IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 21 April 2023, accepted 18 May 2023, date of publication 23 May 2023, date of current version 9 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3279195

RESEARCH ARTICLE

Dynamic Texture Classification Using Directional Binarized Random Features

XIAOCHAO ZHAO, FANG XU, YI MA, ZHEN LIU, MIN DENG, UMER SADIQ KHAN, AND ZENGGANG XIONG

School of Computer and Information Science, Hubei Engineering University, Xiaogan 432000, China
Institute for AI Industrial Technology Research, Hubei Engineering University, Xiaogan 432000, China

Corresponding author: Fang Xu (xf2012@whu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61972136 and Grant 81801784, in part by the Natural Science Foundation of Hubei Province under Grant 2020CFB497, in part by the Hubei Provincial Department of Education Outstanding Youth Scientific Innovation Team Support Foundation under Grant T201410 and Grant T2020017, in part by the Natural Science Foundation of Education Department of Hubei Province under Grant B2020149, in part by the Science and Technology Research Project of the Education Department of Hubei Province under Grant Q20222704, and in part by the Natural Science Foundation of Xiaogan City under Grant XGKJ2022010095 and Grant XGKJ2022010094.

ABSTRACT Dynamic texture description has been studied extensively due to its wide applications in the field of computer vision. Local binary pattern (LBP) and its various variants account for a large part of dynamic texture description methods because of its advantages, such as good discriminability and low computational complexity. However, many LBP-based methods directly extract feature from pixel intensities and only use a proportion of pixels in a local neighborhood. And their good classification performance is usually achieved at the cost of high feature dimensionality, which would limit their application scenarios. We argue that extracting features from the gradient domain will capture more discriminative features due to the additional directional information, and that making use of all the pixels in a local neighborhood would improve performance. In this paper, we propose a simply but effective dynamic texture descriptor that inherits the advantages of LBP while excluding its disadvantages. The proposed method consists of four stages of data processing: 1) gradients extraction; 2) random feature extraction from gradients; 3) binary hashing of directional random features; and 4) histogramming. Gaussian first-order derivatives are used as gradient filters such that stable gradients could be generated. Then random projection is applied to extract random features from each gradients. Both the above two stages are conducted via 3D filtering, and thus they are efficient. Thirdly, the random features from each gradient are binarized and encoded into integer codes, from which a histogram is built. Finally, the histograms from each gradient are concatenated into a feature vector. Because we use 8-bit codes, The feature dimensionality is very low. We evaluate the proposed method on three benchmark dynamic texture datasets with various test protocols. The results demonstrate its effectiveness and efficiency when comparing to many state-of-the-art methods.

INDEX TERMS Dynamic texture, feature extraction, gradient, Gaussian derivative, local binary pattern, random feature.

I. INTRODUCTION

Dynamic textures (DTs) are the extension of static textures in the temporal domain [1]. DTs are videos consisting of moving scenes that exhibit some stability in both spatial and temporal domains. DT examples include sea waves, swaying trees, flames, smoke, and fountains. DT description has been extensively studied by many researchers in the last 20 years due to its applications in many computer vision tasks, such as human interaction [2], visual tracking [3], facial analysis [4], [5], [6], lip reading [7], fire detection [8], crowd management [9], and traffic monitoring [10]. Research works related to DT could be roughly grouped into three classes of segmentation, synthesis, and classification. In this paper,

The associate editor coordinating the review of this manuscript and approving it for publication was Gangyi Jiang.

we focus on the task of DT classification and study how to construct a compact and effective DT descriptor.

To conduct effective DT classification, DTs must be processed and encoded into feature vectors which can be used for classification. However, DT description is not an easy task [11]. Spatial appearance in a DT would be affected by changes of illumination, viewpoint, scale, and rotation. Moreover, turbulent and non-directional motions, and similar motions would make it hard to extract stable and discriminative temporal features. Therefore, DT classification is more challenging than the static case. Considering that DTs are static textures with motion, many researchers tried to extend existing texture descriptors to the spatio-temporal domain such that both spatial and temporal features can be captured. Such an influential work was conducted by Zhao and Pietikainen [12], who extended the famous static descriptor, LBP [13], for DT description and proposed volume LBP (VLBP) to capture spatial and temporal features simultaneously. They later proposed to view a DT along three axes (i.e., the horizontal, vertical, and temporal axes) and treated it as three image sequences, from which LBP features were extracted and concatenated as the final feature vector (denoted as LBP-TOP) [4]. VLBP and LBP-TOP turned out to be a great success for DT classification and started a thread of research that applies binary encoding in a local neighborhood for DT description [5], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24].

VLBP first calculates the differences between a central pixel and P sampled pixels in its spatio-temporal neighborhood, the signs of which are further encoded into a binary string (*i.e.*, a VLBP code). Then a histogram is built from all the codes generated from a given DT. Obviously, the feature dimensionality of VLBP is 2^P. According to experimental results in [4], a relatively larger P would bring good performance while the cost is high dimensionality. To mitigate this problem, LBP-TOP extracts LBP features from three orthogonal planes (TOP) respectively and its feature dimensionality is 3×2^P. Other LBP variants basically follow the sampling schemes of VLBP and LBP-TOP while the binary code generation process are generally improved. Rahtu et al. generated feature codes via phase quantization [14]. Tiwari and Tyagi improved the performance of VLBP and LBP-TOP by introduce additional information such as the central pixel, absolute values of pixel differences, and local contrast [18], [19]. Zhao et al. proposed to count the number of ones in the binary string instead of directly using the binary code, and they also included additional information about the central pixel and absolute values of pixel differences (denoted as CVLBC) [5]. Hong et al. [17] and Ren et al. [15] analyzed the salience and reliability of LBP features respectively, and only the most contributive features are adopted for DT classification. Some other works [16], [20], [21], [22] improved VLBP or LBP-TOP by incorporating a learning process. Arashloo and Kittler proposed to filter images from three orthogonal planes with learned filters at multiple scales, and the filter responses were binarized and encoded into LBP codes (called MBSIF-TOP) [16]. Later, they learned a set of multi-scale filters from each of the three image sequences via principal component analysis (PCA), and all the filters were organized into a network structure, of which the outputs were encoded into binary codes (called PCANet-TOP) [21]. Zhao et al. [20] introduced a PCA version of the work in [16], in which the filters are learned through PCA (called MPCAF-TOP). Zhao et al. [22] argued that those TOP-based approaches only use a proportion of pixels in a local neighborhood, where some discriminative information may be lost. As a result, they decided to directly process DTs with 3D filters rather than using the TOP scheme (denoted as B3DF). In [22], they first learned 3D filters from randomly sampled 3D blocks and then filtered DT with these filters, from which the filter responses are binarized. Additionally, they also binarized central pixels and absolute values of pixel differences, which were finally used for joint histogramming.

In summary, VLBP, LBP-TOP, and their variants (*i.e.*, LBP-based DT descriptors) do have several advantages, such as good representation capability, robustness to illumination change, and computational simplicity. However, their disadvantages also exist. The first disadvantage is the high dimensionality problem. To achieve good classification performance, the number of sample pixels or the length of the binary string should be relatively large. This problem also stems from multi-scale processing and joint histogramming, *e.g.*, the works in [5], [16], [20], and [22]. The dimensions of feature vectors produced by VLBP [4], MBSIF-TOP [16], CVLBC [5], MPCAF-TOP [20], and B3DF [22] are 16384, 6144, 11250, 3840, 65536, respectively. High dimensionality would largely restrict their application in scenarios with limited computational resource, such as mobile devices. As for the second disadvantage, we argue that all the above LBP-based DT descriptors directly process pixel values without utilizing the directional information in the gradient domain, which could contribute to performance improvement. This argument is supported by [25], in which incorporating Gaussian gradients and CLBP brings significant performance improvement. The third problem is raised by Zhao et al. [22], *i.e.*, those methods that need to sample pixels in a neighborhood, such as VLBP [4], LBP-TOP [4], novel LBP [18], and CVLBC [5], only make use of a proportion of the pixels in the neighborhood around a central pixel, ignoring some pixels that may also contain discriminative information. This point is proved to be reasonable because those methods [16], [20], [22] using filtering technique rather than sampling pixels make use of all the pixels in a neighborhood and generally provide better performance.

Additionally, it should also be discussed whether incorporating a learning process is necessary. Generally speaking, learning will bring some superiority. However, this does not always stand, *e.g.*, the novel LBP method [18] outperforms a dictionary-learning method [26] by 2.88% on the challenging DynTex++ [27] dataset. Specifically, the former uses

the simple nearest neighbor classifier while the latter adopts the support vector machine classifier. One may argue that applying deep learning could provide extremely good performance. However, such good performance usually depends on a prerequisite that sufficient training data are available. Unfortunately, Zhao et al. [22] tried to train three networks from scratch on the DynTex++ dataset (containing 3600 DTs) and the results were poor. The reason behind the poor results may be lack of training data. Another point worth mentioning is that learning based method may not generalize well to new data due to its dependence on the original training data. When it comes to new data, learning based methods usually need re-training while learning-free methods only need to change some parameters. Therefore, we argue that a well-designed DT descriptor could also provide good performance without using any learning process.

According the above analysis about VLBP, LBP-TOP, and their variants, we question their predominance with regard to DT representation. In this paper, we aim to build a DT descriptor (directional binarized random features, DBRF) that have the following characteristics: (1) computational simplicity like LBP-based DT descriptors, (2) low dimensionality (less than 1000), (3) processing conducted on DT gradients, (4) using 3D filter responses instead of pixel differences, and (5) learning-free. Specifically, the proposed method consists of the following stages of processing: (1) a given DT is first filtered with 3D Gaussian first-order derivatives to obtain stable DT gradients along X, Y, and Z axes, respectively; (2) random projection is applied to each of the three DT gradients, generating a feature vector at each valid position in space; (3) for each DT gradient, all the feature vectors are binarized according to their signs (1 for positive and 0 for negative) and a histogram is built from these binary codes; (4) three histograms are concatenated into a final feature vector, which will be used for DT classification. And the proposed method uses three parameters, *i.e.*, the size of 3D Gaussian kernel (standard deviation is accordingly determined), size and number of 3D random filters. They could be conveniently set up according to the characteristics of data used in some computer vision application. It is obvious that the proposed method is simple and learning-free. A small number of 3D random filters is adopted to ensure its feature dimensionality is less than 1000. Its effectiveness and the improvement brought by extracting feature in the gradient domain are justified by extensive experiments.

The remainder of this paper is as follows. Related work is briefly summarized in Section II. The proposed DBRF descriptor is explained in Section III. Experimental results are provided in Section IV. Finally, Section V concludes this paper.

II. RELATED WORK

DT description has been studied for over two decades and plenty of DT description methods have been reported. According to the recently published survey on DT representation [11], DT description approaches can be roughly arranged into six categories: optical-flow-based, model-based, geometry-based, filter-based, local-feature-based, and learning-based. The proposed method belongs to the last category. In this section, we mainly introduce those recently published methods in each category.

A. OPTICAL-FLOW-BASED METHODS

The methods in this category aim to capture the motion features of pixels in the spatio-temporal domain. Early methods [28], [29], [30] estimates the normal flow from DTs, from which feature vector are constructed. However, those methods are not evaluated on standard DT datasets. And the normal flow estimation process depends on the assumption of brightness constancy and local smoothness, which are hard to justify. Therefore, we skip their details. Different from the above methods that only use motion features, Nguyen et al. made use of both motion and appearance, and proposed two methods: (1) features of directional trajectories in accordance with motion angle patterns (FD-MAP) [31] and (2) directional dense trajectory patterns (DDTP) [32]. Both the two methods achieve good performance. Couto and Barcelosuto [33] extracted singular patterns [34], which were further pooled to form a DT descriptor called DT-SP.

