DYNAMIC TEXTURE RECOGNITION USING MULTISCALE PCA-LEARNED FILTERS

Xiaochao Zhao[1,2], *Yaping Lin*[1], *Janne Heikkilä*[2]

[1]College of Computer Science and Electronic Engineering, Hunan University, China
[2]Center for Machine Vision and Signal Analysis, University of Oulu, Finland
{s12103017, yplin}@hnu.edu.cn, jth@ee.oulu.fi

ABSTRACT

In this paper, we propose a novel method for dynamic texture recognition using multiscale PCA-learned filters. PCA is utilized to learn multiscale filters from image sequences on three orthogonal planes (XY, XT and YT). Filter responses that contain both spatial and temporal information at multiple scales are then encoded into a descriptor named MPCAF-TOP. The proposed method is simple to derive and implement, and also very effective for dynamic texture recognition. The proposed method is evaluated on two benchmark databases, namely UCLA and DynTex++. Experimental results show that the proposed approach is comparable to state-of-the-art methods.

Index Terms— Dynamic texture recognition, PCA-based filter learning, multiscale analysis

1. INTRODUCTION

Dynamic texture (DTs) or temporary textures are textures with motion[1]. They are sequences of moving scenes that exhibit certain stationary properties in time[2]. Some examples of DT in real world are sea-waves, smoke, fire, etc. The changes in DT patterns are caused by motion (e.g., swaying leaves) and may also be caused by the variation in intensity of the emitted light (e.g., fire). The studies related to dynamic textures range from DT modeling and synthesis to classification and recognition. Modeling and recognition of DT have gained much attention in the field of computer vision because of their usages in various applications of visual processing, such as video indexing/retrieval[3], facial analysis[4], emergency detection[5], etc. In this paper we focus on developing an effective descriptor for DT representation and recognition.

Different from static textures, dynamic textures not only vary on the distribution of spatial texture elements, but also vary on their organization and dynamics over time, making the recognition of DT a challenging problem. To tackle this problem, various methods have been proposed in literature. Most of the early methods used optical flow to directly capture the motion patterns from videos[6, 7]. Later, linear dynamic systems (LDS) model was used to describe DT[2]. Ravichandran et al. used a bag of LDSs for view-invariant DT recognition[8]. Besides LDS model, fractal analysis is also utilized, such as dynamic fractal spectrum (DFS)[9], 3D oriented transform feature (3D-OTF)[10] and wavelet domain multifractal spectrum (WMFS)[11]. Subspace analysis is also used for DT modeling and classification. Arashloo and Kittler proposed multiscale binary statistical feature descriptor (MBSIF) by binarizing the responses of a set of learned filters[12]. Rivera and Chae presented a descriptor based on 3D filtering, named Directional Number Transitional Graph (DNG)[13]. Another category of methods for DT recognition uses the statistical properties of the local spatial distribution of pixel intensities. Zhao and Pietikinen proposed volume local binary pattern (VLBP) and local binary pattern in three orthogonal planes (LBP-TOP) to extend LBP to spatio-temporal domain[4]. Through a maximum margin distance learning (MMDL) scheme, Ghanem and Ahuja combined LBP, pyramid of histograms of oriented gradients (PHOG) and LDS to represent and classify DT[14]. Ren et al. used principal histogram analysis (PHA) to mitigate the reliability issues of LBP histograms[15]. Tiwari and Tyagi included contrast information to improve the performance of VLBP, resulting in a descriptor called CVLBP[16]. They further proposed to combine Michelson contrast [17] and global mean absolute difference with LBP and extract histograms on three orthogonal planes[18]. Deep learning has also been employed to describe DT[19, 20].

Those methods based on optical flow or LDS used only motion information and abandoned appearance information. Fractal analysis based methods achieved good results only with SVM. LBP-based methods only used a certain number of pixels in a local area, which may loss some useful information about nonsampled pixels. On the other hand, filter based approaches used information from the whole local area and showed good performance for DT recognition.

