j\omega_0}\end{bmatrix},\text{ and } B'=\begin{bmatrix}0 & \lambda^2 & 0\end{bmatrix}, \tag{2-33}$$

2-D Rectangular-type CNN:

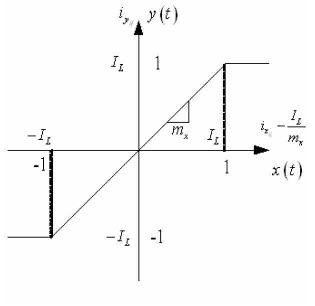
$$A'_r = \begin{bmatrix} 0 & e^{-j\omega_{y0}} & 0 \\ e^{j\omega_{x0}} & -\left(3+\lambda^2\right) & e^{-j\omega_{x0}} \\ 0 & e^{j\omega_{y0}} & 0 \end{bmatrix}, \text{ and } B'_r = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \lambda^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{2-34}$$

Finally, a CNN-based inverse Gabor-like operator can be denoted as

$$CNN^{-1}_{\mathrm{Ga}bor,\omega_{x0},\omega_{y0}}\left(u\left(\mathbf{k}\right)\right). \tag{2-35}$$



(a)



(b)

Figure 2-1. (a) Architecture of CNN; (b) Dynamic route of states in CNN.

3 HEXAGONAL-TYPE CELLULAR NEURAL NETWORK MODEL FOR PRELIMINARY VISUAL PROCESSING

CNNs are especially considered as image processors in many areas. Based on their flexibilities and artificial-intelligence-like abilities, CNNs provide extremely high performance. One of the flexibilities is that CNNs can be considered as one kind of parallel and symmetric structure neural network. Usually, CNNs are considered as eight-connected structure among one cell and its neighbor cells. In fact, CNNs cannot only be considered as eight-connected. If we consider constructing CNN on a flat plane, the cells can be connected as any kind of symmetric structure. One possible structure is six-connected structure, or as the well-known, the hexagonal-type structure.

In fact, constructing image as a hexagonal-type structure is not a new concept. In decades, many researches have examined various aspects of this issue. Peterson initiated the studies of hexagonal sampling [17]. According to Peterson, hexagonal sampling can be considered in the context of a possible alternative sampling regime for a two-dimension Euclidean space. Thus, he concluded that the most efficient sampling schemes were not based on rectangular-type, and same conclusions can explain why the distributions of the cells in the nature present hexagonal-type (see Figure 3-1). Moreover, Rosenfeld proposed distance functions on digital pictures using a variety of coordinate systems [18]. According to Rosenfeld, simple morphological operators were then evaluated in these coordinate systems via some simple applications. Finally, Rosenfeld had a suggestion for displaying the hexagonal-type images by staggering the individual square pixels like the bricks on a wall. The bricks are used to present pixels, and these pixels and their neighborhood construct the hexagonal-type structure. Based on this, Golay developed hexagonal-structured, parallel, computing systems. Other authors also studied the possibilities of using the hexagonal-type structures to represent digital images and graphics. The hexagonal-type structure is considered to be superior to the rectangular-type system in many studies [19-26], and is even proved to be optimal in some applications [26-28].

The degree of symmetry in the hexagonal-type structure is higher than the rectangular-type structure. This fact results in a considerable saving of both storage and computation time [24, 29]. This property also enables the design of simple, accurate, local operators as compared with those currently used in rectangular-type systems. Hexagonal-type pixels are in a uniformly connected, close-packed form, so that greater angular resolution, higher efficiency, and better performance are achieved in many hexagonal-type models [19, 21, 24, 29].

There are some researches mentioned about type representation for CNNs. For example, Radványi (2000) indicates that the rectangular-type structure provides most flexible abilities [30]. However, in the area of image processing, following issues support in initialing the research of CNNs for Hexagonal Image Processing (HIP); first, according to Seiler and Nosse, the symmetry properties of image sampling should be considered. For example, the higher symmetry properties of the hexagonal type lead to a significant reduction in the complexity of the templates [31]. Second, in 1999s, Cauwenberghs and Waskiewicz proposed an analog VLSI focal plane implementation based on hexagonal-type neural networks [32]. In their study, choosing hexagonal-type structure can reduce the complexity of connection between the neighbor cells. Obviously, to consider the cost of connection neighbor cells, the performance of six-connected structure is higher than that of eight-connected one. Third, many studies in the area of image processing mentioned that hexagonal image processing has higher efficiency. According to Nel, the sampling efficiency of triangular-type structure is less than 60.46%, rectangular-type structure is less than 78.56%, and hexagonal-type structure is about 90.69% [33].

Figure 3-2 shows the comparison between different types of the architectures, where Figure 3-2(a), Figure 3-2(b), and Figure 3-2(c), are the 8-connected rectangular-type, the 4-connected rectangular-type, and the 6-connected hexagonal-type CNN respectively, and Figure 3-2(d), Figure 3-2(e), and Figure 3-2(f) are their templates. We can see that in the 8-connected structure, there are some conflicts of connections in the diagonal directions of 8-connected rectangular-type CNN. That is a problem in hardware implementation (see Figure 3-3). This problem was solved based on the architecture of the 4-connected rectangular-type CNN. However, the abilities of the templates are limited. The 6-connected hexagonal-type CNN is not only has no confliction problems, but also has better performance.

Based on the above, we can say that to study the properties of hexagonal-type CNN structure, whichever linear or nonlinear regions, seems still an important issue. This issue leads us to study the field of CNN in hexagonal image processing. In this section, first, we introduce some background and the fundamental of proposed model. Next, the proposed architecture and related knowledge will be presented. Third, some simulations and experiments will be discussed. And finally, the conclusions follow.

3.1 Hexagonal-type Image Sampling Systems

Let $x_a(\mathbf{t})$ be a 2D analog signal which represents the images, where $\mathbf{t} = [t_1 \quad t_2]^T$, and $s(\mathbf{t})$ be a 2D sampling signal and can be represented as follows

$$s(\mathbf{t}) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} \delta(\mathbf{t} - \mathbf{Rn}), \tag{3-1}$$

where

$$\mathbf{n} = [n_1 \quad n_2]^T, \text{ and } \mathbf{R} = [\mathbf{r}_1 \quad \mathbf{r}_2]. \tag{3-2}$$

In Eq.(3-2), $\mathbf{n}$ represents an integer vector, $\mathbf{R}$ is the so-called *sampling matrix*, and $\delta(\cdot)$ represents the *Dirac function*. Sampling matrix plays an important role in a sampling system since it is used to determine how the sampling system works. For example, let $\mathbf{r}_1$ and $\mathbf{r}_2$ be defined as follows

$$\mathbf{r}_1 = \begin{bmatrix} 2T_1 \\ 0 \end{bmatrix}, \text{ and } \mathbf{r}_2 = \begin{bmatrix} -T_1 \\ T_2 \end{bmatrix}. \tag{3-3}$$

Note that T_1 and T_2 are related to the *sampling period* τ, and the type is hexagonal since each sample location has exactly six nearest neighbors when the following equations stand

$$T_1 = \frac{1}{2}\tau, \text{ and } T_2 = \sqrt{3}T_1, \tag{3-4}$$

where τ represents the *sampling period* of the sampling system. In fact, the distance between one sampling point and one of it's six nearest neighbors equals to τ.

Let $x[\mathbf{n}]$ represent the discrete signal which is generated by the sampling system; it can be obtained by the following equation

$$x_s(\mathbf{t}) = x_a(\mathbf{t})s(\mathbf{t}), \tag{3-5}$$

and

$$x[\mathbf{n}] \triangleq x_s(\mathbf{Rn}) = x_a(\mathbf{Rn}), \tag{3-6}$$

where $\mathbf{t} = [t_1 \quad t_2]^T$, $\mathbf{R} = [\mathbf{r}_1 \quad \mathbf{r}_2]$, $\mathbf{n} = [n_1 \quad n_2]^T$, and $x[\mathbf{n}]$ is the discrete sampling signal.

3.2 Re-sampling Images from Rectangular-type to Hexagonal-type

In current display and sampling technology, digital images are usually sampled on rectangular-type structure. It is because sampling on a rectangular-type can be processed as the matrix, and it is suitable for whenever calculating or computing. On the other hand, rectangular-type structure is easier to manufacture. However, we already realized that using hexagonal-type sampling provides several advantages [13, 24]. For example, a hexagon has a twelve-fold symmetry as compared to eight-fold of a square. Due to the improved packing density, hexagonal-type is better suited for representing isotropic and band limited signals. The higher degree of symmetry can also be used to design more isotropic filters to be applied on hexagonal-type [29, 34, 35]. The six neighbors of a hexagonal cell and their connectivity is well-defined [19] and has been used for better edge detection and pattern recognition.

Thus, reconstructing hexagonal-type images from rectangular-type images becomes a more important task. Fortunately, In 2004, Ville and his colleagues proposed a solution for this problem [36]. According to Ville, the image re-sampling between rectangular-type and hexagonal-type can be represented as following: first, an image in the continuous domain is reconstructed; second, resampling the image in the continuous domain based on the desire type.

To construct a spline bases in the hexagonal-type space, the following equations are used

$$\eta_p(\mathbf{x}) = \frac{\eta_1 * \eta_{p-1}(\mathbf{x})}{\Omega}, \quad n \geq 1, \tag{3-7}$$

where $\Omega = |\det(\mathbf{R})|$, where $\mathbf{R}$ is the so-called *sampling matrix* and has been defined in Eq.(3-2). The signal can be represented as

$$S(\eta_n) = \left\{ s(\mathbf{x}) \mid s(\mathbf{x}) = \sum_{\mathbf{k}} c(\mathbf{k}) \eta_n(\mathbf{x} - \mathbf{Rk}); c(\mathbf{k}) \in l_2(\mathbb{Z}^2) \right\}, \tag{3-8}$$

which means that each signal is characterized by its coefficient $c(\mathbf{k})$, which represents being discrete/continuous.

A common way to determine the spline coefficient $c(\mathbf{k})$ is to impose the interpolating condition, $s(\mathbf{Rk}) = g(\mathbf{Rk})$ at the sampling points, where $g(\mathbf{x})$ is the original signal we choose to represent using the splines.

3.3 Hexagonal-type Discrete Fourier Transform

Let $x_a(\mathbf{t})$ be the 2D analog signal which represent the images, where $\mathbf{t} = [t_1 \quad t_2]^T$, and $s(\mathbf{t})$ is 2D hexagonal-type sampling signal and can be represented as follows

$$\begin{aligned}
X_s(j\Omega) = F\{x_s(\mathbf{t})\} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x_s(\mathbf{t}) e^{-j\Omega^T \mathbf{t}} dt_1 dt_2 \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x_a(\mathbf{t}) s(\mathbf{t}) e^{-j\Omega^T \mathbf{t}} dt_1 dt_2 \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x_a(\mathbf{t}) \left(\sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} \delta(\mathbf{t} - \mathbf{Rn}) \right) e^{-j\Omega^T \mathbf{t}} dt_1 dt_2 \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} \left(x_a(\mathbf{Rn}) \delta(\mathbf{t} - \mathbf{Rn}) e^{-j\Omega^T \mathbf{Rn}} \right) dt_1 dt_2 \\
&= \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} \left(x_a(\mathbf{Rn}) e^{-j\Omega^T \mathbf{Rn}} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(\mathbf{t} - \mathbf{Rn}) dt_1 dt_2 \right) \\
&= \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} x_a(\mathbf{Rn}) e^{-j\Omega^T \mathbf{Rn}} = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} x[\mathbf{n}] e^{-j\Omega^T \mathbf{Rn}},
\end{aligned} \tag{3-9}$$

where $\mathbf{t} = [t_1 \quad t_2]^T$, $\mathbf{R} = [\mathbf{r}_1 \quad \mathbf{r}_2]$, and $\mathbf{n} = [n_1 \quad n_2]^T$.

According to Eq.(3-6), and let $\Omega = \omega / \tau$, where τ is the *hexagonal-type sampling period*, the hexagonal-type discrete Fourier transformation can be obtained as follows

$$X_s(e^{j\omega}) = F\{x[\mathbf{n}]\} = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} \left(x[\mathbf{n}] e^{-j\omega^T \mathbf{n}} \right), \tag{3-10}$$

where $\boldsymbol{\omega} = [\omega_1 \quad \omega_2]^T$ and $\mathbf{n} = [n_1 \quad n_2]^T$. The hexagonal-type Fourier transform of $x[\mathbf{n}]$ also can be obtained by the hexagonal-type Fourier transform of $x_a(\mathrm{t})$, $X_a(j\Omega)$. According to Mersereau [37], Eq.(3-10) can be represented as follows

$$X_s\left(e^{j\boldsymbol{\omega}}\right)=\frac{1}{\left|\det \mathbf{R}\right|}\sum_{n_1=-\infty}^{\infty}\sum_{n_2=-\infty}^{\infty}X_a\left(j\left(\mathbf{R}^{-1}\right)^T\left(\boldsymbol{\omega}-2\pi\mathbf{n}\right)\right), \tag{3-11}$$

where

$$X_a\left(j\boldsymbol{\Omega}\right)=\int_{-\infty}^{\infty}x_a\left(\mathbf{t}\right)e^{-j\Omega^T\mathbf{t}}d\mathbf{t}. \tag{3-12}$$

Clearly, the inverse hexagonal-type discrete Fourier transformation can be represented as following

$$x[\mathbf{n}]=F^{-1}\left\{X_s\left(e^{j\boldsymbol{\omega}}\right)\right\}=\frac{1}{(2\pi)^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}X_s\left(e^{j\boldsymbol{\omega}}\right)e^{j\boldsymbol{\omega}^T\mathbf{n}}d\omega_1 d\omega_2\,, \tag{3-13}$$

where $\boldsymbol{\omega}=\left[\omega_1 \quad \omega_2\right]^T$ and $\mathbf{n}=\left[n_1 \quad n_2\right]^T$.

3.4 Hexagonal-type Image Indexing Scheme

In the previous session, we introduced sampling methodology in hexagonal-type image processing. Although the hexagonal-type image processing is more similar to biological phenomenon and it provides more compact structure [5]. The one of disadvantage is that there is no orthogonal axis in hexagonal-type images. Consequently, the mathematical processing in hexagonal-type image processing becomes a difficult task. Fortunately, Middleton and Jayanthi (2001) proposed his research which provides a scheme named Hexagonal Image Processing (HIP) Frameworks [38]. In HIP, the indexing scheme can be constructed intuitively by using an analog based upon tiling the image. Consider a single hexagon to be a tile at layer 0. This can then be surrounded moving anti-clockwise by a further 6 hexagons. This new structure forms a tile at layer 1, with the new tile being the super-tile of layer 0. Similarly, this process can be extended to any number of layers. An example of layer 2 is illustrated in Figure 3-4. The number of hexagons that are contained in a given layer λ super-tile are 7λ. Each hexagon can then be numbered uniquely as a sequence of numbers where every digit gives its position in a given tile. The highlighted hexagon in Figure 3-1 is in the fourth tile in layer 2 and the second tile at layer 1. Thus, all the hexagons in a layer λ super-tile can be addressed uniquely by a λ-digit base 7 number. As previously stated, this number encodes spatial location, and as such, the number can be viewed analogously to a vector. This vector property can be used to define various arithmetic operations such as hexagonal addition and hexagonal multiplication [38].

Since convolution is more computationally efficient when computed in frequency domain, developing the Fourier transformation becomes an important task. There are several methods that can be used to implement Fourier transformation on the hexagonal image. Middleton and Jayanthi proposed their study which is based on Cooley-Turkey algorithm [13, 39]. In their research, the 3-coordinate system was chosen for Discrete Time Fourier Transform in HIP [40]. In this system, a function $c(\cdot)$ is defined that converts from the single index to the 3-coordinate system which has been proposed by Her [40]. For this purpose, the pair of functions is defined as follows

$$h(g)=\frac{1}{3}\sum_{j=1}^{\lambda} N_{j-1}\begin{bmatrix}1 & 1 & -2\\ -1 & 2 & -1\end{bmatrix}c(g_j), \tag{3-14}$$

and

$$H(g)=\frac{1}{3}\sum_{j=1}^{\lambda} (N_{j-1})^T\begin{bmatrix}-1 & 2 & -1\\ 2 & -1 & -1\end{bmatrix}c(g_j), \tag{3-15}$$

where g is a *λ-level* index, g_j corresponds to the *j-th* digit of the index, and N_{j-1} is equivalent to N^{j-1}. Base on those and the previous section, the discrete Fourier transformation for HIP framework can be developed as follows

$$x(g)=\sum_{k\in G^{\lambda}} X(k)e^{2\pi jH(k)^{T}N_{\lambda}^{-1}h(g)}. \tag{3-16}$$

And the inverse function is

$$X(k)=\sum_{g\in G^{\lambda}} x(g)e^{-2\pi jH(k)^{T}N_{\lambda}^{-1}h(g)}. \tag{3-17}$$

3.5 Cellular Neural Networks for Hexagonal Image Processing

In this section, we focus on Hexagonal-type Cellular Neural Network (hCNN). hCNN can be represented as follows

$$\begin{aligned}\dot{x}_t &= -x_t+\sum_{h\in N_r(g)} A(h;g)y_t(h)\\ &+\sum_{h\in N_r(g)} B(h;g)u(h)+I,\end{aligned} \tag{3-18}$$

$$y_t=f(x_t)=\frac{1}{2}\left(|x_t+1|-|x_t-1|\right), \tag{3-19}$$

where g and h represent the index in hexagonal image. Basically, Eq.(3-18) is a special case of Eq.(2-1).

For convenience, we use the following mathematical symbols to present the templates in hCNN

$$A=\left\langle \begin{matrix} & a_3 & & a_2 & \\ a_4 & & a_0 & & a_1 \\ & a_5 & & a_6 & \end{matrix}\right\rangle,\quad B=\left\langle \begin{matrix} & b_3 & & b_2 & \\ b_4 & & b_0 & & b_1 \\ & b_5 & & b_6 & \end{matrix}\right\rangle, \text{ and } I=z, \tag{3-20}$$

where A is the *feedback template*, B is the *control template*, and I is the bias. Note that the indexes in the templates are as same as Middleton's HIP.

3.5.1 The hCNN Stable Central Linear Systems

If we consider the CNN as an image processor, then many properties of the states which have been located in the non-saturated region will need to be analyzed. Crounse and Chua mentioned how to analyze CNN-based image processing in frequency domain. The similar mechanism can be applied on hexagonal-type CNN [11]. Assume the cells operate in the linear region. That is, $|x_g| < 1$, then $y_g = x_g$. Thus, Eq.(3-18) can be reformulated as follows

$$\begin{aligned} \dot{x}_t &= -x_t + \sum_{h \in N_r(g)} A(h;g) x_t(h) \\ &+ \sum_{h \in N_r(g)} B(h;g) u(h) + I, \end{aligned} \tag{3-21}$$

Assume $a_n \in A$, and the notation can be shifted so that spatial indices are written as arguments and time as a subscript. The *linearized template* can be represented as follows

$$a(h) = \begin{cases} a_0 - 1, & h = 0 \\ a_h, & h \in N_r(g) \\ 0, & \text{otherwise,} \end{cases}$$

and

$$b(h) = \begin{cases} B_h, & h \in N_r(g) \\ 0 & \text{otherwise.} \end{cases} \tag{3-22}$$

Then the dynamics can be presented in convolution form as follows

$$\frac{d}{dt} x_t(g) = a(g) \otimes x_t(g) + b(g) u(g) + I\,, \tag{3-23}$$

where $\otimes$ represents hexagonal convolution.

Assume all of the discrete hexagonal Fourier transformations exist. The dynamics can be written into the following

$$\frac{d}{dt}\tilde{X}_t(\omega) = \tilde{A}(\omega)\tilde{X}(\omega) + \tilde{B}(\omega)\tilde{U}(\omega) + I\delta(\omega). \tag{3-24}$$

For the dynamics, assume the time goes to infinite and all of the discrete hexagonal Fourier transformations exist, then it can be represents as follows

$$\tilde{X}_\infty(\omega) = \tilde{H}(\omega)\tilde{U}(\omega),$$

where

$$\tilde{H}(\omega) = \frac{-\tilde{B}(\omega)}{\tilde{A}(\omega)}. \tag{3-25}$$

We already know that if $A(\omega) < 0$ for all ω, then the central linear system will be stable. Thus, the equilibrium state can be thought of as a version of the input that has been spatially filtered by the *hexagonal transfer function* $\tilde{H}(\omega)$.

3.5.2 Implement hexagonal-type CNN on rectangular-type CNN

Since rectangular-type eight-connected CNN structure can provide most efficient flexibility [30], most CNN technology currently is based on this structure. This fact implies that we can implement hexagonal-type CNN by current technology. However, there are some fundamental different between rectangular-type eight-connected CNNs and hexagonal-type six-connected CNNs. Hence, some extra processing is required.

In this section, we propose two ways which can be considered to implement hexagonal-type six-connected CNN on a rectangular-type eight-connected CNN: one is image-rotated method;

another is image-expended method. Those two methods will be introduced as the following sections.

3.5.2.1 Image-screwed Method

According to Radványi, the hexagonal-type structure can be implemented quite easily by rectangular-type eight-connected CNNs, since the connection of neighbor cells can be reconnected based on graph theories [30]. In the case of hexagonal-type CNNs, the templates can be referred as

$$A=\begin{bmatrix}0 & a_2 & a_1\\ a_3 & a_0 & a_6\\ a_4 & a_5 & 0\end{bmatrix},\ B=\begin{bmatrix}0 & b_2 & b_1\\ b_3 & b_0 & b_6\\ b_4 & b_5 & 0\end{bmatrix},\text{ and } I=z\,. \tag{3-26}$$

Clearly, the input and initial-state images of the CNNs should rotate $\frac{1}{4}\pi$. The relationship between the templates and the images can be represented as Figure 3-5.

This method seems quite efficient. The cost is that the input and initial-state images are about twice size and pre-processing, which is used to rotate the input and initial-state image, has been required. However, considering the complexity of image rotating algorithm, realizing this method becomes very difficult. Moreover, this method will waste huge amount of memory if we use general CNN-UM.

3.5.2.2 Image-expended Method

If the 5×5 CNN array is available, and the buffer size of CNN array is big enough, then this method should be considered. Based on the requirement of symmetric property, the hexagonal-type CNN can be implemented by the following 5×5 templates

$$A=\begin{bmatrix}0 & a_3 & 0 & a_2 & 0\\ 0 & 0 & 0 & 0 & 0\\ a_4 & 0 & a_0 & 0 & a_1\\ 0 & 0 & 0 & 0 & 0\\ 0 & a_5 & 0 & a_6 & 0\end{bmatrix},\quad B=\begin{bmatrix}0 & b_3 & 0 & b_2 & 0\\ 0 & 0 & 0 & 0 & 0\\ b_4 & 0 & b_0 & 0 & b_1\\ 0 & 0 & 0 & 0 & 0\\ 0 & b_5 & 0 & b_6 & 0\end{bmatrix},\quad Z=z\,. \tag{3-27}$$

Obviously, the input and initial state images I can be reconstructed as follows

$$I=\begin{bmatrix}p_{0,0} & 0 & p_{0,1} & 0 & \cdots & p_{0,N} & 0\\ 0 & 0 & 0 & 0 & \cdots & 0 & 0\\ 0 & p_{1,0} & 0 & p_{1,1} & \cdots & 0 & p_{1,N}\\ 0 & 0 & 0 & 0 & \cdots & 0 & 0\\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots\\ 0 & 0 & 0 & 0 & \cdots & 0 & 0\\ p_{M,0} & 0 & p_{M,1} & 0 & \cdots & p_{M,N} & 0\end{bmatrix}, \tag{3-28}$$

where $p_{i,j}$ represents the intensity of pixel in specific location (i,j) in the hexagonal images. Clearly, since the template which has been described in Eq.(3-27) is fully symmetric in all direction, the output and state images also can be described as Eq.(3-28). That means after the iteration of CNN converged, the output images also should be reconstructed.

In this method, we can see that the 5×5 templates are required, and the size of CNN array is about 4 times of the size of input and initial-state images. It seems the cost of this method is higher than the previous one. However, considering the complexity of the pixel relocating algorithm, the image-expended method might be the better choice.

Based on the proposed architecture, a hexagonal image sampling method has been required. There are several methods can be applied on the proposed architecture. Consider the balance between the complex computation and the precise in geometry. We choose a method as shown in Figure 3-6 and it can be represented as following

$$(x,y)\in\left\{(x,y)\,|\,x\in\mathbf{E}\text{ if }\frac{y}{2}\in\mathbf{E}\text{ and }x\in\mathbf{O}\text{ if }\frac{y}{2}\in\mathbf{O},y\in\mathbf{E}\right\},\qquad(3\text{-}29)$$
$$\mathbf{E}+\mathbf{O}=\mathbf{R},$$

where $\mathbf{E}$ represents the domain of evens, and $\mathbf{O}$ represents the domain of odds in the domain of real numbers $\mathbf{R}$.

However, assume every sampling pixel is perfect square, then the geometrical relationship among pixels can be represented as Figure 3-7. As we can see, the sampling rate will not be consistent in different direction. The error of distance between different directions is about 11.8%. Even so, the phenomenon can be compensated if we made the sampling rate of vertical direction higher than the horizontal direction. For example, we can make the sampling rate on vertical direction equal to $2/\sqrt{3}$ times of horizontal direction.

3.5.3 Hexagonal Image Presentation

Currently, implementing hexagonal-type structure in a rectangular display system is not an easy task. In our simulations, we used multiple pixels to present a hexagonal-type. On the other words, the rectangular display system, such as a computer monitor, has been considered as a digital signal sampling processor, and the resolution of the sampling processor is higher than the signals we processed.

Most of current image display technologies use rectangular-type for ease of calculation or presentation of the image. In order to represent an image described by the hexagonal-type, we used pseudo hexagonal pixel in the experiments. The pseudo hexagonal pixel is composed of small rectangular pixel which is widely used in current display technology as shown in Figure 3-8. With this representation, the aspect ratio of an image changes from 1:1 to 1:1.17.

We can compare the results between the image reconstructions by hexagonal-type pixels and by rectangular-type pixels. An image of plate has been shown in Figure 3-9. We can see that the reconstruction by hexagonal-type and rectangular-type present different result. Also, one of the comparisons of natural image can be found in Figure 3-14. We can find hexagonal-type image presents better performance in the edge.

3.5.4 Implementing Existed CNN-based Application on hCNN

Since CNN has been introduced to the world, it has been widely applied on many fields. Consequently, huge numbers of applications have been published. However, most of them are based on rectangular-type CNN. According to many published research results, hexagonal-type image can produce better performance. If CNN is considered as an image processor for hexagonal-type image, then it is possible to improve the performance of existing CNN applications. Based on this, we choose some well-known CNN-based applications and re-implemented them on hCNN.

3.5.4.1 hCNN-based Laplace-like Filtering

A Laplace operator can be represented as

$$l_h(\mathbf{k}) = \left\langle \begin{array}{ccccc} & -1 & & -1 & \\ -1 & & 6 & & -1 \\ & -1 & & -1 & \end{array} \right\rangle. \tag{3-30}$$

The zero-frequency component of the Fourier transform of Eq.(3-30) is zero. This fact implies the inverse operation of Eq.(3-30) is difficult in the discrete environment. Thus, a small positive value ε^2 is added to Eq.(3-30). Accordingly, the Laplace-like operator is as follows

$$l_h(\mathbf{k}) = \left\langle \begin{array}{ccccc} & -1 & & -1 & \\ -1 & & 6+\varepsilon^2 & & -1 \\ & -1 & & -1 & \end{array} \right\rangle. \tag{3-31}$$

If and only if the Fourier transform of the Laplace-like operator $l(\mathbf{n})$ exists and is denoted as

$$\mathcal{F}_{\mathbf{w}}\left\{l(\mathbf{n})\right\} = \tilde{L}(\mathbf{w}). \tag{3-32}$$

Next, map Eq.(3-32) into Eq.(3-25). Since $\tilde{A}(\mathbf{w}) < 0$ for all $\mathbf{w}$ are required, we choose $\tilde{A}(\mathbf{w}) = -1$. Thus

$$\frac{-\tilde{B}(\mathbf{w})}{\tilde{A}(\mathbf{w})} = \tilde{L}(\mathbf{w}) = \frac{-\tilde{L}(\mathbf{w})}{-1}. \tag{3-33}$$

According to Eq.(3-22), the coefficients of the system that has been described in Eq.(3-33) can be represented as

$$a(\mathbf{n}) = -\delta(\mathbf{n}), \text{ and } b(\mathbf{n}) = l(\mathbf{n}). \tag{3-34}$$

Thus, the templates are given by $A = a(\mathbf{n}) + \delta(\mathbf{n})$, and $B = b(\mathbf{n})$, where $\delta(\cdot)$ represents *Dirac Delta function*. Finally, for the CNN-based Laplace-like operator, the templates are as follows

$$A_h = \left\langle \begin{array}{ccccc} & 0 & & 0 & \\ 0 & & 0 & & 0 \\ & 0 & & 0 & \end{array} \right\rangle, \text{ and } B_h = \left\langle \begin{array}{ccccc} & -1 & & -1 & \\ -1 & & 6+\varepsilon^2 & & -1 \\ & -1 & & -1 & \end{array} \right\rangle. \tag{3-35}$$

For convenience, this study denotes the hCNN-based Laplace-like operator as follows

$$hCNN_{\text{Laplace},\varepsilon}\left(u(\mathbf{k})\right), \tag{3-36}$$

where $u(\mathbf{k})$ denotes the input, ε is a corresponding parameter, and the initial state of all cells is zero.

In fact, for any CNN, if the initial condition x_0, the template A, and the bias I are zero, then the system is simply a FIR system and the template B can be considered as the FIR operator.

For the inverse Laplace-like operator, the inverse system of Eq.(3-33) can be used. That is

$$\frac{-\tilde{B}(\mathbf{w})}{\tilde{A}(\mathbf{w})} = \tilde{L}^{-1}(\mathbf{w}) = \frac{-1}{-\tilde{L}(\mathbf{w})}. \qquad (3\text{-}37)$$

Notably, since the condition, $\tilde{A}(\mathbf{w}) < 0$ for all $\mathbf{w}$ must be satisfied, this study sets $\tilde{A}(\mathbf{w}) = -\tilde{L}(\mathbf{w})$. Thus, the templates can be obtained as following

$$A_h = \left\langle \begin{matrix} & 1 & & 1 & \\ 1 & & -\left(5+\lambda^2\right) & & 1 \\ & 1 & & 1 & \end{matrix} \right\rangle, \text{ and } B_h = \left\langle \begin{matrix} & 0 & & 0 & \\ 0 & & 1 & & 0 \\ & 0 & & 0 & \end{matrix} \right\rangle. \qquad (3\text{-}38)$$

For convenience, the hCNN-based Inverse Laplace-like Operator is denoted here as follows

$$hCNN^{-1}_{\text{Laplace},\varepsilon}\left(u(\mathbf{k})\right), \qquad (3\text{-}39)$$

where $u(\mathbf{k})$ denotes the input, and ε represents the corresponding parameter in Eq.(3-35). The initial condition x_0 and the bias I are zero.

