A New Method for Classification of Structural Textures

Bongkyu Lee

Abstract: In this paper, we present a new method that combines the characteristics of edge information and second-order neural networks for the classification of structural textures. The edges of a texture are extracted using an edge detection approach. From this edge information, classification features called *second-order features* are obtained. These features are fed into a second-order neural network for training and subsequent classification. It will be shown that the main disadvantage of using structural methods in texture classifications, namely, the difficulty of the extraction of texels, is overcome by the proposed method.

Keywords: Structural texture, classification, second-order feature space, MLP.

1. INTRODUCTION

In image analysis, texture is broadly classified into two main categories, statistical and structural [1]. Textures that are random in nature are well suited for statistical characterization, for example, as realizations of random fields. They do not have easily identifiable primitives (e.g., bark, sand, etc.). Structural textures, on the other hand, are characterized by a set of primitives (*texels*) and placement rules. The placement rules define the spatial relationships between the *texels* and these spatial relationships may be expressed in terms of adjacency, closet distance or periodicities. The *texels* themselves may be defined by their gray level, shape or homogeneity of some local property. Many real-world textures contain the structural characteristic. A large number of woven fabrics and commercial furniture are good examples of purely structural or semi-deterministic textures. Microscopic images of electron beam textures in steel surface and human endothelium [2] also have structural characteristics. Thus, structural texture classification has many industrial applications, such as automatic fabric inspection, steel surface testing and electronic catalogues. For these reasons, structural texture classification is an important task in pattern recognition applications.

Texture classification approaches can also be organized into two main categories: statistical and structural approaches [3]. Statistical approaches consider textures as complicated pictorial patterns on which sets of statistics can be defined to characterize these patterns. In the structural methods, the texture is considered to be a cellular and an ordered phenomenon. Hence, the purpose of the first stage of the analysis is to define the *texel*. Since structural methods involve numerous image pre-processing procedures to extract *texels* so that they are time-consuming, statistical approaches are the more efficient approach for texture matching [4]. Thus, most of the former methods used statistical approaches regardless of the class of textures. In the classification of structural textures (Examples can be seen in Fig. 1), however, structural approaches are superior to statistical methods since the spatial structure is more strongly emphasized in the structural approach [5]. Liu *et al.* [6] observed that the MRSAR proposed in [7] (it belongs to statistical approaches) is incapable of distinguishing images where structural textures are involved. This result showed limitations of the statistical methods and the effectiveness of the structural methods in measuring perceptual similarity.

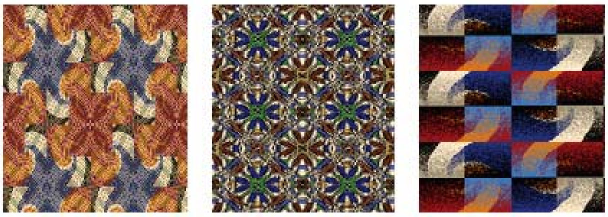

(a) Textile 1. (b) Textile 2. (c) Textile 3.

Fig. 1. Examples of structural textures.

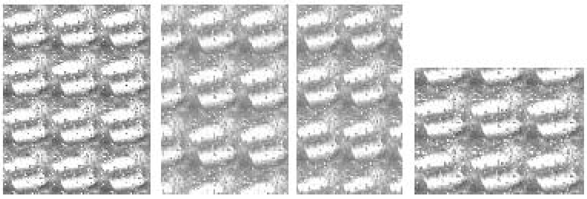

(a) Conf. 1. (b) Conf. 2. (c) Conf. 3. (d) Conf. 4.

Fig. 2. Various configurations of a structural texture.

Manuscript received September 9, 2002; revised April 30, 2003; accepted September 25, 2003. Recommended by Editor Sung-Kwan Oh. This work was supported by the Institute for Basic Sciences in Cheju National University.

BongKyu Lee is with the Department of Computer & Statistics, Cheju National University, Ara-1 dong, ChejuDo, 690-756, Korea (e-mail: bklee@venus1.cheju.ac.kr).

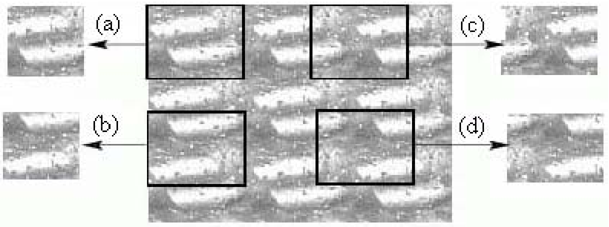


Fig. 3. Various subsamples of a texture.

Although structural approaches are well suited for structural textures, not many researchers have developed texture analysis techniques using structural methods since it is difficult to find an appropriate *texel* in an input texture during the classification. The difficulty of the extraction of *texels* is due to two major problems. Since the image textures to be analyzed generally have *texels* of different sizes, it is difficult to automatically determine the size of *texel* of each input texture during the recognition. The other is that it is complicated to define the correct *texel* since textures with the same *texel* may have more than one configuration (Fig. 2). Furthermore, various types of subimages (Fig. 3) can be extracted in each texture.

