1. INTRODUCTION

In image analysis texture is a fundamental property that is often used for recognizing objects or materials. There are numerous applications for texture based pattern recognition including surface inspection, medical image analysis, and remote sensing [1]. Texture analysis has been also widely studied in computer vision. However, in most cases noise is assumed to be the main source of error that can degrade the image quality. In reality, image acquisition is also subject to other degradations such as blurring that usually originates from camera motion or misfocused optics. The performance of pattern recognition systems may deteriorate significantly if the image descriptors computed are blur sensitive. One way to deal with this problem is to deblur the image via sharpening or deconvolution, but usually these methods are computationally expensive and they may introduce notable artifacts. Because the restored image is not necessarily required in classification, it is more desirable to design recognition algorithms that are robust to these degradations.

The focus of this paper is on blur insensitive image analysis using texture information. It is quite surprising that there are not many texture analysis methods in the literature that are considered to be insensitive to blur although the problem of image blurring is so apparent. A blur robust descriptor based on color constancy was proposed in [2]. Also, blur invariant moments [3] or the modified Fourier phase [4] could be used in principle, but they are mainly intended for global object recognition, not local texture analysis.

In this paper, we investigate a new blur insensitive texture classification method called local phase quantization (LPQ) that was recently proposed in [5]. The method is based on the quantized phase of the discrete Fourier transform computed locally for small image patches. The LPQ operator uses a similar binary encoding scheme as the LBP method [7], where the descriptor is formed from a histogram of codewords computed for some image region. The descriptor obtained is insensitive to centrally symmetric blur that includes linear motion, out of focus, and atmospheric turbulence blur [6]. In particular, we consider different ways of computing the LPQ operator in this paper. We also analyze the performance of these LPQ variants experimentally using artificially blurred images of texture. For comparison, we perform the same experiments with the LBP method and a method based on a bank of Gabor filters [8] that can both be considered to represent state-of-the-art in texture analysis.

First we introduce the conditions under which the DFT phase is invariant to blur in Section 2. Then, Section 3 describes the basic LPQ operator and its decorrelation scheme. Section 4 proposes three approaches for computing the local phase information. Section 5 contains the experimental results and Section 6 finally presents the concluding remarks.

2. BLUR INvariance of PHASE sPectRum

In digital image processing, the discrete model for spatially invariant blurring of an original image \( f(x) \) resulting in an observed image \( g(x) \) can be expressed by a convolution [6], given by

\[
g(x) = (f * h)(x),
\]

where \( h(x) \) is the point spread function (PSF) of the blur, \( * \) denotes 2-D convolution and \( x \) is a vector of coordinates \([x, y]^T\). In the Fourier domain, this corresponds to

\[
G(u) = F(u) \cdot H(u),
\]
where $G(u,v)$, $F(u,v)$, and $H(u,v)$ are the discrete Fourier transforms (DFT) of the blurred image $g(x)$, the original image $f(x)$, and the PSF $h(x)$, respectively, and $u$ is a vector of coordinates $[u,v]^T$ in the frequency domain. We may separate the magnitude and phase parts of (2), resulting in

$$|G(u,v)| = |F(u,v) \cdot H(u,v)| \quad \text{and}$$

$$\angle G(u,v) = \angle F(u,v) + \angle H(u,v). \quad (3)$$

If we assume that the blur PSF $h(x)$ is centrally symmetric, namely $h(x) = h(-x)$, its Fourier transform is always real-valued, and as a consequence its phase is only a two-valued function, given by

$$\angle H(u) = \begin{cases} 0 & \text{if } H(u) \geq 0 \\ \pi & \text{if } H(u) < 0 \end{cases}. \quad (4)$$

In our previous work [4] we have used this property to construct blur invariants by doubling the phase angle modulo $2\pi$ that has the effect of canceling the non-zero phase shift of $H(u)$. The resulting phase blur invariants

$$B(u) = 2\angle F(u) \mod 2\pi \quad (5)$$

are insensitive to any centrally symmetric PSF for all $u$. We also notice from (4) that

$$\angle G(u) = \angle F(u) \quad \text{for all } H(u) \geq 0, \quad (6)$$

which means that doubling the phase angle is unnecessary if we know the frequencies where $H(u)$ is positive. At those frequencies phase angle is directly blur invariant.

