Contents lists available at SciVerse ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Scale-space texture description on SIFT-like textons [☆]

Yong Xu [a,*], Sibin Huang [a,b], Hui Ji [b], Cornelia Fermüller [c]

[a] *School of Computer Science & Engineering, South China University of Technology, Guangzhou 510006, China*
[b] *Department of Mathematics, National University of Singapore, Singapore 117543, Singapore*
[c] *Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, USA*

ARTICLE INFO

Article history:
Received 17 May 2011
Accepted 14 May 2012
Available online 24 May 2012

Keywords:
Texture
Multi-fractal analysis
Image feature
Wavelet tight frame

ABSTRACT

Visual texture is a powerful cue for the semantic description of scene structures that exhibit a high degree of similarity in their image intensity patterns. This paper describes a statistical approach to visual texture description that combines a highly discriminative local feature descriptor with a powerful global statistical descriptor. Based upon a SIFT-like feature descriptor densely estimated at multiple window sizes, a statistical descriptor, called the multi-fractal spectrum (MFS), extracts the power-law behavior of the local feature distributions over scale. Through this combination strong robustness to environmental changes including both geometric and photometric transformations is achieved. Furthermore, to increase the robustness to changes in scale, a multi-scale representation of the multi-fractal spectra under a wavelet tight frame system is derived. The proposed statistical approach is applicable to both static and dynamic textures. Experiments showed that the proposed approach outperforms existing static texture classification methods and is comparable to the top dynamic texture classification techniques.

1. Introduction

Visual texture has been found a powerful cue for characterizing structures in the scene, which give rise to certain patterns that exhibit a high degree of similarity. Classically, static image texture was used for classification of materials, such as cotton, leather or wood, and more recently it has been used also on unstructured parts of the scene, such as forests, buildings, grass, trees or shelves in a department store. Dynamic textures are video sequences of moving scenes that exhibit certain stationary properties in time, such as sequences of rivers, smoke, clouds, fire, swarms of birds, humans in crowds, etc. A visual texture descriptor becomes useful for semantic description and classification, if it is highly discriminative and at the same time robust to environmental changes [51]. Environmental changes can be due to a wide range of factors, such as illumination changes, occlusions, non-rigid surface distortions and camera viewpoint changes.

Starting with the seminal work of [21], static image texture has been studied in the context of various applications [15,20]. Earlier work was concerned with shape from texture (e.g. [1,16,28]), and most of the recent works are about developing efficient texture representations for the purpose of segmentation, classification, or synthesis. There are two components to texture representations: statistical models and local feature measurements. Some widely used statistical models include Markov random fields (e.g. [11,43]), joint distributions, and co-occurrence statistics (e.g. [22,23,36]). Local measurements range from pixel values over simple edge responses to local feature descriptors and filter bank responses (e.g. [7,19,24,25,30–32,43,46,47,49]).

Approaches employing sophisticated local descriptors usually compute as statistics various texton histograms based on some appearance based dictionary. Depending on the percentage of pixel information used in the description, these approaches can be classified into two categories: *dense* approaches and *sparse* approaches. Dense approaches apply appearance descriptors to every pixel. For example, Varma et al. [43] used the responses of the MR8 filter bank, consisting of a Gaussian, a LOG filter and edges in different directions at a few scales. In contrast, sparse approaches employ appearance-based feature descriptors at a sparse set of interest points. For example, Lazebnik et al. [24] obtained impressive results by combining Harris and Laplacian keypoint detectors and RIFT and Spin image affine-invariant appearance descriptors. Both the sparse and dense approaches have advantages and disadvantages. The sparse approaches achieve robustness to environmental changes because the features are normalized. However, they may lose some important texture primitives by using only a small percentage of the pixels. Also, there are stability and repeatability issues with the keypoint detection of existing point or region detectors. By using all pixels, the dense approaches provide rich information for local texture characterizations. However, on the negative side, the resulting descriptions tend to be more sensitive

[☆] This paper has been recommended for acceptance by Tinne Tuytelaars.

[*] Corresponding author.

E-mail addresses: yxu@scut.edu.cn (Y. Xu), maths@nus.edu.sg (S. Huang), matjh@nus.edu.sg (H. Ji), fer@umiacs.umd.edu (C. Fermüller).

1077-3142/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.cviu.2012.05.003

to significant environmental changes, such as changes in viewpoint, that affect the local appearance of image pixels. To rectify the local appearance, we would need adaptive region processes. Such processes, however, would require strong patterns in the local regions of image pixels, which are not available for most image points. Thus, rectification, which is standard for sparse sets of image points, cannot be adapted for dense sets.

In addition to the static texture in single images, dynamic texture analysis also considers a stochastic dynamic behavior in the temporal domain. Chetverikov and Péteri [5] gave a brief survey of methods on dynamic texture description and recognition. Earlier dynamic texture classification systems (e.g. [33,10,38,45]) often explicitly modeled the underlying physical process, and then distinguished different dynamic textures by the values of the associated model parameters. For example, Doretto et al. [10] used *linear dynamical system* (LDS) to characterize dynamic texture processes. The LDSs of the different textures were then compared in a space described by Stiefel manifolds using the Martin distance. Ghanem and Ahuja [18] introduced a phase-based model for dynamic texture recognition and synthesis. Dynamic characteristics of dynamic texture were measured in Fazekas and Chetverikov [13] using optical flow based statistical measurements. However, it appears that so far no universal physical process has been found that can model a large set of dynamic textures. Thus, recently, appearance based discriminative methods have become more popular for dynamic texture classification [3,37,44,50]. Wildes and Bergen [44] constructed spatiotemporal filters to qualitatively classify local motion patterns into a small set of categories. The descriptor proposed by Zhao and Pietikäinen [50] is based on local spatio-temporal statistics, specifically an extension of the local binary pattern (LBP) in 2D images to the 3D spatio-temporal volumes. To compare different descriptors efficiently the co-occurrence of LBPs was computed in three orthogonal planes. Ravichandran et al. [37] combined local dynamic texture structure analysis and generative models. They first applied the LDS model to local space-time regions and then constructed a bag-of-words model based on these local LDSs. Chan and Vasconcelos [3] used kernel PCA to learn a non-linear kernel dynamic texture and applied it for video classification.

In order to achieve good robustness necessary for semantic classification, both components of texture description, the local appearance descriptors and the global statistical characterization, should accommodate environmental changes. In the past, very robust local feature descriptors have been developed, such as the widely used SIFT feature [27] in image space. Most approaches making use of these feature points use histograms for global statistical characterization. However, such histograms are not invariant to global geometrical changes. Furthermore, important information about the spatial arrangement of local features is lost. An interesting statistical tool, the so-called MFS (multi-fractal spectra) was proposed in [47] as an alternative to the histogram. The advantage of the MFS is that it is theoretically invariant to any smooth transform (bi-Lipschitz geometrical transforms), and it encodes additional information regarding the regularization of the spatial distribution of pixels. A similar concept was used also in other texture applications, for example in texture segmentation [6]. In [47] the MFS was applied to simple local measurements, the so-called *local density function*, and in [48] it was applied to wavelets. Although the MFS descriptor proposed in [47] has been demonstrated to have strong robustness to a wide range of geometrical changes including viewpoint changes and non-rigid surface changes, its robustness to photometric changes is weak. The main reason is that the local feature description is quite sensitive to photometric changes. Moreover, the simple local measurements have limited discriminative information. On the other hand, local feature descriptors, such as SIFT [27], have strong robustness to photometric changes as has been demonstrated in many applications. In particular, the gradient orientation histogram used in SIFT and variations of SIFT has been widely used in many recognition and classification tasks including texture classification (e.g. [24]).

