Texture Segmentation in 2D vs. 3D: Did 3D Developmentally Precede 2D?

Sejong Oh and Yoonsuck Choe
Department of Computer Science
Texas A&M University
3112 TAMU, College Station, TX 77843-3112
{sejong,choe}@tamu.edu

Abstract

Texture boundary detection (or segmentation) is an important capability in human vision. 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, thus, we can hypothesize that 2D texture segmentation may be an outgrowth of the ability to discriminate surfaces in 3D. In this paper, we 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 our expectation. Furthermore, we have shown that the learned ability to segment texture in 3D transfers well into 2D texture segmentation, bolstering our initial hypothesis, and providing a possible explanation for the developmental origin of 2D texture segmentation function in human vision.*

1. Introduction

Detection of a tiger in the shrub 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 capability. Figure-ground separation is based on many different cues such as luminance, color, texture, etc. In case of the tiger in the jungle, texture plays a critical role. What are the visual processes that enable perceptual agents to separate figure from ground using texture cues? This intriguing question led many researchers in vision to investigate the mechanisms of texture perception.

Beck [2][3] and Julesz [4] conducted psychological experiments investigating the 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 [5]. Julesz also proposed the texton theory, in which textures are discriminated if they differ in the density of simple, local textural features, called textons [6]. Most models based on these observations lead to a feature-based theory, in which segmentation occurs when feature differences (such as difference in orientation) exist. Furthermore, 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 [7][8].

In the current studies of texture segmentation and boundary detection, texture is usually defined in 2D. However, an interesting hypothesis arises when we ask an important question regarding the nature of textures: *What are textures, and why did the ability to discriminate textures evolve or develop?* One possible answer to the question is that texture is that which defines physically distinct surfaces, belonging to different objects, and that texture segmentation function may have evolved out of the necessity to distinguish different surfaces. Human visual experience with textures can be, therefore, in most cases to use them as cues for surface perception, depth perception, and 3D structure perception. In fact, psychological experiments by Nakayama and He [9][10] showed that the visual system cannot ignore information regarding surface layout in texture discrimination and proposed that surface representation must actually precede perceptual functions such as texture perception (see the discussion section for more on this point).

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 parallax may provide unambiguous information about the surface. Thus, we can hypothesize that 3D surface perception could

have contributed in the formation of early texture segmentation processing capabilities in human vision. In this paper, through computational experiments using artificial neural networks, we investigate the relative difficulty of learning to discriminate texture boundaries in 2D vs. 3D arrangements of texture. We will also study whether the learned ability to segment texture in 3D can transfer into 2D. In the following, we will first describe in detail the methods we used to prepare the 2D and 3D texture inputs (Section 2.1), and the procedure we followed to train multilayer perceptrons to discriminate texture boundaries (Section 2.2). Next, we will present our main results and interpretations (Section 3), followed by discussion (Section 4) and conclusion (Section 5).

2. Methods

To test our hypothesis proposed in the introduction, we need to conduct texture discrimination experiments with 2D and 3D arrangements of texture. In this section, we will describe in detail how we prepared the two different arrangements (Section 2.1), and explain how we trained two standard multi-layer perceptrons to discriminate these texture arrangements (Section 2.2). We trained two separate networks that are identical in architecture, one with input prepared in a 2D arrangement (we will refer to this network as the *2D-net*), and the other in a 3D arrangement (the *3D-net*).

2.1. Input preparation

We prepared three sets of texture stimuli S_1, S_2, and S_3. Textures in S_1 were simple artificial texture images (oriented bars of orientation 0, $\frac{\pi}{4}$, $\frac{\pi}{2}$, or $\frac{3\pi}{4}$ at two different spatial frequencies); those in S_2 were more complex texture images such as crosses and circles, adapted from Krose [11] and Julesz [12]; and those in S_3 were real texture images from the widely used Brodatz texture collection [13] (Figure 1). For the training of the 2D-net and the 3D-net, the eight simple texture stimuli in 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, we used Gabor filters. Previous results have shown that Gabor filters closely resemble experimentally measured receptive fields in the visual cortex [14] and they have been widely used to model the response of visual cortical neurons. A number of texture analysis studies also used oriented Gabor filters or difference of Gaussian (DOG) filters to extract local image features [15][16][17].