In this paper, we propose a filter-based method (MPCAF-TOP) that exploits PCA to learn multiple sets of filters at different scales. The obtained filters are applied to a DT video on three orthogonal planes. Filter responses are encoded via binary encoding. A histogram is finally extracted to represent a DT. The key innovation here is the usage of PCA-learned filters on three orthogonal planes to capture both motion and texture information. MPCAF-TOP takes all the pixels in a

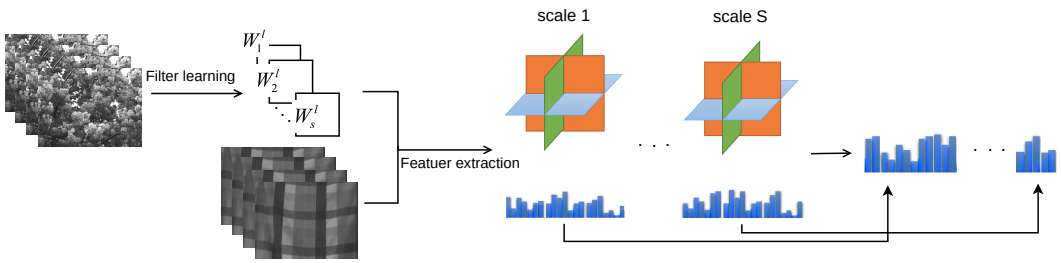

Fig. 1. Flowchart of the feature extraction process.

local area into consideration like those filter based methods. MPCAF-TOP improves the performance of the famous LBP-TOP, and outperforms some state-of-the-art methods on two benchmark databases.

The rest of this paper is organized as follows. Section 2 presents details of the proposed method. Experimental results are reported in Section 3. Section 4 concludes this paper.

2. THE PROPOSED METHOD

2.1. PCA based filter learning

For filter learning, we use the learning technique proposed in PCANet[21]. PCANet is comprised of two filter-learning stages in a cascaded way. In this paper we only use the learning strategy of the first stage.

Suppose that we have N input training images $\{I_i\}_{i=1}^N$ of size $m \times n$ and the filter size is $k_1 \times k_2$. For each pixel in an image, a $k_1 \times k_2$ patch is sampled. Zero padding is applied when it comes to the border pixels. Thus, mn (overlapping) patches are collected from the ith image, which are $x_{i,1}, x_{i,2}, \ldots, x_{i,mn} \in R^{k_1 k_2}$ with $x_{i,j}$ being the jth vectorized patch in I_i. Then patch mean is subtracted from each patch and here we get $\overline{X}_i = [\overline{x}_{i,1}, \overline{x}_{i,2}, \ldots, \overline{x}_{i,mm}]$. By applying the same operation to all input images and putting them together, we have

$$X = [\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_N] \in R^{k_1 k_2 \times Nmn}. \quad (1)$$

Suppose we want to learn L filters of size $k_1 \times k_2$. PCA is used to minimize the reconstruction error within a family of orthonormal filters as

$$\min_{V \in R^{k_1 k_2 \times L}} ||X - VV^T X||_F^2, \ \ s.t. \ V^T V = I_L \quad (2)$$

where I_L is identity matrix of size $L \times L$. The solution is the first L principal eigenvectors of XX^T after sorting the eigenvalues in descending order such that the eigenvectors can capture the main variation of mean-removed patches. The L eigenvectors, as columns of V, are used as filters $\{W_l\}_{l=1}^L$

$$W_l = map_{k_1,k_2}(q_l(XX^T)) \in R^{k_1 \times k_2}, l = 1, 2, \ldots, L, \quad (3)$$

where $map_{k_1,k_2}(v)$ is a function that reshapes $v \in R^{k_1 k_2}$ to a matrix $W \in R^{k_1 \times k_2}$, and $q_l(XX^T)$ denotes the lth leading principle eigenvector of XX^T.

