Volume Local Phase Quantization for Blur-Insensitive Dynamic Texture Classification

Juhani Päivärinta, Esa Rahtu, and Janne Heikkilä

Machine Vision Group, Department of Electrical and Information Engineering,
University of Oulu, P.O. Box 4500, 90014, Finland
{juhanipa,erahtu,jth}@ee.oulu.fi
http://www.ee.oulu.fi/mvg

Abstract. In this paper, we propose a blur-insensitive descriptor for dynamic textures. The Volume Local Phase Quantization (VLPQ) method introduced is based on binary encoding of the phase information of the local Fourier transform at low frequency points and is an extension to the LPQ operator used for spatial texture analysis. The local Fourier transform is computed efficiently using 1-D convolutions for each dimension in a 3-D volume. The data achieved is compressed to a smaller dimension before a scalar quantization procedure. Finally, a histogram of all binary codewords from dynamic texture is formed. The performance of VLPQ was evaluated both in the case of sharp dynamic textures and spatially blurred dynamic textures. Experiments on a dynamic texture database DynTex++ show that the new method tolerates more spatial blurring than LBP-TOP, which is a state-of-the-art descriptor, and its variant LPQ-TOP.

Keywords: Local Phase Quantization, Short-Term Fourier Transform, spatio-temporal domain, blur-insensitivity, dynamic texture.

1 Introduction

Dynamic textures can be seen as sequences of images of moving scenes that exhibit certain stationarity properties in time [1]. Some examples of dynamic textures in the real world are fire, moving clouds, a waving flag, and sea waves. Dynamic texture analysis is essential in many applications, such as facial expression recognition, action recognition, and background subtraction. Chetverikov and Péteri created a survey on the existing descriptors for dynamic texture recognition in [2] and divided the approaches into five classes: methods based on optical flow, methods computing geometric properties in the spatio-temporal domain, methods based on local spatio-temporal filtering, methods using global spatio-temporal transforms, and model-based methods that use estimated model parameters as features.

Among the most popular approaches for characterizing the local dynamics of dynamic texture are the methods based on optical flow. For example, in [3], Péteri et al. achieved promising results using normal flow features combined with periodicity features, and their features were translation invariant.

A. Heyden and F. Kahl (Eds.): SCIA 2011, LNCS 6688, pp. 360–369, 2011.

In [4], Zhao et al. introduced two methods based on Local Binary Patterns (LBP) [5]: Volume Local Binary Patterns (VLBP) and Local Binary Patterns from Three Orthogonal Planes (LBP-TOP). Of these two, LBP-TOP was shown to be the most efficient approach, and the results achieved were very promising. These methods are based on local characteristics, which is also our approach to dynamic texture classification.

One can also combine multiple descriptors. Recently, Ghanem et al. used Maximum Margin Distance Learning (MMDL) in [6] for dynamic texture recognition. They modeled the distance between two dynamic textures as a positively weighted sum of their elementary distances: spatial texture element, spatial texture layout, and dynamics. However, despite the comprehensiveness, their method is computationally very expensive, since it consists of computing multiple descriptors.

In some applications, there are degradation factors which complicate the actual recognition procedure. One common category of degradation is blur caused by e.g. atmospheric turbulence, motion, or out-of-focus. These blur types can be seen and considered as spatial blurring. To our knowledge, there are no dynamic texture descriptors that are claimed to be robust to spatial blurring. In [7], Ojansivu et al. proposed a method called Local Phase Quantization (LPQ) for blur-insensitive spatial texture analysis. LPQ can be also used for dynamic textures, e.g., when a straightforward generalization of LBP-TOP is made to form a method in which the LPQ descriptors are calculated from three orthogonal planes. For comparison, we have built a descriptor using this approach (LPQ-TOP), but we also propose a more elaborated approach.