B. MODEL-BASED METHODS

Model-based methods attempt to construct an underlying model that generates DTs and to use the model parameters for DT description. The pioneering work was done by Doretto et al. [1], [35], who proposed the linear dynamical system (LDS) as a DT generative model. Other model-based methods are generally variants of the LDS method. Chan et al. proposed a series of LDS-based methods for DT analysis. They incorporated a probabilistic kernel with LDS [36]. They also proposed to estimate LDS parameters via kernel-PCA instead of PCA because a non-linear function would describe complex motions better [37]. Later, they presented a generative model called DT mixtures, which were further clustered for DT description (denoted as HEM-DTM) [38]. Instead of measuring the distance between two sets of model parameters, Ravichandran et al. [39], [40] estimated LDS parameters from sampled sub-DTs and a low-dimensional Euclidean embedding of these models was conducted such that clustering on these models could be applied to generate a codebook, which is used for DT representation. Specifically, two clustering algorithms (hierarchical K-means and K-Medoid) are adopted and thus two methods (denoted as BoS-HK [39] and BoS-KM [40]) are reported. To improve the BoS methods, Mumtaz et al. [41] proposed BoS tree (BoST) based on their early work [38]. Some other works also applied various techniques to construct codebooks from LDS models [42], [43], [44], [45].

Besides the above LDS-based models, some other modeling techniques have also been explored. Ribas et al. [46] adopted the diffusion network and model DT features as a directed network. Gonçalves et al. [47] built a complex

network using the Euclidean distance between related pixels [47]. Ribas and Bruno [46] adopted the deterministic partially self-avoiding walk for DT feature extraction. The hidden Markov model [48], [49] and the finite mixture of von Mises distributions [50] have also been utilized for DT description.

C. GEOMETRY-BASED METHODS

Methods in this class mainly focus on the structure information in DTs. Mandelbrot [51] proposed the notion of fractal structure that means an object displays self-similarity across multiple scales. By viewing a DT as a 3D volume, Xu et al. [52], [53] proposed the dynamic fractal spectrum (DFS), which captured the self-similarity information in both the 3D volume and 2D slices of the 3D volume. The DFS method was further extended into two new methods, i.e., 3D oriented transform feature (3D-OTF) [54] and wavelet domain multiple fractal spectrum (WMFS) [55]. As a specialized concept in fractal geometry, lacunarity [51] is used to measure how patterns fill space. Quan et al. [56] made use of this measure and proposed a DT descriptor called spatio-temporal lacunarity spectrum (STLS). 2D discrete wavelet transform uses separable filters and can decompose an image into a low-frequency sub-band and three high-frequency sub-bands, which convey different structure information. Dubois et al. used three filters to decompose DTs in multiple scales and the absolute values of wavelet coefficients in each scale and sub-band are averaged to form a DT descriptor [57]. Later, they did a similar work using curvelet transform [58]. Additionally, Baktashmotlagh et al. [59] applied non-linear stationary subspace analysis to separate the stationary parts of DTs from the non-stationary parts and then only the stationary parts were utilized for DT representation.

D. FILTER-BASED METHODS

Methods in this class usually first filter DTs with some filters and then construct DT feature vectors from filter responses. The filters can be learned or non-learned. Methods using learned filters include MBSIF-TOP [16], MPCAF-TOP [20], and B3DF [22]. Details could be found in Section I. There are also several methods using non-learned filters. Derpanis and Wildes [60] used Gaussian-gradient filters to extract spatio-temporal oriented energy for DT description. Rivera and Chae [61] proposed to filter DTs with 2D/3D Kirsch masks, of which the filter responses were adapted to a directional transitional number graph (DNG) for DT classification. Jansson and Lindeberg [62] used space-time separable kernels to extract spatio-temporal receptive field (STRF) responses, and then PCA is applied for dimension reduction. Nguyen et al. [25], [63], [64], [65] incorporated completed LBP with Gaussian gradients, and reported a series of methods for DT representation.

E. LOCAL-FEATURE-BASED METHODS

Almost all the methods in this class are variants of LBP. Several works [4], [5], [18], [19] have been introduced in Section I. Hereafter, we briefly introduce other LBP-based methods. Ren et al. [66] proposed the data-driven LBP (DDLBP), in which LBP structures are optimized globally. Sun et al. [67] combined lacunarity analysis [51] with local ternary pattern (LTP) [68] and proposed the LTP-Lacunarity (LTP-Lac) features. On the basis of LBP features, Xie and Fang [69] applied collaborative representation to build the video set based collaborative representation. Nguyen et al. [70] extended completed local structure pattern (CLSP) [71] with the TOP scheme (CLSP-TOP). Later, they further extended CLSP and proposed the complete statistical adaptive pattern (CSAP-TOP) [72]. In 2020, they proposed three other methods: (1) hierarchical local pattern [73], (2) local Rubik pattern (LRP) [74], and (3) applying the completed version of local derivative pattern [75] to extract momental directional patterns [76]. Tiwari and Tyagi [77] combined Weber's law with LBP features (WLBPC). Meantime, they determined the local threshold according to local neighborhood differences and proposed the edge-weighted local structure pattern (EWLSP) descriptor [78].

F. LEARNING-BASED METHODS

Methods in this class mainly uses the popular and effective deep learning techniques and dictionary learning techniques. Trand et al. [79] proposed to train a 3D convolutional neural network (C3D) on a large set of videos and applied it for DT classification. Note that C3D does not use DT datasets for training because the benchmark DT datasets are not large enough to train a deep network. As a result, Several works [80], [81], [82] proposed to transfer the representation power from existing deep networks (*e.g.*, AlexNet [83], GoogleNet [84], and VGGNet [85]) that are trained on other large datasets. Qi et al. [80] fed DT frames and frame differences at an interval into VGGNet, respectively. Statistical features were respectively extracted from two kinds of outputs and concatenated as the spatio-temporal transferred ConvNet features (st-TCoF). Hong et al. [81] used the outputs of the fully connected layer and the convolutional layer of VGGNet as intermediate features, which were further encoded by Fisher vector [86] encoding. Andrearczyk and Whelan [82] chose the TOP scheme and fed each of the three image sequences into AlexNet/GoogleNet, of which the outputs were first aggregated and then concatenated into a feature vector (denoted as DT-CNN). There also exists some network-based methods that does not actually use typical deep learning techniques. Wang and Hu [87] adopted the deep belief network to extract high-level features from chaotic and low-level features, while Zrira et al. [88] fed LBP features into such a network. Koleini et al. [89] used Bayesian network to combine multiple features. Hadji and Wildes [90] organized a set of pre-defined 3D Gaussian third-order derivative filters into a convolutional network with pooling layers (SOE-Net) and achieved good DT classification performance. Note that SOE-Net is actually learning-free. Junior et al. [91] adopted the randomized neural network that has only two layers.

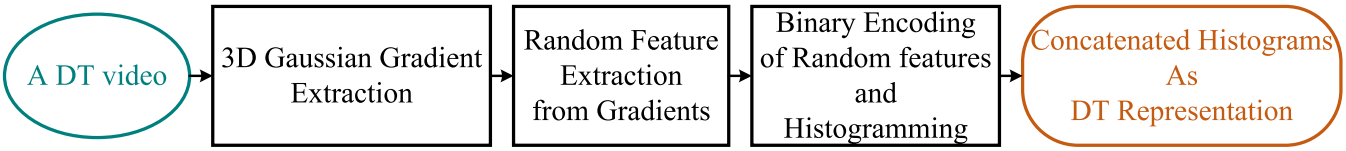

FIGURE 1. Overview of the proposed method.

As for dictionary learning, Harandi et al. [92] applied sparse coding on Grassmann manifolds to learn a dictionary from LBP-TOP features. Quan et al. also utilized sparse coding technique and proposed orthogonal tensor dictionary learning (OTDL) [93] and equiangular kernel dictionary learning (EKDL) [26] for DT representation. However, it remains unclear how they choose a portion of atoms from the learned dictionaries. Fisher vector [86] encoding has also been used to build dictionaries from DTs. Zhao et al. [23] first extracted 3D random features from sampled 3D blocks and then trained a Gaussian mixture model (GMM), after which Fisher vector encoding was applied to generate DT feature vectors (denoted as 3DRF). Later, Xiong et al. [24] extended 3DRF by replacing the random filters with those learned via independent component analysis. However, the performance improvement is marginal.

III. THE PROPOSED METHOD

In this section, we provide the details about the derivation of the proposed DT representation method. The processing framework is first introduced and then each of its components is explained in details.

A. OVERVIEW OF THE PROPOSED METHOD

We follow the notion of local processing and aim to propose a simple but effective DT representation. Our main idea is to extract binarized random features from Gaussian gradients of a given DT (denoted as DBRF). The proposed framework is illustrated in Fig. 1.

First, we generate 3D Gaussian first-order derivatives along X, Y, and Z directions in the spatio-temporal domain. By viewing a DT video as a 3D volume, three directional gradients are extracted. Due to the smoothing property of Gaussian kernel, the gradients are stable and robust against noise to some extent. Extracting gradients introduces extra directional information, which has been proved to be beneficial for DT presentation [65]. Second, random projection is conducted at every valid position in the three gradients, respectively. Specifically, the local neighborhood of a certain size at each valid position is convolved respectively with L 3D random filters, generating a vector consisting of L features. As a dimensionality reduction method, a small number of random projections could capture enough salient information in the signal [94]. Moreover, its learning-free property makes it very convenient for feature extraction, especially for resource-restricted scenarios. Third, the signs of the L random features are encoded into a L-bit binary code. And a histogram is constructed from the binary codes with respect to each of the three gradients. After concatenating the three histograms, a global feature vector of 3×2^L dimensions is obtained to represent the given DT. Details about the above three components are described in the following sections.

B. 3D GAUSSIAN GRADIENTS EXTRACTION

As we challenge many existing LBP-based methods that do not make use of directional information and directly extract features from raw pixels, we decided to extract features from gradients such that the useful directional information is utilized. As for gradient computation, there exists at least five operators, *i.e.*, Sobel operator, Prewitt operator, central difference operator, intermediate difference operator, and Gaussian first-order derivative. According to our previous experience on face recognition task [95] and the success of Nguyen et al. [65], we adopt the Gaussian first-order derivatives for gradient extraction.

Here we briefly introduce how 3D Gaussian first-order derivatives are computed. A Gaussian kernel in 3D domain is defined as follows.

$$G(x, y, z, \sigma) = \frac{1}{(\sigma\sqrt{2\pi})^3} \exp(-\frac{x^2 + y^2 + z^2}{2\sigma^2}), \quad (1)$$

where $x, y, z \in [-3\sigma, 3\sigma]$ indicate the coordinate in the local neighborhood and σ is the standard deviation. According to (1), three 3D Gaussian first-order derivatives along x, y, and z directions can be computed by

$$G_x(x, y, z, \sigma) = \frac{-x}{\sigma^2} \frac{1}{(\sigma\sqrt{2\pi})^3} \exp(-\frac{x^2 + y^2 + z^2}{2\sigma^2}), \quad (2)$$

$$G_y(x, y, z, \sigma) = \frac{-y}{\sigma^2} \frac{1}{(\sigma\sqrt{2\pi})^3} \exp(-\frac{x^2 + y^2 + z^2}{2\sigma^2}), \quad (3)$$

$$G_z(x, y, z, \sigma) = \frac{-z}{\sigma^2} \frac{1}{(\sigma\sqrt{2\pi})^3} \exp(-\frac{x^2 + y^2 + z^2}{2\sigma^2}). \quad (4)$$

Given a DT D of size $X \times Y \times Z$ ($X \times Y$ is the spatial size and T is the temporal size), three directional gradients are obtained by convolving it respectively with G_x, G_y, and G_z as

$$g_x = G_x * D, \quad (5)$$

$$g_y = G_y * D, \quad (6)$$

$$g_z = G_z * D, \quad (7)$$

where $*$ is the convolution operator. Note that no padding is applied for border pixels. Assume that the 3D Gaussian kernel

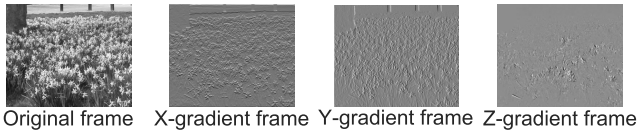

FIGURE 2. Example of a DT frame and its corresponding gradient frames from g_x, g_y, and g_z with $\sigma = 0.7$.

size is $k \times k \times k$, where k is an odd number. Then, the size of the three Gaussian gradient volumes is $(X - \lfloor \frac{k}{2} \rfloor) \times (Y - \lfloor \frac{k}{2} \rfloor) \times (T - \lfloor \frac{k}{2} \rfloor)$. As an illustration, it can be observed in Fig. 2 that three types of directional features are captured by 3D Gaussian first-order derivatives.

