A comparative study of filter based texture operators using Mahalanobis distance

S. E. Grigorescu, N. Petkov, and P. Kruizinga
Institute of Mathematics and Computing Science
University of Groningen
P.O. Box 800 9700 AV Groningen The Netherlands
simona@cs.rug.nl, petkov@cs.rug.nl, peterkr@cs.rug.nl

Abstract

Texture feature extraction operators, which comprise linear filtering, eventually followed by post-processing, are considered. The filters used are Laws' masks, filters derived from well-known discrete transforms, and Gabor filters. The post-processing step comprises non-linear point operations and/or local statistics computation. The performance is measured by means of the Mahalanobis distance between clusters of feature vectors derived from different textures. The results show that post-processing improve considerably the performance of filter based texture operators.

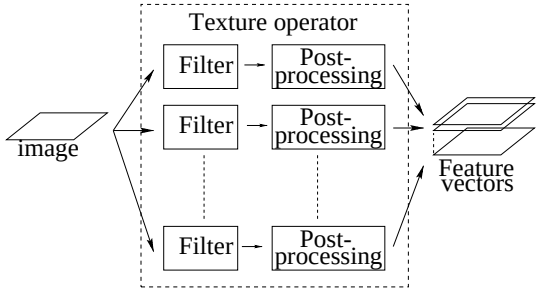

Figure 1. General structure of filter based texture operators.

1. Introduction

A number of texture feature extraction operators have been proposed in the literature which are similar in that they comprise two processing steps: (i) linear filtering followed by (ii) post-processing (Fig. 1). The post-processing step typically involves a non-linear point operation followed by the computation of some local statistics.

Laws proposed a specific type of linear filtering followed by smoothing based on local averaging applied to the absolute values of the filter output [5]. Unser used for filtering various well-known transforms like discrete sine (DST), discrete cosine (DCT), discrete even sine (DEST), discrete real even Fourier (DREFT) and discrete real odd Fourier (DROFT) transforms, and for post-processing the computation of channel variances [9]. In [2, 6], a Gabor filter bank followed by thresholding was used. For further references on texture operators, which comprise filtering followed by post-processing, see [8].

Different filters and different types of post-processing are used in the methods mentioned above. Since the results achieved with these methods are also different, a natural question arises of which filter scheme and which type of post-processing give the best results.

Previous studies in this direction typically focus on the joint performance of the linear filtering, the post-processing and a subsequent classification or segmentation stage (for references see [4, 8]). In a very thorough recent work, Randen and Husøy [8] bring systematics into the matter by fixing the type of post-processing and comparing the effect of different linear filtering schemes. Similar to previous studies they evaluate the performance of texture operators on the basis of the classification and segmentation results that are obtained when the feature vectors are fed into a given classifier. Such an evaluation has the drawback that it does not measure the performance of the texture operator only, but the joint performance of the texture operator and a subsequent classifier [4]. Moreover these studies do not evaluate the contribution of each individual processing step to the overall performance of the texture feature operator.

For evaluation of the performance of texture operators, we use a new method that was proposed elsewhere [4]. It is based on a statistical approach (Fisher criterion and Mahalanobis distance) to evaluate the capability of a feature operator to generate discriminable feature vectors for different textures. This method can be used to compare the texture operators only, regardless of any subsequent classification or segmentation operation. Furthermore, this eval-

uation method can be applied after each processing step so that the contribution of each step to the overall performance can be assessed.

The rest of the paper is organized as follows. In Section 2 an overview of the filtering schemes used in our experiments is given. In Section 3 we discuss different types of post-processing. Comparison criteria and experimental results are presented in Section 4. Finally, in Section 5 conclusions are drawn.

2. Filter banks

For linear filtering Laws [5] used a set of square convolution kernels obtained as the outer product of pairs of vectors from a set of n n-dimensional vectors. He conducted experiments for $n = 3$, 5, and 7 and obtained similar results. In our experiments we used Laws' convolution kernels derived from the following set of $n = 5$ vectors [5]:
$V_1 = [1, 4, 6, 4, 1]$, $V_2 = [-1, -2, 0, 2, 1]$, $V_3 = [-1, 0, 2, 0, -1]$, $V_4 = [-1, 2, 0, -2, 1]$, and $V_5 = [1, -4, 6, -4, 1]$; resulting in a bank of 25 filters.

