LEARNING TO SEGMENT TEXTURE IN 2D VS. 3D : A COMPARATIVE STUDY

A Thesis

by

SE JONG OH

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2004

Major Subject: Computer Science

LEARNING TO SEGMENT TEXTURE IN 2D VS. 3D : A COMPARATIVE STUDY

A Thesis

by

SE JONG OH

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

Yoonsuck Choe (Chair of Committee)	Ricardo Gutierrez-Osuna (Member)
Takashi Yamauchi (Member)	Valerie Taylor (Head of Department)

August 2004

Major Subject: Computer Science

ABSTRACT

Learning to Segment Texture in 2D vs. 3D : A Comparative Study. (August 2004)

Se Jong Oh, B.S., Korean Air Force Academy

Chair of Advisory Committee: Dr. Yoonsuck Choe

Texture boundary detection (or segmentation) is an important capability of the human visual system. Usually, texture segmentation is viewed as a 2D problem, as the definition of the problem itself assumes a 2D substrate. However, an interesting hypothesis emerges when we ask a question regarding the *nature* of textures: *What are textures, and why did the ability to discriminate texture evolve or develop?* A possible answer to this question is that textures naturally define physically distinct surfaces or objects, thus, we can hypothesize that 2D texture segmentation may be an outgrowth of the ability to discriminate surfaces in 3D. In this thesis, I investigated the relative difficulty of learning to segment textures in 2D vs. 3D configurations. It turns out that learning is faster and more accurate in 3D, very much in line with what was expected. Furthermore, I have shown that the learned ability to segment texture in 3D transfers well into 2D texture segmentation, but not the other way around, bolstering the initial hypothesis, and providing an alternative approach to the texture segmentation problem.

To My loving wife, Jinhee Jun, and our daughters, Suzin and Rebecca

Page

ACKNOWLEDGMENTS

I would like to express my special thanks my advisor, Dr. Yoonsuck Choe, for his brilliant supervision, constant scientific teaching, encouragement and guidance right from the conceptual stage to the completion of this thesis. His enthusiasm and scientific insights have been the leading force behind my research, and his method of guiding by providing examples and attention to detail have been really motivating for my research. I have learned a lot about research from him through many interesting discussions. I really appreciate his efforts that have been mirrored in every page of this thesis.

I would also like to express my gratitude to my committee members, Dr. Ricardo Gutierrez-Osuna and Dr. Takashi Yamauchi, for their valuable and insightful comments on the draft and during the presentation of my ideas.

I would like to acknowledge the contributions of my fellow research group members, Yingwei Yu and Heejin Lim, who provided useful feedback. Thanks are also due to a former research group member, S. Kumar Bhamidipati, for contributing to some basic parts of the thesis.

Most of all, I want to thank my beloved family for their constant encouragement and for sacrificing a lot for me throughout my graduate studies.

Especially, I would like to thank the Republic of Korea Air Force for generously providing financial support for my research. This research was supported in part by the Texas Higher Education Coordinating Board ATP program grant #000512-0217-2001.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

FIGURE Page

FIGURE Page

CHAPTER I

INTRODUCTION

Detection of a tiger in the bush is a perceptual task that carries a life or death consequence for preys trying to survive in the jungle [1]. Here, figure-ground separation becomes an important perceptual skill. Figure-ground separation is based on many different cues such as luminance, color, and texture. In case of the tiger in the jungle, texture plays a critical role. Texture segmentation divides a scene into regions of uniform textures, each of which corresponds to the surface of a distinct object or object pattern. The visual system can then recognize what each object is. What are the visual processes that enable perceptual systems to separate figure from ground using texture cues? This intriguing question has led many researchers in vision to investigate the mechanisms of texture perception.

There have been numerous studies targeted at understanding the neural mechanisms of human visual system underlying texture segmentation. Based on the early works of Beck [2][3] and Julesz [4] many studies have investigated the features that characterize two abutting textures such that they can be pre-attentively separated from each other [5][3][6]. In these studies, orientation and size of bars or blobs are considered as key features in the segmentation of texture regions [7]. These studies focused on labeling the texture regions, so their approaches can be called segmentation based on *classification*. On the other hand, psychophysical and physiological studies have shown that human texture processing can also be based on the detection of *boundaries* between bordering, heterogeneous textures [7], which can be referred to *segmentation without classification*. Common to all of these texture segmentation and

---

The journal model is *IEEE Transactions on Neural Networks*.

texture boundary detection studies is that the problem of texture is viewed as a 2D problem, where textures are defined on 2D surfaces. However, an additional approach emerges when we ask a question regarding the *nature* of texture: *What are textures, and why did the ability to discriminate texture evolve or develop?* In this thesis, the proposed answer to the question is that textures naturally define physically distinct surfaces, thus, it is hypothesized that 2D texture processing ability may have been an outgrowth of the ability to discriminate textured surfaces in 3D.

This thesis investigates the relative difficulty of learning to segment textures in 2D vs. 3D configurations and test whether the learned ability of 3D texture processing can easily be transferred to 2D texture tasks. The rest of this chapter provides a review on current texture segmentation and boundary detection studies in 2D and ideas on 3D surface representation. Next, the motivation and research problems will be presented, followed by the main approach taken in this thesis and a brief overview of the organization of the remaining chapters.

A. Problem Overview

For a better understanding of the problem addressed in this thesis, a brief review of current studies in texture segmentation is first provided. Next, the concept of surface representation in vision, which provided motivation for this research, is introduced.

1. Texture Segmentation

Julesz [4] and Beck [2][3] conducted psychological experiments investigating the visual features that enable humans to discriminate one texture from another. These studies suggested that texture segmentation occurs based on the distribution of simple properties of "texture elements", such as brightness, color, size, and the orientation

of contours, or other elemental descriptors [8]. Julesz further proposed the texton theory, in which textures are discriminated if they differ in the density of simple, local textural features called textons [9].

Texture segmentation models based on these observations naturally lead to a feature-based theory, in which segmentation occurs when feature differences (such as difference in orientation) exist. According to Chubb et al. [10], any first-order (quasi-linear) mechanism cannot detect the boundary that emerges between two textures of equal mean luminance but composed of differently oriented micropatterns. So, computation of texture boundary must take the responses from the first-order channels as its input (first filtering process) and then apply some strong nonlinearity (e.g., rectification) to these channel outputs to sense the texture boundary appropriately. Several texture segmentation models such as Malik's [11] and Bergen's [12] adopted this approach. In this prototypical second-order model, rectification-type nonlinearity that separates two linear-filtering stages in complex channels was used [13] [12] [11]. Whereas studies based on the feature-based theory viewed texture processing with classification of texture features, psychophysical and physiological studies have shown that human texture processing may be based on the detection of texture boundaries between heterogeneous textures using contextual influences via intra-cortical interactions in the primary visual cortex [14][15][7]. These latter studies focused on neural correlates of boundary detection at relatively early stages of cortical processing [16].

However, some have proposed that bottom up approach alone cannot fully explain texture segmentation. For example, Theilscher and Neuman [7] proposed a novel computational model of texture boundary detection. Their neural model for texture boundary detection consisted of bi-directionally linked cortical areas V1, V2 and V4, and it integrated the data obtained from a variety of methods and experimental results. Their model was built upon the two key hypotheses that (i) texture

segmentation is based on boundary detection and that (ii) texture border detection is mainly a function of higher visual cortical areas such as V4, which were motivated by the principle of recurrent interaction for response integration and cortical prediction [17]. Their model is different from the other bottom up approaches such as Malik and Perona's [11] and Li's [14][15] models, which are based on feedforward projections of signals and the intra-cortical interactions within V1.

Many approaches presented above are based on the response of the simple cells in primary visual cortex, which are estimated by linear filters. The output energy of a filter bank that consists of filters with different orientation and frequency channels is computed. It is further processed for output combinations from different channels, or for input into a decision stage [12][18][19]. Inspired by these theoretical foundations based on physiological and psychological experiments, many computational/engineering models of classification and segmentation of texture in digital image have been proposed. Main features used in these models can be categorized into five classes: statistical [19], geometrical, structural [20], model-based [21], and signal processing features [1]. All of the models reviewed above assume that texture segmentation and boundary detection to be a 2D problem. This means that texture perception is understood in terms of 2D features and filtering, so the performance is determined by differences in the output of neurons in low-level visual processing.

2. Surface Representation

We live in a 3D world, which means that we cannot expect to see only one surface at a time within any given scene of the external world. Often we see multiple surfaces in local regions of the visual scene, with closer objects at least partially covering those behind. Thus many surface regions have no counterpart projected on the retina [22]. But we do not feel much loss of information even when part of a surface is not visible

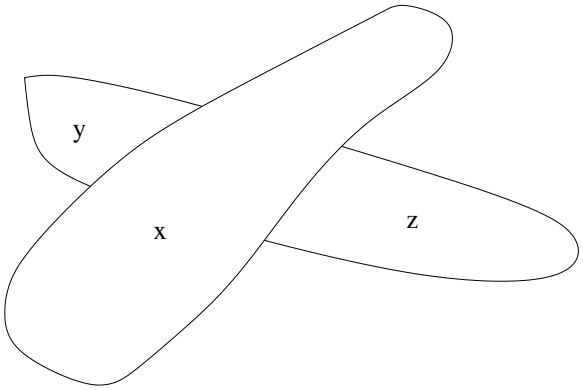
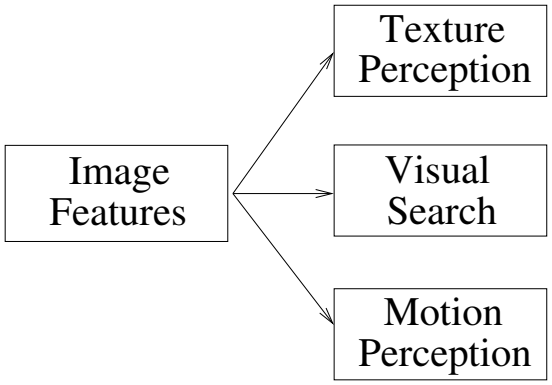


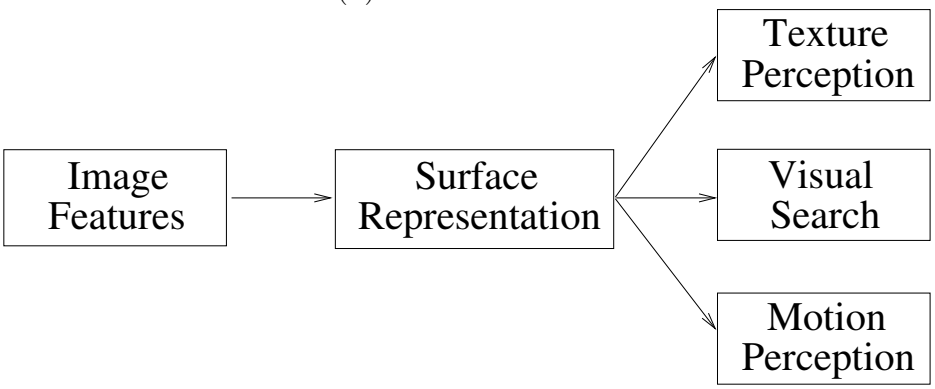
Fig. 1. Three two-dimensional areas, labeled x, y and z. Even though they do not look like recognizable objects, x is perceived to be in front of a partially occluded regions y and z, and y and z are perceived to be part of a same region. Adapted from [22].

by occlusion, that is we do not consider invisible surface regions as not existing. Figure 1 shows an example, in which we perceive depth and infer that surface x is covering surface y and z. Even though these patches are not recognizable as familiar objects, we can still infer that y and z constitute one connected surface. Nakayama et al. [22] view that such inferences are embedded in the visual system and can occur at early stages, independent of our knowledge about familiar objects.

Other similar observations as the above led them to propose that surface representations form a critical intermediate stage in vision, poised between the earliest processing stage of image information and later stages such as object recognition. This view of intermediate visual processing is shown in figure 2 [22]. In this view, the visual surface representation stage is considered to serve a critical link between lower-level vision and higher-level vision. Here, an indispensable part of visual perception is the encoding of surfaces, whereas the output signal of lower-level vision is projected directly to the higher-level in the traditional view. In line with this view, He et al. [23] showed experimental results suggesting that in rapid texture segmentation,



(a) Traditional view



(b) An alternative view

Fig. 2. Two views of intermediate visual processing. (a) Texture perception, visual search and motion perception depend on feature processing in early cortical areas. (b) Surface representation must precede other intermediate visual tasks [22]. Adapted from [22].

the visual system cannot ignore information regarding surface layout. However, the proposed surface representation stage is mainly based on psychological experiments using stereograms, without physiological considerations.

This idea about surface representation stage presented above provided me with the insight that texture boundary detection in 3D may go through quite different processing and the 3D texture detection ability may have contributed to the 2D counterpart.

3. Motivation

From the review of research regarding texture segmentation and boundary detection, we can observe that textures are normally defined on 2D surfaces rather than in 3D configurations and thus texture boundary detection is basically seen as a 2D problem. However, an interesting hypothesis arises when we ask an important question regarding the *nature* of textures and consider the proposed view above about the surface representation stage: *What are textures, and why did the ability to discriminate textures evolve or develop?* One possible answer to this question is that, in line with the above, texture is that which defines physically distinct surfaces, belonging to different objects or object parts, and that texture segmentation function may have evolved out of the necessity to distinguish different surfaces. Human visual experience with textures in life can be, therefore, in most cases, to use them as cues for surface perception, depth perception, and 3D structure perception. In fact, as reviewed earlier, psychological experiments by Nakayama and He [23][22] showed that the visual system cannot ignore information regarding surface layout in texture discrimination and proposed that surface representation must actually precede other perceptual representations such as texture.

From the discussion above, we can reasonably infer that texture processing may be closely related to surface discrimination. Surface discrimination is fundamentally a 3D task, and 3D cues such as stereopsis and motion provide unambiguous information about the surface. Thus, it can also be hypothesized that 3D surface perception could have contributed in the formation of early texture segmentation processing capabilities in human vision.

In this thesis, through computational experiments using artificial neural networks, the relative difficulty of learning to detect texture boundaries in 2D vs. 3D

arrangement of textures will be investigated. I will also study whether the learned ability to segment texture in 3D can transfer into 2D, to test the above hypothesis. Thus, the relevant research problem to be addressed in this thesis can be summarized as follows :

1. Relative difficulty of learning to detect boundaries in 3D vs. 2D arrangement of textures in terms of speed and accuracy.

2. The ability to transfer 3D texture boundary detection skills into its 2D counterpart.

B. Approach

In this thesis, I will answer the specific problems listed above through computational experiments with textures defined in 3D and 2D. The input texture will be preprocessed according to the properties of receptive field responses in the early visual system. Since several studies using single- or multi-unit recordings in the primary visual cortex of primates provided evidence that V1 is involved in the processing of texture [7], I will use a model of simple cells and complex cells, which is a filter bank that consists of filters with different orientation and frequency channels. These filters typically include oriented Gabor filters [24] and differences of Gaussian (DOG) filters [7]. However, in this thesis, for simplicity, I will only use oriented Gabor filters to estimate the spatial feature responses of receptive fields.

The texture segmentation will be accomplished without explicit classification, which means that the learning process will focus on detecting the boundary, and not on the classification of each texture based on image features.

For the learning part, a standard backpropagation and mutilayer perceptron (MLP) will be used. Even though there are several limitations of the backpropagation

algorithm such as local minima, many researches have successively used it in texture processing [25] [26]. To overcome some of the limitations of backpropagation, I will apply several standard improvements such as momentum and adaptive learning rate. Two MLPs will be trained with different training sets, the 2D training set and the 3D training set, and the speed of learning and the accuracy of boundary detection over prepared texture images will be compared. I will also test the performance of the network trained with 3D inputs on 2D texture boundary detection tasks.

C. Outline of the Thesis

This thesis is organized as follows. In Chapter II, I will first present the property of the early visual system and a brief overview of the backpropagation learning algorithm. In Chapter III, details about the method I used for my experiments will be described, which includes details about input preparation and preprocessing, training of the MLPs, and testing of the trained MLPs. Next, the experimental results and their analysis will be presented in Chapter III. The speed and accuracy of learning in each network will be compared and the performance will be statistically analyzed. Finally, discussion about the results and the relation with other studies will be presented in Chapter IV, followed by future work and conclusion.