C. RANDOM FEATURE EXTRACTION

Given a 3D gradient volume $g \in \{g_x, g_y, g_z\}$, local 3D cubes of size $d \times d \times d$ are densely sampled at every valid position. As we do not apply border padding, there are in total $N = (X - \lfloor \frac{k}{2} \rfloor - \lfloor \frac{d}{2} \rfloor) \times (Y - \lfloor \frac{k}{2} \rfloor - \lfloor \frac{d}{2} \rfloor) \times (T - \lfloor \frac{k}{2} \rfloor - \lfloor \frac{d}{2} \rfloor)$ cubes, denoted as $\{\mathbf{x}_i\}_{i=1}^{N}$. To prevent the potential problem that the extracted random features are all positive or all negative, each cube $\mathbf{x}_i$ is normalized to have zero mean by subtracting the local mean gradient value $x_i^m = \frac{1}{d^3} \sum_{x=1}^{d} \sum_{y=1}^{d} \sum_{t=1}^{d} \mathbf{x}_i(x, y, t)$ from each gradient value in it.

To conduct random projection, we randomly generate a set of L 3D filters of size $d \times d \times d$, denoted as $\{\mathbf{w}_l\}_{l=1}^{L}$. For each normalized 3D cube, a set of L random features $\{f_l^i\}_{l=1}^{L}$ are obtained by convolving it with each of the L random filters as

$$f_l^i = \mathbf{w}_l * (\mathbf{x}_i - x_i^m), \tag{8}$$

$(\mathbf{x}_i - x_i^m)$ means subtracting x_i^m from each gradient value in $\mathbf{x}_i$.

It is obvious that we extract features directly from the local 3D cube rather than three orthogonal planes. The main difference is that we make use of all the values in a local neighborhood while many TOP-based methods uses only a proportion of them. This point is clearly illustrated in Fig. 3. We argue that those ignored components also contain information that is beneficial for DT representation.

Additionally, the sampling process and zero-mean normalization explained above would be quite time-consuming if the two processes are conducted in a straight forward manner. But, there exists an efficient way to do the same thing. Specifically, convolving a zero-mean-normalized cube with a given 3D filter is equivalent to convolving the original cube with a zero-mean-normalized 3D filter. Please refer to our early work [22] for the detailed derivation. As a result, we directly convolve the gradient g with zero-mean-normalized 3D random filters in implementation, which would make the feature extraction process fast to compute (because the time-consuming sampling step is skipped).

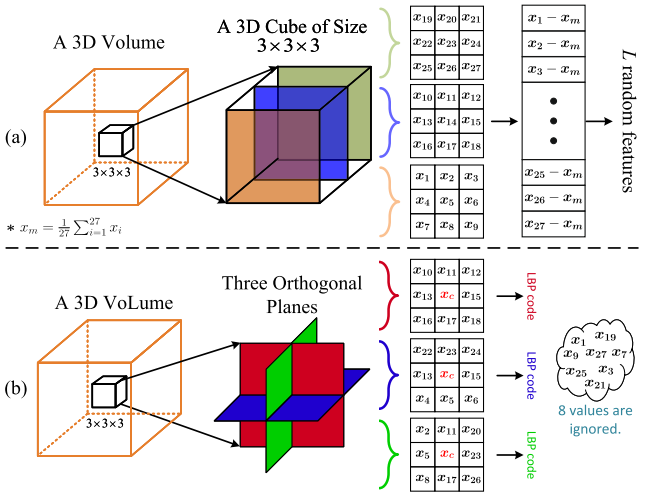

FIGURE 3. Comparison of how random features are extracted from a 3D cube of size 3 × 3 × 3 (top) and how LBP-TOP processes the same 3D cube (bottom).

D. BINARY ENCODING AND HISTOGRAMMING

After obtaining a bunch of vectors of L random features from a DT, the task is to aggregate those local features into a global feature vector that could be used for DT classification. Several commonly used techniques are dictionary learning [26], [92], [93], k-means clustering [96], Fisher vector encoding [23], and histogramming on local binary codes [22], [25]. The former three techniques require a training process to generate a dictionary or a set of cluster centers (used as a dictionary), which is not in line with our goal. Therefore, we decide to encode a random feature vector into a binary string as

$$DBRF_{g,\sigma,d,L} = \sum_{l=1}^{L} 2^{l-1} s(f_l), \tag{9}$$

where $g \in \{g_x, g_y, g_z\}$ the function $s(x)$ returns 1 if $x > 0$, otherwise 0.

Then, for each gradient, a histogram of 2^L bins are constructed from the corresponding binary codes by

$$H_{g,\sigma,d,L}^{o}(j) = \sum_{x,y,t} I(DBRF_{g,\sigma,d,L} = j), \tag{10}$$

where $j \in [0, 2^{L-1}]$ is an integer (*i.e.*, a binary code) and the function $I(t)$ return 1 if t is true, otherwise 0. It is obvious that the above histogramming process actually counts the probability distributions of all binary codes.

In some real application scenarios, two DTs of different sizes belonging to the same class would be mis-classified if the original histogram $H_{g,\sigma,d,L}^{o}$ is directly used. To avoid this problem caused by the DT size, the histogram needs to be normalized as

$$H_{g,\sigma,d,L}(j) = \frac{H_{g,\sigma,d,L}^{o}(j)}{\sum_{j'} H_{g,\sigma,d,L}^{o}(j')}, \tag{11}$$

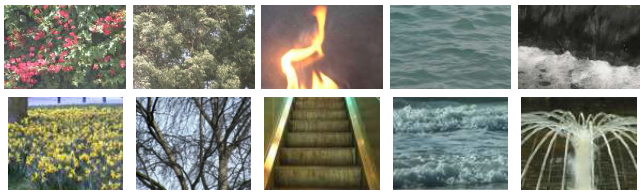

FIGURE 4. DT examples from the two datasets of UCLA (top) and DynTex (bottom).

such that the histogram is no longer relevant to the DT size and a coherent DT representation is generated. Finally, a global feature vector is obtained by concatenating the three histograms as

$$H_{\sigma,d,L} = [H_{g_x,\sigma,d,L}, H_{g_y,\sigma,d,L}, H_{g_z,\sigma,d,L}], \qquad (12)$$

which contains 3×2^L dimensions and will be used for DT classification.

IV. EXPERIMENTS

In this section, we evaluate the proposed DBRF descriptor on three benchmark DT datasets, which are UCLA [35], DynTex [97], and DynTex++ [27]. To assess the representation power of our DBRF descriptor, we deliberately adopt the simple nearest neighbor (NN) classifier for DT classification, such that the emphasis is on the contribution of the DBRF descriptor. Moreover, the Chi-square statistic is used as the dissimilarity measure. Comparison with state-of-the-art results are provided.

A. DATASETS AND EVALUATION PROTOCOLS

Here we detailedly introduce how the three benchmark datasets are formed, as well as the corresponding evaluation protocols, according to which experiments are conducted. A few examples from the UCLA and DynTex datasets are shown in Fig. 4. Note that those color videos are converted into gray-scale ones before conducting experiments. As for classification, the NN classifier is applied unless otherwise specified.

1) UCLA DATASET

This dataset is consisting of 200 DT videos of size $160 \times 110 \times 75$, which are originally grouped into 50 classes with four videos belonging to each class. We choose a preprocessed version[1], in which all the videos are cropped to be of size $48 \times 48 \times 75$ such that only the key dynamical properties are captured. As for experimental evaluation, there are four protocols as follows.

- **50-class leave-one-out (LOO) classification**: Each time one DT is used as the test sample and the other 199 DTs are for training. The final classification accuracy is obtained via dividing the number of correct classifications by 200.
- **50-class 4-fold cross validation (4CFV)**: 200 DTs are split into four groups, each of which contains one DT from each class. The split scheme is attached with the data. Every time one group is used as test data while the other groups are the training data. Four classification rates are averaged as the final rate.
- **9-class half-to-half validation**: Due to the observation that some DTs in different classes could be semantically categorized into the same class, 200 DTs are re-organized into nine classes, which are smoke (4), fire (8), boiling (8), water (12), flower (12), sea (12), waterfall (16), fountain (20), and plant (108) (the number in parentheses means the group size). Then half of the DTs in each class are randomly selected from training and the other half for test. This evaluation is repeated 20 times and the classification rates are averaged.
- **8-class half-to-half validation**: In 9-class breakdown, the number of DTs in plant class is 108, which may cause biased results. Therefore, the plant class is discarded and the half-to-half validation is repeated on the remaining data for 20 times.

2) DynTex DATASET

This dataset is a large and challenging dataset consisting of 679 videos of size $352 \times 288 \times 250$. Part of this data are selected and arranged into four subsets for evaluation with the LOO classification scheme. The four subsets are organized as follows.

- **DynTex35**: This subset is the early version of the DynTex dataset, and includes 35 videos of size $400 \times 300 \times 250$. Each video belongs to an unique class. Following the setting in [12], each video is viewed in 3D space and cropped at the point ($x = 170$, $y = 130$, and $t = 100$), bringing forth eight non-overlapping sub-videos. Together with two other sub-videos by only cropping the original video at $t = 100$, there are 10 sub-videos of different sizes in each class. And sub-videos of the same size are put into one group (ten groups in total). Each time one group is used for test and the other nine ones are for training. The NN classifier is applied for classification with our DBRF. Some existing methods adopt the nearest class center (NCC) classifier. When using the NCC classifier, training feature vectors belonging to the same class are averaged as the corresponding class center. A test feature vector is only compared to the class center and classified into the class, to whose class center the distance is minimal. Using NCC may make the classification process a little bit more challenging. Finally, ten classification rates are averaged.
- **Alpha**: This subset contains 60 videos belonging to three classes of grass, sea and trees. Each class has 20 videos.
- **Beta**: This subset is consisting of 162 videos, which belong to ten classes of sea (20), vegetation (20),

[1] http://www.bernardghanem.com/datasets

TABLE 1. Analysis of data used in each evaluation protocol with respect to DT size and intra-class variation.

Dataset	DT size	Intra-class variation
UCLA 50-class	small	small
UCLA 9-class	small	moderate
UCLA 8-class	small	moderate
DynTex35	large	large
Alpha	large	large
Beta	large	large
Gamma	large	large
DynTex++	small	large

trees (20), flags (20), calm water (20), fountains (20), traffic (9), smoke (16), escalator (7), and rotation (10).

- **Gamma**: This subset contains 275 videos, which also belong to ten classes of flowers (29), sea (38), naked trees (25), foliage (35), escalator (7), calm water (30), flags (31), grass (23), traffic (9), and fountains (37).

3) DynTex++ DATASET

This dataset is composed from 345 videos of DynTex by clipping them into 3600 sub-videos of size $50 \times 50 \times 50$. These sub-videos are preprocessed and then organized into 36 classes, each with 100 sub-videos. The evaluation on this dataset follows the half-to-half validation scheme, *i.e.*, randomly selected half of the data in each class for training and the other half for test. This evaluation is repeated 10 times and the classification rates are averaged as the final result.