In this paper, we introduce a novel method called Volume Local Phase Quantization (VLPQ), which is a dynamic texture descriptor insensitive to centrally symmetric spatial blurring. Our method is an extension to the original 2-D LPQ, and the characterization of dynamic texture is made using the quantized phase information of the Discrete Fourier Transform (DFT) computed in pixel volume neighborhoods. Our method uses Short-Term Fourier Transform (STFT) to evaluate the DFT, and we use the 13 low 3-D frequency points in the evaluation. The computational performance of our method is increased by calculating the STFT using 1-D convolutions for each dimension in a 3-D volume. In our method, dimension reduction is essential for the data before scalar quantization to achieve a reasonable sized binary codeword. After the quantization, a histogram of all codewords from a dynamic texture volume is formed.

2 Volume Local Phase Quantization

Dynamic texture is a sequence of spatial frames. Therefore, we can first consider spatial images suffering from blur. The discrete model for spatially invariant blurring of an original image $s(\mathbf{x})$ resulting in an observed image $g(\mathbf{x})$ can be expressed by a convolution, given by

$$g(\mathbf{x}) = (s * h)(\mathbf{x}) \ , \tag{1}$$

where $h(\mathbf{x})$ is the point spread function (PSF) of the blur, $*$ denotes 2-D convolution, and $\mathbf{x}$ is a vector of coordinates $[x, y]^T$. When (1) is taken to the Fourier domain, the convolution turns into a product. If we consider only the phase of the spectrum, we get

$$\angle G(\mathbf{u}) = \angle S(\mathbf{u}) + \angle H(\mathbf{u}) \ . \tag{2}$$

In the case of centrally symmetric PSFs, $H(\mathbf{u})$ is real valued, and the phase angle $\angle H(\mathbf{u})$ must equal 0 or π. For a typical PSF, the shape of $H(\mathbf{u})$ is similar to a low-pass filter. This often implies that at least the low frequency values of $H(\mathbf{u})$ are positive. At these frequencies, $\angle H(\mathbf{u}) = 0$, and hence $\angle S(\mathbf{u})$ is a blur-invariant property. [7]

In practice, the blur-invariance is partly disturbed because of the finite size of the observed images resulting to a loss of information at the borders. The convolution of the ideal image frame with the blur PSF extends beyond the borders of the observed image, and this results to a loss of information. When the Fourier transform is computed from local patches, this effect increases. However, the local computation allows the blur to vary within a single frame. Ojansivu et al. indicated in [7] that a highly blur-insensitive texture descriptor can be constructed by applying the aforementioned theory. The experiments in [7], [8], and [9] show clearly the blur-insensitive property. When we consider video sequences to consist of multiple spatial frames, this theory can also be extended to construct a blur-insensitive descriptor for dynamic textures. In the next sections, we propose a method for constructing VLPQ descriptor for dynamic textures.

2.1 Short-Term Fourier Transform in the Spatio-Temporal Domain

Dynamic texture consists of a video sequence of multiple frames spread over the time axis. Therefore, each position $\mathbf{x}$ in a sequence $f(\mathbf{x})$ can be expressed in 3-D coordinates, and every position in a sequence has a 3-D neighborhood. Since dynamic textures are textures in the spatio-temporal domain, dynamic texture is mainly a local property. Therefore, the Fourier transform estimation is performed locally using Short-Term Fourier Transform (STFT). STFT is computed over an M-by-M-by-N neighborhood $\mathcal{N}_{\mathbf{x}}$ centered at each position $\mathbf{x}$. M and N denote the size of the neighborhood in the spatial and the temporal domains, respectively. STFT of a sequence $f(\mathbf{x})$ can be defined by

$$F(\mathbf{u}, \mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{N}_{\mathbf{x}}} f(\mathbf{x} - \mathbf{y}) e^{-j2\pi \mathbf{u}^T \mathbf{y}} \ , \tag{3}$$

where $\mathbf{u}$ is a 3-D frequency variable, and $j = \sqrt{-1}$. Using vector notation, we can rewrite (3) as

$$F(\mathbf{u}, \mathbf{x}) = \mathbf{w}_{\mathbf{u}}^T \mathbf{f}_{\mathbf{x}} \ , \tag{4}$$

where $\mathbf{w}_{\mathbf{u}}$ is the basis vector of the 3-D DFT at frequency $\mathbf{u}$, and $\mathbf{f}_{\mathbf{x}}$ is a vector containing all pixels from the neighborhood $\mathcal{N}_{\mathbf{x}}$. Because of the separability of the basis functions, the STFT can be efficiently evaluated for each pixel position

using 1-D convolutions for each dimension. This increases the computational efficiency considerably.

