Rotation Invariant Texture Recognition Using a Steerable Pyramid

H. Greenspan, S. Belongie, R. Goodman and P. Perona
California Institute of Technology 116-81 — Pasadena, CA 91125 — (hayit@micro.caltech.edu)

Abstract

A rotation-invariant texture recognition system is presented. A steerable oriented pyramid is used to extract representative features for the input textures. The steerability of the filter set allows a shift to an invariant representation via a DFT-encoding step. Supervised classification follows. State-of-the-art recognition results are presented on a 30 texture database with a comparison across the performance of the K-nn, Back-Propagation and Rule-Based classifiers. In addition, high accuracy estimation of the input rotation angle is demonstrated.

1 Introduction

A number of texture recognition systems have been recently proposed in the literature, giving very high-accuracy classification rates (e.g., [1-4]). The next challenge in the texture recognition arena is to achieve rotation- and scale-invariant recognition systems. In this work we concentrate on rotation-invariance. Rotation-invariant classification is essential for most real-life applications, however it has been mostly overlooked in the literature due to the complexity of the task. Some results on small databases may be found in the literature (e.g., [5-8]). In these works, rotation-invariance is obtained either only on a discrete set of orientations or by eliminating the orientation information altogether.

We present a rotation-invariant texture recognition system which achieves rotation-invariant classification on a large database of textures as well as extraction of the rotation of the input image relative to the prestored texture database. We have attempted to constitute a large and heterogeneous database, which can resemble a real-world scenario. It is composed of the 30 textures shown in Fig. 1. Most of the textures are taken from the standard Brodatz database of textures [9]; the others have been collected by us from a variety of real-world texture sources (e.g., jeans, printed text).

The need for rotation-invariance in a recognition system can be shown by the decline in performance of a "classical" recognition framework, which is not designed with rotation-invariance in mind. Fig. 2 shows a misclassification curve for a system which is comprised of a multi-scale (pyramidal) feature-extraction stage followed by a supervised classification scheme (the system has been presented in [4]). The classification results are averaged over the entire 30 texture database. The degradation in performance with increasing rotation angle of the input test images is evident, and unacceptable beyond 20°.

Rotation-invariance can be achieved in one of two ways,

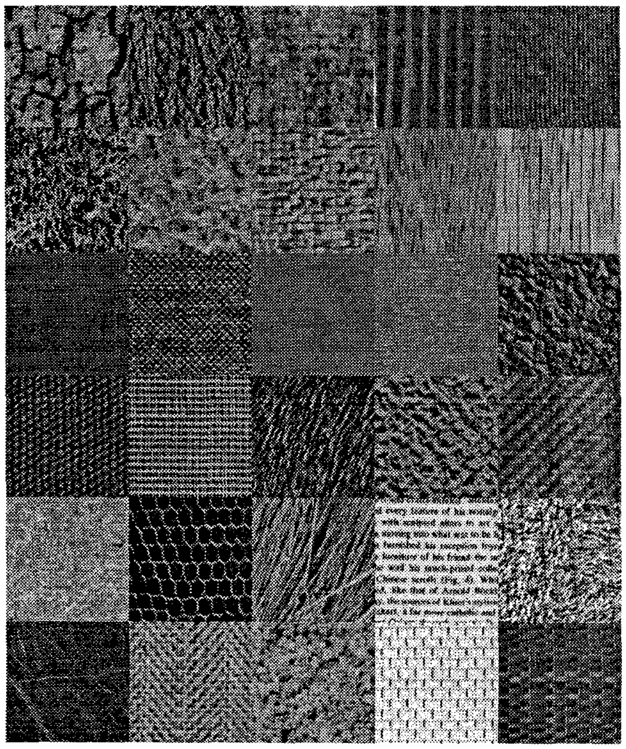

Fig. 1: 30 texture database. Labels: ROW 1: bark(D12), calf(D24), cloth(D19), cardboard, jeans. ROW 2: grass(D9), pig(D92), raffia(D84), water(D38), wood(D68). ROW 3: backpack, bookbox, brownbag, check-book cover, cork(D32). ROW 4: cotton canvas(D77), furcanv(D20), fur(D93), handmade paper(D57), napkin. ROW 5: particle board, reptile(D3), straw(D15), text, towel. ROW 6: vinyl, herringbone (D16), sand(D29), wire(D6), strawmat(D55). Textures taken from the Brodatz book are labeled with the corresponding plate number. Others have been scanned in from real-world texture sources.