Here we propose a new statistical framework that combines the global MFS statistical measurement and local feature descriptors using the gradient orientation histogram. The new framework is applicable to both static and dynamic textures. Such a combination will lead to a powerful texture descriptor with strong robustness to both geometric and photometric variations. Fig. 1 gives an outline of the approach for static image textures. First, the scale-invariant image gradients are derived based on a modification of the scale-selection method introduced in [26]. Next, at every pixel multi-scale gradient orientation histograms are computed with respect to multiple window sizes. Then, using a rotation-invariant pixel classification scheme defined on the orientation histograms, pixels are categorized, and the MFS is computed for every window size. The MFSs corresponding to different window sizes together make up an MFS pyramid. The final texture descriptor is derived by sampling the leading coefficients (that is, coefficients of large magnitude) of the MFS pyramids under a tight wavelet frame transform [8].

The approach for dynamic textures is essentially the same as that for static textures with the 2D image SIFT feature replaced by the 3D SIFT feature proposed in Scovanner et al. [40]. Our approach falls in the category of appearance-based discriminative approaches. Its main advantage stems from its close relationship to certain stochastic self-similarities existing in a wide range of dynamic processes capable of generating dynamic textures.

The rest of the paper is organized as follows. Section 2 gives a brief review of the basic tools used in our approach. Section 3 presents the algorithm in detail, and Section 4 is devoted to experiments on static and dynamic texture classification. Section 5 concludes the paper.

2. Preliminaries: multi-fractal analysis

In this section, we give a brief review on multi-fractal analysis. A review on tight framelet systems is given in Appendix A. Multi-fractal analysis [12] is built upon the concept of the *fractal dimension*, which is defined on point sets. Consider a set of points E in the 2D image plane with same value of some attribute, e.g., the set of image points with same brightness. The fractal dimension of such a point set E is a statistical measurement that characterizes how the points in E are distributed over the image plane when one zooms into finer scales. One definition of the fractal dimension, associated with a relatively simple numerical algorithm, is the so-called *box-counting* fractal dimension, which is as follows: Let the image plane be covered by a square mesh of total $n \times n$ elements. Let $\#(E, \frac{1}{n})$ be the number of squares that intersect the point set E. Then the *box-counting* fractal dimension, denoted as $\dim(E)$, is defined as

$$\dim(E) = \lim_{n\to\infty} \frac{\log \#(E, \frac{1}{n})}{-\log \frac{1}{n}}. \tag{1}$$

In other words, the *box-counting* fractal dimension $\dim(E)$ measures the *power law* behavior of the spatial distribution of E over the scale $1/n$:

$$\#\left(E, \frac{1}{n}\right) \propto \left(\frac{1}{n}\right)^{-\dim(E)}.$$

In a practical implementation, the value of n is bounded by the image resolution, and $\dim(E)$ is approximated by the slope of the line fitted to

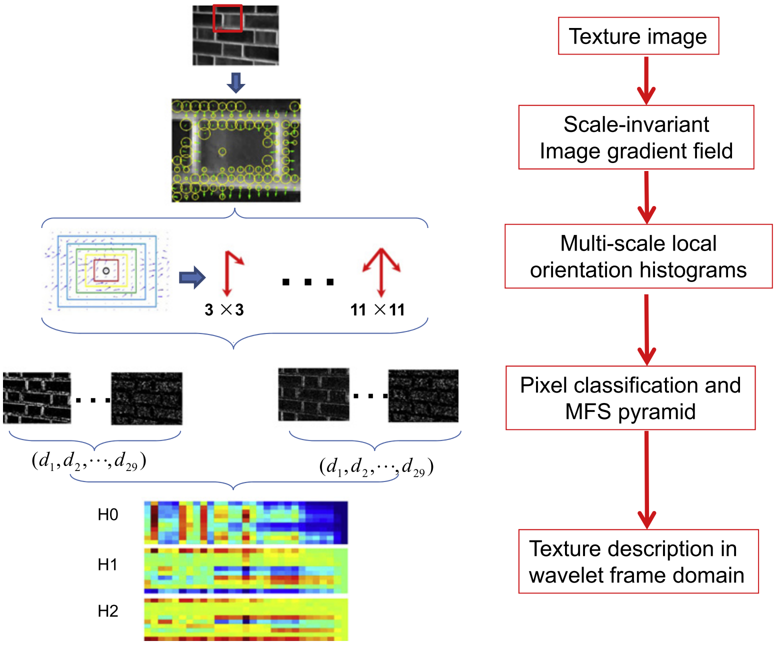

3 ×3
11 ×11
$(d_1, d_2, \cdots, d_{29})$
$(d_1, d_2, \cdots, d_{29})$
H0
H1
H2
Texture image
Scale-invariant Image gradient field
Multi-scale local orientation histograms
Pixel classification and MFS pyramid
Texture description in wavelet frame domain

Fig. 1. Outline of the proposed approach.

$$\log\#\left(E,\frac{i}{N}\right)\text{with respect to }-\log\frac{i}{N}\quad\text{for } i=1,2,\ldots,m,\quad m<N,$$

with N denoting the image resolution. In our implementation we use the least squares method at points at $i = 4, 5, 6, 7$ to estimate the slope.

Multi-fractal analysis generalizes the concept of the fractal dimension. One approach of applying multi-fractal analysis to images is to classify the pixels in the image into multiple point sets according to some associated pixel attribute α. For each value of α in its feasible discretized domain, let $E(\alpha)$ be the collection of all points with the same attribute value α. The MFS of E then is defined as the vector $\dim(E(\alpha))$ vs α. In other words,

$$\mathrm{MFS}=[\dim(E(\alpha_1)),\dim(E(\alpha_2)),\ldots,\ \dim(E(\alpha_n))].$$

For example, in [47] the density function (a function describing the local change of the intensity over scale) was used as the pixel attribute. The density was quantized into n values, and then the fractal dimensions of n sets associated with these n values were concatenated into a MFS vector.