CHAPTER II

BACKGROUND

In this chapter, I will summarize the properties of the early visual system and the Gabor filter, which the preprocessing stage of the network input is mainly based upon. Next, a brief overview of the backpropagation learning algorithm, which is the foundation of the computational experiments, will be provided.

A. The Early Visual System

Various psychophysical and electrophysiological experiments and advances in brain imaging technology have provided us with a lot of knowledge about the visual system of the primates and mammals [27]. Even though there are obvious differences between the human visual system and the primate visual system, it has also been widely accepted that the general structure is quite similar [28]. In this section, I will briefly provide a general overview of the early visual system, the properties of its neurons and a computational model of the simple cells.

1. Organization of the Visual System

Figure 3 shows an illustration of the early stages of the visual pathway in primates (adapted from [29]). The signals generated by photoreceptors in the retina, which is a layer of cells at the back of the eye, are transmitted to the lateral geniculate nucleus (LGN) in the thalamus through the optic nerve, and are further sent to the primary visual cortex (V1). V1 is considered to be the first location where the visual information is processed by the cerebral cortex. The signals are processed in V1 and then sent to other locations in the extra-striate cortex such as V2, V3, V4, and V5.

The central visual pathway can be divided into two pathways, which are the

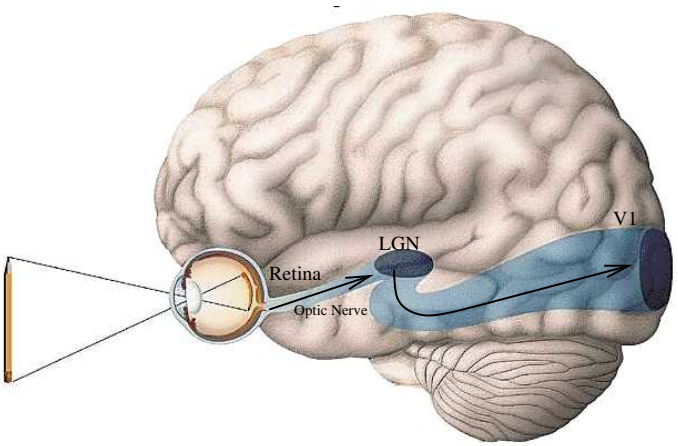


Fig. 3. Central visual pathway in primates. The major pathway that visual information goes through from the eye to the primary visual cortex is shown. Signals are produced by rod and cone receptors in the retina and are then transferred to a major relay station, the LGN (lateral geniculate nucleus) via the optic nerve. Signals then travel to selected areas of the primary visual cortex (V1). Signals are sent to higher areas of cortex from there on. Adapted from [29].

m-pathway and the *p-pathway*. The m-pathway is characterized by the following properties [30]:

- poor spatial resolution and good temporal resolution,
- m ganglion cells,
- magnocellular layers (LGN),
- an upper subdivision of layer 4 in V1,
- V2 (thick stripes),
- MT ('the motion area'), and
- parietal lobes ('where').

On the other hand, the p-pathway has the following properties [30]:

- poor temporal resolution and good spatial resolution,
- p ganglion cells,
- parvocellular layers (LGN),
- a lower subdivision of layer 4 in V1,
- V2 (thin stripes and interstripes),
- V4 ('the color area'), and
- inferotemporal lobes ('what').

Physiological studies have shown that neurons in areas along the p-pathway respond selectively to visual features with respect to object identification such as color, texture, shape and binocular disparity [31]. On the other hand, neurons in areas along the m-pathway respond selectively to spatial aspects of the stimuli, such as direction of motion, speed of motion, and tracking eye movements [32]. Many studies have supported the distinction between these two pathways [33]. However, important inter-connections between the two streams have been shown and the functional impact for such inter-connections is believed to subserve the integration of visual information at higher stages of the processing [34]. This thesis is partly motivated by such an integration of visual information in the m-pathway and the p-pathway.

Now let us look at the response property of a typical neuron in the early visual pathways, which plays an important role in texture processing and the preprocessing stages of the experiments in this thesis. The response of a neuron depends on the pattern of input of a small area of the visual field, called the receptive field (RF). Thus changes in the input stimulus in the receptive field will lead to changes in the firing pattern of the corresponding neuron. The receptive fields in different stages of

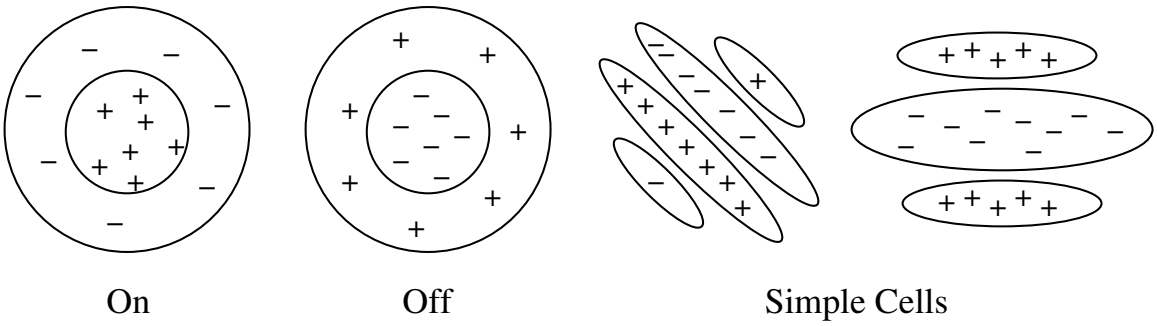


Fig. 4. Typical receptive fields of the neurons in the early visual pathway. Positive signs denote excitation and negative signs denote inhibition. (Left) the RFs of retinal ganglion cells and LGN cells show a center-surround property. (Right) The RFs of V1 neurons show orientation, phase, and frequency selectivity. Adapted from [28].

the visual pathway are known to exhibit different properties. The receptive fields of the retinal ganglion cells and of the LGN show a center-surround property, which is exhibited by on-center/off-surround cells or off-center/on-surround cells. On-center cells respond best to a point of light in a dark field and off-center cells respond best to a dark point in a light field [35] [36] (figure 4). On the other hand, in the primary visual cortex (V1), the receptive fields exhibit orientation, phase, and frequency tuned properties, as illustrated in figure 4 (adapted from [28]).

It is also known that neurons in the visual cortex (as in other cortical areas) show graded response to specific stimuli. Also, nearby locations in the visual field are found to be mapped to nearby neurons in the visual cortex. A consequence of the finding about the receptive fields shown in figure 4 is that the early visual processing can be modeled as a sequence of filter convolutions. The center-surround receptive fields can be modeled as the difference of two Gaussian kernels, a classical model of which is given by the Difference-of-Gaussian (DoG) filter. The orientation selective receptive fields can be modeled by Gabor filters which are products of sinusoidal gratings and Gaussian envelopes [37]. Gabor filters are closely related to the function

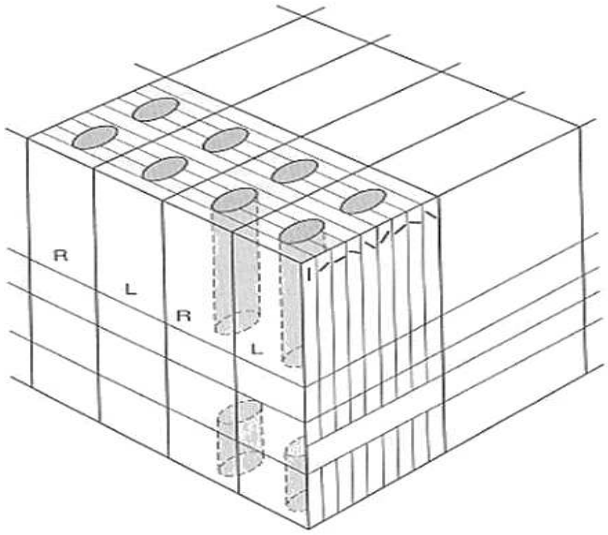


Fig. 5. The hypercolumn organization of the cortex. It is shown with the addition of columns of color-opponent cells (shaded). These areas are called blobs because of their appearance when the cortex is stained. Adapted from [38].

of simple cells in the primary visual cortex of primates. There are alternating columns of cells parallel to the surface of V1, which are driven predominantly by inputs to a single eye. These alternations between left and right eye are referred to the ocular dominance columns. Changes of the preferred orientation from horizontal to vertical also make orientation columns perpendicularly to the surface of V1. These two types of columns constitute *hypercolumn* as shown in figure 5 [38]. The Gabor response matrix, the orientation response matrix, and the frequency matrix in the following chapter is based on this concept of hypercolumn.

2. Model of Simple Cell in V1

Since simple cells play a critical role in texture boundary detection, a computational model of this type of cell is briefly introduced. This section is largely based on [39] and [40]. The response r of a simple cell which is characterized by a receptive field

function $g(x, y)$ to a luminance distribution image $f(x, y)$ is computed as follows [41]:

$$r = \int \int f(x - x', y - y')g(x, y)dx'dy', \tag{2.1}$$

where x' and y' denote a rectangular window. This operation is referred to as convolution. The 2D Gabor functions to model the spatial summation properties of simple cells are given as follows [24] [42] [43]:

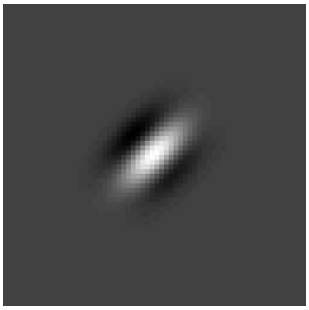
$$G_{\theta,\phi,\sigma,\omega}(x, y) = \exp^{-\frac{x'^2+y'^2}{2\sigma^2}} \cos(2\pi\omega x' + \phi), \tag{2.2}$$

where θ is the orientation, ϕ the phase, σ the standard deviation (width) of the envelope, ω the spatial frequency, (x, y) represents the pixel location, and x' and y' are defined as:

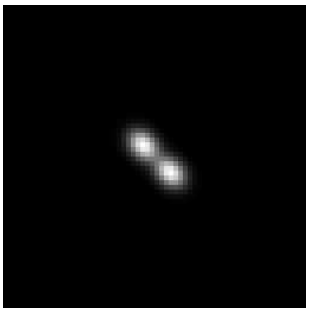
$$x' = x\cos(\theta) + y\sin(\theta), \tag{2.3}$$

$$y' = -x\sin(\theta) + y\cos(\theta), \tag{2.4}$$

where the parameter $\theta \in [0, \pi)$ specifies the orientation of the normal to the parallel excitatory and inhibitory stripe zones which can be observed in the receptive fields of simple cells. The standard deviation σ of the Gaussian factor determines the linear size of the receptive field. The parameter ω, which is the frequency of the cosine factor $\cos(2\pi\omega x' + \phi)$, determines the preferred spatial frequency of the receptive field function $G_{\theta,\phi,\sigma,\omega}(x, y)$. Finally, the parameter ϕ, which is a phase offset in the argument of the harmonic factor $\cos(2\pi\omega x' + \phi)$, determines the symmetry of the function $G_{\theta,\phi,\sigma,\omega}(x, y)$: for $\phi = 0$ and $\phi = \pi$ it is symmetric with respect to the center of the receptive field; for $\phi = -\frac{\pi}{2}$ and $\phi = \frac{\pi}{2}$ it is asymmetric. An intensity map of a receptive field function with a particular position, size, orientation, and symmetry is shown in figure 6(a). Figure 6(b) shows the corresponding spatial frequency response.



(a)



(b)

Fig. 6. 2D Gabor function in (a) spatial ($\theta = \frac{\pi}{4}$) and (b) in spatial frequency domain.

Although such a model is quite simplistic, it has been found to be quite effective as a model for preprocessing of visual input to study visual responses [44].

B. Backpropagation Algorithm and Its Improvements

Artificial neural networks are composed of a simple element, called neuron, operating in parallel and these elements are inspired by the biological nervous systems. A neural network can be trained to perform a particular functions by adjusting the values of the connections between the elements. Out of many learning methods, I used neural networks because of the following useful properties (adapted from [45]) :

- Nonlinearity. It is a highly important property due to the inherent nonlinearity of the underlying input signals, which are texture feature signals in the current

case.

- Input-output mapping. The experiments in this thesis, which require supervised learning, involves modification of the synaptic weights of a neural network by applying a set of labeled training samples and corresponding responses.
- Adaptivity. The natural capability of a neural network for pattern classification make it a useful tool in the experiments done here, which can be considered as adaptive pattern recognition.

In this section, I will briefly present the general structure of artificial neural networks and a summary of the backpropagation algorithm. This section is closely follows [46] and [45].

1. Neuron

A neuron is an information processing unit that is fundamental to the operation of a neural network. Figure 7 shows the model of a neuron, which forms the basis of designing artificial neural networks. There are three basic elements of the neuronal model;

- A set of synapse or connecting links, each of which is characterized by a weight. The signal input x_m is multiplied by the synaptic weight w_{km}.
- An adder for summing up the input signals, weighted by the respective synapses of the neuron.
- An activation function f for limiting the amplitude of the output of a neuron.

The model of a neuron in figure 7 also includes bias (b_k), which has the effect of increasing or decreasing the net input of the activation function depending on whether it is positive or negative, respectively.

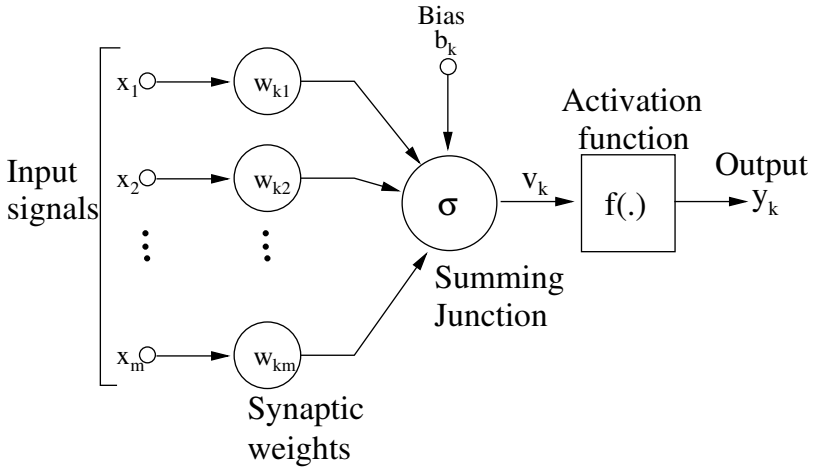


Fig. 7. Nonlinear model of a neuron. x_i denotes input signals to the neuron, w_{kj} denotes synaptic weights, b_k denotes the bias, $f(\cdot)$ denotes the activation function and y_k denotes output of the neuron. Adapted from [46].

A neuron k can be described by the following pair of equations:

$$v_k = \sum_{j=1}^{m} w_{kj} x_j + b_k \tag{2.5}$$

$$y_k = f(v_k), \tag{2.6}$$

where $x_1, x_2, ..., x_m$ are the input signals, $w_{k1}, w_{k2}, ..., w_{km}$ are the synaptic weights of neuron k, u_k is the linear combiner output due to the input signals, b_k is the bias, $f(\cdot)$ is the activation function, and y_k is the output signal of the neuron. The activation function $f(\cdot)$ defines the output of a neuron in terms of the induced local field v. Here are several types of activation functions:

- Threshold Function:

$$f(v) = \begin{cases} 1, & \text{if } v \geq 0 \\ 0, & \text{if } v < 0. \end{cases} \tag{2.7}$$

A neuron that has this form of activation function is referred to as the McCulloch-Pitts model. In this model, the output of a neuron takes on the value of 1 if the induced local field of that neuron is nonnegative, and 0 otherwise.

- Piecewise-Linear Function:

$$f(v) = \begin{cases} 1, & \text{if } v \geq +\frac{1}{2} \\ v, & \text{if } +\frac{1}{2} > v > -\frac{1}{2} \\ 0, & \text{if } v \leq -\frac{1}{2}. \end{cases} \tag{2.8}$$

  This form of activation function may be viewed as an approximation to a non-linear amplifier.

- Sigmoid Function (Figure 8*a*):

$$f(v) = \frac{1}{1 + \exp(-av)}, \tag{2.9}$$

  where a is a slope parameter. For the corresponding form of a sigmoid function the hyperbolic tangent function ($f(v) = a\tanh(bv)$, where parameter a and b will be provided in Chapter III) can be used. This form of function is most commonly used in the construction of artificial neural networks. This function is differentiable, whereas the threshold function is not.