2.2. MPCAF-TOP

In this section, we give details about how to use the previous learned filters for DT description. We first describe the single scale version of the proposed method followed by multiscale analysis.

By viewing a given DT video as a 3D volume, we extract images from XY, XT and YT planes, generating three sets of images to learn filters for the three orthogonal planes respectively. After images have been extracted from all the training DT videos, for each set of images, we apply the filter learning procedure in Subsection 2.1 to them and obtain L filters for each plane.

Around each pixel in a DT video (zero padding also used for border pixels), L filters are applied to the local patch on each of the three orthogonal planes with the pixel being on them. For each plane, the L filter responses are thresholded with 0 and encoded via binary encoding as

$$C_{x,y,t}^p = \sum_{l=1}^{L} 2^{l-1} S(P_{x,y,t}^p * W_l^p), p = XY, XT, YT, \quad (4)$$

where (x, y, t) is the location of the pixel, $P_{x,y,t}^p$ is the patch around the pixel on the p-plane, $S(x)$ returns the sign of x and the code word $C_{x,y,t}^p$ is an integer in the range of $[0, 2^L - 1]$.

After encoding, we extract a histogram (2^L bins) for each of the three planes. The histograms are concatenated to generate the single scale PCAF-TOP.

Now it comes to the problem of how to obtain multiscale PCAF-TOP (MPCAF-TOP), which is mainly about using various filter sizes to take into account the variations of DT video over different support regions. As smaller filter captures high frequency variation of texture and larger filter can better deal with blurring effects and low frequency component, we decide to use a sequence of filter sizes which include both small and large sizes, like $\{3, 5, 7, 9, 11, 13, 15\}$ (square filters), to construct a multiscale representation. At each scale, L filters

for each plane are learned. The histograms from each scale are finally concatenated to represent a DT sequence. The flowchart of feature extraction process is shown in Fig. 1.

The number of filters, L, is another parameter that affects the frequency content of the feature. Suppose the filter size is fixed at $k_1 \times k_2$. By increasing the value of L, more high frequency components will be retained, increasing sensitivity to smaller details. This is because more eigenvectors with smaller eigenvalues will be included. However, using large L may produce large feature vector. To avoid this situation, we choose to use $L = 8$, which gives acceptable results in experiments.

3. EXPERIMENTS

In this section, we report the experimental evaluation in order to demonstrate how the number of scales affects the performance of MPCAF-TOP and to compare the effectiveness of the proposed method with the state-of-the-art. In experiment, the data for filter training are all from the training set only, without involving the test data. Normalized histograms are classified using the nearest neighbor (1NN) classifier with Chi-Square statistic being the measure of dissimilarity. We evaluate the proposed method on two DT databases, namely the UCLA[22] and the DynTex++[14]. Note that the experimental results of other methods are from literature. These results were also obtained with 1NN classifier, unless otherwise stated.

3.1. UCLA database

The UCLA database[22] is comprised of 50 classes of 200 DTs. Each class contains 4 gray-scale sequences with 75 frames of 160×110 pixels. The sequences are all clipped to frames of 48×48 to show the representative dynamics. These 50 classes can be further grouped into 9 classes or 8 classes for performance evaluation. Hence, we have three test scenarios, i.e., 50-class breakdown, 9-class breakdown and 8-class breakdown.

For 50-class breakdown, there are two classification schemes in literature which use different portion of the database as training and test sets. The first one is the leave-one-out scheme. Similar to the works in [22] and [12], a test sequence is classified correctly when its nearest neighbor is one of three remaining sequences from the same class. The second scheme is the four cross-fold classification in which three of the four sequences of each class are used for training and the rest for test. The experiment is repeated four times, each time with a different sequence as the test sample. The recognition rates are averaged as the final result.