Unser [9] used several different filter banks. Similar to Laws' approach, the convolution kernels proposed by Unser are computed as the outer product of pairs of vectors. Each set of vectors used in Unser's experiments corresponds to one of the following transforms: identity (shift) transform, DST, DCT, DEST, DREFT, and DROFT. More details about these transforms and the filter banks derived from them can be found in [1, 9]. We conducted experiments with all the filter banks derived from the transforms mentioned above. In all cases a set of 25 convolution kernels of size 5×5 was used for conformity with the experiments with Laws' operator. Such a filter bank leads to a 25-dimensional feature space.

In [2, 6], filter banks based on Gabor function kernels were used. In our experiments we used two sets of Gabor filters: one employing symmetric Gabor function kernels (1) and the other employing antisymmetric Gabor function kernels (2). In both cases, we used 24 Gabor filters tuned to three preferred spatial frequencies ($f = 2^{-k}\sqrt{2}, k = 0, 1,$ and 2) and eight preferred orientations ($\Theta = m\frac{\pi}{8}, m = 0 \dots 7$). For all Gabor filters the product of the preferred spatial frequency f and the standard deviation σ of the Gaussian factor was kept constant ($f\sigma = 2$) which means that the filters have a constant spatial frequency bandwidth. More details about these types of Gabor filter banks can be found in [4, 7].

Besides these Gabor filters, the filter banks used in our experiments also comprise a Gaussian filter (3) for DC component computation.

$$g^{(s)}_{f,\Theta}(i,j) = \exp\left(-\frac{x^2 + (2y)^2}{2\sigma^2}\right) \cdot \cos(2\pi f x) \quad (1)$$

$$g^{(a)}_{f,\Theta}(i,j) = \exp\left(-\frac{x^2 + (2y)^2}{2\sigma^2}\right) \cdot \sin(2\pi f x) \quad (2)$$

$$g(i,j) = \exp\left(-\frac{i^2 + j^2}{2\sigma_1^2}\right), \quad \sigma_1 = 3 \quad (3)$$

where:

$$\begin{aligned} i, j &= -2 \dots 2 \\ x &= i\cos\Theta + j\sin\Theta \\ y &= -i\sin\Theta + j\cos\Theta \end{aligned}$$

This combination of Gabor and Gaussian filters results in banks of 25 filters with a coverage of the spatial frequency domain similar to that of Laws' and Unser's filter banks.

3. Post-processing

We applied several types of post-processing to the output of the filter banks described in Section 2. The post-processing is applied individually to the output of each channel in a filter bank. The types of post-processing used in our experiments fall in two categories: one-step post-processing and two-step post-processing. The one-step post-processing comprises either a nonlinear point operation (NLPO) or computation of local statistics. The two-step post-processing comprises an NLPO followed by computation of local statistics.

In our experiments we used as NLPO thresholding and modulus computation. For local statistics we used mean, variance and standard deviation computation. In previous studies only the mean and the variance were used as statistical measures of local texture properties. We performed experiments with standard deviation as well because it offers a statistical characterization similar to that given by the variance and, at the same time, it takes values in the same range as the one which is characteristic of averaging. All the local statistics are computed in a square neighborhood of size 17×17 around each concerned pixel (similar to [5, 8]).

Below, we present the results obtained with texture operators having one of the following structures: linear filtering step only (see the first column in Table 1), linear filtering followed by one-step post processing (see the second and third columns in Table 1 and the first columns in Tables 2, 3, and 4), and linear filtering followed by two-step post-processing (see the second and third columns in Tables 2, 3, and 4).

In all tables, χ and $|.|$ designate thresholding and modulus computation, respectively.