B. PARAMETER SETTING

As shown in (12), the proposed DBRF descriptor depends on three parameters, *i.e.*, the standard deviation of Gaussian kernel (σ), the size of random filters (d), and the number of random filters (L). Among the three parameters, only L is relevant to the length of DBRF feature vectors. Because one of our goals is low dimensionality, the value of L is fixed at 8, such that the DBRF feature vector only has $3 \times 2^8 = 768$ dimensions. For the other two parameters, we empirically investigate $\sigma \in \{0.5, 0.7, 1, 1.5, 2, 2.2, 2.5\}$ (accordingly $k \in \{3, 5, 7, 9, 11, 13, 15\}$) and $d \in \{3, 5, 7, 9, 11, 13, 15\}$.

As a learning-free DT descriptor similar to VLBP, the representation power of DBRF mainly depends on the parameters and the characteristics of DT data. Some researchers may conduct grid search on various parameter settings for each dataset. In our case, there are 36 combinations. We think conducting grid search is time-consuming and has little guiding significance for related real-word applications. Instead, we decide to first analyze the characteristics of data used in each evaluation protocol. We focus on two properties, DT size and intra-class variation. According to DT size, the 50-class, 9-class, and 8-class breakdowns of UCLA dataset, and DynTex++ dataset contain small DTs while the four protocols on the DynTex dataset use large DTs. On the other hand, intra-class variation in UCLA 50-class breakdown is small while UCLA 9-class and 8-class breakdowns contains median intra-class variation due to the re-organization of data. The four subsets from the DynTex dataset have large intra-class variation due to various motions and complex backgrounds. As a recompilation of the DynTex dataset, the DynTex++ dataset contains a large number of DTs from various videos and has very large intra-class variation. This discussion is summarized in Table 1.

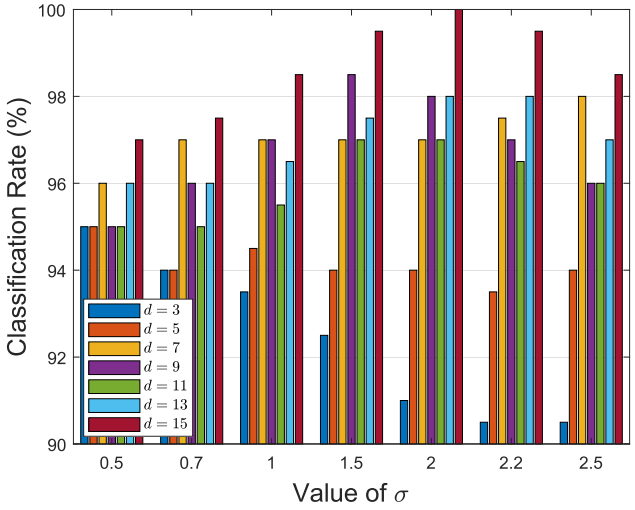

FIGURE 5. Performance comparison of various parameter combinations on UCLA 50-class 4CFV.

According to the above discussion, we decide to conduct grid search with UCLA 50-class 4CFV classification, UCLA 9-class half-to-half validation, Beta, and Dyntex++. To verify whether the knowledge obtained by grid search can be instructive for other similar tasks, we directly apply these found parameters in experiments conducted on UCLA 50-class LOO, UCLA 8-class half-to-half validation, and other three subsets from the DynTex dataset, respectively. The results of grid search on UCLA 50-class 4CFV and 9-class half-to-half validation are presented in Fig. 5 and Fig. 6, respectively.

In Fig. 5, it can be observed that increasing the value of σ would cause decrease of performance when $d \in \{3, 5\}$. For other choices of d, the classification rates generally first increase to peaks and then decrease when the value of σ increases. The best performance is obtained with $d = 15$ and $\sigma = 2$. Similar observations can be found in Fig. 6. When increasing the value of σ, $d \in \{3, 5, 13\}$ degrades performance while $d \in \{7, 9, 11, 15\}$ first increases classification rate to the highest point, after which performance decreases. And $d = 9$ and $\sigma = 2.2$ provide the highest classification rate (99%).

As for the grid search on Beta subset (shown in Fig. 7), the classification rates generally fluctuate as the value of σ increases, in spite of the choice of d. Additionally, two peaks could be observed. The lower one is around $\sigma = 0.7$ and the higher one is near $\sigma = 2$. If we focus on the value of d, $d = 13$ gives the best performance. When it comes to the grid search on DynTex++ (shown in Fig. 8), things are very straightforward, *i.e.*, increasing either the value σ or the value

TABLE 2. Comparison of processing time (in seconds) among DBRF, VLBP and LBP-TOP.

Method	$\text{DBRF}_{\sigma=0.5,d=3}$	$\text{DBRF}_{\sigma=2,d=13}$	$\text{DBRF}_{\sigma=2,d=15}$	$\text{DBRF}_{\sigma=2.2,d=9}$	VLBP	LBP-TOP
Processing Time	0.04	0.40	0.55	0.24	0.68	0.22

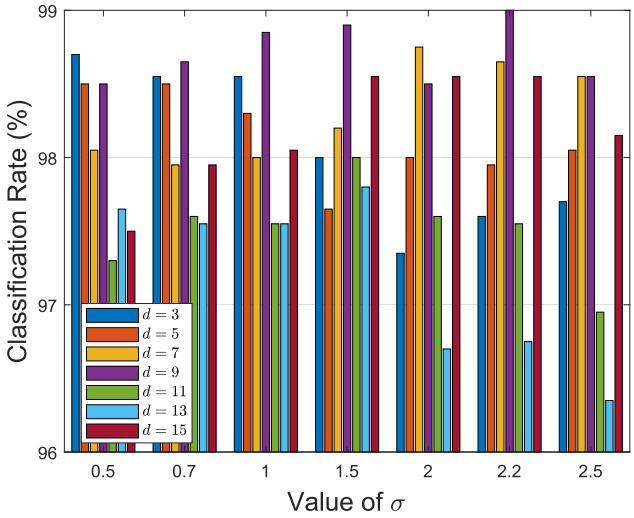

FIGURE 6. Performance comparison of various parameter combinations on UCLA 9-class.

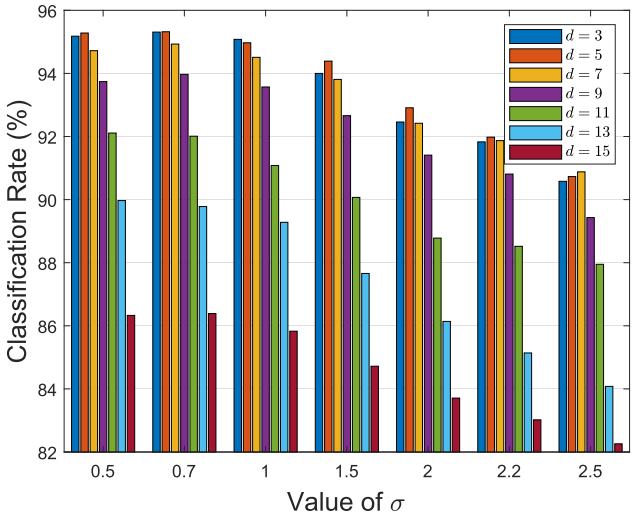

FIGURE 8. Performance comparison of various parameter combinations on DynTex++ dataset.

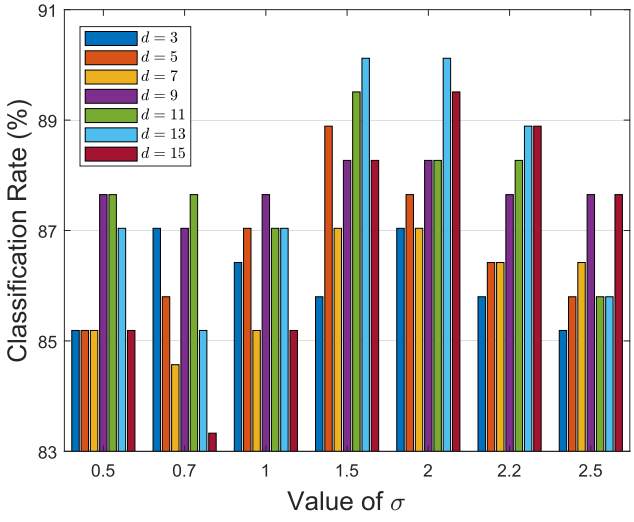

FIGURE 7. Performance comparison of various parameter combinations on Beta dataset.

of d would gradually decrease classification rates. Therefore, $d = 3$ and $\sigma = 0.5$ should be adopted.

According to the above experimental test and analysis, several suggestions could be made for the application of the proposed DBRF descriptor. If the data size is small, a relatively small d if its intra-class variation is large. Unless the intra-class variation is very large, a σ close to 2 is a good choice, otherwise a very small σ should be adopted. If the data size is large, both d and σ should be relatively large.

As for the following experiments and comparison with existing methods, we use $\sigma = 2$ and $d = 15$ for experiments on UCLA 50-class breakdown, $\sigma = 2.2$ and $d = 9$ for UCLA 9-class and 8-class breakdowns, $\sigma = 2$ and $d = 13$ for the four DynTex subsets, $\sigma = 0.5$ and $d = 3$ for DynTex++ dataset, respectively.

C. COMPUTATIONAL COMPLEXITY

It is commonly agreed that computational complexity and classification accuracy are of equal importance [17]. Therefore, we need to report how efficient the proposed DBRF is. Here we decide to record the elapsed time between feeding a DT to the feature extraction program and obtaining the corresponding feature vector. Considering that many researchers did not release their code and that the hardware setting can hardly be the same, we follow two works [5], [25] to evaluate the efficiency of our DBRF. Specifically, we compare DBRF with VLBP and LBP-TOP[2]($P = 8$, $R = 1$). We randomly choose a DT from the DynTex++ dataset and extract the corresponding LBP, LBP-TOP and DBRF features for ten times. Note that all the methods for comparison are implemented in native Matlab codes, which are executed in single-threading on a 64-bit Windows desktop with a Core i7-7700 3.6GHz CPU and 16G RAM.

The averaged processing times are summarized in Table 2. The computational efficiency of the proposed DBRF is related to four operations, which are Gaussian filtering (3 times), random projection (8 times), binary encoding and histogramming. Given a DT of a certain size, the processing time would be mainly affected by the sizes of Gaussian kernel and random filters because the time cost of binary encoding

[2]The Matlab implementations of VLBP and LBP-TOP is available at: https://github.com/I2Cvb/retinopathy/tree/master/src/matlab/STLBP_Matlab

TABLE 3. Performance comparison of the proposed DBRF with other methods on the three datasets under different evaluation protocols.