Table 1. Comparison of performance of the proposed method to other approaches on UCLA database in a leave-one-out scheme

Method	Recognition Rate(%)
AR-LDS[22]	89.50
MBSIF-TOP[12]	**99.50**
LBP with Michelson contrast[18]	95.00
MPCAF-TOP$_1$	96.50
MPCAF-TOP$_{1,2}$	97.00
MPCAF-TOP$_{1,2,3}$	98.50
MPCAF-TOP$_{1,2,3,4}$	99.00
MPCAF-TOP$_{1,2,3,4,5}$	**99.50**

Table 2. Comparison of performance of the proposed method to other approaches on UCLA database in a four cross-fold validation scheme

Method	Recognition Rate(%)
VLBP[4]	89.50
LBP-TOP[4]	94.50
DL-PEGASOS[14]	99.00
DFS[9]	89.50
3D-OTF[10]	99.25
WMFS[11]	99.12
MBSIF-TOP[12]	**99.50**
CVLBP[16]	93.00
LBP with Michelson contrast[18]	95.00
MPCAF-TOP	**99.50**

(Note: DL-PEGASOS used MMDL + 1NN.)

Table 3. Comparison of performance of the proposed method to other approaches on UCLA database with 9-class breakdown

Method	Recognition Rate(%)
VLBP[4]	96.30
LBP-TOP[4]	96.00
DL-PEGASOS[14]	95.60
3D-OTF[10]	96.32
WMFS[11]	96.95
MBSIF-TOP[12]	98.75
DNG[13]	98.10
CVLBP[16]	96.90
LBP with Michelson contrast[18]	98.35
MPCAF-TOP	**99.15**

(Note: DL-PEGASOS used MMDL + 1NN.)

The results in the leave-one-out scheme are reported in Table 1. We test the proposed method with five scale configurations, namely single scale (3×3), two scales (3×3 + 5×5), three scales ($3 \times 3 + 5 \times 5 + 7 \times 7$), four scales ($3 \times 3$ + $5 \times 5 + 7 \times 7 + 9 \times 9$) and five scales ($3 \times 3 + 5 \times 5 + 7 \times 7 + 9 \times 9 + 11 \times 11$). The subscript of MPCAF-TOP indicate the scale(s) used. We can find that a good result of 96.5% is obtained with single scale MPCAF-TOP. Including more scales (e.g., $L = 8$) should give better result but using five scales yields 99.5% which is still significantly higher performance than some other methods. Therefore, we use the five-scale MPCAF-TOP in the rest of this paper and do not explicitly point out the scales used. When comparing MPCAF-TOP with other state-of-the-art methods, MPCAF-TOP is on par with MBSIF-TOP and much better than others. Note that MBSIF-TOP uses 7 scales while MPCAF-TOP uses only 5 scales.

Results for the four cross-fold validation scheme are listed

Table 4. Comparison of performance of the proposed method to other approaches on UCLA database with 8-class breakdown

Method	Recognition Rate(%)
AR-LDS[22]	54.12
VLBP[4]	91.96
LBP-TOP[4]	93.67
BoS[8]	80.00
3D-OTF[10]	95.80
WMFS[11]	97.18
MBSIF-TOP[12]	97.80
DNG[13]	97.00
CVLBP[16]	95.65
LBP with Michelson contrast[18]	97.50
MPCAF-TOP	**98.26**

in Table 2. The results in this scheme is similar to that in the leave-one-out scheme. MPCAF-TOP is on par with MBSIF-TOP again and better than others.

As for 9-class breakdown, the works of [8] and [14] found that many of the sequences in the UCLA database are semantically capturing the same scenes. Thus, authors of [14] reorganized the UCLA database into nine semantic categories being boiling water (8), fire (8), flowers (12), fountains (20), plants (108), sea (12), smoke (4), water (12) and waterfall (16). Here the numbers in parentheses are the number of sequences in each class. In this scenario, we follow the experimental setting in [14]. Half of the sequences from each class are used for training and the other half for test. A correct classification is obtained when the nearest neighbor of a test sequence is one of the training sequences of the same scene. The experiment is repeated 20 times with random splits of the database. As shown in Table 3, the proposed method gives a result of 99.15%, which is the topmost recognition rate in this scenario. It beats MBSIF-TOP by 0.4% with only 5 scales used.