In this paper, we present a new method that combines the characteristics of edge informations and second-order neural networks that achieves a high classification rate with structural textures. Several studies [8, 9] have previously shown that using edge information in the texture features can lead to the achievement of good classification performance. The edges of a texture are extracted using an edge detection approach. From edge informations, classification features called *second-order feature spaces* are obtained. These features are fed into a multilayer perceptron (MLP) for training and subsequent classification. The network can overcome the difficulty of the extraction of *texels* by the second-order features and the modified recognition step. Moreover, it requires just one learning sample per texture. Thus, the proposed method has simpler architecture and faster learning capability compared to the existing methods. Experiments were performed with structural textures extracted from the Brodatz texture database [10]. The results were compared with another neural-based model proposed in [11]. Although the proposed method is limited to the classification of periodic textures, the underlying principles will provide an important foundation for ongoing researches to develop more general methods for designing models to classify textures.

The rest of the paper is organized as follows. In Section 2, we review various approaches for texture classification and explain why structural texture classification is important. In Section 3, we first describe basic concepts of the second-order neural network. Then we propose a new generalized second-order neural network based on the *second-order feature spaces* for structural texture classifications. Section 4 presents the architecture of the proposed classification scheme. Section 5 presents the experimental results and Section 6 is the conclusion.

2. TEXTURES AND THEIR CLASSIFICATION - REVIEWS

Texture is observed in the structural patterns of surfaces of objects such as wood, grain, grass and cloth. The term texture generally refers to repetition of basic texture elements called *texels* [11]. A *texel* contains several pixels, whose placement could be periodic, quasi-periodic or random. Natural textures are generally random, whereas artificial textures are often deterministic or periodic. Texture may be coarse, fine, smooth, granulated, rippled, regular, irregular or linear.

A large number of approaches for texture feature extraction and classification have been developed [12, 13]. Methods using Markov Random Field (MRF) models were proposed [14-17]. Gimel *et al.* [17] proposed a MRF model with a Gibbs probability distribution for describing particular classes of uniform stochastic textures. Mao *et al.* [7] proposed simultaneous auto-regressive models to perform texture classification and segmentation. Haralick [12] and others [18, 19] used gray tone dependence co-occurrence matrices to represent texture. Unser *et al*. [4] and others [20-22] proposed methods using adaptive spatial filters. Gabor filter based methods were also proposed [23-25]. The use of Gabor filters in extracting textured image features is motivated by the fact that the Gabor representation has been shown to be optimal in the sense of minimizing the joint two-dimensional uncertainty in space and frequency [24]. Chen *et al.* [3] used statistical geometrical features for textures. Wavelet based feature extraction methods [26, 27], neural network based filtering methods [11, 28-31] and methods using Fourier power spectrum [32] were also proposed for texture classification and segmentation. Liu *et al.* [6] and others [33, 34] used Wold transform to represent textures. The structural approaches use the geometrical features of texture primitives as the texture features. Several edge-based methods have been proposed [35, 36], these generally attempt to locate texture edges based on the computation of a multifeature gradient-like operator. Patel *et al.* [37] calculate edged direction using 3×3 masks and then used rank order statistics to produce the texture features. Hierarchical approaches using pyramid node linking [38] or applying the split-and-merge algorithm to the co-occurrence matrix [39] have been also described.

Since statistical methods characterize the interaction among neighboring image pixels, they are appropriate for modeling random fields with continuous spectra and random textures. When compared to statistical approaches, structural approaches have some advantages where deterministic textures are considered. Rao *et al.* [40] has indicated that the three most important perceptual dimensions in natural texture discrimination can be described as “repetitiveness,” “directionality” and “complexity”. Among them "repetitiveness" is the most important dimension of human perception for structural textures. Since structural approaches try to find an elementary region of a texture and use this for classification, they can measure the preceptual similarity well. To deal with structural textures efficiently, we propose a new second-order neural network using *second-order feature spaces* in Section 3.

3. SECOND-ORDER FEATURE EXTRACTION SCHEME

3.1 Second-order neural networks - basic concepts

The output of a node i, denoted by y_i in a general higher-order neural network is given by

$$\begin{aligned} y_i &= \Theta(h_i) \\ &= \Theta(\textstyle\sum_j W_{ij} x_j + \sum_j \sum_k W_{ijk} x_j x_k + ...), \quad (1) \end{aligned}$$

where Θ is a nonlinear threshold function, h_i is the net input of node *i*, the values of *x* are the values of input nodes, and the interconnection matrix elements *W*. The second-order neural network uses only the second-order term in the activation function of a higher-order neural network. Thus, the output for a second-order network is given by

$$y_i = \Theta(h_i) = \Theta(\textstyle\sum_j \sum_k W_{ijk} x_j x_k) . \quad (2)$$

The inputs are first combined in pairs and then output is determined from a weighted sum of these products. Fig. 4 illustrates the architecture of a strictly second-order neural network. Giles *et al.* [41] showed that the invariances achieved using this network depend on the constraints placed on the weights.

