Supervised Pixel-Based Texture Classification with Gabor Wavelet Filters

Jaime Melendez[1], Miguel Angel Garcia[2], Domenec Puig[1]

[1]Intelligent Robotics and Computer Vision Group
Department of Computer Science and Mathematics
Rovira i Virgili University
Av. Països Catalans 26, 43007 Tarragona, Spain
{jaime.melendez, domenec.puig}@urv.cat
[2]Department of Informatics Engineering
Autonomous University of Madrid
Ctra. Colmenar Viejo Km 15, 28049 Madrid, Spain
{miguelangel.garcia}@uam.es

Abstract. This paper proposes an efficient technique for pixel-based texture classification based on multichannel Gabor wavelet filters. The proposed technique is general enough to be applicable to other texture feature extraction methods that also characterize the texture around image pixels through feature vectors. During the training stage, a clustering technique is applied in order to compute a suitable set of prototypes that model every given texture pattern. Multisize evaluation windows are also utilized for improving the accuracy of the classifier near boundaries between regions of different texture. Experimental results with Brodatz compositions show the benefits of the proposed scheme in contrast with alternative approaches in terms of efficiency, memory and classification rates.

1 Introduction

Following the assumption that images are constituted by regions of different uniform texture patterns, texture classifiers aim at recognizing some or all of those patterns, and thereby, at identifying regions of interest in the image. In particular, pixel-based texture classifiers aim at recognizing the texture patterns to which the pixels of a given image belong [1][2]. In order to accomplish this task, it is necessary to compute a set of texture features by evaluating one or more texture feature extraction methods in a neighborhood of every pixel. This neighborhood is usually defined as a square window centered at that pixel.

A wide variety of texture feature extraction methods have been proposed in the literature [1][3][4][5][6][7][8]. Among them, multichannel filtering techniques based on Gabor filters have received considerable attention due to some particular properties like optimal joint localization in both the spatial and frequency domains [9], and the physiological fact that 2-D Gabor filters can approximate the simple cells in the visual cortex of some mammals [10]. Typically, in a multichannel filtering scheme, an input

This work has been partially supported by the Spanish Ministry of Education and Science under project DPI2004-07993-C03-03.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

image is decomposed into a number of filtered images, each of which contains intensity variations over a narrow range of frequencies and orientations [11]. Measures computed from these filtered images can be used as features for classification or segmentation (see [12] for some examples).

Unsupervised texture segmentation has been an active research field in the past years and most work with Gabor filters has been focused on this topic (e.g., [13][14][15][16][17]). However, texture segmentation is not directly applicable to the problem of pixel-based classification as it does not pretend the recognition of the texture patterns of interest contained in the input images. In addition, texture segmentation algorithms do not take advantage of the *a priori* knowledge corresponding to the texture patterns to be recognized. Therefore, texture classification algorithms can identify the sought texture regions better than the segmentation counterparts as they incorporate and use this additional knowledge, as it was already shown in early work by the authors [18].

Besides texture segmentation, Gabor filters have also been applied to the classification of single-textured images (e.g., [7][19]). In this case, every texture pattern is usually characterized with a single feature vector (prototype). A feature vector is then computed in the same way from the given input image to be recognized. Finally, this image is classified into the texture pattern whose prototype is closest to the image's feature vector.

However, single-textured image classifiers can not be directly applicable to the problem of pixel-based classification. To start with, in this problem it is necessary to compute a single feature vector per pixel, not per image. This implies that those vectors must be determined based on the information contained in the neighborhood of every pixel. This process must be carried out over both the input image to be classified and the sample images corresponding to the different texture patterns. Regarding the latter case, this posses the problem of deciding how many prototypes are utilized to characterize each texture pattern, as every sample image potentially generates as many prototypes as pixels. The use of so many prototypes could be prohibitive in real applications in terms of both computational time and memory usage. This problem has not been previously addressed in the literature.