3. Main components of the texture descriptor

Our algorithm, taking as input a static texture image, consists of four computational steps:

1. The first step is to calculate scale-invariant image gradients in the scale-space of the texture image. At each point the scale is determined by the maximum of the Laplacian measure resulting in a scale-invariant image gradient field.
2. Next, using as input the scale-invariant image gradient field, at every pixel local orientation histograms are computed over m window sizes ($m = 5$ in our implementation). Similar as in the SIFT feature approach, we use 8 directions in the orientation histogram. Two types of orientation histogram are used: one simply counts the number of edges in each direction and the other uses the summation of edge energy in each direction. Thus, in total we obtain $2{*}m$ sets of local orientation histograms for the given image.
3. Then the MFS pyramid is computed. The orientation histograms are discretized into n ($n = 29$ in our implementation) classes using rotation-invariant templates, and an MFS vector is computed on this classification. We then combine the m MFS vectors corresponding to the m window sizes into an MFS pyramid. At the end of this step, we have 2 MFS pyramids of size $m \times n$.
4. Finally, a sparse tight framelet coefficient vector of each MFS pyramid is estimated, by keeping only the frame coefficients of largest magnitude and setting to 0 all others.

The algorithms for static texture images and dynamic texture sequences are similar, but a SIFT-type descriptor in 2D image space is used in the former case and a SIFT-type descriptor in 3D spatio-temporal volume (see [40]) in the latter. Next, we give a detailed description of every step described in the algorithm above.

3.1. Scale-invariant image gradient field

The texture measurement of the proposed method is built upon the image gradients of the given image. To suppress variations of image gradients caused by possible scale changes, we compute the image gradients in scale-space. Given an image $I(x,y)$, its linear scale-space $L(x,y;\sigma)$ is obtained by convolving $I(x,y)$ with an isotropic Gaussian smoothing kernel of standard deviation σ:

$$g(x,y;\sigma)=\frac{1}{2\pi\sigma^2}e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)},\tag{2}$$

such that

$$L(x,y;\sigma)=(g(\cdot,\cdot;\sigma)*I)(x,y)\tag{3}$$

with a sequence of $\sigma = \{1,\ldots,K\}$ ranging from 1 to K ($K = 10$ in our implementation). Then, at each pixel (x,y), its associated image gradient is calculated as

$$[\partial_x L(x,y;\sigma_*(x,y)),\partial_y L(x,y;\sigma_*(x,y))]$$

for a particular standard deviation $\sigma_*(x,y)$. The value $\sigma_*(x,y)$ is determined by the scale selection method proposed in [26] which selects at every point the scale at which some image measurement takes on the extreme value. We use the Laplacian measurement, defined as

$$M_L = \sigma^4 (L_{x^2} + L_{y^2}) \tag{4}$$

with $L_{x^m y^n}(x, y; \sigma) = \partial_{x^m y^n}(L(x, y; \sigma))$. In our implementation, the Prewitt filters are used for computing the partial derivatives in scale-space. Then, the scale is derived by taking the maximum value of the Laplacian measurement over scale. The gradient magnitude and orientation are computed by applying the finite difference operator to $L(x, y; \sigma_*)$. See Fig. 2 for an illustration of the scale selected at each pixel and the corresponding image gradients.

3.2. Multi-scale local orientation histograms

Our proposed local feature descriptor relies on the local orientation histogram of image pixels, which also is used in SIFT [27] and similar features. Its robustness to illumination changes and invariance to in-plane rotations has been demonstrated in many applications. For each image gradient field computed in the previous step, at every pixel, two types of local orientation histograms are computed. One simply counts the number of orientations; the other weighs them by the gradient magnitude. The gradient orientations are quantized into 8 directions, covering 45 degrees each. To capture information of pixels in a multi-scale fashion, for each pixel, we compute the orientation histograms at 5 window sizes ranging from 3 × 3 to 11 × 11. The orientation histograms, as in SIFT, are rotated to align the dominant orientation with a canonical direction.

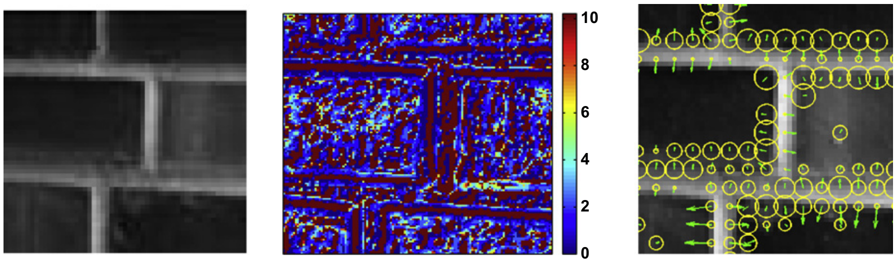
3.3. Pixel classification and the MFS

The next step is to compute the MFS vector. The MFS vector depends on how the pixels are classified. To obtain a reasonable statistics of the spatial distribution of pixels, the number of pixels in each class needs to be sufficiently large. We thus need a meaningful way of discretizing the very large amount of possible orientation histograms. Our approach is to introduce a fixed bin partitioning scheme based on a set of basic orientation histogram templates.

First, the estimated orientation histograms are quantized as follows. For each bin the value is set to 0 if the magnitude is less than $\frac{1}{8}$ of the overall magnitude and to 1 otherwise. We then define a partitioning scheme based on the topological structure of orientation histograms, with a total of 29 classes. See Fig. 3a and b for an illustration. The proposed templates are defined on the basis of the number of significant image gradient orientations and their relative positions. Each template class contains the basic element shown in Fig. 3a and all of its rotated and mirror-reflected copies as shown in Fig. 3b for one of the elements.

Next, for each window size the corresponding MFS feature vector is calculated as follows: For each template class (out of 29 classes), a binary image is derived by setting the value of the pixel to 1 if its associated template falls into the corresponding template class and to 0 otherwise (see Fig. 4). Thus, there are 29 binary images. For each binary image the box-counting fractal dimension is computed, and the fractal dimensions are concatenated into a 29-dim MFS vector. The MFS feature vectors corresponding to different window sizes are then combined into a multi-scale MFS pyramid. The size of this MFS pyramid is 5 × 29.

It is noted that the box-counting fractal dimension amounts to fitting the slope of the line in the co-ordinate space of $\log \#(E, \frac{i}{N})$ vs. $-\log \frac{i}{N}$. Thus, the validity of the MFS largely depends on how applicable such linearity assumption is for the given data. In our application we used four points only (corresponding to four window sizes) in the computation, and we found the variance in the fitting reasonably small to justify the fitting. Fig. 5a and b (the forth



(a) Sample texture region (b) Scale σ_* (c) Image gradient

Fig. 2. (a) Sample texture region. (b) Selected scale σ_* based on the maximum of the Laplacian measure in scale-space with the scale ranging from 1 to 10. (c) Corresponding image gradient field, where the circle at a point denotes the size of the Gaussian smoothing kernel (defined by σ_*) when computing the gradient.