Transforms/filters		NLPO	
		χ	$\lvert\cdot\rvert$
Identity transform	1.31	1.31	1.31
Laws' filters	1.31	2.10	2.62
DST	1.31	1.85	2.59
DCT	1.31	1.84	2.29
DEST	1.31	1.64	2.24
DREFT	1.31	2.11	3.04
DROFT	1.31	1.83	2.43
Lin. sym. Gabor filters	1.31	1.54	1.80
Lin. antisym. Gabor filters	1.21	2.03	2.14

Table 1. Average Mahalanobis distances for features obtained by linear filtering only (first column) and for features obtained by linear filtering followed by an NLPO (second and third columns).

Transforms/filters		NLPO	
		χ	$\lvert\cdot\rvert$
Identity transform	2.29	2.29	2.29
Laws' filters	2.30	9.42	10.13
DST	2.26	9.10	9.98
DCT	2.28	8.92	9.30
DEST	2.27	8.21	9.08
DREFT	2.28	9.62	10.23
DROFT	2.28	9.19	9.95
Lin. sym. Gabor filters	2.42	11.80	11.96
Lin. antisym. Gabor filters	2.61	15.26	16.28

Table 2. Average Mahalanobis distances for features obtained by computing the local mean of the output of: (i) linear filters (first column) and (ii) linear filters followed by an NLPO (second and third columns).

Transforms/filters		NLPO	
		χ	$\lvert\cdot\rvert$
Identity transform	1.47	1.47	1.47
Laws' filters	6.54	6.14	5.84
DST	6.72	6.64	6.78
DCT	6.80	6.81	6.84
DEST	6.60	6.35	6.37
DREFT	6.77	6.36	6.02
DROFT	6.63	6.41	6.53
Lin. sym. Gabor filters	10.76	10.14	11.16
Lin. antisym. Gabor filters	10.87	9.65	9.55

Table 3. Average Mahalanobis distances for features obtained by computing the local variance of the output of: (i) linear filters (first column) and (ii) linear filters followed by an NLPO (second and third columns).

Transforms/filters		NLPO	
		χ	$\lvert\cdot\rvert$
Identity transform	1.56	1.56	1.56
Laws' filters	8.99	8.52	8.00
DST	9.28	9.12	9.18
DCT	9.14	9.23	8.94
DEST	8.95	8.35	8.39
DREFT	9.33	8.81	8.29
DROFT	9.08	8.60	8.75
Lin. sym. Gabor filters	15.50	14.43	15.23
Lin. antisym. Gabor filters	15.38	13.74	13.61

Table 4. Average Mahalanobis distances for features obtained by computing the local standard dev. of the output of: (i) linear filters (first column) and (ii) linear filters followed by an NLPO (second and third columns).

4. Comparison

The feature vectors computed in different points of a texture image using a given operator are not identical; they rather form a cluster in the multi-dimensional feature space. It is desirable that feature clusters, which correspond to different textures, be separable. The larger this separability the better the operator used for texture feature computation. As a measure of cluster separability we use the Mahalanobis distance [3, 4].

We evaluated the performance of the various operators presented in the previous sections by looking at the pairwise separability of the clusters of feature vectors obtained from nine test images, each containing a single oriented texture (Fig. 2). To build a cluster, one thousand feature vectors were taken at random positions from each texture image. The Mahalanobis distance was computed for every pair of texture images and for each operator. For brevity, only the averages of the 36 Mahalanobis distances computed with each operator are given here: each number in the Tables 1 to 4 is the average of 36 Mahalanobis distances between nine clusters of feature vectors computed with a given method.

Regarding the statistical interpretation of the given values in terms of misclassification probability, two clusters having normal distributions with the same standard deviation and a value of the Mahalanobis distance of 1.31 – see, for example, the first column of Table 1 – overlap for 25.89%. For Mahalanobis distance of 2 the overlap is 14.76%, while for values of the Mahalanobis distance greater than 5 the overlap is smaller than 0.1% which means

that the clusters are well separable.