The current paper provides a solution to the aforementioned problem by choosing a subset of prototypes per texture pattern. Those prototypes are obtained after applying a modified *k*-means clustering algorithm to the feature vectors extracted from the available sample images. Then, a *k*-nearest neighbor classifier (k-NN) is used to finally assign a label to every pixel within the classified image. The parameters of both the proposed variation of *k*-means and *k*-nearest neighbor are automatically selected for every texture pattern. Feature vectors are obtained by processing the images with a Gabor filter bank, following the pixel-based philosophy described above. Multisized windows are also utilized in order to improve the accuracy of the classifier near boundaries between regions of different texture. The use of multiple window sizes has already proven to be advantageous for pixel-based texture classification [20][21]. The proposed technique is effective in terms of classification rates, memory consumption and computational cost.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

The paper is organized as follows. Section 2 gives a detailed explanation of the stages of the proposed pixel-based texture classifier. Section 3 describes the technique for automatic parameter selection. Section 4 shows experimental results. Finally, conclusions and future work are given in section 5.

2 Pixel-Based Texture Classifier

The proposed texture classifier consists of four stages, namely: feature extraction, model reduction, *k*-nearest neighbor classification and post-processing. The following subsections explain them in detail.

(a) *Feature extraction.* A multichannel Gabor wavelet filtering approach has been used for the texture feature extraction stage. In particular, the design originally proposed in [7] and that has been widely used in texture classification and segmentation tasks [22][23][24] is followed. For instance, the MPEG-7 homogeneous texture descriptor is based on this design methodology, where the frequency space is partitioned into 30 channels with equal division in the angular direction and octave division in the radial direction [25].

For the task at hand, the filter bank parameters are set based on [7][26]: Low frequency 0.05, high frequency 0.4, six scales and four orientations. After filtering an input image, the texture features that will characterize every pixel and its surrounding neighborhood (window) are the mean and standard deviation of the module of Gabor wavelet coefficients. Therefore, every feature vector will have a total of $4 \times 6 \times 2 = 48$ dimensions. All dimensions are normalized between 0 and 1.

In order to take advantage of the output produced by the filter bank, the texture features mentioned above are computed for W different window sizes (W is set to 6 in this case): 3×3, 5×5, 9×9, 17×17, 33×33 and 65×65. This means that, given a set of T texture patterns to be recognized, there will be $T \times W$ sets of feature vectors at the training stage. Only W sets will be used for testing as there is only one input image. In this way, it is expected to have a better characterization of each of the texture patterns of interest.

For the setup to be complete, there is still one parameter to be configured: the kernel size. The study conducted in [26] in the context of image retrieval concludes that a kernel size of 13×13 is the best choice. However, previous experimentation suggests that a variable kernel size is better than a fixed one, at least for the proposed classification technique, since it uses different window sizes. Thus the kernel size is set to be the same as the window size.

(b) *Model reduction*. After the feature extraction stage, a 48-dimensional feature vector per image pixel is obtained. Taking into account the size of the training images associated with the various texture patterns, it is clear that the number of vectors that will model each texture pattern given a window size is enormous. While this is a requirement for a test image, in which every individual pixel has to be classified, it is not necessary nor desirable for modeling the texture patterns of interest as the core of the classifier is a k-NN algorithm. Hence the classification cost is quadratic in the number of prototypes.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

In order to reduce the number of necessary texture prototypes, a variation of the *k*-means clustering algorithm is applied, in such a way that the process is not directed by the number of clusters (this value is unknown *a priori*), but by a resolution parameter *R* that determines the size of the clusters. The algorithm also has other interesting properties, such as being deterministic, which is a desirable characteristic as explained later.

The clustering algorithm has two main stages: splitting and refinement. It proceeds as follows. For the splitting stage, suppose there are already C clusters, with their respective centroids modeling the feature space. For each cluster, the hypercube that delimits its volume is found and the length of its longest diagonal computed. If this value is greater than the resolution parameter *R*, which is defined as a fraction of the longest diagonal of the hypercube that bounds the whole feature space, then that cluster must be split in two. The value of *R* is computed as described in section 3. Therefore, after one pass of the algorithm, there will be between C (if no clusters are split) and $2C$ (in case all previous clusters are split) new clusters.