3.5.4.2 hCNN-based Gaussian-like Filtering

Kobayashi *et al.* suggested an active resister network for Gaussian-like filtering for the image [11]. The proposed system can be described as follows [14, 15]

$$h(\mathbf{n}) = \frac{v(\mathbf{n})}{u(\mathbf{n})}, \qquad (3\text{-}40)$$

where

$$u(\mathbf{n}) = \begin{bmatrix} 1 & -\left(2+\lambda^2\right) & 1 \end{bmatrix}, v(\mathbf{n}) = \begin{bmatrix} 0 & -\lambda^2 & 0 \end{bmatrix}. \qquad (3\text{-}41)$$

The Fourier transforms are denoted as $\mathrm{F}_{\mathbf{w}}\{u(\mathbf{n})\} \triangleq \tilde{U}(\mathbf{w})$, and $\mathrm{F}_{\mathbf{w}}\{v(\mathbf{n})\} \triangleq \tilde{V}(\mathbf{w})$, then the system can be described as

$$\tilde{H}(\mathbf{w}) = \frac{-\tilde{B}(\mathbf{w})}{\tilde{A}(\mathbf{w})} = \frac{\tilde{V}(\mathbf{w})}{\tilde{U}(\mathbf{w})}. \tag{3-42}$$

Thus, the templates can be obtained by

$$A = u(\mathbf{n}) + \delta(\mathbf{n}), \text{ and } B = -v(\mathbf{n}). \tag{3-43}$$

Hence, the 2-D hCNN-based Gaussian-like Operator can be represented as

$$A_h = \left\langle \begin{matrix} & 1 & 1 & \\ 1 & -\left(5+\lambda^2\right) & 1 & \\ & 1 & 1 & \end{matrix} \right\rangle, \text{ and } B_h = \left\langle \begin{matrix} & 0 & 0 & \\ 0 & \lambda^2 & 0 & \\ & 0 & 0 & \end{matrix} \right\rangle. \tag{3-44}$$

For convenience, this study denotes the CNN-based Gaussian-like operator as follows

$$hCNN_{\text{Gaussian},\lambda}\left(u(\mathbf{k})\right), \tag{3-45}$$

where $u(\mathbf{k})$ is the input, λ is the corresponding parameter, and the initial condition x_0 and the bias I are zero.

For the inverse Gaussian-like operator, we have

$$\tilde{H}(\mathbf{w}) = \frac{-\tilde{B}(\mathbf{w})}{\tilde{A}(\mathbf{w})} = \frac{\tilde{U}(\mathbf{w})}{\tilde{V}(\mathbf{w})}. \tag{3-46}$$

Since the condition $\tilde{V}(\mathbf{w}) < 0$ for all $\mathbf{w}$ must be met, it is concluded that $\tilde{B}(\mathbf{w}) = -\tilde{U}(\mathbf{w})$. The templates can be obtained from $A = v(\mathbf{n}) + \delta(\mathbf{n})$ and $B = -u(\mathbf{n})$.

The hCNN-based inverse Gaussian-like operator can be represented as

$$A_h' = \left\langle \begin{array}{ccc} & 0 \quad 0 & \\ 0 & -\lambda^2+1 & 0 \\ & 0 \quad 0 & \end{array} \right\rangle, \text{ and } B_h' = \left\langle \begin{array}{ccc} & 1 \quad 1 & \\ 1 & -\left(6+\lambda^2\right) & 1 \\ & 1 \quad 1 & \end{array} \right\rangle. \tag{3-47}$$

Finally, a CNN-based inverse Gaussian-like operator can be denoted as

$$hCNN_{\text{Gaussian},\lambda}^{-1}\left(u\left(\mathbf{k}\right)\right), \tag{3-48}$$

where $u(\mathbf{k})$ denotes the input, λ represents the degree of Gaussian function and the initial condition x_0 and the bias I are zero.

3.5.4.3 hCNN-based Gabor-type Filtering

Gabor proposed an adaptive band-pass filtering method [16], which is a complex exponential function modulated by a Gaussian filter:

$$g\left(x,y\right)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\left(x^2+y^2\right)/2\sigma^2}e^{j\left(\omega_{x_0}x\omega_{y_0}y\right)}, \tag{3-49}$$

where ω_{x_0} and ω_{x_0} represent the angular frequency in x and y directions respectively, and σ is the standard deviation of the Gaussian. The Gabor filter can be used as an adjustable band-pass filter tuned to frequencies near ω_{x_0} and ω_{x_0}.

In 1998, Shi implemented Gabor-type filtering in space and time with CNNs [14]. According to Shi, a low-pass filtering modulating a complex exponential function will result in Gabor-type filtering. In Shi's research, the following templates can be used to represent low-pass filtering in CNN:

$$A=\begin{bmatrix} 0 & 1 & 0 \\ 1 & -\left(4-\lambda^2\right) & 1 \\ 0 & 1 & 0 \end{bmatrix}, \quad B=\begin{bmatrix} 0 & 0 & 0 \\ 0 & \lambda^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{3-50}$$

and the following equation can represent a low-pass filtering modulating a complex exponential function:

$$A = \begin{bmatrix} 0 & e^{-j\omega_{y0}} & 0 \\ e^{j\omega_{x0}} & -\left(4-\lambda^2\right) & e^{-j\omega_{x0}} \\ 0 & e^{j\omega_{y0}} & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \lambda^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{3-51}$$

Clearly, Shi's CNN-based Gabor-type filter is orthogonal in the domain. This fact implies a problem since in hexagonal-type image processing, the sampled signals present non-orthogonal structure. Thus, the parameters, ω_{x_0} and ω_{y_0}, cannot be referred in hCNN-based Gabor-type filtering.

Assume ω_{u_0}, ω_{v_0}, and ω_{u_0}, represent angular frequencies in the horizontal and the two diagonal directions. Then, the proposed hCNN-based Gabor-type filter can be presented as the following

$$A = \left\langle \begin{matrix} & e^{-j\omega_{u0}} & & e^{-j\omega_{v0}} & \\ e^{j\omega_{w0}} & & -\left(6-\lambda^2\right) & & e^{-j\omega_{w0}} \\ & e^{j\omega_{v0}} & & e^{j\omega_{u0}} & \end{matrix} \right\rangle, \quad B = \left\langle \begin{matrix} & 0 & & 0 & \\ 0 & & \lambda^2 & & 0 \\ & 0 & & 0 & \end{matrix} \right\rangle, \tag{3-52}$$

where the parameters ω_{x_0} and ω_{y_0} in fact can be obtained by the linear combination in frequency domain. Assume we let the sampling conditions in the horizontal direction is the same in both the orthogonal-type and hexagonal-type CNNs, the following procedure can help us to obtain the desired parameters. Let $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$ represent unit vectors on the horizontal and two diagonal directions, *i.e.*,

$$\mathbf{U} = (1,0), \quad \mathbf{V} = \left(\tfrac{1}{2}, \tfrac{\sqrt{3}}{2}\right), \text{ and } \mathbf{W} = \left(\tfrac{-1}{2}, \tfrac{\sqrt{3}}{2}\right). \tag{3-53}$$

Let $\mathbf{\Omega} = \left(\omega_{x_0}, \omega_{y_0}\right)$ represent the vector which is combined by the angular frequencies in x and

y directions. Thus, ω_{u_0}, ω_{v_0}, and ω_{u_0} can be obtained by inner product:

$$\omega_{u_0} = \mathbf{\Omega} \cdot \mathbf{U}\,,\ \ \omega_{v_0} = \mathbf{\Omega} \cdot \mathbf{V}\,,\text{ and }\ \omega_{u_0} = \mathbf{\Omega} \cdot \mathbf{W}\,. \tag{3-54}$$

Finally, a hCNN-based Gabor-like operator can be denoted as

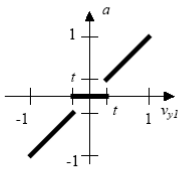
$$hCNN_{Gabor,\omega_{x0},\omega_{y0}}\left(u\left(\mathbf{k}\right)\right). \tag{3-55}$$

3.5.4.4 hCNN-based Retinex Model

Light intensities in an image carry information about the reflectance properties of surfaces from which light was reflected. However, sometimes the condition of luminance makes great effect on the obtained image. Therefore, several methods have been proposed to estimate and separate the effect of illumination, such as contrast stretching, histogram equalization, etc. In 1985, an effectual method named Retinex theory was proposed to attack this problem successfully [41, 42]. It was inspired by the receptive field structures of neurophysiology. Therefore, the Retinex theory can not only be used in gray-scale image processing, but also has been applied on color environment and obtained many good results. The first Retinex algorithm for lightness computations was proposed by E. H. Land and McCann in [43]. It was based on the model of the lightness and color perception of human vision system [41]. It deals with compensation for illusion effects in images. The primary goal is to decompose a given image S into two different images, the reflectance image R and the illumination image L, such that, at each point (x,y) in the image domain, $S(x,y) = R(x,y) \cdot L(x,y)$. The benefits of decomposition include the possibility of removing illumination effects of back/front lighting and enhancing shots that include spatially varying illumination such as images that contain indoor and outdoor zones [41].

Zarándy proposed a method to implement Horn's Retinex model based on CNN [44] using the following templates:

$$B_{11}=\begin{bmatrix}-0.125 & -0.125 & -0.125\\ -0.125 & 1 & -0.125\\ -0.125 & -0.125 & -0.125\end{bmatrix}, A_{21}=\begin{bmatrix}0 & 0 & 0\\ 0 & a & 0\\ 0 & 0 & 0\end{bmatrix},$$



$$A_{22}=\begin{bmatrix}0.125 & 0.125 & 0.125\\ 0.125 & 1 & 0.125\\ 0.125 & 0.125 & 0.125\end{bmatrix}. \tag{3-56}$$

Although Horn suggests hexagonal structure for the 2D Retinex model, this algorithm implemented Horn's model on rectangular-type arrangement. Since processing the image in the hexagonal type has many advantages because it can provide more compact structure, it is desirable to implement the 2D Retinex model on hCNN. The proposed 2D Retinex model based on hCNN can be designed as follows. As Zarándy mentioned, the CNN-based Retinex can be separated into two steps (see Figure 3-12). The first step processes the input image using the hCNN with the following templates:

$$A=0,\quad B=\left\langle\begin{array}{ccccc} & -\tfrac{1}{6} & & -\tfrac{1}{6} & \\ -\tfrac{1}{6} & & 1 & & -\tfrac{1}{6}\\ & -\tfrac{1}{6} & & -\tfrac{1}{6} & \end{array}\right\rangle. \tag{3-57}$$

Then a threshold is applied using the hCNN with the following templates:

$$A=\left\langle\begin{array}{ccccc} & \tfrac{1}{6} & & \tfrac{1}{6} & \\ \tfrac{1}{6} & & 1 & & \tfrac{1}{6}\\ & \tfrac{1}{6} & & \tfrac{1}{6} & \end{array}\right\rangle,\quad B=\left\langle\begin{array}{ccccc} & 0 & & 0 & \\ 0 & & 1 & & 0\\ & 0 & & 0 & \end{array}\right\rangle. \tag{3-58}$$

According to Horn, the most suitable sampling method for the Retinex model should be the hexagonal-type method. Hence the proposed hCNN can be used to implement Horn's Retinex model quite perfect as demonstrated in Figure 3-12.

3.6 Experiments

In this section, we use some examples to show the advantages of the hexagonal-type methods and the proposed hexagonal-type CNN. First, we transfer an input image into both the rectangular-type image and hexagonal-type image, and then, we compare the *Peak Signal and Noise Rate (PSNR)* between the input and out images. A 1024x1024 Grey Level Lenna image (Figure 3-14) was chosed as the input image. The *PSNR* can be represented as follows

$$PSNR = 10 \times \log \left(\frac{p^2}{\frac{\sum_{i=1}^{M}\sum_{j=1}^{N}\left(I_{i,j} - O_{i,j}\right)^2}{MN}} \right), \tag{3-59}$$

where p is the peak value of the images, M, N are the size of images, and I, O represent the images. The *PSNR* of a hexagonal-type image is 24.7096, and the *PSNR* of a rectangular-type image is 24.1634. This fact indicates that the hexagonal-type sampling method can provide higher performance.

The comparison between the frequency response of the rectangular-type image and hexagonal-type image has also been analyzed. The testing target images are shown in Figure 3-14. Figure 3-15 shows the results of the comparisons, where Figure 3-15(a) is the error of frequency response in a specific direction between the original image and the rectangular-type images. Figure 3-15(b) is the error between the original and hexagonal-type images. Clearly, the error between the original image and hexagonal-type image is smaller than the other, in higher frequency especially.

In the experiments, we implemented hCNN in MATLAB. The results can be compared with those of the well-known MATLAB toolbox for CNN, *i.e.*, MatCNN. We implement a few image processing functions on both CNN and hCNN for comparisons. Figure 3-16 shows the results, where Figure 3-16(a), (b) are the input images for the rectangular-type CNN and the hexagonal-type

CNN respectively, and Figure 3-16(c), (d) are the output images processed by the rectangular-type CNN and the hexagonal-type CNN, respectively, which performed the CONTOUR_DETECTION function. The templates for the rectangular-type CNN are as follows

$$A=\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B=\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}, I=-0.5, \tag{3-60}$$

and the templates for the hexagonal-type CNN are as follows

$$A=\left\langle \begin{array}{ccccc} & 0 & & 0 & \\ 0 & & 1 & & 0 \\ & 0 & & 0 & \end{array} \right\rangle, B=\left\langle \begin{array}{ccccc} & -1 & & -1 & \\ -1 & & 6 & & -1 \\ & -1 & & -1 & \end{array} \right\rangle, I=-0.5. \tag{3-61}$$

Figure 3-16(e), (f) are the output images processed by the rectangular-type and hexagonal-type CNNs, respectively, which performed the “SMOOTHING with BINARY OUTPUT” function. The templates for the rectangular-type CNN are as follows

$$A=\begin{bmatrix} 0 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 0 \end{bmatrix}, B=\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, I=0, \tag{3-62}$$

and the templates for the hexagonal-type CNN are as follows

$$A=\left\langle \begin{array}{ccccc} & 1 & & 1 & \\ 1 & & 2 & & 1 \\ & 1 & & 1 & \end{array} \right\rangle, B=\left\langle \begin{array}{ccccc} & 0 & & 0 & \\ 0 & & 0 & & 0 \\ & 0 & & 0 & \end{array} \right\rangle, I=0, \tag{3-63}$$

and Figure 3-16(g), (h) present the BINARY_THRESHOLDING function. The templates for the rectangular-type CNN are as follows

$$A=\begin{bmatrix}0&0&0\\0&2&0\\0&0&0\end{bmatrix},B=\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix},I=0, \tag{3-64}$$

and the templates for the hexagonal-type CNN are as follows

$$A=\left\langle\begin{array}{ccccc}&0&&0&\\0&&2&&0\\&0&&0&\end{array}\right\rangle,B=\left\langle\begin{array}{ccccc}&0&&0&\\0&&0&&0\\&0&&0&\end{array}\right\rangle,I=0. \tag{3-65}$$

We also implemented a well-known CNN function, "GLOBAL CONNECTIVITY DETECTION". The templates for the rectangular-type CNN are as follows

$$A=\begin{bmatrix}0&0.5&0\\0.5&3&0.5\\0&0.5&0\end{bmatrix},B=\begin{bmatrix}0&-0.5&0\\-0.5&3&-0.5\\0&-0.5&0\end{bmatrix},I=-4.5, \tag{3-66}$$

and the templates for the hexagonal-type CNN are as follows

$$A=\left\langle\begin{array}{ccccc}&0.5&&0.5&\\0.5&&3&&0.5\\&0.5&&0.5&\end{array}\right\rangle,B=\left\langle\begin{array}{ccccc}&-0.5&&-0.5&\\-0.5&&3&&-0.5\\&-0.5&&-0.5&\end{array}\right\rangle,I=-4.5. \tag{3-67}$$

Figure 3-17 shows the examples and results of this function, where Figure 3-17(a) is the input image of general CNN, Figure 3-17(b) is the input image of hCNN, Figure 3-17(c) is the initial state of general CNN, Figure 3-17(d) is the initial state of hCNN, Figure 3-17(e) is the output image of general CNN, and Figure 3-17(f) is the output image of hCNN.

3.7 Discussions

In this section, we proposed the hexagonal-type CNN (hCNN) for Hexagonal Image Processing

(HIP). HIP contains many advantages; however, some disadvantages still remain. For example, to analyze and compute the processing procedure in HIP is still a difficult task. In this section, we introduced an approach to overcome those disadvantages based on CNN. And some examples have been given to demonstrate how CNN works in Hexagonal Image Processing, called hCNN. In hCNN, each cell connects to only six neighbor cells, but not eight. Consequently the designs of the integrated circuits of CNN-based image processors are able to be simplified. On the other point of view, one of the most important features of CNN is about the nonlinear template design. We hope it will prove useful for the design of nonlinear processing of Cellular Neural Networks in Hexagonal Image Processing.

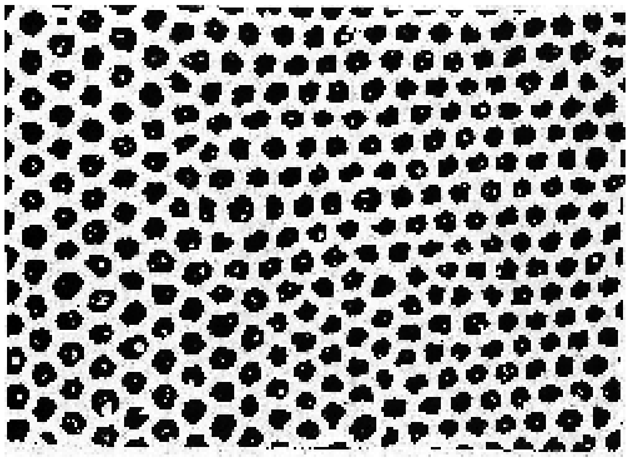
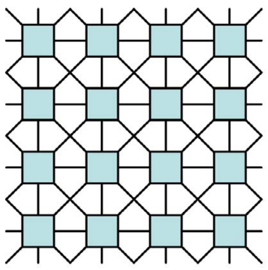
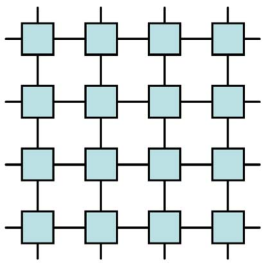
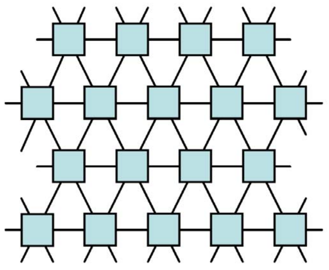


Figure 3-1. Distribution of cones on the retina of mammalian.


(a)


(b)


(c)

$$T = \begin{bmatrix} t_1 & t_2 & t_3 \\ t_4 & t_5 & t_6 \\ t_7 & t_8 & t_9 \end{bmatrix}$$

(d)

$$T = \begin{bmatrix} 0 & t_1 & 0 \\ t_2 & t_3 & t_4 \\ 0 & t_5 & 0 \end{bmatrix}$$

(e)

$$T = \left\langle \begin{array}{ccccc} & t_3 & & t_2 & \\ t_4 & & t_0 & & t_1 \\ & t_5 & & t_6 & \end{array} \right\rangle$$

(f)

Figure 3-2. The comparison between different layout of CNNs, where (a) is rectangular 8-connected rectangular-type CNN, (b) is rectangular 4-connected rectangular-type CNN, and (c) is hexagonal 6-connected hexagonal-type CNN.

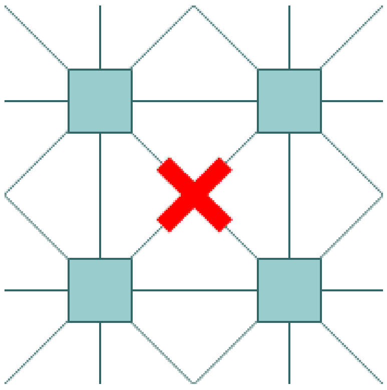


Figure 3-3. The confliction of connection in diagonal direction of 8-connected CNN.

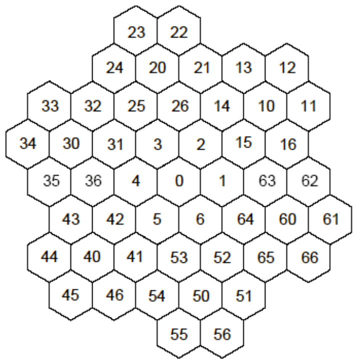


Figure 3-4. Hexagonal-type image sampling and indexing scheme.

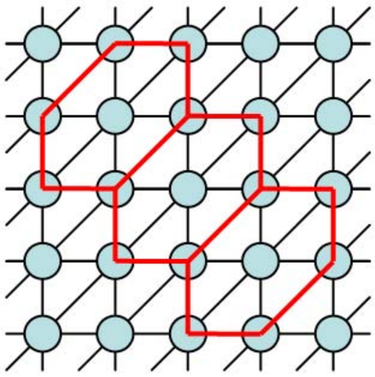


Figure 3-5. An example of hexagonal-type CNN implementation based on rectangular-type CNN. For each cells, the upper-left and lower-right direction of synapses has been disconnected.

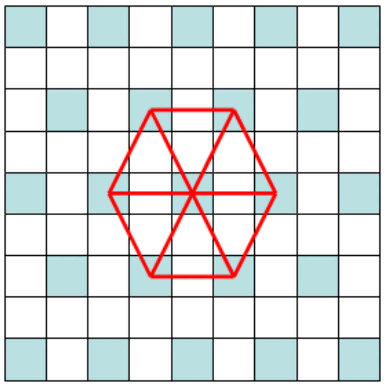


Figure 3-6. The sampling method of proposing model.

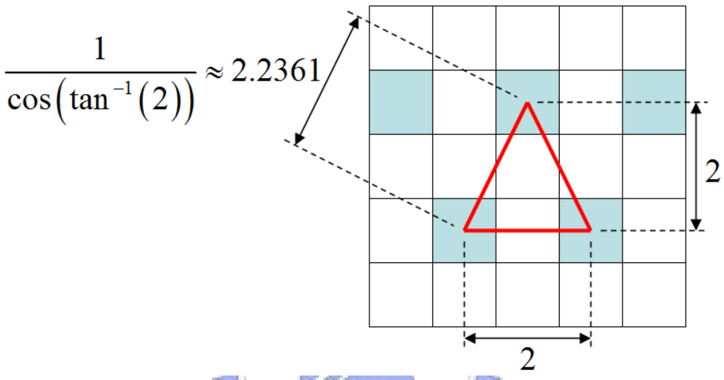


Figure 3-7. The geometrical relationship among pixels, where the error of distance between different directions is about 11.8%.

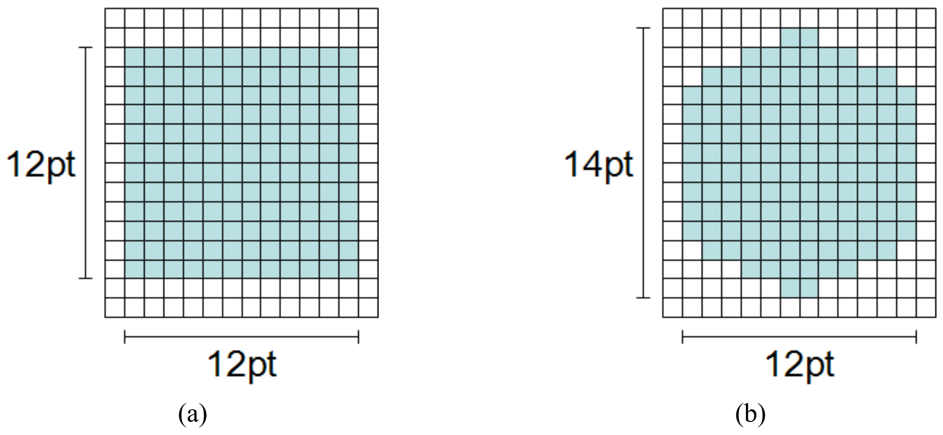


Figure 3-8. The sampling pixel we used in this section, where (a) is rectangular-type pixel, and (b) is hexagonal-type pixel.

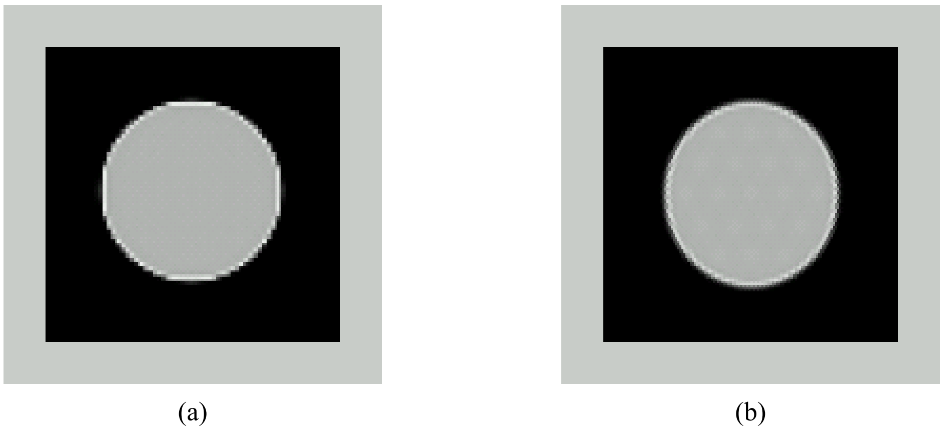


Figure 3-9. The comparison of artificial images between the image which has been constructed by square-shape pixels and the image which has been constructed by hexagonal-shape pixels.

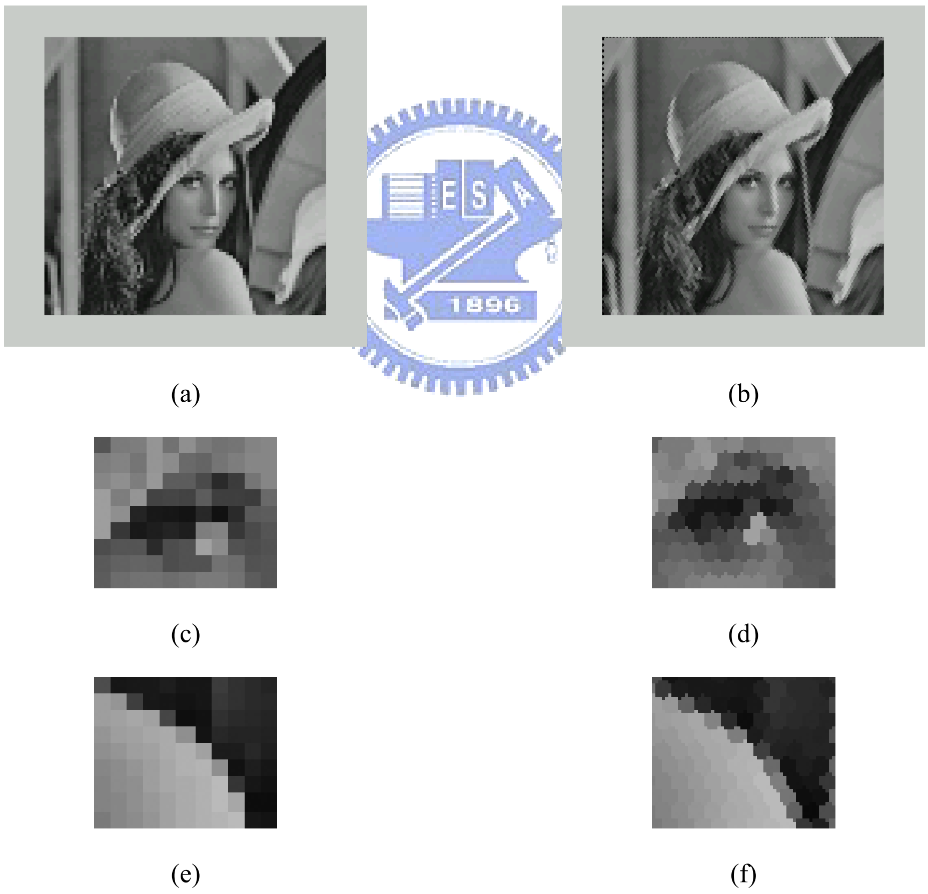


Figure 3-10. The example of different type of sampling methods, where (a) is the rectangular-type image, (b) is hexagonal-type image, (c) and (e) show the details of (a), (d) and (f) show the details of (b).

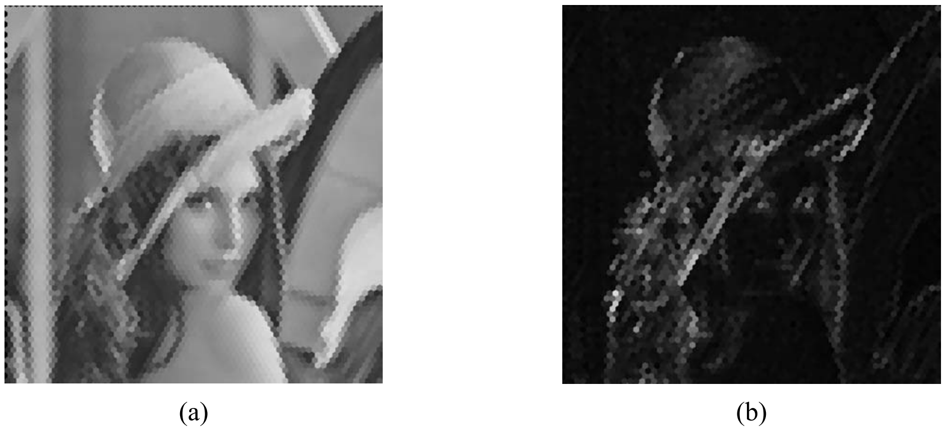


Figure 3-11. The processing result of hCNN-based Gabor-type filtering, where (a) is the input hexagonal-type image and (b) is the output. Note that the resolutions of the images have been reduced and therefore the hexagonal pixels can be exposed.