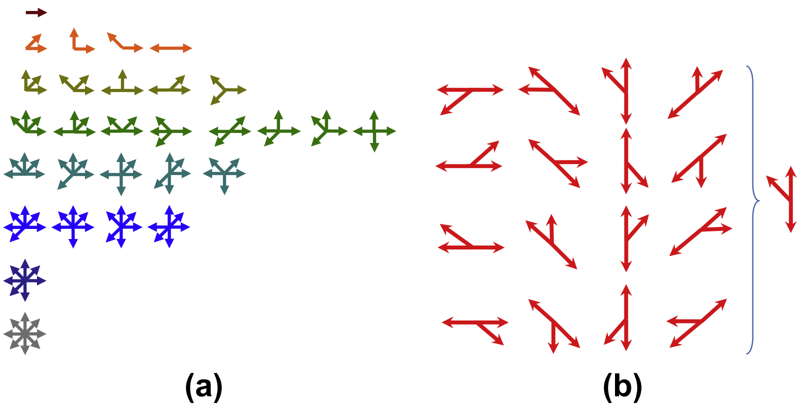


Fig. 3. (a) Representative elements for each of the 29 classes of orientation histogram templates. (b) All the elements in one orientation histogram template class, which are obtained from the possible mirror-reflections and rotations of the basic element.

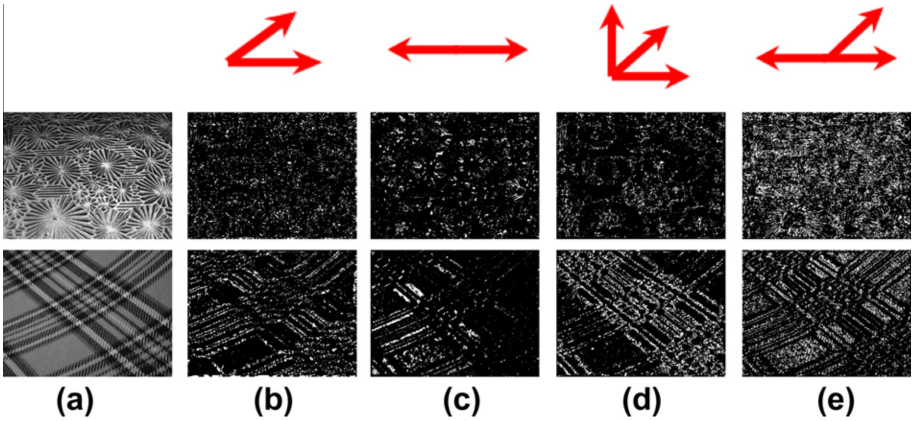
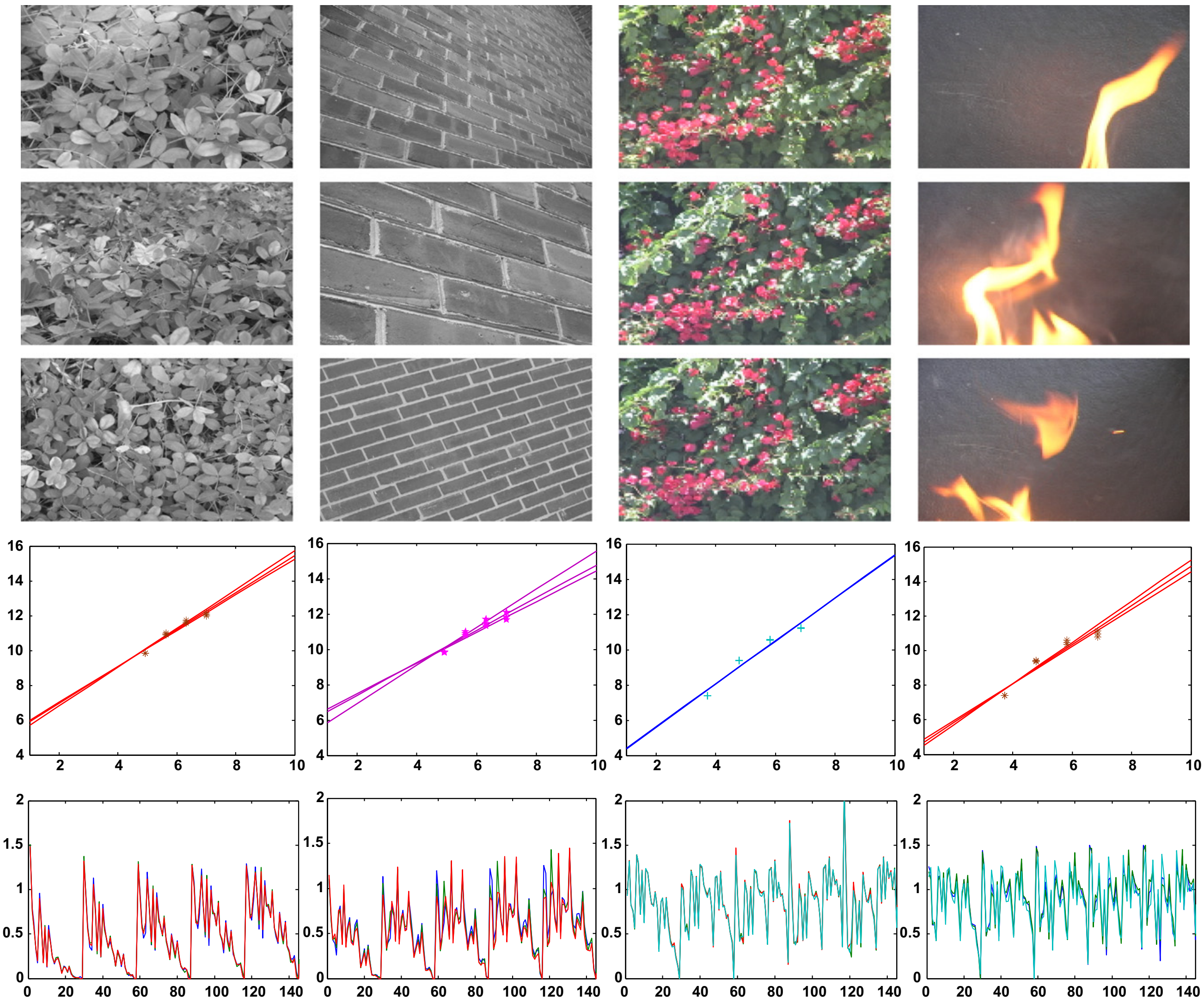


Fig. 4. (a) Two texture images in UIUC dataset [27]. (b)–(e) Examples of binary images with respect to pixel classification based on the orientation histogram templates.



(a) $\sigma = 0.05$ (b) $\sigma = 0.11$ (c) $\sigma = 0.01$ (d) $\sigma = 0.04$

Fig. 5. Illustration of the MFS and the linear fitting behavior when computing the fractal dimensions for 2 static texture classes in (a, b) and 2 dynamic texture classes in (c, d). The sample static textures are from the UMD dataset [47], and the sample dynamic textures are from Ref. [9]. For each class, the first three rows show three sample static texture images, or key frames of three sample dynamic textures. The forth row shows for one particular orientation histogram template, the graph of linear fitting in the coordinates of $\log \#(E, \frac{i}{N})$ vs. $-\log \frac{i}{N}$, $i = 4, \ldots, 7$. The mean variances of the line fitting were found as 0.05, 0.11, 0.01 and 0.04 respectively. The fifth row shows the MFS pyramids (as vectors) of the corresponding texture images and dynamic texture sequences.

row) illustrate the behavior of the linear fitting in log-log coordinates for three images, each from two of the classes in the UMD dataset [47], which represent one of the best and one of the worst cases in the set, with variances of 0.05 and 0.11, respectively. The last row in Fig. 5 illustrates the corresponding MFSs. As can be seen, both for (a) and (b), the MFS pyramids of the three textures are almost the same, demonstrating that the MFS descriptor captures well the identity of texture classes.