The most severe limitation of second-order neural networks is that the number of input nodes required for an $m \times n$ image is $O((mn)^2)$. This makes implementation difficult. Spirkovska *et al.* [42] solved this problem using coarse coding, which involves the use of overlaying fields of coarser pixels in order to represent smaller pixels.

There have been other approaches based on the invariant features for unraveling the problem. Schmidt *et al.* [43] showed that the constraints placed on the weights in a second-order network can be implemented using the summations of products of each pair of input pixels. This summation of the product at given relative positions or prescribed positions is called SOP (***Summation of Products***). They concluded that a second-order neural network to a specific deformation can be considered as a standard MLP using second-order features that are invariant to a deformation. Fig. 5 shows a second-order neural network that was implemented using MLP and second-order features. Using this scheme, Lee *et al.* [44] and Kwon *et al.* [45] proposed second-order neural networks invariant to types B and C in Fig. 6, which have $O(mn)$ input nodes.

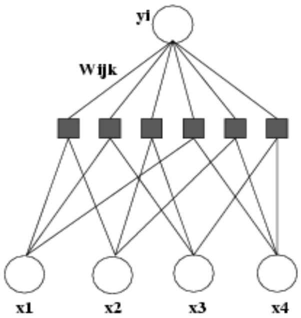


Fig. 4. A strictly second-order neural network.

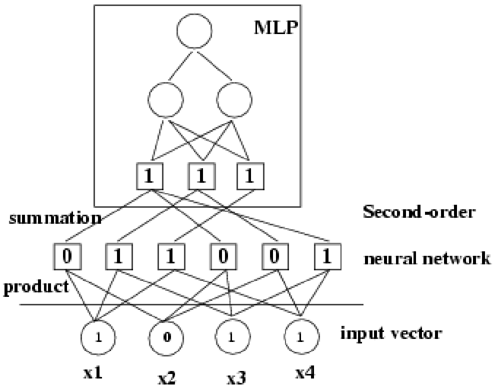


Fig. 5. A second-order neural network implemented using MLP and SOP.

In the following, we will show the weight constraint for the invariance to all four types of wrap-translations and implement the weight constraint using generalized translation invariant second-order features.

3.2 Wrap-translation invariant second-order features

We now describe how to extract generalized second-order features. We first consider a one-dimensional vector *X* of size *n* as an input. To get invariance for the translating with *p* positions, we should update simultaneously each weight that corresponds to a pair of elements x_j and x_k $(j < k)$ of *X* according to (3) when we perform the learning.

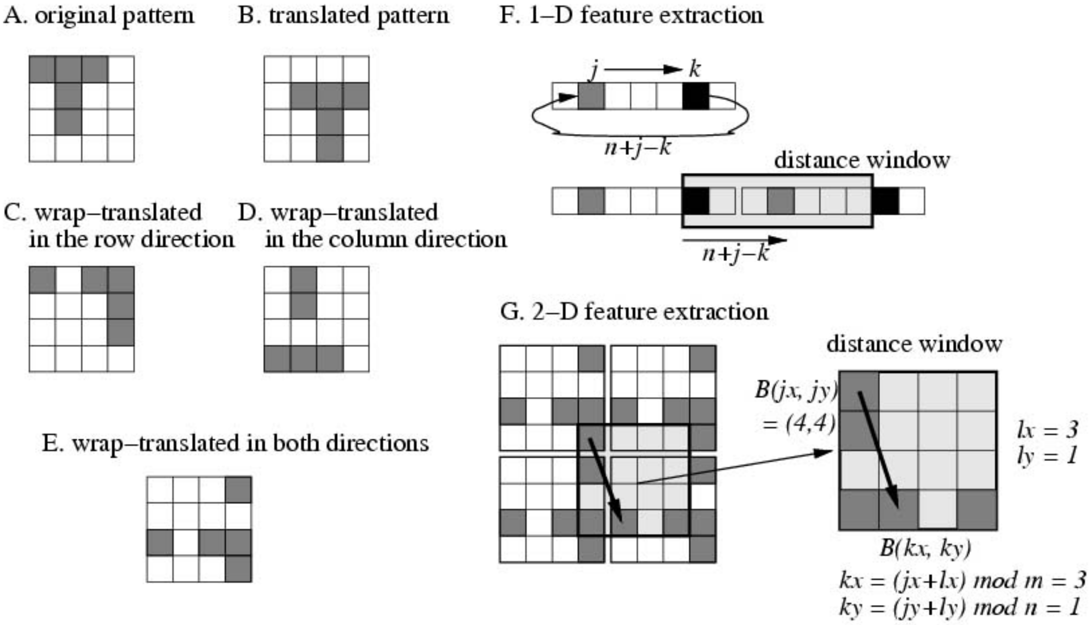

A. original pattern
B. translated pattern
F. 1–D feature extraction
j
k
n+j–k
distance window
n+j–k
C. wrap–translated in the row direction
D. wrap–translated in the column direction
G. 2–D feature extraction
distance window
B(jx, jy) = (4,4)
lx = 3
ly = 1
B(kx, ky)
kx = (jx+lx) mod m = 3
ky = (jy+ly) mod n = 1
E. wrap–translated in both directions