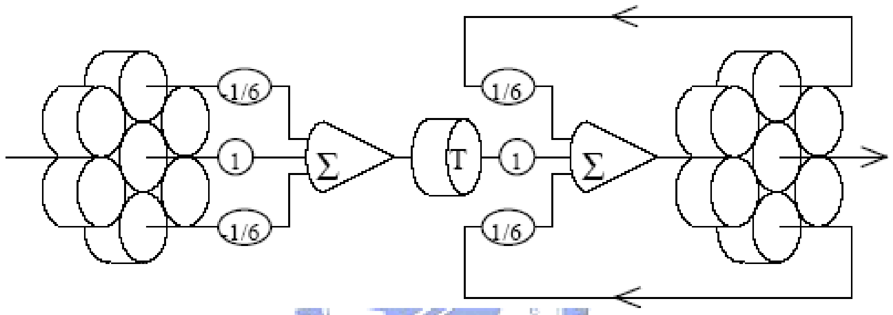


Figure 3-12. The schematic of Horn's model using summing and threshold elements. The feedforward structure calculates the Laplace operator, while the feedback loop structure calculates the inverse Laplace operation. Note that for the shake of clarity of the figure not all the feedback and feedforward interconnections are indicated. This figure has been obtained from [45].

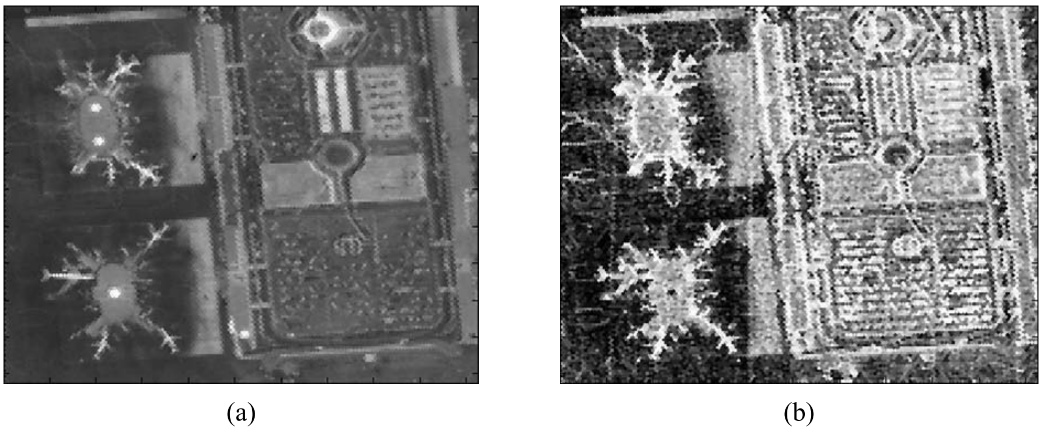


Figure 3-13. The processing result of hCNN-based Retinex-model, where (a) is the input hexagonal-type image, and (b) is the output.

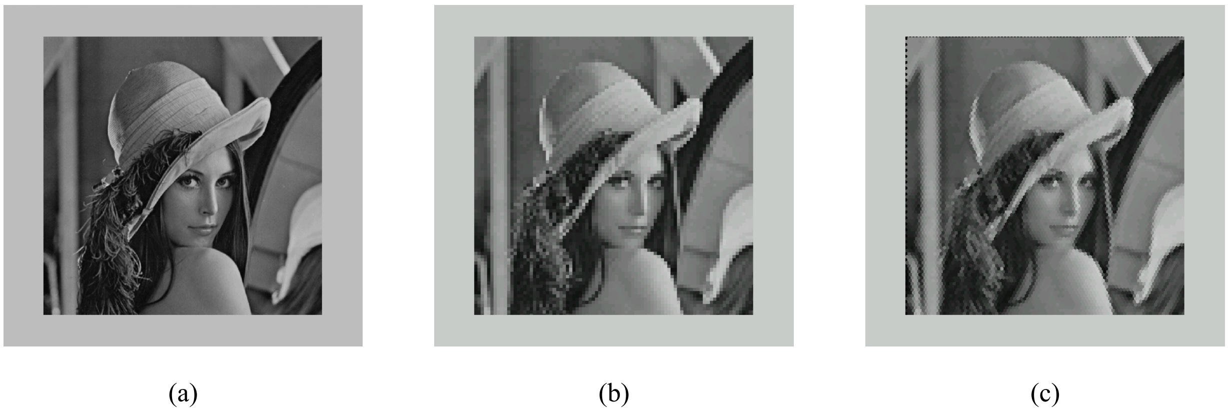


Figure 3-14. The example of different type of sampling methods, where (a) is the original image, (b) is the image which has been sampled by the rectangular-type method, and (c) is the image which has been sampled by the hexagonal-type method.

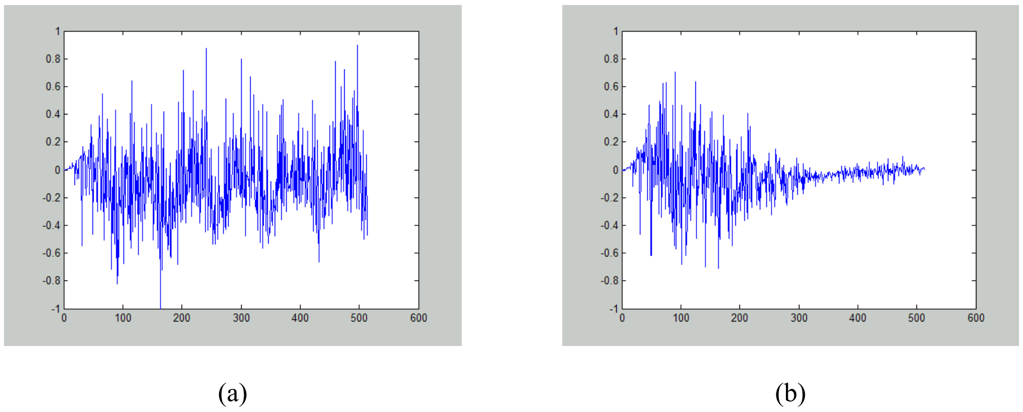


Figure 3-15. The result of comparisons, where (a) is the error between the original image and the rectangular-type image, and (b) is the error between the original and hexagonal-type image. Clearly, the error between the original image and hexagonal-type image is smaller than the other.

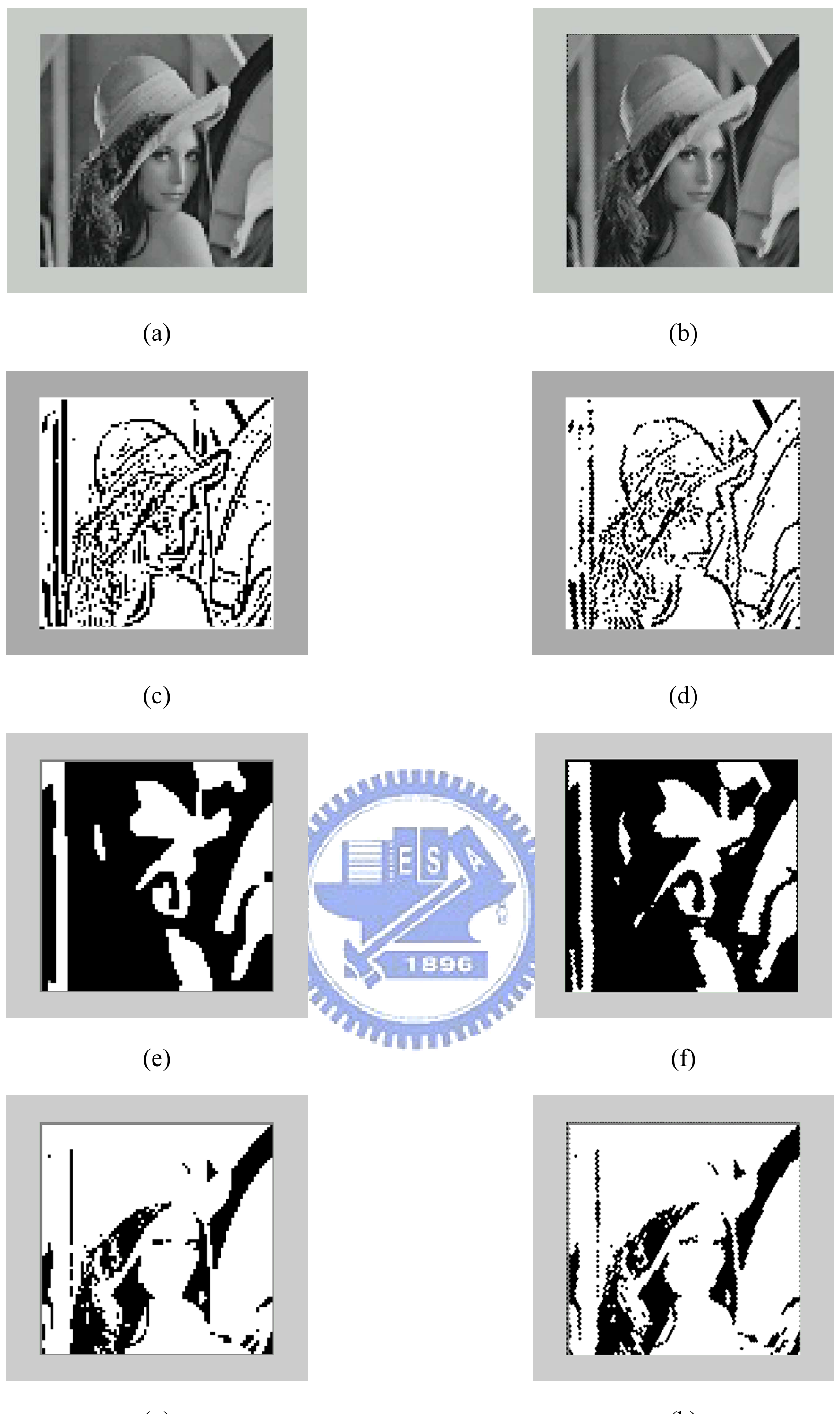


Figure 3-16. Experimental results were shown in this figure, where (a), (b) are the input images for the rectangular-type CNN and the hexagonal-type CNN respectively. First, (c) and (d) are the output images processed by the rectangular-type CNN and the hexagonal-type CNN, respectively, where they present CONTOUR_DETECTION function. Next, (e) and (f) are as same as the above and they present SMOOTHING_with_BINARY_OUTPUT function. Finally, (g) and (h) present BINARY_THRESHOLDING function.

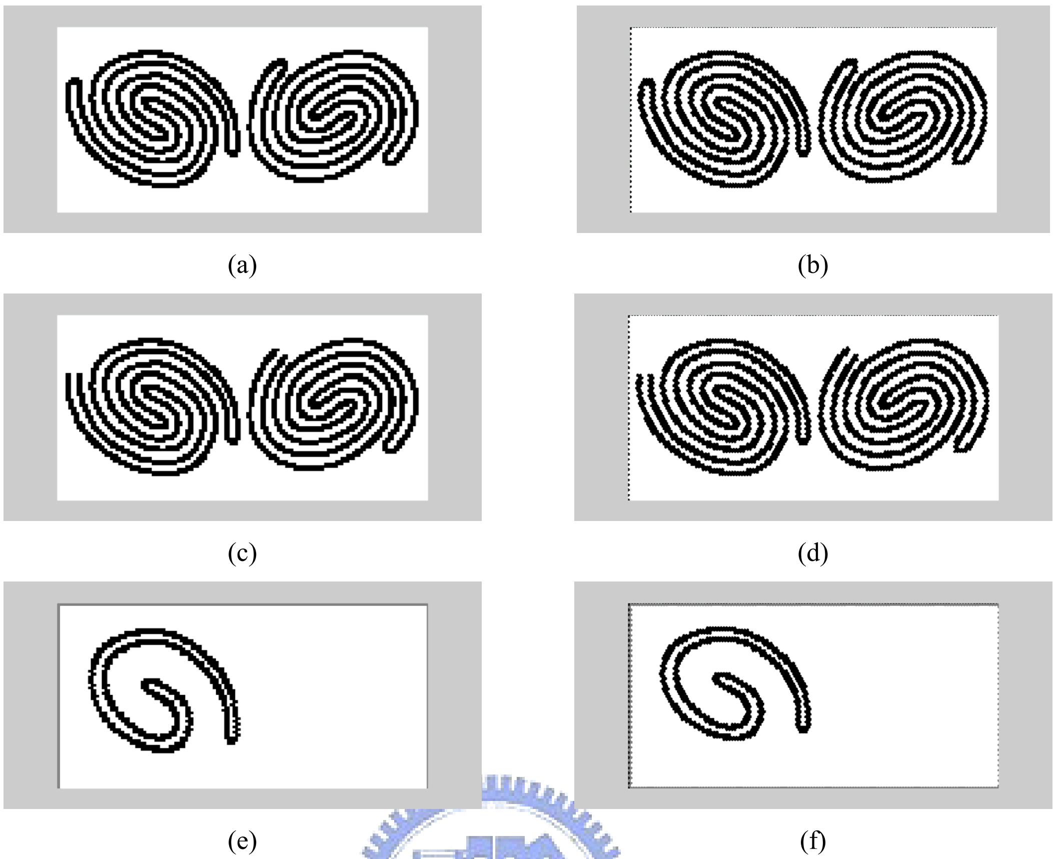


Figure 3-17. The results of GLOBAL_CONNECTIVITY_DETECTION, where (a) is the input image of general CNN, (b) is the input image of hCNN, (c) is the initial state of general CNN, (d) is the initial state of hCNN, (e) is the output image of general CNN, and (f) is the output image of hCNN.

4 CNN-BASED COMPUTER FOVEA MODEL

In this section, first the biological structure of a retina must be understood. Five major types of neurons exist in the five layers of the retina.Outer nuclear layer contains photoreceptors, Inner nuclear layer contains horizontal cells, amacrine cells and bipolar cells, and ganglion layer contains ganglions. Moreover, Outer plexiform layer contains the synapse connections among the photoreceptors, the horizontal cells and the bipolar cells. Finally, inner plexiform layer contains the synapse connections among the bipolar cells, the amacrine cells and the ganglions [1, 46].

The HVS contains four kinds of photoreceptor: L-cone, M-cone, S-cone and rod cells. These different types of photoreceptors react to different wavelengths of light (Figure 4-1). Rod cells can sense light intensity, while L-cone, M-cone, S-cone cells can detect color information. Sometimes the cooperation of these cone cells can also detect the light intensity. We are already aware that The ganglion is usually activated by a set of the photoreceptors via other bipolar, horizontal, and other types of cells [2, 47]. This set contains several varieties of photoreceptors. The differences among those different types of photoreceptors results in the variation among the ganglions. The two main types of ganglions are center-on/surround-off and center-off/surround-on. Figure 4-2 illustrates two examples, where Figure 4-2(a) shows an example of center-surround ganglion. The center of a group of photoreceptors reacts to the stimulation differently to the peripheral part of the group. The ganglions can generally be classified as red-green (RG) ganglion, blue-yellow (BY) ganglion, and black-white (BW) ganglion (see Figure 4-2(b) and (c)) and others [1].

Ganglion is important in the human vision system and processes most visual information. Visual information includes light intensity and color information. The mechanism which is used to process this visual information in the human vision system is so-called early vision system, also known as the pre-attentive vision system. The early vision system represents a set of the first stage information processing mechanisms of visual processing. Those mechanisms are operated in parallel across the visual field, and are believed to be used for detecting certain fundamental visual features [1].

According to Jain and Farrokhnia *et al.*, the human vision system possesses two fundamental features: first, in some respects the retina acts like a low-pass filter. Generally, the result of the low-pass filtering represents an average intensity of light for a specific local area. The result of this operation is termed "the first order feature." Second, a difference exists between the intensity of the external light and that projected into the retina. According to some studies, the boundary detection operation in the human vision system is based on this kind of feature, and is related to the zero crossing of a Laplacian of two Gaussians (LoG). Sometimes the result of this operation is called "the second order feature [48]."

The mechanisms of information processing in the retina are rather complex and remain unclear. This study thus mainly aims to design and approximate the Receptive Fields (RFs) of the cells on a retinal fovea. From the signal processing perspective, the RFs can be referred to as a finite impulse response of a spatial filter [4, 47, 49]. In this investigation, the well-known Cellular Neural Network (CNN) is used to realize the spatial filter.

CNN represents the next generation of computational architecture. Similar to a biological system, each cell in the CNN can communicate only with its immediate neighborhood. Consequently, using the CNN to implement the proposed model makes sense. According to some previous studies, hexagonal image processing is much reasonable for image processing, particularly for bio-inspired models (refer to Figure 4-1) [12, 15, 20, 27, 45]. Thus, this study suggests a special type of CNN) — the hexagonal-type Cellular Neural Network (hCNN). Furthermore, hCNN provides a means of reducing implementation problems without increasing the complexity [12].

Based on the biological investigations and the abilities of the CNN, this work proposes an hCNN-based computer fovea model. The proposed model simulates certain biological mechanisms of the retina and the fovea, including the photoreceptors, bipolar cells, horizontal cells, ganglions, and the co-operations of those cells in a fovea and a retina. Consequently, some properties of the human vision system can be simulated. The human vision system possesses various interesting properties. Some of those properties can even be used to enhance the visual information. This study also presents how these properties provide visual enhancement.

This study first briefly introduces the hCNN, and also discusses the stable central linear system for the hCNN and develops its implementations. Those implementations are required for the proposed model, including CNN-based Laplace-like operators, CNN-based Gaussian-like operators and their inverse operators. Subsequently, the CNN-based computer fovea model is introduced. Building on the above, several experiments are presented, including vision enhancing algorithms based on this model. Finally, conclusions are drawn.

4.1 Modeling the Biological Structures of the Cells in the Fovea and the Retina

Since a retina is a highly structured network of neurons, it has become a valuable research topic in the field of Human Computer Interaction (HCI). Many studies have studied the structure of the retina and the biological evidence regarding the functions of the human vision system [50, 51]. Notably, some researchers have even analyzed visual information processing in the retina [4, 47]. Building on these previous works, this study constructs a new computer fovea model.

The retinal fovea is located at the center of the retina, and is the region with the highest visual acuity. The fovea is a 0.2~0.4 mm diameter rod-free area with very thin, densely packed photoreceptors. The photoreceptors in the fovea are arranged in a roughly hexagonal pattern (see Figure 4-1(a)) [3], and the average cone spacing (csp) has been estimated at around 2.5 to 2.8 μm, where has been considered as the most important area in a retina. The retinal fovea is directed towards the desired object of study. The retinal fovea almost exclusively contains high density cones.

Figure 4-5 shows the proposed hCNN-based computer fovea model, where Figure 4-5(a) illustrates the top-view of the computer fovea. The computer fovea is constructed using a set of photoreceptors, which are hexagonally arranged (see Figure 4-5(b)). Figure 4-5(c) is the signal processing system of the cells in the fovea model. Thiem mentioned that because there are direct synaptic connections between bipolar cell and the ganglion, and only few influences by the amacrin cell in the fovea. Hence, in his research, he suggested that the bipolar cell and the amacrin cell are neglected [5]. However, the horizontal cells are already known to exist and connect directly to the bipolar cells [47]. Thus, this study suggests keeping the bipolar cell and the ganglion in the system, and considering the ganglion as a direct synaptic connection.

A simplified version of the proposed model is required to obtain the parameters of the proposed architecture. The simplified version of the proposed model ignores the differences between the L-cone, the M-cone, and the S-cone cells. Restated, the system considered for obtaining the parameters is assumed to be monochromatic. Thus, $x'_{R'}$ equals x_R. The following sections, discuss behavioral variation among the cone cells.

4.1.1 Ganglion

The ganglions of the mammalian retina have been characterized into X-, Y-, and W-types based on the spatiotemporal distribution of the excitation and inhibition. The cells are classified according to their latency from the optic chiasm stimulations [47]. The X-type ganglions behave linearly in both the space and time domain and also provide the best spatial resolution, but the most ganglions are considered direct synapse connections. Consequently, in this study, the ganglion has been considered as an amplifier which contains a bias g.

Based on the above, $h_G(\mathbf{k})$ can take the form

$$h_G(\mathbf{k}) = g \cdot h_B(\mathbf{k}) = g \cdot h_{RB}(\mathbf{k}) \otimes h_R(\mathbf{k})$$
$$= g \cdot \left(\delta(\mathbf{k}) - b \cdot h_{RH}(\mathbf{k})\right) \otimes h_R(\mathbf{k}). \tag{4-1}$$

Some physiological experiments indicated that the RF of the ganglion exhibits a center/surround characteristic. Furthermore, Thiem stated that the RF of ganglions can be modeled as follows [5]

$$h_G(\mathbf{k}) = \Delta\left(g_{\sigma_G}(\mathbf{k})\right), \tag{4-2}$$

where $\Delta(\cdot)$ represents the Laplace operator, and $g_{\sigma_G}(\cdot)$ is a Gaussian function. According to Hubel, under the optimum lighting condition, the central part of RF is about 10 μm (4 csp) [1]. Thus, Thiem also recommends a standard deviation of $\sigma_G = \frac{5\mu m}{\sqrt{2}} = \sqrt{2}$ (csp) [5].

A combination of the CNN-based Laplace-like operator and the CNN-based Gaussian-like operator can be employed to derive the $h_G(\mathbf{k})$. That is

$$h_G(\mathbf{k}) = CNN_{\text{Laplace},\varepsilon}\left(CNN_{\text{Gaussian},\lambda_G}(\mathbf{k})\right). \tag{4-3}$$

where ε can be any extremely small positive value but not zero, and λ_G can be obtained by performing GA. Based on the experiments performed here, it was concluded that if σ_R equals $\sqrt{2}$ and $g = 1$, then λ_R is approximately 0.567700. (see Table 4-1).

4.1.2 Photoreceptor

An impulse response of a photoreceptor can be represented as a Difference of two Gaussians (DoG). In the retinal fovea, the DoG can be described as a Gaussian function, as shown below, in most cases [4, 5, 47]

$$g_{\sigma_R}(\mathbf{k}) = \frac{1}{\sqrt{2\pi\sigma_R^2}} e^{-\frac{\|\mathbf{k}\|^2}{2\sigma_R^2}}. \tag{4-4}$$

As mentioned in [47], parameter σ_R represents the standard deviation with a range from 1.5 to 12 (csp).

In the proposed approach, a CNN-based Gaussian-like operator is used for the simulation of the Gaussian function. That is

$$CNN_{\text{Gaussian},\lambda_R}\left(\delta(\mathbf{k})\right) \approx g_{\sigma_R}(\mathbf{k}), \tag{4-5}$$

where λ_R indicates the diffusion level of the Gaussian-like function. The next question is how to obtain the parameter λ_R and make the final state of the CNN approximate that shown in Eq.(4-4). To obtain the corresponding value λ_R, Genetic Algorithm (GA) is used again. Based on the results of the experiments, it was concluded that if σ_R equals 1.5, λ_R is approximately 0.536040, and if σ_R equals 12, then λ_R is approximately 0.071233 (see Table 4-1).

Clearly, the RF of the photoreceptor is used to determine the average intensity of a specific area of a visual signal. Generally, it acts like a low-pass filter. The output of the filter, described in Eq.(4-5), is known as the first order feature.

4.1.3 Horizontal Cell

Eq.(4-1) implies that a horizontal cell can be implemented by the following equation

$$h_{RH}(\mathbf{k}) = \frac{1}{b}\cdot\left(\delta(\mathbf{k}) - h_{RB}(\mathbf{k})\right). \tag{4-6}$$

Thus, estimating $h_{RB}(\mathbf{k})$ results in the realization of a horizontal cell $h_{RH}(\mathbf{k})$, if and only if the inverse operator of $h_R(\mathbf{k})$ exists. Thus, $h_{RB}(\mathbf{k})$ can be obtained as

$$h_{RB}(\mathbf{k}) = \frac{1}{g} \cdot h_G(\mathbf{k}) \otimes h_R^{-1}(\mathbf{k}). \tag{4-7}$$

Above it is assumed that the subject system is a monochromatic system. This means that the input of the bipolar and horizontal cells derives from the same set of the photoreceptors. If the inputs of the two types of cells do not derive from the same set of photoreceptors, then the RF of the horizontal cell needs to be known. Assume the input of a horizontal cell derives from photoreceptor system $h_{R'}(\mathbf{k})$ which is illustrated in Figure 4-5(c), then based on Eq.(4-6), $h_{R'H}(\mathbf{k})$ can be obtained as follows

$$h_{R'H}(\mathbf{k}) = \frac{1}{b} \cdot h_{R'}^{-1}(\mathbf{k}) \otimes \left(h_R(\mathbf{k}) - \frac{1}{g} \cdot h_G(\mathbf{k}) \right). \tag{4-8}$$

Based on CNN, the horizontal cell $h_H(\mathbf{k})$ can be implemented by

$$\begin{aligned} &h_H(\mathbf{k}) \\ &= \frac{1}{b} \cdot \left(\delta(\mathbf{k}) - CNN_{\text{Laplace},\varepsilon} \left(\frac{1}{g} \cdot CNN_{\text{Gaussian},\lambda_G} \left(CNN^{-1}_{\text{Gaussian},\lambda_R}(\mathbf{k}) \right) \right) \right). \end{aligned} \tag{4-9}$$

Horizontal cell determines the difference between the output signal of the photoreceptors in the center and the surroundings of a RF. This kind of feature is termed the second order feature. Meanwhile, the final feature involves the determination of the parameters b and g. From a biological perspective, value b represents the related weights of the input signals of the horizontal cells, while g denotes the related weight of the ganglion. Unfortunately the values are unknown. However, considering the cooperation among the photoreceptor, horizontal cell, bipolar cell, and ganglion, it can be concluded that if an input is nothing but gray, which means there is no stimulation, then the output of the ganglions should be zero. This fact can help us to determine the values of b and g in a specific environment.

Finally, this study concludes that the photoreceptors, horizontal cells, and ganglions can be simulated as follows

$$CNN_{\text{Photoreceptor},\lambda_R}\left(u(\mathbf{k})\right) = CNN_{\text{Gaussian},\lambda_R}\left(u(\mathbf{k})\right), \tag{4-10}$$

$$\begin{aligned} &CNN_{\text{Horizontal},\varepsilon,b,g,\lambda_R,\lambda_G}\left(u(\mathbf{k})\right) \\ &= \frac{1}{b}\cdot\left(\delta(\mathbf{k}) - CNN_{\text{Laplace},\varepsilon}\left(\frac{1}{g}\cdot CNN_{\text{Gaussian},\lambda_G}\left(CNN^{-1}_{\text{Gaussian},\lambda_R}\left(u(\mathbf{k})\right)\right)\right)\right), \end{aligned} \tag{4-11}$$

and

$$\begin{aligned} &CNN_{\text{C-Ganglion},\varepsilon,b,g,\lambda_{R_u},\lambda_{R_v},\lambda_G}\left(u(\mathbf{k}),v(\mathbf{k})\right) \\ &\quad = CNN_{\text{Photoreceptor},\lambda_{R_u}}\left(u(\mathbf{k})\right) - CNN_{\text{Horizontal},\varepsilon,b,g,\lambda_R,\lambda_G}\left(v(\mathbf{k})\right). \end{aligned} \tag{4-12}$$

Notably, $u(\mathbf{k})$ and $v(\mathbf{k})$ are the inputs of the photoreceptors. For a monochromatic system, the ganglion function is

$$\begin{aligned} &CNN_{\text{M-Ganglion},\varepsilon,b,g,\lambda_{R_u},\lambda_G}\left(u(\mathbf{k})\right) \\ &\quad = \left(CNN_{\text{Photoreceptor},\lambda_{R_u}}\left(u(\mathbf{k})\right) - CNN_{\text{Horizontal},\varepsilon,b,g,\lambda_{R_u},\lambda_G}\left(u(\mathbf{k})\right)\right). \end{aligned} \tag{4-13}$$

4.2 Experiments

4.2.1 Biology-related Visual Response and Illumination

Figure 4-6 and Figure 4-7 illustrate the impulse response of a horizontal cell in the hCNN-based computer fovea model. Note that Figure 4-6(a) can be compared with the well known RF structure (see Figure 4-2). Notably, Figure 4-7(b) resembles a common illumination in the human vision system.

Figure 4-8 shows another example usign the proposed model, where the input is a natural image. Figure 4-8(a) is the input, Figure 4-8(b) is the output of horizontal cells, Figure 4-8(c) is the output of the ganglions, and finally, Figure 4-8(d) is the output of the rectification. Notably, to present the hexagonal structure, the resolution of the images in this example is reduced.

4.2.2 Simulation of a retina and Central Retinal Vein Occlusion (CRVO)

Color vision is quite an arbitrary experience, and thus a standard distribution of the sensitivity of the L-cone, the M-cone, and the S-cone cells in the human vision system is used for the simulation purpose in this study (see Figure 4-11(a)), and mapped into an image frame. The data of the image frame is described using the specifications of a standard display device. The specifications include the spectral power distributions of the RGB primaries, and a transformation look-up table that describes the nonlinear relationship between the frame-buffer values in the image frame. The intensity of the light emitted by the each of the primaries on the display devices is labeled the Display Gamma Curve (DGC). Figure 4-11(b), and Figure 4-11(c) shown those data.

Thus, a transform operator can be obtained as follows

$$\begin{bmatrix} l(k) \\ m(k) \\ s(k) \end{bmatrix} = s \cdot T \cdot \begin{bmatrix} gR(k) \\ gG(k) \\ gB(k) \end{bmatrix}, \tag{4-14}$$

where

$$T = C' \cdot D = \begin{bmatrix} 0.0209 & 0.0760 & 0.01126 \\ 0.0079 & 0.0764 & 0.0163 \\ 0.0009 & 0.0080 & 0.0766 \end{bmatrix}.$$

Here C denotes the normalized response of the L-cone, M-cone, and S-cone cells in the human vision system, and D represents the radiance data of red, green, and blue phosphor of the standard CRT. Notably, the input data $gR(\mathbf{k})$, $gG(\mathbf{k})$, and $gB(\mathbf{k})$ are adjusted using the GDC coefficient g, and s is a necessary scale value. The related data $u_R(\mathbf{k})$, $u_G(\mathbf{k})$, $u_B(\mathbf{k})$, and $u_Y(\mathbf{k})$ then can be obtained as follows

$$u_R(\mathbf{k}) = l(\mathbf{k}),\ u_G(\mathbf{k}) = m(\mathbf{k}),\ u_B(\mathbf{k}) = s(\mathbf{k}),\text{ and}$$

$$u_Y(\mathbf{k}) = \frac{1}{2}\left(l(\mathbf{k}) + m(\mathbf{k})\right), \tag{4-15}$$

where $l(\mathbf{k})$, $m(\mathbf{k})$, and $s(\mathbf{k})$ represent the related signals of the L-cone, M-cone, and S-cone cells.

To present the image, the experiments use CIE-L*a*b* color space. The CIE-L*a*b* diagram is known as the Psychometric Color Diagram. The colors of the diagram lie at right angles to one another in two directions, and the plane thus created is distributed at right angles to the achromatic axis. The resultant uniform color-space is of course based on the four psychological basic-colors of red, green, blue, and yellow — first described by Ewald Hering in his opponent-theory — which are now known to be transmitted directly to the brain.

