A Statistical Approach to Material Classification Using Image Patch Exemplars

Manik Varma and Andrew Zisserman

Abstract—In this paper, we investigate material classification from single images obtained under unknown viewpoint and illumination. Our objective, in this paper, is the classification of materials from their appearance in single images taken under unknown viewpoint and illumination conditions. The task is difficult as materials typically exhibit large intraclass, and small interclass, variability (see Fig. 1) and there are no widely applicable yet mathematically rigorous models which account for such transformations. The task is made even more challenging if no a priori knowledge about the imaging conditions is available.

Early interest in the texture classification problem focused on the preattentive discrimination of texture patterns in binary images [3], [26], [27], [38]. Later on, this evolved to the classification of textures in gray-scale images with synthetic 2D variations [20], [22], [47]. This, in turn, has been superseded by the problem of classifying real-world textures with 3D variations due to changes in camera pose and illumination [6], [11], [29], [31], [44], [54]. Currently, efforts are on to extend the problem to the accurate classification of entire texture categories rather than of specific material instances [9], [23]. Another trend investigates how regularity information can be exploited for the preattentive discrimination of texture patterns in binary images [3], [26], [27], [38]. Later on, this evolved to the classification of textures in gray-scale images with synthetic 2D variations [20], [22], [47]. This, in turn, has been superseded by the problem of classifying real-world textures with 3D variations due to changes in camera pose and illumination [6], [11], [29], [31], [44], [54]. Currently, efforts are on to extend the problem to the accurate classification of entire texture categories rather than of specific material instances [9], [23]. Another trend investigates how regularity information can be exploited for the preattentive discrimination of texture patterns in binary images [3], [26], [27], [38].

A common thread through this evolution has been the success that filter bank-based methods have had in tackling the problem. As the problem has become more difficult, such methods have coped by building richer representations of filter responses. The use of large support filter banks to extract texture features at multiple scales and orientations has gained wide acceptance.

However, in this paper, we question the dominant role that filter banks have come to play in the field of texture classification. Instead of applying filter banks, we develop an alternative image patch representation based on the joint distribution of pixel intensities in a neighborhood.

We first investigate the advantages of this image patch representation empirically. The VZ algorithm [54] gives one of the best 3D texture classification results on the Columbia-Utrecht database using the Maximum Response 8 (MR8) filters with support as large as 49 × 49 pixels square. We demonstrate that substituting the new patch-based representation in the VZ algorithm leads to the following two results: that 1) very good classification performance can be achieved using extremely compact neighborhoods (starting from as small as 3 × 3) and that 2) for any fixed size of the neighborhood, image patches lead to superior classification as compared to filter banks with the same support. The superiority of the image patch representation is empirically demonstrated by classifying all 61 materials present in the Columbia-Utrecht database and showing that the results outperform the VZ algorithm using the MR8 filter bank. Results are also presented for the UIUC [30], San Francisco [29], and Microsoft Textile [42] databases.

We then discuss theoretical reasons as to why small image patches can correctly discriminate between textures with large global structure and also challenge the popular belief that filter bank features are superior for classification as compared to the source image patches from which they were derived. Finally, we present results on texture...
synthesis and denoising to reinforce the fact that the new representation can be learned accurately, even in high-dimensional, image patch space. A preliminary version of this work appeared in [52].

2 Background

Texture research is generally divided into five canonical problem areas:

1. synthesis,
2. classification,
3. segmentation,
4. compression, and
5. shape from texture.

The first four areas have come to be heavily influenced by the use of wavelets and filter banks, with wavelets being particularly effective at compression, while filter banks have led the way in classification, segmentation, and synthesis.

The success in these areas was largely due to learning a fuller statistical representation of filter bank responses. It was fuller in three respects: First, the filter response distribution was learned (as opposed to recording just the low order moments of the distribution); second, the joint distribution, or co-occurrence, of filter responses was learned (as opposed to independent distributions for each filter); and third, simply more filters were used than before to measure texture features at many scales and orientations.

These filter response distributions were learned from training images and represented by clusters or histograms. The distributions could then be used for classification, segmentation, or synthesis. For instance, classification could be achieved by comparing the distribution of a novel texture image to the model distributions learned from the texture classes. Similarly, synthesis could be achieved by constructing a texture having the same distribution as the target texture. As such, the use of filter banks has become ubiquitous and unquestioned.

However, even though there has been ample empirical evidence to suggest that filter banks and wavelets can lead to good performance, not much rigorous theoretical justification has been provided as to their optimality or even, for that matter, their necessity for texture classification, synthesis, or segmentation. In fact, the supremacy of filter banks for texture synthesis was brought into question by the approach of Efros and Leung [15]. They demonstrated that superior synthesis results could be obtained using local pixel neighborhoods directly, without resorting to large-scale filter banks. In a related development, Zalesny and Van Gool [58] also eschewed filter banks in favor of a Markov random field (MRF) model. More recently, and following the same trend, [56] showed that small patches can provide an alternative to filter banks for texture edge detection and segmentation.

Both [15], [58] put MRFs firmly back on the map as far as texture synthesis was concerned. Efros and Leung gave a computational method for generating a texture with similar MRF statistics to the original sample, but without explicitly learning or even representing these distributions. Zalesny and Van Gool, using a subset of all available cliques present in a neighborhood, showed that it was possible to learn and sample from a parametric MRF model given enough computational power.

In this paper, it is demonstrated that the second of the canonical problems, texture classification, can also be tackled effectively by employing only local neighborhood distributions, with representations inspired by MRF models.

2.1 The Columbia-Utrecht Database

In this section, we describe the Columbia-Utrecht (CUReT) database [12] and its level of difficulty for single image classification. The database contains images of 61 materials and includes many surfaces that we might commonly see in our environment. It has textures that are rough, those which have specularities, exhibit anisotropy, are man-made, and many others. The variety of textures present in the database is shown in Fig. 2.

Each of the materials in the database has been imaged under 205 different viewing and illumination conditions. The effects of specularities, interreflections, shadowing, and other surface normal variations are plainly evident and can be seen in Fig. 1 where their impact is highlighted due to varying imaging conditions. This makes the database far more challenging for a classifier than the often-used Brodatz collection where all such effects are absent.

While the CUReT database has now become a benchmark and is widely used to assess classification performance, it
also has some limitations. These are mainly to do with the way the images have been photographed and the choice of textures. For the former, there is no significant scale change for most of the materials and very limited in-plane rotation. With regard to choice of texture, the most serious drawback is that multiple instances of the same texture are present for only a few of the materials, so intraclass variation cannot be thoroughly investigated. Hence, it is difficult to make generalizations. Nevertheless, it is still one of the largest and toughest databases for a texture classifier to deal with.