The splitting of a cluster is deterministically done by dividing the bounding box of the original cluster by its longest dimension. New centroids are set at the center of the two resulting bounding boxes. Once the splitting is completed, the refinement stage follows. It simply consists of the traditional *k*-means algorithm, but using all the available centroids as initial seeds. The algorithm iterates until convergence. The splitting and refinement stages are repeated until all clusters meet the resolution criterion. The algorithm is always initialized with a single cluster, whose centroid is the average of all the feature vectors.

This algorithm has at least three advantages over the traditional *k*-means. First, as stated earlier, there is no need for setting the desired number of clusters *a priori*. In turn, the desired resolution is introduced and the algorithm finds out the number of clusters that satisfy it. However, a resolution parameter is more intuitive than a number of clusters. Second, the algorithm always behaves in the same way given the same input points. Hence, due to its deterministic nature, there is no need to run different trials and keep the best set of centroids according to some measure, as it is done when random initialization is chosen for *k*-means. Third, it can handle a variable number of prototypes, a property that makes it more flexible when modeling a complex feature space.

A disadvantage of this kind of algorithms, however, is the time necessary to converge when the size of the sought clusters is too small (e.g., $R < 0.3$), specially when having lots of high-dimensional vectors. However, since this procedure is done during the training stage, classification times are not affected. Moreover, as will be shown later, there is no need for *R* to be that small in order to achieve good classification rates.

(c) *k-nearest neighbor classifier.* The k-NN classifier is implemented without modifications. It proceeds by comparing a vector that characterizes a test image given a window size, with all the vectors that constitute each of the T texture models for the same window size. Once all window sizes have been evaluated, a list with the K best distances among all $T \times W$ sets of possible values is obtained. Then, the most voted texture among the candidates is chosen to be the texture to which the

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

analyzed pixel belongs. If there is a tie, the winner texture will be the one with the smallest distance. The value of parameter K is computed as described in section 3.

(d) *Post-processing*. This last stage aims at removing the noisy regions that usually appear after following a classification scheme such as the one described above. It consists of two steps. First, a small window of 5×5 pixels is displaced along the entire image pixel by pixel. The enclosed region is then filled with its dominant texture, but only if the number of pixels for that texture is greater than a certain threshold. This process is repeated until the rate of changed pixels is not significant. The second step ensures that there will be no regions of uniform texture with less than a certain number of pixels by assigning them to the adjacent region with the largest number of neighboring pixels.

3 Automatic Parameter Selection

The classifier described in section 2 depends on two parameters: The resolution parameter R and the number of neighbors K. The performance of the final algorithm greatly depends on the choice of those parameters. Therefore, a simple parameter selection algorithm entirely based on the training set is also proposed.

The selection algorithm starts by using the first three stages of the classifier to compute classification rates for the sample images corresponding to all the texture patterns included in the training set (i.e., this first classification phase is performed T times). The classifier is run with different combinations of R and K. Since an exhaustive search of R and K is prohibitive, a sampled search is performed with R varying from 1.0 to 0.3 with decrements of 0.1, and K having the form 2^n, where n varies from 0 to 8 with increments of 1.

One important thing to note is that the number of prototypes for the different texture patterns and window sizes is not necessarily the same across the training set (in fact, it is always different in our experiments) as this number is not chosen *a priori*. For this reason, if the voting phase in the k-NN classifier is performed without taking this issue into account, it is possible to reach a point where all the p available votes for a texture pattern have been used (the available votes for a texture pattern coincide with the number of modeling prototypes considering all window sizes); but as K is greater than p, the remaining $K - p$ votes will only consider the textures that still can vote. Obviously, this behavior is not appropriate, so the search for K given a resolution R is stopped when K is greater than the minimum number of prototypes among the T sets of interest.