Fig. 6. Translation invariances in a second-order neural network.

```
Algorithm: Extract-feature
1: for each position $(j_x, j_y)$ in an image $B$ do
2:     for $l_x \leftarrow 0$ to $m-1$ do
3:         for $l_y \leftarrow 0$ to $n-1$ do
4:             if $(l_x, l_y)$ is not $(0,0)$ then
5:                 $k_x \leftarrow (j_x + l_x) \bmod m$;
6:                 $k_y \leftarrow (j_y + l_y) \bmod n$;
7:                 $Second(l_x, l_y) \leftarrow Second(l_x, l_y) + B(j_x, j_y) \times B(k_x, k_y)$;
8:             fi
9:         od
10:     od
11: od
```

Fig. 7. The algorithm for extracting second-order features.

$$W_{ijk} = W_{i(j-p)(k-p)} = W_{i(j-p)(k-p+n)} \qquad (3)$$

In the (3), we can define two types of distances between two positions *j* and *k*. The *inner* distance *(k-j)* is defined by $W_{i(j-p)(k-p)}$. The *outer* distance *(n+j-k)* is defined by $W_{i(j-p)(k-p+n)}$ (See Fig. 6(F)) In order to compute the wrap-translated second-order feature easily, we use the notion of *distance windows*. That is, we consider *inner* and *outer* distances at the same time in a distance window. We compute second-order features of a one-dimensional input vector as follows. We consider another copy of *X* as shown in Fig. 6(F), and then slide the distance window *DW* of size *n* on the concatenation of two copies of *X*. If *DW* is aligned at position *j* in *X*, we compute the *inner* and *outer* distances between two elements x_j and x_{j+l} for all *l* ($1 \le l < n$).

Now, we consider a two-dimensional image *B* of size $m \times n$ as an input. In this case, we extend the notion of distance windows to two dimensions. Since we allow all kinds of wrap-translated patterns in this paper, we consider four copies of *B* to compute second-order features as shown in Fig. 6(G). Fig. 7 shows the algorithm *Extract-feature* that computes second-order features of input image *B* of size $m \times n$. We use a two-dimensional array *Second* of size $m \times n$ to save second-order features. At line 1, we align the distance window at position (j_x, j_y) in *B*. For all distances (l_x, l_y) except the zero distance on the distance window, we compute the product of two pixels $B(j_x, j_y)$ and $B(k_x, k_y) = B((j_x + l_x)$ mod *m*, $(j_y + l_y)$ mod *n*), and then save it into $\text{Second}(l_x, l_y)$ at line 7. Hence, we can correctly compute the second-order features of image *B* after performing the algorithm.

An example of computing an outer distance between the position *(4,4)* and the position *(3,1)* in *B* is shown in Figure 6-G. When the distance window is aligned at position *(4,4)* in *B*, $j_x = 4$ and $j_y = 4$, the algorithm starts the computation at line 2 with $l_x = 0$ and $l_y = 0$. After some iterations, the algorithm sets (l_x, l_y) to *(3,1)*. Then the algorithm computes the (k_x, k_y) at lines 5, 6 and sets it to *(3,1)*. This means that the outer distance between *B(4,4)* and *B(3,1)* is *(3,1)*, 3 in the row direction and 1 in the column direction. Thus, the algorithm saves the product of *B(4,4)* and *B(3,1)* into *Second (3,1)*.

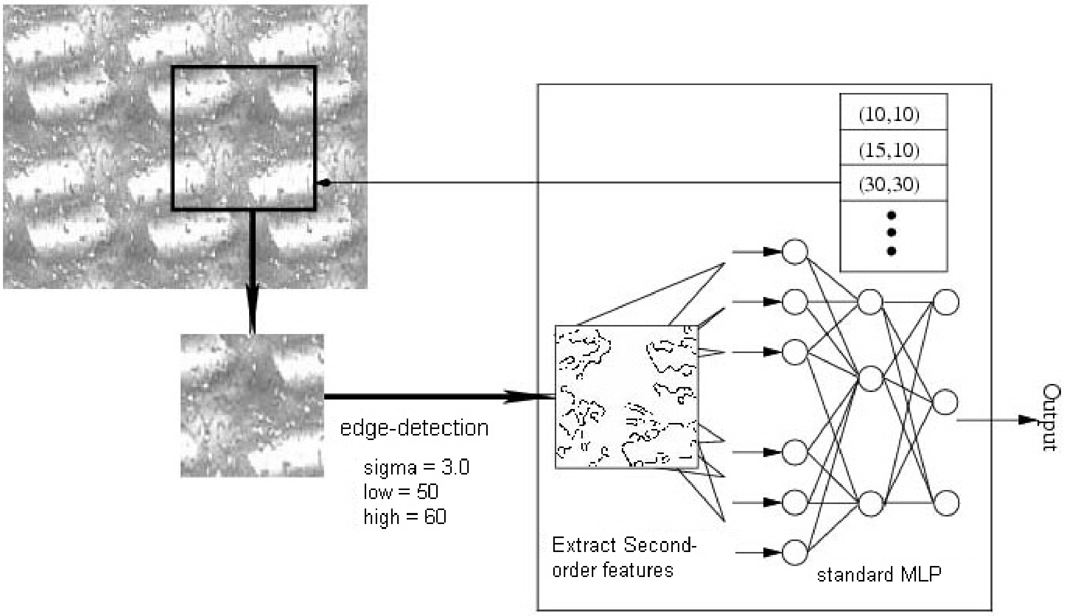


Fig. 8. Block diagram of the proposed network.