All 61 materials present in the database are included in the experimental setup used in Sections 4 and 5. For each material, there are 118 images where the azimuthal viewing angle is less than 60 degrees. Out of these, 92 images are chosen for which a sufficiently large region of texture is visible across all materials. The remaining images are not included as they do not have large enough foreground texture regions where large support filter banks can be applied. A central 200 × 200 region is cropped from each of the selected images and the remaining background discarded. The selected regions are converted to gray scale and then intensity normalized to have zero mean and unit standard deviation. Thus, no color information is used in any of the experiments and we make ourselves invariant to affine changes in the illuminant intensity. The cropped CUReT database has a total of 61 × 92 = 5,612 images. Out of these, 46 images per class are randomly chosen for training and the remaining 46 per class are chosen for testing. The cropped CUReT database can be downloaded from [1].

3 A REVIEW OF THE VZ CLASSIFIER

The classification problem being tackled is the following: Given an image consisting of a single texture obtained under unknown illumination and viewpoint, categorize it as belonging to one of a set of prelearned texture classes. Leung and Malik’s influential paper [31] established much of the framework for this area—filter response textons, nearest neighbor classification using the $\chi^2$ statistic, testing on the CUReT database, etc. Later algorithms, such as the BFH classifier [11] and the VZ classifier [54], have built on this paper and extended it to classify single images without compromising accuracy. In turn, [6], [9], [23] have achieved even superior results by keeping the MR8 filter bank representation of the VZ algorithm but replacing the nearest neighbor classifier with SVMs or Gaussian-Bayes classifiers.

The VZ classifier [54] is divided into two stages: a learning stage, where texture models are learned from training examples by building statistical descriptions of filter responses, and a classification stage, where novel images are classified by comparing their distributions to the learned models.

In the learning stage, training images are convolved with a chosen filter bank to generate filter responses. These filter responses are then aggregated over images from a texture class and clustered. The resulting cluster centers form a dictionary of exemplar filter responses which are called textons. Given a texton dictionary, a model is learned for a particular training image by labeling each of the image pixels with the texton that lies closest to it in filter response space. The model is the normalized frequency histogram of pixel texton labelings, i.e., an $S$-vector of texton probabilities for the image, where $S$ is the size of the texton dictionary. Each texture class is represented by a number of models corresponding to training images of that class.

In the classification stage, the set of learned models is used to classify a novel (test) image into 1 of the 61 texture classes. This proceeds as follows: The filter responses of the test image are generated and the pixels labeled with textons from the texton dictionary. Next, the normalized frequency histogram of texton labelings is computed to define an $S$-vector for the image. A nearest neighbor classifier is then used to assign the texture class of the nearest model to the test image. The distance between two normalized frequency histograms is measured using the $\chi^2$ statistic, where

$$
\chi^2(x, y) = \frac{1}{2} \sum_{x, y} \frac{(x_i - y_i)^2}{x_i + y_i}.
$$

The performance of six filter banks was contrasted in [50]. These include four filter banks based on the Maximum Response filter set (BFS, MR8, MR4, and MRS4), the filter bank of Schmid (S) [43], and the filter bank of Leung and Malik (LM) [31], which was also used by Cula and Dana [11]. It was demonstrated that the rotationally invariant, multiscale, MR8 filter bank (described below) yields better results than any of the other filters. Hence, in this paper, we present comparisons with the MR8 filter bank.

3.1 Filter Bank

The MR8 filter bank consists of 38 filters but only 8 filter responses (Fig. 3). The filters include a Gaussian and a Laplacian of a Gaussian (LOG) filter both at scale $\sigma = 10$, an edge (first derivative) filter at six orientations and three scales and a bar (second derivative) filter also at six orientations and the same three scales $(\sigma_x, \sigma_y) = \{(1, 3), (2, 6), (4, 12)\}$. The response of the isotropic filters (Gaussian and LOG) is used directly. However, in a manner somewhat similar to [41], the responses of the oriented filters (bar and edge) are “collapsed” at each scale by using only the maximum filter responses across all orientations. This gives eight filter responses in total and ensures that the filter responses are rotationally invariant. The MR4 filter bank only employs the $(\sigma_x, \sigma_y) = (4, 12)$ scale. Another 4D variant, MRS4, achieves rotation and scale invariance by selecting the maximum response over both orientation and scale [50]. Matlab code for generating these filters, as well as the LM and S sets, is available from [2].
VARMA AND ZISSERMAN: A STATISTICAL APPROACH TO MATERIAL CLASSIFICATION USING IMAGE PATCH EXEMPLARS

The best results for MR8 are 46 models per texture. This will henceforth be referred to as the VZ Benchmark. The bank achieves an accuracy rate of 96.93 percent while merging textons as in [31]. On the contrary, larger texton dictionaries tended to give better performance (see [53], [54]) and no other changes are made to the classifier. Hence, in the first stage of learning, all of the image patches from the selected training images in a texture class are aggregated and clustered. The set of cluster centers from all of the classes comprises the texton dictionary. The textons now represent exemplar image patches rather than exemplar filter responses (see Fig. 4).

Fig. 5. The only difference between the Joint and the VZ MR8 representations is that the source image patches are used directly in the Joint representation as opposed to the derived filter responses in VZ MR8.

97.43 percent obtained when a dictionary of 2,440 textons is used, with 40 textons being learned per class.

4 THE IMAGE PATCH-BASED CLASSIFIERS

In this section, we investigate the effect of replacing filter responses with the source image patches from which they were derived. The rationale for doing so comes from the observation that convolution to generate filter responses can be rewritten as an inner product between image patch vectors and the filter bank matrix. Thus, a filter response is essentially a lower-dimensional projection of an image patch onto a linear subspace spanned by the vector representation of the individual filters (obtained by row reordering each filter mask).

The VZ algorithm is now modified so that filter responses are replaced by their source image patches. Thus, the new classifier is identical to the VZ algorithm except that, at the filtering stage, instead of using a filter bank to generate filter responses at a point, the raw pixel intensities of an $N \times N$ square neighborhood around that point are taken and row reordered to form a vector in an $N^2$-dimensional feature space. All preprocessing and postprocessing steps are retained (images are made zero mean and unit variance, while patch vectors are contrast normalized using Weber’s law) and no other changes are made to the classifier. Hence, in the first stage of learning, all of the image patches from the selected training images in a texture class are aggregated and clustered. The set of cluster centers from all of the classes comprises the texton dictionary. The textons now represent exemplar image patches rather than exemplar filter responses (see Fig. 4).

However, the model corresponding to a training image continues to be the histogram of texton frequencies and novel image classification is still achieved by nearest neighbor matching using the $\chi^2$ statistic. This classifier will be referred to as the Joint classifier. Fig. 5 highlights the main difference in approach between the Joint classifier and the MR8-based VZ classifier.