We 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 [18]:

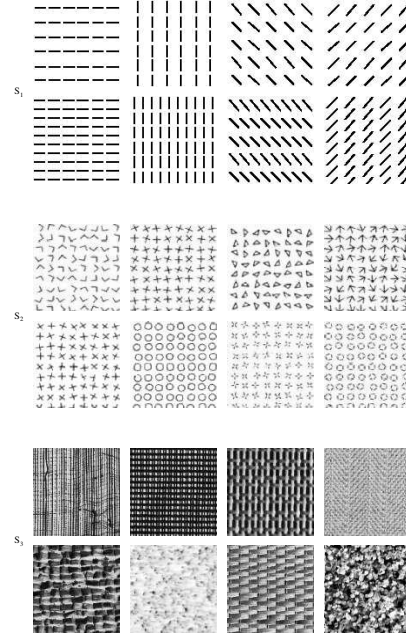$$G_{\theta,\phi,\sigma,\omega}(x,y) = \exp^{-\frac{x'^2+y'^2}{2\sigma^2}} \cos(2\pi\omega x' + \phi), \quad (1)$$

Figure 1. Texture stimuli. Three texture sets S_1, S_2, and S_3 are shown from the top to the bottom row.

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:

$$x' = x\cos(\theta) + y\sin(\theta) \quad (2)$$
$$y' = -x\sin(\theta) + y\cos(\theta). \quad (3)$$

For simplicity, only four different orientations (0, $\frac{\pi}{4}$, $\frac{\pi}{2}$, $\frac{3\pi}{4}$) were used for θ. (Below, we will refer to $G_{\theta,\phi,\sigma,\omega}$ as simply G.) To adequately sample the spatial-frequency features of input stimuli, three frequency ranges (1 to 3 cycles/degree) were used for ω. The size of 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 2. To get the Gabor response matrix C_i, a gray-level intensity matrix I was obtained from the images randomly selected from S_1 and convolved with the filter bank G_i:

$$C_i = I * G_i, \quad (4)$$

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 [19]. Thus, half-wave rectification is commonly used to provide a nonlinear response characteristic following linear filtering. However, in our experiments, full-wave rectification was used as in [20], which is similar to half-wave rec-
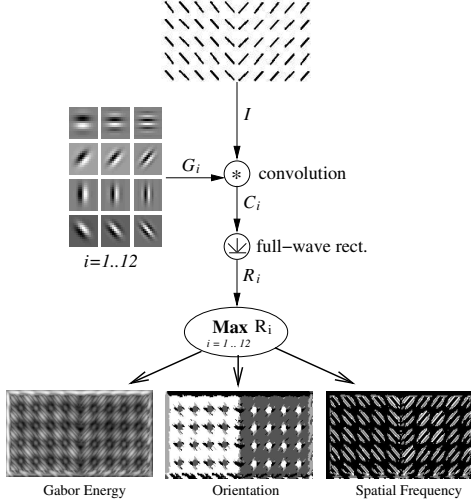
Figure 2. Gabor filter bank. The process used to generate two orientation response matrices is shown. The texture I is first convolved with the Gabor filters G_i (for $i = 1..12$), and the resulting responses are passed through a full-wave rectifier resulting in R_i. Finally, we get the Gabor energy matrix $E(x, y)$, Orientation response matrix $O(x, y)$, and Frequency response matrix $F(x, y)$.

tification, but is simpler to implement. Full-wave rectification is equivalent to summing the outputs of the two corresponding half-wave rectification channels (see, e.g. Bergen and Adelson [21] [19]). The final full-wave rectified Gabor feature response matrix is calculated as

$$R_i = |C_i|, \qquad (5)$$

for $i = 1..12$. For each sample texture pair, we acquired three Gabor response matrices (or maps), which were the Gabor energy response matrix E, the orientation response matrix O, and the frequency response matrix F. 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 the given location. The orientation response matrix O had orientation index $(1 \leq O(x, y) \leq 4)$ of the filter that had 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 had 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 2 shows the Gabor filter bank and the three response matrices E, O, and F of the given texture pair.