As mentioned, the low frequency points are likely to satisfy $H(\mathbf{u}) > 0$. Therefore, we construct the descriptor using 13 lowest non-zero frequency points: $\mathbf{u}_1 = [\alpha, 0, 0]^T$, $\mathbf{u}_2 = [\alpha, 0, \beta]^T$, $\mathbf{u}_3 = [\alpha, 0, -\beta]^T$, $\mathbf{u}_4 = [0, \alpha, 0]^T$, $\mathbf{u}_5 = [0, \alpha, \beta]^T$, $\mathbf{u}_6 = [0, \alpha, -\beta]^T$, $\mathbf{u}_7 = [\alpha, \alpha, 0]^T$, $\mathbf{u}_8 = [\alpha, \alpha, \beta]^T$, $\mathbf{u}_9 = [\alpha, \alpha, -\beta]^T$, $\mathbf{u}_{10} = [\alpha, -\alpha, 0]^T$, $\mathbf{u}_{11} = [\alpha, -\alpha, \beta]^T$, $\mathbf{u}_{12} = [\alpha, -\alpha, -\beta]^T$, and $\mathbf{u}_{13} = [0, 0, \beta]^T$, where $\alpha = 1/M$ and $\beta = 1/N$. The selected frequency points are illustrated as closed circles in Fig. 1. The other frequency points illustrated in Fig. 1 are ignored, because they are the complex conjugates of the selected ones.

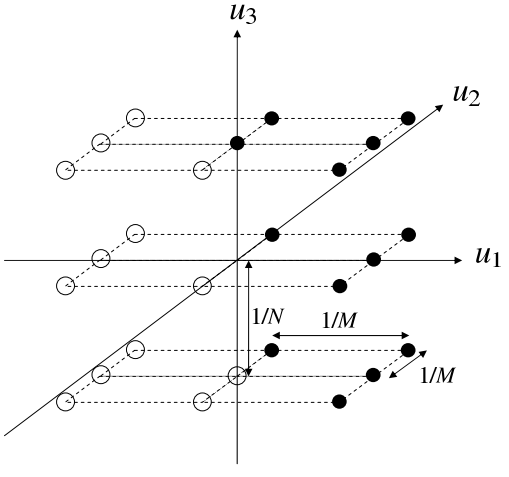

Fig. 1. Frequency points used to calculate STFT

At each position $\mathbf{x}$, after separating the real and imaginary parts of each component, we get a vector

$$\mathbf{F}_\mathbf{x} = [\mathrm{Re}\{F(\mathbf{u}_1, \mathbf{x})\}, \mathrm{Im}\{F(\mathbf{u}_1, \mathbf{x})\}, \ldots, \mathrm{Re}\{F(\mathbf{u}_{13}, \mathbf{x})\}, \mathrm{Im}\{F(\mathbf{u}_{13}, \mathbf{x})\}]^T \ . \quad (5)$$

The corresponding 26-by-M^2N transform matrix can be written as

$$\mathbf{W} = [\mathrm{Re}\{\mathbf{w}_{\mathbf{u}_1}\}, \mathrm{Im}\{\mathbf{w}_{\mathbf{u}_1}\}, \ldots, \mathrm{Re}\{\mathbf{w}_{\mathbf{u}_{13}}\}, \mathrm{Im}\{\mathbf{w}_{\mathbf{u}_{13}}\}]^T \ . \quad (6)$$

Hence, the vector form of the STFT for all frequencies $\mathbf{u_1}, \ldots, \mathbf{u_{13}}$ can be written as

$$\mathbf{F}_\mathbf{x} = \mathbf{W}\mathbf{f}_\mathbf{x} \ . \quad (7)$$

2.2 Dimension Reduction

If we take into account all 13 frequency points and their real and imaginary parts, the length of the resulting descriptor at each position would be 26 real numbers.