In the case of ideal motion and out of focus blur, the cross-section of $h(x)$ is rectangular [6]. This results in a spectrum $H(u,v)$ of which cross-section is a sinc function containing also negative values. The values of $H(u,v)$ are always positive before the first zero crossing at frequency $\approx$ (sampling frequency)/(blur length) that satisfies (6). For a Gaussian PSF, which models, for example, atmospheric turbulence blur [6], $H(u,v)$ is also Gaussian with only positive values that always fulfill the condition (6).

In practice, blur invariance cannot be achieved completely from phase information because of the border effect. Linear convolution assumed in (1) has the property of enlarging the resulting image size. The parts that spread over the borders will be clipped in the observed image, and hence some information will be lost. When the extent of the blur is large enough compared to the image size, this border effect becomes noticeable. The best we can achieve with the phase angle is therefore some insensitivity to blur, not complete invariance.

3. LOCAL PHASE QUANTIZATION

In this paper we use phase information for deriving blur-insensitive representation of local image characteristics, which means that we need to determine the phase for every pixel location. Instead of doubling the phase angle, we use only low frequency components that are likely to satisfy the condition (6) as pointed out in the previous section.

3.1. Local Frequency Analysis

Let $D_m$ be an $m \times m$ square region defined by

$$D_m = \{ y \in \mathbb{Z}^2 ; |y|_\infty \leq r \ ; \ m = 2r + 1 \}. \quad (7)$$

For all pixel locations $x = \{x_1, x_2, \ldots, x_N\}$ of an $N \times N$ image $f(x)$ we use local image patches

$$f_x(y) = f(x-y) \quad \forall y \in D_m \quad (8)$$

to derive the local frequency domain representation

$$F_x(u) = \sum_{y \in D_m} f_x(y)\phi_u(y), \quad (9)$$

where $i = 1, \ldots, m^2, u = \{u_1, u_2, \ldots, u_m\}$ is a set of 2-D frequency variables, and $\phi_u(y)$ is a complex valued basis function of a linear transformation. Different choices for $\phi_u(y)$ are discussed in Section 4.

Using vector notation we can rewrite (9) to

$$F_x(u) = \phi_u^T f_x, \quad (10)$$

where

$$\phi_u^T = [\phi_u(y_1), \phi_u(y_2), \ldots, \phi_u(y_{m^2})], \quad (11)$$

and

$$f_x = [f_x(y_1), f_x(y_2), \ldots, f_x(y_{m^2})]^T. \quad (12)$$

3.2. Phase Quantization

In principle, we could take the phase angles of $F_x(u)$ at some frequencies $u \in \{u_1, u_2, \ldots, u_L\}$ that satisfy the condition (6) and use them as a blur-insensitive local descriptor. However, this will result in a feature vector of length $L$ for every pixel, which is not a very compact representation. Instead, we quantize the phase into four quadrants. This is efficiently implemented by the following quantizer:

$$Q(F_x(u)) = \{\Re\{F_x(u)\} > 0 \} + 2\{\Im\{F_x(u)\} > 0\}, \quad (13)$$

where $\Re\{\cdot\}$ and $\Im\{\cdot\}$ return the real and the imaginary parts of a complex number, respectively.

The benefit of using this quantizer is that the phase angle does not need to be evaluated explicitly. The quantizer results in a 2-bit representation for a single frequency component at every pixel. For $L$ coefficients the number of bits per pixel becomes $2L$. These bits can be concatenated into one codeword.

To demonstrate the effect of quantization, we have performed this operation to the "cameraman" image shown in Figure 1 a). After quantization the frequency coefficients are reconstructed by preserving their original magnitudes. The image shown in Figure 1 b) has been reconstructed from the quantized coefficients using inverse discrete Fourier transform. We can notice that although
the distortion caused by this coarse quantization is significant, most of the details in the image are still recognizable. Hence, one can conclude that the degradation caused by the phase quantization is not crucial from the image analysis point of view.