Next, the average of all classification results per combination of R and K considering all texture sample images is computed. For instance, Fig. 1 shows the average classification rate curves for patterns belonging to the first test image in Fig. 2; the best combination of R and K is marked with a circle. In general, the finer the resolution, the larger the number of neighbors necessary to achieve good classification results. However, this is only true until a certain point is reached where the maximum performance is achieved. Once this point is surpassed, classification rate degrades. Thus, the values of R and K that yield this maximum are finally kept and used to configure the proposed classifier for testing.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

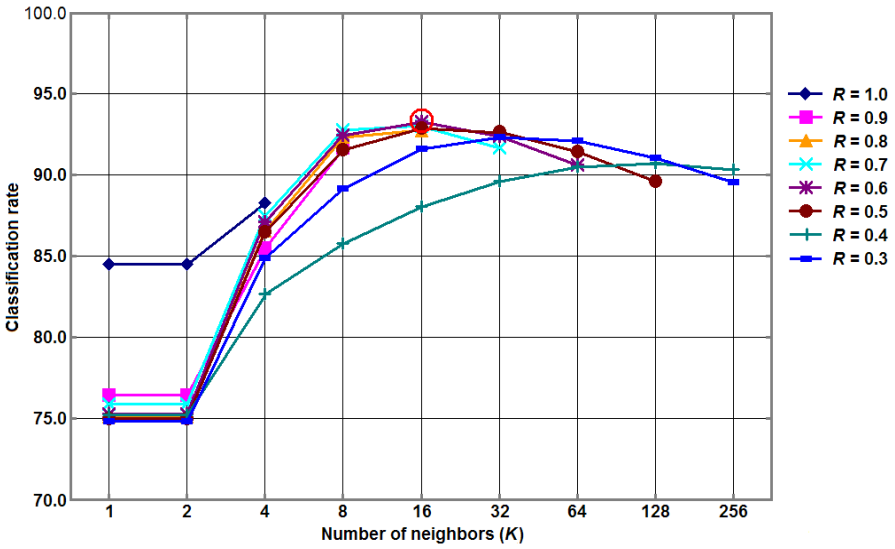

Fig. 1. Average classification rate curves for patterns belonging to the first test image in Fig. 2; the best combination of resolution R and number of neighbors K is marked with a circle

4 Experimental Results

The proposed techniques have been evaluated on composite images made of patches of well-known Brodatz textures [27]. These images are based on those used in [1]. The first column in Fig. 2 shows the five input test images. The second column of the same figure shows the ground-truth for each case, which corresponds to the perfect segmentation map.

First, to validate the proposed methodology for automatic parameter selection, the same sampled search has been performed in order to find the combination of the resolution parameter R and the number of neighbors K that yields the best classification results, but this time when considering the given test images. Then, the classification rates produced by the best combination of parameters have been compared with those obtained when using the parameters chosen by the proposed selection algorithm. Table 1 summarizes the results of this comparison.

These results indicate that the proposed selection algorithm is effective in finding a reasonable pair of parameters R and K. Indeed, it always chooses a combination that leads to a classification rate above the top ten possible scores and is able to find the best configuration in one case.

Next, a comparison in terms of classification rates, memory consumption and computation time has been performed, considering the two extreme approaches that naturally fit in the line of the proposed classifier, namely: the minimum distance classifier, which only uses both one prototype per texture pattern and window size, and one