(a) Texture with boundary (b) Response to (a)

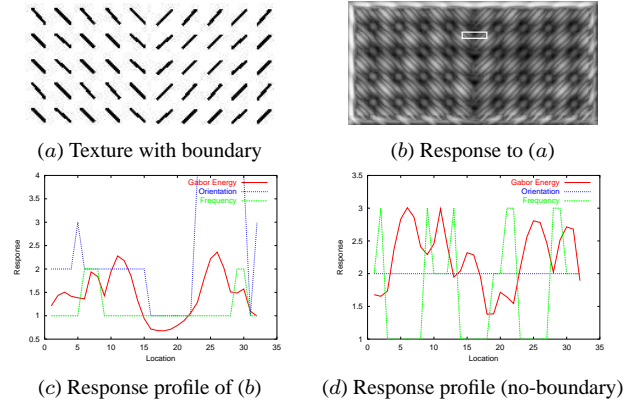(c) Response profile of (b) (d) Response profile (no-boundary)

Figure 3. Generating the 2D input set (2D pre-processing). The procedure used to generate the 2D training data is shown. (a) Input with a texture boundary. (b) Orientation response calculated from (a). Only the E matrix is shown. (c) The response profile from the 32-pixel wide area marked with a white rectangle in (b). The three curves represent the profiles in the E, O, and F matrices. (d) A similarly calculated response profile in a different input texture, for an area without a texture boundary (note the periodic peaks).

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 2). 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 the 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 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 3b), 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 has 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 $(\xi_k^{2D}, \zeta_k^{2D})$ in each class, either boundary or no boundary, was balanced so that each class is equally represented. Figure 3c shows an example vector ξ_k^{2D} when there was a texture boundary, and figure 3d 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 4. 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 5a) was convolved with the oriented Gabor filter bank followed by full-wave rectification as in the 2D preprocessing case (figure 5b). To generate a single training input-target pair $(\xi_k^{3D}, \zeta_k^{3D})$ 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, and y_c was selected randomly for each new input-target pair but remained the same within the same pair (the white square in figure 5b shows an example). Figure 5c shows an example of such a vector ξ_k^{3D} (note that the x-axis represents time, unlike in the 2D case where it is space) for a case containing a texture boundary, and figure 5d for a case without a boundary. The target value ζ_k^{3D} of the input-target pair (ξ_k^{3D}, ζ_k^{3D}) was set in a similar manner as in the 2D case, either to 0 (no boundary) or to 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 was on top of the texture on the right side with self-motion of observer from right to left. In the second configuration, the texture on the right was on top of the texture on the left side with self-motion of observer from left to right. For an unbiased 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 target value of 1, and the same number of samples with target value of 0. This resulted in 4,800 input-target samples for each case (*1 ≤ k ≤ 4,800*). These 4,800 input-target samples from each training set were then randomly ordered during training.

2.2. Training the texture segmentation networks

We used standard multilayered perceptrons (MLPs) to perform texture boundary detection. The networks (2D-net and 3D-net), which consisted of two layers including 96 in-
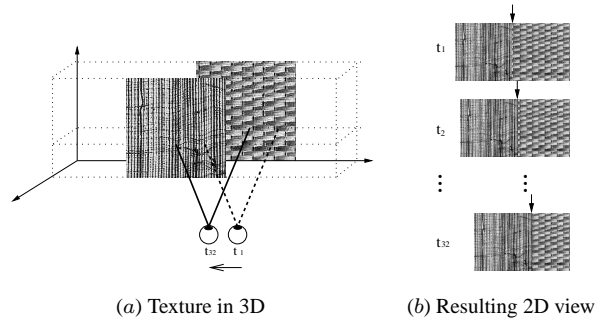

(a) Texture in 3D   (b) Resulting 2D view

Figure 4. 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.

put units, 16 hidden units and 1 output unit, were trained for 2,000 epochs each using standard backpropagation[1]. 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 $f(v) = a \tanh(bv)$, where $a = 1.7159$ and $b = \frac{2}{3}$ respectively (following [22]). For the activation function of the output layer that consisted of one unit, radial basis function (RBF) was used. The use of the radial basis function in standard MLP is not common: 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 the 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 us to use the radial basis function, which has a Gaussian profile. 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 shown here). For the training, the input vectors were drawn from the texture set S_1. Backpropagation with momentum and adaptive learning rate was applied to train the weights.