As mentioned above we use only a set of low frequency components for achieving a blur-insensitive representation. Low frequency coefficients usually also contain most of the image energy, and therefore they have better signal to noise ratio than the high frequency components. More specifically, we use the following four frequencies: \( u_1 = [a, 0]^T \), \( u_2 = [0, a]^T \), \( u_3 = [a, a]^T \), and \( u_4 = [a, -a]^T \), where \( a \) is a scalar frequency below the first zero crossing of \( H(u) \). In practice, we use \( a = 1/m \) in the following discussion. The frequency samples are illustrated in Figure 2 when \( m = 5 \) and \( a = 1/5 \). As a result we get an 8-bit codeword that describes the local texture around each pixel. These local phase quantization (LPQ) codewords can be histogrammed over the whole image or some region of interest that leads to a histogram of 256 bins. In pattern recognition the resulting LPQ descriptors can be used as feature vectors for classification.

The disadvantage of utilizing only low frequency components is that high frequency details are lost. In practice, this is not a major problem since we can always adjust the window size where the frequency coefficients are computed. For the smallest \( 3 \times 3 \) window all the coefficients are preserved, but also blur-insensitivity will be compromised. Larger window size means higher tolerance to blurring but less details. This means that there is always a trade-off between the discrimination power of the descriptor and its blur-insensitivity.

3.3. Decorrelation

The scalar quantization performed in the previous subsection is efficient from the information theoretic point of view only if the coefficients to be quantized are statistically independent. In the case of correlated coefficients vector quantization is a more efficient approach. Another possibility usually employed in source coding is to decorrelate the samples before scalar quantization. Using this idea we will next discuss how to improve the performance of the basic LPQ method described above.

For frequency coefficients \( F_x(u) \) we separate the real and the imaginary parts, and concatenate them into a vector

\[
F_x = [F_{x,R}, F_{x,I}]^T, \tag{14}
\]

where

\[
F_{x,R} = \text{Re}\{ [F_x(u_1), F_x(u_2), \ldots, F_x(u_L)] \}, \tag{15}
\]

and

\[
F_{x,I} = \text{Im}\{ [F_x(u_1), F_x(u_2), \ldots, F_x(u_L)] \}. \tag{16}
\]

We can easily see from (10) that the vectors \( F_x \) and \( f_x \) have a linear dependence

\[
F_x = \Phi f_x, \tag{17}
\]

where

\[
\Phi = [\Phi_R, \Phi_I]^T, \tag{18}
\]

\[
\Phi_R = \text{Re}\{ [\phi_{u_1}, \phi_{u_2}, \ldots, \phi_{u_4}] \}, \tag{19}
\]

and

\[
\Phi_I = \text{Im}\{ [\phi_{u_1}, \phi_{u_2}, \ldots, \phi_{u_4}] \}. \tag{20}
\]

We assume that the image patch \( f_x(y) \) is a realization of a random process, where the correlation coefficient between adjacent pixel values is \( \rho \), and the variance of each sample is \( \sigma_x^2 \). Without a loss of generality we can also assume that \( \sigma_x^2 = 1 \). As a result, the covariance between two positions \( y_i \) and \( y_j \) becomes

\[
\sigma_{i,j} = \rho |y_i - y_j|, \tag{21}
\]

where \( || \cdot || \) denotes \( L_1 \) norm. Consequently, the covariance matrix of \( f_x \) can be expressed by

\[
C = \begin{bmatrix}
1 & \sigma_{1,2} & \cdots & \sigma_{1,m^2} \\
\sigma_{2,1} & 1 & \cdots & \sigma_{2,m^2} \\
& & \ddots & \ddots \\
& & & 1
\end{bmatrix}. \tag{22}
\]

This covariance matrix is assumed to be space invariant. In other words, the same matrix applies to all image positions \( x \).

Based on the linear dependence (17) the covariance matrix of \( F_x \) can be computed from

\[
D = \Phi C \Phi^T. \tag{23}
\]

If we use the same frequency samples \( u_1, u_2, u_3, \) and \( u_4 \) as in the previous section, \( D \) is an \( 8 \times 8 \) matrix, and
Φ is an \( 8 \times m^2 \) matrix. In general \( \mathbf{D} \) is not diagonal for \( \rho > 0 \), meaning that the frequency coefficients are correlating. Hence, scalar quantization of the coefficients is not an optimal approach.

A standard method for decorrelating sample vectors is to employ whitening transform

\[
\mathbf{G}_x = \mathbf{V}^T \mathbf{F}_x, \tag{24}
\]

where \( \mathbf{V} \) is an orthonormal matrix derived from the singular value decomposition (SVD)

\[
\mathbf{D} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T. \tag{25}
\]

Notice, that \( \mathbf{V} \) can be solved in advance for a fixed value of \( \rho \).