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

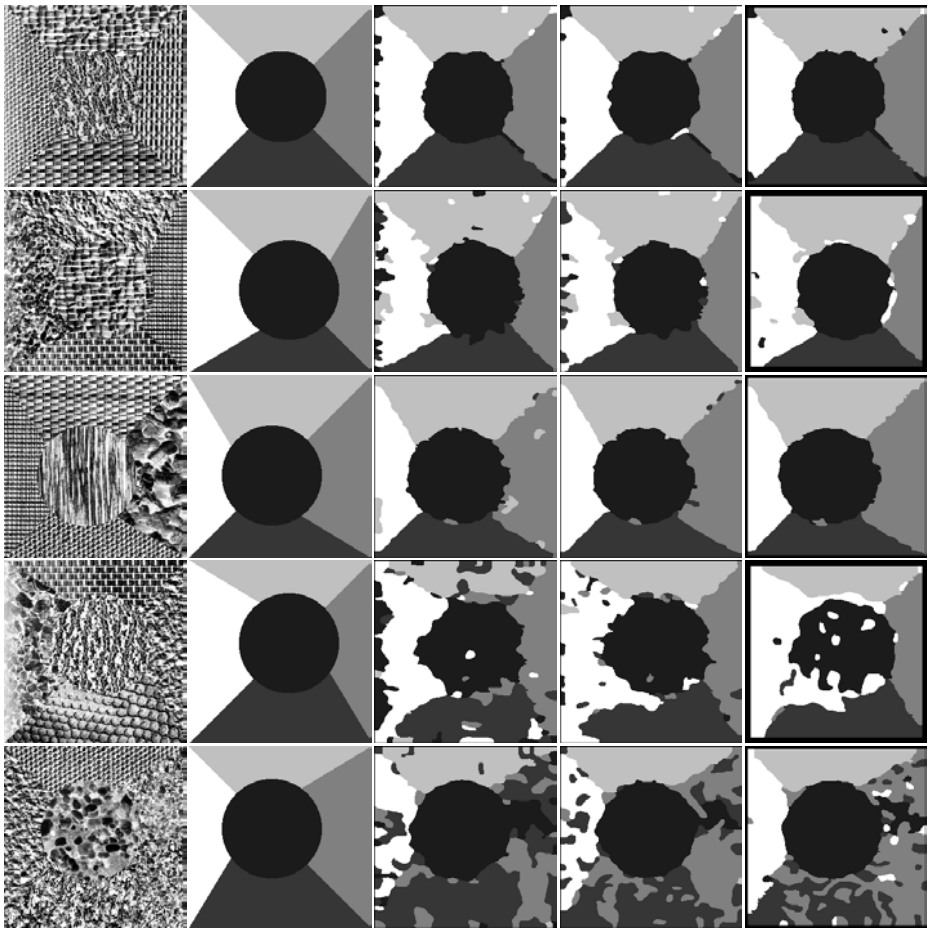

Fig. 2. Test images (*first column*), ground-truth (*second column*), and classification maps produced by the minimum distance classifier (*third column*), the proposed classifier (*fourth column*) and the classifier that stores all feature vectors as texture models (*fifth column*)

nearest neighbor; and the classifier that stores all available feature vectors given a texture pattern. A single window size is used in the latter in order to obtain a comprehensive computation time. The window size that yields the best classification rate is chosen in that case.

Results are shown in Table 2. Columns third to fifth in Fig. 1 display the segmentation maps produced by the three approaches. The black borders appearing in the maps are for those pixels that could not be classified because of the minimum window size.

From Table 2, it is clear that the proposed classifier is a good trade-off between the two extreme approaches, as the memory usage is, in average, above 2,600 times lower than the memory used when storing as many prototypes as pixels. The computational time to classify an input image is also favorable to the proposed classifier, as the latter

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

Table 1. Classification rates (%) for the best case with optimal parameters and for the classifier configured by the proposed selection algorithm

Image	Parameters and classification rate for the best case			Selected parameters and corresponding classification rate			Clas. rates ranking
1	$R = 0.8$	$K = 16$	91.1	$R = 0.6$	$K = 16$	89.5	7/51
2	$R = 0.7$	$K = 32$	86.2	$R = 0.7$	$K = 32$	86.2	1/52
3	$R = 0.5$	$K = 32$	91.6	$R = 0.5$	$K = 16$	91.3	5/51
4	$R = 0.7$	$K = 32$	78.4	$R = 0.6$	$K = 32$	77.4	5/52
5	$R = 0.7$	$K = 16$	72.9	$R = 0.8$	$K = 16$	71.1	7/53