- Radial Basis Function (Figure 8*b*):

$$f(v) = \exp(-v^2). \tag{2.10}$$

  Radial basis function has maximum of 1 when its input is 0.

One most common class of a feedforward neural network is multilayer perceptron network, which consists of a set of input units that constitute the input layer, one or more hidden layer of computation nodes, and an output layer of computation nodes. The input signals propagate through the network in a forward direction, on a layer-by-layer basis. These neural networks are commonly referred to as multilayer perceptrons (MLPs). MLPs have been applied successfully to solve difficult and

diverse problems by training them in a supervised manner with the backpropagation algorithm, which is based on gradient descent learning. In the next section, a brief review about backpropagation algorithm, which is a popular training algorithm for MLPs, will be provided along with the heuristics for making the backpropagation algorithm perform better.

2. Backpropagation

Single layered perceptrons can only do a linear classification, but MLPs trained by backpropagation algorithm are capable of various nonlinear classifications. The backpropagation algorithm employs gradient descent to attempt to minimize the sum of squared error between the network output values and the target values for these outputs. The simplest implementation of the backpropagation algorithm updates the network weights and biases in the direction in which the error function decreases most rapidly. In the following, I will briefly describe the backpropagation algorithm and several heuristics, which I applied in the current experiments, to make the backpropagation algorithm perform faster and stabler.

a. Backpropagation Algorithm

The backpropagation algorithm can be described as follows [47]:

- Each training example is a pair of the form $< \vec{x}, \vec{t} >$, where $\vec{x}$ is the vector of network input values, and $\vec{t}$ is the vector of target network output values.
- η is the learning rate. n_{in} is the number of network inputs, n_{hidden} is the number of units in hidden layer, and n_{out} is the number of output units.
- The input from unit i into unit j is denoted x_{ji}, and the weight from unit i to j is denoted w_{ji}.

- Create a feed-forward network with n_{in} inputs, n_{hidden} hidden units, and n_{out} output units.
- Initialize all network weights to small random numbers (e.g., between -0.05 and 0.05).
- Until the termination condition is met, For each $< \vec{x}, \vec{t} >$ in training examples, do:
  1. Propagate the input forward through the network:
  2. Input the instance $\vec{x}$ to the network and compute the output o_u or every unit u in the network.
  3. Propagate the errors backward through the network:
  4. For each network output unit k, calculate its error term δ_k

$$\delta_k = o_k(1 - o_k)(t_k - o_k). \tag{2.11}$$

  5. For each hidden unit h, calculate its error term δ_h

$$\delta_h = o_h(1 - o_h) \sum_{k \in outputs} w_{kh}\delta_k. \tag{2.12}$$

  6. Update each network weight w_{ji}

$$w_{ji} = w_{ji} + \Delta w_{ji} \tag{2.13}$$

$$\Delta w_{ji} = \eta \delta_j x_{ji}. \tag{2.14}$$

The backpropagation algorithm provides an approximation to the trajectory in weight space computed by the method of steepest descent. The smaller the learning rate η is set, the smaller the changes to the synaptic weights in the network will be for each iteration, and the smoother the trajectory in weight space will be. This, however, is

attained at the cost of a slower rate of learning. If, on the other hand, the learning rate η is too high in order to speed up the rate of learning, the resulting large changes in the synaptic weights assume such a form that the network becomes unstable (i.e., oscillatory). A simple method of increasing the rate of learning but avoiding the danger of instability is to modify equation 6 by including a momentum term:

$$\Delta w_{ji} = \alpha \Delta w_{ji}(n-1) + \eta \delta_j x_{ji},$$

where α is usually a positive number called the *momentum constant*. It has some beneficial effects on the learning behavior of the algorithm and prevents the learning process from terminating in a shallow local minimum on the error surface.

b. Heuristics for Faster Backpropagation

There are numerous factors involved in the design of the networks. Here are some of the heuristics for making the backpropagation algorithm perform faster, which I employed in my experiment (adapted from [48]). The details on how these are applied are provided in the following chapter.

- Sequential vs. batch method. The sequential mode of backpropagation learning is computationally faster than the batch mode, especially when the training data set is large and highly redundant.

- Activation Function. A network trained with the backpropagation algorithm may, in general, learn faster when the sigmoid activation function is antisymmetric ($f(-v) = -f(v)$) than when it is nonsymmetric. A popular example of such a sigmoid activation function is the hyperbolic tangent function.

- Target values. It is important that the target values be chosen within the range of the activation function.

- Normalizing the inputs. Each input variable should be preprocessed so that its mean value, averaged over the entire training set, is close to zero, or else it is small compared to its standard deviation.

- Learning rates (backpropagation with adaptive learning rate).

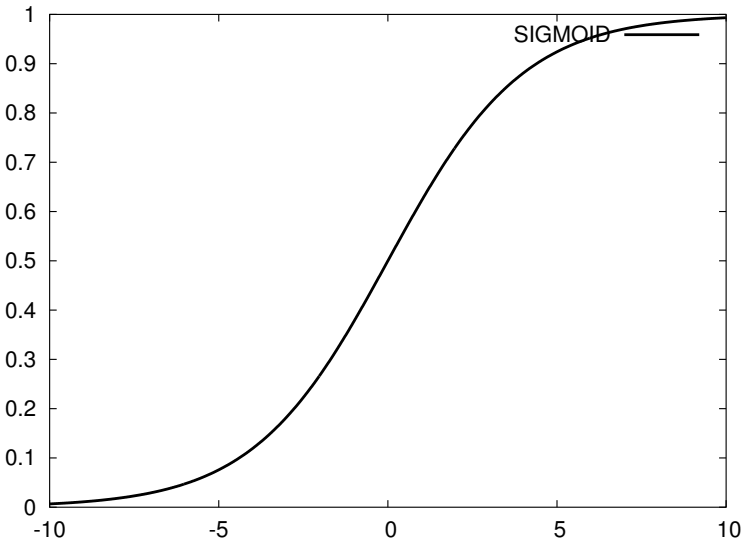
    1. Every adjustable network parameter of the cost function should have its own individual learning-rate parameter.

    2. Every learning rate parameter should be allowed to vary from one iteration to the next.

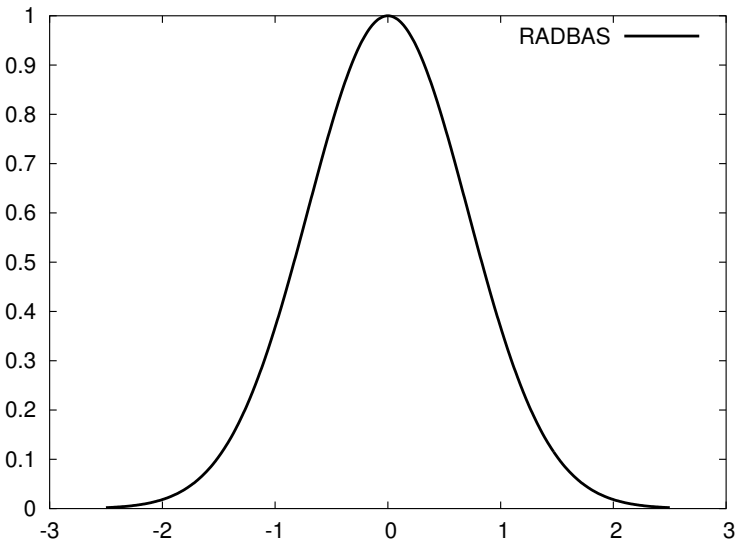
    3. When the derivative of the cost function with respect to a synaptic weight has the same algebraic sign for several consecutive iterations of the algorithm, the learning rate parameter for that particular weight should be increased.

    4. When the algebraic sign of the derivative of the cost function with respect to a particular synaptic weight alternates for several consecutive iterations of the algorithm, the learning rate parameter for that weight should be decreased.

In this chapter, I provided a brief summary of the properties of early visual system and the Gabor filter, which plays a critical role in texture processing in the visual system and, therefore, in the preprocessing stages of the experiments in this thesis. A brief overview of the backpropagation learning algorithm, which is the foundation of the computational experiment, was also provided. In the following chapter, the detailed experimental method based on the background presented above will be provided.



(a) Sigmoid function



(b) Radial basis function

Fig. 8. Activation functions. (a) Sigmoid function and (b) Radial basis function.

CHAPTER III

EXPERIMENTAL METHOD

To investigate the relative difficulty of learning to segment textures in 2D vs. 3D configurations and to find out whether the learned ability of 3D texture processing can easily be transferred to 2D perception of texture, texture boundary detection on 2D surface and in 3D space were simulated using Matlab. In this chapter, I will describe in detail how I prepared the two different arrangements (Section A), and explain how I trained two standard MLPs to discriminate these texture arrangements (Section B). Two separate networks that are identical in structure were trained, one with input prepared in a 2D arrangement (I will refer to this network as the *2D-net*), and the other in a 3D arrangement (the *3D-net*).

A. Input Preparation

Three sets of texture stimuli S_1, S_2, and S_3 were prepared for the experiments. Textures in S_1 were simple artificial texture images (bars of orientation 0, $\frac{\pi}{4}$, $\frac{\pi}{2}$, or $\frac{3\pi}{4}$ in 2 different spatial frequencies) and those in S_2 were more complex texture images (bars with orientations different from S_1 or more complex patterns such as crosses and circles), which were adopted from Krose [49] and Julesz [5]. Textures in S_3 were real texture images from the widely used Brodatz texture collection [50], as shown in figure 9.

For the training of the 2D-net and the 3D-net, eight simple texture stimuli from S_1 were used. For testing the performance of the 2D-net and the 3D-net, all sets of texture stimuli, S_1, S_2, and S_3, were used. To extract the primitive features in a given texture, I used a bank of Gabor filters as explained in Chapter II. Previous studies have shown that Gabor filters closely resemble experimentally measured receptive

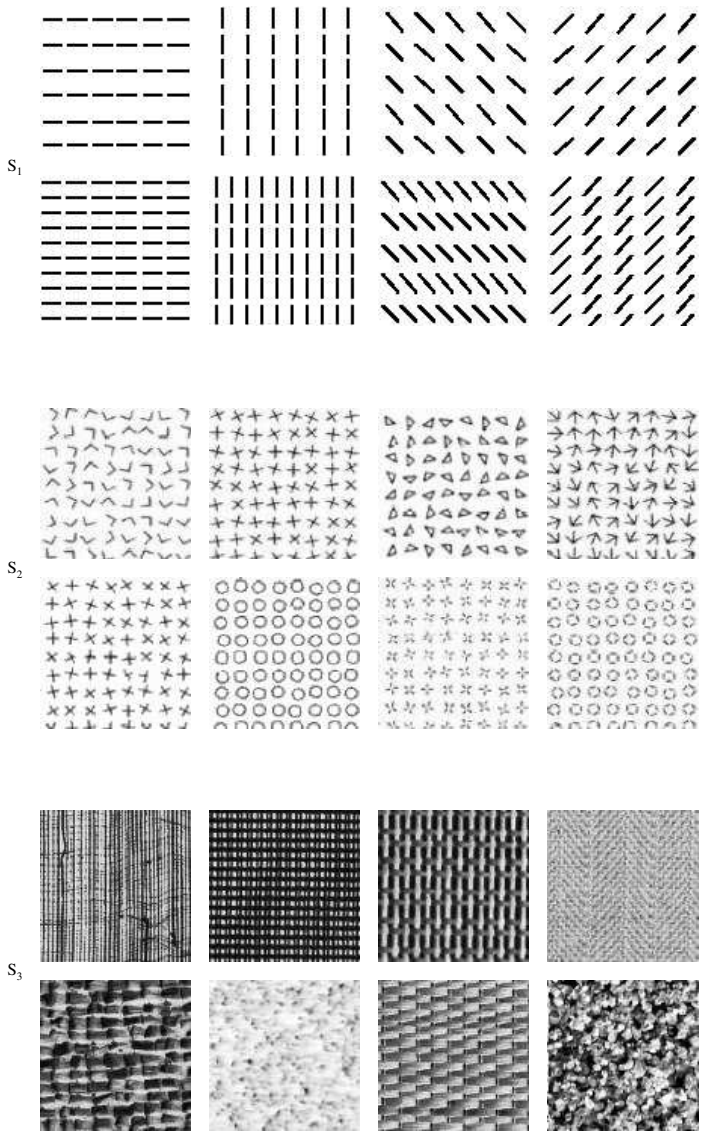


Fig. 9. Texture stimuli. Three texture sets S_1, S_2, and S_3 are shown from the top to the bottom row.

fields in the visual cortex [44] [24] and have been widely used to model the response of visual cortical neurons. Thus, I used a bank of oriented Gabor filters to approximate the responses of simple cells in the primary visual cortex. The Gabor filter is defined as follows [37]:

$$G_{\theta,\phi,\sigma,\omega}(x,y) = \exp^{-\frac{x'^2+y'^2}{2\sigma^2}} \cos(2\pi\omega x' + \phi), \tag{3.1}$$

where θ is the orientation, ϕ is the phase, σ is the standard deviation (width) of the envelope, ω is the spatial frequency, (x, y) represents the pixel location, and x' and y' are defined as:

$$x' = x\cos(\theta) + y\sin(\theta) \tag{3.2}$$

$$y' = -x\sin(\theta) + y\cos(\theta). \tag{3.3}$$

For simplicity, only four different orientations $(0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4})$ were used for θ. (Below, I will refer to $G_{\theta,\phi,\sigma,\omega}$ as simply G.) To adequately sample the spatial-frequency feature of the input stimuli, three frequency ranges (1 to 3 cycles/degree) were used for ω. The size of the filter was 16×16, $\sigma = 16/3$, and $\phi = \pi/2$. This resulted in 12 filters G_i (for $i = 1..12$) for the computation of simple cell responses as shown in figure 10. To get the Gabor energy matrix C, a gray-level intensity matrix I was obtained from the images randomly selected from S_1 and convolved with the filter bank G_i:

$$C_i = I * G_i, \tag{3.4}$$

where $(i = 1..12)$ denotes the index of a filter in the filter bank, and $*$ represents the convolution operator. The Gabor filtering stage is linear, but models purely based on linear mechanisms are not able to reproduce experimental data [11]. Thus, half-wave rectification is commonly used to provide a nonlinear response characteristic following linear filtering. However, in the current experiments, full-wave rectification was used

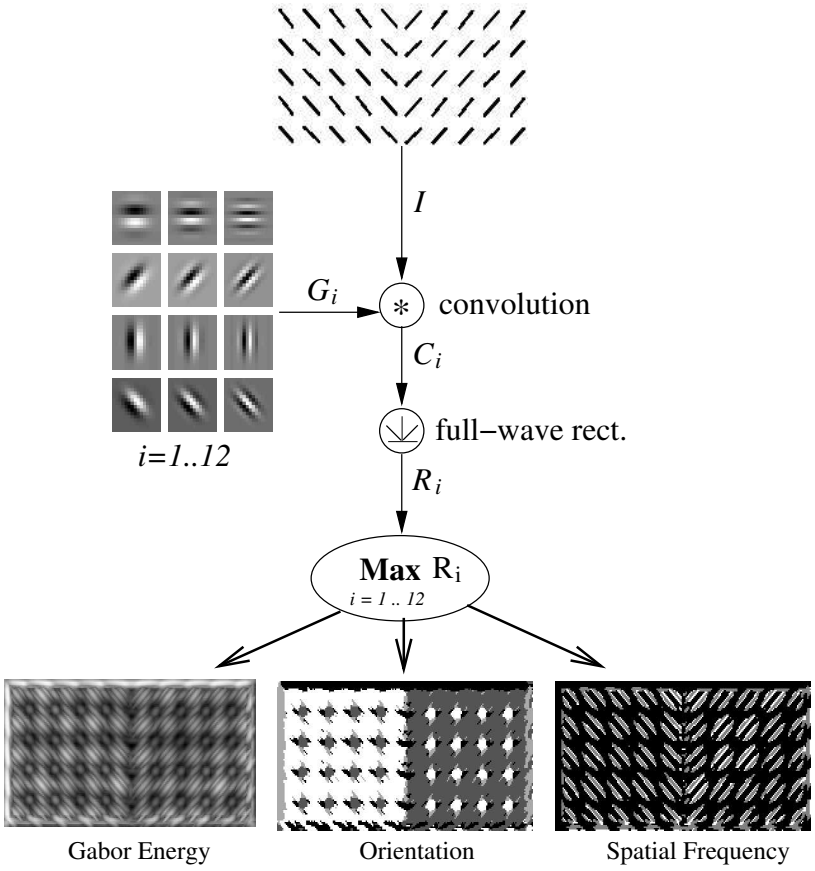


Fig. 10. Gabor filter bank.The process used to generate two orientation response matrices is shown. The texture I is first convolved with the Gabor filters G_i (for $i = 1..12$), and the resulting responses are passed through a full-wave rectifier resulting in R_i. Finally, we get the Gabor energy matrix $E(x, y)$, Orientation response matrix $O(x, y)$, and Frequency response matrix $F(x, y)$.