We also design two variants of the Joint classifier—the Neighborhood classifier and the MRF classifier. Both of these are motivated by the recognition that textures can often be considered realizations of a MRF. In an MRF

Authorized licensed use limited to: Rutgers University. Downloaded on October 20, 2009 at 10:39 from IEEE Xplore. Restrictions apply.
the same dictionary of 610 textons. Then, for each of the image patch, only the eight neighbors of every central pixel depends only on its neighborhood. Formally,
\[
p(I(x_c)|I(x), \forall x \neq x_c) = p(I(x_c)|I(x), \forall x \in N(x_c)),
\]
where \(x_c\) is a site in the 2D integer lattice on which the image \(I\) has been defined and \(N(x_c)\) is the neighborhood of that site. In our case, \(N\) is defined to be the \(N \times N\) square neighborhood (excluding the central pixel). Thus, although the value of the central pixel is significant, its distribution is conditioned on its neighbors alone. The Neighborhood and MRF classifiers are designed to test how significant this conditional probability distribution is for classification.

For the Neighborhood classifier, the central pixel is discarded and only the neighborhood is used for classification. Thus, the Neighborhood classifier is essentially the Joint classifier retrained on feature vectors drawn only from the set of \(N\), i.e., the set of \(N \times N\) image patches with the central pixel left out. For example, in the case of a \(3 \times 3\) image patch, only the eight neighbors of every central pixel are used to form feature vectors and textons.

For the MRF classifier, we go to the other extreme and, instead of ignoring the central pixel, explicitly model
\[
p(I(x_c), I(N(x_c))),
\]
i.e., the joint distribution of the central pixels and its neighbors. Up to now, textons have been used to implicitly represent this joint PDF. The representation is implicit because, once the texton frequency histogram has been formed, neither the probability of the central pixel nor the probability of the neighborhood can be recovered straightforwardly by summing (marginalizing) over the appropriate textons. Thus, the texton representation is modified slightly so as to make explicit the central pixel’s PDF within the joint and to represent it at a finer resolution than its neighbors (in the Neighborhood classifier, the central pixel PDF was discarded by representing it at a much coarser resolution using one bin).

To learn the PDF representing the MRF model for a given training image, the neighbors’ PDF is first represented by textons as was done for the Neighborhood classifier—i.e., all pixels but the central are used to form feature vectors in an \(N^2 - 1\)-dimensional space, which are then labeled using the same dictionary of 610 textons. Then, for each of the \(S_N\) textons in turn (\(S_N = 610\) is the size of the neighborhood texton dictionary), a 1D distribution of the central pixels’ intensity is learned and represented by an \(S_C\) bin histogram. Thus, the representation of the joint PDF is now an \(S_N \times S_C\) matrix. Each row is the PDF of the central pixel for a given neighborhood intensity configuration as represented by a specific texton. Fig. 6 highlights the differences between MRF models and models learned using the Joint representation. Using this matrix, a novel image is classified by comparing its MRF distribution to the learned model MRF distributions by computing the \(\chi^2\) statistic over all elements of the \(S_N \times S_C\) matrix.

Table 1 presents a comparison of the performance of the Joint, Neighborhood and MRF classifiers when tested on the CURET database (see Section 2.1 for experimental setup details). Image patches of size \(3 \times 3\), \(5 \times 5\), and \(7 \times 7\) are tried while using a dictionary of 610 textons. For the Joint classifier, it is remarkable to note that classification results of over 95 percent are achieved using patches as small as \(3 \times 3\). In fact, the classification result for the \(3 \times 3\) neighborhood is actually better than the results obtained by using the MR4 (91.70 percent), MRS4 (94.23 percent), LM (94.65 percent), or S (95.22 percent) filter banks. This is strong evidence that there is sufficient information in the joint distribution of the nine intensity values (the central pixel and its eight neighbors) to discriminate between the texture classes. For the Neighborhood classifier, as shown in Table 1b, there is almost no significant variation in classification performance as compared to using all the pixels in an image patch. Classification rates for \(N = 5\) are slightly better when the central pixel is left out and marginally poorer for the cases of \(N = 3\) and \(N = 7\). Thus, the joint distribution of the neighbors is largely sufficient for classification. Table 1c presents a comparison of the performance of the Joint and Neighborhood classifiers to the MRF classifier when a resolution of 90 bins is used to store the central pixels’ PDF. As can be seen, the MRF classifier does better than both the Joint and Neighborhood classifiers. What is also interesting is that the performance of the MRF classifier using \(7 \times 7\) patches (97.47 percent) is at

![Figure 6. MRF texture models as compared to those learned using the Joint representation.](Image)

**Table 1**

<table>
<thead>
<tr>
<th>(N \times N)</th>
<th>Joint Classifier (%)</th>
<th>Neighborhood Classifier (%)</th>
<th>MRF with 90 bins (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 (\times) 3</td>
<td>95.33</td>
<td>94.90</td>
<td>95.87</td>
</tr>
<tr>
<td>5 (\times) 5</td>
<td>95.62</td>
<td>95.97</td>
<td>97.22</td>
</tr>
<tr>
<td>7 (\times) 7</td>
<td>96.19</td>
<td>96.08</td>
<td>97.47</td>
</tr>
</tbody>
</table>

(a) All of the pixels in an image patch are used to form vectors in an \(N^2\) feature space. (b) All but the central pixel are used (i.e., an \(N^2 - 1\) space). (c) The MRF classifier where 90 bins are used to represent the joint neighborhood and central pixel PDF. A dictionary of 610 textons learned from all 61 textures is used throughout. Notice that the performance using these small patches is as good as that achieved by the multiorientation, multiscale, MR8 filter bank with 49 \(\times\) 49 support (96.93 percent using 610 textons and 97.43 percent using 2,440 textons).
least as good as the best performance achieved by the multiscale MR8 filter bank with support $49 \times 49$ (97.43 percent using 2,440 textons).

This result showing that image patches can outperform filters raises the important question of whether filter banks are actually providing beneficial information for classification, for example, perhaps by increasing the signal to noise ratio or by extracting useful features. We first address this issue experimentally by determining the classification performance of filter banks across many different parameter settings and seeing if performance is ever superior to equivalent patches.

In order to do so, the performance of the VZ classifier using the MR8 filter bank (VZ MR8) is compared to that of the Joint, Neighborhood, and MRF classifiers as the size of the neighborhood is varied. In each experiment, the MR8 filter bank is scaled down so that the support of the largest filters is the same as the neighborhood size. Once again, we emphasize that the MR8 filter bank is chosen as its performance is better than all the other filter banks studied. Fig. 7 plots the classification results. It is apparent that, for any given size of the neighborhood, the performance of VZ MR8 using 610 textons is worse than that of the Joint or even the Neighborhood classifiers also using 610 textons. Similarly, VZ MR8 Best is always inferior, not just to MRF Best but also to MRF. To assess statistical significance, we repeat the experiment for VZ MR8 610 and Joint 610 over a thousand random partitionings of the training and test set. The results are given in Table 2 and show that, for each neighborhood size, the Joint classifier outperforms VZ MR8. The results are statistically significant since the $p$-value was always zero. This would suggest that using all the information present in an image patch is more beneficial for classification than relying on lower-dimensional responses of a preselected filter bank. A classifier which is able to learn from all the pixel values is superior.