It is easy to see that the orientation histogram templates provide a pixel classification scheme which is invariant to rotation and mirror-reflection; in addition, the robustness to illumination changes is guaranteed by the orientation histogram itself [27]. Using the MFS as the replacement of the histogram for statistical characterization leads to better robustness to global geometric changes (see [47] for more details).

3.4. Robustifying the texture descriptor in the wavelet frame domain

The final step is to construct the texture descriptor by only taking the leading coefficients of the MFS pyramids in a wavelet frame domain. The purpose is to further increase the robustness of the texture descriptor to environmental changes. The construction is done as follows: We first decompose the MFS pyramid using the 1D un-decimal linear-spline framelet transform [8], as it has been empirically observed that the corresponding tight frame coefficients tend to be highly relevant to the essential structure of textures.

Let the matrix $E(s,n)$ denote the MFS pyramid where s denotes the scale (window size of local orientation histogram) and n denotes the index of the template class. Let $\mathcal{F}$ denote the L-level decomposition of $E(s,n)$ under a 1D tight framelet system with respect to s defined as

$$\mathcal{F}(j,s,n) := AE(s,n),$$

where A is the frame decomposition operator, and j denotes the level of the frame decomposition. See Appendix A for more details on the frame decomposition operator A. The multi-dimensional matrix $\mathcal{F}$ consists of two kinds of components: one low-pass framelet coefficient component H_0, the output of applying the low-pass h_0 on the pyramid at scale 2^{-L}; and multiple high-pass framelet coefficient components $H_1, \ldots, H_r$, the outputs of applying high pass filters $h_1, \ldots, h_r$ to the pyramid at multiple levels ranging from $2^{-1}, \ldots, 2^{-L}$. Each high-pass framelet coefficient component has three variables: scale 2^{-j}, $j = 1, \ldots, L$, level s, $s = 1, \ldots, 5$ and bin index n, $n = 1, 2, \ldots, 29$. See Fig. 6 for an illustration of the single level frame coefficients of the sample images in Fig. 4.

Recall that the un-decimal framelet tight frame is a redundant transform, and thus the information encoded in the framelet coefficients of $\mathcal{F}$ is redundant. In contrast to orthogonal mappings, redundant transforms are likely to yield sparse leading coefficients with large magnitude. The next step then involves extracting these leading coefficients such that the resulting descriptor provides strong robustness to inter-class texture variations. In our approach, we simply keep the 70% leading coefficients with largest amplitude and set all others to 0. The final texture descriptor then consists of only leading framelet coefficients of all MFS pyramids. The final dimension of the resulting descriptor in our implementation is $3 \times 2 \times 5 \times 29 = 870$.

3.5. Dynamic texture

Dynamic textures are image motion sequences that vary not only in the spatial distribution of texture elements, but also in their dynamics over time. Dynamic texture can be regarded as a 3D volume of data, which encodes both spatial distribution and temporal variations of texture pixels. To capture the spatio-temporal nature of dynamic texture, the 3D SIFT descriptor [40] is used in our approach. The procedure is essentially the same as the one for 2D image textures. Thus, in this section, we only highlight the differences in the four steps.

In our approach, a dynamic texture is viewed as a 3D volume of data with three orthogonal axes, i.e. two spatial axes (x-axis and y-axis) and a time axis (t-axis). For Step 1 (Section 3.1), the spatio-temporal gradients of each pixel (x,y,t) in 3D volume, denoted by L_x, L_y and L_t, are computed using the finite difference operator. Since most dynamic textures don't have large scale changes, the step of calculating scale-invariant image gradients is omitted. Instead, we just use the standard image gradients.

The main difference lies in Step 2 (Section 3.2). We need to define local orientation histograms that capture the spatio-temporal aspect of dynamic textures. Following [40], for a given point in 3D volume, we parameterize its orientation by the angle vector $[\phi, \psi]$ defined as