Since this number of variables is excessive, dimension reduction is needed in order to compress the data. To do this, we first employ Principal Component Analysis (PCA) to transform the original, possibly correlated, set of variables to a smaller number of uncorrelated variables. For PCA, we use a correlation model with only two parameters. Finally, scalar quantization is performed for the uncorrelated samples.

We assume that the correlation coefficient between two adjacent pixels is ρ_s in the spatial domain, and ρ_t in the temporal domain. We also assume without loss of generality that the variance of each sample is $\sigma^2 = 1$. The covariance between two pixel values $f(\mathbf{x}_i)$ and $f(\mathbf{x}_j)$ can be written as

$$\sigma_{ij} = \rho_s^{d_{ij}^s} \rho_t^{d_{ij}^t} \ , \tag{8}$$

where $d_{ij}^s = \sqrt{\sum_{k=1}^{2} |\mathbf{x}_i(k) - \mathbf{x}_j(k)|^2}$ and $d_{ij}^t = |\mathbf{x}_i(3) - \mathbf{x}_j(3)|$. The covariance matrix of all M^2N pixel positions in a neighborhood $\mathcal{N}_\mathbf{x}$ can then be expressed as a M^2N-by-M^2N matrix $\mathbf{C}$, whose ijth element is σ_{ij}.

Based on the linear dependence (7), we can express the corresponding covariance matrix of $\mathbf{F_x}$ as $\mathbf{D} = \mathbf{WCW}^T$. To obtain uncorrelated sample vectors, we use a whitening transformation matrix $\mathbf{V}$ that is an orthonormal matrix derived from the singular value decomposition (SVD) of $\mathbf{D}$ that is $\mathbf{D} = \mathbf{U\Sigma V}^T$. Only L most important eigenvectors $\mathbf{v}$ from SVD are picked to calculate the whitening transformation. Hence, the final equation for whitening transformation can be defined as

$$\mathbf{G_x} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_L]^T \mathbf{F_x} \ . \tag{9}$$

In our method, the covariance matrix $\mathbf{C}$ is created using a correlation model that is based on assumptions on the correlation between pixel positions. However, these assumptions can be incorrect in the case of some blur PSFs that are not isotropic. Therefore, different correlation models could be used to form $\mathbf{C}$. One approach could also be to estimate $\mathbf{C}$ from the data.

2.3 Quantization

After calculation of the vector $\mathbf{G_x}$ for each volume position, quantization is performed. Since the samples to be quantized are now approximately uncorrelated, we use a simple scalar quantization method similar to the one used in [7]. The quantizer can be defined as

$$q_x(j) = \begin{cases} 1, & \text{if } g_x(j) \geq 0 \\ 0, & \text{otherwise} \ , \end{cases} \tag{10}$$

where $g_x(j)$ denotes the jth component of $\mathbf{G_x}$. The quantized coefficients are further represented as integer values, whose range depends on the number of eigenvectors L picked in the dimension reduction (9). The integer values can be formed using simple binary coding

$$b_x = \sum_{j=1}^{L} q_x(j) 2^{j-1} \ . \tag{11}$$

Finally, we form a histogram of the integer values obtained from all volume positions $\mathbf{x}$. This histogram is used as a 2^L dimensional feature vector.

3 Experiments

VLPQ algorithm was implemented using MATLAB. The algorithm computes the required STFTs using 1-D convolutions, and the convolutions are computed using only valid areas, i.e., areas that can be computed without zero-padding. The convolutions that occur multiple times in the process are calculated only once, and the results are stored for later usage in order to reduce the execution time. In the dimension reduction, L was selected to be 10. This results to a histogram of length $2^{10} = 1024$. The correlation coefficients used in the case of VLPQ were $\rho_s = 0.1$, and $\rho_t = 0.1$.

Also LPQ-TOP was implemented using MATLAB, and the algorithm calculates LPQ histograms from three orthogonal planes similar to LBP-TOP. The algorithm uses a correlation model with parameters $\rho_s = 0.1$, and $\rho_t = 0.1$. All the parameters were selected experimentally. The source codes for VLPQ and LPQ-TOP are available online[1].