Next, \( \mathbf{G}_x \) is computed for all image positions, and the resulting vectors are quantized using the scalar quantizer

\[
g_j = \begin{cases} 1, & \text{if } g_j \geq 0 \\ 0, & \text{otherwise} \end{cases}, \tag{26}
\]

where \( g_j \) is the \( j \)th component of \( \mathbf{G}_x \). The quantized coefficients are represented as 8-bit integers in the same way as described in previous section.

Using the whitening transform has the effect of increasing the information content of the LPQ descriptor. However, it also assumes that the blur PSF is isotropic. For out of focus blur or Gaussian PSF this is not a problem, but for example, for motion blur the assumption is not met. In practice, it might be useful to try out both approaches (with and without decorrelation) and to select the one which gives better results.

### 4. Computing Local Phase

In this section, we discuss different approaches for computing the local frequency representation that is needed to extract the phase information. An essential requirement for the representation is that each frequency coefficient should be independent of the DC-component, i.e., the mean value of the image. This condition is satisfied if all the basis functions \( \phi_i(y) \) have zero mean:

\[
\sum_y \phi_i(y) = 0 \quad \forall i = 1, 2, 3, 4. \tag{27}
\]

If the condition is not met, a non-zero DC-value causes bias to the frequency coefficients, and hence, some quadrants will be more likely than others in phase quantization. This will have the effect of reducing the information content (or the entropy) of the descriptor.

#### 4.1. Short-Term Fourier Transform

The most straightforward approach is to compute the 2-D discrete Fourier transform (DFT) or, more precisely, a short-term Fourier transform (STFT) for the image \( f(x) \) over \( n \times m \) patches \( f_x(y) \). The STFT is defined by the basis function

\[
\phi^x_u(y) = e^{-j2\pi uy} \nu, \tag{28}
\]

An efficient way of implementing the STFT is to use 2-D convolutions \( f(x) \ast \phi^x_u(y) \) for all \( u = \{u_1, u_2, u_3, u_4\} \). Since the basis functions are separable, computation can be performed using 1-D convolutions for the rows and columns successively.

Figure 3 a) illustrates the 8 real valued filters or basis images of this transform for \( n = 31 \). These images are outer products of the 1-D basis functions shown in Figure 4 a). The frequency responses of these 1-D functions are plotted in Figure 5 a). As we can see the first function is constant and the last two functions are essentially bandpass filters that have zero response when \( u = 0 \). This satisfies the condition (27) for all basis images illustrated in Figure 3 a). Notice that the basis image corresponding to the DC component \( (u = [0, 0]^T) \) is not included.

An apparent problem of the STFT is that the basis functions have sharp edges which causes a clear sinc-effect to the frequency responses. This can enhance the border effect and make the resulting frequency coefficients more dependent on the high-frequency components of the spectrum.

#### 4.2. Gabor Filters

A typical approach for performing local frequency analysis is to use Gabor filters. If we consider the same frequency components \( u = \{u_1, u_2, u_3, u_4\} \) as above, the Gabor basis functions are defined by

\[
\phi^x_u(y) = e^{-\frac{1}{2} \pi \sigma^2 y^T y} e^{-j2\pi uy^T y}, \tag{29}
\]

where \( \sigma \) is the standard deviation of the Gaussian envelope. Notice that this is a slightly simplified version of the standard Gabor filter, which has more parameters in its general form.

Gabor filters do not directly satisfy the condition (27), because their real parts have non-zero DC responses. One possibility to overcome this problem is to normalize the filters to have zero mean.
Figure 4. The 1-D basis functions of low frequency a) short-term Fourier transform, b) Gabor filters, and c) least squares filters.

Figure 3 b) illustrates the zero mean corrected basis images when $\sigma = 7.5$, and Figure 4 b) shows the corresponding 1-D basis functions. The frequency responses are represented in Figure 5 b). We can observe that due to the Gaussian weighting Gabor filters have smoother edges than the STFT that increases the stopband attenuation in the frequency domain.