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

---

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

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

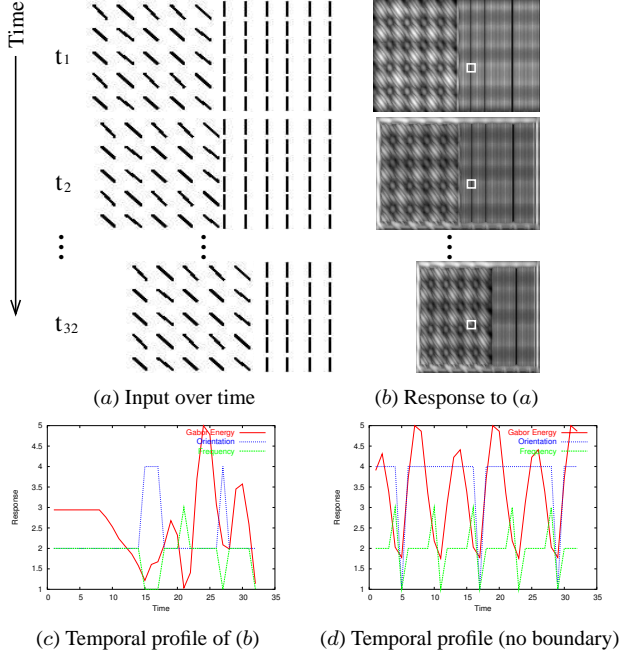(c) Temporal profile of (b)   (d) Temporal profile (no boundary)

Figure 5. Generating 3D input set through motion (3D preprocessing). (a) Texture pair images resulting from simulated motion: I'_j for each $j = t_1..t_{32}$. (b) The response matrix of the texture pair: R^{3D}_{ij}. (c) Response profile obtained over time near the boundary of two different texture images (marked by the small squares in c). (d) A similarly measured response profile collected over time, using a different input texture, near a location without a texture boundary (note the periodic peaks).

the experiment can be directly compared. Each training set consisted of 280 examples, drawn from S_1 and processed by the preprocessing procedure. The training process continued for 1,000 epochs. The MLPs with other combination of parameters failed to converge. Based on these preliminary training tests, we chose the learning parameters as follows: learning rate $\eta = 0.01$, and momentum constant $\alpha = 0.5$.

We also applied standard heuristics to speed up and stabilize the convergence of the networks. 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}:

$$\eta_n = \eta_{n-1}\eta_{\text{inc}}, \qquad (6)$$

where n is the epoch. If MSE increased by more than 1.04, the learning rate was adjusted by the factor of η_{dec}:

$$\eta_n = \eta_{n-1}\eta_{\text{dec}}. \qquad (7)$$

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 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. 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. The results from these experiments will be presented in the following section.

3. Experiments and Results

We compared the performance of the two trained networks (2D-net and 3D-net), and also compared the performance of the two networks over novel texture images that were not used in training the networks.

3.1. Speed of convergence and accuracy on the training set

Figure 6 shows the 3 best learning curves of each network out of 10 trials during the 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 6 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 the network. These results indicate that the 3D-net is easier to train than the 2D-net. In other words, texture arrangements represented in 3D may be easier to segment than those in 2D. The misclassification rate, which was computed by using a threshold of 0.5 on the output response, 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.
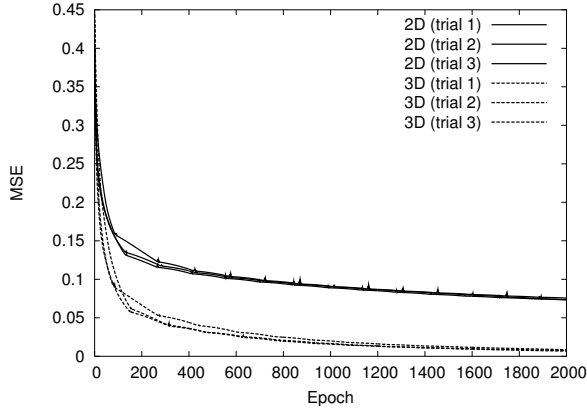
Figure 6. Learning curves of the networks. The learning curves of the 2D-net and the 3D-net up to 2,000 epochs of training on texture set S_1 are shown. The 3D-net is more accurate and converges faster than the 2D-net (near 100 epochs), suggesting that the 3D preprocessed training set may be easier to learn than the 2D set.