1051-4651/94 $04.00 © 1994 IEEE

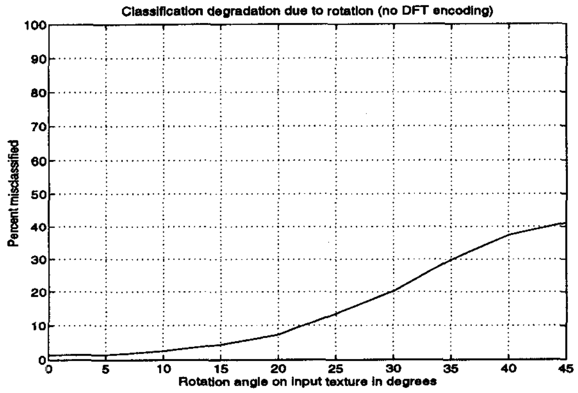

Fig. 2: Degradation in the classification performance with rotation angle, 30 texture database.

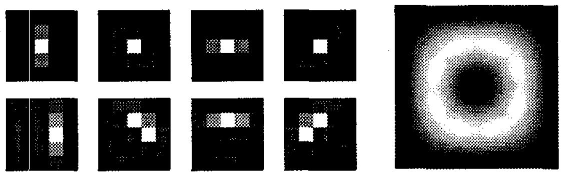

Fig. 3: Left: A set of oriented pyramid filters. Real and imaginary components are presented, top and bottom, respectively. Right: Power spectra characteristics for the chosen filter set (+ conjugate counterparts).

either by extracting rotation-invariant features or by appropriate training of the classifier to make it 'learn' invariant properties. We use the first approach. In section 2 we suggest the use of a steerable pyramid of oriented filters which is the key element of our system in coping with the invariance challenge. In section 2 we demonstrate the use of the Discrete Fourier Transform (DFT) for interpolating the steerable filter outputs to extract rotation-invariant features. The rotation-invariant recognition system is presented in section 3 along with classification results on the 30 texture database of Fig. 1. Rotation-angle estimation results are presented in section 4.

2 Steerable Pyramid and Invariance

The front end of our system consists of the oriented Laplacian pyramid, which is discussed in detail in [4, 10]. The oriented Laplacian pyramid provides us with a computationally efficient scheme to extract a set of orientation-selective bandpass filtered versions of the input textures. Three scales are used (with octave spacing) with four orientations per scale (spaced 45° apart). The set of oriented filters are complex exponentials modulated by Gaussians acting on the Laplacian pyramid. A set of the pyramid filters is shown in Fig. 3 (left). This set of filters was shown to span the orientation space (see Fig. 3 right) and to form a *steerable basis* [10].

Feature vectors are formed from the outputs of the oriented filters, describing the local characteristics (in 8×8 windows) of the original image. The feature vectors are 15-dimensional consisting of the 4 oriented components per scale together with a nonoriented component extracted from the Laplacian pyramid.

For a given input (texture) we define a *feature curve* (per scale) across orientation space, $f_c(\theta)$, as the texture's response to any oriented filter in the 360° space (using symmetry considerations we will concentrate on the 180° space). Using the steerability property we note that the four oriented components (with 45° sampling period), per scale, of the feature vector allow us to reconstruct the continuous feature curve.

As an input texture is rotated, its feature curve, $f_c(\theta)$, shifts across the orientation axis. Alternatively we can visualize the sample points cycling along the continuous curve. It is our goal to find a rotation-invariant representation for the sampled curve . We will next describe how the DFT-encoding can be used for this task.