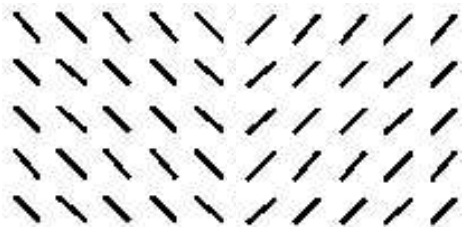
as in [6], which is similar to half-wave rectification, but is simpler to implement[1]. The final full-waved rectified Gabor feature response matrix is calculated as

$$R_i = |C_i|, \tag{3.5}$$

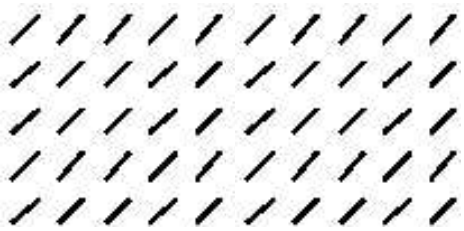
for $i = 1..12$. I acquired three Gabor response matrices (or maps), which are Gabor energy response matrix E, orientation response matrix O and frequency response matrix F, for each sample texture pair. First, to get the Gabor energy response matrix E, only one maximally responding values at location (x, y) from the twelve response matrices R_i were selected. In addition to the Gabor energy matrix, orientation response matrix and frequency response matrix were computed to avoid the loss of orientation and frequency properties at a given location. The orientation response matrix O had orientation index $(1 \leq O(x, y) \leq 4)$ of the filter that are maximum response at location (x, y) out of 12 filters. The frequency response matrix F had frequency index $(1 \leq F(x, y) \leq 3)$ of the filter that are maximum response at location (x, y) out of 12 filters. The same filtering procedure was used for both the 2D and the 3D arrangement of textures, which will be described below. Figure 10 shows the Gabor filter bank and the three response matrices E, O, and F of the given texture pair.

To get the 2D training samples for the 2D-net, two randomly selected textures from S_1 were paired and convolved with the Gabor filter bank (figure 10). Gabor energy response matrix was acquired first, and orientation response matrix and frequency response matrix were computed from the 12 different response matrices that were used to get Gabor energy response matrix. Each training input in the 2D training set consisted of three 32-element vectors (say, ξ_k^{2D}, where k is the training sample

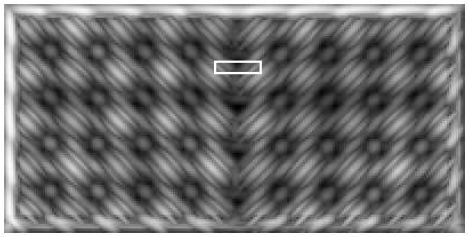
[1]Full-wave rectification is equivalent to summing the outputs of the two corresponding half-wave rectification channels (see, e.g. Bergen and Adelson [12] [11]).



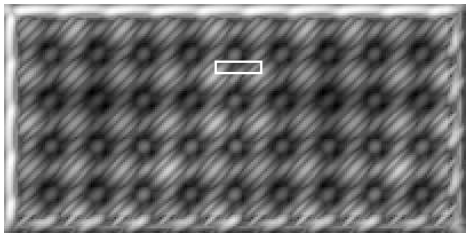
(a) Texture image with boundary



(b) Texture image without boundary

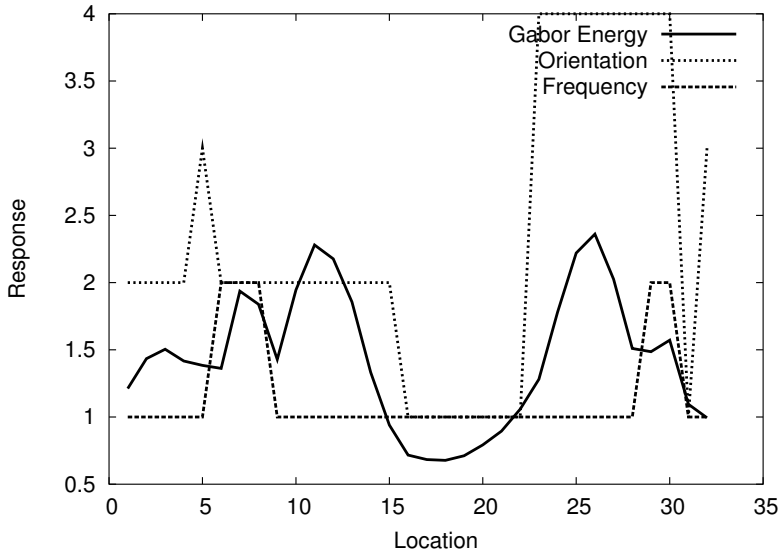


(c) Gabor energy response to (a)

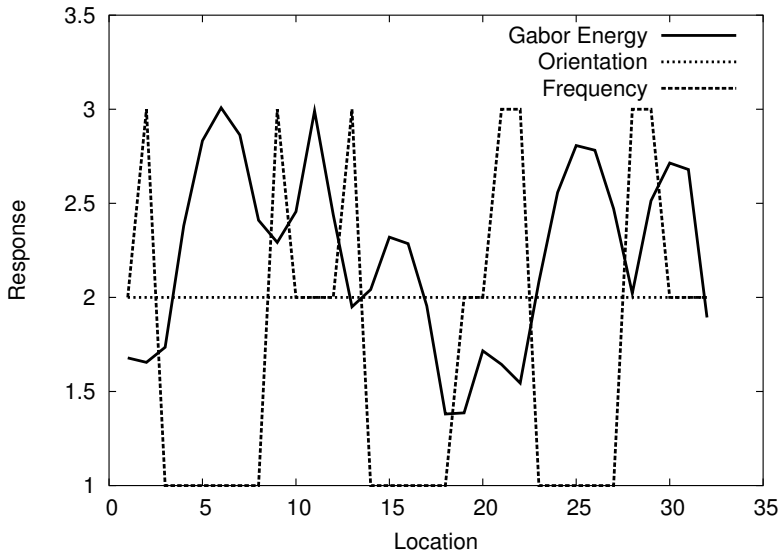


(d) Gabor energy response to (b)

Fig. 11. Generating the 2D input set (2D preprocessing). The procedure used to generate the training data is shown. (a) Input with a texture boundary. (b) Input without a texture boundary (c) Gabor energy response calculated from (a). (d) Gabor energy response calculated from (b).



(e) Response profile of figure 11(c)

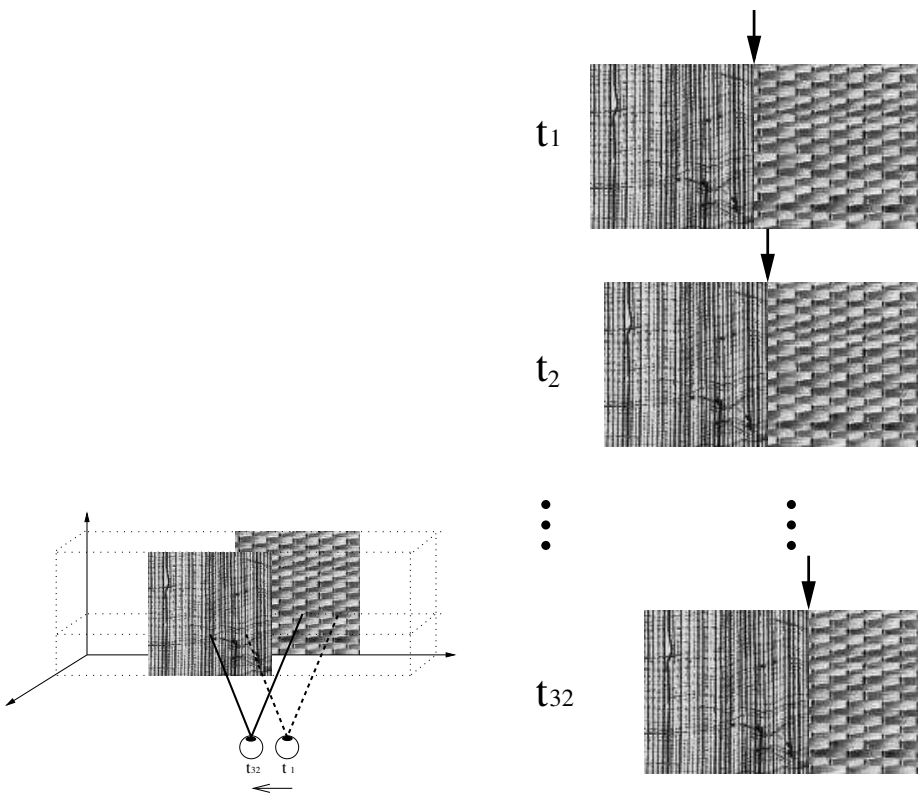


(f) Response profile of figure 11(d)

Fig. 12. Response profiles generated by the 2D preprocessing. (a) The response profile (Gabor energy, orientation response, frequency response) from the 32-pixel wide area marked with a white rectangle in figure 11(c). (b) The response profile from the 32-pixel wide area marked with a white rectangle in figure 11(d).

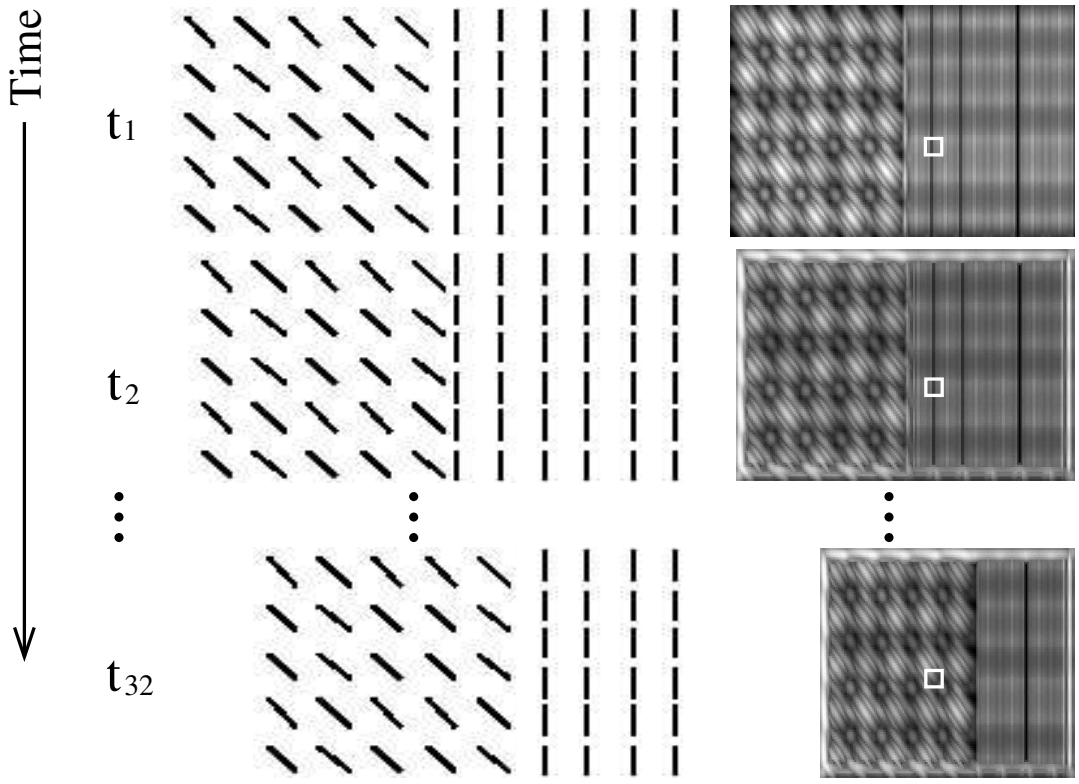
index) taken from a short horizontal strip (response profile) of the Gabor response matrix, the orientation response matrix, and the frequency response matrix, which resulted in a 96-element vector. A single scalar value (say, ζ_k^{2D}) indicating the existence (= 1) or nonexistence (= 0) of a texture boundary within that strip was paired with ξ_k^{2D}. The vector ξ_k^{2D} was taken from a horizontal strip centered at (x_c, y_c) within the Gabor energy matrix, the orientation response matrix, and the frequency response matrix respectively (e.g., the white rectangle in figure 11*c* & *d*), where x_c is the horizontal center where the two textures meet, and y_c is randomly chosen within the full height of the matrix. The Gabor energy matrix was normalized so that each value in the matrix have the range $0 \leq E(x, y) \leq 5$. When the two selected textures were the same, a texture boundary will not occur at the center; and if they were different, a texture boundary will occur. The number of input-target pair (ξ_k^{2D},ζ_k^{2D}) in each class, i.e., boundary vs. no boundary, was balanced so that each class is equally represented. Figure 12*a* shows an example vector ξ_k^{2D} when there was a texture boundary, and figure 12*b* a case without a boundary.

For the training samples for the 3D-net, motion cue was applied to simulate self-motion of an observer as shown in figure 13. One texture from a pair of textures was overlayed on top of the other and the texture above was allowed to slide over the one below, which resulted in successive further occlusion of the texture below. The texture above was moved by one pixel 32 times and each time the resulting 2D image (I'_j, for $j = t_1 ... t_{32}$; figure 14*a*) was convolved with the oriented Gabor filter bank followed by full-wave rectification as in the 2D preprocessing case (figure 14*b*). To generate a single training input pair (ξ_k^{3D}, ζ_k^{3D}) for the 3D-net, at each time step the Gabor energy response value $E(x_c, y_c)$, orientation response value $O(x_c, y_c)$ and frequency response value $F(x_c, y_c)$ were collected into a 92-element vector, where x_c was 16 pixels away to the right from the initial texture boundary in the middle,



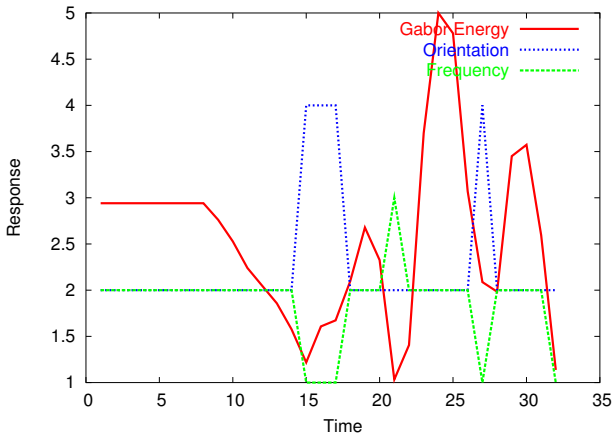
(*a*) Texture in 3D (*b*) Resulting 2D view

Fig. 13. Generating the 3D input set (3D preprocessing). (*a*) A 3D configuration of textures and (*b*) the resulting 2D views before, during, and after the movement are shown. As the viewpoint is moved from the right to the left (t_1 to t_{32}) in 32 steps, the 2D texture boundaries in (*b*) marked by black arrows show a subtle variation.

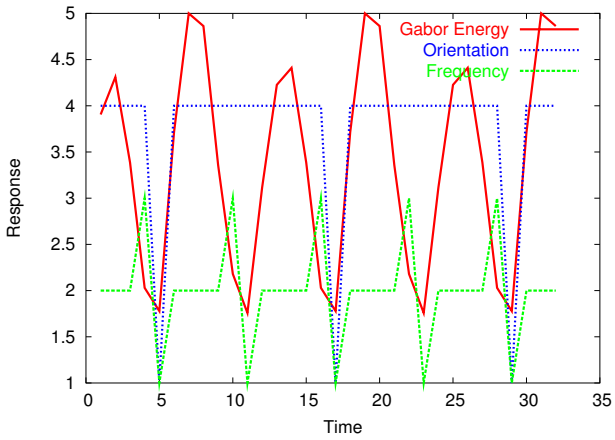


(*a*) Input over time (*b*) Response to (*a*)

Fig. 14. Generating 3D input set through motion (3D preprocessing). (*a*) Texture pair images resulting from simulated motion: I'_j for $j = t_1..t_{32}$. (*b*) The response matrix of the texture pair.