### 4.3. Least Squares Filters

Next we propose an alternative to Gabor filtering for computing the local frequency coefficients using weighted least squares filters. The basic idea of the approach is to approximate a reduced set of the STFT basis functions by giving more weight to the pixels in the middle of the $m \times m$ window where the frequency coefficients are estimated. Notice that in Gabor filtering the Gaussian weight function is used to attenuate directly the pixel values apart from the center, which is a slightly different kind of approach.

Let $z = [-r, -r + 1, \ldots, r - 1, r]^T$, $\psi = e^{j2\pi z/m}$, $\psi_0 = [1, 1, \ldots, 1]^T$, $m = 2r + 1$, and

$$\Psi = [\psi^*, \psi_0, \psi],$$

where $^*$ denotes complex conjugate. The matrix $\Psi$ defines a 1-D inverse DFT for three low frequency components $u = \{-1/m, 0, 1/m\}$.

Effectively, we derive the least squares filters by minimizing the cost function

$$J(\hat{S}) = (s - \Psi \hat{S})^T W (s - \Psi \hat{S}),$$

where $s = [s_1, s_2, \ldots, s_m]^T$ is an arbitrary 1-D signal, $\hat{S}$ is the estimate of its three low frequency components, and $W$ is a weight matrix defined by

$$W = \begin{bmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_m \end{bmatrix}.$$
and

\[ w_i = e^{-2(i-r-1)^2/r^2} \tag{33} \]

The weighted least squares solution for \( \hat{S} \) is obtained from

\[ \hat{S} = \Lambda^H s, \tag{34} \]

where

\[ \Lambda = W\Psi(\Psi^H W\Psi)^{-1}, \tag{35} \]

and \( H \) denotes conjugate transpose. The 1-D least squares filters \( \lambda_{-1}, \lambda_0, \) and \( \lambda_1 \) correspond to the columns of \( \Lambda \):

\[ \Lambda = [\lambda_{-1}, \lambda_0, \lambda_1]. \tag{36} \]

It can be shown that \( \lambda_{-1} = \lambda_1^* \). The 2-D filters are now constructed by taking outer products of the 1-D filters. More specifically, we use

\[
\begin{align*}
\phi_{u1}^c(y) & = \lambda_1 \Lambda_1^T y = \lambda_0 \Lambda_1^T y \tag{37} \\
\phi_{u2}^c(y) & = \lambda_1 \Lambda_1^T y = \lambda_1 \Lambda_1^T y
\end{align*}
\]

In a practical implementation, the separability property of the filters can be utilized to apply only 1-D filters to image rows and columns successively.

Figure 4c) shows the filters \( \lambda_0 \) and the real and imaginary parts of \( \Lambda_1 \), respectively. The corresponding frequency responses are depicted in Figure 5c). The real and imaginary parts of the 2-D filters \( \phi_{u1}^c(y) \), \( \phi_{u2}^c(y) \), \( \phi_{v1}^c(y) \) and \( \phi_{v2}^c(y) \) are illustrated in Figure 3c). The resulting filters have almost the same characteristics as Gabor filters. From the frequency responses we can observe that the DC response for \( \lambda_1 \) is zero implying that the 2-D filters satisfy the condition (27) directly without zero correction that was needed in Gabor filtering. Also, the stopband attenuation is slightly better than in Gabor filtering which makes the filters more likely to fulfill the condition (6). The greatest difference is in the first filter that corresponds to the DC coefficient.

5. EXPERIMENTS

In the experiments, we compared the performance of the different LPQ approaches for classification of sharp and blurred textures. The local frequency representation was formed by using the STFT (LPQ), Gabor filters (LPQg), and weighted least squares filters (LPQw). Each of the approaches was tested with and without decorrelation which precedes the quantization of the phase. In the results, the use of decorrelation is denoted by \( d \) after the abbreviation of the method. As test material we used two applicable test suites of the Outex texture image database \(^1\)[9]. For comparison, we also did the same experiments with two other widely known texture classification methods: local binary pattern (LBP) method \(^2\)[10, 7] and a method based on Gabor filter banks \(^3\)[8]. We used the Matlab implementations of these reference methods that can be found from the Internet. Both methods have also been used previously in conjunction with the Outex texture database [9].

Also the original LPQ method [5] using the STFT is available in the Internet \(^4\). For all LPQ variants the correlation coefficient was selected to be \( \rho = 0.9 \) in the experiments.