In this simulation, the R, G, B, and Y channels can be selected as the inputs. Consequently, the input values of CIE-L*a*b* can be obtained as follows

$$L(\mathbf{k}) = \max\left(u_R(\mathbf{k}), u_G(\mathbf{k}), u_B(\mathbf{k})\right)$$

$$a(\mathbf{k}) = CNN_{\text{C-Ganglion},\varepsilon,b,g,\lambda_{R_u},\lambda_G}\left(u_R(\mathbf{k}), u_G(\mathbf{k})\right),\text{ and}$$

$$b(\mathbf{k}) = CNN_{\text{C-Ganglion},\varepsilon,b,g,\lambda_{R_u},\lambda_G}\left(u_B(\mathbf{k}), u_Y(\mathbf{k})\right). \tag{4-16}$$

Based on the definition of CIE-L*a*b*, the color channels red, green, and blue can be obtained as follows

$$l'(\mathbf{k}) = \begin{cases} a(\mathbf{k}), & a(\mathbf{k}) > 0 \\ 0, & \text{otherwise} \end{cases},\quad m'(\mathbf{k}) = \begin{cases} -a(\mathbf{k}), & a(\mathbf{k}) < 0 \\ 0, & \text{otherwise} \end{cases},\text{ and}$$

$$s'(\mathbf{k}) = \begin{cases} b(\mathbf{k}), & b(\mathbf{k}) > 0 \\ 0, & \text{otherwise} \end{cases}. \tag{4-17}$$

Finally, the work of Eq.(4-14) and GDC coefficient g are reversed. Restated

$$\begin{bmatrix} g^{-1}R'(\mathbf{k}) \\ g^{-1}G'(\mathbf{k}) \\ g^{-1}B'(\mathbf{k}) \end{bmatrix} = s^{-1} \cdot T^{-1} \cdot \begin{bmatrix} l'(\mathbf{k}) \\ m'(\mathbf{k}) \\ s'(\mathbf{k}) \end{bmatrix}, \tag{4-18}$$

where $R'(\mathbf{k})$, $G'(\mathbf{k})$, and $B'(\mathbf{k})$ are the final outputs.

On the other hand, the present experiment simulates a center-red-on/surround-green-off and a center-blue-on/surround-yellow-off ganglion. The new red-to-green and blue-to-yellow chromatic aberrations are thus obtained.

A result is shown in Figure 4-9, where Figure 4-9(a) denotes the input, and Figure 4-9(b) represents the corresponding output. These new chromatic aberrations are used to reconstruct the image and compare it with the original one. The average Peak Signal to Noise Rate (PSNR) is 32.2263 (DB).

Central Retinal Vein Occlusion (CRVO) is a well-known retinal disease. Because a retina needs a lot of oxygen to function, significant blood circulation is necessary. Normally, blood flows into the retina via the Central Retinal Artery (CRA) and leaves via the Central Retinal Vein (CRV). Both of these blood vessels enter the eye through the optic nerve. CRVO is caused by a blood clot in the CRV, which slows or stops blood flow out of the retina. Although initially blood may continue to enter the retina through the CRA, the blockage ultimately stops blood circulation. As a result, blood and fluid are backed up, causing retinal injury and vision loss. Consequently, the brain thus becomes confused and the patient will “sense” a piece of gray in the area of the retina that has lost blood circulation (see Figure 4-10).

A simulation result of CRVO is represented in Figure 4-9(c). Human vision is a complex system, and cannot be comprehensively simulated. Even now, only some of the architecture of this system is known. However, some of the features of the present simulation closely resemble those of human vision system. First, the output of the simulator (Figure 4-9(b)) closely resembles the input (Figure 4-9(a)), even though the color channels are modified. This result implies that the human vision system cannot sense minor differences in color. Second, In the CRVO region, the stimulations of the cells in a specific region are stopped, and the CRVO region can be seen to be gray. This result demonstrates that if there is no signal in a given region, the human vision system will sense a piece of gray in the scene. These conclusions correlate with the results of medical investigations.

4.3 Discussions

This work studied the possibilities for implementing an hCNN-based computer fovea model. Although the details of the fovea, retina, and even the whole of the human vision system remain unknown, based on some results of previous researches on both biology and digital image processing, the fovea model can be roughly realized, and consequently some possible applications can be fulfilled. However, some issues remain open to question. For example, this experiment examines 2 or 3 types of ganglions. In fact, over 20 different structures of ganglions have already been identified. Thus, it is interesting to consider the relationships among these different types of implementations, particularly in cases where the response of a ganglion is related to the time domain. On the other hand, the results of the video processing may differ from those in the proposed model. Roska and Bálya *et al.* discussed the parallel structure of the mammalian retina and its implementation in CNN [6]. Their study indicated demonstrated the complexity of the retina. Studying the relationship between the structure of the retina and possible applications for image processing offers one of the most interesting topics for future research.

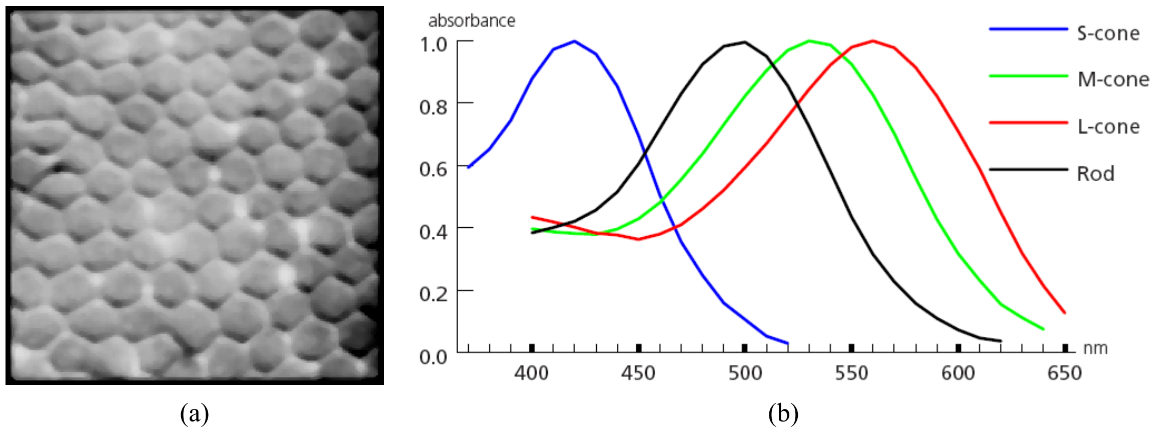


Figure 4-1. The biological structure of the photoreceptors in the fovea of a mammalian, where (a) represents the photoreceptor group in a fovea and (b) represents that the photoreceptors can be classified according to the stimulation they react to [1].

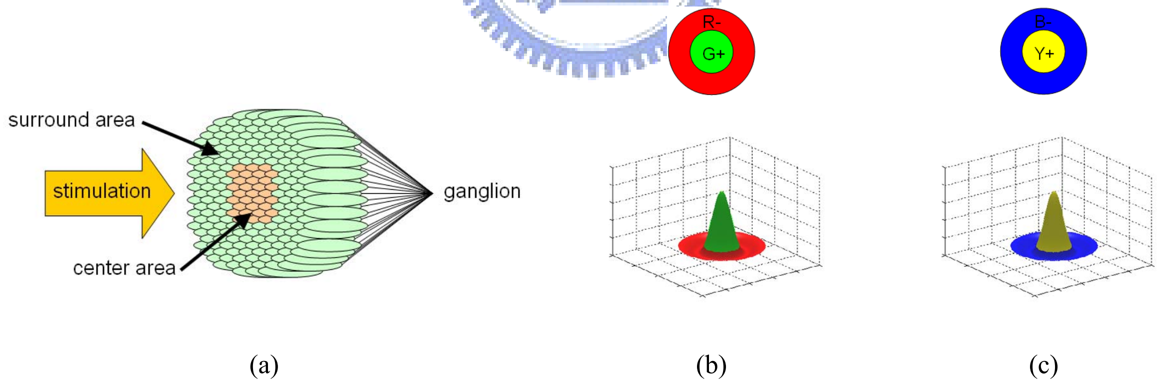


Figure 4-2. An example of a ganglion, where (a) represents that a ganglion collects information from a set of neighboring photoreceptors. The response of the central part of the set differs from that of the peripheral part of the set. Based on the difference between the types of photoreceptor, ganglion can be classified into red-green (RG), blue-yellow (BY), and black-white (BW) types. Meanwhile, (b) shows the RF of a center-green-on/surround-red-off ganglion, and (c) shows the RF of a center-yellow-on/surround-blue-off ganglion.

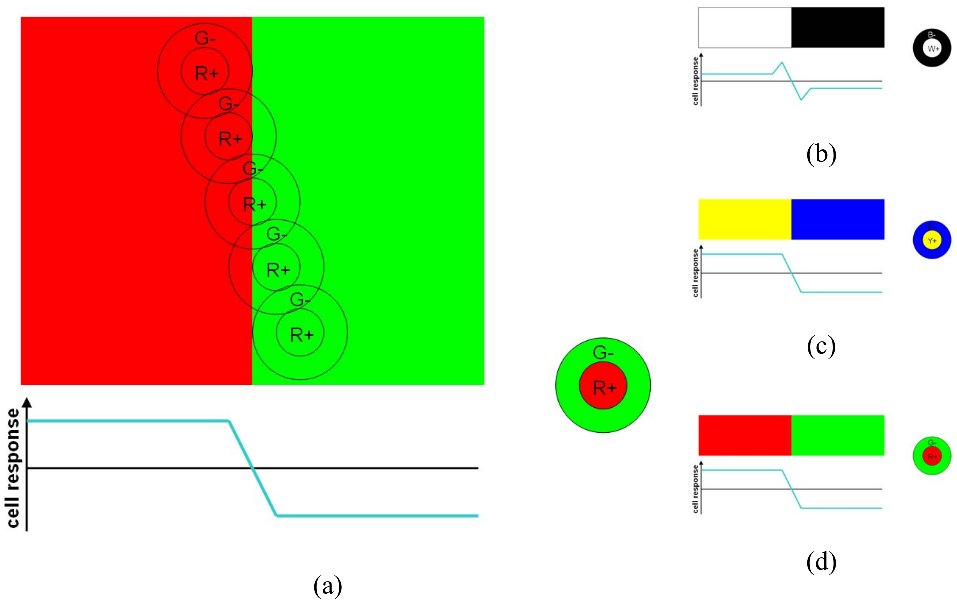
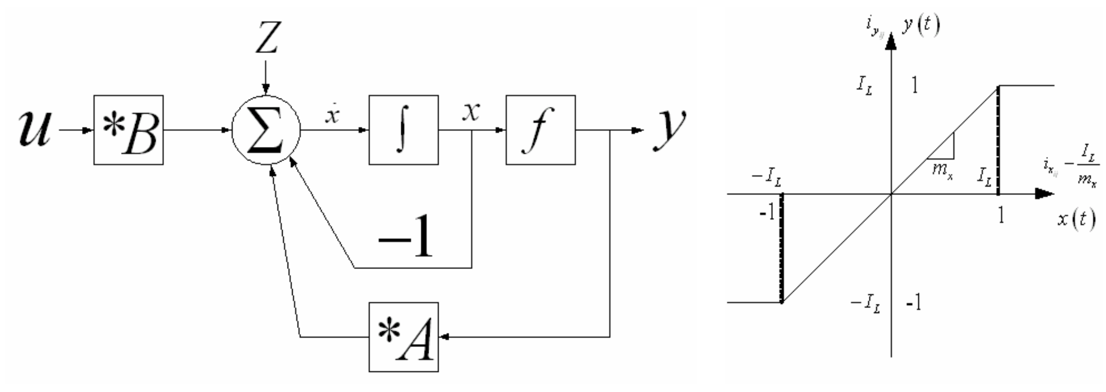
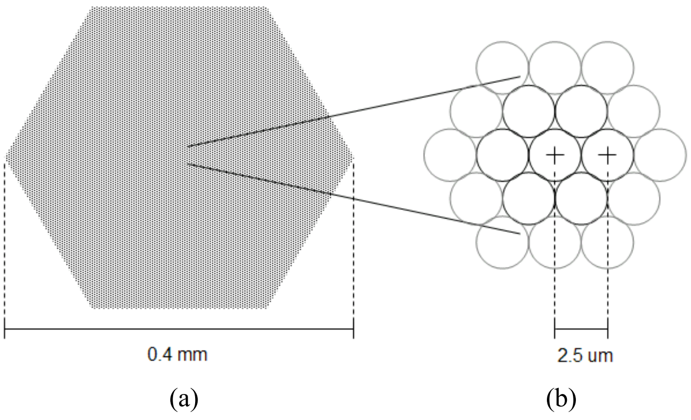


Figure 4-3. Based on the different responses of neighboring ganglions, the visual information can be sensed. The visual information included luminance and color. (a) represents how the center-green-on/surround-red-off ganglions response on the border of red and green region, (b), (c), and (d) are the responses of the different kinds of ganglion [1].



(a) (b)

Figure 4-4. The architecture of CNN is shown in (a), and along with the dynamic route of state in CNN.



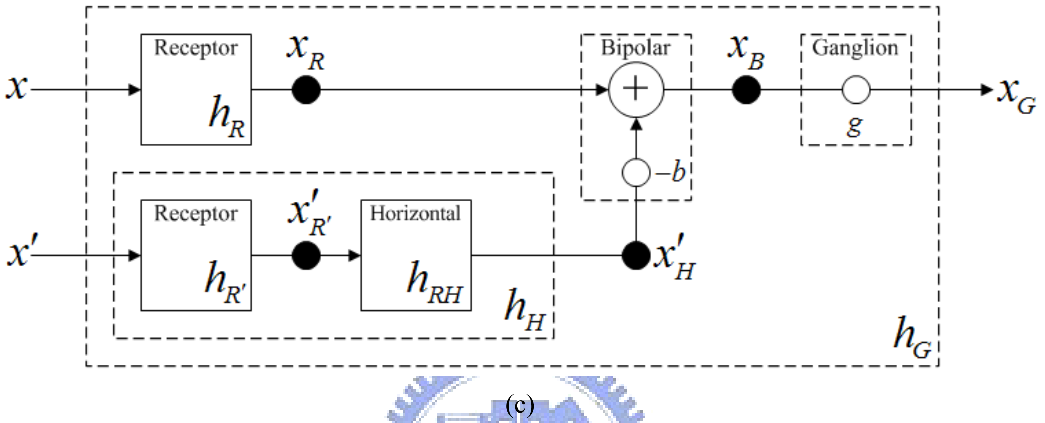
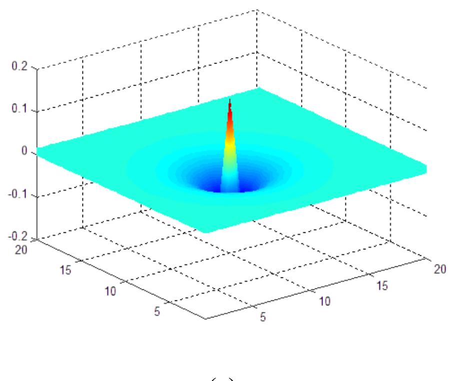


Figure 4-5. The proposed computer fovea model, where (a) denotes a computer fovea which has been constructed using hexagonally arranged cells. The width of the computer fovea is approximately 0.4 mm, and is 161 csp. Moreover, (b) represents a part of the computer fovea and illustrates the arrangement of the photoreceptors. The diameter of the photoreceptor is approximately 2.5 um. Finally, (c) is the signal processing system of each cell in the fovea model. The input x is the input signal of the center area, while x' is the input signal of the surrounding area. For a monotonic input signal, x equals x'.



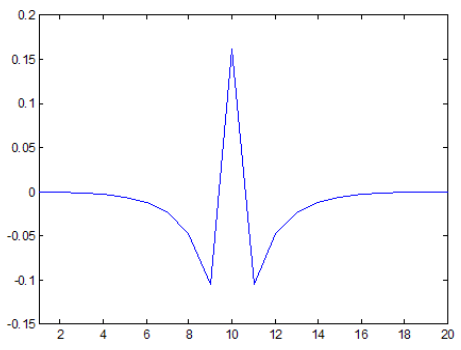
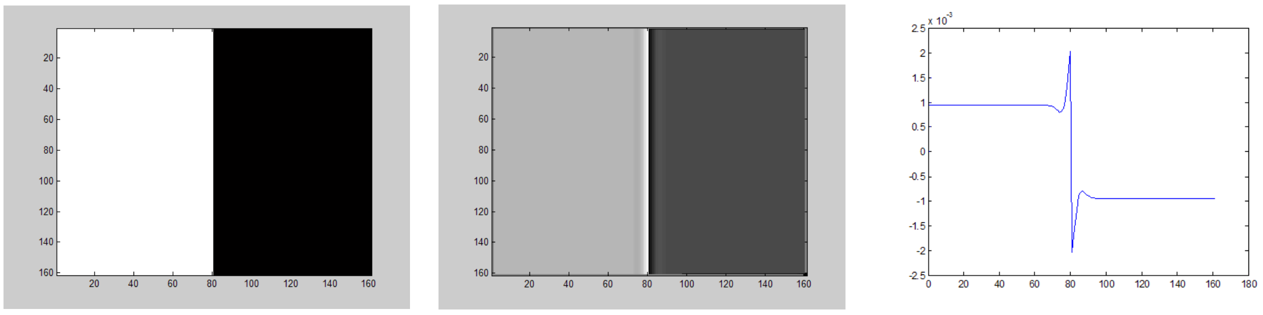
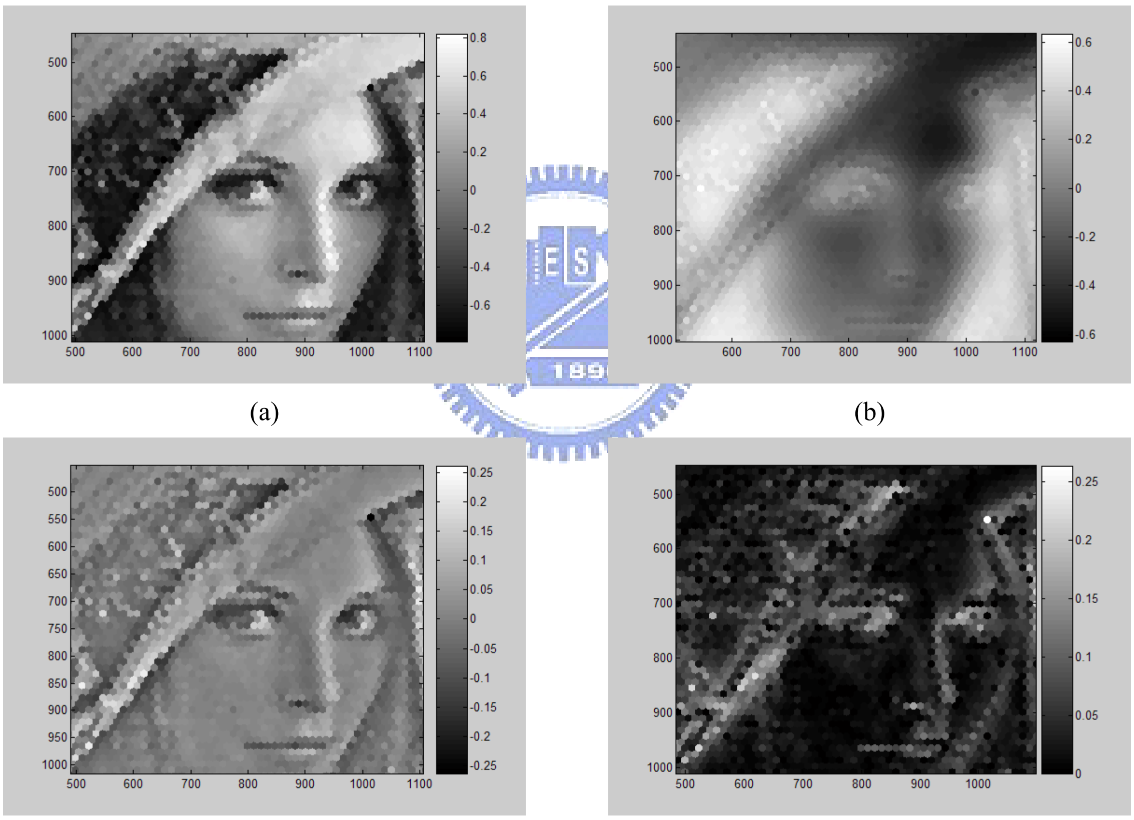


Figure 4-6. An example of a center-on/surround-off RF, where λ_R=0.53604, $\lambda_G = 0.5677$, $\varepsilon = 0.01$, and $b = 0.75$.



(a) (b) (c)

Figure 4-7. An example of the model, where (a) is the input and (b) represents the output of the ganglions, while (c) shows the intensity of a section of (b). Notably, the behaviors of this model resembles that of the human vision visual system (see Figure 4-3).



(a) (b) (c) (d)

Figure 4-8. Another example of the proposed model, where the input is a natural image. (a) is the input, (b) is the output of the horizontal cells, (c) is the output of the ganglions, and finally, (d) is the output following the rectification. Notably, to present the structure of the hexagonal structure, the resolution of the image is reduced in this example.

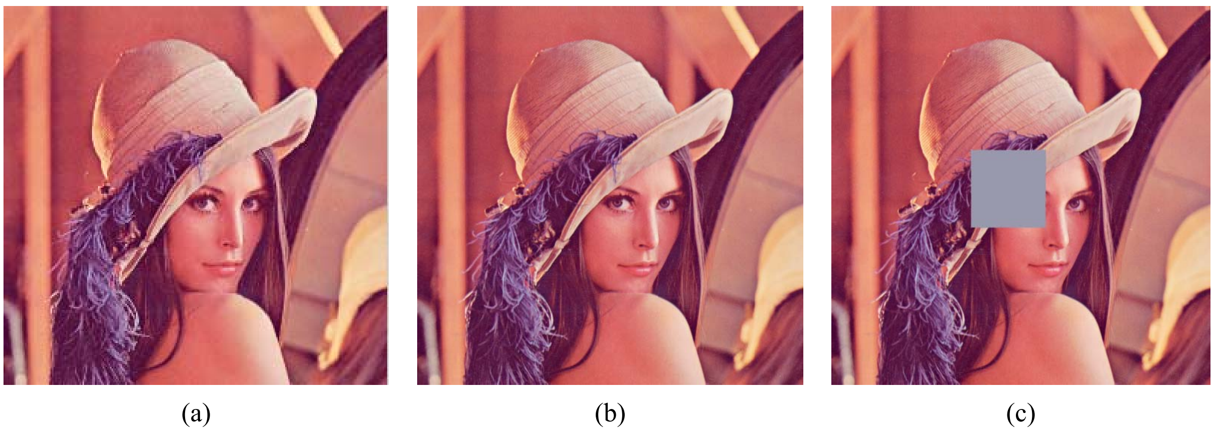


Figure 4-9. This study uses a red-green and a blue-yellow channel as the input of the center-red-on/surround-green-off and center-blue-on/surround-yellow-off ganglions, and consequently the new red-to-green and blue-to-yellow chromatic aberrations are obtained. (a) shows the test image and (b) illustrates the reconstructed result. These new chromatic aberrations were used to reconstruct the image, and compare it with the original. The average of Peak Signal to Noise Rate (PSNR) is 32.2263 (dB), while (c) shows the simulation result of the CRVO.

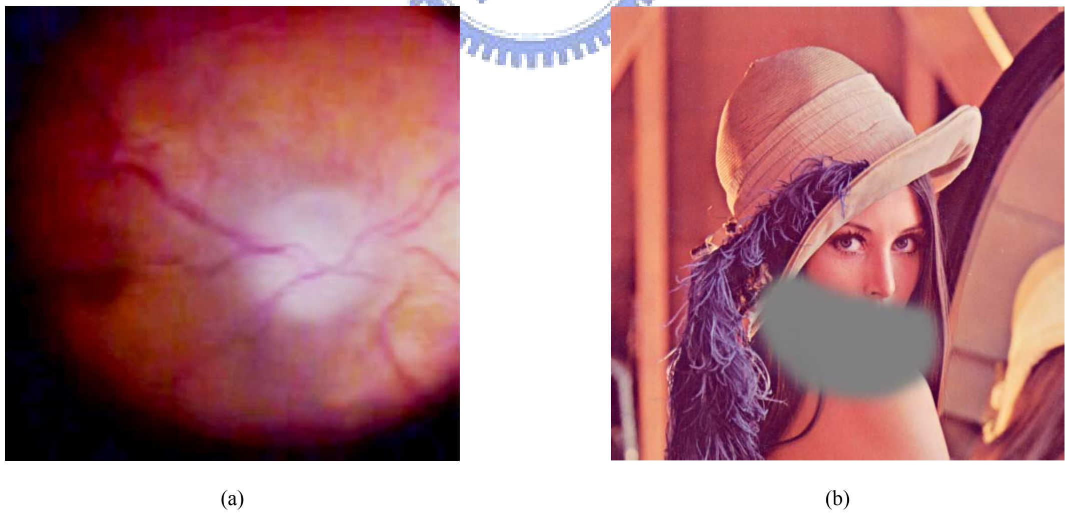


Figure 4-10. A case of a retinal injury and its visual response, where (a) demonstrates a retina with Central Retinal Vein Occlusion (CRVO). The vein on the left side of the fovea is clogged and results retinal injury. (b) demonstrated that the patient will “sense” a piece of gray in the area of the retina that has lost blood circulation.

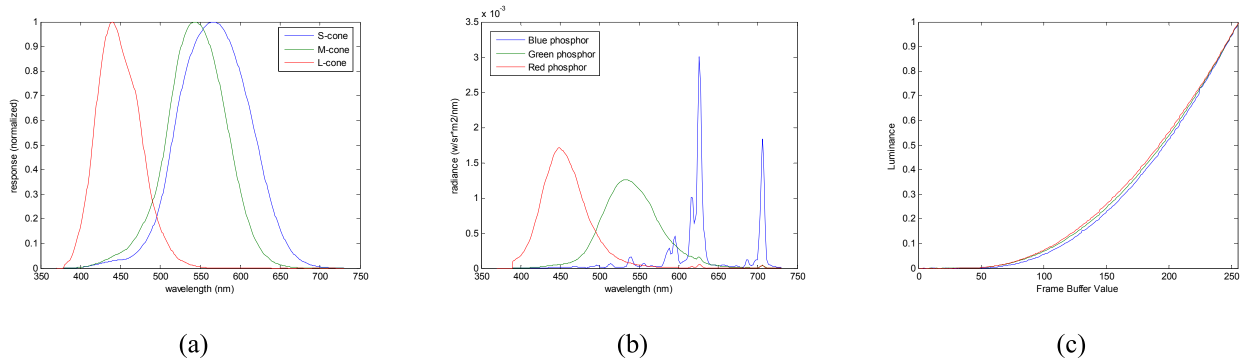


Figure 4-11. (a) represents the corresponding responses of the L-cone, the M-cone, and the S-cone cells in the different wavelengths, (b) is the spectral power distributions of the RGB primaries, and (c) represents the transformation look-up table which describes nonlinear relationship between the frame-buffer values in the image data and the intensity of the light emitted by each of the primaries on the display, named the Display Gamma Curve (DGC).

Table 4-1. Corresponding λ in Eq.(4-5) for deviation σ in Eq.(4-4).

σ in Eq.(4-5)	Corresponding λ in Eq.(4-5)	MSE
$\sqrt{2}$	~0.567700	3.5783×10^{-5}
1.5	~0.536040	2.8497×10^{-5}
12	~0.071233	4.6988×10^{-9}

Table 4-2. The result of Rotation-invariant Texture Classifier. The input texture images have been shown in Figure 6-5. The rate of correct classification is about 97.5%.

	a1	**b1**	**c1**	**d1**	**a2**	**b2**	**c2**	**d2**	**a3**	**b3**	**c3**	**d3**	**a4**	**b4**	**c4**	**d4**
a1	0	0.08	0.07	0	1.2	1.11	1.2	1.2	0.34	0.32	0.35	0.34	0.39	0.39	0.38	0.38
b1	0.08	0	0.01	0.08	1.11	1.05	1.13	1.12	0.32	0.31	0.32	0.32	0.28	0.28	0.27	0.28
c1	0.07	0.01	0	0.07	1.05	1	1.07	1.05	0.32	0.31	0.32	0.32	0.28	0.27	0.26	0.28
d1	0	0.08	0.07	0	1.2	1.11	1.2	1.2	0.34	0.32	0.35	0.34	0.39	0.39	0.38	0.38
a2	1.2	1.11	1.05	1.2	0	0.1	0.15	0	1.82	1.83	1.85	1.82	1.74	1.64	1.66	1.73
b2	1.11	1.05	1	1.11	0.1	0	0.1	0.1	1.73	1.74	1.76	1.73	1.74	1.65	1.65	1.73
c2	1.2	1.13	1.07	1.2	0.15	0.1	0	0.15	1.93	1.94	1.96	1.93	1.89	1.79	1.79	1.88
d2	1.2	1.12	1.05	1.2	0	0.1	0.15	0	1.83	1.83	1.86	1.83	1.75	1.65	1.67	1.74
a3	0.34	0.32	0.32	0.34	1.82	1.73	1.93	1.83	0	0.01	0.01	0	0.28	0.32	0.3	0.28
b3	0.32	0.31	0.31	0.32	1.83	1.74	1.94	1.83	0.01	0	0	0.01	0.24	0.28	0.27	0.24
c3	0.35	0.32	0.32	0.35	1.85	1.76	1.96	1.86	0.01	0	0	0.01	0.25	0.28	0.27	0.25
d3	0.34	0.32	0.32	0.34	1.82	1.73	1.93	1.83	0	0.01	0.01	0	0.28	0.32	0.31	0.28
a4	0.39	0.28	0.28	0.39	1.74	1.74	1.89	1.75	0.28	0.24	0.25	0.28	0	0.02	0.02	0
b4	0.39	0.28	0.27	0.39	1.64	1.65	1.79	1.65	0.32	0.28	0.28	0.32	0.02	0	0.01	0.02
c4	0.38	0.27	0.26	0.38	1.66	1.65	1.79	1.67	0.3	0.27	0.27	0.31	0.02	0.01	0	0.02
d4	0.38	0.28	0.28	0.38	1.73	1.73	1.88	1.74	0.28	0.24	0.25	0.28	0	0.02	0.02	0

Table 4-3. The result of Scale-invariant Texture Classifier. The input texture images have been shown in Figure 6-5. The rate of correct classification is about 92.5%.

	a5	**b5**	**c5**	**d5**	**a6**	**b6**	**c6**	**d6**	**a7**	**b7**	**c7**	**d7**	**a8**	**b8**	**c8**	**d8**
a5	0	0	0.01	0.01	0.3	0.33	0.39	0.34	0.02	0.02	0.02	0.01	0.02	0.02	0.03	0.04
b5	0	0	0	0	0.31	0.34	0.41	0.36	0.03	0.03	0.03	0.02	0.02	0.02	0.03	0.04
c5	0.01	0	0	0	0.31	0.34	0.41	0.36	0.05	0.05	0.05	0.04	0.02	0.03	0.03	0.04
d5	0.01	0	0	0	0.27	0.3	0.37	0.32	0.05	0.05	0.05	0.04	0.02	0.02	0.03	0.03
a6	0.3	0.31	0.31	0.27	0	0	0.01	0	0.39	0.41	0.39	0.36	0.23	0.21	0.19	0.16
b6	0.33	0.34	0.34	0.3	0	0	0.01	0.01	0.42	0.43	0.41	0.38	0.25	0.23	0.21	0.18
c6	0.39	0.41	0.41	0.37	0.01	0.01	0	0	0.48	0.5	0.48	0.44	0.31	0.29	0.27	0.23
d6	0.34	0.36	0.36	0.32	0	0.01	0	0	0.44	0.45	0.43	0.4	0.27	0.25	0.23	0.19
a7	0.02	0.03	0.05	0.05	0.39	0.42	0.48	0.44	0	0	0	0	0.04	0.05	0.07	0.09
b7	0.02	0.03	0.05	0.05	0.41	0.43	0.5	0.45	0	0	0	0	0.04	0.05	0.07	0.09
c7	0.02	0.03	0.05	0.05	0.39	0.41	0.48	0.43	0	0	0	0	0.04	0.05	0.06	0.08
d7	0.01	0.02	0.04	0.04	0.36	0.38	0.44	0.4	0	0	0	0	0.03	0.04	0.05	0.07
a8	0.02	0.02	0.02	0.02	0.23	0.25	0.31	0.27	0.04	0.04	0.04	0.03	0	0	0.01	0.01
b8	0.02	0.02	0.03	0.02	0.21	0.23	0.29	0.25	0.05	0.05	0.05	0.04	0	0	0	0.01
c8	0.03	0.03	0.03	0.03	0.19	0.21	0.27	0.23	0.07	0.07	0.06	0.05	0.01	0	0	0
d8	0.04	0.04	0.04	0.03	0.16	0.18	0.23	0.19	0.09	0.09	0.08	0.07	0.01	0.01	0	0

5 CNN-BASED COMPUTER CORTEX MODEL

Researchers in recent decades have elucidated signal transduction in the retina and the function of the visual cortex. The highly flexible nature of the neural circuits in the visual cortex, especially during the critical period (the early vision period) has been an interesting subject for studying and developing the neural plasticity.