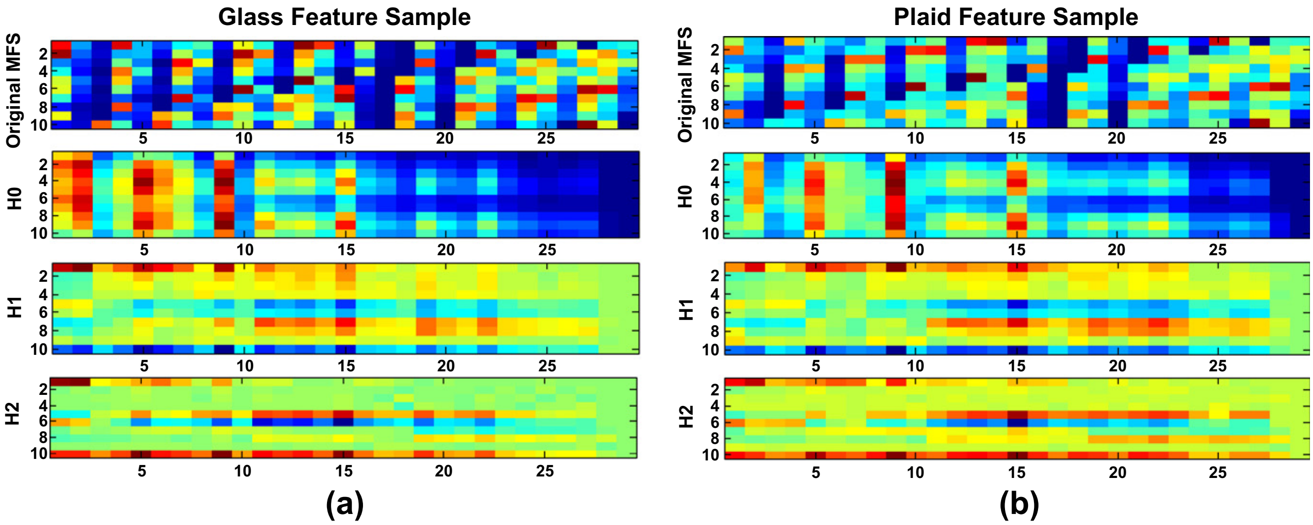
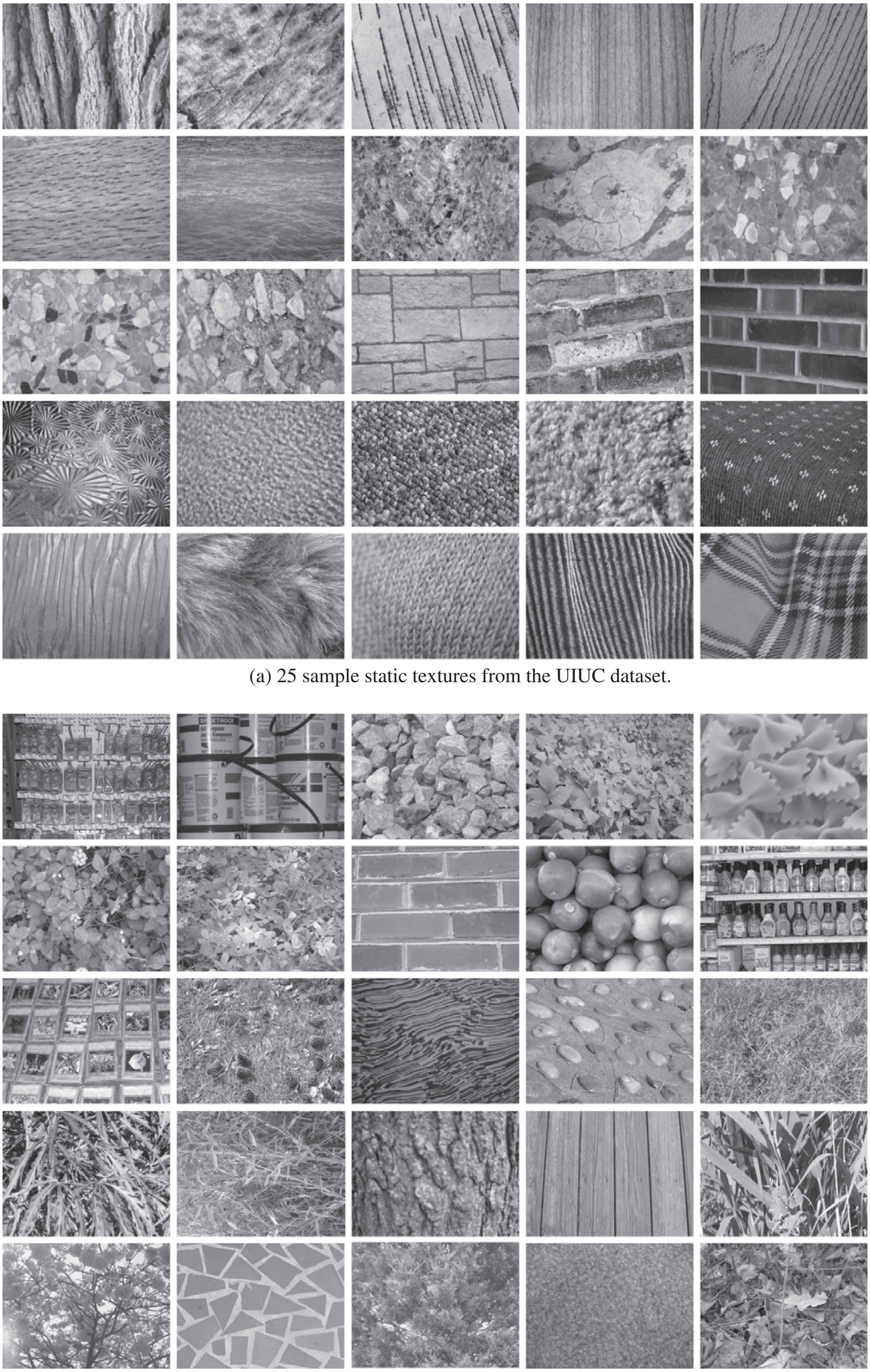


Fig. 6. Illustration of the framelet coefficient components of the MFS vector at a single scale, including the low-pass framelet coefficient component H_0 and two high-pass framelet coefficient components $\{H_1, H_2\}$. (a) Framelet features of the glass image in Fig. 4 for maximum of Laplacian measure. (b) Framelet features of the plaid image in Fig. 4 for maximum of Laplacian measure. In each component of the frame coefficients, the first five rows are corresponded to gradient orientation, and the last five rows are corresponded to gradient magnitude.



(a) 25 sample static textures from the UIUC dataset.

(b) 25 sample static textures from the UMD dataset.

Fig. 7. Sample static texture images.

$$\begin{cases} \phi = \tan^{-1} \frac{L_y}{L_x} \\ \psi = \tan^{-1} \frac{L_t}{\sqrt{L_x^2+L_y^2}}, \end{cases}$$

with the two angles ranging from 0° to 360°. To reduce the computational cost, we only use the orientation variable ψ in the orientation histogram templates. This variable captures the temporal information of dynamic textures. Using the orientation histogram templates described in Section 3.2 with respect to ψ, we obtain the orientation histograms for the 3D volume data. The procedure of computing dynamic texture descriptor is as follows.

1. For each pixel, we compute the orientation histograms with respect to parameter ψ at 5 windows (3D cubes) ranging in size from $3 \times 3 \times 3$ to $11 \times 11 \times 11$. For each scale, we compute two types of local orientation histograms, one based on the number of orientations, the other based on the gradient magnitude.
2. Then we classify the volumetric windows into 29 classes based on the 29 orientation histogram templates described in Section 3.2.
3. Based on this classification we calculate using the 3D *box-counting* fractal dimension (1) the MFS vectors, and concatenate the MFS feature vectors of different window sizes into a multi-scale MFS pyramid.

It is noted that the last step used in the computation of static textures (Section 3.3) is not used here, as it leads to very minor improvements in the classification experiments. The final dimension of the 3D dynamic texture descriptor is $2 \times 5 \times 29 = 290$. Fig. 5c and d illustrate the estimated MFS and the fitting of the line for one 3D orientation histogram template on a good and a bad case, demonstrating the variance sufficiently small to justify the linearity assumption in the estimation of the fractal dimension.

4. Experimental evaluation

The performance of the proposed texture descriptor is evaluated for static and dynamic texture classification. All code is available at [52].

4.1. Static texture

We evaluated the performance of texture classification on two datasets, the UIUC dataset [27] and the high-resolution UMD dataset [47]. Sample images of these datasets are shown in Fig. 7. The UIUC texture dataset consists of 1000 uncalibrated and unregistered images: 40 samples for each of 25 textures with a resolution of 640 × 480 pixels. The UMD texture dataset also consists of 1000 uncalibrated and unregistered images: 40 samples for each of 25 textures with a resolution of 1280 × 900 pixels. In both datasets significant viewpoint changes and scale differences are present, and the illumination conditions are uncontrolled.

In our experiments, the training set is selected as a fixed size random subset of the class, and all remaining images are used as the test set. A final texture description is based on a two-scale framelet-based representation. The reported classification rate is the average over 200 random subsets. An SVM classifier [4,41] is used, which was implemented as in Pontil et al. [35]. The features of the training set are used to train the hyperplane of the SVM classifier using RBF kernels as described in Scholkopf et al. [39]. The optimal parameters are discovered by cross-validation.