**TABLE 2**

<table>
<thead>
<tr>
<th></th>
<th>9 $\times$ 9($)</th>
<th>11 $\times$ 11($)</th>
<th>13 $\times$ 13($)</th>
<th>15 $\times$ 15($)</th>
<th>17 $\times$ 17($)</th>
<th>19 $\times$ 19($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VZ MR8 610</td>
<td>95.06 $\pm$ 0.41</td>
<td>95.57 $\pm$ 0.38</td>
<td>95.92 $\pm$ 0.37</td>
<td>96.16 $\pm$ 0.37</td>
<td>96.30 $\pm$ 0.37</td>
<td>96.37 $\pm$ 0.36</td>
</tr>
<tr>
<td>Joint 610</td>
<td>96.38 $\pm$ 0.35</td>
<td>96.58 $\pm$ 0.34</td>
<td>96.63 $\pm$ 0.35</td>
<td>96.89 $\pm$ 0.33</td>
<td>97.11 $\pm$ 0.32</td>
<td>97.17 $\pm$ 0.32</td>
</tr>
</tbody>
</table>

Statistical significance: the mean and standard deviation for the VZ MR8 classifier with 610 textons and joint classifier also with 610 textons are reported as a function of the neighborhood size. In each case, results are reported over a thousand random partitionings of the training and test set. The performance of the Joint classifier is better than that of VZ MR8 for every one of the thousand splits for each neighborhood size. This resulted in the $p$-value being zero in each case indicating that the results are statistically significant.
These results demonstrate that a classification scheme based on MRF local neighborhood distributions can achieve very high classification rates and can outperform methods which adopt large-scale filter banks to extract features and reduce dimensionality. Before turning to discussing theoretical reasons as to why this might be the case, we first explore how issues such as rotation and scale impact the image patch classifiers.

5 Scale, Rotation, and Other Data Sets

Three main criticisms can be leveled at the classifiers developed in the previous section. First, it could be argued that the lack of significant scale change in the CURET textures might be the reason why image patch-based classification outperforms the multiscale MR8 filter bank. Second, the image patch representation has a major disadvantage in that it is not rotationally invariant. Third, the reason why small image patches do so well could be because of some quirk of the CURET data set and that classification using small patches will not generalize to other databases. In this section, each of these three issues is addressed experimentally and it is shown that the image patch representation is as robust to scale changes as MR8, can be made rotationally invariant, and generalizes well to other data sets.

5.1 The Effect of Scale Changes

To test the hypothesis that the image patch representation will not do as well as the filter bank representation in the presence of scale changes, four texture classes were selected from the CURET database (material numbers 2, 11, 12, and 14) for which additional scaled data is available (as material numbers 29, 30, 31, and 32). The materials are shown in Fig. 8.

Fig. 8. The top row shows one image each from material numbers 2, 11, 12, and 14 from the CURET database. The middle row shows the same textures scaled synthetically by a factor of 2, while the bottom row shows the textures scaled naturally (as material numbers 29, 30, 31, and 32).

Table 3 shows the results of the experiments. It also tabulates the results when the experiments are repeated but this time with the images being scaled synthetically by a factor of 2.

### Table 3

<table>
<thead>
<tr>
<th>Texture Class</th>
<th>Naturally Scaled</th>
<th>Synthetically Scaled ×2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Original (%)</td>
<td>Original + Scaled (%)</td>
</tr>
<tr>
<td>MRF</td>
<td>93.48</td>
<td>65.22</td>
</tr>
<tr>
<td>MR8</td>
<td>81.25</td>
<td>62.77</td>
</tr>
</tbody>
</table>

Comparison of classification results of the MRF and VZ MR8 classifiers for scaled data. Models are learned either from the original textures only or the original + scaled textures while classifying both texture types. In each case, the performance of the MRF classifier is at least as good as that using the multiscale MR8 filter bank.

5.2 Incorporating Rotational Invariance

The fact that the image patch representation developed so far is not rotationally invariant can be a serious limitation. However, it is straightforward to incorporate invariance into the representation. There are several possibilities: 1) Find the dominant orientation of the patch (as is done in the MR filters) and measure the neighborhood relative to this orientation, 2) marginalize the intensities weighted by the orientation distribution over angle, and 3) add rotated patches to the training set so as to make the learned decision boundaries rotation invariant [46]. In this paper, we implement option 1 and, instead of using an $N \times N$ square patch, the neighborhood is redefined to be circular with a given radius. Table 4 lists the results for the Neighborhood textures.
and MRF classifiers using circular neighborhoods with radius 3 pixels (corresponding to a $7 \times 7$ patch) and 4 pixels ($9 \times 9$ patch).

Using the rotationally invariant representation, the Neighborhood classifier with a dictionary of 610 textons achieves 96.36 percent for a radius of 3 pixels and 96.47 percent for a radius of 4 pixels. This is slightly better than that achieved by the same classifier using the standard (noninvariant) representation with corresponding $7 \times 7$ and $9 \times 9$ patches. The rates for the rotationally invariant MRF classifier are 97.07 percent and 97.25 percent using 610 textons and 45 bins. These results are slightly worse than those obtained using the standard representation. However, the fact that such high classification percentages are obtained strongly indicates that rotation invariance can be successfully incorporated into the image patch representation.

5.3 Results on Other Data Sets

We now show that image patches can also be used to successfully classify textures other than those present in the CUReT database. It is demonstrated that the Joint classifier with patches of size $3 \times 3$, $5 \times 5$, and $7 \times 7$ is sufficient for classifying the Microsoft Textile [42] and San Francisco [29] databases. For the UIUC database [30], while $9 \times 9$ patches already yield good results, the best results are obtained by patches of size $17 \times 17$. While the MRF classifier leads to the best results in general, we show that, on these databases, the Joint classifier already achieves very high performance.

5.3.1 The Microsoft Textile Database

This has 16 folded materials with 20 images available of each taken under diffuse artificial lighting (see Fig. 9 for an example). The impact of non-Lambertian effects is plainly visible (as it is in the CUReT database). Furthermore, the variations in pose and the deformations of the textured surface make it an interesting database to analyze.

For this database, the experimental setup is kept identical to the original setup of the authors. Fifteen images were randomly selected from each of the 16 texture classes to form the training set. While all of the training images were used to form models, textons were learned from only three images per texture class. Various sizes of the texton dictionary $S = 16 \times K$ were tried, with $K = 10, \ldots, 40$ textons learned per textile. The test set comprised a total of 80 images. Table 5 shows the variation in performance of the Joint classifier with neighborhood size $N$ and texton dictionary size $S$.