Early vision, also known as the pre-attentive vision, represents a set of information processing mechanisms of the first stage of the visual processing. These mechanisms are operated in parallel across the visual field, and are believed to be used for detecting the most basic visual features, such as contrast, edge, contour, grouping, corner, junction, brightness and lightness computation for surface perception. On the other hand, Jain and Farrokhnia proposed their research results as follows: the human vision system included two fundamental features: one is that the difference of light intensity that projects into the retina, the other is that the behavior of the retina is similar to the band pass filtering [48]. Those facts implicate that the abrupt changes on the intensity of light will bring the visual cortex stronger stimulation, and those abrupt changes become the so-called boundaries. Sometimes those boundaries have been called the first order features. In general, the first order features equal the average of the local areas. According to some investigations, the boundary detection algorithms based on this features are related to the zero crossing of the *Laplacian of Gaussian (LoG)* and, sometimes, the Canny's boundary detector [48]. There is one common issue in computer vision that it does not make a distinction between the contours of objects which are the actual primitives needed in the most application. It reveals that there are still some other mechanisms during the visual perception. Thus, in this section, we will focus on the texture segregation.

The segregation of the visual scenes (better known as boundary detection) is the fundamental process of the early vision. That is also an important and fundamental issue in image processing. Boundary detection can be applied in many areas, such as object tracking, stereo vision, pattern recognition, etc. Basically, boundary detection is used to separate the specific partition of the images into related regions, and thus the boundaries will be extracted. Or in other words, boundary detection is used to compare the difference between the spatial related local features of the images. The procedure of those processing is not an easy problem in the image processing area. However, it seems to be an intuitive ability in the human vision system, obviously.

At first, we should give a definition to texture. Unfortunately, in the author's knowledge, there are not any precise and identical definitions to texture. Although we don't have the best definition for texture, those features are so obvious that we can not neglect it. Some researchers have proposed different texture descriptions. The "definition" of texture is formulated depending on the particular application and there is no generally agreed upon definition. We give some perceptually motivated examples as follows:

- "We may regard texture as what constitutes a macroscopic region. Its structure is simply attributed to the repetitive patterns in which elements or primitives are arranged according to a placement rule [55]."
- "A region in an image has a constant texture if a set of local statistics or other local properties of the picture function are constant, slowly varying, or approximately periodic [56]."
- "The notion of texture appears to depend upon three ingredients: (i) some local 'order' is repeated over a region which is large in comparison to the order's size, (ii) the order consists in the nonrandom arrangement of elementary parts, and (iii) the parts are roughly uniform entities having approximately the same dimensions everywhere within the textured region [57]."

Bovic, Clark and Geisler give a very detailed analysis of the Gabor function which uses localized spatial filters for the texture feature extraction [58]. Bovic mentioned three supervised approaches to select the filter locations using the empirical information based on the power spectrum characteristics of the individual textures: first, for stronger oriented textures, the most significant spectral peak along the dominant orientation direction is used as a filter location. Second, pick the lower fundamental frequency to identify periodic textures. Third, for non-oriented textures, using the center frequencies of the two largest maximum is recommended. It is clear that an automated method is more attractive.

Dunn and Higgins developed a method for selecting optimal filter parameters [59]. This is a supervised approach that focused mainly on using the minimal number of filters. Only the specific filter, that separates two textures optimally, is used to partition an image. The optimal filter responds strongly to one class, and a lack of textural information of the other class may be also expressed. This kind of class is not identified to have a particular characteristic but lacking a characteristic of another. A more global solution to the problem is spreading filters throughout the frequency domain field to capture salient information.

By providing the uniform coverage of the spatial-frequency domain with the Gabor filters, the problem of the selected central frequency is avoided. Jain and Farrokhnia implemented the real Gabor filters for the texture segmentation using frequency bandwidth of one octave, center frequency spacing of one octave, angular bandwidth of 45°, and angular spacing of 45° [48]. The frequencies used for filters are as follows

$$1\sqrt{2},\ 2\sqrt{2},\ 4\sqrt{2},\ \ldots,\ \left(N_c/4\right)\sqrt{2} \quad \text{(cycles per image width).}$$

For the textures with distinct spectral peaks which correspond to some global regularity, T. N. Tan proposed a useful method [60] to design the Gabor filters automatically. The central step in the algorithm is a spectral detection. It detects a global spectral peak, and repeatedly detects conspicuous peaks by *erasing-operation* on the spatial frequency plan. The power spectrum of a small neighborhood (e.g., 7×7) around the detected peak is set to zero. The iteration of the peak detection terminates when the ratio of the magnitude of the current peak to that of the first (e.g., the highest) peak is less than a pre-specified value (e.g., 80%).

In this section, first, the related knowledge from physiology and psychophysics will be introduced. Next, proposed algorithm, so-called CNN-based Hybrid-order Texture Segregation will be presented. Third, some experiments will be shown. Finally the conclusions follow.

5.1 CNN-based Hybrid-order Texture Segregation

In this section, a new boundary detection algorithm is proposed. This algorithm combines the first and the second order features for modeling the pre-attentive stage of HVS.

Figure 5-1 shows the flow chart of the proposed approach: first, the first order features have been extracted by the Gaussian low-pass filters and the second order features have been extracted by the Gabor filters, respectively. Assume that each pixel of the output is defined by a N+1 dimensional vector. After the first order features extraction, the vector contains N Gabor filters and 1 Gaussian filter. Next, we measure the difference of each pixel with its neighbor. Since the pixels, which belong to the same region have similar features, the level of difference among those pixels should be smaller than the difference to pixels existing in other regions of the image. Third, we keep these pixels with values larger than a specific threshold, and set the others to zero. We would get coarse boundaries which are look like Bell-shaped distribution. Consequently, we may go to thin these boundaries by a local peak detection. Finally, we can obtain boundaries similar to HVS. Followed sections are going to introduce each block of the proposed algorithm.

5.1.1 First order Feature Extraction

As we have introduced in the previous section, the ganglion is accomplished by the so-called “center-surround” organization of the receptive field, in which it’s excitatory and inhibitory subfields are organized into circularly symmetric regions. This fact implies the receptive field of the ganglion is similar to the *Difference of two Gaussians (DoG)*.

Subsequently we describe how the DoG function detects boundaries: first, two Gaussian filters with different values of σ are applied in parallel to the images. Afterwards the difference of the two smoothed instances is computed. It can be shown that the DoG operator approximates the LoG one which has been widely used in boundary detection.

The receptive field of the ganglion cells can be considered as the linear spatial weighting function. That is, we can model the retinal ganglion cell as a linearized function, where the receptive field implies where the weights are. Using the function to characterize the shape of the receptive field based on the DoG model, we calculate the output of the retinal ganglion cells as follows

$$O = \sum_{x,y} R(x,y) I(x,y) , \tag{5-1}$$

where $I(x,y)$ is the input image.

For the whole of retinal ganglion cells with identical receptive fields, we calculate the output of each cell in the array as follows

$$O(x_0, y_0) = \sum_{x,y} R(x - x_0, y - y_0) I(x,y) , \tag{5-2}$$

where $O(x_0, y_0)$ is one of the output of the retinal ganglion cells, whose receptive field is centered at position x_0 and y_0.

The operation of DoG function can be divided into two stages: the Gaussian convolution and the gradient operation. The Gaussian convolution is like extracting the mean of the local region which is the so-called first order feature, and the gradient operation as a measure for the variation of the first order feature.

For the sake of combining of the first order and the second order features, first we use the Gaussian convolution for extracting the first order features. And then, the gradient processing will be applied after combining the second order features. Figure 5-2 illustrates the coarse boundary between two patterns based on first order features.

There is more than one type of features that are mixed, and only the first and second order features are insufficient. Figure 5-3 demonstrates the situation for the patterns (D101-D102 from Brodatz textures [61]) with hybrid-order feature. Note that the first order feature is dominant.

For convenience, we use a general low-pass operation to emulate the Gaussian function in the CNN. The template can be defined as follows [62]:

$$A=\begin{bmatrix} 0.1 & 0.15 & 0.1 \\ 0.15 & 0 & 0.15 \\ 0.1 & 0.15 & 0.1 \end{bmatrix}, B=\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, z=0\,. \tag{5-3}$$

5.1.2 Second order Feature Extraction

As we have mentioned in the previous section, the receptive fields of V1 cells are orientation selective, and it can be modeled by the Gabor function [7, 57]. The Gabor function is an adaptive band-pass filtering method which constructs a complete but non-orthogonal basis set. On the other hand, the Gabor function consists of a Gaussian function which is modulated by a sinusoidal function and can be described as follows [16]:

$$g_{K,S}(x,y)=\left(\frac{1}{2\pi\sigma_x\sigma_y}\right)\exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right)\right], \tag{5-4}$$

and the Fourier transform of Eq.(5-4) is as follows

$$G_{K,S}(u,v) = \exp\left[-\frac{1}{2}\left(\frac{(u-W)^2}{\sigma_u^2} + \frac{v^2}{\sigma_v^2}\right)\right], \tag{5-5}$$

where $\sigma_u = 1/(2\pi\sigma_x)$ and $\sigma_v = 1/(2\pi\sigma_y)$.

An example of the Gabor function has been presented in Figure 5-4, where Figure 5-4(a) is the impulse response of a standard Gabor filtering in time domain, and Figure 5-4(b) is its frequency domain representation. Expanding the signal using this basis provides a localized frequency description.

In practice, the Gabor function can be divided into real (even) part and imaginary (odd) part as follows

$$g_c(x,y) = h(x,y)\cos(2\pi Fx'), \text{ and} \tag{5-6}$$

$$g_s(x,y) = h(x,y)\sin(2\pi Fx'), \tag{5-7}$$

where $x' = x\cos\phi + y\sin\phi$, and $h(x,y)$ is a Gaussian function.

We can see that Eq.(5-6) and Eq.(5-7) are very similar to the each other. In fact, Eq.(5-7) is just a phase shifted version of Eq.(5-6). Both of them can be used to extract the local features of the spatial domain. Based on some of the psychophysical theories, Malik and Perona provide justifications which indicate that we can use even-symmetric filters only. In this section we use real-valued, even-symmetric Gabor filters.

For implementation of the 2D Gabor-type filtering, according to Shi, the template in the CNN for the 2D Gabor-type filtering can be represented as follows [14].

$$A = \begin{bmatrix} 0 & e^{-jw_{y0}} & 0 \\ e^{jw_{x0}} & -\left(4+\lambda^2\right) & e^{-jw_{x0}} \\ 0 & e^{jw_{y0}} & 0 \end{bmatrix},$$
$$B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \lambda^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{5-8}$$
$$z = 0.$$

Shi's CNN-based Gabor-type filtering provides good results. However, the complex template becomes a problem when we tried to implement it on the CNN-UM. Thus, we proposed a linear-region, CNN-based, Gabor-type filtering. The proposed algorithm can be separated into a two steps CNN processing .That can be presented as follows

$$\text{First Step: } A_1 = \begin{bmatrix} a & b & c \\ d & -\left(2+\lambda^2\right) & d \\ c & b & a \end{bmatrix}, B_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \lambda^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}, I_1 = 0, \tag{5-9}$$

and

$$\text{Second Step: } A_2 = \begin{bmatrix} c & d & a \\ b & \left(2+\lambda^2\right) & b \\ a & d & c \end{bmatrix}, B_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \lambda^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}, I_2 = 0, \tag{5-10}$$

where a, b, c, and d is used to determine the orientation of the Gabor-type filtering. Here λ can be used to determine the bandwidth of the filtering. For the convenience, the following equation is used to represent the operation of those templates:

$$CNN_{Gabor\_UM,a,b,c,d,\lambda}\left(u(\mathbf{k})\right). \tag{5-11}$$

Figure 5-6 presents the frequency response of the proposed CNN-based Gabor-type filtering with the following parameters: $a=0$, $b=1$, $c=0$, $d=0$, and $\lambda=0.9$. The frequency response is given by:

$$\text{First Step: } H_1\left(e^{j\omega_x}, e^{j\omega_y}\right) = \frac{\lambda^2}{2+\lambda^2-2\cos\left(\omega_y\right)}, \text{ for all } \omega_x, \tag{5-12}$$

and

$$\text{Second Step: } H_2\left(e^{j\omega_x}, e^{j\omega_y}\right) = \frac{\lambda^2}{-2-\lambda^2-2\cos\left(\omega_x\right)}, \text{ for all } \omega_y. \tag{5-13}$$

Thus, the frequency response of the proposed CNN-based Gabor-type filtering can be represented as follows

$$\begin{aligned} H\left(e^{j\omega_x}, e^{j\omega_y}\right) &= H_1\left(e^{j\omega_x}, e^{j\omega_y}\right) H_2\left(e^{j\omega_x}, e^{j\omega_y}\right) \\ &= \frac{\lambda^4}{-\lambda^4-4\lambda^2-2\lambda^2\cos\left(\omega_x\right)+4\cos\left(\omega_y\right)+2\lambda^2\cos\left(\omega_y\right)-4\cos\left(\omega_x\right)+4\cos\left(\omega_x\right)\cos\left(\omega_y\right)-4}. \end{aligned} \tag{5-14}$$

In fact, the proposed linear-region, CNN-based Gabor-type filtering is not able to adjust the frequency-related parameters. However, it can be implemented by the CNN-UM. Consequently, the entire proposed approach can be realized.

5.1.3 Gabor Filtering for the Visual Pathway of a Human Visual System

The Gabor or Gabor-type filtering which we have introduced in the previous are used for general type images. The pixel values on those images represent the level of brightness or color. However, those signals are not the right signals for the visual pathway of a human visual system because the signals on the pathway have been processed by the computer fovea model. Thus, a Gabor-type filtering operator for the output signals of a ganglion is required and can be represented as

$$\begin{aligned}&CNN_{C-Ganglion-Gabor,\varepsilon,b,g,\lambda_{R_u},\lambda_G,\omega_{x_0},\omega_{y_0}}\left(u(\mathbf{k}),v(\mathbf{k})\right)=\\&CNN_{Gabor,\omega_{x_0},\omega_{y_0}}\left(CNN^{-1}_{\text{Gaussian},\lambda_{R_u}}\left(CNN_{\text{M-Ganglion},\varepsilon,b,g,\lambda_{R_u},\lambda_G}\left(u(\mathbf{k})\right)+CNN_{\text{Horizontal},\varepsilon,b,g,\lambda_{R_u},\lambda_G}\left(v(\mathbf{k})\right)\right)\right),\end{aligned}\tag{5-15}$$

or, for a UM environment, we can let

$$\begin{aligned}&CNN_{C-Ganglion-Gabor,\varepsilon,b,g,\lambda_{R_u},\lambda_G,a,b,c,d,\lambda}\left(u(\mathbf{k}),v(\mathbf{k})\right)=\\&CNN_{Gabor\_UM,a,b,c,d,\lambda}\left(CNN^{-1}_{\text{Gaussian},\lambda_{R_u}}\left(CNN_{\text{M-Ganglion},\varepsilon,b,g,\lambda_{R_u},\lambda_G}\left(u(\mathbf{k})\right)+CNN_{\text{Horizontal},\varepsilon,b,g,\lambda_{R_u},\lambda_G}\left(v(\mathbf{k})\right)\right)\right).\end{aligned}\tag{5-16}$$

For monotonic input signals, the equation is as

$$\begin{aligned}&CNN_{M-Ganglion-Gabor,\varepsilon,b,g,\lambda_{R_u},\lambda_G,\omega_{x_0},\omega_{y_0}}\left(u(\mathbf{k})\right)=\\&CNN_{Gabor,\omega_{x_0},\omega_{y_0}}\left(CNN^{-1}_{\text{Gaussian},\lambda_{R_u}}\left(CNN_{\text{M-Ganglion},\varepsilon,b,g,\lambda_{R_u},\lambda_G}\left(u(\mathbf{k})\right)+CNN_{\text{Horizontal},\varepsilon,b,g,\lambda_{R_u},\lambda_G}\left(u(\mathbf{k})\right)\right)\right),\end{aligned}\tag{5-17}$$

and

$$\begin{aligned}&CNN_{M-Ganglion-Gabor,\varepsilon,b,g,\lambda_{R_u},\lambda_G,a,b,c,d,\lambda}\left(u(\mathbf{k})\right)=\\&CNN_{Gabor\_UM,a,b,c,d,\lambda}\left(CNN^{-1}_{\text{Gaussian},\lambda_{R_u}}\left(CNN_{\text{M-Ganglion},\varepsilon,b,g,\lambda_{R_u},\lambda_G}\left(u(\mathbf{k})\right)+CNN_{\text{Horizontal},\varepsilon,b,g,\lambda_{R_u},\lambda_G}\left(u(\mathbf{k})\right)\right)\right).\end{aligned}\tag{5-18}$$

5.1.4 Gabor Filtering Bank Set

Besides the orientation selectivity, the Gabor filters are also frequency selective. With these two properties, Daugman extended the original Gabor filter to a two-dimensional (d2) representation [63]. There are many researches which focus on the Gabor filter bank. Jain and Farrokhnia suggested a bank of Gabor filters, *i.e.*, Gaussian shaped band-pass filters, with dyadic coverage of the radial spatial frequency range and multiple orientations Figure 5-4. Figure 5-5 shows an example of the Gabor filtering bank set.

Gabor has first recognized and introduced a time-frequency version of Heisenberg's inequality as follows

$$\sigma_t \sigma_f \geq \frac{1}{4\pi}, \tag{5-19}$$

where σ_t and σ_f are the time and frequency standard deviations respectively. The Gabor filter is just the modulation of the Gaussian function. For the Gabor function it has been proven that the action only causes a shift in the frequency domain, and it wouldn't affect the resolution of the Gaussian function in both the spatial and the frequency domain. It means that the Gabor function inherit the properties of the Gaussian possessing optimal resolution in both domains, and those properties imply the Gabor filter is suitable for the texture segregation.

Since the goal of this section is designing an algorithm which can be implemented on the CNN-UM, the structure could not be complex-valued. In this section, we use four Gabor filters for extracting the second order features in the experiments. All of these Gabor filters have the same Gaussian shape in the frequency domain and scatter uniformly in four orientations.

5.1.5 Full-wave Rectification

Just like the other filter-rectify-filter model, the rectifying operation is taken after the operation of the Gabor filters. It has been generally acknowledged that V1 cells have a property which looks like the half wave rectification. The intervening rectification ensures that the fine-grain positive and negative portions of the carrier will not disable another when the smoothing operation is performed.

Figure 5-7(b) demonstrates the output of the Gabor filtering without rectification, and Figure 5-7(c) is the result of the same operation but with rectification. The white pixels in the image represent the pixels that the matching features have been detected by the Gabor function. This result is similar to the behavior of the V1 cells. Because of the restriction of display, there are some pixels with negative response which do not appear in Figure 5-7(b). In Figure 5-7(c), the boundary of the regions is more apparently, and that is because of the rectification turning the negative response to positive.

For the implementation of the full-wave rectification by the CNN, the template can be defined as follows

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 0 \end{bmatrix}, z = 0, \tag{5-20}$$

where

$$b = \begin{cases} 1 & \text{if } v_{u_{ij}} \geq 0 \\ -1 & \text{otherwise} \end{cases}. \tag{5-21}$$

Although this operation can not be implemented in the CNN-UM because of the nonlinear feature, we can implement it very easily by integrated circuits design.

5.1.6 Gaussian Post Filtering

After the cells were stimulated by a specific signal, for example, a bar with specific orientations, the output of the V1 cells responding to same direction will aggregate together. The region of cells which contain the same properties will respond stronger than the other regions. It is consistent with the "localization" properties of the textures. This effect can be simulated by a Gaussian post filters. It looks like the averaging with different weighting which is inverse proportional to the distance from the center of the post filter.

Figure 5-8(b) shows the result after the processing of rectification, and Figure 5-8(c) is the result that Figure 5-8(b) has been processed by the Gaussian post filtering. Note that in Figure 5-8(c) there is a ramp-like feature profile. The next step is for detecting the position where the variation of difference is maximal.

For the implementation of the Gaussian post function in the CNN, the template can be defined as the same as in Eq.(5-3) [62].

5.1.7 Difference Measure

The features which have been extracted by Gaussian post filtering can be described by an N-dimensional vector. Each feature vector can be regard as a point in N-dimensional space. According to Chen, the difference is represented by the distance in N-dimensional space [64].

There is an important property of the textures. That is the pixels which are aggregated together usually contain similar features. Based on this property, some algorithms use gradient for extracting the features. In this section, we only calculate the difference between features of each pixel to pixels right behind and below to it.

5.1.8 Saturation

In proposed algorithm, when there are more than two kinds of textures in the test patterns, there will be more than one boundary. Because these boundaries usually do not have similar intensity, choosing threshold becomes an important problem.

For the sake of finding the threshold, we use the mean of the difference of the total pixels as the threshold. Usually, some boundaries with relative lower magnitude are eliminated. This is because of follows: first, a relative huge region be considered for measuring the local feature. Next, the scale of difference between different patterns varies enormously. Obvious boundaries and cause relatively larger difference and raise the mean of difference. The boundaries which are not so obvious causing relative lower difference will be eliminated.

For attacking this problem, a natural-log transform can be used for simulating the saturation effect. It can suppress stronger responses which may affect the threshold too much. Meanwhile, it still keeps the location of maximum difference where we assume boundaries lying.

The strength of the responses reflects the level of differences between two local regions, but it may be not so linearly consistent to our perceptional feeling. According to some biological theories, in the human vision system, the dynamic range of response is limited, and the range of response is not linearly proportional to stimulate. Natural log transform is an ordinary and important operation and it stretch the range of lower responses where we need to judge whether there are boundaries or not.

5.1.9 Local Maximum Detection

The coarse boundaries detected after taking threshold generates a range where the boundaries are probably located on. Thus, local maximum detection is used to detect the best assumption of the location of the boundaries. It assumes that the difference among different patterns should be maximal at their boundary.

Local maximum detection can be implemented by the CNN quiet easily. Figure 5-9 is an example which demonstrates the peak detection in the algorithm. Figure 5-9(a) is an input image, and Figure 5-9(b) is the detected coarse boundary. Figure 5-9(c) is the 3D version of Figure 5-9(b), and in this figure the vertical axis is intensity. Figure 5-9(d) is the result of Figure 5-9(c) by taking peak detection. Figure 5-9(e) is the superposition of Figure 5-9(a) and Figure 5-9(c). From Figure 5-9(e) we can observe that the detected boundaries have high accuracy which is consistent to our assumption.

For the implement of the Local Maximum Detection in CNN, the template can be defined as follows

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} b & b & b \\ b & 0 & b \\ b & b & b \end{bmatrix}, z = 3.5, \tag{5-22}$$

where

$$b = \begin{cases} 0.5 & \text{if } v_{u_{ij}} - v_{u_{kl}} \geq 0 \\ 0 & \text{otherwise} \end{cases}. \tag{5-23}$$

Eq.(5-22) and Eq.(5-23) perform what we need. However, that can not be implemented in CNN-UM because of nonlinear features. Thus, some other solutions for CNN-UM have been required. In this section, BLACK_AND_WHITE_SKELETONIZATION has been suggested for this operation if we need to implement this operation in CNN-UM [65].

5.2 Experiments

In this section, we applied our algorithm to the images which consist of a number of different test patterns. Most of those test patterns are synthesized by textures from Brodatz album [61], and it also has become a standard for evaluating texture algorithms. Each texture pattern we used here are 640*640 pixel 8-bit gray-scale images respectively.

In the experiments, proposed algorithm has been simulated both on MATLAB and CNN-UM. Followed section will discuss the results respectively.

5.2.1 Simulations

In our experiments, first we implement proposed algorithm in computer. There are some parameters need to be selected:

1. The number of Gabor filters and their parameters (U, V, σ) which decide the shape and orientation of the Gabor filters in the frequency domain. The Gabor filtering is

computation intensive, and increasing the number of the Gabor filters will increase computation loading dramatically. On the other hand, unnecessary and useless feature extracted by the wrong-designed Gabor filters may cause wrong boundaries.

2. The σ_g of the post Gaussian filter, which decides the smoothing level. Increasing σ can eliminate more noise, but the accuracy of the boundary may decrease. Because both the Gabor filters and the Gaussian filters have spatial information, the values of σ_g must cooperate with σ to obtain a better result.

Designing above parameters is an important but sophisticated issue. Designing center frequencies of the Gabor filters is discussed in filter-design approaches. There is including the unsupervised methods; such as the algorithm proposed by Jain, and supervised methods; such as algorithm proposed by Dunn [66]. Algorithm in this section is a kind of unsupervised method. That means all of the information of input patterns is unknown. Nevertheless, the emphasis of this section is not on optimizing the design of the Gabor filters, but rather proposing a simple algorithm modeling early vision and being able to implement on CNN. Parameters as follows are empirically chosen and they are all the same in the follows experiments without indicating specifically:

Pattern size(Brodatz texture):	640*640 pixels
Orientation ϕ:	$0°, 45°, 90°, 135°$
Center frequency F:	1/32, 1/16, 3/32, 1/8 cycles/pixel
σ of Gabor filter:	16 pixels
Down sampling rate M:	3
σ_g of post Gaussian filter:	25 pixels
Mask sizes of Gabor and Gaussian:	3σ, $3\sigma_g$

The simulation results have been shown in Figure 5-10. In the results, we have found that most cases of texture boundaries are able to be extracted. Only few cases are not. On the other hand, in the cases which boundaries are not able to be extracted, the results also indicate meaningful properties which consistent to human visual sensation.

5.2.2 Implementation on CNN-UM

In the experiments, proposed algorithm has been implemented on ACE4K Chip. Because of the limitation of current technology, the size of cell array has been limited. That becomes to the major problem in the implementation of proposed algorithm on CNN-UM. For example, the chip we used for implementing proposed algorithm contains cell array 64 by 64. It is too small to analyze the texture. For the sake of higher resolution for obtaining better performance, some necessary operations have been performed before proposed approach: first, each input image has been divided into several sub-images. Next, we process each sub-image respectively with same parameters. Figure 5-11 shows how we divide the input images. Note that there are overlapped areas between sub-images because we have to avoid the boundary effects of the cell array.

Another issue in the implementation of proposed algorithm on CNN-UM is that we are not able to implement non-linear templates on the chip. Thus, several proposed operation is not able to be implemented directly. For those operations, we choose another operation which is able to be implemented on the chip and the result is similar to what we expected. Even more, we have to disable some operations in proposed algorithm and take the trade-off. For example, Local Maximum Detection operation contains nonlinear template and thus, it can not be implemented on ACK4K Chip. Hence, we choose BLACK_AND_WHITE_SKELETONIZATION to replace it. Figure 5-12 shows the results of implementation of proposed algorithm on ACE4K Chip.

5.3 Discussions

In this section, we discuss some properties of the proposed approach. The way we estimate the error is as follows:

1. Only the case that synthesizes two texture patterns in Brodatz texture is considered. In the

algorithm, every boundary is independently. It is hard to judge the accuracy if we consider multi-boundaries simultaneously. Especially when some boundaries are detected and the others are not.

2. The distance between the answer and the result detected by the algorithm is measured in the condition of boundary which is detectable. We define the error by dividing measured distance into the number of total pixels.
3. For simplicity, 70 textures of Brodatz textures, which are generally consistent to our definition of textures, are picked. Each test image is synthesized by choosing two textures from the 70 textures randomly, we test 500 of combinations. Some examples have been shown in Figure 5-13.

Figure 5-14 is a histogram of error estimation in our experiment, and the results with error less than 5% is account for 85% for test images. The average of the error is less than 5%. The smallest error is 0.76%. Note that the images with bigger estimated errors are reasonable. This kind of examples has been shown in Figure 5-15, Figure 5-16, and Figure 5-17. In these examples, the boundaries between different textures (middle line) exist, but they are weaker than local boundaries caused by non-uniform regions. For the sake of simplicity, only the largest peaks are kept during error estimation, so the boundaries in the middle are not kept in the results. Although in these examples, the outputs are consistent to human visual perception. Their errors are quite big. We have found that it is hard to define a generally "correct answer" for all test images in human vision system, and the method we measure the error is probably not suited for those kind of test images. For this reason, the measurement is not necessary for the input images synthesized by the rest 42 textures in Brodatz textures.