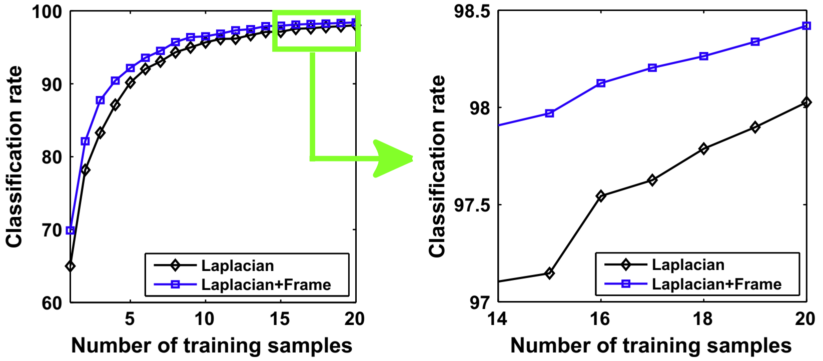


Fig. 8. Comparison of the average classification rates vs. number of training samples of the proposed descriptor with and without wavelet robustification. The experiment was run on the UMD dataset using SVM classification.

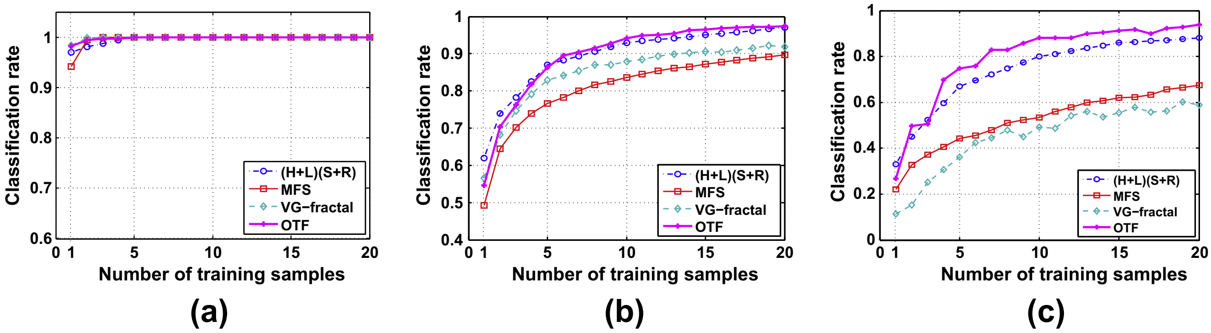


Fig. 9. Classification rate vs. number of training samples for the UIUC dataset based on SVM classification. Four methods are compared: the (H+L)(S+R) method in Lazebnik et al. [24], the MFS method in Xu et al. [47], the VG-Fractal method in Varma et al. [42] and our OTF method. (a) Classification rate for the best class. (b) Mean classification rate for all 25 classes. (c) Classification rate of the worst class.

The approach was implemented in Matlab 2011b and run on a laptop computer with Intel Core 2 Duo with 2.10 GHz and 4 GB memory. For each image in the UIUC dataset, the running time of the proposed feature extraction is about 10 s. Since our proposed approach does not require expensive clustering, the classification is very efficient. The average running time is around 16 s for classifying 750 images of 25 classes from the UIUC dataset using the SVM-based classifier with 10 training samples for each class.

To understand the influence of applying the wavelet transform on feature vectors, we compared the average classification rates of the proposed texture descriptor with and without wavelet frame robustification. Referring to Fig. 8, it can be seen that the wavelet robustification provides a small amount of improvement, although not significant.

The proposed texture descriptor is compared against three other texture descriptors: Lazebnik et al. [24], Varma et al. [42], and Xu et al. [47]. The first one [24] is the so-called (H+L)(S+R) texture descriptor, which is based on a sophisticated point-based texture representation. The basic idea is to first characterize the texture by clusters of elliptic regions. The ellipses are then trans-

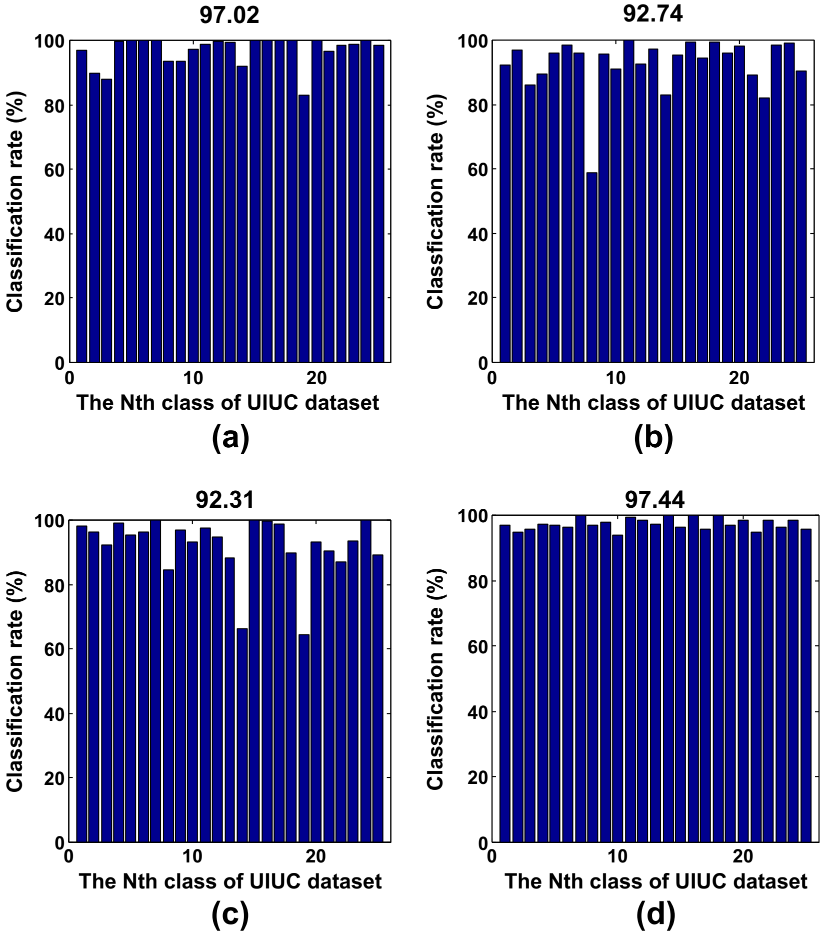


Fig. 10. Classification percentage vs. index of classes for the UIUC dataset based on SVM classification. The number of training samples is 20. The number on the top of each sub-figure is the average classification percentage of all 25 classes. (a) Result of the (H+L)(S+R) method. (b) Result of the MFS method. (c) Result of the VG-Fractal method. (d) Result of our OTF method.