Method	Classification Rate(%)									Dimensionality
	UCLA				DynTex				DynTex++	
	50-class LOO	50-class 4CFV	9-class	8-class	DynTex35	Alpha	Beta	Gamma		
FD-MAP [30]	97.00	98.50	97.30	99.02	95.71	91.67	84.57	80.30	92.87	12,760
DDTP [31]	98.50	98.50	93.85	96.09	96.86	91.67	83.33	82.20	89.24	6768
AR-LDS [34]	89.50	-	-	-	-	-	-	-	-	-
LDS (k-PCA) [35]	-	96.00^{S}	-	-	-	-	-	-	-	-
KDT-MD [36]	-	89.50	-	-	-	-	-	-	-	-
BoS-HK [38]	-	-	-	70.00	-	-	-	-	-	-
BoS-KM [39]	-	-	78.00^{NB}	84.00^{NB}	-	-	-	-	-	96
LDS-SP [41]	-	92.50	-	-	-	-	-	-	-	-
DTM-HEM [38]	-	96.45	-	96.63	97.99	-	-	-	-	-
BoS Tree [40]	-	-	-	96.09	98.29	-	-	-	-	-
DT-complex [46]	-	95.00	-	-	-	-	-	-	-	-
DT-HMM [47]	-	-	-	94.00	-	-	-	-	-	-
Chaotic vector [42]	-	-	85.10	85.00	-	-	-	-	-	-
JVDL [43]	-	90.22	-	-	-	-	-	-	71.04	-
ASF-TOP [17]	-	-	-	-	97.14	91.67	86.42	89.39	95.40	-
DT-Diffusion [45]	-	98.50	97.80	96.22	-	-	-	-	93.80	-
DT-DPS [98]	-	96.00	96.80	96.59	-	-	-	-	94.60	-
3D-OTF [53]	-	99.25	96.32	95.80	96.70^{NCC}	-	-	-	89.17^{S}	290
WMFS [54]	-	99.12	96.95	97.18	-	-	-	-	-	702
DFS [52]	-	100^{S}	97.50^{S}	99.20^{S}	97.16^{NCC}	-	-	-	91.70^{S}	500
STLS [55]	-	99.50^{S}	97.40^{S}	99.50^{S}	98.20^{S}	-	-	-	94.50^{S}	1080
2D+T wavelet [56]	-	-	-	-	-	85.00^{NCC}	65.00^{NCC}	68.00^{NCC}	-	-
2D+T curvelet [57]	-	-	-	-	-	85.00^{NCC}	67.00^{NCC}	63.00^{NCC}	-	5508
MBSIF-TOP [16]	99.50	99.50	98.75	97.80	98.61	90.00	90.70	91.30	97.17	6144
DNG [60]	-	-	98.10	97.00	-	-	-	-	90.20	-
MPCAF-TOP [20]	99.50	99.50	99.15	98.26	-	-	-	-	96.52	3840
STRF N-jet [61]	-	100	99.00	99.10	-	100	93.80	91.20	-	-
FoSIG [62]	99.50^{S}	100^{S}	98.95^{S}	98.59^{S}	99.14^{S}	96.67^{S}	92.59^{S}	92.42^{S}	95.99^{S}	1200
V-BIG [63]	99.50^{S}	99.50^{S}	97.95^{S}	97.50^{S}	99.43^{S}	100^{S}	95.06^{S}	94.32^{S}	96.65^{S}	2400
$HoGF^{2D}$ [65]	100^{S}	100^{S}	99.20^{S}	98.91^{S}	99.71^{S}	100^{S}	97.53^{S}	96.59^{S}	97.19^{S}	7200
$HoGF^{3D}$ [65]	100^{S}	100^{S}	99.25^{S}	99.57^{S}	99.43^{S}	98.33^{S}	98.15^{S}	97.53^{S}	97.63^{S}	9600
$DoDGF^{2D}$ [64]	100^{S}	100^{S}	99.25^{S}	99.13^{S}	99.71^{S}	100^{S}	97.53^{S}	96.21^{S}	97.14^{S}	4800
$DoDGF^{3D}$ [64]	100^{S}	100^{S}	99.55^{S}	99.57^{S}	99.71^{S}	100^{S}	98.15^{S}	96.97^{S}	97.52^{S}	7200
VLBP [4]	-	89.50	96.30	91.96	94.00	-	-	-	83.75	16384
LBP-TOP [4]	-	94.50	96.00	94.34	91.43	86.67	80.86	81.44	89.50	768
DDLBP [66]	-	-	-	-	-	-	-	-	95.80^{S}	-
LTP-lac [67]	-	99.70^{S}	96.80^{S}	99.20^{S}	97.90^{S}	89.60^{S}	80.90^{S}	79.90^{S}	94.80^{S}	-
VSCR-VLBP [69]	-	83.21^{O}	-	-	-	-	-	-	-	-
VSCR-LBPTOP [69]	-	99.43^{O}	-	-	-	-	-	-	-	-
CVLBP [19]	-	93.00	96.90	95.65	85.14^{NCC}	-	-	-	-	512
novel LBP [18]	95.00	95.00	98.35	97.50	98.57	-	-	-	96.28	1536
CLSP-TOP [70]	99.00	99.00	98.30	97.06	97.71	95.00	90.12	89.39	93.73	1152
MEWLSP [78]	96.50	96.50	98.55	98.04	99.71	-	-	-	98.48	4608
WLBPC [77]	-	96.50	97.17	97.61	-	-	-	-	95.01	-
CVLBC [5]	98.50	99.00	99.20	99.02	98.86	-	-	-	91.31	11250
CSAP-TOP [72]	99.50	99.50	97.50	98.15	96.00	91.67	89.51	90.53	-	13200
HILOP [73]	99.50^{S}	99.50^{S}	97.80^{S}	96.30^{S}	99.71^{S}	96.67^{S}	91.36^{S}	92.05^{S}	96.21^{S}	5664
MEMDP [76]	100^{S}	100^{S}	98.90^{S}	98.70^{S}	99.71^{S}	96.67^{S}	96.91^{S}	93.94^{S}	96.03^{S}	3888
RUBIG [74]	100^{S}	100^{S}	99.20^{S}	99.13^{S}	98.86^{S}	100^{S}	95.68^{S}	93.56^{S}	97.08^{S}	21600
the proposed DBRF	99.50	100	99.00	99.67	100	95.00	90.12	87.50	95.18	768
DL-PEGASOS [26]	-	99.00	95.60	-	-	-	-	-	63.70	-
PI-LBP [15]	-	100	98.20	-	-	-	-	-	91.90	-
KGDL [92]	-	-	-	-	-	-	-	-	92.80^{S}	-
OTDL [93]	-	98.50	97.50	97.00	99.00	86.60^{NCC}	69.00^{NCC}	64.20^{NCC}	94.70^{S}	2700
EKDL [25]	-	-	98.60^{S}	-	-	-	-	-	93.40^{S}	-
SOE-Net [90]	-	-	-	-	97.70	98.30	96.90	93.60	94.40^{S}	1600
3DRF [23]	-	-	99.24	98.59	99.43	98.33	89.51	89.77	94.80	1000
B3DF_SMC [22]	99.50	99.50	98.85	98.15	99.71	95.00	90.12	90.91	95.58	65536
ICFV [24]	99.50	99.00	99.25	99.57	99.71	100	92.59	90.91	93.02	1600
DT-RNNs [91]	-	97.05	98.54	97.74	-	-	-	-	96.51	270/990
PCANet-TOP [21]	-	99.50	-	-	-	91.67	90.12	89.39	-	36864
High level features [87]	-	-	92.67^{S}	85.65^{S}	-	-	-	-	69.00^{S}	-
C3D [79]	-	-	-	-	-	100	99.38	96.97	-	-
st-TCoF [80]	-	-	-	-	-	98.33	98.15	98.11	-	8192
D3 [81]	-	-	-	-	-	100^{S}	100^{S}	98.11^{S}	-	-
DT-CNN-AlexNet [82]	-	99.50*	98.05*	98.48*	-	100*	99.38*	99.62*	98.18*	-
DT-CNN-GoogleNet [82]	-	99.50*	98.35*	99.02*	-	100*	100*	99.62*	98.58*	-
LBP-based DBN [88]	-	-	98.48*	-	-	-	-	-	-	-

Note: "-" means unavailable. Superscripts: "S" means SVM classifier; "NB" means naive Bayes classifier; "O" means optimization-based classifier; "NCC" means nearest class center classifier; "∗" means classification by deep learning techniques.

and histogramming is almost fixed. This analysis is in line with the results in Table 2. Because we conduct random projection for eight times, its time cost dominates the whole processing time. When $d < 13$, the DBRF is fast to compute. When $d >= 13$, it is still more efficient than VLBP.

D. COMPARATIVE EVALUATION

In this section, we evaluate the DBRF descriptor on the three benchmark datasets with various protocols (using the parameters determined in Section IV-B). Comparison with existing methods, especially local-feature-based ones, is also conducted. Note that the classification rates and feature lengths (if available) of existing descriptors are quoted directly from the literature. Performance comparison of the proposed DBRF with other methods on the three datasets with different evaluation protocols is summarized in Table 3. All the methods (including the proposed one) are grouped according to the typology used in Section II, with one exception: those deep-learning-based ones are separately put into a group due to their excellent performance.

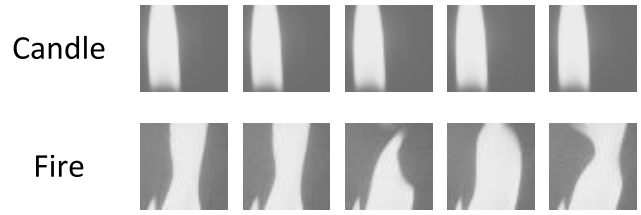

FIGURE 9. The first five frames of the mis-classified candle DT (top) and those of its nearest neighbor belonging to the fire class (bottom).

1) RESULTS ON THE UCLA DATASET

As shown in Table 3, most of the existing approaches are evaluated on the UCLA dataset (some ones may only use part of the four protocols). The proposed DBRF method is almost the best compared to other methods that are evaluated with the four standard protocols.

The proposed method achieves a rate of 99.50% on UCLA 50-class breakdown with LOO scheme. It is on par with MBSIF-TOP [16], MPCAF-TOP [20], FoSIG [63], V-BIG [64], CSAP-TOP [72], HILOP [73], B3DF_SMC [22], and ICFV [24]. Among these methods, MPCAF-TOP is a filter-learning-based multi-scale descriptor and ICFV involves two learning processes of filter learning and codebook construction, while our method is learning-free and thus has no dependence on data. FoSIG, V-BIG and HILOP are evaluated only with SVM classifier, which makes it difficult to distinguish the contribution of the descriptors themselves from that of the classifier. Both CSAP-TOP and B3dF_SMC are learning-free. However, their dimensionalities are very high. The former has 13200 features while the latter has 65536 ones. The feature lengths of MBSIF-TOP, MPCAF-TOP, FoSIG, V-BIG, ICFV and HILOP are 6144, 3840, 1200, 2400, 1600, and 5664, respectively. The proposed DBRF has only 768 dimensions, which is substantially less than other methods. On the other hand, $HoGF^{2D}$ [25], $HoGF^{3D}$ [25], $DoDGF^{2D}$ [65], $DoDGF^{3D}$ [65], MEMDP [76], and RUBIG [74] slightly outperform DBRF by 0.5% and all of them adopt SVM for DT classification. Due to the fact that all of them use features from multiple scales, they generate high-dimensional feature vectors (7200, 9600, 4800, 7200 3888, 21600, respectively), which are at least 4 times longer than ours. Additionally, we also investigate which DT is mis-classified by our DBRF. We find out a candle DT is classified into the fire class. The first five frames of the candle DT and its nearest neighbor in training DTs are illustrated in Fig. 9. It is obvious that both DTs contain a flickered flame and are semantically belonging to the same class.

When using the 4CFV scheme, our DBRF correctly classifies all test samples, *i.e.*, the rate is 100%. The rates provided by DFS [53], STRF N-jet [62], FoSIG [63], $HoGF^{2D}$ [25], $HoGF^{3D}$ [25], $DoDGF^{2D}$ [65], MEMDP [76], RUBIG [74], and PI-LBP [15] are also 100%. Except for STRF N-jet and PI-LBP, all the other methods use SVM classifier. PI-LBP applied PCA twice, which was followed by a discriminant

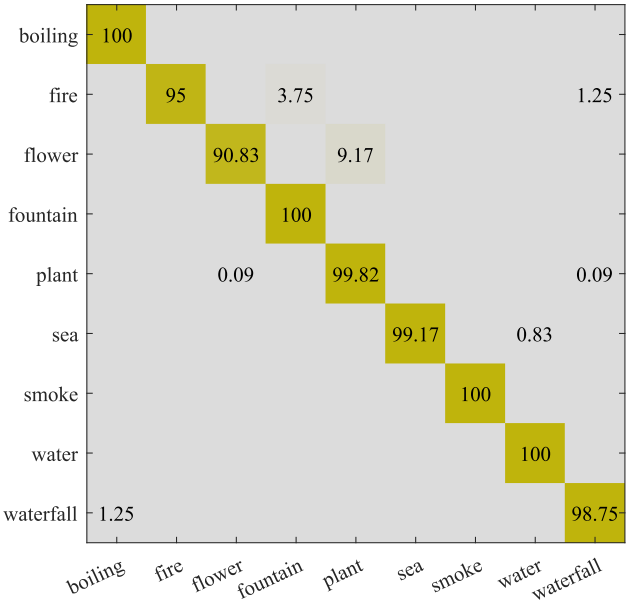

FIGURE 10. Confusion matrix for DBRF on the UCLA 9-class breakdown (the numbers are in percentage).

analysis. We think it is more complex than our DBRF though its feature dimensionality is unreported. STRF N-jet has two stages of multi-scale filtering which is followed by PCA for dimensionality reduction. However, it still has 16348 dimensions (computed according to its parameter setting). Therefore, the proposed DBRF still has some superiority over the compared ones due to its low dimensionality and simplicity.