2.1 Use of the DFT for Rotation Invariance

Let $f(n), n = 0..3$, denote the oriented components at a single scale in a given feature vector. We define a *companion* feature vector $\hat{f}(k)$ to be the Discrete Fourier Transform of $f(n)$ as follows [1]:

$$\hat{f}(k) = \sum_{n=0}^{3} f(n) e^{-i\pi nk/2} \qquad k = 0, 1, 2, 3. \qquad (1)$$

Using the extracted coefficients above, we can represent the original feature curve, $f_c(\theta)$, as:

$$f_c(\theta) = A + B\cos(\theta + C) + \text{rotation-variant term} \quad (2)$$

with the coefficients given by

$$A = \hat{f}(0), B = |\hat{f}(1)|, C = \arg[\hat{f}(1)], \qquad (3)$$
$$\text{Nyquist component} = \hat{f}(2).$$

We note that A and B do not change as a result of rotation, while $C/2$ is equal to the rotation angle of the input (The division by two is necessary since $f(n)$ goes through *two* complete cycles during a rotation of the input image by 360°). The value of B represents the strength along the orientation indicated by $C/2$. In this case we do not have a high-enough sampling rate to extract the phase component of the second harmonic. The Nyquist component,

[1]The complete feature vector includes the DFT components for each scale together with the non-oriented components of the Laplacian pyramid.

which varies with rotation of the input, provides no useful information.

Using the DFT encoding, we can thus create an *invariant* feature vector which, at each scale, consists of the values of A and B (as above), while the value of $C/2$ may be inspected to determine the rotation angle on the input. The DFT-encoded feature vector is the input to the recognition system.

In the above analysis we have addressed the issue of a single dominant orientation in the input textures. We note that in the case of multiple orientations at a single location (such as in a cross pattern) additional harmonics are needed in our representation, or additional filters, to sample the corresponding feature curve. For example, with two harmonics:

$$f_c(\theta) = A + B\cos(\theta + C) + D\cos(2\theta + E) + \text{rot.-variant term.} \quad (4)$$

Thus, to handle multiple orientations, we may extend the above analysis, with a larger set of (narrower frequency tuned) filters. (e.g., a set of 8 kernels, spaced 22.5° apart and with twice the spatial frequency of the above filters). Here, the parameters A, B and D provide us with a rotation-invariant representation while C and E indicate the orientations of any regions in the image with either one or two dominant orientations.

We note that for N multiple orientations, we obtain $N + 1$ invariant terms and N phase components that vary directly with rotation, for a total of $2N + 1$ components. In the above scheme, all of the N phase components are discarded in the recognition process. As an alternative, one phase component (such as $C/2$) may be chosen as the angle by which to either 'derotate' the actual input texture or interpolate the proper translation in the feature vectors, thus giving us a total of $2N$ useful components for recognition.

3 Rotation-Invariant Texture Recognition

Two major stages comprise a general recognition system - a feature extraction stage and a classification stage. In the proposed system, feature-vectors are extracted via the Oriented Pyramid. The DFT encoding step is added next as part of the feature-extraction phase of the system. In the classification phase we compare the k-nearest neighbor algorithm K-nn) [11], the Backprop neural-network algorithm [12] and a Rule-based network classifier (ITRULE) [13] The overall system block diagram is depicted in Fig. 4. For more details on the system the reader is referred to [14]. We next demonstrate the performance of the rotation-invariant system on the 30 texture database of Fig. 1.

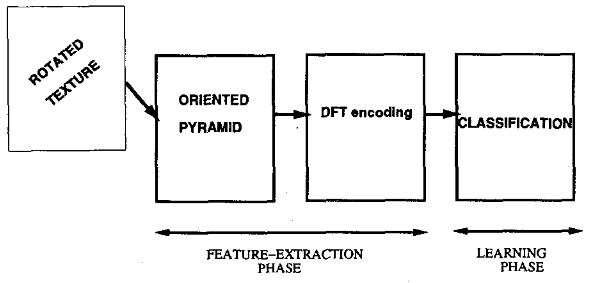

Fig. 4: System Block Diagram

Experimental setup

The 30 texture database is comprised of 256 × 256 images. For each 256 × 256 texture patch, a set of 16 non-overlapping 64 × 64 windows are extracted. 12 of the windows are used for training and 4 *different* ones are used for testing. The test set is constructed by rotating each of the test windows by 5° increments, between 5° and 50°; thus we have 40 test vectors per texture. The training inputs are all rotated by 5° so that all members of the testing and training set have the same amount of error due to the rotation algorithm.