(c) Temporal profile of figure 14(b)



(d) Temporal profile (no boundary)

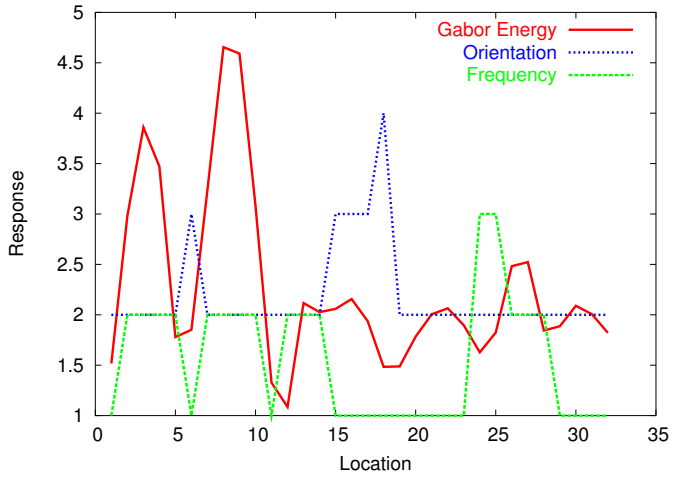
Fig. 15. Response profiles generated by the 3D preprocessing. (a) Response profile obtained over time near the boundary of two different texture images (marked by the small squares). (b) A similarly measured response profile collected over time, using a different input texture, near a location without a texture boundary (note the periodic peaks).

and y_c was selected randomly for each new input pair but remained the same within the same input pair (the white square in figure 14*b* shows an example). Figure 15*a* shows an example of such a vector ξ_k^{3D} (note that the x-axis represents time) for a case containing a texture boundary, and figure 15*b* a case without a boundary. The target value ζ_k^{3D} of the input pair (ξ_k^{3D}, ζ_k^{3D}) was set in a similar manner as in the 2D case, either to 0 (no boundary) or 1 (boundary). When collecting the training samples for the 3D-net, the above procedure was performed with two different 3D configurations. In the first 3D configuration, the texture on the left side is on top of the texture on the right side with self-motion of observer from right to left. In the second 3D configuration, the texture on the right is on top of the texture on the left side with self-motion of observer from left to right. For a balanced training set, the same number of samples were collected for each 3D configuration.

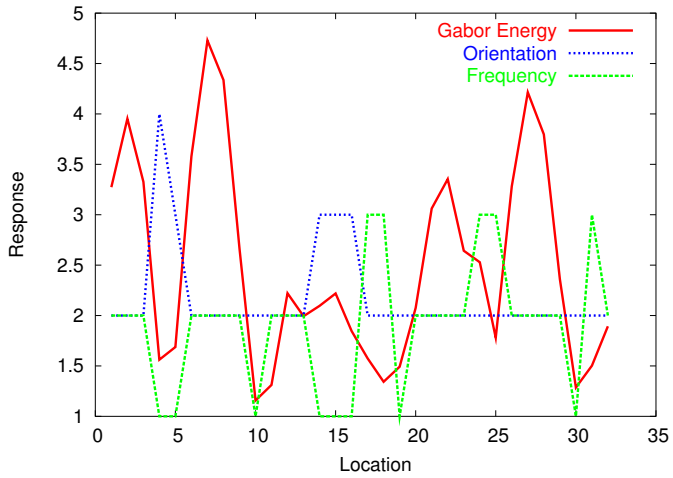
For a fair comparison between the 2D and the 3D arrangements, 400 training samples were collected for each combination of two different textures to make 2,400 samples with a target value of 1, and the same number of samples with a target value of 0. This resulted in 4,800 input-target samples for each case ($1 \leq k \leq 4{,}800$). These 4,800 input-target samples from each training set were then randomly ordered during training. Typical response profiles of 2D processing and 3D processing are shown in figure 16, figure 17, figure 18, figure 19, figure 20, and figure 21 respectively.

B. Training the Texture Segmentation Networks

I used standard multilayered perceptrons (MLPs) to perform texture boundary detection. The networks (2D-net and 3D-net), which consisted of two layers including 96 input units, 16 hidden units and 1 output unit, were trained for 2,000 epochs each

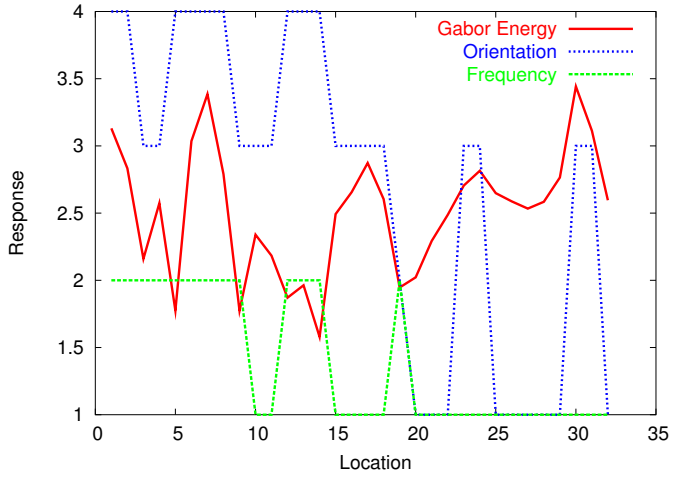


(a) S_1 with boundary

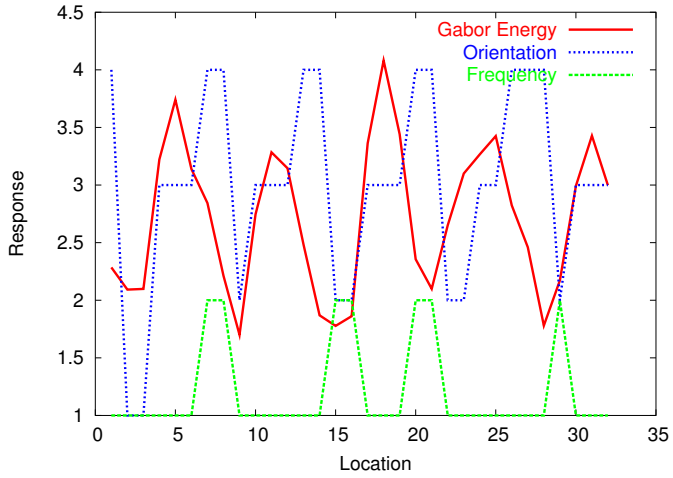


(b) S_1 without boundary

Fig. 16. Typical response profiles of 2D input samples from S_1. Note that the profiles with a boundary (a) is less symmetric about the center compared to those without a boundary (b).

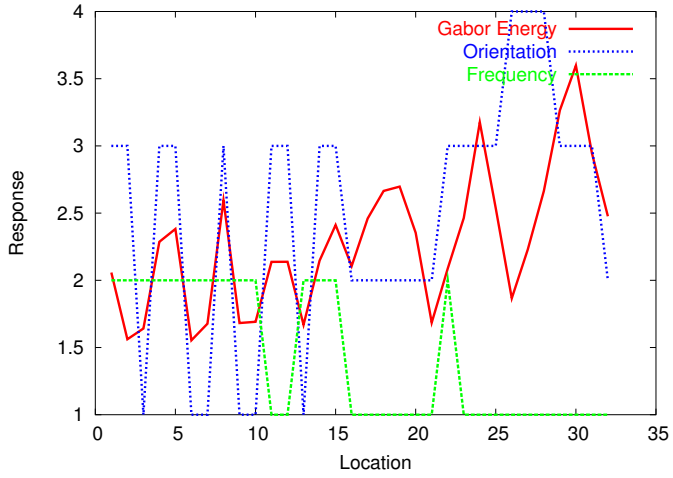


(a) S_2 with boundary

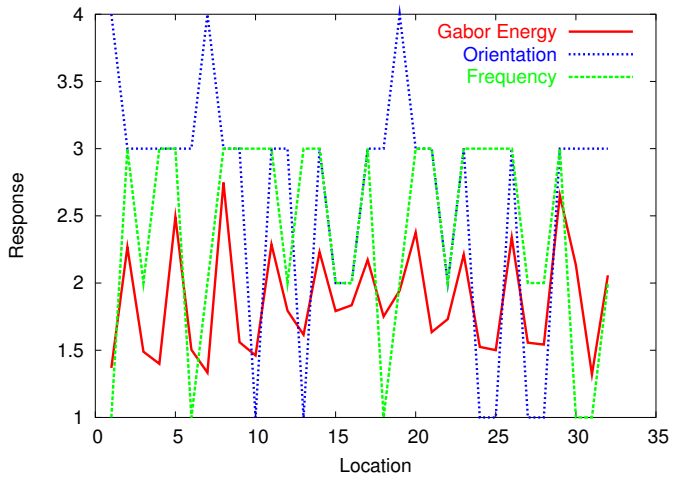


(b) S_2 without boundary

Fig. 17. Typical response profiles of 2D input samples from S_2. Note that (1) the profiles with a boundary (a) is less symmetric about the center compared to those without a boundary (b); and that (2) S_2 is more complex than S_1.

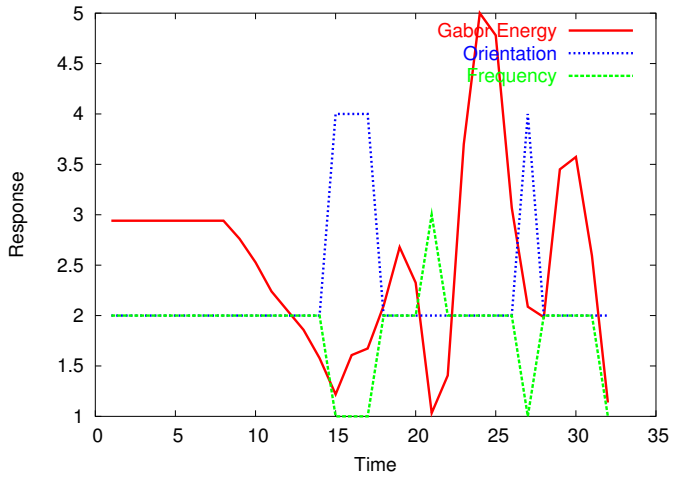


(a) S_3 with boundary

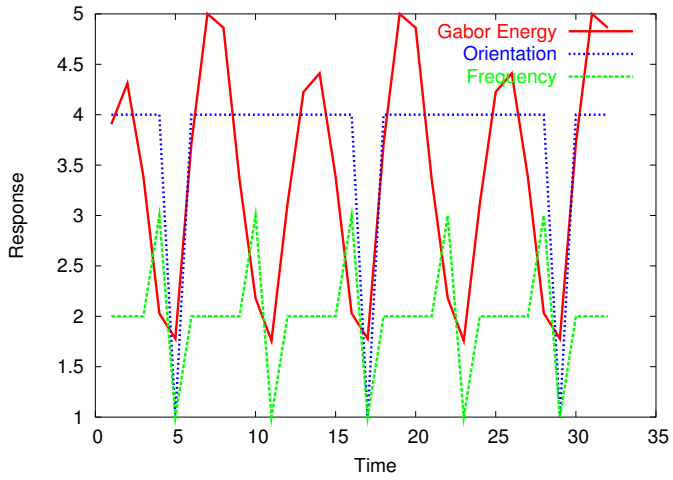


(b) S_3 without boundary

Fig. 18. Typical response profiles of 2D input samples from S_3. Note that (1) the profiles with a boundary (a) are less symmetric about the center compared to those without a boundary (b); and that (2) S_3 is more complex than S_2.

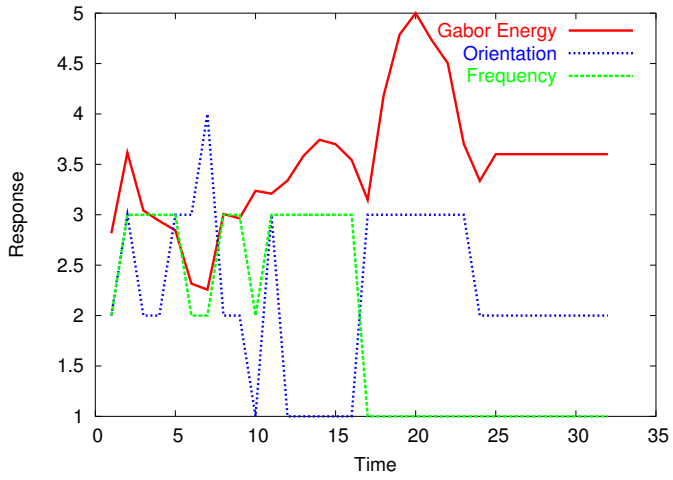


(*a*) S_1 with boundary

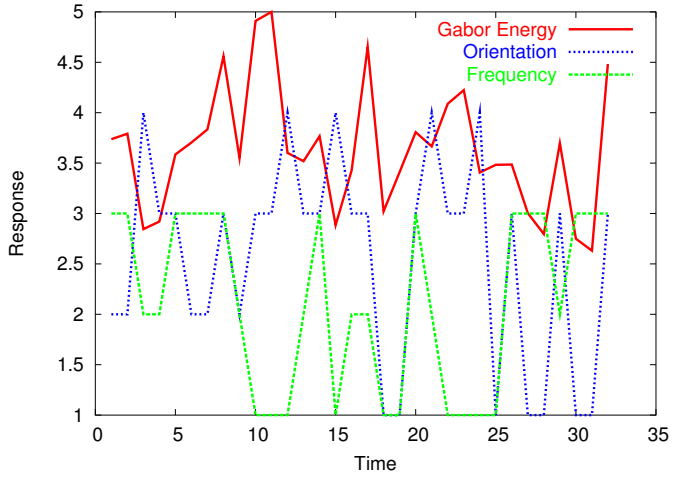


(*b*) S_1 without boundary

Fig. 19. Typical response profiles of 3D input samples from S_1. The same properties as in figure 16 are observed, but the asymmetry in the boundary cases is more pronounced.

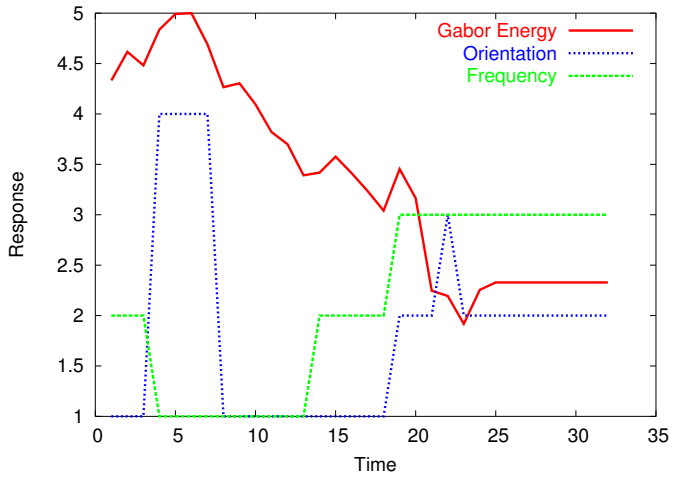


(a) S_2 with boundary

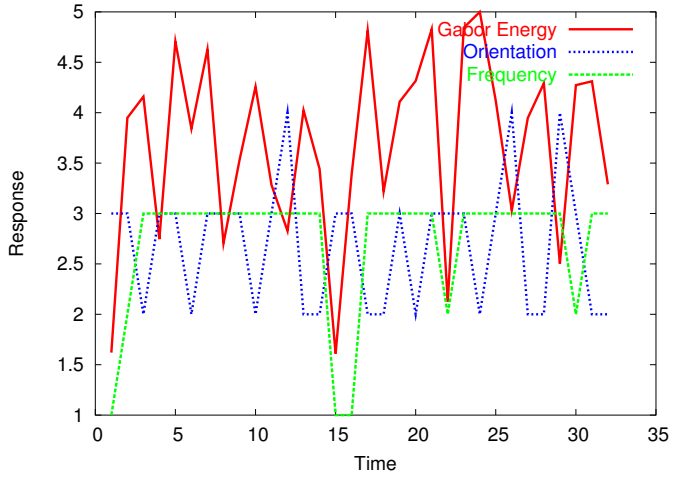


(b) S_2 without boundary

Fig. 20. Typical response profiles of 3D input samples from S_2. The same properties as in figure 17 are observed, but the asymmetry in the boundary cases is more pronounced.