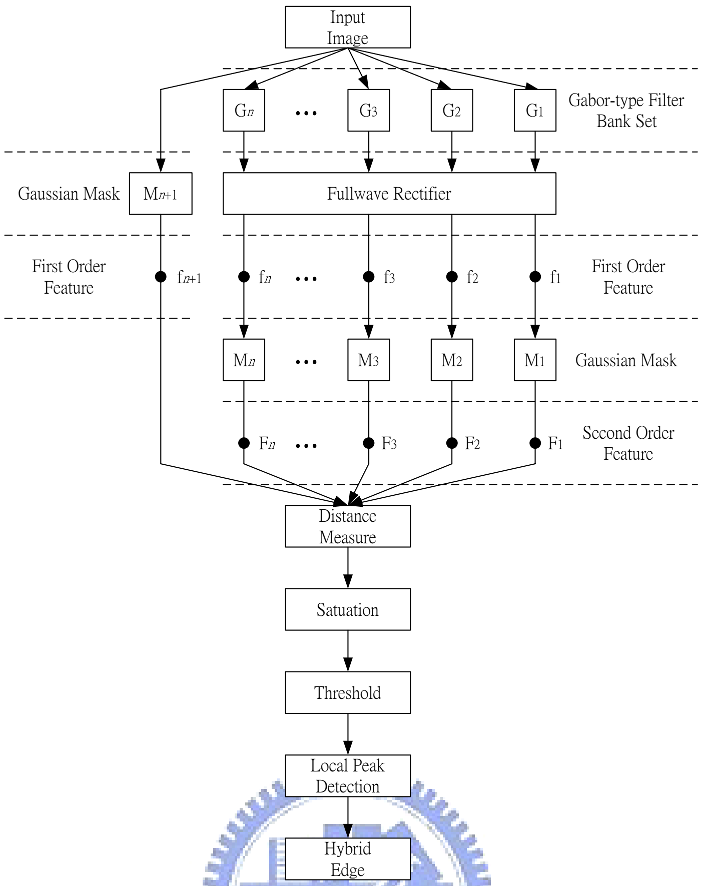


Figure 5-1. The diagram of proposed algorithm

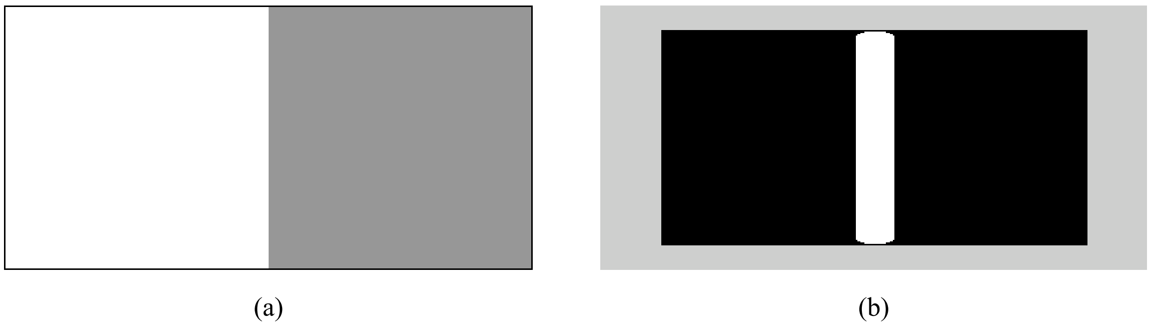
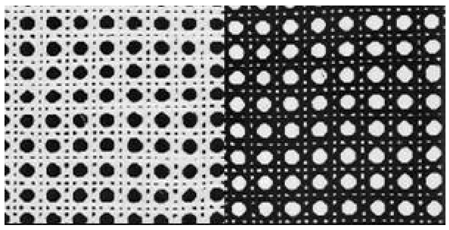
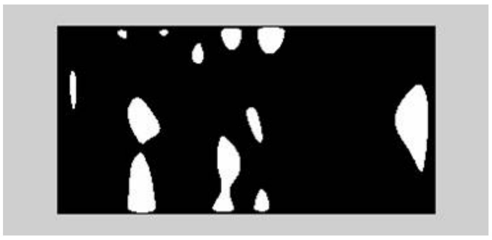


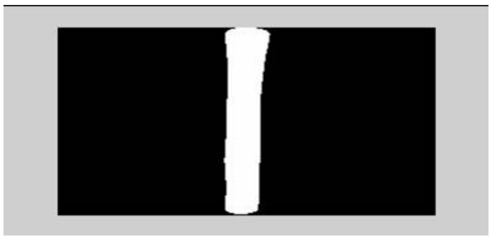
Figure 5-2. An example which demonstrated coarse boundary detected by first order feature, where (a) input image, and (b) represents the boundaries has been extracted from first order features.



(a)

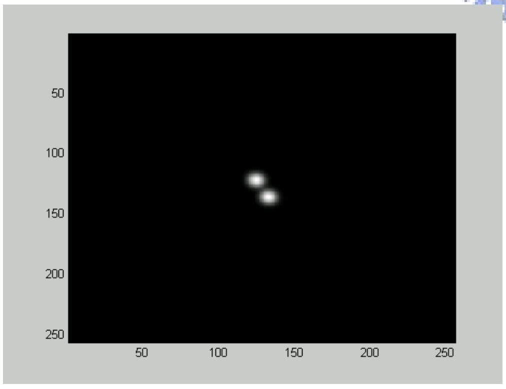


(b)

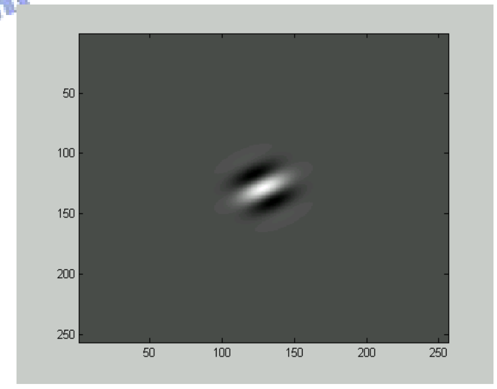


(c)

Figure 5-3. An example which demonstrated the effect of first order feature in boundary detection. (a) is the input images which can be obtained from Brodatz texture database (D101-D102) [61], (b) represents the boundary has been detected by second order feature, and (c) represents the boundary has been detected by first order feature.

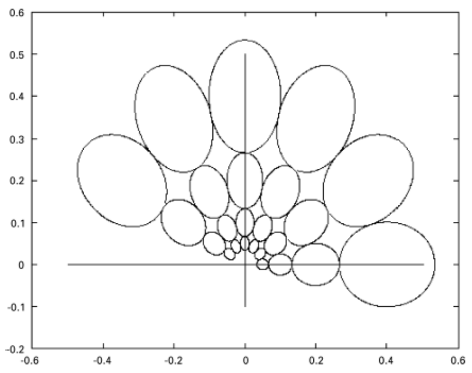


(a)

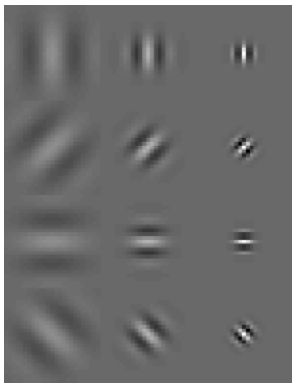


(b)

Figure 5-4. An example of 2D Gabor type filtering. (a) is a standard 2D Gabor type filter in time domain, and (b) is in frequency domain.



(a)



(b)

Figure 5-5. An example of Gabor filter dictionary. (a) represents the Gabor-type filter bank set, and (b) is the Feature space of Gabor filter dictionary.

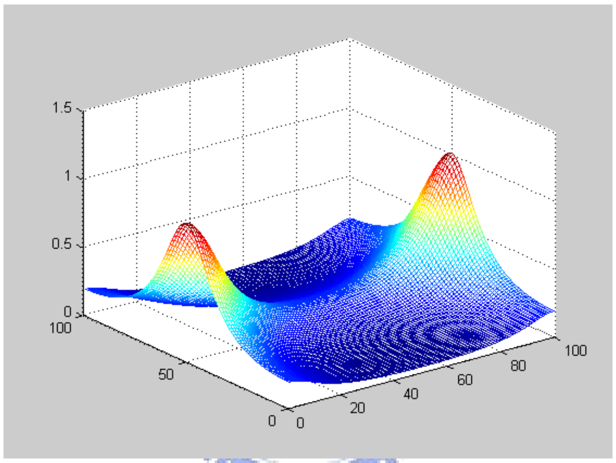
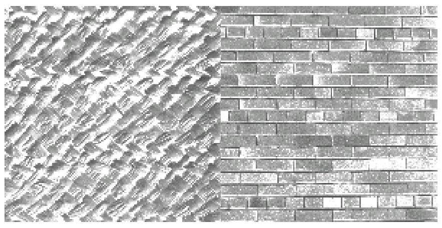
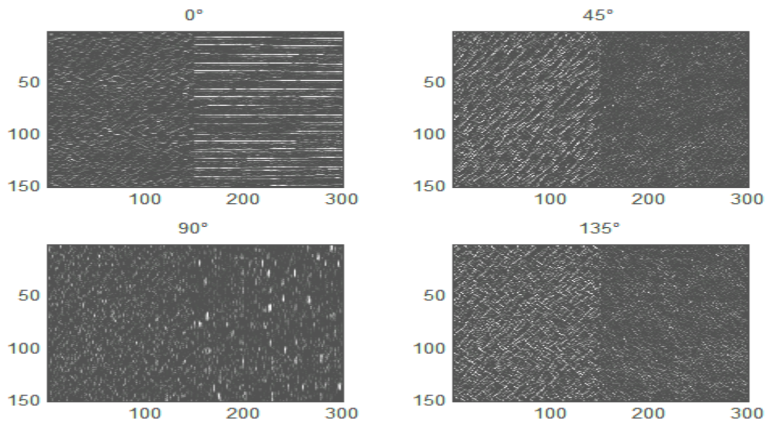


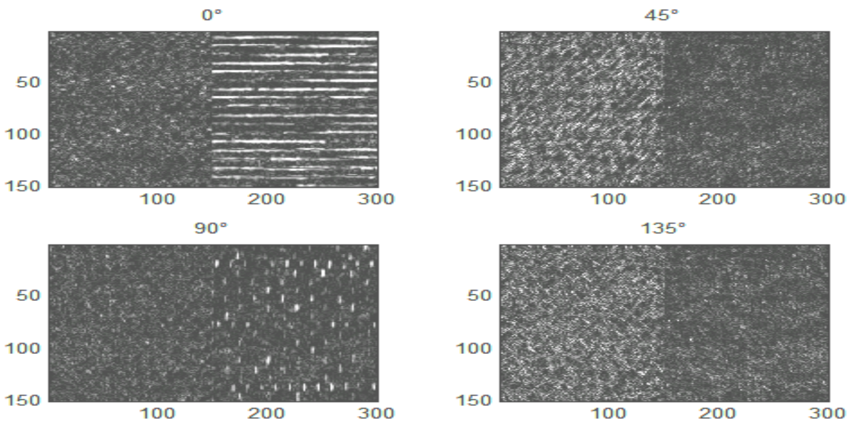
Figure 5-6. An example of proposed Gabor-type filtering.



(a)

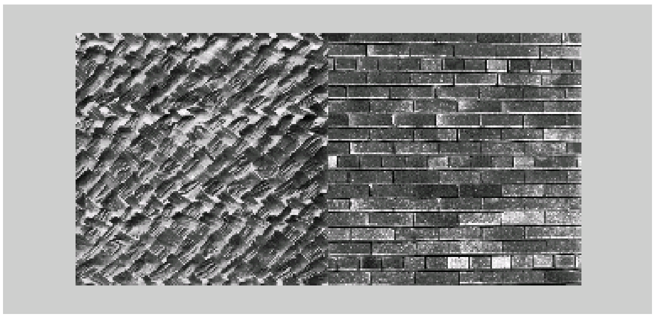


(b)

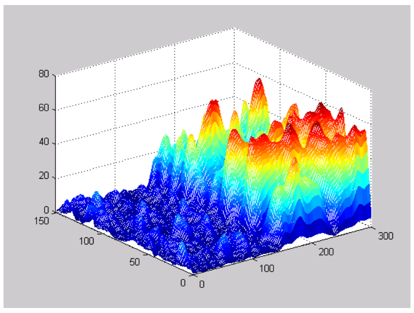


(c)

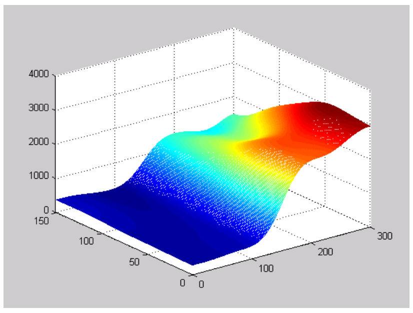
Figure 5-7. An example demonstrates the effect of rectifying (a) input; (b) output without rectifying; (c) output with rectifying



(a)

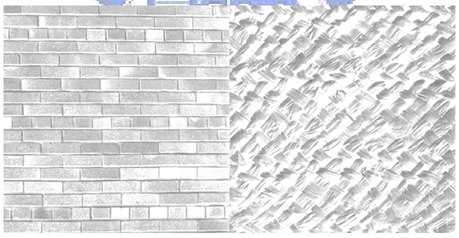


(b)

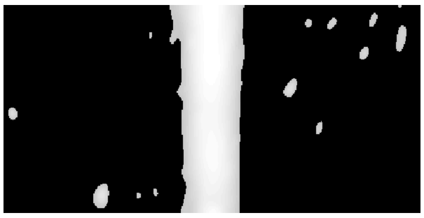


(c)

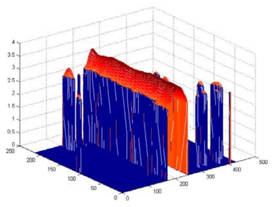
Figure 5-8. (a) input; (b) output before rectification; (c) output after rectification.



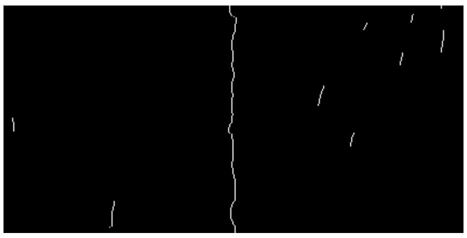
(a)



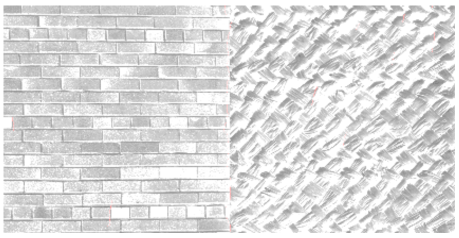
(b)



(c)



(d)



(e)

Figure 5-9. (a) is the input image, (b) is the coarse boundary, (c) is the 3D version of (b), (d) is the results after peak detection.

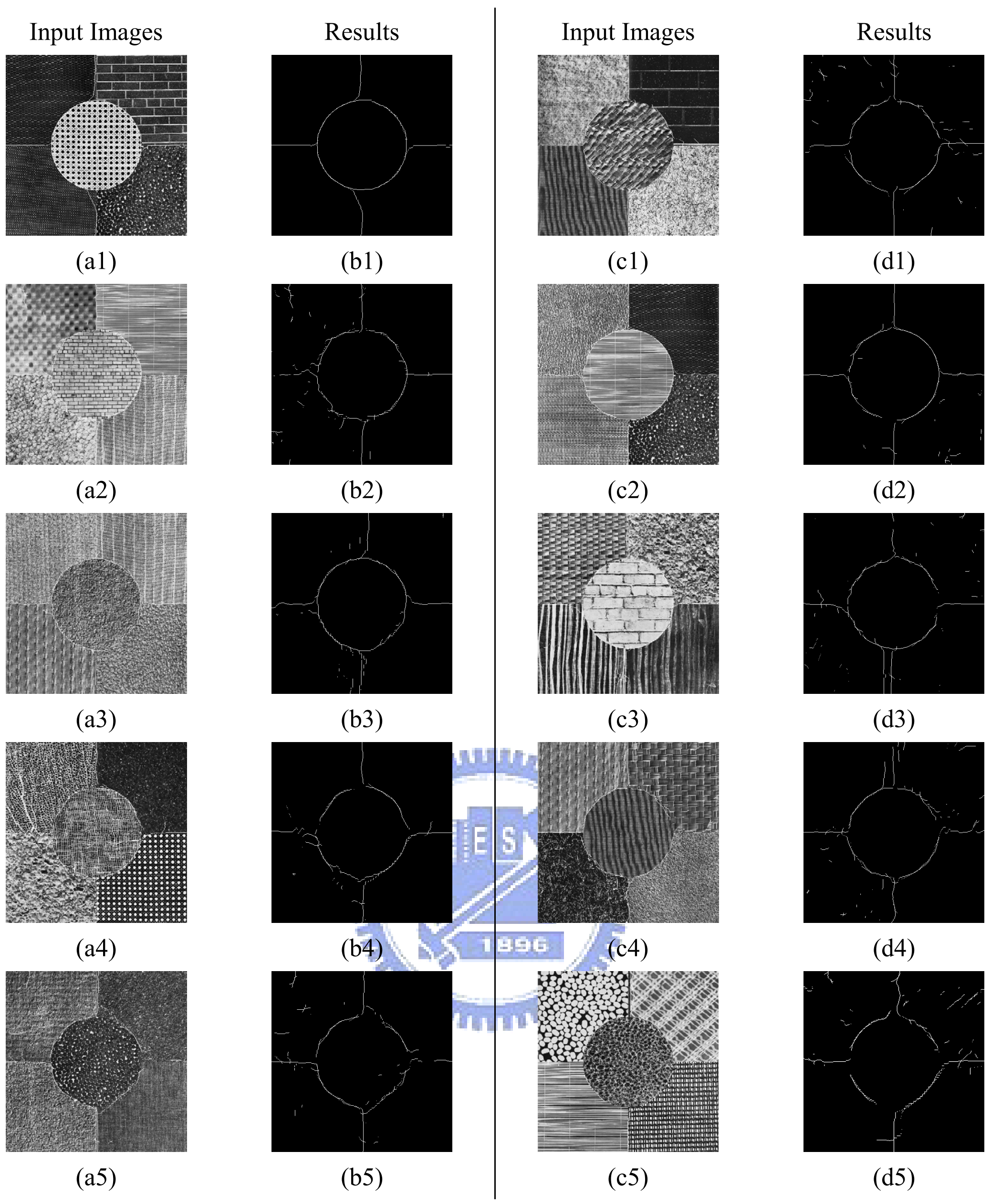


Figure 5-10. The simulation results of proposed algorithm.

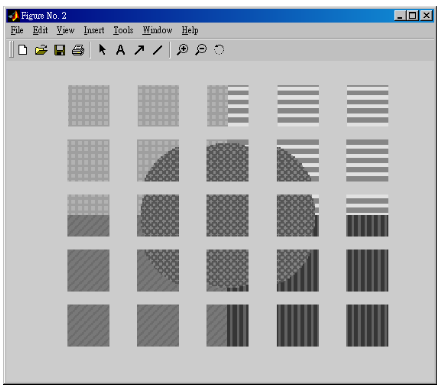


Figure 5-11. The example of dividing of the input images.

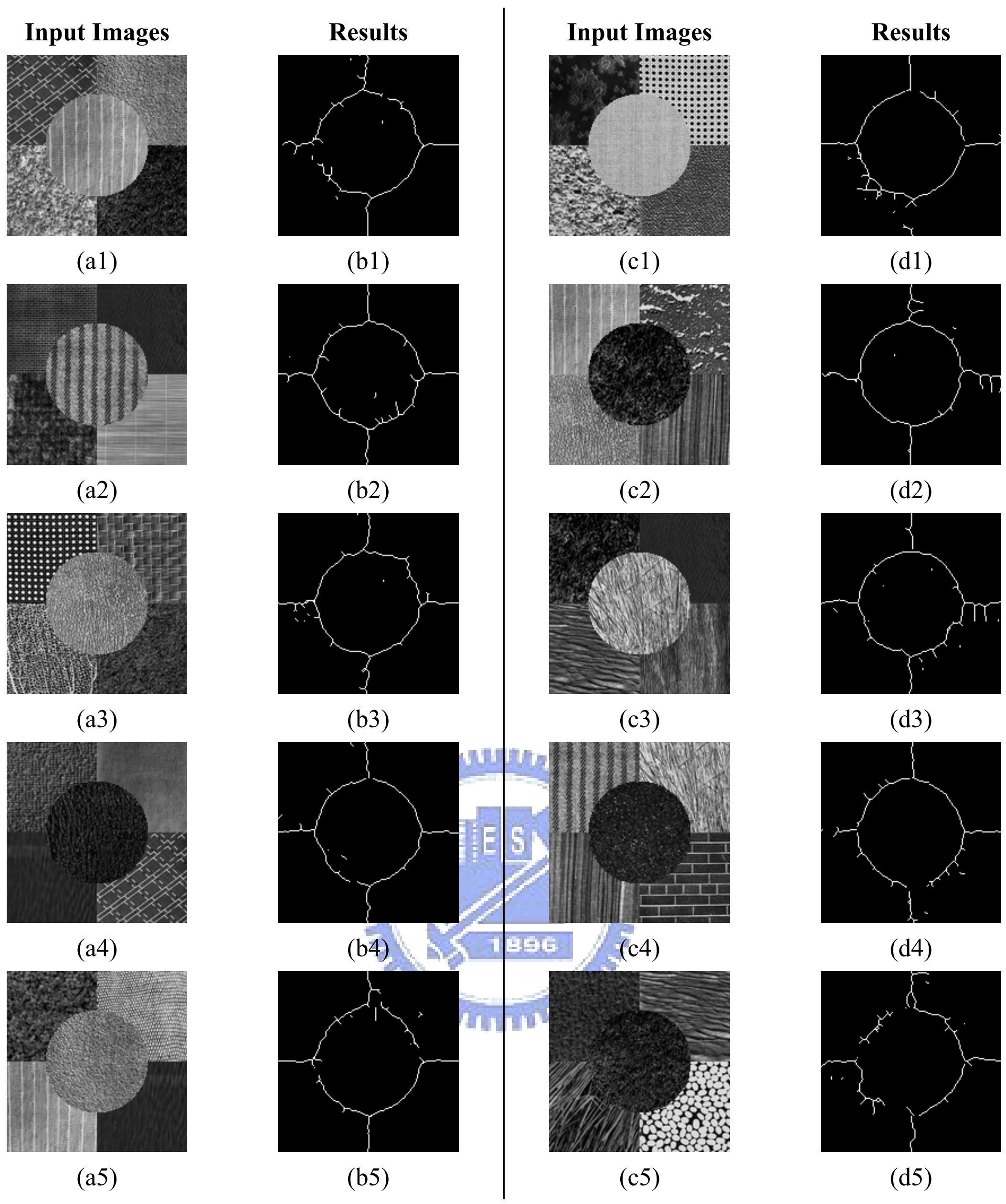


Figure 5-12. The implementation results of proposed algorithm.

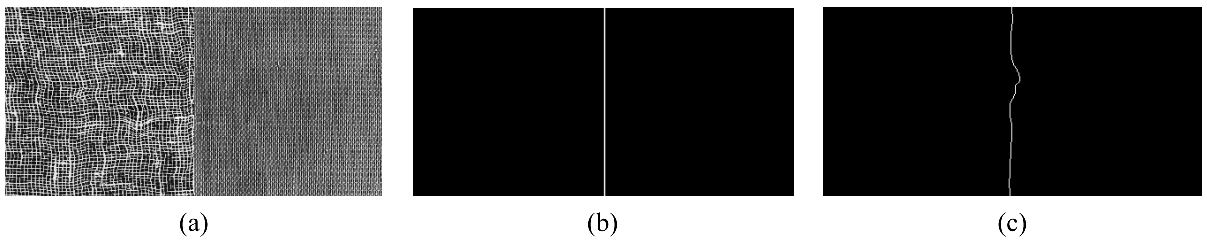


Figure 5-13. An example of error estimation (a) input; (b) answer(middle line); (c) output;

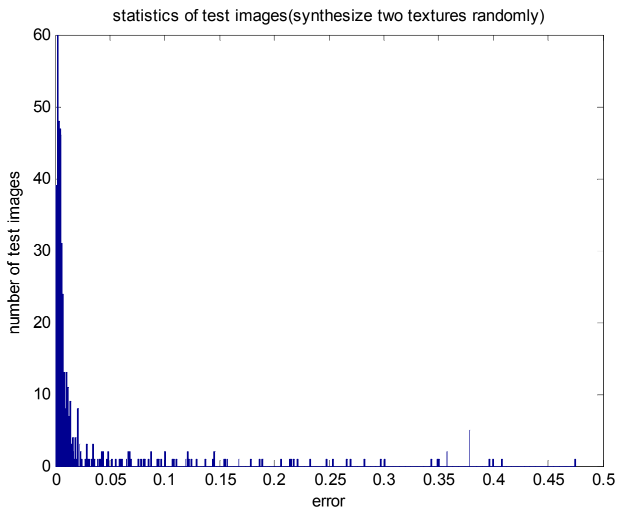


Figure 5-14. Histogram of error estimation

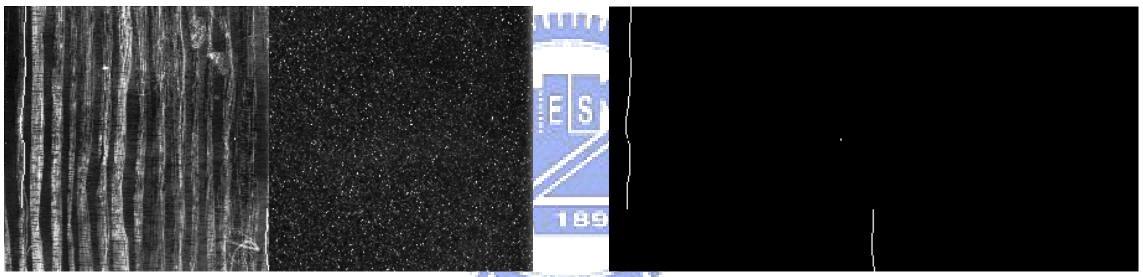


Figure 5-15. An example of test image with big estimation errors(D50D32); (a) input; (b) output;

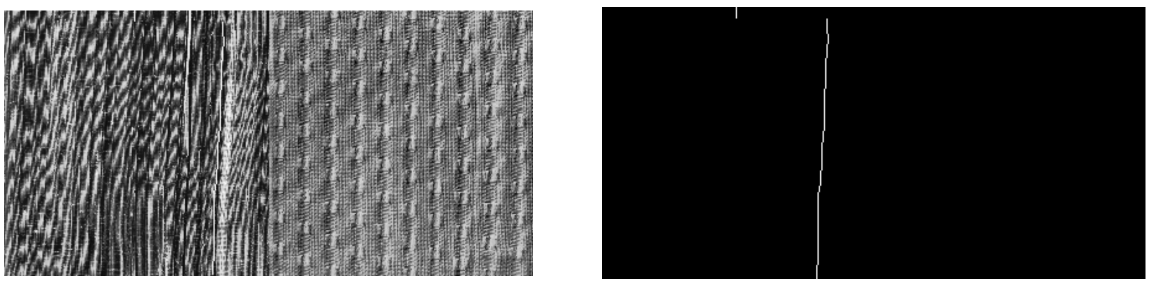
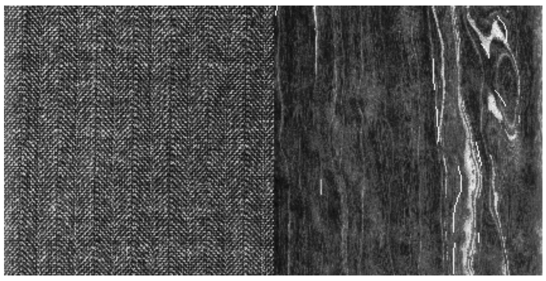


Figure 5-16. An example of test image with big estimation errors(D105D83); (a) input; (b) output;


(a)

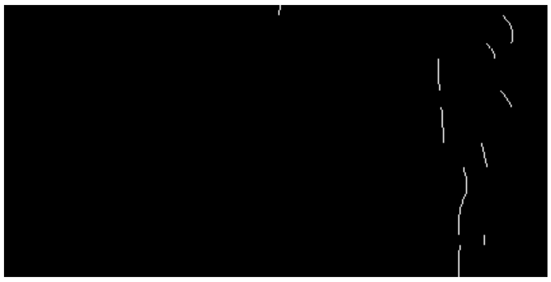

(b)

Figure 5-17. An example of test image with big estimation errors(D17D70); (a) input; (b) output;

6 APPLICATIONS

6.1 Image Sharpness Improvement

In the human vision system, the second order features can provide [the related difference of light intensity between the cells. This is why the human vision system can reliably identify different textures under variable light conditions. On the other hand, a kind of amacrine cell, named biplexiform ganglion cell, is directly connected to the photoreceptors. The biplexform ganglion cell are depolarising in response to increases in photon catch and as a result, it can provide information regarding ambient light level, which is useful for controlling pupil diameter and diurnal body rhythm. Thus, the responses of the connected ganglion can be used to implement the improvement in sharpness. For the luminance channel of CIE-L*a*b*, the following equation can be applied

$$\begin{aligned} L'(\mathbf{k}) = L(\mathbf{k}) &+ \alpha_1 \cdot CNN_{\text{M-Ganglion},\varepsilon_1,b_1,g_1,\lambda_{1,R_u},\lambda_{1,G}}\left(L(\mathbf{k})\right) \\ &+ \ldots + \alpha_n \cdot CNN_{\text{M-Ganglion},\varepsilon_n,b_n,g_n,\lambda_{n,R_u},\lambda_{n,G}}\left(L(\mathbf{k})\right), \end{aligned} \tag{6-1}$$

where $\alpha_1, \alpha_2, \ldots, \alpha_n$ represent the function of the biplexiforms. Figure 6-1 shows an example of image sharpness, where Figure 6-1(a) is the original input, and Figure 6-1(b) is the output. In fact, this algorithm matches the conclusions of Kotera. Korera previously proposed a sharpness improvement algorithm based on the detection of adaptive edges [52].

6.2 Color Constancy

According to the gray world hypothesis, in the absence of other colors the world is perceived as gray [53, 54]. Based on this assumption, the proposed model can be used to estimate the shift in the light. Furthermore, light shifting can be removed via this model.