Figure 7. 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 $S_i^{nD}mD$ 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 $S_1^{2D}2D$.

3.2. 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, we compared the misclassification rate, which is the percentage of misclassification. Misclassification rates were calculated for all 12 cases (= 6 sample sets × 2 networks): Figure 7 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-net outperformed the 2D-net on the sample sets from S_2 and S_3 prepared with 2D preprocessing (third and the fifth pair in figure 7; 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, we compared the absolute error (= $|target - output|$) for each test case for the two networks. The results are shown in figure 8. The plot shows the mean absolute errors and their 99% confidence intervals. The results are comparable to the misclassification rate results reported above. The 3D-net consistently
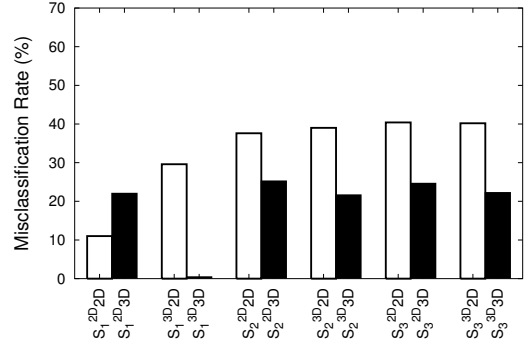
outperformed the 2D-net for the sample sets from S_2 and S_3, and the differences were found to be statistically significant (t-test: $n = 500, p << 0.001$). However, the 2D-net outperformed the 3D-net for sample set from S_1 (figure 8 first pair from the left). Again, since S_1 preprocessed in 2D was used for training the 2D-net, this was expected from the beginning.

4. Discussion

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 boundary detection in both psychophysical research and in pattern recognition research. In most cases their main concerns have been about the texture perception ability of human in 2D. The work presented in this paper suggests an alternative approach to the problem of texture perception, with a focus on boundary detection. First, we demonstrated that texture boundary detection in 3D is easier than in 2D. We 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, our 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.

Our preliminary results allow us to challenge one common belief that many other texture boundary detection stud-
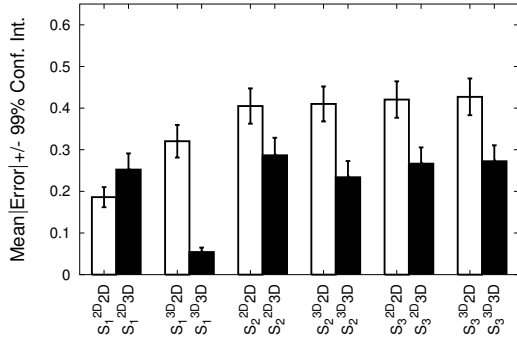
Figure 8. Comparison of output errors. The mean error in the output vs. the target value in each trial and its 99% confidence interval (error bars) are shown for all test cases (white bars represent the 2D-net, and the black bars the 3D-net). In all cases, the differences between the 3D-net and the 2D-net are significant (t-test: $n = 500, p << 0.001$). Note that for $S_1^{2D}, 2D < 3D$.

(a) Traditional view

(b) An alternative view

Figure 9. Two views of intermediate visual processing. *(a)* In the traditional view, texture perception, visual search, motion perception depend on feature processing in early cortical areas. *(b)* In an alternative view, surface representation must precede intermediate visual tasks [10]. (Adapted from [10].)

ies share. In this view, intermediate visual processing such as texture perception, visual search and motion process do not require object (in our context, "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. A similar point as ours was advanced by Nakayama and his colleagues [9][10]. 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 9). Such an observation is in line with our results indicating that 3D performance can easily transfer into a 2D task. (Note that there is yet another possibility, where all of these visual tasks are processed concurrently at the same stage, but we do not have enough evidence to either accept or reject such a proposal.)