The recognition process entails the following steps:
1.The 64 × 64 texture patch is passed through the steerable pyramid to result in a set of 15 8 × 8 filter maps.
2.The 8 × 8 filtermaps are averaged, to produce one representative feature vector per 64 × 64 input window.
3.The extracted feature vector, f, is DFT encoded to generate the companion feature vector, $\hat{f}$.
4.The magnitudes of the set of DFT-encoded feature-vectors are next presented to the classification system for recognition.

In the classification stage we compare across several non-parametric learning schemes. The Rule-based network classifier has been presented for texture classification in [4]. It is composed of an unsupervised K-means clustering stage followed by a supervised classification stage which is an information-theoretic rule-based scheme [13]. An information theoretic measure is used to learn the most informative links or rules between features and class labels. In the testing phase, the a-priori defined rules provide the posterior probability for the output classes. The learned rules can be mapped onto a rule-based neural-network and thus the classification scheme is parallelizable and suitable for implementation using special purpose neural-network hardware. The Rule-based scheme is compared with the more standard K-nearest neighbor classifier and the Backpropagation neural-network. In the K-nn algorithm we average over several K nearest-neighbor values ($K = 1, 3, 5, 7$). A 3-layer Backprop network is used with a 15-dimensional input space, a node for each class in the output layer, and

30 TEXTURE CLASSIFICATION - 4 cases				
rot.	DFT	K-nn	Backprop	Rule-based
N	N	95%	97.25%	97.5%
Y	N	80%	67.5%	77.42%
N	Y	90%	83%	85.83%
Y	Y	91.5%	84.67%	86.58%

Table 1 Classification results for the 30 texture database of Fig. 1

several hidden layer configurations (results are averaged over 30, 60 and 90 hidden units).

To augment the testing results 4 different runs are made, each with a different set of 4 testing windows, and the classification results are averaged over these runs.

Results

We compare results for 4 different classification scenarios, as given in Table 1. The 4 cases reflect the performance of the system as related to the state of the input test patterns (rotated vs. non-rotated), and the input representation space (rotation-invariant (via DFT encoding) vs. non rotation-invariant).

In **case 1** the data is nonrotated and no DFT conversion is performed. High classification rates are achieved. These rates indicate the strength of the recognition system on non-rotated inputs.

In **case 2** the test data is rotated and no DFT conversion is performed. Here the strong decline in performance is evident, as is expected for a non rotation-invariant system. The result presented is averaged over both the oriented and non-oriented textures in the database, with the non-oriented cases (which are less affected by the rotation) actually augmenting the performance. In addition, rotations of $5^\circ - 50^\circ$ were included. A more severe drop in performance would be evident if only the large rotations were included.

In **case 3** the test data is nonrotated but a DFT representation is used. Here we get high classification results, though somewhat reduced from case 1. This is the price paid for shifting to a rotation-invariant representation which makes fewer assumptions about what is known. In using the *magnitude* of the DFT-encoded vectors for the classification task, we ignore the phase information. In the 4 dimensional case we are actually left with only 2 invariant components (we zero out the single phase component and we cannot rely on the Nyquist component). This loss of information results in the reduced performance.

Finally, in **case 4** we have a rotated test set analyzed by the rotation-invariant system. The increase in the results from case 2 are evident. As expected the results are similar to case 3. We are able to classify the varied database of 30 textures rotated at 5° resolution at an accuracy of close to 90%.

Comparable results are achieved across the three classification schemes utilized. The above results represent state-of-the-art recognition results in the domain of large-database rotation-invariant natural texture recognition.

4 Orientation Estimation

In this section we study a method for computing the *rotation angle* of a given test patch relative to a reference texture in the database. In the context of the recognition stage, only the magnitude of the DFT-encoded feature vector is used. Upon identification, the *phase* of the DFT can be inspected to determine the amount of rotation of the input texture relative to a prestored sample of that texture class.

As discussed in section 2, our scheme allows for the phase to be extracted at each scale on each 8×8 image tile. A key question is whether, given a sizeable texture patch, a more reliable estimate of orientation is obtained by direct averaging of the phases of the component 8×8 tiles, or by computing the phase of the average of the filter outputs on the entire patch. In order to address this question, orientation characteristics are investigated on two block sizes: 8×8 and 64×64. The results for both block sizes will be compared and interpreted.