The efficiency of VLPQ was experimented using a dynamic texture database DynTex++ [6], which is a new database compiled from the original DynTex database [10]. The performance was measured in the classification of sharp as well as spatially blurred dynamic textures. For comparison, we used LBP-TOP, which is a state-of-the-art method. Another reference method was our implementation of LPQ-TOP, which is a variant of LBP-TOP. These two methods are currently the best performing single descriptor methods and thus comparable to VLPQ.

DynTex++ database consists of 3600 dynamic textures of size $50 \times 50 \times 50$. The textures are divided into 36 classes, each holding 100 videos. Some example frames of the sequences used in our experiments are illustrated in Fig. 2. In our experiments, 50 % of each class was randomly selected to a training set, and the other 50 % to a test set. We used nearest neighbor method to classify the test set vectors. In classification, χ^2 distance was used as a measurement. Every test was repeated 20 times, and an average recognition rate was calculated.

3.1 Classification Tests

Classification accuracies of the methods used were measured in the case of sharp and spatially blurred dynamic textures. The blur was achieved by convolving the texture frames with spatial filters, and the training was done using the sharp textures. We used three different PSFs: circular blur of radii $\{0, 0.5, \ldots, 4\}$, Gaussian blur with standard deviations $\{0, 0.5, \ldots, 4\}$, and motion blur with lengths $\{0, 1, \ldots, 8\}$. Circular blur can be used to model out-of-focus blur, while Gaussian blur models atmospheric turbulence [11]. For motion blur, we used only horizontal direction.

[1] `http://www.cse.oulu.fi/Downloads/LPQMatlab/`

(a) (b)

Fig. 2. Example frames from DynTex++ sequences: (a) frames from different classes, and (b) a circularly blurred frame, blurred using radii $\{0, 0.5, \ldots, 4\}$

In the case of blurred textures, the neighborhood size is a dominating factor in many cases. Usually, a small neighborhood works well at the low blur levels, but a larger neighborhood becomes more beneficial at the higher blur levels. Therefore, the experiment was performed using neighborhood sizes comparable to each other. In the case of dynamic textures, it is not usually reasonable to use similar number of neighboring points in the spatial domain and in the temporal domain [4]. However, the frame rate of DynTex++ sequences is high enough for using the same number of neighboring points in each direction. Fig. 3(a), Fig. 3(b), and Fig. 3(c) illustrate the achieved classification accuracies of VLPQ with $5 \times 5 \times 5$ neighborhood, LPQ-TOP with 5×5 neighborhood on each plane, and LBP-TOP with 8 samples and radii of 2 in each direction. These methods are denoted as $\text{VLPQ}_{5,5,5}$, $\text{LPQ-TOP}_{5,5,5}$, and $\text{LBP-TOP}_{8,8,8,2,2,2}$.

In addition, the performances of the methods were measured in the case of spatially and temporally varying blurring conditions. Each frame of the test sequences was divided into four regions of the same size. Each region of the first frame was blurred with different amount of blur. The blur was then linearly increased so that each region of the last frame suffered from similar amount of blur. The minimum blur levels of the three blur types were achieved using circular blur of radii $\{0, 1, 2, 3\}$, Gaussian blur with standard deviations $\{0, 1, 2, 3\}$, and motion blur with lengths $\{0, 2, 4, 6\}$. The maximum levels were the same as in the previous test. Fig. 3(d) illustrates the achieved accuracies.

From Fig. 3(a), Fig. 3(b), and Fig. 3(c) we can notice that $\text{VLPQ}_{5,5,5}$ is the best option in general. When no blur is present, the best accuracy (95 %) is achieved by $\text{LBP-TOP}_{8,8,8,2,2,2}$ followed by $\text{LPQ-TOP}_{5,5,5}$ and $\text{VLPQ}_{5,5,5}$. However, the differences are not significant, each accuracy being within 2 %. Each algorithm also achieved an accuracy higher than the one in [6]. When the blur becomes more eminent, the differences between the methods used become more considerable, and the high blur-insensitivity of $\text{VLPQ}_{5,5,5}$ becomes noticeable. In the case of circular or Gaussian blur, $\text{VLPQ}_{5,5,5}$ maintains its performance extremely well compared to the other methods, $\text{LPQ-TOP}_{5,5,5}$ being the second best solution.