(a) S_3 with boundary



(b) S_3 without boundary

Fig. 21. Typical response profiles of 3D input samples from S_3. The same properties as in figure 18 are observed, but the asymmetry in the boundary cases is more pronounced.

using standard backpropagation[2]. The goal of this study was to compare the relative learnability of the 2D vs. the 3D texture arrangements, thus a backpropagation network was good enough for our purpose. The hyperbolic tangent function was used for the activation function of the hidden layer, which is defined as:

$$f(v) = a \tanh(bv), \tag{3.6}$$

where a and b are constants, which are set to 1.7159 and $\frac{2}{3}$ respectively (following [51]). The hyperbolic tangent function is antisymmetric ($f(-v) = -f(v)$), thus considered to learn faster than nonsymmetric activation function [45]. For the activation function of the output layer that consisted of one unit, the radial basis function (RBF) was used. The use of the radial basis function in standard MLP is not common and it is usually used as an activation function of the hidden layer in radial basis networks, which has additional data-independent input to the output layer. In my experiment, as shown in the previous section, an input vector to MLP is symmetric about the center when there is no boundary. On the other hand, an input vector to MLP is quite asymmetric when there is a boundary, but the mirror image of that vector should result in the same class. This observation led me to use the radial basis function, which has a Gaussian profile as shown in figure 8*b*. Several preliminary training trials showed that the use of the RBF as the activation function enabled both the 2D-net and the 3D-net to converge faster (data not provided here).

For the training, the input vectors were drawn from the texture set S_1. Backpropagation with momentum and adaptive learning rate was applied. The delta rule (from the equations (2.13) and (2.14)):

$$w_{ji} = w_{ji} + \Delta w_{ji}, \tag{3.7}$$

[2]Matlab neural networks toolbox was used for the simulations.

$$\Delta w_{ji} = \eta \delta_j x_{ji}, \tag{3.8}$$

was modified by including a momentum term as

$$\Delta w_{ji} = \alpha \Delta w_{ji}(n-1) + \eta \delta_j x_{ji}, \tag{3.9}$$

where α is usually a positive number called the *momentum constant*. The momentum constant must be restricted to the range $0 \leq |\alpha| < 1$ for the network to converge. When α is zero, the backpropagation algorithm operates without momentum. Also the momentum constant α can be positive or negative, although it is unlikely that a negative α would be used in practice [48]. To determine the best learning parameters, several preliminary training runs were done with combinations of learning rate parameter $\eta \in \{0.01, 0.1, 0.5\}$ and momentum constant $\alpha \in \{0.0, 0.5, 0.9\}$. MLP with each combination was trained with the same set of inputs so that the results of the experiment can be compared directly. Each training set consisted of 280 examples, drawn from S_1 and processed by the 2D preprocessing procedure. The training process continued for 1,000 epochs. Some of the resulting learning curves are shown in figure 22. The MLPs with other combination of parameters failed to converge. Based on these preliminary training tests, I chose the learning parameters as shown in Table I. Even though the combinations of $(\eta = 0.1, \alpha = 0.0)$ and $(\eta = 0.1, \alpha = 0.1)$ exhibited quite good final performance, the learning processes caused oscillations in

TABLE I. Optimal combination of parameters for the backpropagation algorithm.

Parameter	Symbol	Value
Optimal learning-rate parameter	η_{opt}	0.01
Optimal momentum constant	α_{opt}	0.5

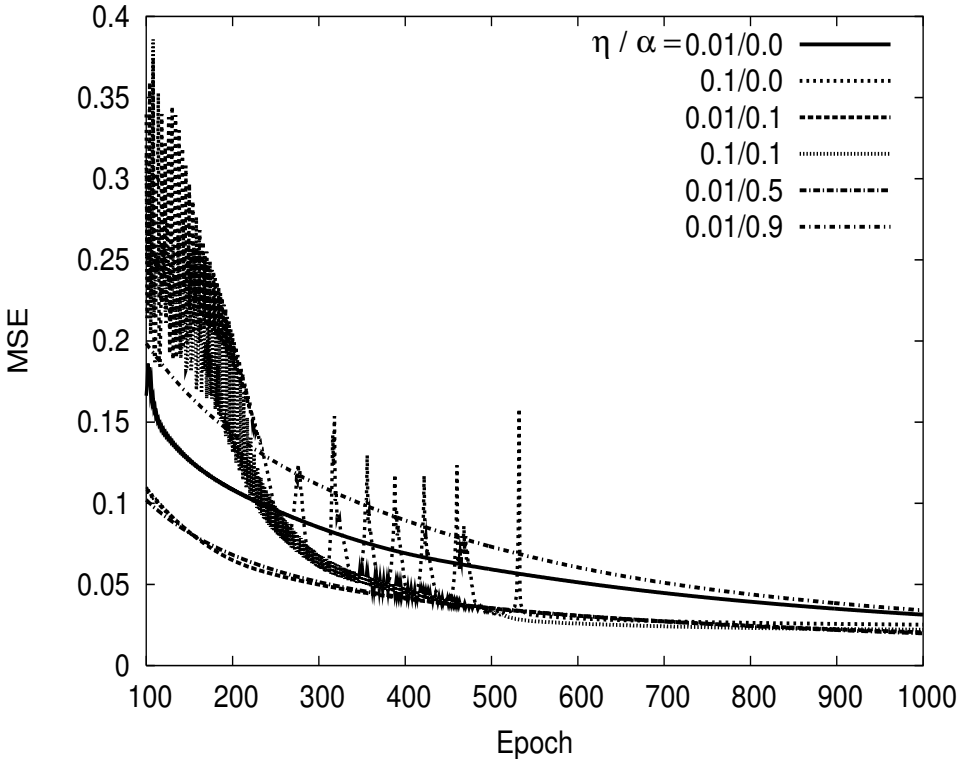


Fig. 22. Learning curves from preliminary training runs with different learning rates (η) and momentum constants (α).

the MSE during learning. Therefore, those configurations were not selected.

I also applied standard heuristics to speed up and stabilize the convergence of the networks as introduced in the previous chapter. First, each input variable was preprocessed so that its mean value, averaged over the entire training set, is close to zero. Secondly, adaptive learning rate was applied. For each epoch, if the mean squared error (MSE) decreased toward the goal (10^{-4}), then the learning rate (η) was increased by the factor of η_{inc}:

$$\eta_n = \eta_{n-1} \times \eta_{\text{inc}}, \tag{3.10}$$

where n is the epoch. If MSE increased by more than $MSE_{\text{MAX}} = 1.04$, the learning

rate was adjusted by the factor of η_{dec}:

$$\eta_n = \eta_{n-1} \times \eta_{\text{dec}}. \tag{3.11}$$

The constants selected above ($\eta = 0.01, \alpha = 0.5$) were used for the second test training to choose the optimal adaptive learning rate factors (η_{inc} and η_{dec}). Combinations of the factors $\eta_{\text{inc}} \in \{1.01, 1.05, 1.09\}$ and $\eta_{\text{dec}} \in \{0.5, 0.7, 0.9\}$ were used during the test training to observe their effects on convergence. The learning curves of five best combinations of factors are shown in figure 23, where only the beginning part (up to 100 epochs) is shown, but the learning process continued up to 1,000 epochs. The combination of factors $\eta_{\text{inc}} = 1.01$ and $\eta_{\text{dec}} = 0.5$ were chosen based on these results.

The 2D-net and the 3D-net were trained 10 times each with parameters chosen from the preliminary training above ($\eta = 0.01$, $\alpha = 0.5$, $\eta_{\text{inc}} = 1.01$, and $\eta_{\text{dec}} = 0.5$). After the training of the two networks, the speed of convergence and the classification accuracy were compared. To test generalization and transfer potentials, test stimuli drawn from the texture sets S_1, S_2, and S_3 were preprocessed using both 2D- and 3D-preprocessing to obtain six sample input sets. The results from these experiments will be presented in the following chapter. These input samples were then presented to the 2D-net and the 3D-net to compare the performances of the two networks on these six sample input sets.

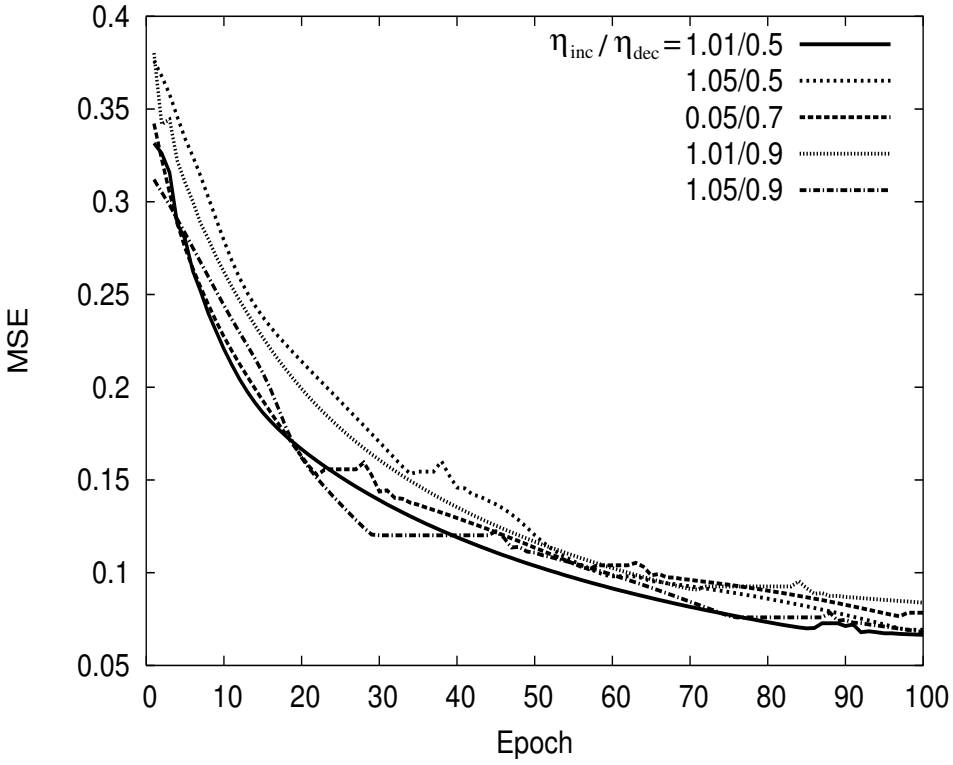


Fig. 23. Learning curves from preliminary training runs with different adaptive learning rate factors η_{inc} and η_{dec}.

CHAPTER IV

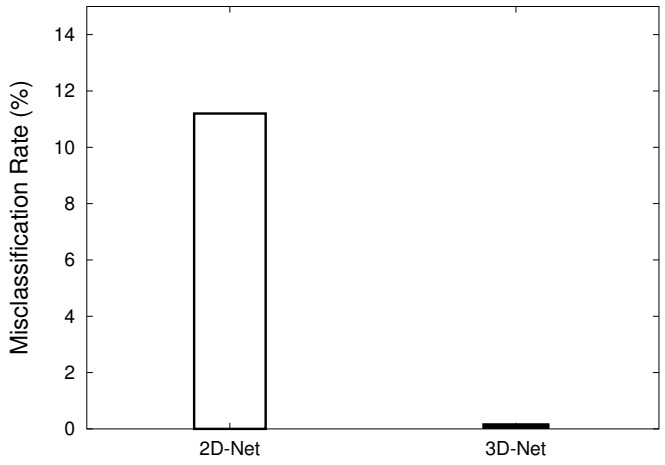
RESULTS

In section A, I will present the performance of the two trained networks (2D-net and 3D-net) in terms of speed and accuracy, and in section B, the performance of the two networks on novel texture images that were not used during training.

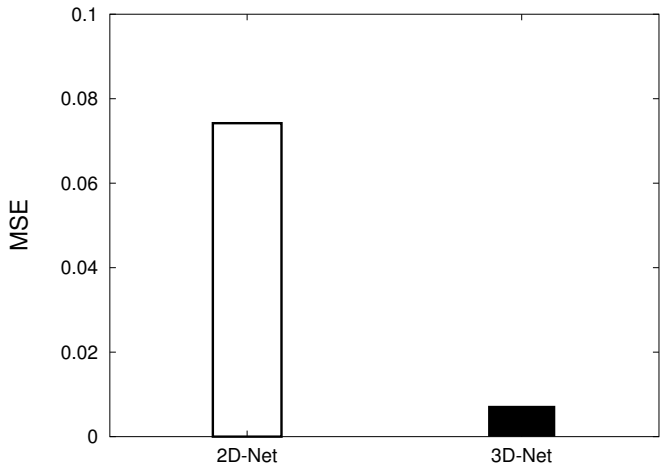
A. Speed of Convergence and Accuracy of Classification on the Training Set

Figure 24 and figure 25 show the 3 best learning curves of each network out of 10 trials of training. The learning processes continued for 2,000 epochs. After 2,000 epochs, the average mean squared error (MSE) of the 2D-net was 0.0742 and that of the 3D-net was 0.0073. For the 10 trials, the results were comparable each time (data not presented here). The fact that the final MSEs of the three curves for each network did not vary significantly as shown in figure 25 suggests that the number of epochs was adequate. A noticeable difference in the two learning curves is that there are significant fluctuations in the learning curves of the 2D-net, which often prevented convergence of a network. These results indicate that the 3D-net is easier to train than the 2D-net. In other words, texture arrangements in 3D may be easier to segment than those in 2D. The misclassification rate, which was computed by using threshold of 0.5, in the 2D-net for the 2D training set was 11.2% and that of the 3D-net for the 3D training set was 0.2%, thus, accuracy was also higher in the 3D-net for the training data.

I analyzed the training sets, from the perspective of statistical analysis of the data, using principal components analysis (PCA), which provides an effective technique for dimensionality reduction and visualization. We may reduce the number of features needed for effective data representation of 2D training set and 3D training



(a) MSE after 2,000 epochs



(b) Misclassification Rate on Training Set

Fig. 24. Performance of the networks. (a) The final MSE of the networks after training for 2,000 epochs, and (b) the misclassification rate of the networks on training set.

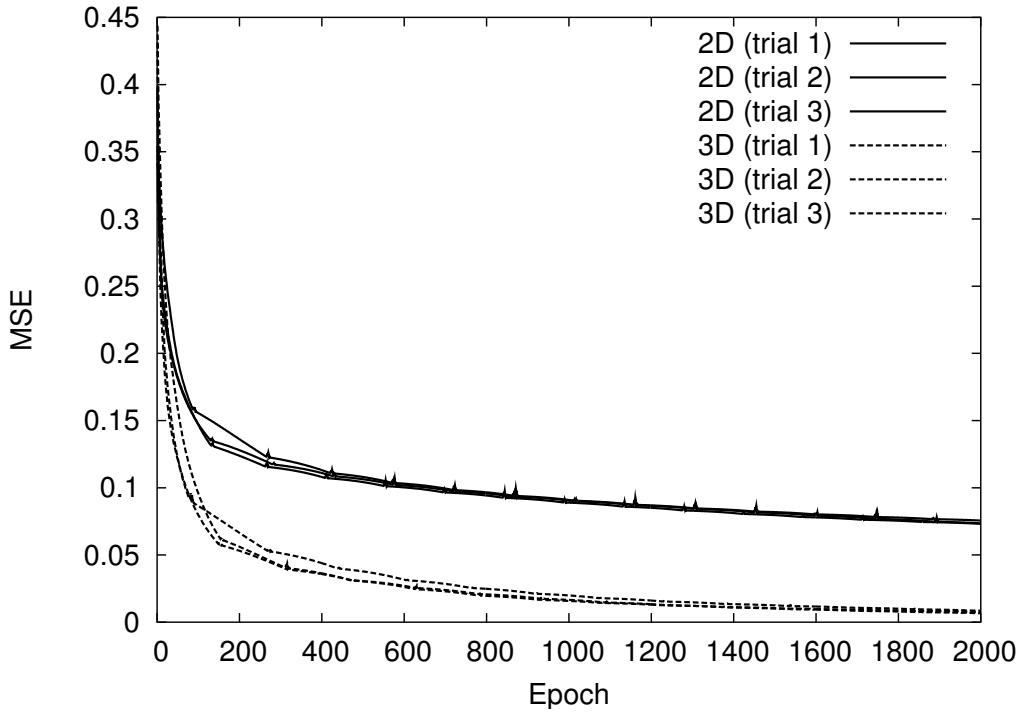
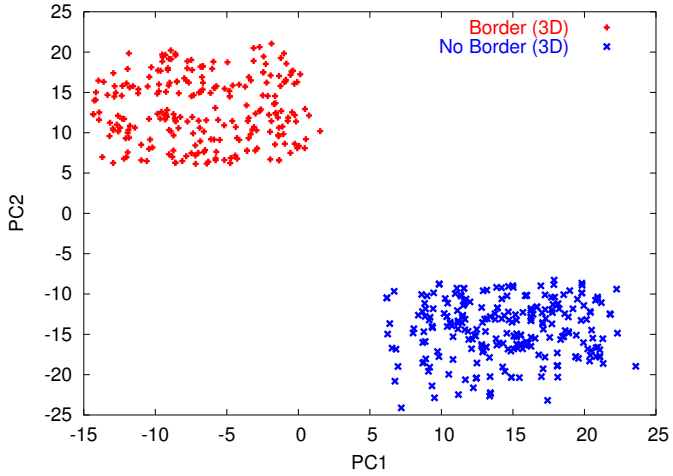


Fig. 25. Learning curves of the networks. The learning curves of the 2D-net and the 3D-net up to 2,000 epochs of training on texture set S_1 are shown. The 3D-net is more accurate and converges faster than the 2D-net (near 100 epochs), suggesting that the 3D preprocessed training set may be easier to learn than the 2D set.