Experimental setup

Each 256×256 textured image in the database is run through the pyramid to produce a set of $32 \times 32 = 1024$ feature-vectors (i.e., each feature-vector represents an 8×8 block in the input image) which are subsequently DFT-encoded. Only the central $30 \times 30 = 900$ feature-vectors are kept in order to avoid edge effects.

The phase of the first harmonic of the DFT components at all three scales of each of the 900 feature-vectors for each texture is next extracted. We will refer to each element of this collection of phase estimates as having the form $\rho_k e^{i\theta_k}$, where ρ_k is the response strength along a particular orientation θ_k.

We examine next the histograms of the phase estimates of each texture, at each of the 3 scales of the system. Fig. 5 shows the plots for the wood, pig and herring textures, top to bottom, respectively. A histogram of the phase estimates (θ_k for all k) is presented left with a log scatter plot shown right. The log plot is a plot of $\log \rho_k e^{i\theta_k}$. Since

$$\log \rho_k e^{i\theta_k} = \log \rho_k + i\theta_k \qquad (5)$$

the log plot is essentially a rectangularized version of a polar plot (of ρ vs. θ). In this representation the high-

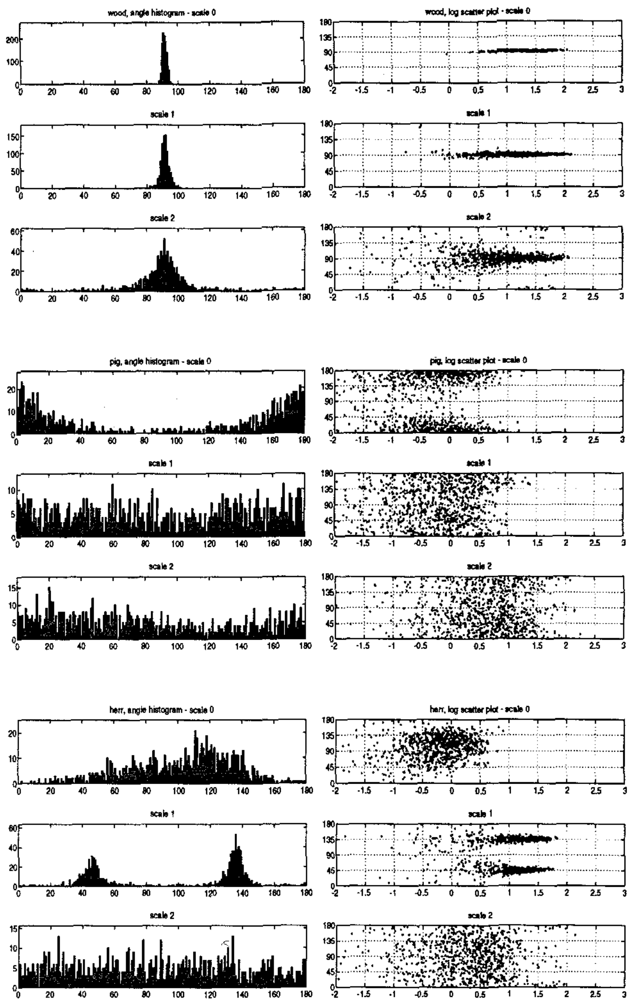
Fig. 5: Wood, pig and herring texture phase characteristics. Presented are the histogram of the phase estimates (left) and a log scatter plot (right).

energy orientations are represented by the rightmost points in the plot. By inspecting the plot it is possible to determine the peak angle in the strongest-power region.

We now describe the procedure for obtaining the data for the 64 $\times$ 64 case. We first note that there are 16 *non-overlapping* 64 $\times$ 64-pixel blocks contained within a given 256 $\times$ 256 image. The first step is to average the feature vectors over each 64 $\times$ 64 block to produce a set of 16 feature vectors. Next, a set of 16 phase estimates for all three scales are extracted. The statistics of the resulting orientation estimates are reported in Table 2.