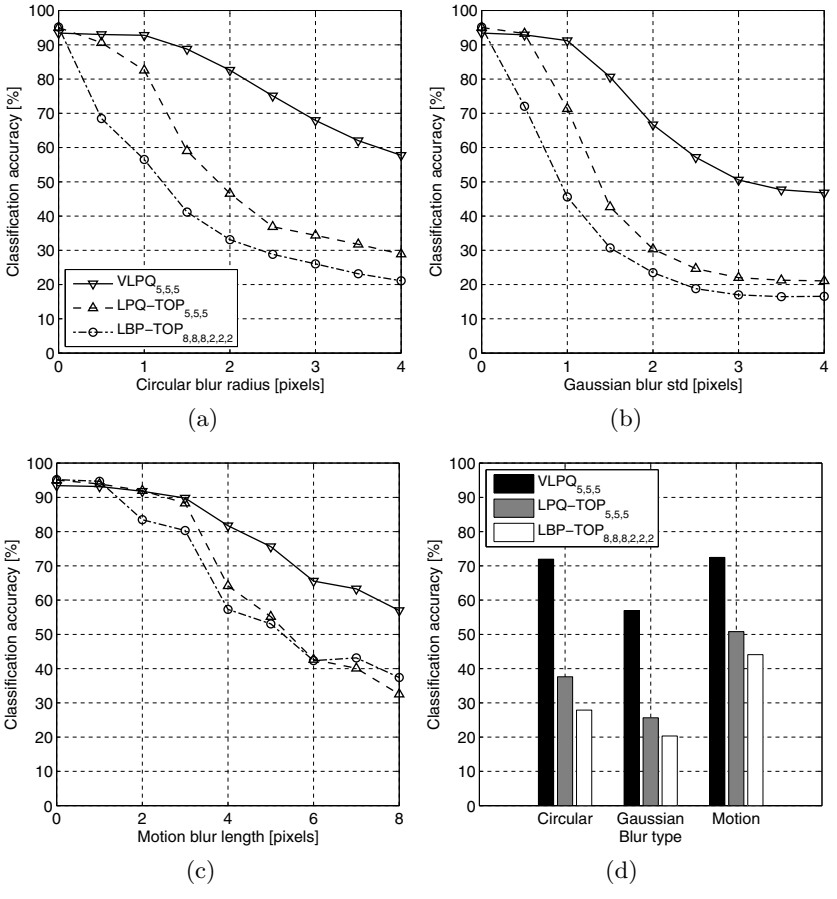

Fig. 3. Classification results in the case of different blur types: (a) circular blur, (b) Gaussian blur, (c) linear motion blur, and (d) varying blur

In the case of linear motion blur, the classification accuracies of the algorithms used are closer to each other than in the previous cases. This behavior can be understood knowing that the PSF of motion blur is not isotropic. Therefore, the correlation model of the LPQ-based methods is not as suitable as before. However, these methods perform well up to relatively high blur levels, $\text{VLPQ}_{5,5,5}$ once again being the best overall solution.

From Fig. 3(d) we can notice that in the case of varying blurring, $\text{VLPQ}_{5,5,5}$ outperforms the two other methods considerably. In all three cases, the classification accuracy of $\text{VLPQ}_{5,5,5}$ remains relatively high compared to the other methods even though the blurring conditions are now varying both spatially and temporally, and the classification is very challenging.

3.2 Comparison on Execution Times

In order to clarify the computational complexity of the methods used, a comparison on their execution times was performed. The test was carried out on DynTex++ video sequences, and we used the MATLAB implementations of the algorithms. Table 1 illustrates the execution times of VLPQ, LPQ-TOP, and LBP-TOP for one DynTex++ video sequence. We used the same neighborhood size in each direction for all methods. In the case of LBP-TOP, neighborhood size M equals to radii (M-1)/2. The times illustrated in Table 1 are mean times of 3600 sequences. Each algorithm was tested with the same configuration[2].