When evaluated on the 9-class breakdown, the proposed DBRF achieves a rate of 99%, which is on par with STRF N-jet [62]. MPCAF-TOP [20], $HoGF^{2D}$ [25], $HoGF^{3D}$ [25], $DoDGF^{2D}$ [65], MEMDP [76], CVLBC [5], RUBIG [74], 3DRF [23], and ICFV [24] marginally outperform our DBRF by 0.15%, 0.2%, 0.25%, 0.25%, 0.55%, 0.2%, 0.2%, 0.24%, and 0.25%, respectively. All of them have much higher dimensionalities and most of the classification rates are obtained with SVM classifier. Fig. 10 is the confusion matrix for DBRF under this protocol. The ''fire-fountain'' and ''flower-plant'' mis-classifications should be addressed carefully in future because some DTs of the two classes are very similar to each other.

On the 8-class breakdown, 99.67% by DBRF is higher than the rates by all the other methods. The advantages of our DBRF are fully revealed when the dominant plant class is discarded. As a learning-free method that has only 768 dimensions, the proposed DBRF outperforms those high-dimensional descriptors that use SVM classifier, and even two deep learning methods. The confusion matrix is shown in Fig. 11. The main confusion is that some DTs of the fire class are classified into the fountain and waterfall classes. We assume the reason could be that the motion of a flickered flame is a little similar to that of moving waters.

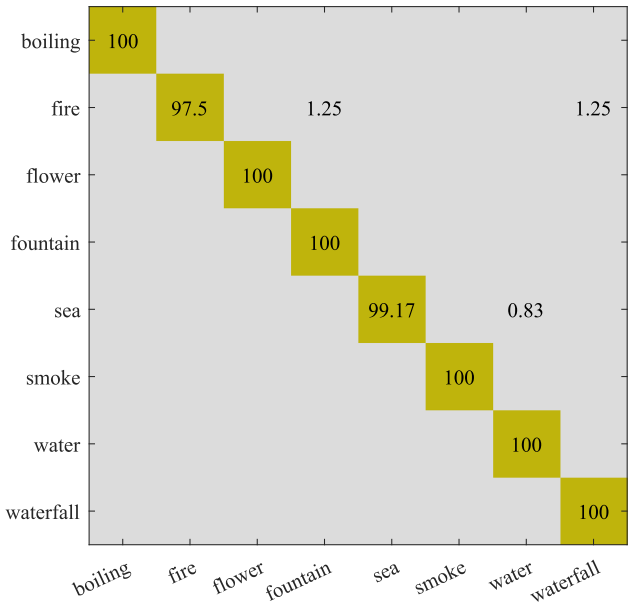

FIGURE 11. Confusion matrix for DBRF on the UCLA 8-class breakdown (the numbers are in percentage).

FIGURE 12. First frames of the three mis-classified DTs (top) and those of their corresponding nearest neighbors in training data (bottom).

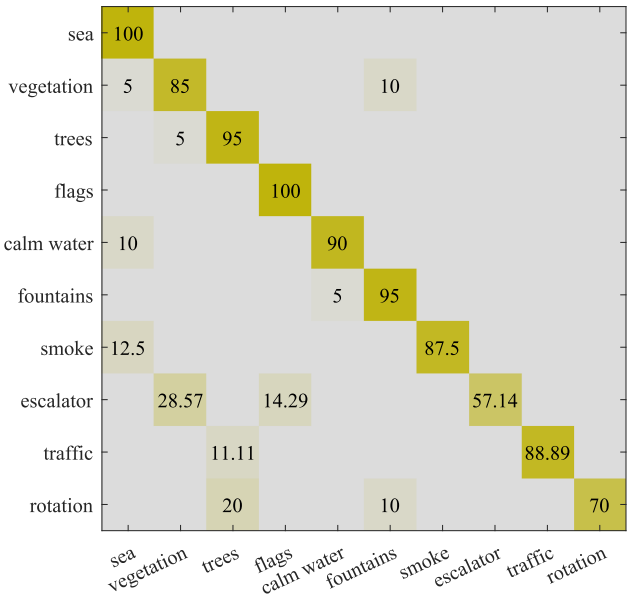

FIGURE 13. Confusion matrix for DBRF on the Beta subset (the numbers are in percentage).

2) RESULTS ON THE DynTex DATASET

The proposed DBRF achieves a rate of 100% on the DynTex35 subset and outperforms all the other methods, especially those recently published ones that use SVM classifier and high-dimensional feature vectors, such as HoGF^{2D} [25], HoGF^{3D} [25], DoDGF^{2D} [65], MEMDP [76], and RUBIG [74].

On the other three subset, the performance of our DBRF is not as good as that on the DynTex35 subset. We can only say acceptable results are obtained, considering the complexity of data in Alpha, Beta, and Gamma subsets.

On the Alpha subset, the proposed DBRF provides a rate of 95%. It outperforms FD-MAP [31], DDTP [32], ASF-TOP [17], CSAP-TOP [72], MBSIF-TOP [16], LBP-TOP [4], and LTP-lac [67] by at least 3.33%. Some of them are recently published and use SVM classifier. Two methods of CLSP-TOP [70] and B3DF_SMC [22] are on par with our DBRF while they use longer feature vectors. STRF N-jet [62], HoGF^{2D} [25], HoGF^{3D} [25], DoDGF^{2D} [65], DoDGF^{3D} [65], RUBIG [74], SOE-Net [90], 3DRF [23], and ICFV [24] significantly outperform the proposed DBRF by a rate of 3.3% to 5%. As shown Fig. 12, three DTs from the grass class are mis-classified into the trees class. The waving grasses seem to have similar motion patterns as the swaying tree branches with no leaf, which may be the reason for the mis-classifications.

The Beta subset is built by adding extra DTs to the Alpha subset and hence it is more challenging than the Alpha subset. The situation is similar to that on the Alpha subset. Our DBRF achieves a rate of 90.12%, which is slightly higher than those of FD-MAP [31], DDTP [32], ASF-TOP [17], CSAP-TOP [72], MBSIF-TOP [16], LBP-TOP [4], LTP-lac [67], CLSP-TOP [70], and 3DRF [23]. And it is outperformed by FoSIG [63], V-BIG [64], HoGF^{2D} [25], HoGF^{3D} [25], DoDGF^{2D} [65], DoDGF^{3D} [65], MEMDP [76], RUBIG [74] SOE-Net [90], and ICFV [24] by at least 2%. Both the motion and appearance patterns in this subset are very complex, causing that at least 10% of the DTs from the six classes of vegetation, calm water, smoke, escalator, traffic and rotation are mis-classified. Fig. 13 illustrates the confusion matrix.

The Gamma subset is built by adding extra DTs to the Beta set and hence it is more challenging than the Beta subset. The rate by DBRF decreases to 87.5% on the Gamma subset and only outperforms FD-MAP [31], DDTP [32], LBP-TOP [4], and LTP-lac [67], which indicates that the advantages of our DBRF are only low dimensionality and simplicity. According to the confusion matrix shown in Fig. 14, nearly half of the escalator DTs are mis-classified.

According the results on the Alpha, Beta, and Gamma subsets, an interesting point can be observed: among those methods that significantly outperform the proposed DBRF, most

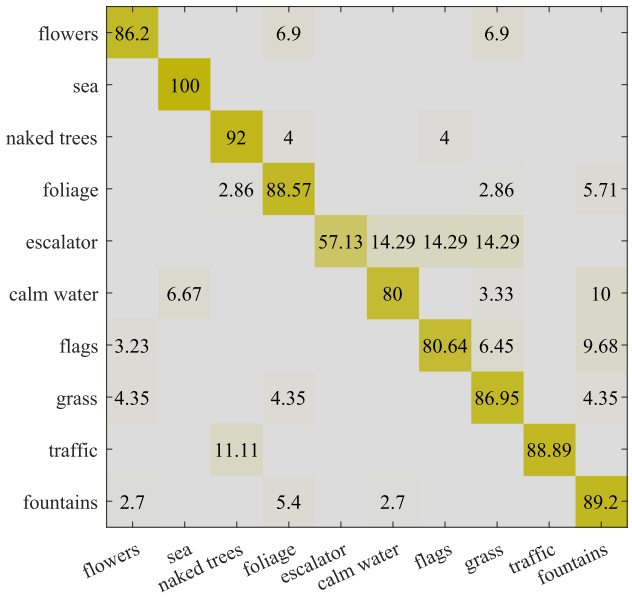

FIGURE 14. Confusion matrix for DBRF on the Gamma subset (the numbers are in percentage).

learning-free methods (*e.g.*, HoGF^{2D} [25], HoGF^{3D} [25], DoDGF^{2D} [65], DoDGF^{3D} [65], RUBIG [74]) use a lot many features (7200, 9600, 4800, 7200, and 21600, respectively) for DT classification while the learning-based ICFV only use 1600 features. It seems that involving a learning process is very beneficial when processing large DTs with complex variations, where relatively good performance and low dimensionality can be obtained simultaneously. One exception is the learning-free SOE-Net [90]. However, good performance is obtained due to the utilization of a deep learning technique, *i.e.*, cascaded convolution and pooling.

3) RESULTS ON THE DynTex++ DATASET

A relatively good result of 95.18% is provided by our DBRF on this dataset. It is slightly outperformed by ASF-TOP [17], MBSIF-TOP [16], MPCAF-TOP [20], FoSIG [63], V-BIG [64], HoGF^{2D} [25], HoGF^{3D} [25], DoDGF^{2D} [65], DoDGF^{3D} [65], DDLBP [66], novel LBP [18], MEWLSP [78], HILOP [73], MEMDP [76], RUBIG [74], B3DF_SMC [22], ICFV [24], and DT-RNNs [91] by 0.22%, 1.99%, 1.34%, 0.81%, 1.47%, 2.01%, 2.45%, 1.96%, 2.34%, 0.62%, 1.1%, 3.3%, 1.03%, 0.85%, 1.9%, 0.4%, and 1.33%, respectively. Despite the fact that many of these methods use SVM classifier, we think it is unworthy to significantly increase complexity and feature dimensionality for marginal performance improvement. Fig. 15 depicts the class-specific classification rates. The rates (labeled in red) on four classes are below 90%, showing the challenge for classification on this dataset.

4) COMPARISON WITH DEEP-LEARNING METHODS

Deep learning technique has been applied for various computer vision tasks and its essence is to learn features

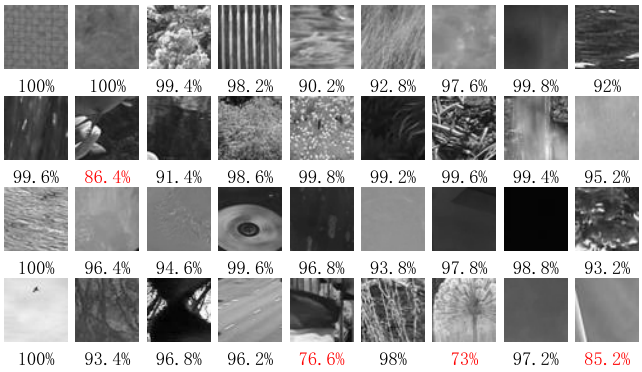

FIGURE 15. Class-specific classification rates obtained by the proposed DBRF (very challenging classes are labeled in red).

from data. One prerequisite is that sufficient data is needed to trained a real deep model. AlexNet [83], GoogleNet [84], VGGNet [85], and C3D [79] are trained on datasets of millions of images or videos and provide excellent classification performance, to which the contribution belongs to both their well-designed structures and large training datasets. Hence, the pre-trained C3D [79] is directly used as DT feature extractor. D3 [81] and st-TCoF [80] use the pre-trained VGGNet [85] as first-stage feature extractor to extract features from image slices in a DT, which is followed by the second-stage feature aggregation. Very excellent results (around 99%) by deep-learning methods are reported on the challenging Alpha, Beta, and Gamma subsets, substantially outperforming all shallow methods. On the other hand, DT-CNN-AlexNet [82] and DT-CNN-GoogleNet [82] only use the structures of AlexNet and GoogleNet, and are trained from scratch with DT data. As there are tens of thousands of images in each of the Alpha, Beta, and Gamma subsets, DT-CNN-AlexNet and DT-CNN-GoogleNet could be well trained and near-perfect results are obtained. However, they are outperformed by many shallow methods such as the propoased DBRF and DoDGF, when evaluated on the UCLA dataset. We think the reason behind this observation is that UCLA dataset contains limited number of small-scale data.