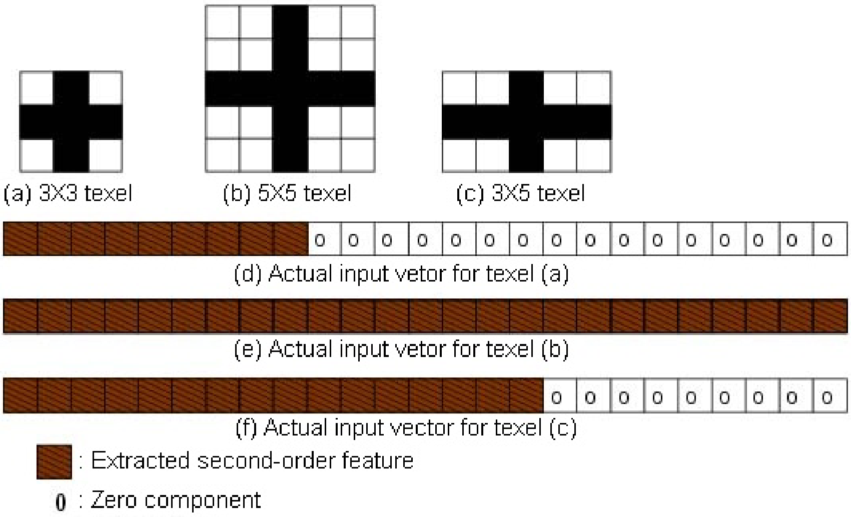
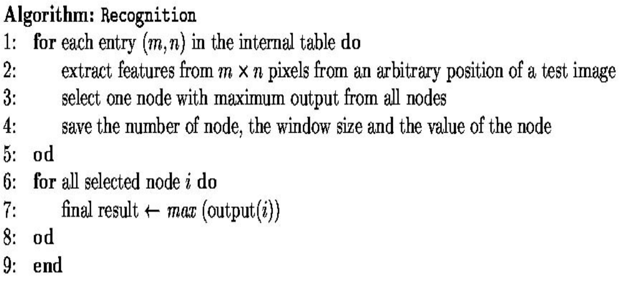

(a) 3X3 texel
(b) 5X5 texel
(c) 3X5 texel
(d) Actual input vetor for texel (a)
(e) Actual input vetor for texel (b)
(f) Actual input vector for texel (c)
▨ : Extracted second-order feature
0 : Zero component

Fig. 9. Input vectors for three different sized images.

4. THE ARCHITECTURE OF THE PROPOSED MODEL

The proposed classification scheme is comprised of a structure of edge extraction and a second-order neural network. First, an edge map of the size $m \times n$ of an input image is generated, where $m \times n$ is the dimensionality of an input image. Then the second-order neural network receives the edge map from the edge extraction stage and extracts $m \times n$ dimensional feature spaces, which are called *second-order feature spaces* in this paper. The transformed feature vectors are fed into the MLP for classification. Fig. 8 illustrates the block diagram of the proposed scheme.

The MLP has two consecutive phases, training phase and recognition phase. In the training phase, two different strategies were adopted in contrast to the traditional MLP training. The number of input nodes was defined by the dimension of the largest *texel*, since the MLP may receive *texels* with various sizes. If three sample *texels* with 3×3, 5×5 and 3×5 sizes are considered, then the MLP has 25 (5×5) input nodes. In order to apply the network to smaller *texels*, outsized vector components are filled with zero. Fig. 9 shows the actual input vectors when three different sizes are applied. The other is that all pairs *(m,n)* of sample *texels* are recorded in the internal table of the



```
Algorithm: Recognition
for each entry $(m,n)$ in the internal table do
    extract features from $m \times n$ pixels from an arbitrary position of a test image
    select one node with maximum output from all nodes
    save the number of node, the window size and the value of the node
od
for all selected node $i$ do
    final result $\leftarrow$ $max$ (output($i$))
od
end
```

Fig. 10. The algorithm for the recognition phase.

network, where *m* is the row size of a *texel* and *n* is the column size. This record will be used to solve the size variance of *texels* in the recognition phase. The contents of the table can be shown in the right side of Fig. 8 once *texels* of sizes 10 × 10, 15 × 10 and 30 × 30 are trained.

The recognition phase of the trained network was modified compared to the traditional recognition phase of the neural network. Using each entry of the internal table, the output of the network is computed with second-order features from $m \times n$ pixels from a random position of a test image. Then the node with the maximum value is selected for the recognition result for each entry. If *(10,10), (15,10)* and *(30,30)* are the current entries of the internal table, three nodes are selected for all entries. Each selected node represents the result for each window size. From all the selected nodes, the node with the maximum value is considered as the final recognition result of a test image. Fig. 10 shows the algorithm Recognition.