Table 1. Execution times of the methods used

Neighborhood size	VLPQ	LPQ-TOP	LBP-TOP
3	0.13 s	0.28 s	0.15 s
5	0.13 s	0.31 s	0.15 s
7	0.14 s	0.29 s	0.14 s
9	0.14 s	0.29 s	0.13 s
11	0.13 s	0.28 s	0.13 s

As we can notice, computationally the fastest algorithms are VLPQ and LBP-TOP. We can also notice that a larger neighborhood does not increase the execution time significantly. In the case of LBP-TOP, the execution time actually seems to decrease, when a larger neighborhood is used. This behavior can be explained by the fact that all of these algorithms use only the valid pixels of a video sequence. With a large neighborhood size, there are less valid neighborhoods, and a smaller part of a video volume can be used in calculation. DynTex++ sequences are small enough for this to have an effect on the execution time. It is also worth mentioning that the server used in the computation was variably stressed by other processes during the test. However, the relations between the execution times are valid.

4 Conclusions

In this paper, a novel dynamic texture descriptor VLPQ is proposed. VLPQ is a 3-D extension to the LPQ method, and it utilizes the Fourier transform phase information calculated locally at every texture volume position using 13 low non-zero frequency points. As a result of separating the real and imaginary parts of the Fourier transform, a vector of length 26 is formed. Dimension reduction is performed to achieve a codeword of a reasonable length, and a histogram is finally formed out of the results from all neighborhoods.

The performance of the method introduced was compared to a state-of-the-art method LBP-TOP and its variant LPQ-TOP. The results of the tests performed

[2] MATLAB R2010a on a 2.4 GHz, 96 GB Sunray server.

show that our method tolerates more centrally symmetric spatial blurring than the two aforementioned methods. It was also shown that VLPQ performs approximately equally compared to LPQ-TOP and LBP-TOP in the case of sharp dynamic textures.

Acknowledgments. This work was supported by the Academy of Finland (grant nos. 127702 and 128975).

References

1. Doretto, G., Chiuso, A., Wu, Y.N., Soatto, S.: Dynamic Textures. International Journal of Computer Vision 51(2), 91–109 (2003)
2. Chetverikov, D., Péteri, R.: A Brief Survey of Dynamic Texture Description and Recognition. In: International Conference on Computer Recognition Systems, pp. 17–26 (2005)
3. Péteri, R., Chetverikov, D.: Dynamic Texture Recognition Using Normal Flow and Texture Regularity. In: Marques, J.S., Pérez de la Blanca, N., Pina, P. (eds.) IbPRIA 2005. LNCS, vol. 3523, pp. 223–230. Springer, Heidelberg (2005)
4. Zhao, G., Pietikäinen, M.: Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI 2007) 29(6), 915–928 (2007)
5. Ojala, T., Pietikäinen, M., Mäenpää, T.: Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI 2002) 24(7), 971–987 (2002)
6. Ghanem, B., Ahuja, N.: Maximum Margin Distance Learning for Dynamic Texture Recognition. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6312, pp. 223–236. Springer, Heidelberg (2010)
7. Ojansivu, V., Heikkilä, J.: Blur Insensitive Texture Classification Using Local Phase Quantization. In: Elmoataz, A., Lezoray, O., Nouboud, F., Mammass, D. (eds.) ICISP 2008. LNCS, vol. 5099, pp. 236–243. Springer, Heidelberg (2008)
8. Ojansivu, V., Rahtu, E., Heikkilä, J.: Rotation Invariant Local Phase Quantization for Blur Insensitive Texture Analysis. In: 19th International Conference on Pattern Recognition (ICPR 2008), pp. 1–4. Tampa, FL (2008)
9. Ahonen, T., Rahtu, E., Ojansivu, V., Heikkilä, J.: Recognition of Blurred Faces Using Local Phase Quantization. In: 19th International Conference on Pattern Recognition (ICPR 2008), pp. 1–4. Tampa, FL (2008)
10. Péteri, R., Fazekas, S., Huiskes, M.J.: DynTex: A Comprehensive Database of Dynamic Textures. Pattern Recognition Letters 31(12), 1627–1632 (2010), `http://www.cwi.nl/projects/dyntex/`
11. Banham, M.R., Katsaggelos, A.K.: Digital Image Restoration. IEEE Signal Processing Magazine 41(2), 24–41 (1997)