Because the number of sequences of plants far outnumbers that of other classes in 9-class breakdown, the work in [8] discarded the plants class, leaving 8 classes for performance evaluation. In this scenario, the experimental setting is the same as in 9-class breakdown. Half of the database is used for training and the other half for test. The experiment is repeated 20 times. The results shown in Table 4 clearly illustrate the superiority of the proposed descriptor to some other approaches.

3.2. DynTex++ database

The DynTex++ database[14] is a compiled version of DynTex database[23], providing several appealing properties.

In this database, there is only one spatio-temporal texture class present in each sequence. The sequences are filtered, preprocessed (e.g., cropped) from their raw data to show its representative dynamics without any background or any other dynamic patterns. All the sequences are labeled to make a benchmark database like the UCLA database. All the sequences in this database are organized into 36 classes, each with 100 sequences. For performance evaluation, we follow the experimental setting in [14]. Half of the 100 sequences of each class are randomly chosen for training and the rest for test. We repeat this experiment 10 times. The recognition rates are averaged as the final result.

Table 5. Comparison of performance of the proposed method to other approaches on DynTex++ database

Method	Recognition Rate(%)
VLBP[4]	87.35
LBP-TOP[4]	89.50
DL-PEGASOS[14]	63.70
DFS[9]	89.90^S
3D-OTF[10]	89.17^S
PCA-cLBP/PI-LBP/PD-LBP[15]	91.90
MBSIF-TOP[12]	**97.17**
DNG[13]	90.20
LBP with Michelson contrast[18]	96.28
MPCAF-TOP	96.52

(*Note*: The superscript "S" means evaluation with SVM classifier. DL-PEGASOS used MMDL + 1NN.)

Table 5 summarizes the results of various methods. It can be found that the proposed method is superior to those using SVM classifier. When compared with VLBP and LBP-TOP, MPCAF-TOP beats them by above 7%. The best result on this database is obtained by MBSIF-TOP (with 7 scales used), 97.17%, which is only 0.65% higher than MPCAF-TOP. MPCAF-TOP achieves 96.52%, demonstrating its effectiveness for DT recognition.

4. CONCLUSION

We proposed a novel approach for dynamic texture recognition using PCA-learned filters on three orthogonal planes. We use PCA to learn multiscale filters to construct a multiscale representation of dynamic texture. This method achieved the best results on the UCLA database and gave comparable result on the DynTex++ database. The proposed method is simple but effective. Future extension will exploit the contrast information among filter responses to improve performance.

5. ACKNOWLEDGMENT

This work is partly supported by the National Natural Science Foundation of China under Grant No. 61472125, the Research Foundation of Chinese Ministry of Education and China Mobile Communications Corporation under Grant No. MCM20122061, Academy of Finland under Grant No. 297732 and the scholarship from China Scholarship Council.

6. REFERENCES

[1] Martin Szummer and Rosalind W Picard, "Temporal texture modeling," in *ICIP*. IEEE, 1996, vol. 3, pp. 823–

826.

[2] Gianfranco Doretto, Alessandro Chiuso, Ying Nian Wu, and Stefano Soatto, "Dynamic textures," *IJCV*, vol. 51, no. 2, pp. 91–109, 2003.

[3] Micha Haas, Joachim Rijsdam, Bart Thomee, and Michael S Lew, "Relevance feedback: perceptual learning and retrieval in bio-computing, photos, and video," in *SIGMM Workshop*. ACM, 2004, pp. 151–156.

[4] Guoying Zhao and Matti Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *TPAMI*, vol. 29, no. 6, pp. 915–928, 2007.