The stimulation outputs of a horizontal cell are fed back to the connected photoreceptors to reduce the influence of photoreceptors on the bipolar cells. This lateral inhibition results in activation of any one photoreceptor reducing that of surrounding photoreceptors. In some cases, the central and peripheral photoreceptors react to same kind of the stimulations and thus exert a mutually depressive effect [1]. A monochromatic system provides an example. Thus, this study concludes that a special case of central/surround structure stands as follows

$$a'(\mathbf{k}) = \alpha_a \cdot \left(a(\mathbf{k}) - CNN_{\text{M-Ganglion},\varepsilon,b,g,\lambda_R}\left(a(\mathbf{k})\right)\right) + \beta_a \text{, and}$$

$$b'(\mathbf{k}) = \alpha_b \cdot \left(b(\mathbf{k}) - CNN_{\text{M-Ganglion},\varepsilon,b,g,\lambda_R}\left(b(\mathbf{k})\right)\right) + \beta_b \text{,} \tag{6-2}$$

where ε is a small value, $b = 0$, $g = 1$, $\lambda_R = 0.6$, α_a and α_b are the related scale, and β_a and β_b represent the definition of "gray". Generally, the α and β are unnecessary. However, in some situations the related environment of the input image is quite extreme and thus the range of the color space is limited. In such cases, α and β are required to restore the image. Figure 6-2 shows an example and the results.

6.3 Video Auto Adjustment and Enhancement

Since the model parameters are obtained from the input image, the computer fovea model can be used to perform adaptive adjustment and enhancement of sequential images (*i.e.* video). This experiment applied the proposed visual information from the enhancing algorithm for the each image in the videos. Figure 6-3 shows the experiment results, where Figure 6-3(a) and (c) are the inputs, while (b) and (d) are the video outputs. Notably, the algorithm is adaptively adjusted during processing.

6.4 Texture Classification Algorithm with Rotation / Scale Invariant Properties

The algorithm we proposed in this section can be described as following; First, we pickup two of texture images which we want to compare. Next, we put those two texture images into the feature extractor, respectively. The feature extractor is a set of Gabor type filter. Each filter in the set extracts specific feature. After we collected all features we need, we used feature classifier to determine how those two input texture image similar to each other. The output of feature classifier is a result of Euclidean Distance. That can be explained as the “difference” between those two input texture images. On the other hand, lower result means those two of input texture image is more similar to each other. Following describes the method we need.

Performing the Principal Component Analysis (PCA) of a stationary multivariate random process means computing the eigenvectors of it's covariance matrix corresponding to the largest eigenvalues, and the projection of the samples of the multivariate process on the eigenvectors to obtain a number of principal components. Based on this well known concept, Oja (1982) first introduced a learning rule enabling a linear neuron $y = \omega \cdot x$ to extract the first principal component of it's input data:

$$\omega(t+1) = \omega(t) + \eta(t)\left[y(t)x(t) - y^2(t)\omega(t)\right], \tag{6-3}$$

where x is the input vector, ω is the weight vector, y is the output, and η is the so called learning rate which maybe decreased with time and usually between 0.01 and 0.0001.

The term $y(t)x(t)$ is maximizing output variance which also known as Hebbian algorithm and $y^2(t)\omega(t)$ is a normalizing factor which has been designed to keep $\|\omega\|$ close to 1. The architecture of described Neural PCA can be shown in Figure 6-4.

In proposed approach, we used Brodatz texture image database. Some samples of database have been shown in Figure 6-5, the approaches in proposed thesis have been described as followed sections.

6.4.1 Rotation-invariant Texture Classifier

Rotation-invariant Texture Classifier can be described as following: First, we used a Gabor filter dictionary to generate a set of data for each texture image:

$$R_{K,S}(x,y) = I(x,y) \otimes g_{K,S}(x,y), \tag{6-4}$$

where $g_{K,S}(x,y)$ is the so-called Gabor-like operator which has been described before, $I(x,y)$ is input image, and $\otimes$ represented mathematical convolution.

Next, we summarized the magnitude of the output which has been generated by Eq.(6-4) as following

$$T_{K,S} = \sum_{x,y} \left[R_{K,S}(x,y) \right], \tag{6-5}$$

where K represented the orientation, and S, the frequency.

Third, we applied Eq.(6-5) for all frequency and orientation. And then, we obtained a table which has K rows and S columns. Each row represents a vector which has been represented by x_i.

Forth, we made each vectors to be zero-mean vector. We worked out the mean of each vectors and adjusted it:

$$m = \frac{1}{M} \sum_{i=1}^{M} x_i, \tag{6-6}$$

Finally, we have to find a set of e_i which have the largest possible projection onto each of the ω_i. Thus, let's go to find a set of M orthonormal vectors e_i for which the quantity:

$$\lambda_i = \frac{1}{M}\sum_{n=1}^{M}\left(e_i^T \omega_n\right)^2, \tag{6-7}$$

is maximized with the orthonormality constraint:

$$e_l^T e_k = \delta_{lk}. \tag{6-8}$$

Hence, an ordered vector had been obtained; higher value in the diagonal of eigenvalues e_i represented higher deviance. We chosen the one element which mapped to largest deviance in the eigenvector λ_i. This element can be thought as the feature of input texture image. Assume we had two of texture image which had been represented by I_a and I_b, and the features of those two of texture image had been represented by e_a and e_b. We can calculate the Euclidian Distance as following

$$E_{a,b} = \sum\left(e_a - e_b\right)^2. \tag{6-9}$$

Consequently, higher $E_{a,b}$ means difference degree between two texture image. Note that the rotation of the texture image would not cause any effects in the results.

6.4.2 Scale-invariant Texture Classifier

The Scale-invariant Texture Classifier is similar to Rotation-invariant Texture Classifier, expect the role of K and S. Originally, in Rotation-invariant Texture Classifier, \$K\$ is represented as orientation, and S is frequency. But in Scale-invariant Texture Classifier, K is represented as frequency, and S is orientation. On the other hand, the only difference between Rotation-invariant Texture Classifier and Scale-invariant Texture Classifier is that the role of axis in the feature space has been exchanged.

6.4.3 Experiments

In proposed thesis, we used Brodatz texture image database. Some images have been shown in Figure 6-5. Here from Figure 6-5(a1) to Figure 6-5(d4), some of images there are similar to others expect orientation. From Figure 6-5(a5) to Figure 6-5(d8), some of images there are similar to others expect scale.

6.4.3.1 Rotation-invariant Texture Classifier

Figure 6-6 is the value in the diagonal of eigenvalues e_i represented deviation degree. We can see that almost all information located at first 5% to 10%. This fact implied the deviation degree of data in all dimension have quite large different. Thus, we are able to apply proposed method for each texture image in Figure 6-5 and extracted the first principle component which has highest deviation degree. As shown in Figure 6-7, the feature vector of Figure 6-5(a1), (b1), (c1), and (d1) are shown in Figure 6-7(a), and the feature vector of Figure 6-5(a1), (a2), (a3), and (a4) are shown in Figure 6-7 (b). Clearly, Figure 6-5(a1), (b1), (c1), and (d1) are same texture e

Consequently, as shown in Figure 6-7, the difference between the feature vectors of Figure 6-5(a1), (a2), (a3), and (a4) are quite small. On the other hand, Figure 6-5(a1), (a2), (a3), and (a4) is quite different texture. As result, the feature vectors of Figure 6-5(a1), (a2), (a3), and (a4) is quite different. That have been shown in Figure 6-7(b).

Table 4-2 represents the mutual comparison of Euclidian Distance between the results of proposed thesis with the input texture images in Figure 6-5. This table identified the most interesting part in this section. For example, Figure 6-5(a1), (b1), (c1), and (d1) is same texture but different orientation. As shown in Table 4-2, the comparison of Euclidian Distance between Figure 6-5(a1), (b1), (c1), and (d1) are quite small, and the others are bigger. In the experiment of Rotation-invariant Texture Classifier, the percentage of correct classification is about 97.5%.

6.4.3.2 Scale-invariant Texture Classifier

The proposed Scale-invaliant Texture Classifier is very similar to the Rotation-invaliant Texture Classifier. The only different is the structure of the input data of the PCA Neural Networks. Figure 6-8 is the value in the diagonal of eigenvalues e_i represented deviation degree. We can see that almost all information located at first 5% to 10%. As the description of last section, we are also able to apply proposed method for each texture image in Figure 6-5 and extracted the first principle component which has highest deviation degree. As shown in Figure 6-9, the feature vector of Figure 6-5(a5), (b5), (c5), and (d5) are shown in Figure 6-9(a), and the feature vector of Figure 6-5(a5), (a6), (a7), and (a8) are shown in Figure 6-9(b). Clearly, Figure 6-5(a5), (b5), (c5), and (d5) is quite small. On the other hand, Figure 6-5(a5), (a6), (a7), and (a8) are also quite different. Those have been shown in Figure 6-9(b).

Table 4-3 represents the mutual comparison of Euclidian Distance between the results of proposed thesis with the input texture images in Figure 6-5. This table identified the most interesting parts in this section. For example, Figure 6-5(a5), (b5), (c5), and (d5) are quite small, and the others are bigger. In the experiment of Scale-invariant Texture Classifier, the percentage of correct classification is about 92.5%.

6.4.4 Discussions

In this section, we proposed a pair of approaches which can use in rotated texture and scaled texture image classification. The result is good. In our experiments, the correct classify rate of rotated texture image classification is about 97.5%, and the correct classify rate of scaled texture image classification is about 92.5%.

On the other hand, the proposed method in this section should be used for rotation and scale respectively. However, in the nature world, the rotation and scale phenomena usually appeared at same time. It will cause lower correct classify rate. It's a major problem in proposed thesis. How to conquer this problem will be our future works.

6.5 Texture Identification Algorithm based on CNN-based Gabor-type Filtering and Self-Organized Fuzzy Inference Neural Networks

In human's vision system, texture is a basic cue for human beings to recognize different kinds of objects in the world. The research of texture analysis is a very important task in computer vision and its applications. In other words, studying the boundary between different textures also becomes a more important task. In decades, those two areas have been a very active topic.

Studying the frequency responses of textures is the trunk stream of texture segmentation area. One of major research is Gabor-type filtering, which is proposed by Gabor (1946) [16]. In recent years, researchers found that Gabor-type filtering is also suitable for texture analysis. Thus, several papers which have been proposed are based on this architecture [67]. However, there are two questions we may ask; first, how can we extract the information based on Gabor type filtering, second, how to select the parameters of proposed methods. In this thesis, we proposed an alternate solution for these problems.

This thesis is inspired by the architecture of retina. As the well known, there is not only one kind of cell exists on the retina. In fact, several kinds of cell have been distributed on it. These cells are in response to different tasks. For example, we have cone and rod cells which are in response to color and intensity respectively. Thus, to study the mechanisms between cells which provide different functions become an interesting task.

6.5.1 Self-Organized Fuzzy Inference Neural Networks

The proposed method can be described as two major components; one is Gabor-type Filtering Cellular Neural Networks which used to simulate human's eyes, Self-Organized Fuzzy Inference

Neural Networks which used to simulate the brain. Gabor-type Filtering Cellular Neural Network is described in the previous chapter. Self-Organized Fuzzy Inference Neural Network is introducted as the following.

Self-Organized Fuzzy Inference Neural Network, also known as SOFIN, is first introduced by Lin [68]. SOFIN is a fuzzy neural network which provides self-adaptive ability. In this thesis, SOFIN has been used to simulate two major ability of human's brain; one is inference, the other is memory. The architecture has been shown in Figure 6-10. Following described the major function of SOFIN.

Layer 1: No computation is done by this layer. Each node in this layer, which corresponds to one input variable, only transmits the input values to the next layer directly, that is

$$f = u_i^{(1)}, o_i^{(1)} = a^{(1)}(f) = f. \tag{6-10}$$

From the above equation, the link weight in layer one $\left[\omega_i^{(1)}\right]$ is unity.

Layer 2: Each node in this layer corresponds to one linguistic value (small, large, etc.) of one of the input variables in Layer 1. In other words, the membership value which specifies how the input value belongs to the fuzzy set which is generated in Layer 2. There are many choices for the types of membership functions for using, such as triangular, trapezoidal, or Gaussian.

Layer 3: A node in this layer represents one fuzzy logic rule and performs precondition matching of a rule. In this thesis, we use followed AND operation for each Layer 3 node, where

$$f\left[u_i^{(3)}\right] = \prod_i u_i^{(3)}, o_i^{(3)} = a^{(3)}(f) = f. \tag{6-11}$$

The link weight in Layer 3 $\left[\omega_i^{(3)}\right]$ is unity. The output f of a Layer 3 node represents the firing strength of the corresponding fully rule.

Layer 4: This layer is so called consequent layer. Different nodes in Layer 3 maybe connect to same node in Layer 4, which means that same consequent fuzzy set is specified for different rules. Whole function which has been performed by this layer can be represented as following;

$$f\left[u_i^{(4)}\right] = u_i^{(4)}, o_i^{(4)} = a^{(4)}\left(f\right) = a^{(4)} \cdot f. \tag{6-12}$$

Layer 5: Each node in this layer corresponds to one output variable. The node integrates all the actions recommended by Layer 4 and acts as a defuzzifier as following;

$$f\left[u_i^{(5)}\right] = \sum_i u_i^{(5)}, y_{out} = a^{(5)}\left(f\right) = f. \tag{6-13}$$

6.5.2 Proposed Algorithm

The algorithm which has been proposed in this study can be described as seven layers. The architecture is shown in Figure 6-11. Following described the details of proposed algorithm:

Input Layer: This layer actually does not contain any computing. In fact, this layer can be considered as an image buffer. The input image has been loaded in this buffer and distributed into next layer, that is as following;

$$O_{1\text{st}}\left(i,j\right) = L_{1\text{st}}\left(I\left(i,j\right)\right). \tag{6-14}$$

where $O_{1\text{st}}\left(i,j\right)$ is the output image of this layer, $I\left(i,j\right)$ is the input of this layer, and $L_{1\text{st}}\left(\cdot\right)$ represents the processor of this layer.

Gabor-type Filtering CNN Array Layer: This layer is used to collect the information of texture which exists in the input images. Clearly, second layer looks like an array which is constructed by many CNN processors. Each CNN processor provides Gabor-type filtering with different parameters. The function of this layer can be represented as following;

$$O_{2\mathrm{nd},m,n}(i,j) = L_{2\mathrm{nd},m,n}\left(O_{1\mathrm{st}}(i,j)\right), 1 \le m \le M, 1 \le n \le N, \qquad (6\text{-}15)$$

where $O_{2\mathrm{nd},m,n}(i,j)$ is the output image of second layer, $O_{1\mathrm{st}}(i,j)$ is the input of second layer which is also the output image of first layer, and $L_{2\mathrm{nd},m,n}(\cdot)$ are the CNN processors in second layer. Note that MN represents the number of CNN processors in this layer.

The output image of this layer will be putted into next layer. However, it is not applied directly. We need to reconstruct it as following;

$$I_{3\mathrm{rd}}(i,j) = O_{2nd}^{mn}(i,j), 1 \le m \le M, 1 \le n \le N. \qquad (6\text{-}16)$$

Thus, we can see that $I_{3\mathrm{rd}}(i,j)$ becomes to a multi-dimension value and dimension are M by N.

Rectification Layer: This layer provides two functions; one is rectification operator, the other is to collect the output from different Gabor-type Filtering CNN. Rectification operator is used to compensate the effects of phase of output of Gabor-type Filtering. In other words, we only consider about the degree of difference between different responses of Gabor-type filtering. Each node in this layer will collect the output from different Gabor-type Filtering CNN. Thus, the output of this layer becomes to multi-dimension values. The function of this layer is as following;

$$O_{3rd}^{mn}(i,j) = L_{3\mathrm{rd}}\left(I_{3rd}^{mn}(i,j)\right), L_{3\mathrm{rd}}(\cdot) = abs(\cdot). \qquad (6\text{-}17)$$

Input Linguistic Layer: This layer is the Layer 1 of SOFIN. This layer contains no any computation but uses to store the input variable. For details please refer to previous chapter.

Input Term Layer: This layer is the Layer 2 of SOFIN. This layer provides fuzzification function for the input variables. In proposed algorithm, previous node will provide multi-dimension values. Thus, this layer needs to fuzzify each dimension in every node. And then, this layer needs to distribute the value of each node and each dimension to next layer. In this thesis, we used Gaussian Probability Distribution Function as the membership function. The examples have been shown in Figure 6-12. Note the computation of the node of this layer also related to each node of neighbor pixels. This is different to traditional SOFIN.

Rule Layer: This layer is used to store the training results. Each training stage maybe (or maybe not) grow the connection and the scale of this layer. The details please refer to previous chapter.

Output Term Layer: This layer can be considered as a consequent operator. The details please refer to previous chapter.

Output Linguistic Layer: This layer is the defuzzification layer. In training stage, the desired values will be assigned in this layer. And in testing stage, the testing results will be generated by this layer. The details please refer to previous chapter.

6.5.3 Experiments

We used Brodatz texture image database [61] to be the training and testing texture database. A typical result has been shown in Figure 6-13 and Figure 6-14. Firstly, we put the training pattern which has been shown in Figure 6-13 and assign the desired output for each testing pattern. The desired output for Figure 6-13(a) is 0.75, for Figure 6-13(b) is 0.25, for Figure 6-13(c) is -0.25, and for Figure 6-13(d) is -0.75. Note that the range of output is $[-1,1]$. We can see that the testing output image is much similar to desired output image. Note that the left-bottom image of Figure 6-14(a) contained some error. The causes related the parameters of Gabor-type Filtering CNN.

6.5.4 Application: A Car Plate Segmentation Approach based on Proposed Model

Based on the proposed algorithm, a car plate segmentation has been implemented. We simply use an training image as a training pattern for the segmentation of black-white type car plate. Figure 6-15(a) shows a training pattern. The desired output is shown in Figure 6-15(b). The testing results have been shown in Figure 6-16. In the results, the area of the car plate has been successfully extrated.

6.5.5 Discussions

The algorithm which has been proposed in this thesis presents very well performance. We had found few items which can be our future works. First, not only the parameters of Gabor-type Filtering CNN Array need to be set by manually, but also the SOFIN itself. Currently the parameters setting are based on our experience. A more advanced architecture which can be used to provide those parameters seems to be required. Second, we are going to analyze the features of the border between two different textures. In our experiments, the border presents some interesting features. Those will be our major future works.

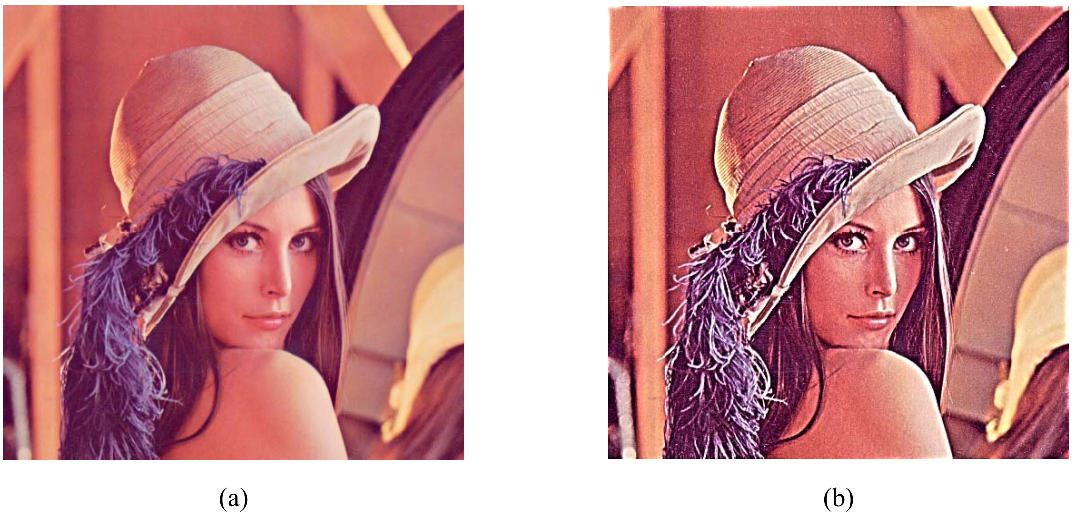
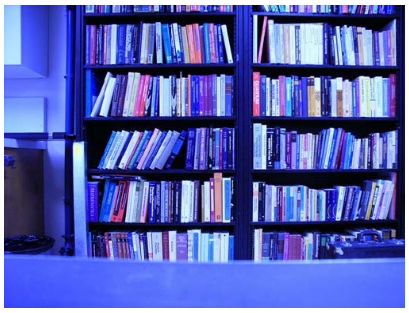
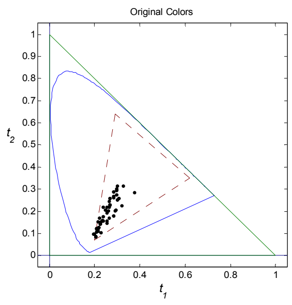
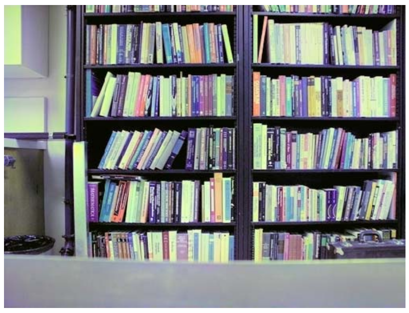
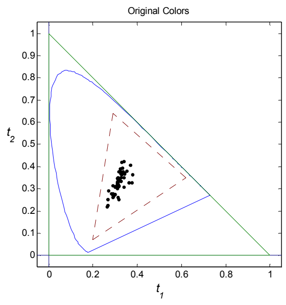


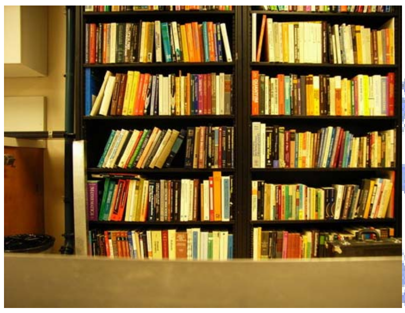
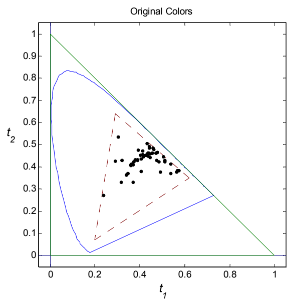
Figure 6-1. An example of improvement in image sharpness, where (a) is the input and (b) shows the output. The images can be obtained at (http://cnn.cn.nctu.edu.tw/~chhuang/paper/hCNNCFM/).

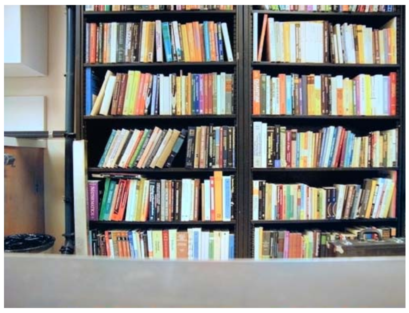
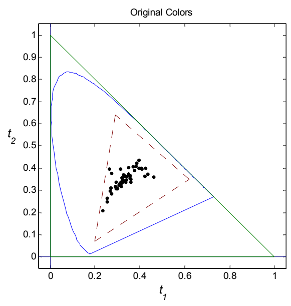
(a)

(b)

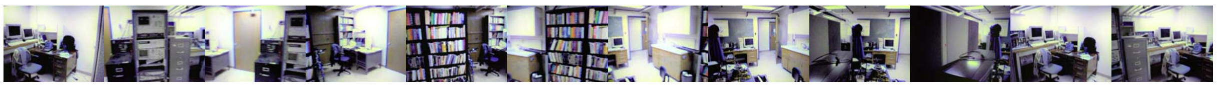
(c)

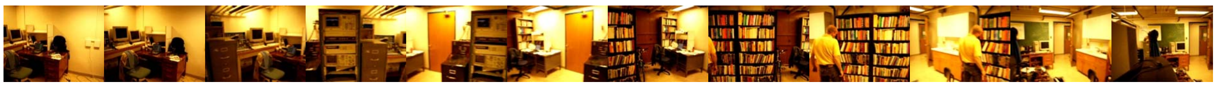



(d)

Figure 6-2. Two results of the proposed image color constancy algorithm. First, a photo is taken under a blue light condition. The photo is shown in (a), and the reconstructed result is shown in (b). Next, a photo is taken of the same scene under yellow light condition, as shown in (c), and the reconstructed result is shown in (d). The chromatic diagrams are shown to the side of the images. The images can be obtained at (http://cnn.cn.nctu.edu.tw/~chhuang/paper/hCNNCFM/).

(a)


(b)


(c)

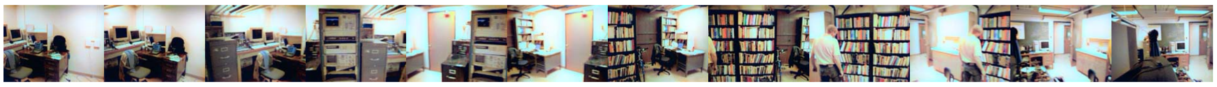

(d)

Figure 6-3. Two experimental results of the integrated video auto adjustment and enhancing algorithm, where(a), (c) are the input videos, (c) and (d) are the reconstructed outputs. The full video can be obtained at (http://cnn.cn.nctu.edu.tw/~chhuang/paper/hCNNCFM/).

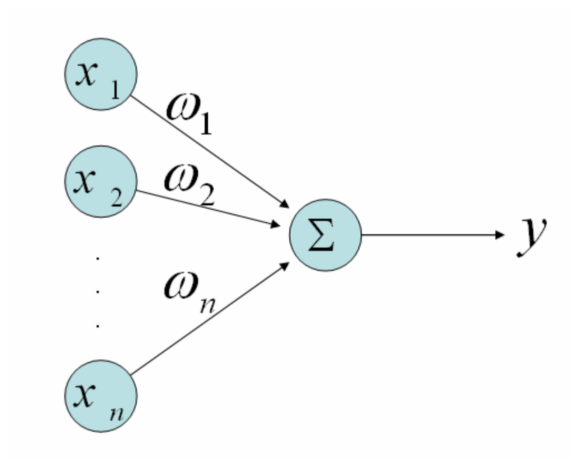


Figure 6-4. The architecture of Neural PCA.

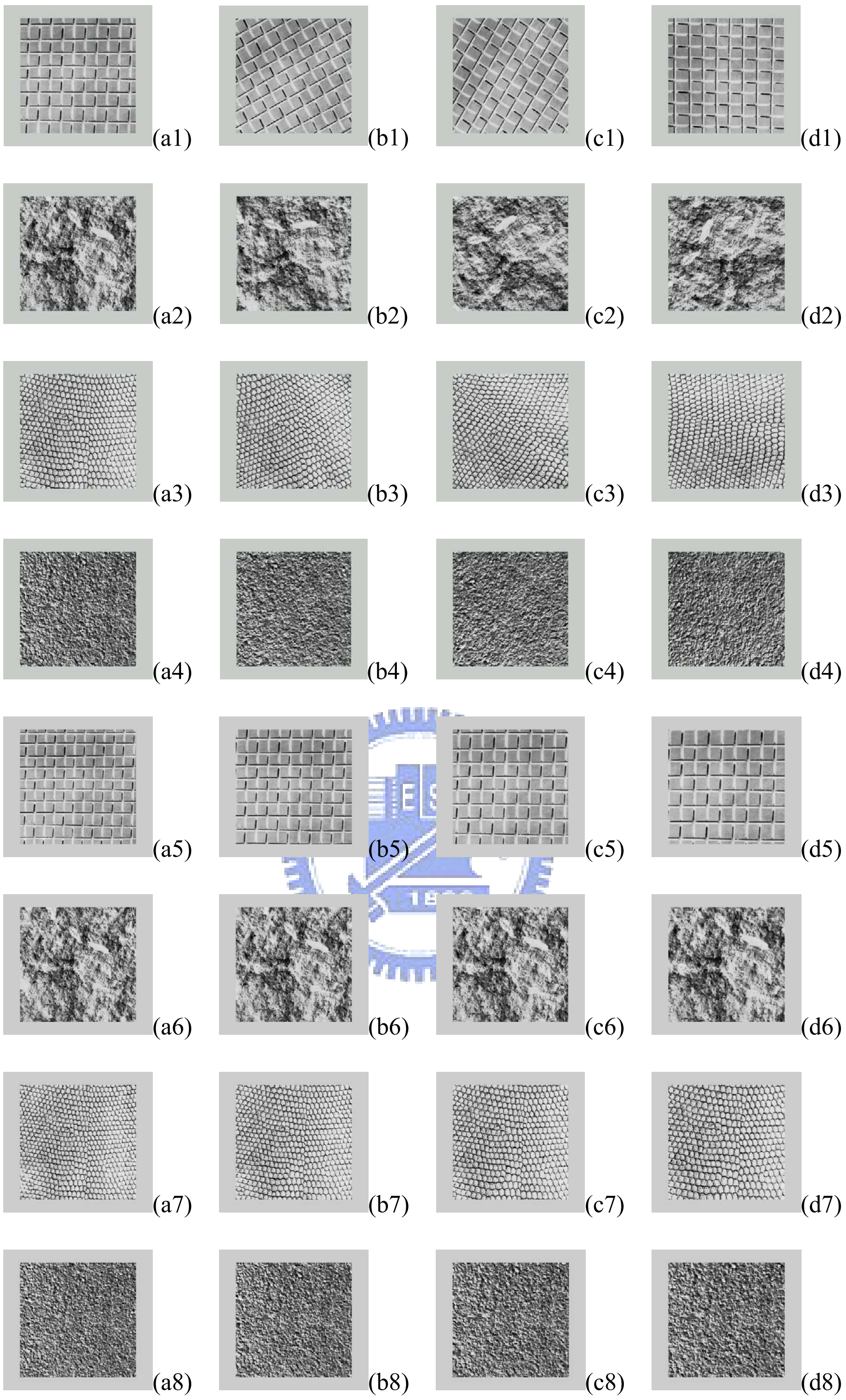


Figure 6-5. Texture example in this experiment. Here we have 32 images. From (a1) to (d4), some of those images is similar to each other expect the rotation effects. From (a5) to (d8), some of those images is similar to each other expect the scale effects.