As can be seen, excellent results are obtained using very small neighborhoods. In fact, only a single image is misclassified using $5 \times 5$ patches (see Fig. 9). These results reinforce the fact that very small patches can be used to classify textures with global structure far larger than the neighborhoods used (the image resolutions are $1,024 \times 768$).

5.3.2 The San Francisco Database

This database has 37 images of outdoor scenes taken on the streets of San Francisco. Konishi and Yuille have segmented the images by hand [29] into six classes: Air, Building, Car, Road, Vegetation, and Trunk. We work with the given segmentations and our goal is to classify each of the regions selected by Konishi and Yuille. Note that, since each image has multiple texture regions present in it, the global image mean is not subtracted as was done in previous cases.

A single image is chosen for training the Joint classifier. Fig. 10 shows the selected training image and its associated hand-segmented regions. All the rest of the 36 images are kept as the test set. Performance is measured by the proportion of pixels that are labeled correctly during classification of the hand-segmented regions. Using this setup, the Joint classifier achieves an accuracy rate of 97.9 percent, i.e., almost all the pixels are labeled correctly in the 36 test images. Fig. 11 shows an example of a test image and the regions that were classified in it. This result again validates the fact that small image patches can be

![Fig. 9. Misclassifications in the Microsoft Textile database: Only a single image of Black Linen (a) is misclassified as an instance of Black Pseudo Silk (b) by the Joint classifier using $5 \times 5$ patches.](image)

<table>
<thead>
<tr>
<th>Size of Texton Dictionary $S$</th>
<th>$N \times N$</th>
<th>160 (%)</th>
<th>320 (%)</th>
<th>480 (%)</th>
<th>640 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$3 \times 3$</td>
<td>96.82</td>
<td>96.82</td>
<td>96.82</td>
<td>96.82</td>
<td></td>
</tr>
<tr>
<td>$5 \times 5$</td>
<td>99.21</td>
<td>99.21</td>
<td>99.21</td>
<td>99.21</td>
<td></td>
</tr>
<tr>
<td>$7 \times 7$</td>
<td>96.93</td>
<td>97.62</td>
<td>96.82</td>
<td>97.62</td>
<td></td>
</tr>
</tbody>
</table>

The joint classifier performs excellently on the Microsoft Textile database—only a single image is misclassified using $5 \times 5$ patches. These results reinforce the fact that very small patches can be used to classify textures with global structure far larger than the neighborhoods used (the image resolutions are $1,024 \times 768$).
used to successfully classify textured images. In fact, using small patches is particularly appealing for databases such as the San Francisco set because large-scale filter banks will have problems near region boundaries and will also not be able to produce many measurements for small, or irregularly shaped, regions.

5.3.3 The UIUC Database
The UIUC texture database [30] has 25 classes and 40 images per class. The database contains materials imaged under significant viewpoint variations. Fig. 12 shows examples of the materials in the database and also highlights the extreme scale and viewpoint changes.

We compare the performance of the Joint classifier to the performance of the rotation invariant MR8, and rotation and scale invariant MRS4, filter banks. We also compare results to the nearest neighbor state-of-the-art, affine invariant methods of Lazebnik et al. (LSP) [30] and bi-Lipschitz invariant methods of Xu et al. (XJF) [57] and Varma and Garg (VG) [51]. For the Joint classifier, we adopt the rotationally invariant patch-based representation developed in Section 5.2. While circular patches of radius 4 already give good results, the best results are obtained by patches of radius 8. The best filter bank-based results are obtained using filters of support $49/C^2$. Texton dictionaries of size 2,500 are used for all the classifiers. To assess classification performance, $M$ training images are randomly chosen per class, while the remaining $40 - M$ images per class are taken to form the test set. Table 6 presents classification results averaged over a thousand random splits of the training and test sets (the results for the LSP, XJF, and VG methods are taken from [51, Table 1]).

The performance of the Joint classifier is significantly superior (with $p$-value 0 in all experiments) to that of MR8 and MRS4. The performance gap increases as fewer and fewer images are used for training. This runs contrary to traditional expectation and bolsters the claim that the patch-based representation is not necessarily adversely affected by large-scale variations as compared to multiscale filter banks. Surprisingly, the performance of the Joint classifier is also superior to the state-of-the-art bi-Lipschitz and affine invariant classifiers of [30], [51], [57].

6 Why Does Patch-Based Classification Work?

The results of the previous sections have demonstrated two things. First, neighborhoods as small as $3 \times 3$ can lead to very good classification results even for textures whose global structure is far larger than the local neighborhoods used. Second, classification using image patches is superior to that using filter banks with equivalent support. In this section, we discuss some of the theoretical reasons as to why these results might hold.

6.1 Classification Using Small Patches

The results on the CUReT, San Francisco, and Microsoft Textile databases show that small image patches contain sufficient information to discriminate between different textures. One explanation for this is illustrated in Fig. 13. In Fig. 13a, three images are selected from the Limestone and Ribbed Paper classes of the CUReT data set and scatter plots of their gray-level co-occurrence matrix are shown for the displacement vector $(2, 2)$ (i.e., the joint distribution of the top left and bottom right pixel in every $3 \times 3$ patch). Notice how the distributions of the two images of Ribbed Paper can easily be associated with each other and distinguished from the distribution of the Limestone image. Another example in Fig. 13b shows the same trend. Thus, $3 \times 3$ neighborhood distributions can contain sufficient information for successful discrimination.

To take a more analytic example, consider two functions $f(x) = A \sin(\omega f t + \delta)$ and $g(x) = A \sin(\omega g t + \delta)$, where $\omega_f$ and $\omega_g$ are small so that $f$ and $g$ have large structure. Even though $f$ and $g$ are very similar (they are essentially the same function at different scales), it will be seen that they are easily distinguished by the Joint classifier using only
two point neighborhoods. Fig. 14 illustrates that, while the intensity distributions of $f$ and $g$ are identical, the distributions of their derivatives, $f_x$ and $g_x$, are not. Since derivatives can be computed using just two points, these functions can be distinguished by looking at two point neighborhoods alone.

In a similar fashion, other complicated functions, such as triangular and sawtooth waves, can be distinguished using compact neighborhoods. Furthermore, the Taylor series expansion of a polynomial of degree $2N - 1$ immediately shows that a $[-N, +N]$ neighborhood contains enough information to determine the value of the central pixel. Thus, any function which can be locally approximated by a cubic polynomial can actually be synthesized using a $[-2, 2]$ neighborhood. Since, in general, synthesis requires much more information than classification, it is therefore expected that more complicated functions can still be distinguished just by looking at small neighborhoods. This illustrates why it is possible to classify very large-scale textures using small patches.

There also exist entire classes of textures which cannot be distinguished on the basis of local information alone. One such class is comprised of textures made up of the same textons and with identical first-order texton statistics, but which differ in their higher-order statistics. To take a simple example, consider texture classes generated by the repeated tiling of two textons (a circle and a square for instance) with sufficient spacing in between so that there is no overlap between textons in any given neighborhood. Then, any two texture classes which differ in their tiling pattern but have identical frequencies of occurrence of the textons will not be distinguished on the basis of local information alone.