Both of the test suites of the Outex texture database used in our experiments, Outex TC_00001 and Outex TC_00002, contained images from 24 texture classes and had 100 different test cases that divided the images into training and test sets differently. Figure 6 shows examples of the images of test suite Outex TC_00001. Outex TC_00002 more challenging of the two test suites, as it contains more and smaller images. We used a \( k \)-nearest neighbor (\( k \)-NN) classifier, which was trained and tested using the appropriate sets of the images. The value of \( k \) was 3 and 15 for test suites Outex TC_00001 and Outex TC_00002, respectively. We used the Chi square distance measure for the LPQ and LBP histograms. For the Gabor features we used the distance measure proposed in [8].

For LBP we took 8 samples at radius \( r \) as is done in the classical LBP. This results in a code with values in the range \( \{0, \ldots, 255\} \), similar to LPQ. For LPQ and LBP, the window dimension is \( m = 2r + 1 \). A larger radius \( r \) for LPQ and LBP, which provides the comparable spatial extent of the operators, results in greater robustness to blur but at the same time looses discrimination power for small details of the images.

In the first experiment, we tested the classification performance of the different methods for the sharp texture images of the challenging Outex TC_00002 test suite. The test suite includes 8832 images of size \( 32 \times 32 \); hence, 368 images per class. We used radius \( r = 1 \) for the LPQ methods and LBP, which results in the basic forms of these operators. In this case LPQw reduces to the same operator as LPQ and therefore it has been omitted from the experiments. The classification accuracy as percentages is shown in Table 1.

We can see that decorrelation improves the results of all LPQ variants. Without decorrelation, LBP and Gabor methods result in the best classification accuracy, and the best of the LPQ variants is the basic LPQ. If decorrelation is used both LPQd and LPQgd outperform the LBP and Gabor methods. In this case, the best LPQ variant is LPQgd. This experiment shows that LPQ based meth-

\(^1\)http://www.outex.oulu.fi/
\(^2\)http://www.ee.oulu.fi/mvg/page/lbp.matlab/
\(^3\)http://vision.ece.ucsb.edu/texture/software/
\(^4\)http://www.ee.oulu.fi/mvg/download/lpq/
The second experiment demonstrates the effect of increasing the radius \( r \) of the LPQ operators in classification of blurred textures. For demonstration we chose LPQd operator, but also the other LPQ variants behave similarly. We used Outex\_TC\_00001 database, which includes 2112 images of size \( 64 \times 64 \) and thus 88 images per class. The artificial blur had circular PSF with radius \( \{0, 0.25, \ldots, 2\} \), which mimics the blur caused, for example, by defocus of the lens [6]. Figure 7 illustrates three examples of one texture image with blur radii 0, 1, and 2, respectively. Table 2 shows the results.

As it can be noticed from Table 2, the best classification accuracy (bold letters) is achieved with larger radius of the operator when the extent of the blur increases. The same applies also for the other LPQ variants. This illustrates the trade-off between discriminability and blur insensitivity of LPQ with different \( r \). Increasing the radius of the LBP also increases its blur insensitivity slightly.

In the third experiment, we compared the texture classification accuracy of all test methods for different blur types and extents. We chose to use radius \( r = 3 \) for LPQ variants and LBP as a compromise of discriminability and blur insensitivity. We used again the Outex\_TC\_00001 test suite. We did the same experiment using circular blur with radius \( \{0, 0.25, \ldots, 2\} \), motion blur with length \( \{0, 0.5, \ldots, 4\} \), and Gaussian blur with standard deviation \( \{0.5, 0.75, \ldots, 1.5\} \). Figure 8 illustrates the results.