5. EXPERIMENTAL RESULTS AND DISCUSSION

5.1 Preparation

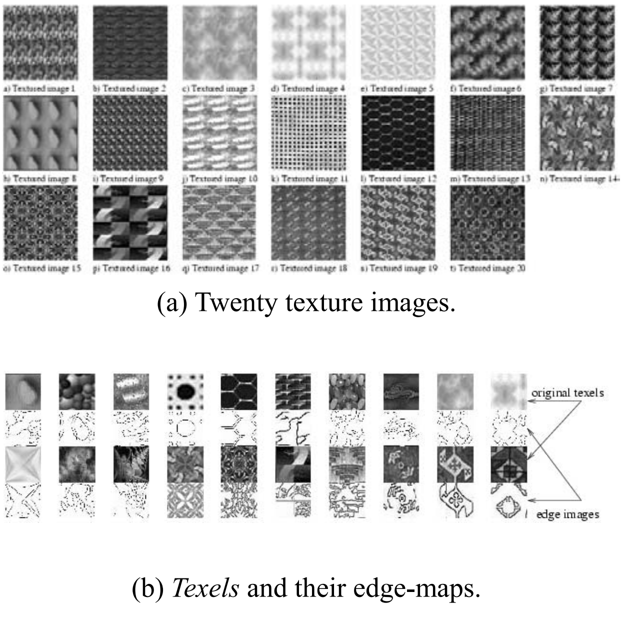
The performance of the proposed scheme was analyzed using a variety of structural textures, including the Brodatz photo album [10]. Twenty deterministic textures of size 128 × 128 (Fig. 11) were used for the experiments. These images were categorized into four sets according to the size of *texel*, which are *15 × 15, 15 × 20, 20 × 20 and 30 × 30.* For the training of the network, *texels* within images were extracted manually and applied to the network. The MLP adapted its weights according to the learning rule (backpropagation) and recorded the size of the applied *texel.* Since the proposed network is a second-order neural network, the network needs to be trained on just one *texel* of each texture, not on numerous distorted views. Such generalization has been demonstrated in numerous simulations [42]. In the recognition phase, the trained network extracted a pattern from a random position of each texture and classified it using the algorithm `Recognition`. This recognition test was performed 20 times for one image. Thus, the total number of recognition tests was 400 (20 times per texture).

The performance of the proposed scheme was compared with the model proposed in [11], which belongs to the statistical approach. This model used a set of Gabor filters for extracting texture features and a structure that combined the characteristics of SOM (Self Organizing Map) and first-order MLP for classifications. It was trained using features from 400 (*20 × 20*) pixels from the center of each texture. Then, the classifications were performed using the identical method used in the proposed system. The comparison between two systems will show the superiority of the proposed system to statistical systems in the classification of deterministic textures.

5.2. Simulation and results

The implementation details of the second-order network are listed in Table 1. Since the size of the largest *texel* in the data set was *30 × 30*, the number of input nodes is 900 (*30 × 30*). The number of output nodes represents the number of image categories.

The number of hidden nodes was determined as follows. We implemented several networks having different numbers of hidden nodes. Then, we chose the number of hidden nodes having the best learning capability. The criterion was the standard deviation *std.* of PSS (Pattern Square Sum error: sum of squared errors of output nodes for each pattern) values. Table 2 shows *std.* of PSS for each network. In the Table, 10 hidden nodes presented the least value. Thus, we chose 10 hidden nodes to implement the network.



(a) Twenty texture images.

(b) *Texels* and their edge-maps.

Fig. 11. Images used in the simulations.

Table 1. Configurations of the second-order neural network.

Parameters	Values
# of input nodes	900
# of hidden nodes	10
# of output nodes	20
Learning rate	0.26
Activation function	Sigmoid ($y_i = 1/(1+e^{-h_i})$)

Table 2. The comparison of learning capabilities according to the hidden nodes.

# of hidden nodes	4	6	8	10	14	20	30
Std. of PSS	0.309	0.219	0.160	0.092	0.133	0.226	0.275

Table 3. The filter used in the compared system.

Component name	Gabor Filter
Orientation	0, 45, 90, 135 degrees
Bandwidths	6.25
Frequencies	2.5π , 5π , 10π pixels/cycle

Table 4. Network Specifications of the compared system.

Network Type	SOM	MLP
# of input nodes	400	25
# of hidden nodes	0	4 (Randomly selected)
# of output nodes	$25(5\times 5)$	20
Learning method	Kohonen Rule	Backpropagation

Table 3 and Table 4 show the implementation details of the compared model for experiments. We used 12 Gabor filters, having single bandwidth, three frequencies and four orientations, constituting a 12-dimensional feature vector representing each pixel in the image.

The simulation results are listed in Table 5. Both of the systems failed to achieve 100% accuracy in the experiments. However, the proposed system produced superior recognition results than the compared system. Since it cannot be predicted which type of translation has occurred, the first-order MLP required additional training patterns [41-43], while the second-order MLP using second-order feature spaces did not require more than one pattern. We also observed that errors can be eliminated by some parameters of the edge detection. Various edge images can be obtained by changing three parameters of the edge detection algorithm.