---

**TABLE 6**

<table>
<thead>
<tr>
<th>$M$</th>
<th>Joint (%)</th>
<th>MR8 (%)</th>
<th>MRS4 (%)</th>
<th>VG [51] (%)</th>
<th>LSP [30] (%)</th>
<th>XIF [57] (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>97.83±0.66</td>
<td>92.94±1.06</td>
<td>90.29±1.26</td>
<td>95.40±0.92</td>
<td>93.62±0.97</td>
<td>93.04</td>
</tr>
<tr>
<td>15</td>
<td>96.94±0.77</td>
<td>91.16±1.11</td>
<td>88.47±1.25</td>
<td>94.09±0.98</td>
<td>92.42±0.99</td>
<td>91.11</td>
</tr>
<tr>
<td>10</td>
<td>95.18±0.94</td>
<td>88.29±1.32</td>
<td>85.43±1.34</td>
<td>91.64±1.18</td>
<td>90.17±1.11</td>
<td>88.79</td>
</tr>
<tr>
<td>05</td>
<td>90.17±1.44</td>
<td>81.12±1.74</td>
<td>78.44±1.77</td>
<td>85.35±1.69</td>
<td>84.77±1.54</td>
<td>82.99</td>
</tr>
</tbody>
</table>

UIUC results as the number of training images $M$ is varied. Means and standard deviations have been computed over 1,000 random splits of the training and test set.

---

Fig. 13. Information present in $3 \times 3$ neighborhoods is sufficient to distinguish between textures. (a) The top row shows three images drawn from two texture classes, Limestone and Ribbed Paper. The bottom row shows scatter plots of $I(x)$ against $I(x + (2, 2))$. On the left are the distributions for Limestone and Ribbed Paper 1, while on the right are the distributions for all three images. The Limestone and Ribbed Paper distributions can easily be distinguished and, hence, the textures can be discriminated from this information alone. Another example is shown in (b) with the same notation.
However, the fact that classification rates of nearly 98 percent have been achieved using extremely compact neighborhoods on three separate data sets indicates that real textures are not as simplistic as this.

The arguments in this section indicate that small patches might be effective at texture classification. The arguments do not imply that the performance of small patches is superior to that of arbitrarily large filter banks. However, in the next section, arguments are presented as to why filter banks are not superior to equivalent sized patches.

6.2 Filter Banks Are Not Superior to Image Patches

We now turn to the question of why filter banks do not provide superior classification as compared to their source image patches. To fix the notation, \( f_+ \) and \( f_- \) will be used to denote filter response vectors generated by projecting \( N \times N \) image patches \( i_+ \) and \( i_- \), of dimension \( d = N^2 \), onto a lower dimension \( N_f \) using the filter bank \( F \). Thus,\n
\[
f_{\pm N_f \times 1} = F_{N_f \times d} i_{\pm k_{\times 1}}. \tag{3}\n\]

In the following discussion, we will focus on the properties of linear (including complex) filter banks. This is not a severe limitation as most popular filters and wavelets tend to be linear. Nonlinear filters can also generally be decomposed into a linear filtering step followed by nonlinear postprocessing. Furthermore, since one of the main arguments in favor of filtering comes from dimensionality reduction, it will be assumed that \( N_f < d \), i.e., the number of filters must be less than the dimensionality of the source image patch. Finally, it should be clarified that, throughout the discussion, performance will be measured by classification accuracy rather than the speed with which classification is carried out. While the time complexity of an algorithm is certainly an important factor and can be critical for certain applications, our focus here is on achieving the best possible classification results.

The main motivations which have underpinned filtering (other than biological plausibility) are: 1) dimensionality reduction, 2) feature extraction at multiple scales and orientations, and 3) noise reduction and invariance. Arguments from each of these areas are now examined to see whether filter banks can lead to better performance than image patches.

6.2.1 Dimensionality Reduction

Two arguments have been used from dimensionality reduction. The first, which comes from optimal filtering, is that an optimal filter can increase the separability between key filter responses from different classes and is therefore beneficial for classification [25], [40], [49]. The second argument, from statistical machine learning, is that reducing the dimensionality is desirable because of better parameter estimation (improved clustering) and also due to regularization effects which smooth out noisy filter responses and prevent overfitting [5], [8], [14], [21]. We examine both arguments in turn to see whether such factors can compensate for the inherent loss of information associated with dimensionality reduction. For a more comprehensive discussion of these issues, please refer to [5], [45].

**Increasing separability.** Since convolution with a linear filter is equivalent to linearly projecting onto a lower-dimensional space, the choice of projection direction determines the distance between the filter responses. Suppose we have two image patches \( i_+ \) and \( i_- \), with filter responses \( f_+ \) and \( f_- \). Then, the distance between \( f_+ \) and \( f_- \) is clearly less than the distance between \( i_+ \) and \( i_- \) (where the rows of \( F \) span the hyperplane orthogonal to the projection direction). The choice of \( F \) affects the separation between \( f_+ \) and \( f_- \) and the optimum filter maximizes it, in the manner of a Fisher Linear Discriminant, but the scaled distance between the projected points cannot exceed the original. This holds true for many popular distance measures, including the euclidean, Mahalanobis, and the signed perpendicular distance [4] (analogous results hold when \( F \) is not orthogonal). It is also well known [28] that under Bayesian classification, the Bayes error either increases or remains at least as great when the dimensionality of a problem is reduced by linear projection. However, the fact that the Bayes error has increased for the low-dimensional filter responses does not mean the classification is necessarily worse. This is because of issues related to noise and overfitting which brings us to the second argument from dimensionality reduction for the superiority of filter banks.

**Improved parameter estimation.** The most compelling argument for the use of filters comes from statistical machine learning, where it has often been noted that dimensionality reduction can lead to fewer training samples being needed for improved parameter estimation (better clustering) and can also regularize noisy data and thereby prevent overfitting. The assumptions underlying these claims are that textures occupy a low-dimensional subspace of image patch space and if the patches could be projected onto this true subspace (using a filter bank), then the dimensionality of the problem would be reduced without resulting in any information loss. This would be particularly beneficial in cases where only a limited amount of training data is available as the higher-dimensional patch representation would be prone to overfitting (see Fig. 15).
While these are undoubtedly sound claims, there are three reasons why they might not lead to the best possible classification results. The first is due to the great difficulty associated with identifying a texture’s true subspace (in a sense, this itself is one of the holy grails of texture analysis). More often than not, only approximations to this true subspace can be made and these result in a frequent loss of information when projecting downward.