Although the deep-learning methods are indeed vary powerful (especially those trained on large datasets), their application in real-world scenarios are limited in three aspects: (1) sufficient data are needed, (2) either training or running requires large amount of computing resources, (3) they can hardly process data of various sizes. Therefore, those shallow methods, including the proposed one, do have the meaning of existence as well as application scenarios.

V. CONCLUSION

In this paper, we have proposed a simple but effective method for DT description. Its effectiveness and efficiency have been verified by experimental evaluations with various protocols. In the processing framework, we first extract 3D Gaussian gradients from DTs. Then eight random projections are applied to the gradients such that a set of low-dimensional

random feature vectors are obtained. Each feature vector is encoded into a binary code. Finally, the binary codes are counted to form three histograms, which are concatenated into one histogram for DT description. All of our design goals are achieved. The proposed method is a learning-free and 3D-filtering-based DT description method. It is very fast to compute when the filter size is smaller than 13 and the feature length is 768, which is smaller than that of most existing methods. The contribution of using DT gradients is also verified by comparison with non-gradient-based methods such as 3DRF and CSAP-TOP. Due to the above valuable properties of the proposed method, our DBRF can be directly applied and is very suitable for resource-restricted scenarios. Future efforts should seek performance enhancement on vary challenging DT datasets.

REFERENCES

[1] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, "Dynamic textures," *Int. J. Comput. Vis.*, vol. 51, no. 2, pp. 91–109, 2003.

[2] X. S. Nguyen, T. P. Nguyen, F. Charpillet, and N.-S. Vu, "Local derivative pattern for action recognition in depth images," *Multimedia Tools Appl.*, vol. 77, no. 7, pp. 8531–8549, Apr. 2018.

[3] K. J. Cannons, J. M. Gryn, and R. P. Wildes, "Visual tracking using a pixelwise spatiotemporal oriented energy representation," in *Proc. 11th Eur. Conf. Comput. Vis. (ECCV)*. Berlin, Germany: Springer, 2010, pp. 511–524.

[4] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 915–928, Jun. 2007.

[5] X. Zhao, Y. Lin, and J. Heikkilä, "Dynamic texture recognition using volume local binary count patterns with an application to 2D face spoofing detection," *IEEE Trans. Multimedia*, vol. 20, no. 3, pp. 552–566, Mar. 2018.

[6] X. Li, X. Hong, A. Moilanen, X. Huang, T. Pfister, G. Zhao, and M. Pietikäinen, "Towards reading hidden emotions: A comparative study of spontaneous micro-expression spotting and recognition methods," *IEEE Trans. Affect. Comput.*, vol. 9, no. 4, pp. 563–577, Oct. 2018.

[7] G. Zhao, M. Barnard, and M. Pietikainen, "Lipreading with local spatiotemporal descriptors," *IEEE Trans. Multimedia*, vol. 11, no. 7, pp. 1254–1265, Nov. 2009.

[8] B. U. Töreyin, Y. Dedeoğlu, U. Güdükbay, and A. E. Cetin, "Computer vision based method for real-time fire and flame detection," *Pattern Recognit. Lett.*, vol. 27, no. 1, pp. 49–58, Jan. 2006.

[9] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–7.

[10] A. B. Chan and N. Vasconcelos, "Classification and retrieval of traffic video using auto-regressive stochastic processes," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2005, pp. 771–776.

[11] T. T. Nguyen and T. P. Nguyen, "A comprehensive taxonomy of dynamic texture representation," *ACM Comput. Surv.*, vol. 55, no. 1, pp. 1–39, Jan. 2023.

[12] G. Zhao and M. Pietikäinen, "Dynamic texture recognition using volume local binary patterns," in *Proc. Int. Worksop Dyn. Vis.*, Beijing, China, Graz, Austria: Springer, 2007, pp. 165–177.

[13] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.

[14] E. Rahtu, J. Heikkilä, V. Ojansivu, and T. Ahonen, "Local phase quantization for blur-insensitive image analysis," *Image Vis. Comput.*, vol. 30, no. 8, pp. 501–512, Aug. 2012.

[15] J. Ren, X. Jiang, and J. Yuan, "Dynamic texture recognition using enhanced LBP features," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 2400–2404.

[16] S. R. Arashloo and J. Kittler, "Dynamic texture recognition using multiscale binarized statistical image features," *IEEE Trans. Multimedia*, vol. 16, no. 8, pp. 2099–2109, Dec. 2014.

[17] S. Hong, J. Ryu, and H. S. Yang, "Not all frames are equal: Aggregating salient features for dynamic texture classification," *Multidimensional Syst. Signal Process.*, vol. 29, pp. 1–20, Nov. 2016.

[18] D. Tiwari and V. Tyagi, "A novel scheme based on local binary pattern for dynamic texture recognition," *Comput. Vis. Image Understand.*, vol. 150, pp. 58–65, Sep. 2016.

[19] D. Tiwari and V. Tyagi, "Dynamic texture recognition based on completed volume local binary pattern," *Multidimensional Syst. Signal Process.*, vol. 27, no. 2, pp. 563–575, Apr. 2016.

[20] X. Zhao, Y. Lin, and J. Heikkilä, "Dynamic texture recognition using multiscale PCA-learned filters," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 4152–4156.

[21] S. R. Arashloo, M. C. Amirani, and A. Noroozi, "Dynamic texture representation using a deep multi-scale convolutional network," *J. Vis. Commun. Image Represent.*, vol. 43, pp. 89–97, Feb. 2017.

[22] X. Zhao, Y. Lin, L. Liu, J. Heikkilä, and W. Zheng, "Dynamic texture classification using unsupervised 3D filter learning and local binary encoding," *IEEE Trans. Multimedia*, vol. 21, no. 7, pp. 1694–1708, Jul. 2019.

[23] X. Zhao, Y. Lin, and L. Liu, "Dynamic texture recognition using 3D random features," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 2102–2106.

[24] Z. Xiong, F. Mo, X. Zhao, F. Xu, X. Zhang, and Y. Wu, "Dynamic texture classification based on 3D ICA-learned filters and Fisher vector encoding in big data environment," *J. Signal Process. Syst.*, vol. 94, no. 11, pp. 1129–1143, Nov. 2022.

[25] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Prominent local representation for dynamic textures based on high-order Gaussian-gradients," *IEEE Trans. Multimedia*, vol. 23, pp. 1367–1382, 2021.

[26] Y. Quan, C. Bao, and H. Ji, "Equiangular kernel dictionary learning with applications to dynamic texture analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 308–316.

[27] B. Ghanem and N. Ahuja, "Maximum margin distance learning for dynamic texture recognition," in *Proc. 11th Eur. Conf. Comput. Vis. (ECCV)*. Berlin, Germany: Springer, 2010, pp. 223–236.

[28] C.-H. Peh and L.-F. Cheong, "Synergizing spatial and temporal texture," *IEEE Trans. Image Process.*, vol. 11, no. 10, pp. 1179–1191, Oct. 2002.

[29] R. Péteri and D. Chetverikov, "Qualitative characterization of dynamic textures for video retrieval," in *Proc. Int. Conf. Comput. Vis. Graph. (ICCVG)*, Warsaw, Poland. Dordrecht, The Netherlands: Springer, 2006, pp. 33–38.

[30] Z. Lu, W. Xie, J. Pei, and J. Huang, "Dynamic texture recognition by spatio-temporal multiresolution histograms," in *Proc. 7th IEEE Workshops Appl. Comput. Vis. (WACV/MOTION)*, vol. 2, Jan. 2005, pp. 241–246.

[31] T. T. Nguyen, T. P. Nguyen, F. Bouchara, and X. S. Nguyen, "Directional beams of dense trajectories for dynamic texture recognition," in *Proc. 19th Int. Conf. Adv. Concepts Intell. Vis. Syst. (ACIVS)*, Poitiers, France. Cham, Switzerland: Springer, Sep. 2018, pp. 74–86.

[32] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Directional dense-trajectory-based patterns for dynamic texture recognition," *IET Comput. Vis.*, vol. 14, no. 4, pp. 162–176, Jun. 2020.

[33] L. N. Couto and C. A. Barcelos, "Singular patterns in optical flows as dynamic texture descriptors," in *Proc. 23rd Iberoamerican Congr. (CIARP)*, Madrid, Spain. Cham, Switzerland: Springer, 2019, pp. 351–358.

[34] W. Liu and E. Ribeiro, "Detecting singular patterns in 2D vector fields using weighted Laurent polynomial," *Pattern Recognit.*, vol. 45, no. 11, pp. 3912–3925, Nov. 2012.

[35] P. Saisan, G. Doretto, Y. N. Wu, and S. Soatto, "Dynamic texture recognition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Dec. 2001, p. 2.

[36] A. B. Chan and N. Vasconcelos, "Probabilistic kernels for the classification of auto-regressive visual processes," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, Jun. 2005, pp. 846–851.

[37] A. B. Chan and N. Vasconcelos, "Classifying video with kernel dynamic textures," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–6.

[38] A. Mumtaz, E. Coviello, G. R. G. Lanckriet, and A. B. Chan, "Clustering dynamic textures with the hierarchical EM algorithm for modeling video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 7, pp. 1606–1621, Jul. 2013.

[39] A. Ravichandran, R. Chaudhry, and R. Vidal, "View-invariant dynamic texture recognition using a bag of dynamical systems," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1651–1657.

[40] A. Ravichandran, R. Chaudhry, and R. Vidal, "Categorizing dynamic textures using a bag of dynamical systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 2, pp. 342–353, Feb. 2013.

[41] A. Mumtaz, E. Coviello, G. R. G. Lanckriet, and A. B. Chan, "A scalable and accurate descriptor for dynamic textures using bag of system trees," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 4, pp. 697–712, Apr. 2015.

[42] B. Ghanem and N. Ahuja, "Sparse coding of linear dynamical systems with an application to dynamic texture recognition," in *Proc. 20th Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 987–990.

[43] Y. Wang and S. Hu, "Chaotic features for dynamic textures recognition," *Soft Comput.*, vol. 20, no. 5, pp. 1977–1989, May 2016.

[44] X. Wei, Y. Li, H. Shen, F. Chen, M. Kleinsteuber, and Z. Wang, "Dynamical textures modeling via joint video dictionary learning," *IEEE Trans. Image Process.*, vol. 26, no. 6, pp. 2929–2943, Jun. 2017.

[45] L. Wang, H. Liu, and F. Sun, "Dynamic texture video classification using extreme learning machine," *Neurocomputing*, vol. 174, pp. 278–285, Jan. 2016.

[46] L. C. Ribas, W. N. Gonçalves, and O. M. Bruno, "Dynamic texture analysis with diffusion in networks," *Digit. Signal Process.*, vol. 92, pp. 109–126, Sep. 2019.

[47] W. N. Gonçalves, B. B. Machado, and O. M. Bruno, "A complex network approach for dynamic texture recognition," *Neurocomputing*, vol. 153, pp. 211–220, Apr. 2015.