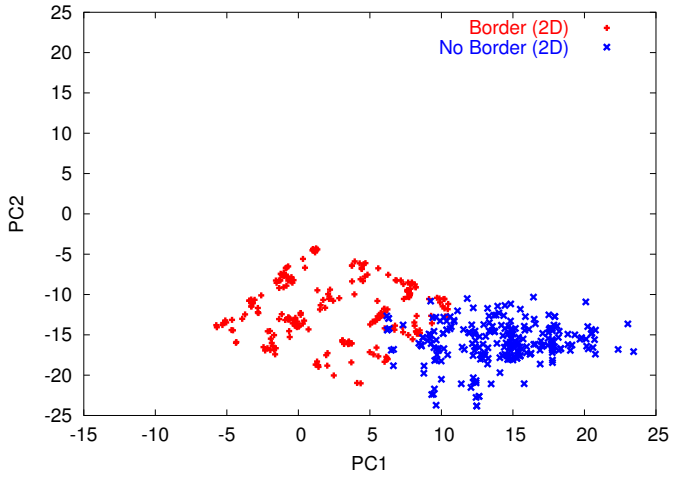
set acquired from texture images in S_1. The two dimensional PCA feature space of the two training sets are shown in Figure 26, where it is clearly shown that the 3D training set may be easier to separate than 2D set. An interesting observation is that the data sets without texture boundary for both the 2D and the 3D set have similar features (cluster in the lower right marked "X"). It seems that the invariance of Gabor filter bank responses (Gabor energy response, orientation response, and frequency response) in one half of each response profile (either the left side or the right side) in the 3D cases with texture boundary (figure 19) gives more separability than different responses in 2D cases. This invariance of Gabor filter bank responses are due to motion. Thus, it suggests that motion and 3D arrangement of textures can allow us to separate two texture surfaces more easily than with spatial features in 2D.

B. Generalization and Transfer

The 2D-net and the 3D-net trained with the texture set S_1 were tested on texture pairs from S_1, S_2 and S_3. (Note that for the texture set S_1, input vectors different from those in the training set were used.) For this testing, 500-sample sets of 2D and 500-sample of 3D per each texture set, which were prepared in the same manner as the training samples sets, were used. All six sample sets were presented to the 2D-net and the 3D-net. Two methods to compare the performance of the networks were used. First, I compared the misclassification rate, which is the percentage of misclassification. Misclassification rates were calculated for all 12 cases (= 6 sample sets $\times$ 2 networks): Figure 27 shows the result. The 3D-net outperformed the 2D-net in all cases, except for the sample set from S_1 with 2D preprocessing, which was similar to those used for training the 2D-net. It is also notable that the 3D-



(a) 3D Training set



(b) 2D Training set

Fig. 26. Principal Component Analysis (PCA) feature space representation of the training sets. (a) PCA representation of the 3D training set and (b) the 2D training set from texture images in S_1 shows that the 3D training set may be easier to separate than the 2D set. Also note that the data sets without texture boundary are similar for both 2D and 3D (lower right cluster marked "X" in both sets).

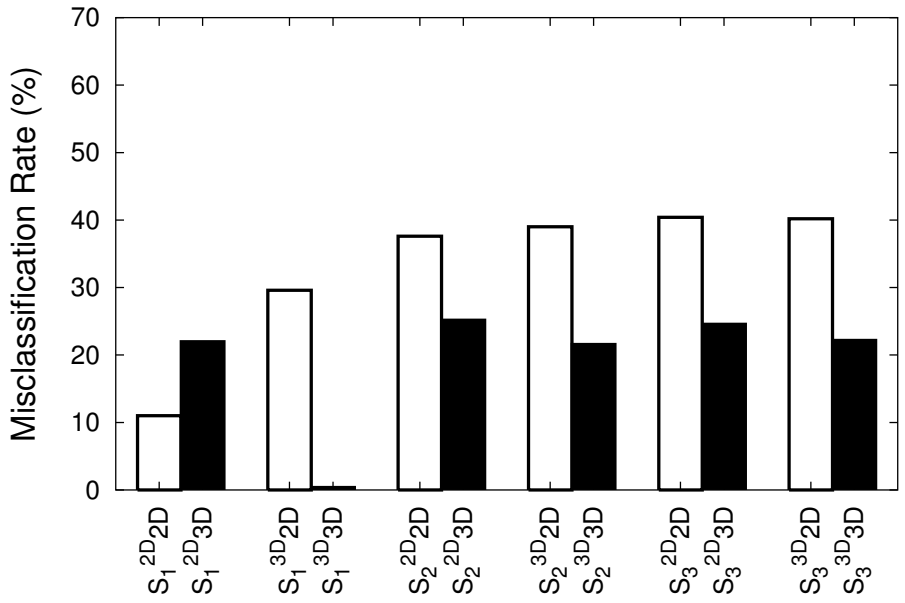


Fig. 27. Comparison of misclassification rates. The misclassification rates of the different test conditions are shown (white bars represent the 2D-net, and the black bars the 3D-net). The x-axis label $\mathrm{S}_i^{n\mathrm{D}}m\mathrm{D}$ indicates that input set i preprocessed in n-D was used as the test input, and the m-D network was used to measure the performance. In all cases, the 3D-net shows a lower misclassification rate compared to that of the 2D-net, except for $\mathrm{S}_1^{2\mathrm{D}}2\mathrm{D}$.

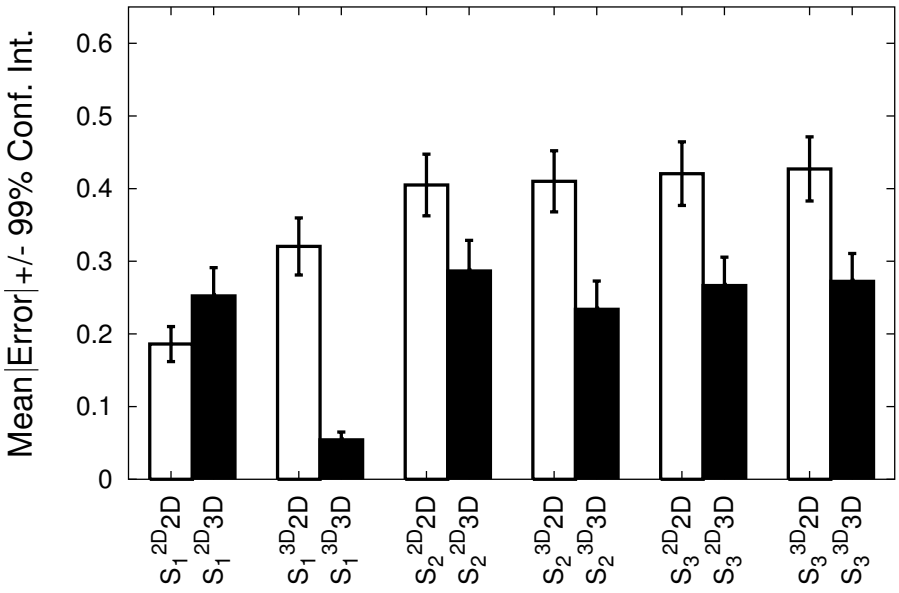


Fig. 28. Comparison of output errors. The mean error in the output vs. the target value in each trial and its 99% confidence interval (error bars) are shown for all test cases (white bars represent the 2D-net, and the black bars the 3D-net). In all cases, the differences between the 3D-net and the 2D-net are significant (t-test: $n = 500, p << 0.001$) (Note that for $S_1^{2D}, 2D < 3D$).

net outperformed the 2D-net on the sample sets from S_2 and S_3 prepared with 2D preprocessing (3rd and the 5th column in figure 27; these are basically a 2D texture segmentation problem), where one would normally expect the 2D-net to perform better because of the manner in which the input was prepared.

As another measure of performance, I compared the absolute error ($= |target - output|$) for each test case for the two networks. The results are shown in figure 28. The plot shows the mean absolute errors and their 99% confidence intervals. The results are comparable with those reported above. The 3D-net consistently outperformed the 2D-net for the sample sets from S_2 and S_3, and the differences were found to be statistically significant (t-test: $n = 500, p << 0.001$). However, the 2D-net out-

performed the 3D-net for sample set from S_1 (figure 28 first pair from the left). Again, since S_1 preprocessed in 2D was used for training the 2D-net, this was expected from the beginning.

As to why the 3D-net outperformed the 2D-net on sample sets from S_2 and S_3 (even for the 2D sets preprocessed in 2D) may be stated as follows based on careful observation of the data:

- 2D-net shows relatively high miss rate even on sample set from S_1, which were used for the training of 2D-net. This means that the 2D-net may not have been trained well enough to acquire the ability to classify more complex response patterns as that of S_2 and S_3. The reason for that may be that the 2D input representation was not adequate for the task.

- The 2D-net may have been trained adequately enough to classify the more complex response profiles in S_2 and S_3 differences of change in patterns of response profiles, On the other hand, the 3D-net seems to have been trained to classify whether there is significant difference between the average response of the one half and the average response of the other half. If there is a significant difference, then there is a boundary. Otherwise, no boundary. This is true with the S_2 and the S_3 cases after both 2D preprocessing and 3D preprocessing, whereas the difference is less significant in S_1 with 2D preprocessing. To quantitatively measure the difference, let us define the mean of one half of the response profile as,

$$\mu_{k,c,r} \quad = \quad \underset{x \in X}{\mathrm{AVG}}(E_k^c(x, y_c)), \text{for } X = \begin{cases} \{1, ..., 16\} & \text{if } r = \mathrm{L} \\ \{17, ..., 32\} & \text{if } r = \mathrm{R} \end{cases}, \qquad (4.1)$$

where $\mathrm{AVG}_x(\cdot)$ denotes a function that computes the mean value over x, k denotes the sample index, c denotes the class $\in \{\mathrm{B}, \mathrm{NB}\}$, where B corresponds

to 'boundary' and NB corresponds to 'no boundary', and E denotes the Gabor energy response matrix of the texture pair. From this we can define the degree of asymmetry in the mean values of the response profile as follows:

$$\mu_{\mathrm{B}} = \underset{1\le k\le 500}{\mathrm{AVG}}(|\mu_{k,\mathrm{B,L}} - \mu_{k,\mathrm{B,R}}|), \tag{4.2}$$

$$\mu_{\mathrm{NB}} = \underset{1\le k\le 500}{\mathrm{AVG}}(|\mu_{k,\mathrm{NB,L}} - \mu_{k,\mathrm{NB,R}}|), \tag{4.3}$$

$$\mu = |\mu_{\mathrm{B}} - \mu_{\mathrm{NB}}|. \tag{4.4}$$

Table II shows μ for each case. Note that μ of the 2D preprocessed S_1 (the value marked *) is significantly smaller (0.003182) than all the rest($n = 500, p < 0.006$).

TABLE II. The degree of asymmetry (μ) in the Gabor energy profiles.

	S_1	S_2	S_3
2D Preprocessing	0.003182*	0.014493	0.070150
3D Preprocessing	0.022924	0.017543	0.031973

Another point that I noticed from the results is the relatively poor performance on texture sample sets of S_2, even though the texture pairs from S_2 are highly separable and the texture elements that constitute the texture image in S_2 are simple enough, which suggests that only local feature of difference in orientation and spatial frequency is not enough to segregate textures in S_2, for instance, a texture pair that consists of a texture composed of 'L's and one with 'X's. In fact, Bergen and Adelson [6] showed such a pair can be discriminated by using size-tuned mechanisms, which suggests that filters with different sizes may be necessary. On the other hand, Julesz and Krose [52] viewed that the pair can be segmented even without multiple filter

sizes if textons are extracted, e.g., the existence or absence of *crossing* in this case ('X' or 'L', respectively).

These results suggest that (1) developing a texture segmentation function in 3D can be easier than in 2D, and (2) the ability to segment texture in 3D may transfer very well into solving texture segmentation problems in 2D.

CHAPTER V

DISCUSSION

This chapter discusses the main contribution of the current study with respect to relevant works by other researchers (section A), followed by discussion on issues related to the current experimental approaches (section B), and future directions of this work (section C).

A. Main Contribution

Since the early works of Julesz[4] and Beck[2] on texture perception, many studies have been conducted to understand the mechanisms of the human visual system underlying texture segmentation and texture boundary detection in both psychophysical and pattern recognition research. In most cases, their main concern has been about the texture perception ability of humans in 2D. In this thesis, I proposed an additional approach to the problem of texture perception, with a focus on boundary detection rather than texture classification. First, I demonstrated that texture boundary detection in 3D is easier than in 2D. I also showed that the learned ability to find texture boundary in 3D can easily be transferred to texture boundary detection in 2D. Based on these results, my careful observation is that the outstanding ability of 2D texture boundary detection of the human visual system may have been derived from an analogous ability in 3D.

The results of this thesis allow us to question one common belief that many other texture boundary detection studies share. In the current view, intermediate visual processing such as texture perception, visual search and motion processing do not require object (in the context of this thesis, "3D") knowledge, and thus perform rapidly; and texture perception is understood in terms of features and filtering, so the

performance is determined by differences in the response profiles of receptive fields in low-level visual processing. Nakayama and his colleagues advanced a view which is similar to that presented in this thesis [23][22]. They proposed that surface representation forms a critical intermediate stage of vision between lower level and higher level vision. In Nakayama's alternative view on intermediate visual processing, visual surface representation is necessary before other visual tasks such as texture perception, visual search, and motion perception can be accomplished (figure 2 in Chapter I and figure 29 below). In his alternative view, the most important characteristics of a world defined by surfaces is that it is in 3D. Such an observation is in line with the results of this thesis indicating that 3D performance can easily transfer into 2D tasks.

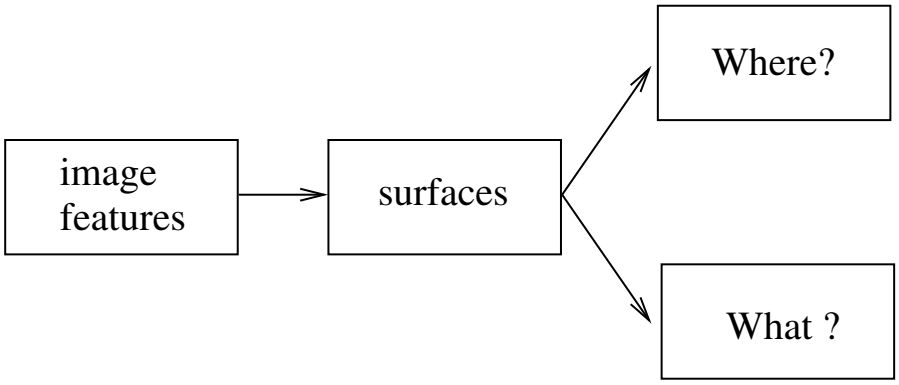


Fig. 29. Presumed placement of surface representation in relation to lower level and higher level visual functions. Adapted from [22].