Table 2. Performance and resource usage for alternative approaches and the proposed classifier

Image	Minimum distance classifier			Proposed classifier			"All-prototypes" modeling classifier		
	Clas. rate (%)	Time (s)	Num. of prot.	Clas. rate (%)	Time (s)	Num. of prot.	Clas. rate (%)	Time (s)	Num. of prot.
1	94.9 (82.0)	2	30	95.2 (89.5)	12	763	96.1 (94.3)	2930	307.5 x 10^3
2	90.4 (78.9)	2	30	91.7 (86.2)	7	329	92.9 (91.9)	2569	288 x 10^3
3	95.0 (81.2)	2	30	97.0 (91.3)	25	1633	97.2 (96.0)	2929	307.5 x 10^3
4	79.2 (57.1)	2	30	88.2 (77.4)	15	790	86.9 (83.4)	2578	288 x 10^3
5	73.0 (59.2)	2	30	78.6 (71.1)	4	186	80.0 (71.0)	2936	307.5 x 10^3

takes 220 times less. In terms of classification rate, the degradation experimented by the proposed classifier is small and about 5 points in average. Note that comparisons between classification rates have been done without considering the post-processing stage (see the numbers in parenthesis); in this way, the "true" output of the classifier is considered.

On the other hand, the minimum distance classifier is the fastest among the three compared alternatives as expected, but it is only 12 times faster than the proposed technique in average. It depends on the difficulty to model a given set of textures. Memory usage by the proposed classifier is certainly bigger, but it can be easily handled by current computers. In terms of classification rate, the difference is notorious in favor of the proposed classifier, specially in the fourth and fifth test images, which are consid-

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

ered the most difficult as demonstrated by the low achieved scores. This large difference when the post-processing stage is not applied also indicates that the minimum distance classifier produces extremely noisy results, and thus is less robust.

5 Conclusions

The paper presents a new pixel-based texture classifier based on Gabor wavelet filters that achieves good classification results with low computation time and a relatively reduced memory usage. The proposed classifier has been compared against the simplest strategy that stores only one prototype per texture pattern and window size, and uses only one neighbor; and with the other extreme approach, which stores all available feature vectors extracted from the training set to model a given texture.

A technique to automatically choose the configuration parameters corresponding to: (a) the resolution parameter R, which limits the size of the clusters and so the number of prototypes for k-means; and (b) the number of neighbors K to use for classification, has been developed. It has been shown that this selection algorithm determines good configuration parameters that yield classification rates very close to the optimal ones.

Future work will consist of studying better schemes for integration of different window sizes in order to reduce the dimensionality of the feature space and thus speed up the classification algorithm. Additionally, as the proposed methodology is thought to be valid for any texture method or group of texture methods whenever they produce feature vectors as output, we are planning to extend this technique in order to integrate different feature extraction methods in a coherent way.

References

1. Randen, T., Husoy, J.H.: Filtering for Texture Classification: A Comparative Study. IEEE Trans Pattern Anal Mach Intell, 21(4), (1999) 291-310
2. Bhanu, B., Fonder, S.: Functional Template-Based SAR Image Segmentation. Pattern Recogn, 37(1), (2004) 61-77
3. Reed, T.R., Hans du Buf, J.M.: A Review of Recent Texture Segmentation and Feature Extraction Techniques. Comput Vis Image Understand, 57(3), (1993) 359-372
4. Haralick, R.M., Shanmugan, K., Dinstein, I.: Textural Features for Image Classification. IEEE Trans Syst Man Cybern, 6(3), (1973) 610-622
5. Laws, K.I.: Textured Image Segmentation. USC ISG-TRI-IPI-940, Ph.D. Thesis (EE), (1980)
6. Malik, J., et al.: Contour and Texture Analysis for Image Segmentation. In: Boyer, K.L., Sarkar, S. (eds.): Perceptual Organization for Artificial Vision Systems. Kluwer Ac., (2000)
7. Manjunath, B.S., Ma, W.Y.: Texture Features for Browsing and Image Retrieval. IEEE Trans Pattern Anal Mach Intell, 18(8), (1996) 837-842
8. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns. IEEE Trans Pattern Anal Mach Intell, 24(7), (2002) 971-987
9. Daugman, J.G.: Uncertainty Relation for Resolution in Space, Spatial Frequency, and Orientation Optimized by Two-Dimensional Visual Cortical Filters. J Opt Soc Am A, 2(7), (1985) 1160-1169