[5] B Uğur Töreyin, Yiğithan Dedeoğlu, Uğur Güdükbay, and A Enis Cetin, "Computer vision based method for real-time fire and flame detection," *Pattern recognition letters*, vol. 27, no. 1, pp. 49–58, 2006.

[6] Renaud Péteri and Dmitry Chetverikov, "Dynamic texture recognition using normal flow and texture regularity," in *Pattern Recognition and Image Analysis*, pp. 223–230. Springer, 2005.

[7] Dmitry Chetverikov and Renaud Péteri, "A brief survey of dynamic texture description and recognition," in *Computer Recognition Systems*, pp. 17–26. Springer, 2005.

[8] Avinash Ravichandran, Rizwan Chaudhry, and René Vidal, "View-invariant dynamic texture recognition using a bag of dynamical systems," in *CVPR*. IEEE, 2009, pp. 1651–1657.

[9] Yong Xu, Yuhui Quan, Haibin Ling, and Hui Ji, "Dynamic texture classification using dynamic fractal analysis," in *ICCV*. IEEE, 2011, pp. 1219–1226.

[10] Yong Xu, Sibin Huang, Hui Ji, and Cornelia Fermüller, "Scale-space texture description on sift-like textons," *Computer Vision and Image Understanding*, vol. 116, no. 9, pp. 999–1013, 2012.

[11] Hui Ji, Xiong Yang, Haibin Ling, and Yong Xu, "Wavelet domain multifractal analysis for static and dynamic texture classification," *TIP*, vol. 22, no. 1, pp. 286–299, 2013.

[12] Shervin Rahimzadeh Arashloo and Josef Kittler, "Dynamic texture recognition using multiscale binarized statistical image features," *TMM*, vol. 16, no. 8, pp. 2099–2109, 2014.

[13] Adin Ramirez Rivera and Oksam Chae, "Spatiotemporal directional number transitional graph for dynamic texture recognition," *TPAMI*, vol. 37, no. 10, pp. 2146–2152, 2015.

[14] Bernard Ghanem and Narendra Ahuja, "Maximum margin distance learning for dynamic texture recognition," in *ECCV*. Springer, 2010, pp. 223–236.

[15] Jianfeng Ren, Xudong Jiang, and Junsong Yuan, "Dynamic texture recognition using enhanced lbp features.," in *ICASSP*, 2013, pp. 2400–2404.

[16] Deepshikha Tiwari and Vipin Tyagi, "Dynamic texture recognition based on completed volume local binary pattern," *Multidimensional Systems and Signal Processing*, vol. 27, no. 2, pp. 563–575, 2016.

[17] Albert Abraham Michelson, *Studies in optics*, Courier Corporation, 1995.

[18] Deepshikha Tiwari and Vipin Tyagi, "A novel scheme based on local binary pattern for dynamic texture recognition," *Computer Vision and Image Understanding*, 2016.

[19] Graham W Taylor, Rob Fergus, Yann LeCun, and Christoph Bregler, "Convolutional learning of spatio-temporal features," in *ECCV*. Springer, 2010, pp. 140–153.

[20] Xing Yan, Hong Chang, Shiguang Shan, and Xilin Chen, "Modeling video dynamics with deep dynencoder," in *ECCV*. Springer, 2014, pp. 215–230.

[21] Tsung-Han Chan, Kui Jia, Shenghua Gao, Jiwen Lu, Zinan Zeng, and Yi Ma, "Pcanet: A simple deep learning baseline for image classification?," *TIP*, vol. 24, no. 12, pp. 5017–5032, 2015.

[22] Payam Saisan, Gianfranco Doretto, Ying Nian Wu, and Stefano Soatto, "Dynamic texture recognition," in *CVPR*. IEEE, 2001, vol. 2, pp. II–58.

[23] Renaud Péteri, Sándor Fazekas, and Mark J Huiskes, "Dyntex: A comprehensive database of dynamic textures," *Pattern Recognition Letters*, vol. 31, no. 12, pp. 1627–1632, 2010.