Results

First some comments on the general aspect of texture orientation statistics. Three examples are presented in Fig. 5. An example of a strongly (single) oriented texture is the wood texture (see Fig. 5 top). We note the strong peak at 90° at all three scales. An example of a non-oriented texture is presented in the center of the figure, for the pigskin texture. Here we note the large variance in angles in each of the three scales. Looking at the log plot we note the large variance in the power axis as well, with votes for the angles spread out in the entire angle axis.

Most of the textures in the 30 texture database fall into one of the above two categories. One extreme case of interest is the herring texture which consists of two dominant orientations per scale. This case is presented in the bottom of Fig. 5. Scale 2 has in this case the most power in the oriented components, and the least variance. It is also this scale in which we see two strong peaks in orientation, at 45° and 135°. The strongest of the two peaks is used as the "reference" peak. Still, the information about the second peak is very important in the reliability of the estimated rotation angles. With the 4-dimensional DFT representation we extract a single phase component. In a single oriented texture, we are thus able to retrieve rotations within a 180° range. In the case of two dominant orientations per scale, our estimate can be accurate only to within a 90° interval.

Table 2 summarizes the orientation analysis for each of the 30 textures in our database. For each texture we have indicated the following information:
- The dominant scale from which to determine orientation. This scale is chosen as the one with the smallest standard deviation around the peak in orientation.
- The mean orientation (peak location) in that scale.
- The standard deviation of the orientation.
- In the case where two dominant orientations exist, a second mean and standard deviation is given for the second peak.

The method employed in this work for calculating the mean and standard deviation (std.) for a given angle distribution is specially designed for the 180° periodicity of the measurements. The mean angle $\bar{\theta}_k$ is computed by first finding the phase of the vector resulting from the sum of $e^{i2\theta_k}$ for all k, and then dividing this phase by 2. If the magnitude of $\sum_k e^{i2\theta_k}$ is sufficiently small, (less than 150 for the 8 $\times$ 8 case and less than 4 for the 64 $\times$ 64 case), however, a second check on $\sum_k e^{i4\theta_k}$ is performed to see if a bimodal distribution exists, as is the case for herringbone in Fig. 5, scale 1.

The std. values are obtained by first translating the histogram by $90° - \bar{\theta}_k$ to center the peak at 90° and then computing the std. of the set of θ_k's using the conventional std. definition. The standard-deviation (std.) of the angle distribution provides us with a characterization of the textures. Highly structured textures (e.g., wood(10) and jeans(5)) tend to have small std., while unstructured textures (e.g., particle-board(21) and handmade-paper(19))

	8 × 8 blocks			64 × 64 blocks		
Texture	scale	mean	std.	scale	mean	std.
1.bark	1	102.46	45.31	1	103.71	12.78
2.calf	0	81.91	11.84	1	84.35	6.88
3.cloth	0	66.69	32.93	0	68.21	10.16
4.crdbrd	1	91.04	9.21	1	91.21	1.25
5.jeans	0	87.28	2.44	0	87.40	0.54
6.grass	0	106.56	31.91	0	108.10	10.67
7.pig	0	177.45	39.18	0	176.50	9.95
8.raffia	0	175.35	23.05	0	176.46	3.77
9.water	0	92.34	5.03	0	92.56	1.48
10.wood	0	91.77	1.63	1	91.58	0.93
11.backpack	0	90.74	0.94	0	90.75	0.71
12.bookbox	0	0.63	7.99	0	0.59	1.91
13.brownbag	0	84.44	40.85	0	85.38	8.87
14.chbkcover	0	0.01	36.64	0	179.55	4.16
15.cork	0	59.60	11.34	0	60.66	4.23
16.cotcanv	0	133.69	7.10	0	134.64	1.54
17.frcanv	1	178.69	1.02	1	178.62	0.27
18.fur	0	111.41	7.88	1	110.24	7.65
19.hmpaper	1	16.69	33.16	1	16.74	8.74
20.napkin	2	135.62	20.04	2	135.61	2.73
21.prtboard	0	92.35	42.42	0	94.15	7.78
22.reptile	0	71.02	24.04	0	73.80	6.67
23.straw	0	144.26	13.23	1	143.36	9.70
24.text	1	1.15	19.26	1	1.42	1.27
25.towel	0	148.33	37.03	0	147.76	22.65
26.vinyl	0	43.64	34.07	0	44.05	14.92
27.herring	1	45.60	12.35	0	106.56	22.87
	1	135.75	10.78			
28.sand	0	109.99	37.20	0	110.76	7.58
29.wire	1	90.25	5.36	1	90.72	0.43
30.strawmat	1	165.02	9.02	1	166.02	2.23