Table 5. Recognition results of the experiments.

Pattern id	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	T18	T19	T20
The proposed (# of correct recognitions)	20	20	19	20	20	19	20	20	19	20	20	19	20	20	20	20	17	20	20	18
The compared (# of correct recognitions)	15	14	15	13	16	10	12	17	19	8	20	18	0	7	16	15	18	16	6	16

The used values for the three parameters were seen in Fig. 8.

From the evaluation results, it can be seen that the proposed method has good characteristics for deterministic texture classification compared to the existing models. These characteristics are due to capabilities of *second-order feature spaces*.

6. CONCLUSION

In this paper we have proposed a new two-stage model for the classification of deterministic textures. The model used *second-order feature spaces* of the edge map of each *texel* for feature extraction. The results were achieved by a second-order neural network trained by the backpropagation algorithm. The evaluation of the model with a set of deterministic textures was performed and compared to another neural-based method proposed in [11]. As a result, our method showed better classification results than the compared model.

It was found that *second-order feature spaces* solved the basic problem occurring in the deterministic texture recognition, the translation variance of *texel*. Moreover, modified learning and recognition phases solved the size variation of *texels*. It can also be implemented more easily than the existing methods because of its simplicity. Finally, the proposed model is size limited since the dimension of *second-order feature spaces* is also of the size $m\times n$ for an $m\times n$ input image.

Further research will be conducted for recognizing textures consisting of more than one *texel*. Researches for classification of deterministic textures with noise distortions will be also conducted.

REFERENCES

[1] A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, 1989.

[2] W. Zhang, A. Hasegawa, K. Itoh, and Y. Ichioka, "Image processing of human corneal endothe-

lium base on a learning network," *Applied Optics*, vol. 30, pp. 4211-4217, 1991.

[3] Y. Q. Chen, M. S. Nixon, and D.W. Thomas, "Statistical geometrical features for texture classification," *Pattern Recognition*, vol. 24, no. 4, pp. 537-552, 1995.

[4] M. Unser and M. Eden, "Multiresolution feature extraction and selection for texture segmentation," *IEEE Trans. on PAMI*, vol. 11, no. 7, pp. 717-728, 1989.

[5] O. D. Faugeras and W. K. Pratt, "Decorrelation methods of texture feature extraction," *IEEE Trans. on PAMI,* vol. 2, no. 2, pp. 323-332, 1980.

[6] F. Liu and R. W. Picard, "Periodicity, directionality and randomness: wold features for image modeling and retrieval," *IEEE Trans. on PAMI,* vol. 18, no. 7, pp. 722-733, 1996.

[7] J. Mao and A. K. Jain, "Texture Classification and segmentation using multiresolution simultaneous autoregressive models," *Pattern Recognition*, vol. 25, no. 2, pp. 173-188, 1992.

[8] H. Tamura, S. Mori, and T. Yamawaki, "Texture features corresponding to visual perception," *IEEE Trans. on System, Man and Cybernetics*, vol. 8, no. 6, pp. 460-473, 1978.

[9] L. S. Davis, S. A. Johns, and J. K. Aggarwal, "Texture analysis using generalized cooccurrence matrices," *IEEE Trans. on PAMI*, vol. 1, no. 3, pp. 251-259, 1979.

[10] P. Brodatz, *Textures*, Dover, New York, 1996.

[11] P. P. Raghu, R. Poongodi, and B. Yegnanarayana, "A combined neural network approach for texture classification," *Neural Networks*, vol. 8, no. 6, pp. 975-987, 1995.

[12] R. M. Haralick, "Statistical and structural approaches to texture," *Proc. of IEEE*, vol. 67, no. 5, pp. 786-804, 1979.

[13] T. R. Reed and J. M. Hans Du Buf, "A review of recent texture segmentation and feature extraction techniques," *CVGIP: image understanding*, vol. 57, pp. 359-372, 1993.

[14] G. R. Gross and A. K. Jain, "Markov random field texture models," *IEEE Trans. on PAMI*, vol. 5, no. 1, pp. 25-39, 1983.

[15] R. Chellapa, S. Chatterjee, and R. Bagdazian, "Texture synthesis and compression using Gaussian-Markov random field models," *IEEE Trans. on System, Man and Cybernetics*, vol. 15, pp. 298-303, 1985.

[16] B. R. Povlow and S. M. Dunn, "Texture classification using noncausal hidden markov models," *IEEE Trans. on PAMI*, vol. 17, no. 10, pp. 1010-1014, 1995.

[17] G. L. Gimel'farb, "Texture modeling by multiple pairwise pixel interactions," *IEEE Trans. on PAMI,* vol. 18, no. 11, pp. 110-1114, 1995.

[18] R. W. Conners and C. A. Harlow, "A theoretical comparison of texture algorithms," *IEEE Trans. on PAMI*, vol. 2, no. 3, pp. 204-222, 1980.

[19] M. Unser, "Sum and difference histograms for texture classification," *IEEE Trans. on PAMI*, vol. 8, no. 1, pp. 118-125, 1986.