The main goal of our work was to understand the nature of textures, and from that emerged the importance of 3D cues in understanding the texture detection mechanism in human visual processing. To emulate 3D depth, we employed motion cues to provide depth. This imposes potential limitations on our work, which is that additional information in 3D input may have become available to the 3D-net–some form of temporal information that that 2D inputs do not have. This can be seen as an unfair advantage for the 3D-net, but on the other hand, the 2D-net had additional spatial information which the 3D-net did not have, so eventually these two relative advantages may have canceled out.
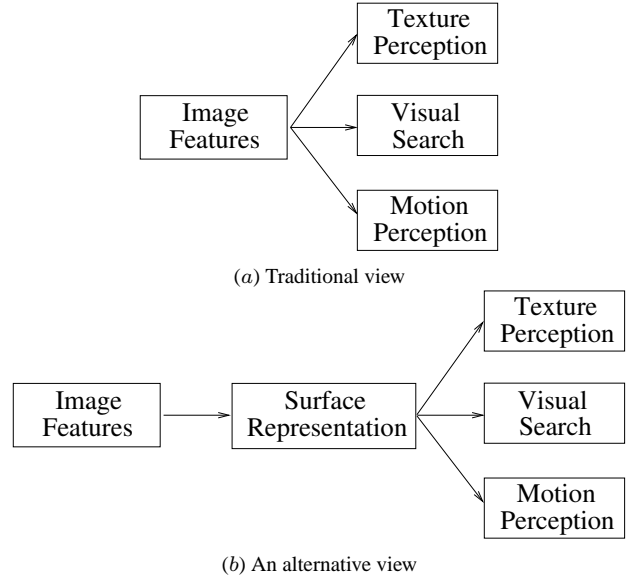
One way of addressing this issue may be to normalize (or equalize) the information content in the 2D vs. the 3D input preparation, which may allow us to more fairly assess the differences between the two modes of texture processing. Finally, another potential criticism may be that we only used S_1 for training. Would a contradictory result emerge if S_2 or S_3 was used to train the networks? We are currently investigating this issue as well, but we believe our main conclusion in this paper will hold even in different training scenarios.

5. Conclusion

We began with the simple question regarding the nature of textures. 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 our insight, we compared texture boundary detection performance of two neural networks trained on textures arranged in 2D and in 3D. Our results revealed that texture boundary detection in 3D is easier to learn than in 2D, and that the network trained in 3D easily solved the 2D problem as well, but not the other way around. Based on these results, we carefully conclude that the human ability to segment texture in 2D may have originated from a module evolved to handle 3D tasks. One

immediate future direction is to extend our current approach to utilize stereo cues as well as monocular cues used in this paper.

Acknowledgments

References

[1] M. Tuceryan, *The Handbook of Pattern Recognition and Computer Vision*, 2nd ed. World Scientific Publishing Co., 1998, ch. 2.1 Texture Analysis, pp. 207–248.

[2] J. Beck, "Effect of orientation and of shape similarity on grouping," *Perception and Psychophysics*, 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] M. S. Landy and N. Graham, *The Visual Neurosciences*. MIT Press, 2004, ch. Visual perception of texture, pp. 1106–1118.

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

[7] Thielscher and Neumann, "Neural mechanisms of cortico-cortical interaction in texture boundary detection: a modeling approach," *Neuroscience.*, vol. 122, no. 4, pp. 921–939, 2003.

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

[9] Z. J. He and K. Nakayama, "Perceiving textures: beyond filtering," *Vision Research*, vol. 34(2), pp. 151–62, 1994.

[10] K. Nakayama, Z. J. He, and S. Shimojo, *Visual Cognition*, 2nd ed. MIT Press, 1995, ch. Visual Surface Representation: A Critical Link between Lower-level and Higher-level vision, pp. 1–70.

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

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

[13] P. Brodatz, *Textures: A Photographic Album for Artists and Designer*. New York: Dover Publication, 1966.

[14] 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.

[15] I. Fogel and D. Sagi, "Gabor filters as texture discriminator," *Biological Cybernetics*, vol. 61, pp. 102–113, 1989.

[16] A. Bovik, M. Clark, and W. Geisler, "Multichannel texture analysis using localized spatial filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 1, pp. 55–73, January 1990.

[17] H.-C. Lee and Y. Choe, "Detecting salient contours using orientation energy distribution," in *Proceedings of the International Joint Conference on Neural Networks*. IEEE, 2003, pp. 206–211.

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

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

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

[21] J. Bergen and M. Landy, "Computational modeling of visual texture segregation," 1991.

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