The second counter argument comes from the recent successes of boosting [48] and kernel methods [45]. Dimensionality reduction is necessary if one wants to accurately model the true texture PDF. However, both boosting and kernel methods have demonstrated that, for classification purposes, a better solution is to actually project the data nonlinearly into an even higher (possibly infinite) dimensional space, where the separability between classes is increased. Thus, the emphasis is on maximizing the distance between the classes and the decision boundary rather than trying to accurately model the true texture PDF (which, though ideal, is impractical). In particular, the kernel trick, when implemented properly, can lead to both improved classification and generalization without much associated overhead and with none of the associated losses of downward projection. The reason this argument is applicable in our case is because it can be shown that with some minor modifications, can be thought of as a Mercer kernel [55]. Thus, the patch-based classifiers take the distribution of image patches and project it into the much higher-dimensional $\chi^2$ space where classification is carried out. The filter bank-based VZ algorithm does the same but it first projects the patches onto a lower-dimensional space which results in a loss of information. This is the reason why the performance of filter banks, such as MR8, is consistently inferior to their source patches.

The third argument is an engineering one. While it is true that clustering is better and that parameters are estimated more accurately in lower-dimensional spaces, Domingos and Pazzani [13] have shown that even gross errors in parameter estimation can have very little effect on classification. This is illustrated in Fig. 16, which shows that, even though the means and covariance matrices of the true likelihood are estimated incorrectly, 98.6 percent of the data is still correctly classified as the probability of observing the data in much of the incorrectly classified regions is vanishingly small.

Another interesting result, which supports the view that accurate parameter estimation is not necessary for accurate classification, is obtained by selecting the texton dictionary at random (rather than via $K$-Means clustering) from among the filter response vectors. In this case, the classification result for VZ MR8 drops by only 5 percent and is still well above 90 percent. A similar phenomenon was observed in [19] when Mean-Shift clustering was used to approximate

---

**Fig. 15.** Projecting the data onto lower dimensions can have a beneficial effect when not much training data is available. A nearest neighbor classifier misclassifies a novel point in the original, high-dimensional space, but classifies it correctly when projected onto the $x$-axis. This problem is mitigated when there is a lot of training data available. Note that it is often not possible to know a priori the correct projection directions. If it were, then misclassifications in the original, high-dimensional space can be avoided by incorporating such knowledge into the distance function. Indeed, this can even lead to superior classification unless all of the information along the remaining dimensions is noise.

**Fig. 16.** Incorrect parameter estimation can still lead to good classification results: The true class conditional densities of two classes (defined to be Gaussians) are shown in (a) along with the MAP decision boundary obtained using equal priors (dashed red curves). In (b), the estimated likelihoods have gross errors. The estimated means have relative errors of 100 percent and the covariances are estimated as being diagonal, leading to a very different decision boundary. Nevertheless, the probability of misclassification (computed using the true Gaussian distributions for the probability of occurrence and integrating the classification error over the entire 2D space) is just 1.4 percent. Thus, 98.6 percent of all points submitted to the classifier will be classified correctly despite the poor parameter estimation.
the filter response PDF. Thus, accurate parameter estimation
does not seem to be essential for accurate texture
classification and the loss due to inaccurate parameter
estimation in high dimensions might be less than the loss
associated with projecting into a lower-dimensional sub-
space even though clustering may be improved.

6.2.2 Feature Extraction
The main argument from feature extraction is that many
features at multiple orientations and scales must be
detected accurately for successful classification. Furthemore,
studies of early vision mechanisms and preattentive
texture discrimination have suggested that the detected
features should look like edges, bars, spots, and rings. These
have most commonly come to be implemented using Gabor
or Gaussian filters and their derivatives. However, results
from the previous sections have shown that a multiscale,
multiorientation large support filter bank is not necessary.
Small image patches can also lead to successful classifica-
tion. Furthermore, while an optimally designed bank might
be maximizing some measure of separability in filter space,
it is hard to argue that “off-the-shelf” filters such as MR8,
LM, or S (whether biologically motivated or not) are the
best for any given classification task. In fact, as has been
demonstrated, a classifier which learns
from all the input
data present in an image patch should do better than one
which depends on these predefined features bases.

It can also be argued that patch-based features might not
perform well in the presence of large, nonlinear illumi-
nation changes. Edge-based features, computed by threshold-
ing filter responses, might be more stable in this case.
However, the same effect can be achieved by putting a
suitable prior over patches while learning the texton
dictionary. Thresholding to find edges would then corre-
spond to the vector quantization step in our algorithm. Note
that the CURET database already contains images taken
under significant illumination variation (see examples in
Fig. 1 as well as images of aluminum foil and leaves in the
database). Nevertheless, it was noticed that the patch-based
classifiers gave better results than filter banks even when
only a small number of training images were used. On a
related note, patch-based methods are also beginning to
provide viable alternatives to filter banks for texture edge
detection and segmentation tasks [56].

6.2.3 Noise Reduction and Invariance
Most filters have the desirable property that, because of
their large smoothing kernels (such as Gaussians with large
standard deviation), they are fairly robust to noise. This
property is not shared by image patches. However,
preprocessing the data can solve this problem. For example,
the classifiers developed in this paper rely on vector
quantization of the patches into textons to help cope with
noise. This can actually provide a superior alternative to
filtering, because even though filters reduce noise, they also
smooth the high-frequency information present in the
signal. Yet, as has been demonstrated in the 3 × 3 patch
case, this information can be beneficial for classification.
Therefore, if image patches can be denoised by preprocess-
ing or quantization without the loss of high-frequency
information, then they should provide a superior represen-
tation for classification as compared to filter banks.

Virtually the same methods can be used to build
invariance into the patch representation as are used for
filters—without losing information by projecting onto lower
dimensions. For example, patches are preprocessed and
made to have zero mean and unit standard deviation to
achieve invariance to affine transformations in the illumi-
nant’s intensity. Similarly, as discussed in Section 5.2, to
achieve rotational invariance, the dominant orientation can
be determined and used to orient the patch. This does have
the drawback of being potentially unstable if the dominant
direction cannot be determined accurately. For instance,
corners have two dominant orientations and, in the
presence of noise, can be transformed incorrectly upon
reduction to the canonical frame. One solution to the
problem could be to discard such ambiguous patches
altogether. Another would be to take appropriately
weighted linear combinations of all transformations. At
the other extreme, one can even include many transformed
copies of the patch (for instance, all rotated versions) in the
training set to overcome this problem.

It should be noted that the arguments presented in this
section do not imply that any arbitrary patch-based
classifier is better than every filter bank-based one. We
gave a constructive example of how local patches can make
classification mistakes which can be avoided by larger-scale
filter banks. Furthermore, Fig. 15 also illustrates how a filter
bank, designed using prior knowledge, can perform better
if there is limited training data. Instead, our arguments
focus on two points. First, small local patch-based classifiers
can give surprisingly good results in many real world
situations. Second, given equivalent prior knowledge,
patches should do as well, if not better, than filter banks
with equivalent support.