[48] Y. Qiao and L. Weng, "Hidden Markov model based dynamic texture classification," *IEEE Signal Process. Lett.*, vol. 22, no. 4, pp. 509–512, Apr. 2015.

[49] Y.-L. Qiao and Z.-Y. Xing, "Dynamic texture classification using multivariate hidden Markov model," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E101.A, no. 1, pp. 302–305, 2018.

[50] Q. Liu, X. Men, Y. Qiao, B. Liu, J. Liu, and Q. Liu, "Dynamic texture classification with relative phase information in the complex wavelet domain," in *Proc. 12th Int. Conf. Genetic Evol. Comput.*, Changzhou, China. Singapore: Springer, Dec. 2019, pp. 641–651.

[51] B. Mandelbrot, *The Fractal Geometry of Nature*. New York, NY, USA: Freeman, 1982.

[52] Y. Xu, Y. Quan, H. Ling, and H. Ji, "Dynamic texture classification using dynamic fractal analysis," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1219–1226.

[53] Y. Xu, Y. Quan, Z. Zhang, H. Ling, and H. Ji, "Classifying dynamic textures via spatiotemporal fractal analysis," *Pattern Recognit.*, vol. 48, no. 10, pp. 3239–3248, Oct. 2015.

[54] Y. Xu, S. Huang, H. Ji, and C. Fermüller, "Scale-space texture description on sift-like textons," *Comput. Vis. Image Understand.*, vol. 116, no. 9, pp. 999–1013, 2012.

[55] H. Ji, X. Yang, H. Ling, and Y. Xu, "Wavelet domain multifractal analysis for static and dynamic texture classification," *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 286–299, Jan. 2013.

[56] Y. Quan, Y. Sun, and Y. Xu, "Spatiotemporal lacunarity spectrum for dynamic texture classification," *Comput. Vis. Image Understand.*, vol. 165, pp. 85–96, Dec. 2017.

[57] S. Dubois, R. Péteri, and M. Ménard, "A comparison of wavelet based spatio-temporal decomposition methods for dynamic texture recognition," in *Proc. 4th Iberian Conf. Pattern Recognit. Image Anal. (IbPRIA)*. Póvoa de Varzim, Portugal. Berlin, Germany: Springer, Jun. 2009, pp. 314–321.

[58] S. Dubois, R. Péteri, and M. Ménard, "Characterization and recognition of dynamic textures based on the 2D+T curvelet transform," *Signal, Image Video Process.*, vol. 9, no. 4, pp. 819–830, May 2015.

[59] M. Baktashmotlagh, M. Harandi, B. C. Lovell, and M. Salzmann, "Discriminative non-linear stationary subspace analysis for video classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 12, pp. 2353–2366, Dec. 2014.

[60] K. G. P. Derpanis and R. Wildes, "Spacetime texture representation and recognition based on a spatiotemporal orientation analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 6, pp. 1193–1205, Jun. 2012.

[61] A. R. Rivera and O. Chae, "Spatiotemporal directional number transitional graph for dynamic texture recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 10, pp. 2146–2152, Oct. 2015.

[62] Y. Jansson and T. Lindeberg, "Dynamic texture recognition using time-causal and time-recursive spatio-temporal receptive fields," *J. Math. Imag. Vis.*, vol. 60, no. 9, pp. 1369–1398, Nov. 2018.

[63] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Smooth-invariant Gaussian features for dynamic texture recognition," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 4400–4404.

[64] T. T. Nguyen, T. P. Nguyen, F. Bouchara, and N.-S. Vu, "Volumes of blurred-invariant Gaussians for dynamic texture classification," in *Proc. 18th Int. Conf. Comput. Anal. Images Patterns (CAIP)*, Salerno, Italy. Cham, Switzerland: Springer, Sep. 2019, pp. 155–167.

[65] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "A novel filtering kernel based on difference of derivative Gaussians with applications to dynamic texture representation," *Signal Process., Image Commun.*, vol. 98, Oct. 2021, Art. no. 116394.

[66] J. Ren, X. Jiang, J. Yuan, and G. Wang, "Optimizing LBP structure for visual recognition using binary quadratic programming," *IEEE Signal Process. Lett.*, vol. 21, no. 11, pp. 1346–1350, Nov. 2014.

[67] Y. Sun, Y. Xu, and Y. Quan, "Characterizing dynamic textures with space-time lacunarity analysis," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jun. 2015, pp. 1–6.

[68] X. Tan and B. Triggs, "Enhanced local texture feature sets for face recognition under difficult lighting conditions," *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1635–1650, Jun. 2010.

[69] J. Xie and Y. Fang, "Dynamic texture recognition with video set based collaborative representation," *Image Vis. Comput.*, vol. 55, pp. 86–92, Nov. 2016.

[70] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Completed local structure patterns on three orthogonal planes for dynamic texture recognition," in *Proc. 7th Int. Conf. Image Process. Theory, Tools Appl. (IPTA)*, Nov. 2017, pp. 1–6.

[71] N. Shrivastava and V. Tyagi, "An effective scheme for image texture classification based on binary local structure pattern," *Vis. Comput.*, vol. 30, no. 11, pp. 1223–1232, Nov. 2014.

[72] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Completed statistical adaptive patterns on three orthogonal planes for recognition of dynamic textures and scenes," *J. Electron. Imag.*, vol. 27, no. 5, 2018, Art. no. 053044.

[73] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Dynamic texture representation based on hierarchical local patterns," in *Proc. 20th Int. Conf. Adv. Concepts Intell. Vis. Syst. (ACIVS)*, Auckland, New Zealand. Cham, Switzerland: Springer, Feb. 2020, pp. 277–289.

[74] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Rubik Gaussian-based patterns for dynamic texture classification," *Pattern Recognit. Lett.*, vol. 135, pp. 180–187, Jul. 2020.

[75] B. Zhang, Y. Gao, S. Zhao, and J. Liu, "Local derivative pattern versus local binary pattern: Face recognition with high-order local pattern descriptor," *IEEE Trans. Image Process.*, vol. 19, no. 2, pp. 533–544, Feb. 2010.

[76] T. T. Nguyen, T. P. Nguyen, F. Bouchara, and X. S. Nguyen, "Momental directional patterns for dynamic texture recognition," *Comput. Vis. Image Understand.*, vol. 194, May 2020, Art. no. 102882.

[77] D. Tiwari and V. Tyagi, "Improved Weber's law based local binary pattern for dynamic texture recognition," *Multimedia Tools Appl.*, vol. 76, no. 5, pp. 6623–6640, Mar. 2017.

[78] D. Tiwari and V. Tyagi, "Dynamic texture recognition using multiresolution edge-weighted local structure pattern," *Comput. Electr. Eng.*, vol. 62, pp. 485–498, Aug. 2017.

[79] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4489–4497.

[80] X. Qi, C.-G. Li, G. Zhao, X. Hong, and M. Pietikäinen, "Dynamic texture and scene classification by transferring deep image features," *Neurocomputing*, vol. 171, pp. 1230–1241, Jan. 2016.

[81] S. Hong, J. Ryu, W. Im, and H. S. Yang, "D3: Recognizing dynamic scenes with deep dual descriptor based on key frames and key segments," *Neurocomputing*, vol. 273, pp. 611–621, Jan. 2018.

[82] V. Andrearczyk and P. F. Whelan, "Convolutional neural network on three orthogonal planes for dynamic texture classification," *Pattern Recognit.*, vol. 76, pp. 36–49, Apr. 2018.

[83] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1106–1114.

[84] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[85] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015, pp. 1–14.
[86] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the Fisher kernel for large-scale image classification," in *Proc. 11th Eur. Conf. Comput. Vis. (ECCV)*, Heraklion, Greece. Berlin, Germany: Springer, Sep. 2010, pp. 143–156.
[87] Y. Wang and S. Hu, "Exploiting high level feature for dynamic textures recognition," *Neurocomputing*, vol. 154, pp. 217–224, Apr. 2015.
[88] N. Zrira, K. Mouhcine, I. Benmiloud, and E. H. Bouyakhf, "Dynamic texture-based scene classification using deep belief networks," in *Proc. Int. Conf. Learn. Optim. Algorithms, Theory Appl.*, May 2018, pp. 1–6.
[89] M. Koleini, M. R. Ahmadzadeh, and S. Sadri, "A new efficient feature-combination-based method for dynamic texture modeling and classification using semi-random starting parameter dynamic Bayesian networks," *Multimedia Tools Appl.*, vol. 76, no. 14, pp. 15251–15278, Jul. 2017.
[90] I. Hadji and R. P. Wildes, "A spatiotemporal oriented energy network for dynamic texture recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3066–3074.
[91] J. J. D. M. Sá Junior, L. C. Ribas, and O. M. Bruno, "Randomized neural network based signature for dynamic texture classification," *Expert Syst. Appl.*, vol. 135, pp. 194–200, Nov. 2019.
[92] M. Harandi, C. Sanderson, C. Shen, and B. Lovell, "Dictionary learning and sparse coding on Grassmann manifolds: An extrinsic solution," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 3120–3127.
[93] Y. Quan, Y. Huang, and H. Ji, "Dynamic texture recognition via orthogonal tensor dictionary learning," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 73–81.
[94] E. J. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?" *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5406–5425, Dec. 2006.
[95] X. Zhao, Y. Lin, B. Ou, J. Yang, and Z. Wu, "Face recognition using local gradient binary count pattern," *J. Electron. Imag.*, vol. 24, no. 6, Nov. 2015, Art. no. 063003.
[96] L. Liu and P. Fieguth, "Texture classification from random features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 3, pp. 574–586, Mar. 2012.
[97] R. Péteri, S. Fazekas, and M. J. Huiskes, "DynTex: A comprehensive database of dynamic textures," *Pattern Recognit. Lett.*, vol. 31, no. 12, pp. 1627–1632, Sep. 2010.
[98] L. C. Ribas and O. M. Bruno, "Dynamic texture classification using deterministic partially self-avoiding walks on networks," in *Proc. 20th Int. Conf. Image Anal. Process. (ICIAP)*, Trento, Italy. Cham, Switzerland: Springer, Sep. 2019, pp. 82–93.

XIAOCHAO ZHAO received the B.S. and Ph.D. degrees in software engineering from Hunan University, Changsha, China, in 2012 and 2018, respectively. He has been a Lecturer with Hubei Engineering University, since 2019. His research interests include image processing and computer vision.

FANG XU received the M.S. and Ph.D. degrees from Wuhan University, Wuhan, China, in 2009 and 2016, respectively. He is currently a Professor with the School of Computer and Information Science, Hubei Engineering University, Hubei, China. His research interests include artificial intelligence, computer vision, and wireless mobile networks.

YI MA received the Ph.D. degree in photogrammetry and remote sensing from Wuhan University, Wuhan, China, in 2021. He has been a Lecturer with Hubei Engineering University, since 2021. His research interests include remote sensing image processing, quantitative inversion, and the modeling of vegetation parameters.

ZHEN LIU received the Ph.D. degree in control science and engineering from Jiangnan University. In 2022, he joined the School of Computer and Information Science, Hubei Engineering University. His research interests include machine learning, artificial intelligence, and pattern recognition.

MIN DENG received the M.S. degree from Wuhan University, Wuhan, China, in 2010. She is currently a Lecturer with the School of Computer and Information Science, Hubei Engineering University, Xiaogan, China. Her research interests include artificial intelligence, wireless mobile networks, and image processing.

UMER SADIQ KHAN received the Ph.D. degree in computer science and technology from Xian Jiaotong University, Xi'an, China. He is currently a Professor with the Department of Computer and Information Science, Hubei Engineering University, Xiaogen, Hubei, China. His research interests include computer vision, image processing, pattern recognition, and computer graphics.

ZENGGANG XIONG received the M.S. degree in computer application from Hubei University, in 2005, and the Ph.D. degree in computer application from the University of Science and Technology Beijing, in 2009. He is currently a Professor in computer science with Hubei Engineering University. His research interests include cloud computing and big data.