When there is no blur, all the methods in diagrams of Figure 8 (a-c) are quite close to each other. The best classification result, 99.1 %, is achieved by LPQd. The order of the rest is: LPQd (98.4 %), LPQwd (98.1 %), LBP (97.8 %), LPQg (97.6 %), LPQw (98.2 %), Gabor (96.8 %), and LPQ (96.5 %). When blur level increases the ranking of the methods becomes different and depends also on the blur type. For the maximum circular and Gaussian blur the ranking of the methods is the same. The best accuracy is achieved by LPQd followed by other LPQ variants. The LBP and Gabor methods are clearly not very insensitive to blur. The behaviour of the LPQg is not consistent with other LPQ variants. When blur level increases, at some point the non-decorrelated version of this descriptor becomes more accurate compared to the decorrelated version. In the case of motion blur the ranking of the methods is different: the non-decorrelated LPQ variants are better than their decorrelated counterparts, and the best method is the ordinary LPQ. This is due to the fact that for the motion blur, of which PSF is not isotropic, the correlation between the pixels becomes anisotropic, and the statistical model used in decorrelation does not hold any more. According to the results, it seems that the best choice for blurred texture classification is always either the LPQd or LPQ descriptor, depending to the blur type.

### 6. CONCLUSION

Local features used in image analysis need to be discriminative but also robust to common image degradations such as blurring. In this paper, we have shown that local phase information provides a reliable basis for constructing image descriptors that are highly discriminative and insensitive to centrally symmetric blur. In the experiments carried out, all six variants of the local phase quantization (LPQ) descriptors performed much better with blurred images than those descriptors that have not been designed to be blur insensitive. Although the optimal size of the operator depends on the blur extent, a fixed sized operator can deal with a relatively large range of blur lengths. If decorrelation is used the new descriptors can outperform state-of-the-art methods also in the case of sharp images. When comparing the three approaches for computing the local phase, there is some variability in the results depending on the type and the extent of the blur. For sharp images Gabor filtering gave the best accuracy, but in general the most simplest one, Short-Term Fourier Transform (STFT), performed well in all cases while Gabor filtering and least squares filtering suffered more from the effects of the blur. Hence, using Gaussian weighting does not seem to have a positive effect if the images are blurred. From these results we can conclude that the phase computed using STFT can provide a good compromise between the discriminability and robustness. It should be also noticed that in the decorrelation scheme presented the image pixels are assumed to follow an isotropic correlation model. If this model is not satisfied and the blur PSF is anisotropic, such as motion blur, decorrelation should not be used or otherwise it may deteriorate the classification accuracy.
Table 2. Texture classification accuracy of the LPQd descriptor for Outex_TC_00001 test suite when blur radius and the radius $r$ of the descriptor is varied. Best result for each blur radius is shown with bold letters.

<table>
<thead>
<tr>
<th>Blur radius</th>
<th>0</th>
<th>0.25</th>
<th>0.50</th>
<th>0.75</th>
<th>1.00</th>
<th>1.25</th>
<th>1.50</th>
<th>1.75</th>
<th>2.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>98.9%</td>
<td>93.6%</td>
<td>78.1%</td>
<td>73.4%</td>
<td>70.4%</td>
<td>62.3%</td>
<td>47.0%</td>
<td>28.5%</td>
<td>21.4%</td>
</tr>
<tr>
<td>2</td>
<td>99.2%</td>
<td>99.2%</td>
<td>99.3%</td>
<td>98.9%</td>
<td>98.5%</td>
<td>95.0%</td>
<td>82.3%</td>
<td>63.0%</td>
<td>47.9%</td>
</tr>
<tr>
<td>3</td>
<td>98.4%</td>
<td>98.3%</td>
<td>98.4%</td>
<td>98.1%</td>
<td>98.1%</td>
<td>97.5%</td>
<td>94.6%</td>
<td>89.3%</td>
<td>78.6%</td>
</tr>
<tr>
<td>4</td>
<td>94.1%</td>
<td>94.5%</td>
<td>94.6%</td>
<td>94.5%</td>
<td>94.3%</td>
<td>93.4%</td>
<td>91.0%</td>
<td>87.1%</td>
<td>82.0%</td>
</tr>
<tr>
<td>5</td>
<td>85.2%</td>
<td>85.0%</td>
<td>85.0%</td>
<td>84.6%</td>
<td>84.4%</td>
<td>83.5%</td>
<td>81.8%</td>
<td>79.7%</td>
<td>76.5%</td>
</tr>
</tbody>
</table>

Figure 8. Texture classification results for Outex_TC_00001 test suite when the extent of the blur is increased: (a) circular blur, (b) motion blur, and (c) Gaussian blur.

7. ACKNOWLEDGMENTS

This work has been carried out in a project funded by the Academy of Finland (grant no. 127702).

8. REFERENCES