[20] A. C. Bovik, M. Clark, and W. S. Geisler, "Multichannel texture analysis using localized spatial filters," *IEEE Trans. on PAMI*, vol. 12, no. 1, pp. 55-73, 1990.

[21] A. C. Bovik, "Analysis of multichannel narrowband filters for image texture segmentation," *IEEE Trans. on Signal Processing*, vol. 39, no. 9, pp. 2025-2043, 1991.

[22] T. N. Tan, "Rotation invariant texture features and their use in automatic script identification," *IEEE Trans. on PAMI*, vol. 20, no. 7, pp. 751-756, 1998.

[23] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognition*, vol. 24, no. 12, pp. 1167-1186, 1993.

[24] B. S. Manjuath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. on PAMI*, vol. 18, no. 8, pp. 837-842, 1996.

[25] G. M. Haley and B. S. Manjuath, "Rotation invariant texture classification using modified Gabor filters," *Proc. IEEE ICIP95*, pp. 262-265, 1995.

[26] T. Chang and C.-C. J. Kuo, "Texture analysis and classification with tree structured wavelet transform," *IEEE Trans. on Image Processing*, vol. 2, no. 4, pp. 429-441, 1993.

[27] W. Y. Ma and B. S. Manjuath, "A comparison of wavelet features for texture annotation," *Proc. IEEE Int'l Conf. Image Processing '95*, vol. II, pp. 256-259, 1995.

[28] N. R. Dupaguntla and V. Vemuri, "A neural network architecture for texture segmentation and labeling," *Proc. IJCNN'89*, vol. I, pp. 127-133, 1989.

[29] V. Ari, "A texture classifier based on neural network principles," *Proc. IJCNN '90*, vol. I, pp. 491-496, 1990.

[30] F. A. DeCosta and M. F. Chouikha, "Neural network recognition of textured images using third order cumulants as functional links," *Proc. ICASSP '92*, vol. IV, pp. 77-80, 1992.

[31] A. K. Jain and K. Karu, "Learning texture discrimination masks," *IEEE Trans. on PAMI*, vol. 18, no. 2, pp. 195-205, 1996.

[32] J. O. Eklundh, "On the use of fourier phase features for texture discrimination," *Computer, Graphics and Image Processing*, vol. 9, pp. 191-201, 1979.

[33] J. M. Francos, A. Z. Meiri, and B. Porat, "A Unified texture model based on a 2-D Wold like

decomposition," *IEEE Trans. on Signal Processing*, pp. 2665-2678, 1993.

[34] J. M. Francos, A. Narasimhan, and J.W. Woods, "Maximum likelihood parameter estimation of textures using a Wold decomposition based model," *IEEE Trans. on Image Processing*, pp. 1655-1666, 1995.

[35] S. Grinaker, "Edge based segmentation and texture separation," *Proc. of International Conference on Pattern Recognition, 80*, pp. 776-780, 1980.

[36] A. Khotanzad and J. Y. Chen, "Unsupervised segmentation of textured images by edge detection in multidimensional features," *IEEE Trans. on PAMI*, vol. 11, no. 4, pp. 414-421, 1989.

[37] D. Patel and T. J. Stonham, "Texture image classification and segmentation using rank order clustering," *Proc. of International Conference on Pattern Recognition, 92*, pp. 92-95, 1992.

[38] M. Pietikainen and A. Rosenfeld, "Image segmentation by texture using pyramid node linking," *IEEE Trans. on System, Man and Cybernetics*, vol. 11, pp. 822-825, 1981.

[39] P. C. Chen and T. Pavlidis, "Segmentation by texture using a co-occurrence matrix," *Computer, Graphics and Image Processing*, pp. 172-182, 1979.

[40] A. R. Rao and G. L. Lohse, "Towards a texture naming system: identifying relevant dimensions of texture," *Proc. IEEE Conf. Visualization*, pp. 220-227, 1993.

[41] C. L. Giles and T. Maxwell, "Learning, invariance and generalization in high-order neural networks," *Applied Optics*, vol. 26, pp. 2972-2978, 1987.

[42] L. Spirkovska and M. B. Reid, "Coarse-coded higher-order neural networks for PSRI object recognition," *IEEE Trans. on Neural Networks*, vol. 4, no. 2, pp. 276-283, 1993.

[43] W. A. C. Schmidt and J. P. Davis, "Pattern recognition properties of various feature spaces for higher order neural networks," *IEEE Trans. on PAMI*, vol. 15, no. 8, pp. 795-801, 1993.

[44] B. Lee, S. Cho, and Y. Cho, "Translation, scale and rotation invariant pattern recognition using PCA and reduced second-order neural network," *Int. Journal of Neural, Parallel and Scientific Computation*, vol. 3, no. 3, pp. 417-429, 1995.

[45] H. Y. Kwon, B. C. Kim, and H. Y. Hwang, "Rotation and scale invariant pattern recognition using complex log mapping and augmented second-order neural network," *IEE Electronics Letters*, vol. 29, no. 7, pp. 620-621, 1992.

Bongkyu Lee received the Ph.D. in Computer Engineering from Seoul National University in 1995. His research interests include AI, Pattern recognition, Bioinformatics and Biometrics.