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

10. Daugman, J.G.: Two-Dimensional Spectral Analysis of Cortical Receptive Field Profiles. Vis Res, 20, (1980) 847-856

11. Jain, A.K., Bhattacharjee, S.K.: Address Block Location on Envelopes Using Gabor Filters. Pattern Recogn, 25(12), (1992) 1459-1477

12. Grigorescu, S.E., Petkov, N., Kruizinga, P.: Comparison of Texture Features Based on Gabor Filters. IEEE Trans Image Process, 11(10), (2002) 1160-1167

13. Ohashi, T., Aghbari, Z., Makinouchi, A.: Fast Segmentation of Texture Image Regions Based on Hill-Climbing. IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, PACRIM 2003, 2, 848-851

14. Sagiv, C., Sochen, N.A., Zeevi, Y.Y.: Integrated Active Contours for Texture Segmentation. IEEE Trans Image Process, 15(6), (2006) 1633-1646

15. Zhu, H., Basir, O.: Proximity Measure Image Based Region Merging for Texture Segmentation through Gabor Filtering and Watershed Transform. Proceedings of the IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, 2, (2003) 742-747

16. Dong, X., Pollak, I.: Multiscale Segmentation with Vector-Valued Nonlinear Diffusions on Arbitrary Graphs. IEEE Trans Image Process, 15(7), (2006) 1993-2005

17. Muneeswaran, K. et al.: Texture Image Segmentation Using Combined Features from Spatial and Spectral Distribution. Pattern Recogn Lett, 27, (2006) 755-764

18. Garcia, M.A., Puig, D.: Pixel Classification by Divergence-Based Integration of Multiple Texture Methods and its Application to Fabric Defect Detection. In: Michaelis, B., Krell, G. (eds.): LNCS 2781, Pattern Recognition, 25th DAGM Symposium, Springer-Verlag, (2003) 132-139

19. Liu, X., Wang, D.: Texture Classification Using Local Spectral Histograms. IEEE Trans Image Process, 12(6), (2003) 661- 670

20. Garcia, M.A., Puig, D.: Supervised Texture Classification by Integration of Multiple Texture Methods and Evaluation Windows. Image Vis Comput, (2006) (http://dx.doi.org/10.1016/j.imavis.2006.05.023)

21. Puig, D., Garcia, M.A.: Automatic Texture Feature Selection for Image Pixel Classification. Pattern Recogn, 39(11), (2006) 1996-2009

22. Ma, W.Y., Manjunath, B.S.: A Texture Thesaurus for Browsing Large Aerial Photographs. J Am Soc Inform Sci, 49(7), (1998) 633-648

23. Ma, W.Y., Manjunath, B.S.: NeTra: A Toolbox for Navigating Large Image Databases. Multimed Syst, 7, (1999) 184-198

24. Ma, W.Y., Manjunath, B.S.: EdgeFlow: A Technique for Boundary Detection and Image Segmentation. IEEE Trans Image Process, 9(8), (2000) 1375-1388

25. Manjunath, B.S., et al.: Color and Texture Descriptors. IEEE Trans Circ Syst Vid Tech, 11, (2001) 703-715

26. Chen, L., Lu, G., Zhang, D.: Effects of Different Gabor Filter Parameters on Image Retrieval by Texture. Proceedings of the International Multimedia Modelling Conference, (2004) 273-278

27. Brodatz, P.: Textures: A Photographic Album for Artists and Designers. Dover & Gree Publishing Company, (1999)

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de