Also, the experiments in this thesis share a common approach as that of *active vision*, which empathizes that vision cannot be performed in isolation and that vision system can benefit from continuously interacting with the environment [53]. This concept was employed to solve many computer vision problems, such as stereo matching, with low complexity algorithm by using controlled sensor motion and interaction with environment [54] (see, e.g., [55] [56] [57]). In the experiments of this thesis, self-motion of observer was simulated to acquire input sets for the 3D case.

The experiments showed that the 3D case significantly outperformed the 2D case that has no motion involved. This results are in line with those in active vision, where active vision system outperforms passive vision system, and generally leads to a simpler solution. In this thesis, even though the 3D-net did not have any goal directedness as most other active vision systems do, the experimental results showed the usefulness of motion and interaction with the environment, which supports the main argument for active vision.

B. Limitations of the Model

The main goal of this thesis was to understand the nature of textures, and from that emerged the importance of 3D cues in understanding the texture detection mechanism in vision systems. To emulate 3D depth, motion cues were employed. The experimental model in this thesis is based on an assumption that moving features within a visual image remain projectively attached to fixed locations on the surface of an object, but various phenomena argues against this assumption [58], which means that a retinal position does not remain fixed on the surface of an object when it is observed during motion. Although this violation may weaken my argument as well as many current models regarding shape from motion in other studies, the assumption about fixation of surface are easily interpretable for human observers and commonly assumed in most shape from motion algorithms [59] [58], thus it may be reasonable.

One concern is that using motion may have given an unfair advantage to the 3D-net. That is, additional information may have become available to the 3D-net; some form of temporal information that the 2D inputs do not have. However, we should note that the 2D-net has additional spatial information, which the 3D-net does not have, so eventually these two relative advantages may have canceled out.

In top-down perceptual modeling approaches such as Gestalt theories of visual perception, texture perception task is considered to require global information as other visual perception tasks do and there are many experimental results that support these approaches [60]. In this thesis, I did not employ any top-down information such as Gestalt properties or object information, but it may be more appropriate to take Gestalt approaches than neural modeling approaches as in this thesis. However, the purpose of this thesis was to show that a seemingly difficult task that may potentially require at least some amount of object knowledge can be easily solved with motion and 3D-cues, thus it may be enough to employ neural modeling approach for this purpose.

In the experiments of this thesis, for the textures with a boundary, the input sample sets were acquired exactly from the center of the boundary (a fixed boundary position $x_c = 16$, the measuring range $1 \leq x \leq 32$). One may think that x_c should be also variable, and that I made the problem so simple that this model cannot be used in a computer vision system detecting texture boundary. However, it can be resolved by carefully selecting a threshold (0.5 in this thesis). In addition to this, when considering the purpose of the experiments, this problem may not change the conclusion of this thesis, because the condition was the same for in the 2D case as well.

One conern may be that the results would not hold if the Gabor filter parameters such as filter size are changed. In fact, the use of multiresolution approach was shown to affect the performace of texture processing [49]. However, since the change of parameter values should be applied to both the 3D case and the 2D case uniformly, the results are not expected to change.

Lastly, even though I applied the heuristics for making backpropagation converge faster, these may not have been enough to get optimally trained networks. There is

also a possibility that what was applied in this thesis can be advantageous to either 2D network or 3D network but not both. This point will be discussed more in the following section.

C. Future Work

In the experiments of this thesis, only motion cues were employed to emulate 3D depth. One immediate future direction, therefore, is to extend the current approach to utilize binocular cues as well as monocular cues. Binocular disparity is another relevant cue for the perceptual analysis of 3D structure. The analysis of *structure from motion* and *shape from binocular disparity* is similar in that it generally requires two distinct stages of processing. The first of these stages is to compute an optical flow field from changing patterns of light on the retina and the next is to incorporate these results to estimate the 3D structure of an observed scene [58]. Thus, a similar computational/experimental method as used in this thesis can be employed to emulate binocular disparity. One of the benefits that can be attained from using binocular disparity is its relative computational simplicity.

Another direction for the future work would be to use more powerful learning methods such as support vector machines (SVM). While neural networks usually require problem specific heuristics to optimize the performance, SVMs use a systematic approach and can be easily applied in different problem domains. SVMs also show better generalization performance. The nature of neural network models is that the longer they get trained the better they learn. In fact, there is no precise way to tell how long the training process is enough. Thus, it would be worth employing SVM for testing the hypothesis proposed in this thesis, since the current neural network training scheme in this study may not be optimal for the given task. Furthermore,

the current model can be more easily applied to computer vision system when SVMs are used along with other 3D depth cues like binocular disparity.

Lastly, the sample set for training neural networks were acquired from simple textures that have only four orientations and two spatial frequencies. In future experiments, the number of sample texture for training can be increased so that the trained network can perform better on any given natural texture image.

CHAPTER VI

CONCLUSION

I began this thesis with a simple question regarding the nature of textures: *What are textures, and why did the ability to discriminate texture evolve or develop?* The tentative answer was that textures naturally define distinct physical surfaces, and thus the ability to segment texture in 2D may have grown out of the ability to distinguish surfaces in 3D. To test this insight, I compared texture boundary detection performance of two neural networks trained on textures arranged in 2D and in 3D. The results revealed that texture boundary detection in 3D is easier to learn than in 2D in terms of speed and accuracy, and that the network trained in 3D easily solved the 2D problem as well, but not the other way around. Based on these results, I carefully conclude that the human ability to segment texture in 2D may have originated from a module evolved to handle 3D tasks. The results from this thesis can help us take a fresh look at the problem of texture segmentation, and allow us to design more powerful computer vision algorithms in the future.

REFERENCES

[1] M. Tuceryan, "Texture analysis," in *The Handbook of Pattern Recognition and Computer Vision*, 2nd ed. Singapore: World Scientific Publishing Co., 1998, pp. 207–248.

[2] J. Beck, "Effect of orientation and shape similarity on perceptual grouping," *Perception and Psychophysics*, vol. 1, pp. 300–302, 1966.

[3] ——, "Textural segmentation, second-order statistics and textural elements," *Biological Cybernetics*, vol. 48, pp. 125–130, 1983.

[4] B. Julesz, "Texture and visual perception," *Scientific American*, vol. 212, pp. 38–48, Feb 1965.

[5] ——, "Textons, the elements of texture perception, and their interactions," *Nature*, vol. 290, pp. 91–97, 1981.

[6] J. R. Bergen and E. H. Adelson, "Early vision and texture perception," *Nature*, vol. 333, pp. 363–364, May 1988.

[7] A. Thielscher and H. Neumann, "Neural mechanisms of cortico-cortical interaction in texture boundary detection: a modeling approach," *Neuroscience.*, vol. 122, no. 4, pp. 921–39, 2003.

[8] M. Landy and N. Graham, "Visual perception of texture," in *The Visual Neurosciences*. Cambridge, MA: MIT Press, 2004, pp. 1106–1118.

[9] B. Julesz and J. Bergen, "Texton theory of preattentive vision and texture perception," *Journal of the Optical Society of America*, vol. 72, pp. 1756–1757, 1982.

[10] L. O. C. Chubb and A. Derrington, "Second-order processes in vision: introduction," *Optical Society of America*, vol. 18, no. 9, pp. 2175–2178, September 2001.

[11] J. Malik and P. Perona, "Preattentive texture discrimination with early vision mechanisms," *Journal of Optical Society of America A*, pp. 923–932, 1990.

[12] J. R. Bergen and M. S. Landy, "Computational modeling of visual texture segregation," in *Computational Models of Visual Processing*, M. S. Landy and J. A. Movshon, Eds. Cambridge, MA: MIT Press, 1991, pp. 253–272.

[13] N. Graham, "Nonlinearities in texture segregation," in *Higher-order Processing in the Visual System Proceedings from The Ciba Foundation Symposium*, London, England, Oct. 1993, pp. 184–185.

[14] Z. Li, "Pre-attentive segmentation in the primary visual cortex," *Spatial Vision*, vol. 13, no. 1, pp. 25–50, 2000.

[15] ——, "A saliency map in primary visual cortex," *Trends in Cognitive Science*, vol. 6, pp. 9–16, January 2002.

[16] M. S. A.G. Leventhal, Y.C. Wang and Y. Zhou, "Neural correlates of boundary perception," *Visual Neuroscience*, vol. 15, pp. 1107–1118, 1998.

[17] S. Grossberg and E. Mingolla, "Neural dynamics of surface perception: boundary webs, illuminants, and shape-from-shading," *Computer Vision, Graphics, and Image Processing*, vol. 37, no. 1, pp. 116–165, January 1987.

[18] A. S. N. Graham and C. Venkatesan, "Spatial-frequency- and orientation-selectivity of simple and complex channels in region segregation," *Vision Research*, vol. 33, no. 14, pp. 1883–1911, 1993.

[19] D. M. T.S. Lee and A. Yuille, "Texture segmentation by minimizing vector-valued energy functionals: The coupled-membrane model," in *European Conference on Computer Vision*, 1992, pp. 165–173.

[20] M. Tuceryan, "Moment based texture segmentation," *Pattern Recognition Letters*, vol. 15, pp. 659–668, 1994.

[21] J. Malik, S. Belongie, J. Shi, and T. K. Leung, "Textons, contours and regions: cue integration in image segmentation," in *ICCV99*, 1999, pp. 918–925.

[22] K. Nakayama, Z. J. He, and S. Shimojo, "Visual surface representation: A critical link between lower-level and higher-level vision," in *Visual Cognition*, 2nd ed. Cambridge, MA: MIT Press, 1995, pp. 1–70.

[23] Z. J. He and K. Nakayama, "Perceiving textures: beyond filtering," *Vision Research*, vol. 34, no. 2, pp. 151–62, 1994.

[24] J. D. Daugman, "Two dimensional spectral analysis of cortical receptive field profiles," *Vision Research*, vol. 20, pp. 847–856, 1980.

[25] T. Bench-Capon, "Neural networks and open texture," in *Proceedings of the Fourth International Conference on AI and Law*, Amsterdam, June 1993, pp. 292– 297.

[26] P. Raghu and B. Yegnanarayana, "Supervised texture classification using a probabilistic neural network and constraint satisfaction model," *IEEE Transactions on Neural Networks*, vol. 9, pp. 516–522, 1998.

[27] A. Silliton and H. Jones, "Feedback systems in visual processing," in *The Visual Neurosciences*, L. Chalupa and J. Werner, Eds. Cambridge, MA: The MIT Press, 2004, vol. 1.

[28] P. Hoyer, "Probabilistic models of early vision," Ph.D. dissertation, Helsinki University of Technology, Helsinki, Finland, 2002.

[29] G. Montgomery, "Seeing, hearing and smelling the world," 1997, howard Hughes Medical Institute, URL http://www.hhmi.org/senses.

[30] M. Schmolesky, "The primary visual cortex," 2003, in 'Webvision: The organization of the retina and visual system,' edited by Kolb et al. URL: http://webvision.med.utah.edu/.

[31] R. V. P. Janssen and G. Orban, "Macaque inferior temporal neurons are selective for disparity-defined three dimensional shapes," in *Proc. Natl. Acad. Sci. USA*, vol. 96, 1999, pp. 8217–22.

[32] J. Maunsell and W. Newsome, "Visual processing in monkey extrastriate cortex," *Annual Review of Neuroscience*, vol. 10, pp. 363–401, 1987.

[33] L. Ungerleider and J. Haxby, "What and where in the human brain," *Current Opinons on Neurobiology*, vol. 4, pp. 157–65, 1994.

[34] S. Sunaert, "Functional magnetic resonance imaging studies of visual motion processing in the human brain," Ph.D. dissertation, Leuven University, Netherlands, 2001.

[35] V. Casagrande and T. Norton, "Lateral geniculate nucleus: A review of its physiology and function," in *The Neural Basis of Visual Function.* Boca Raton, FL: CRC Press, 1989, pp. 41–84.

[36] E. Kaplan, "The receptive field structure of retinal ganglion cells in cat and monkey," in *The Neural Basis of Visual Function.* Boca Raton, FL: CRC Press, 1989, pp. 10–40.

[37] J. Daugman, "Two-dimensional spectral analysis of cortical receptive field profiles," *Vision Research*, vol. 20, pp. 847–856, 1980.

[38] M. Livingston and D. Hubel, "Anatomy and physiology of a color system in the primate visual cortex," *Journal of Neuroscience*, vol. 4, pp. 309–356, 1984.

[39] N. Petkov, "Biologically motivated computationally intensive approaches to image pattern recognition," *Future Generation Computer Systems*, vol. 11, no. 4-5, pp. 451–465, 1995.

[40] P. Kruizinga and N. Petkov, "A computational model of periodic-pattern-selective cells," in *International Work-Conference on Artificial Neural Networks*, 1995, pp. 90–99. [Online]. Available: citeseer.nj.nec.com/kruizinga95computational.html

[41] H. A. Mallot, *Computational Vision*. Cambridge, MA: The MIT Press, 2000.

[42] S. Marcelja, "Mathematical description of the response of simple cortical cells," *Journal of Optical Society of America, A*, vol. 70, no. 11, pp. 1297–1300, 1980.

[43] P. B. J. J. Kulikowski, "Fourier analysis and spatial representation in the visual cortex," *Experientia*, vol. 37, pp. 160–163, 1981.

[44] J. P. Jones and L. A. Palmer, "An evaluation of the two-dimensional gabor filter model of simple receptive fields in cat striate cortex," *Journal of Neurophysiology*, vol. 58, no. 6, pp. 1223–1258, 1987.

[45] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Englewood Cliffs, NJ: Prentice Hall, 1998.

[46] H. Demuth and M. Beal, *Neural Network Toolbox for Use with Matlab*, 3rd ed. Natick, MA: The Mathworks Inc., 1992.

[47] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.

[48] R. A. Jacobs, "Increased rates of convergence through learning rate adaptation," *Neural Networks*, vol. 1, pp. 295–207, 1988.

[49] B. J. Krose, "A description of visual structure," Ph.D. dissertation, Delft University of Technology, Delft, Netherlands, 1986.

[50] P. Brodatz, *Textures: A Photographic Album for Artists and Designers*. New York: Dover Publications, 1966.

[51] Y. LeCun, "Efficient learning and second-order methods," in *A tutorial in Neural Information Processing Systems*, Denver, 1993, pp. 76–77.

[52] B. Julesz and B. Krose, "Features and spatial filters," *Nature*, vol. 333, pp. 302–303, May 1988.

[53] I. W. J.Y. Aloimonos and A. Bandopadhay, "Active vision," *International Journal on Computer Vision*, pp. 333–356, 1987.

[54] J. L. Crowley, "Active computer vision tutorial," 1995, uRL: www-prima.imag.fr.

[55] J. R. Bajcsy, "Active perception," in *Proceedings of the IEEE, Special issue on Computer Vision*, vol. 76, August 1988, pp. 996–1006.

[56] D. Ballard, "Animate vision," *Artificial Intelligence*, vol. 48, pp. 1–27, 1991.

[57] C. Brown, "Prediction and cooperation in gaze control," *Biological Cybernetics*, vol. 63, pp. 61–70, 1990.

[58] J. T. Todd, "Perception of three-dimensional structure," in *The Handbook of Brain Theory and Neural Networks*, 1st ed. Cambridge, MA: The MIT Press, 2002, pp. 868–871.

[59] J. F. Norman and J. T. Todd, "The perception of rigid motion in depth from the optical deformations of shadows and occlusion boundaries," *J. Exp. Psychol. Hum. Percpt. Perform.*, vol. 20, pp. 343–356, 1994.

[60] S. Lehar, "Gestalt isomorphism and the primacy of subjective conscious experience: a Gestalt bubble model," *Behavioral and Brain Sciences*, vol. 26, no. 4, pp. 375–444, 2002.

VITA

Se Jong Oh was born in Seoul, South Korea on October 9th, 1971. After graduating from the Republic of Korea Air Force Academy, he worked for the Korean air force as an officer for eight years. In the fall of 2002, he entered the Department of Computer Science at Texas A&M University, College Station, Texas to pursue the Master of Science degree in computer science.

Permanent Address:

110-501 Yuhyun Shindonga
Poongmu-dong, Kimpo
Gyung-gi province 415-755
South Korea

Email Address:

sejong.oh@gmail.com

The typist for this thesis was Se Jong Oh.