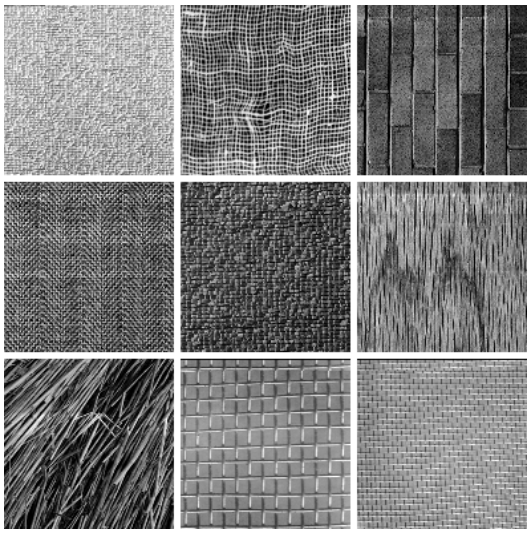

Figure 2. Test images used in the experiments.

5. Conclusions

From Tables 1, 2, 3, and 4 one can deduce the significance of each processing step of the concerned texture operators. The main conclusion that can be drawn is that post-processing can improve considerably the performance of filter based texture operators. Post-processing schemes, which include the computation of local statistics, perform better than those comprising an NLPO only do. Depending on the type of local statistics, the presence of an NLPO in a post-processing scheme can improve or degrade results. Computing the local mean, for instance, gives very good results if it is preceded by an NLPO (compare second and third column of Table 2 with the first columns of Tables 1 and 2). Similar improvement is achieved by local standard deviation computation. In the latter case, no NLPO is needed between the filtering step and the local standard deviation computation: such an intermediate step has negative effect on the results (compare the first column of Table 4 with the second and third columns of the same table).

Computing the standard deviation gives better results than computing the variance (compare Tables 3 and 4). In both cases, the use of an intermediate NLPO degrades the results (compare first columns with the second and the third columns in Tables 3 and 4).

On average, taking the modulus as a point operation and computing the local mean gives the best results (see last column of Table 2).

With a given type of post-processing the different filtering schemes lead to different results. The identity transform gives the worst results: the chance of misclassification for two normally distributed clusters with a Mahalanobis distance of 1.56 is 21.79%. With an average Mahalanobis distance of 9 which corresponds to a misclassification of $10^{-7}\%$, all sinusoidal transforms and the Laws' filter bank give comparable results. Still, they are worse than those obtained with Gabor filters, where the average Mahalanobis distance of 15 corresponds to a misclassification probability of $10^{-12}\%$.

The differences between the results obtained with Gabor filter banks and the other filter banks can be due to the differences in the tiling of the frequency domain. All of the sinusoidal transforms result in filters with a rectangular power spectrum with only two possible orientations (0^o, 90^o) while Gabor filters have an elliptical power spectrum with eight possible orientations. Taking in consideration that the test material contains oriented textures only, it seems plausible to explain the good results obtained with Gabor filters by their orientation selectivity.

References

[1] A. Jain. A sinusoidal family of unitary transforms. *IEEE-PAMI*, 1(4):356–365, 1979.

[2] A. Jain and F. Farrokhnia. Unsupervised texture segmentation using gabor filters. *Pattern Recognition*, 24(12):1167–1186, 1991.

[3] P. Kruizinga and N. Petkov. Grating cell operator features for oriented texture. In A. Jain, S. Venkatesh, and B. Lovell, editors, *Proc. of the Int. Conf. on Pattern Recognition*, pages 1010–1014, Brisbane Australia, 1998.

[4] P. Kruizinga and N. Petkov. Non-linear operator for oriented texture. *IEEE Trans. on Image Processing*, 8(10):1395–1407, 1999.

[5] K. Laws. Rapid texture identification. In *Proc. SPIE Conf. on Image Processing for Missile Guidance*, volume 238, pages 376–380, 1980.

[6] J. Malik and P. Perona. Preattentive texture discrimination with early vision mechanisms. *Journal of the Optical Society of America, A*, 7(5):923–932, 1990.

[7] N. Petkov and P. Kruizinga. Computational models of visual neurons specialized in the detection of periodic and aperiodic oriented visual stimuli: Bar and grating cells. *Biological Cybernetics*, 76:83–96, 1997.

[8] T. Randen and J. H. Husoy. Filtering for texture classification: a comparative study. *IEEE-PAMI*, 21(4):291–310, 1999.

[9] M. Unser. Local linear transforms for texture measurements. *Signal Processing*, 11(1):61–79, 1986.