Table 2 Orientation characteristics of textures via a histogram analysis. Shown for each texture are the dominant scale (0, 1 or 2), the mean (peak) orientation angle, and the std.

have large std. For purposes of comparsion, note that the maximum std. for a range of 180° is approximately 52°.

We can now draw comparisons between the statistics for each block size. Since there are 64 8×8 blocks contained in one 64×64 block, by averaging 8×8 tile orientations we may obtain orientation estimates whose std. is $1/\sqrt{64} = 1/8$ that of the individual 8×8 tiles. By comparing such values (Table 2, $\frac{1}{8}\cdot$ the 4th column) with the 64×64 orientation std., it is easy to conclude that averaging 8×8 tile orientations is the better method for estimating texture orientation.

5 Summary and Conclusions

We have presented a novel rotation-invariant texture recognition system together with a method for estimating the rotation-angle of textures. The features are obtained from oriented pyramid filters which present particularly good properties of "discriminability" for texture classification and are computationally efficient. The orientation estimation method is particularly reliable in that confidence measures are estimated along with the orientation. We have demonstrated state-of-the-art results both in classification and orientation estimation on a set of 30 natural and real-world textures. Future work will include extending this framework to scale-invariant recognition and scale estimation.

Acknowledgements

This work was supported in part by Pacific Bell, and in part by ARPA and ONR under grant no. N00014-92-J-1860. H. Greenspan was supported in part by an Intel fellowship. P. Perona is supported by NSF Research Initiation grant IRI 9211651 and by an NSF NYI award.

References

[1] M. Unser, "Sum and difference histograms for texture classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 1, pp. 118–125, 1986.

[2] A. Jain and F. Farrokhnia, "Unsupervised texture segmentation using gabor filters," *Pattern recognition*, vol. 24, no. 12, pp. 1167–1186, 1991.

[3] A. Lain and J. Fan, "Texture classification by wavelet packet signatures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1186–1191, 1993.

[4] H. Greenspan, R. Goodman, R. Chellappa, and C. Anderson, "Learning texture discrimination rules in a multiresolution system," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 9, September 1994.

[5] L. S. Davis, S. Johns, and J. K. Aggarwal, "Texture analysis using generalized co-occurance matrices," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, pp. 251–259, 1979.

[6] F. S. Cohen, Z. Fan, and M. A. Patel, "Classification of rotated and scaled textured images using Gaussian Markov Random Field models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 2, pp. 192–202, 1991.

[7] R. L. Kashyap and A. Khotanzad, "A model-based method for rotation invariant texture classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 4, pp. 472–481, 1986.

[8] S. J. Perantonis and P. J. G. Lisboa, "Translation, rotation, and scale invariant pattern recognition by high-order neural networks and moment classifiers," *IEEE Transactions on Neural Networks*, vol. 3, no. 2, pp. 241–251, 1992.

[9] P. Brodatz, *Textures*. New York: Dover, 1966.

[10] H. Greenspan, S. Belongie, P. Perona, R. Goodman, S. Rakshit, and C. H. Anderson, "Overcomplete steerable pyramid filters and rotation invariance," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 222–228, June 1994.

[11] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. John Wiley and Sons Inc., 1973.

[12] D. Rumelhart, G. Hinton, and R. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing* (D. Rumelhart and J. McClelland, eds.), ch. 8, Cambridge MA: MIT Press, 1986.

[13] R. M. Goodman, C. Higgins, J. Miller, and P. Smyth, "Rule-based networks for classification and probability estimation," *Neural Computation*, vol. 4, pp. 781–804, 1992.

[14] H. Greenspan, *Multi-Resolution Image Processing and Learning for Texture Recognition and Image Enhancement*. PhD thesis, California Institute of Technology, 1994.