7 SYNTHESIS AND DENOISING
In this section, we investigate how accurately distributions
in high-dimensional spaces can be learned given limited
training data. The concern is that the high-dimensional,
patch-based representation is incapable of capturing a
texture’s statistics as compared to lower-dimensional filter
responses. Most of the arguments in Sections 5.3 and 6.2
revolve around this central issue. While demonstrating
good classification results is one way of addressing the
concern, another way is to synthesize or denoise textures by
sampling from the learned PDF. If the reconstruction is
adequate, then that provides additional evidence that our
PDF representation is sufficiently accurate. Therefore, in
this section, we also demonstrate that our MRF representa-
tion can be used to synthesize and denoise textures.

7.1 Texture Synthesis
Our texture synthesis algorithm is very similar to [15]
except for the fact that we explicitly learn and sample from
the texture’s PDF. Given an input texture block to be
synthesized, the first step is to learn its MRF statistics using
the matrix representation of the PDF of image patches. The
parameters that can be varied are N, the size of the
neighborhood, and K the number of textons used to
represent the neighborhood distribution. The central pixel PDF is stored in 256 bins in this case. Next, to synthesize the texture, the input block is initially tiled to the required dimensions. The output image is synthesized from this tiled image by repeating the following procedure for each pixel. The pixel’s neighborhood is mapped to the closest texton and the pixel's value then replaced by randomly sampling from the learned central pixel distribution given the texton (i.e., neighborhood). Multiple passes are made over the image until a desired synthesis is obtained. Results are shown in Fig. 17. As can be seen, the synthesized textures are very similar to the originals, thereby indicating that the MRF representation can form an adequate representation of the texture’s statistics. Note that no higher-order statistics or image regularity information [34], [36] has been used and this can only improve results. Furthermore, once the representation has been learned explicitly, the synthesized image can be generated quickly as exhaustive image search [15] is no longer required. On the other hand, a disadvantage of our method is that the neighborhood is fixed while it can be adapted in [15].

7.2 Texture Denoising

Our denoising algorithm is inspired by [7]. In the first stage, the MRF representation of the noisy images is learned exactly as had been done for synthesis. This involves clustering all \( N \times N \) patches of the noisy image into \( K \) textons and then learning the central pixel PDF given each of the \( K \) textons. Denoising is carried out by labeling each patch in the noisy image by its closest texton and then replacing the central pixel in the patch by the median of the corresponding central pixel distribution (other statistics, such as the mean or the mode, can also be used if found to be more appropriate for a given noise model). As such, the algorithm is identical to our synthesis algorithm except that the pixels in the denoised image are generated by choosing the median of the appropriate central pixel PDF rather than sampling from it.

Fig. 18 shows some typical results using \( N = 7 \) and \( K = 1,000 \). The central pixel PDF was stored using 256 bins. No attempt was made to optimize these parameters. In contrast to [7], no Gaussian smoothing of the neighborhood was performed. Furthermore, statistics were computed over the entire image rather than over a restricted subregion around the central pixel being denoised.

Again, the good results indicate that the high-dimensional image patch PDF has been learned accurately enough and that one does not have to reduce dimensionality using a filter bank in order to capture a texture’s statistics.

8 Conclusions

We have described a classification method based on representing textures as a set of exemplar patches. This
representation has been shown to be superior to one based on filter banks.

Filter banks have a number of disadvantages compared to smaller image patches: First, the large support they require means that far fewer samples of a texture can be learned from training images (there are many more 3 × 3 neighborhoods than 50 × 50 in a 100 × 100 image). Second, the large support is also detrimental in texture segmentation, where boundaries are localized less precisely due to filter support straddling region boundaries. A third disadvantage is that the blurring (e.g., Gaussian smoothing) in many filters means that fine local detail can be lost.

The disadvantage of the patch representation is the quadratic increase in the dimension of the feature space with the size of the neighborhood. This problem may be tackled by using a multiscale representation. For instance, an image pyramid could be constructed and patches taken from several layers of the pyramid if necessary. An alternative would be to use large neighborhoods but store the pixel information away from the center at a coarser resolution. A scheme such as Zalesny and Van Gool’s [58] could also be implemented to determine which long-range interactions were important and use only those cliques.

Before concluding, it is worthwhile to reflect on how the image patch algorithms and their results relate to what others have observed in the field. In particular, [16], [32], [40] have all noted that, in their segmentation and classification tasks, filters with small support have outperformed the same filters at larger scales. In addition, [56] uses small 5 × 5 patches to detect texture edges. Thus, there appears to be emerging evidence that small support is not necessarily detrimental to performance.

It is also worth noting that the “new” image patch algorithms, such as the synthesis method of Efros and Leung and the Joint classifier developed in this paper, have actually been around for quite a long time. For instance, Efros and Leung discovered a strong resemblance between their algorithm and that of [17]. Furthermore, both the Joint classifier and Efros and Leung’s algorithm are near identical in spirit to the work of Popat and Picard [39]. The relationship between the Joint classifier and Popat and Picard’s algorithm is particularly close as both use clustering to learn a distribution over image patches which then forms a model for novel texture classification. Apart from the choice of neighborhoods, the only minor differences between the two methods are in the representation of the PDF and the distance measure used during classification. Popat and Picard use a Gaussian mixture model with diagonal covariances to represent their PDF, while the texton representation used in this paper can be thought of as fitting a spherical Gaussian mixture model via $K$-Means. During classification, Popat and Picard use a naive Bayesian method which, for the Joint classifier, would equate to using nearest neighbor matching with KL divergence instead of the $\chi^2$ statistic as the distance measure [53].

Certain similarities also exist between the Joint classifier and the MRF model of Cross and Jain [10]. In particular, Cross and Jain were the first to recommend that $\chi^2$ over the distribution of central pixels and their neighbors could be used to determine the best fit between a sample texture and a model. Had they actually used this for classification rather than just model validation of synthesized textures, the two algorithms would have been very similar apart from the functional form of the PDFs learned (Cross and Jain treat the conditional PDF of the central pixel given the neighborhood as a unimodal binomial distribution).

Thus, alternative approaches to filter banks have been around for quite some time. Perhaps the reason that they did not become popular then was due to the computational costs required to achieve good results. For instance, the synthesis results in [39] are of a poor quality, which is perhaps why their theory did not attract the attention it deserved. However, with computational power being readily accessible today, MRF and image patch methods are outperforming filter bank-based methods.

ACKNOWLEDGMENTS

The authors are grateful to Alexey Zalesny, David Forsyth, and Andrew Fitzgibbon for many discussions and valuable feedback, Varun Gulshan for helping with the CURTe and UIUC experiments, Alan Yuille for making available the San Francisco database, and Antonio Criminisi and Silvio Savarese for the Microsoft Textile database. The investigations reported in this contribution have been supported by a University of Oxford Graduate Scholarship in Engineering at Jesus College, an ORS award, and the European Union (FP5-project “CogViSys,” IST-2000-29404).

REFERENCES