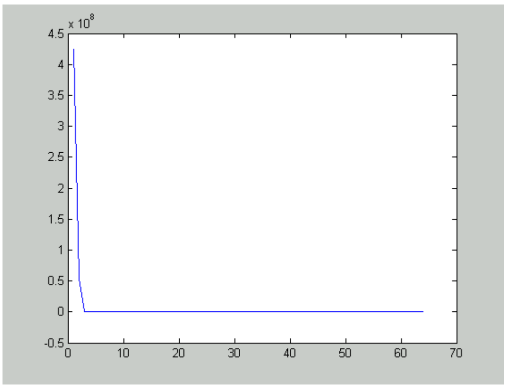
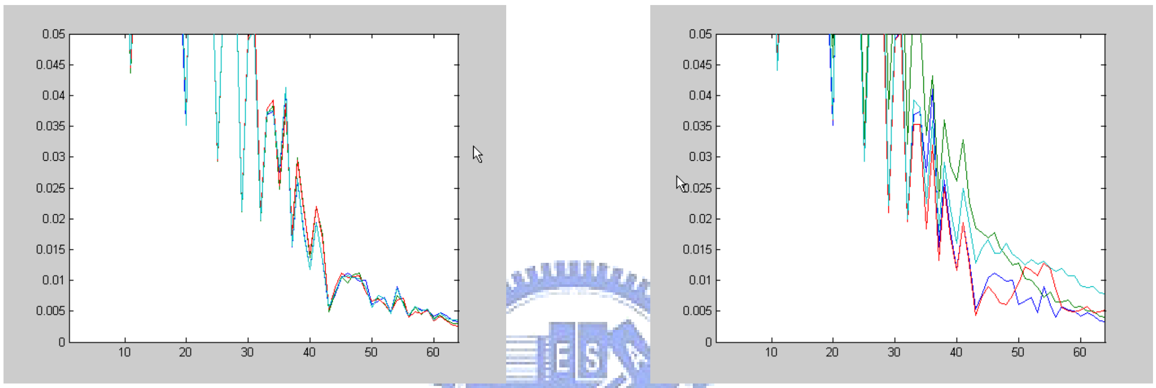


Figure 6-6. The eigenvalues in the proposed rotation-invariant texture classifier.



(a) (b)

Figure 6-7. We plot the feature vectors of texture image in Figure 6-5 as sequences. The feature vectors of Figure 6-5(a1), (b1), (c1), and (d1) have been shown in (a), and the feature vectors of Figure 6-5(a1), (a2), (a3), and (a4) have been shown in (b).

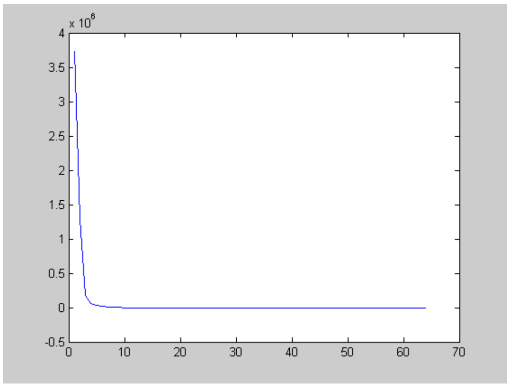
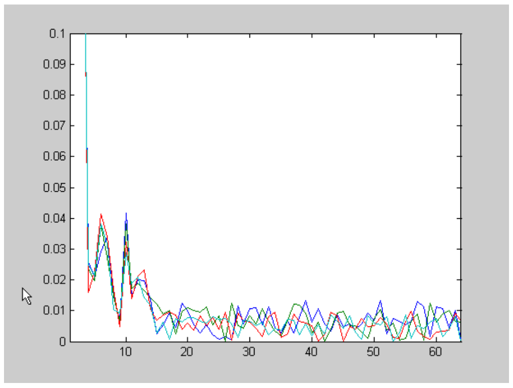


Figure 6-8. The eigenvalues in the proposed Scale-invariant Texture Classifier.


(a)

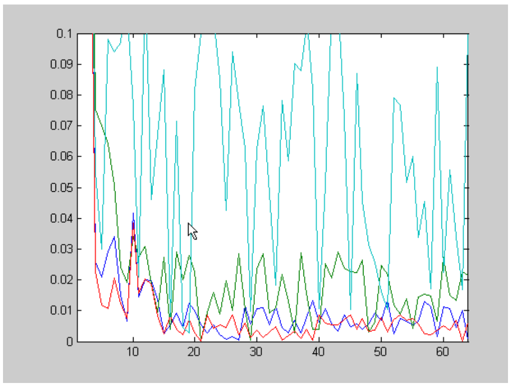

(b)

Figure 6-9. We plot the feature vectors of texture image in Figure 6-5 as sequences. The feature vectors of Figure 6-5(a5), (b5), (c5), and (d5) have been shown in (a), and the feature vectors of Figure 6-5(a5), (a6), (a7), and (a8) have been shown in (b).

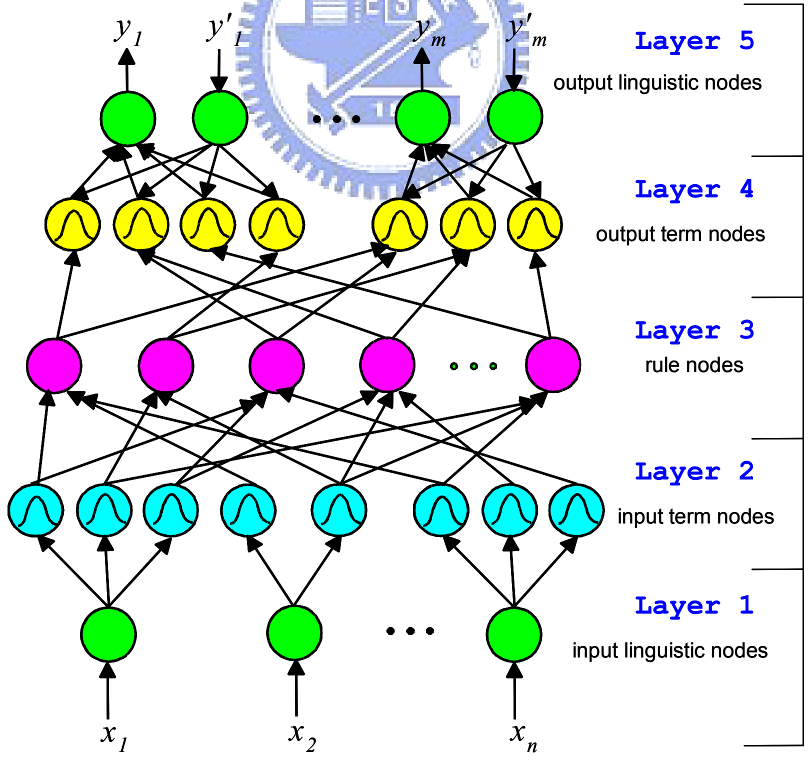


Figure 6-10. The architecture of SOFIN.

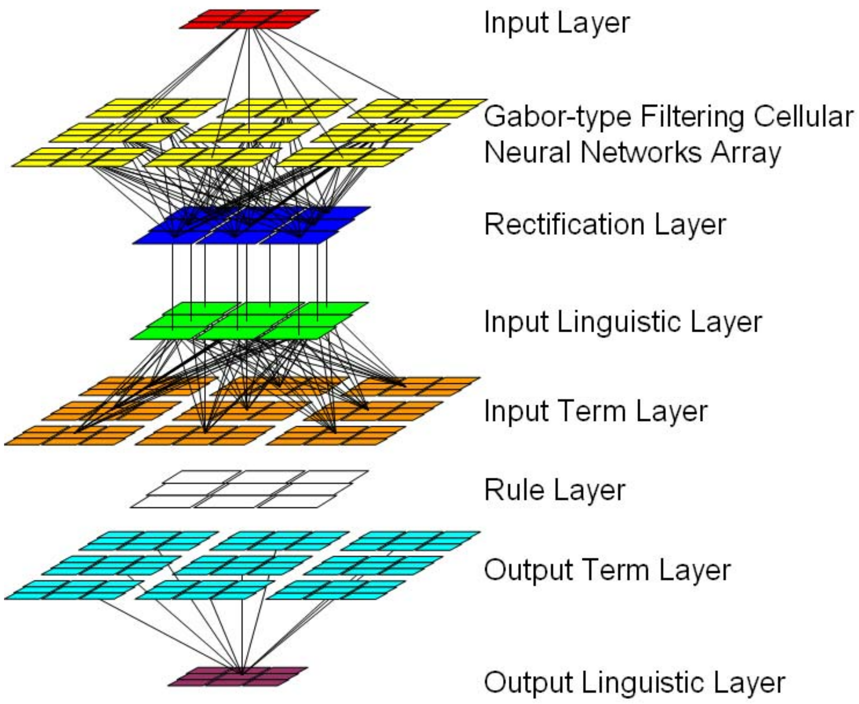


Figure 6-11. The architecture of the algorithm which has been proposed in this thesis. The connection of Rule Layer will be generated after training stage finished.

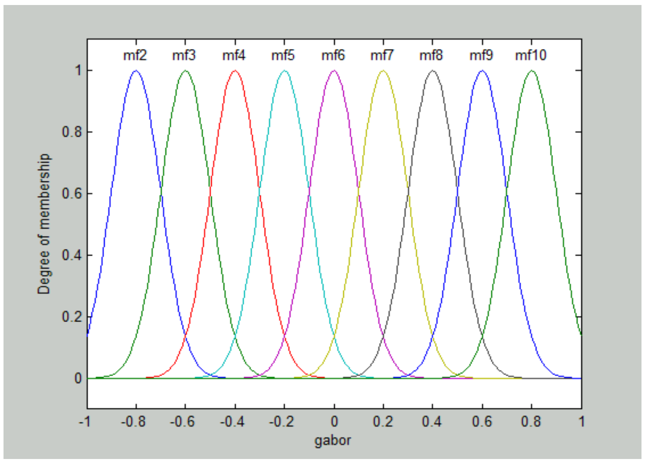
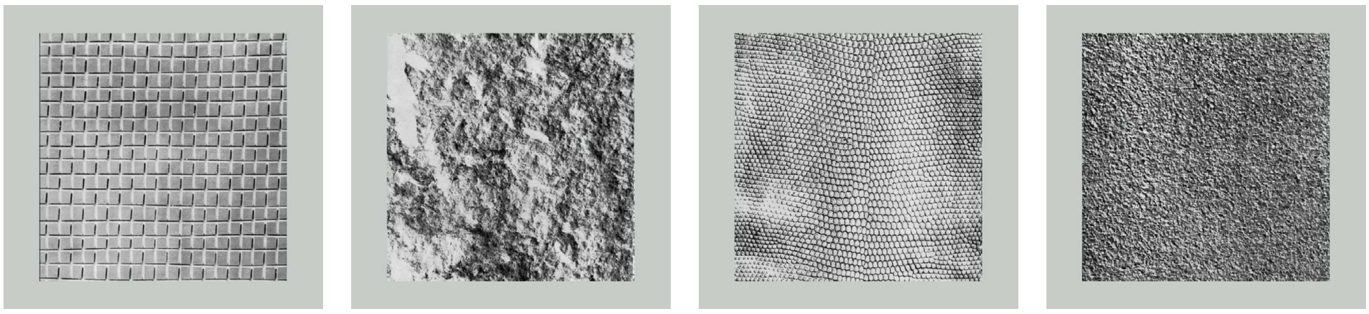
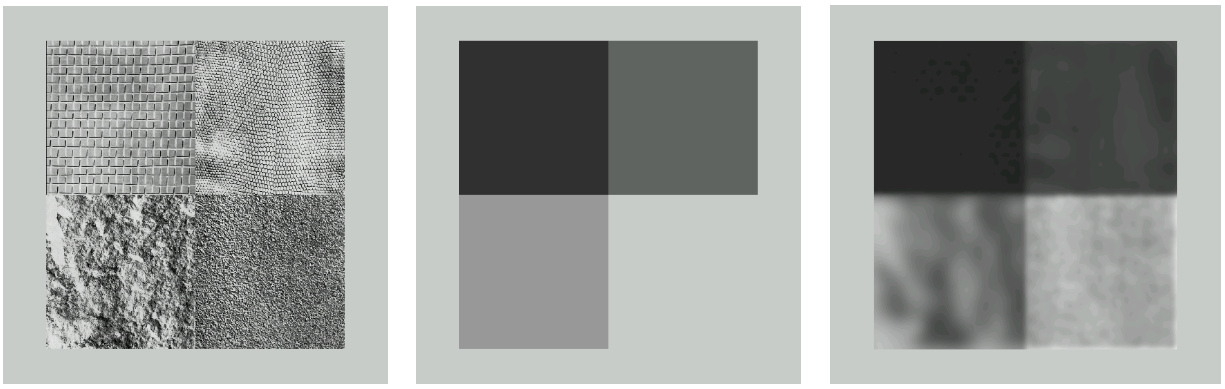


Figure 6-12. The membership function of fuzzification and defuzzification in proposed algorithm



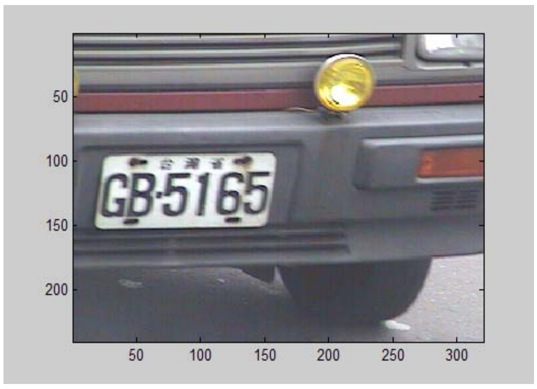
(a) (b) (c) (d)

Figure 6-13. The training patterns. The desired output for (a) is 0.75, for (b) is 0.25, for (c) is -0.25, and for (d) is -0.75.

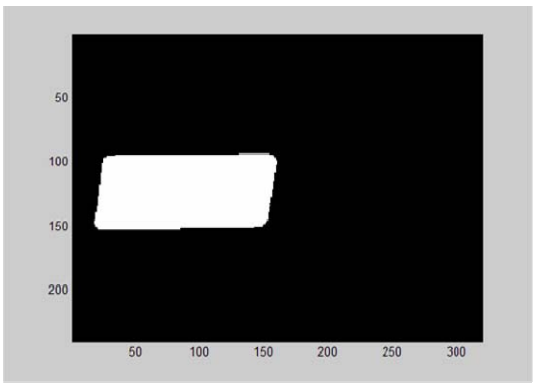


(a) (b) (c)

Figure 6-14. (a) is the testing pattern, (b) is the desired output, and (c) is the output of testing pattern.

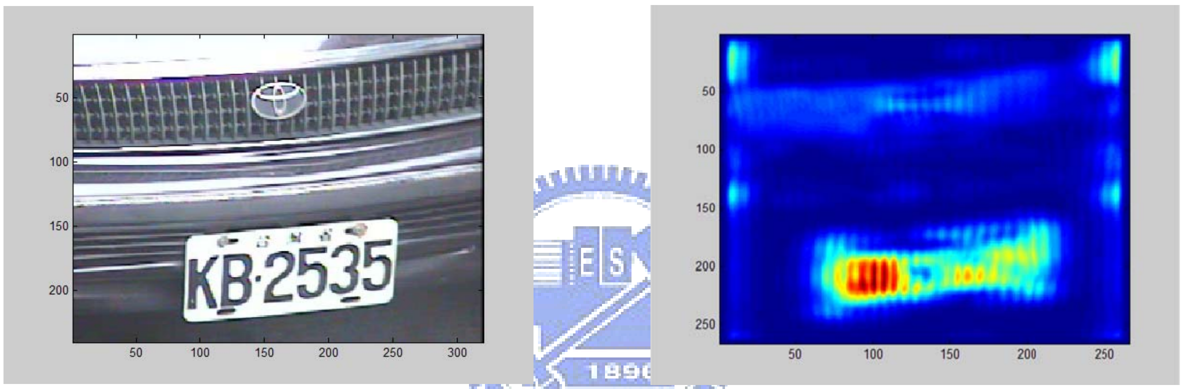


(a)

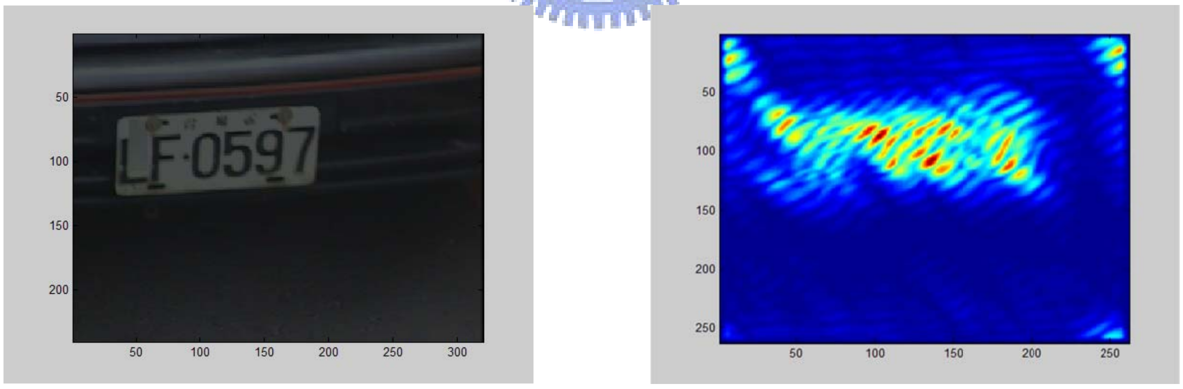


(b)

Figure 6-15. The training pattern of a car plate segmentation.



(a)



(b)

Figure 6-16. The results of a car plate segmentation.

7 CONCLUSIONS AND PERSPECTIVES

This work studied the possibilities for implementing a computer retina and cortex model. Although the details of the retina, cortex, and even the whole of the human vision system remain unknown, based on some results of previous researches on both biology and digital image processing, the fovea model can be roughly realized, and consequently some possible applications can be fulfilled. However, some issues remain open to question. For example, this experiment examines 2 or 3 types of ganglions. In fact, over 20 different structures of ganglions have already been identified. Thus, it is interesting to consider the relationships among these different types of implementations, particularly in cases where the response of a ganglion is related to the time domain. Studying the relationship between the structure of the retina and possible applications for image processing offers one of the most interesting topics for future research.

This study also proposed a simulation of human visual cortex is proposed. It mimics the mechanism of the early stage of the human vision, and experimental results are generally consistent to the human visual sensation. After post processing, the detected boundaries also have adequate accuracy for the other image processing applications such as stereo, and pattern recognition. By implementing the proposed algorithm on the Cellular Neural Networks (CNN), the computational time will greatly decrease. The real-time processing capability is critical in some applications, such as the object tracking.

Although the proposed algorithm is widely tested to detect boundaries of synthetic textures successfully, there are still some problems demanding to be overcome:

1. As same as the other algorithms for textures analysis which is also based on the Gabor filters, there are too many parameters need to be determined. Determining the parameters will much more complex when the synthesized texture patterns increased. Currently, we

are still looking for the solutions for those kinds of problems.

2. For the sake of keeping the structure simple and combining the hybrid-order features easily without the clustering methods, we use same resolution for all of the Gabor filters in the approach.
3. In this approach, we only consider the first and the second order features. According to some research results, there are still some higher-order features that can be utilized. For example, color is one of them. We do believe proposed approach can be extended to color textures by integrating color information.

REFERENCES

[1] D. H. Hubel, *Eye, brain, and vision*. Yew York: Scienceific American, 1988.

[2] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction,and functional architecture in the cat's visual cortex," *J. Physiol. (Lond.),* vol. 160, pp. 106-154, 1962.

[3] J. I. Yellot, "Spectral consequences of photoreceptor sampling in the rhesus retina," *Science,* vol. 221, pp. 382-385, 1983.

[4] S. Shah and M. D. Levine, "Visual information processing in primate cone pathways - part ii: experiments," *IEEE Transactions on Systems, Man, and Sybernetics - Part B: Cybernetics,* vol. 26, pp. 275-289, April 1996.

[5] J. Thiem and G. Hartmann, "Biology-inspired design of digital Gabor filters upon a hexagonal sampling scheme," in *International Conference on Pattern Recognition, 2000. (ICPR'00)* Barcelona, Spain, 2000, pp. 445-448.

[6] D. Bálya, B. Roska, T. Roska, and F. S.Werblin, "A CNN framwork for modeling parallel processing in a mammalian retina," *International Journal of Circuit Theory and Applications (CTA),* vol. 30, pp. 363-393, 2002.

[7] I. Chen, "Visual and visuomotor processes," in *Vision and cognition* Taipei: Yuan-Liou, 1999, pp. 128-158.

[8] D. H. Hubel and T. N. Wiesel, "Sequence regularity and geometry of orientation columns in themonkey striate cortex," *J. Comput. Neurol.,* vol. 158, pp. 267-293, 1974.

[9] L. O. Chua and L. Yang, "Cellular neural networks: applications," *IEEE Transactions on Circuits and Systems,* vol. 35, pp. 1273-1290, Oct. 1988.

[10] L. O. Chua and L. Yang, "Cellular neural networks: theory," *IEEE Transactions on Circuits and Systems,* vol. 35, pp. 1257-1272, Oct. 1988.

[11] K. R. Crounse and L. O. Chua, "Methods for image processing and pattern formation in cellular neural networks," *IEEE Transactions on Circuits and Systems - I,* vol. 42, pp. 583-601, Oct. 1995.

[12] C. H. Huang and C. T. Lin, "Cellular neural networks for hexagonal image processing," in *Cellular Neural Networks and their Applications, 2005. CNNA2005. The 9th International Workshop on*, Hsinchu, Taiwan, 2005, pp. 81-84.

[13] L. Middleton, J. Sivaswamy, and G. Coghill, "The FFT in a hexagonal-image processing framework," in *Image and Vision Computing 2001. IVCNZ 2001. New Zealand Conference on*, Dunedin, New Zealand, 2001, pp. 231-236.

[14] B. E. Shi, "Gabor-type filtering in space and time with cellular neural networks," *IEEE Transactions on Circuits and Systems - I,* vol. 45, pp. 121-132, Feb. 1998.

[15] H. Kobayashi, J. L. White, and A. A. Abidi, "An active resister network for Gaussian filtering of images," *IEEE Journals on Solid-state Circuits,* vol. 26, pp. 738-748, May 1991.

[16] D. Gabor, "Theory of communication," *J. Inst. Elect. Eng.,* vol. 93, pp. 429-457, 1946.

[17] D. P. Peterson and D. Middleton, "Sampling and reconstruction of wave-number-limited

function in N-dimensional Euclidean spaces," *Information and Control,* vol. 5, pp. 279-323, 1962.

[18] A. Rosenfeld and J. L. Pfaltz, "Distance function on digital pictures," *Pattern Recognition,* vol. 1, pp. 33-61, 1968.

[19] M. J. E. Golay, "Hexagonal parallel pattern transformations," *IEEE Transactions on Computing,* vol. C-8, pp. 733-740, Aug. 1969.

[20] I. Her, "A symmetrical coordinate frame on the hexagonal grid for computer graphics and vision," *ASME Journals of Mechanical Design,* vol. 115, pp. 447-449, Sept. 1993.

[21] I. Her and C. T. Yuan, "Re-sampling on a psudo-hexagonal grid," in *Graphics Models Image Processing Computer Vision, Graphics, and Image Processing, International Conference on, 1994 (CVGIP'94)*, 1994, pp. 336-347.

[22] E. Luczak and A. Rosenfeld, "Distance on a hexagonal grid," *IEEE Transactions on Computing,* vol. C-25, pp. 532-533, 1976.

[23] J. Serra, *Image analysis and mathematical morphology*. London: Acadmic, 1982.

[24] R. C. Staunton, "The design of hexagonal sampling structures for image digitization and their use with local operators," *Image and Vision Computing,* vol. 7, pp. 162-166, Aug. 1989.

[25] R. Ulichney, *Digital halftoning* vol. 1: The MIT Press, 1987.

[26] C. A. Wuthrich and P. Stuck, "An algorithmic comparison between square-and hexagonal-based grids," in *Graphical Models and Image Processing Computer Vision, Graphics, and Image Processing, International Conference on, 1991 (CVGIP'91)*, 1991, pp. 324-329.

[27] S. B. M. Bell, F. C. Holroyd, and D. C. Mason, "A digital grometry for hexagonal pixels," *Image and Vision Computing,* vol. 7, pp. 194-204, Aug 1989.

[28] E. S. Deutsh, "On parallel operations on hexagonal arrays," *IEEE Transactions on Computing,* vol. C-19, pp. 982-983, Oct 1970.

[29] R. M. Mersereau, "The processing of hexagonally sampled two-dimensional signals," *Proceedings of IEEE,* vol. PROC-67, pp. 930-949, Jan. 1979.

[30] A. G. Radványi, "On the rectangular grid representation of general CNN networks," in *Cellular Neural Networks and their Applications, IEEE International Workshop on, 1992 (CNNA'92)*, 1992, pp. 387-393.

[31] G. Seiler and J. A. Nossek, "Symmetry properties of cellular neural networks on square and hexagonal grids," in *Cellular Neural Networks and their Applications, 1992. CNNA1992. International Workshop on*, 1992, pp. 258-263.

[32] G. Cauwenberghs and J. Waskiewicz, "Focal-plane analog VLSI cellular implementation of the boundary contour system," *IEEE Transactions on Circuits and Systems - I,* vol. 46, Feb. 1999.

[33] A. L. Nel, "Hexagonal image processing," in *Communications and Signal Processing, Southern African Conference on, 1989 (COMSIG'89)*, 1989, pp. 109-113.

[34] T. Huang, "Two-dimensional nonrecursive filter design," in *Applied Physics* New York: Springer-Verlag, 1981.

[35] W. Li and A. Fettweis, "Interpolation filters for 2-D hexagonally sampled signals," *International Journals on Circuits Theories and their Applications,* vol. C-18, pp. 733-740, 1997.

[36] D. V. Dille, T. Blu, M. Unser, W. Philips, I. Lemahieu, and R. V. Walle, "Hex-splines: a novel spline family for hexagonal lattices," *IEEE Transactions on Image Processing,* vol. 13, June 2005.

[37] R. M. Mersereau and D. E. Dudgeon, *Multidimensional digital signal processing*. New Jersey: Prentice-Hall, Inc., 1984.

[38] L. Middleton and J. Sivaswamy, "A framework for partical hexagonal-image processing," *Journal on Electronic Imaging,* 2001.

[39] R. Mersereau and T. Speake, "A unified treatment of Cooley-Turkey algorithms for the evaluation of the multi-dimensional DFT," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* vol. 29, pp. 1011-1018, 2005.

[40] I. Her, "Geometric transformations on hexagonal grid," *IEEE Transactions on Image Processing,* vol. 4, pp. 1213-1222, Sept. 1995.

[41] Z. Rahman, D. J. Jobson, and G. A. Woodell, "Retinex processing for automatic image processing," NASA 1986.

[42] J. Thumblin and G. Turk, "Tone reproduction for computer generated images," *IEEE Computer Graphics and Applications,* vol. 13, pp. 42-48, 1993.

[43] E. H. Land and J. L. Huertas, "Lightness theory," *J. Opt. Soc.,* vol. 62, 1971.

[44] A. Zarándy, L. Orzo, E. Grawes, and F. Werblin, "CNN based early vision models for color vision and visual illusions," *IEEE Transactions on Circuits and Systems - I: Special Issue on Bio-inspired Processors and Cellular Neural Networks,* vol. 46, pp. 229-238, 1999.

[45] B. K. P. Horn, "Determining lightness from an image," *Computer Graphics and Image Processing,* pp. 277-299, 1974.

[46] B. Roska and F. S. Werblin, "Vertical interactions accross ten paralel, stacked representations in the mammalian retina," *Nature,* vol. 410, pp. 583-587, 2001.

[47] S. Shah and M. D. Levine, "Visual information processing in primate cone pathways - part i: a model," *IEEE Transactions on Systems, Man, and Sybernetics - Part B: Cybernetics,* vol. 26, pp. 259-274, April 1996.

[48] A. K. Jain, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognition,* vol. 24, pp. 1167-1186, 1991.

[49] A. Jacobs and F. S. Werblin, "Spatiotemporal patterns at the retinal output," *Journal of Neurophysiology,* vol. 80, pp. 447-451, 1998.

[50] H. Steklis and J. Erwin, "The primate retina," *Comparative Primate Biology, Neuroscience,* vol. 4, pp. 203-278, 1988.

[51] S. Shah and M. D. Levine, "The primate retina: a biological summary," in *Center for*

Intelligent Machines. vol. Ph.D. Montreal: McGill University, 1992.

[52] H. Kotera, Y. Yamada, and K. Shimo, "Sharpness improvement adaptive to edge strength of color images," in *Color Imaging Conference 2000*, Scottsdale, ZA, USA, 2000, pp. 149-154.

[53] R. Gershon, A. D. Jepson, and T. K. Tsotsos, "From {R, G, B} to surface reflectance: computing color constant descriptors in images," in *Artificial Intelligence, International Joint Conference on*, 1987.

[54] G. Buchsbum, "A spatial processor model for object colour perception," *Journal of the Franklin Institute,* vol. 310, pp. 337-350, 1980.

[55] H. Tamura, S. Mori, and Y. Yamawaki, "Textural features corresponding to visual perception," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-8, pp. 460-473, 1978.

[56] J. Sklansky, "Image segmentation and feature extraction," *IEEE Transactions on Systems,Man, and Cybernetics,* vol. SMC-8, pp. 237-247, 1978.

[57] J. K. Hawkins, "Picture Processing and Psychopictorics," in *Textural Properties for Pattern Recognition*, B. Lipkin and A. Rosenfeld, Eds. New York: Academic Press, 1969.

[58] A. C. Bovik, "Analysis of multichannel narrow-band fiters for image texture segmentation," *IEEE Trans. Signal Process,* vol. 39, pp. 2025-2043, 1990.

[59] D. Dunn and W. E. Higgins, "Optimal Gabor filter for texture segmentation," *IEEE trans. Image processing,* vol. 4, pp. 947-964, Jul. 1995.

[60] T. N. Tan, "Texture edge detection by modelling visual cortical channels," *Pattern Recognition,* vol. 28, pp. 1283-1298, July 1995.

[61] P. Brodatz, *Textures: a photographic album for artists and designers*. New York: Dover Publications, 1966.

[62] *ALADDIN CNN Software Library (Templates and Algorithms)*. Budapest: Analogic Computers Ltd., 2000.

[63] J. Daugman, "Uncertainty relation for resolution in space, spatial frequency, and orientation optimised by two-dimensional visual cortical filters," *Journal of the Optical Society of America,* vol. 2, pp. 1160-1169, 1985.

[64] I. Chen, "Texture perception: a linear system approach," University of California, Berkeley, 1994.

[65] *ALADDIN CNN Software Library for ACE4K Chip (Templates and Algorithms)*. Budapest: Analogic Computers Ltd., 2000.

[66] T. P. Weldon and W. E. Higgins, "Designing multiple Gabor filters for multi-texture image segmentation," *Opt. Eng.,* vol. 35, pp. 2852-2863, Oct. 1996.

[67] A. Materka and M. Strzelecki, "Texture analysis methods - a review," University of Lodz, Institute of Electronics 1998.

[68] C. T. Lin, *Neural Fuzzy Systems with Structure and Parameter Learning*. New York: World Scientific, 1994.