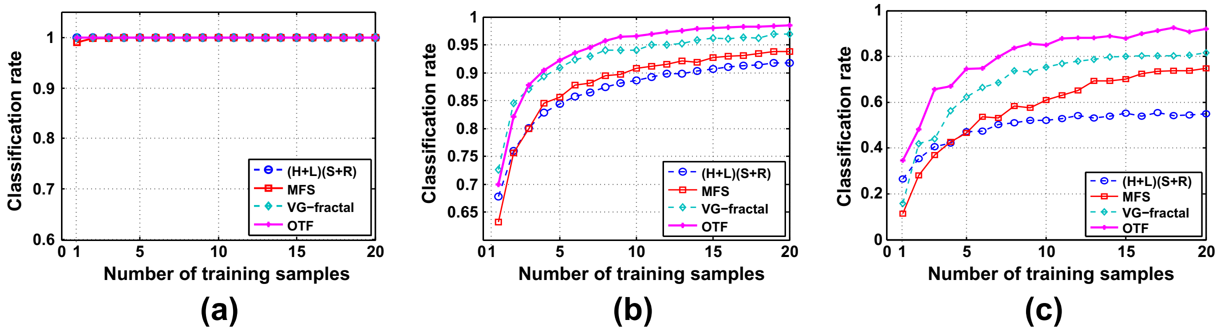


Fig. 11. Classification rate vs. number of training samples for the UMD dataset using SVM classification. Four methods are compared: the (H+L)(S+R) method, the MFS method, the VG-Fractal method and our OTF method. (a) Classification rate for the best class. (b) Mean classification rate for all 25 classes. (c) Classification rate of the worst class.

formed to circles such that the local descriptor is invariant to affine transforms. Two descriptors (SPIN and SIFT) are defined on each region. The resulting texture descriptor is the histogram of clusters of these local descriptors, and the descriptors are compared using the EMD distance. The second method is the VG-fractal method by Varma and Garg [42], which uses properties of the local density function of various image measurements resulting in a 13 dimensional descriptor. The resulting texture descriptor is the histogram

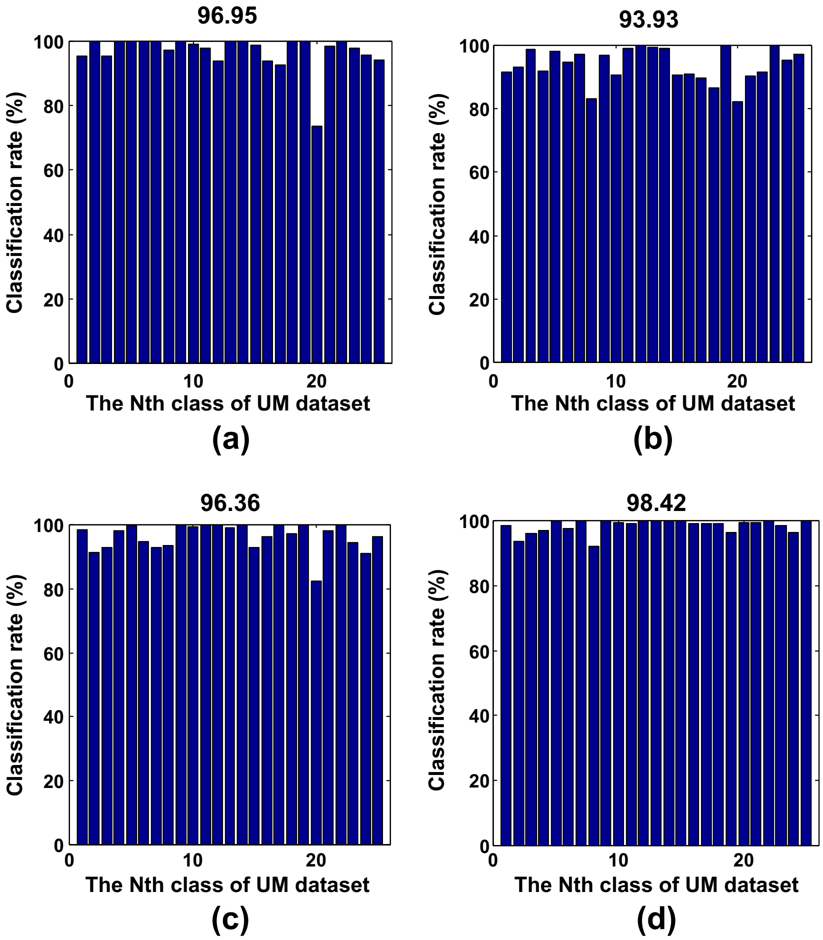


Fig. 12. Classification percentage vs. index of classes for the UMD dataset based on SVM classification. The number of training samples is 20. The number on the top of each sub-figure is the average classification percentage of all 25 classes. (a) (H+L)(S+R) method. (b) MFS method. (c) VG-fractal method. (d) OTF method.

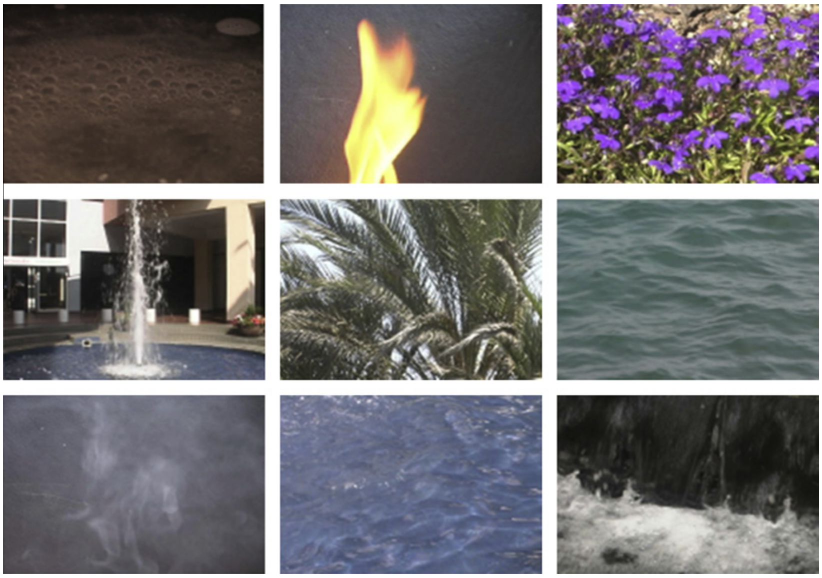
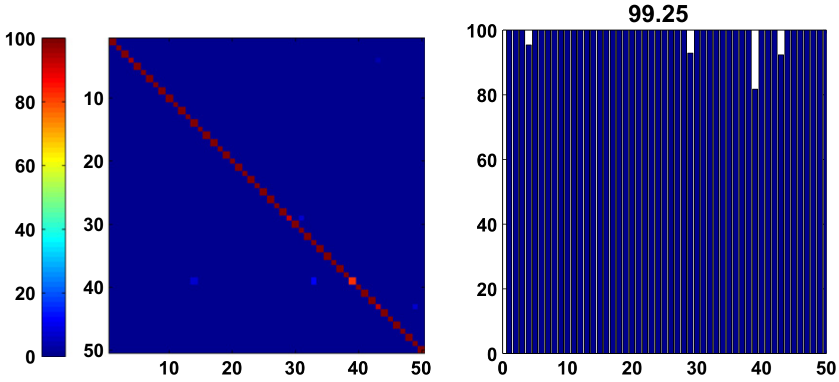


Fig. 13. Samples images from the dynamic textures in the DT-9 dataset.

Table 1
Classification results (in %) for the UCLA dataset. Note: Superscript "*M*" is used to denote results using maximum margin learning (followed by 1NN) [17]; "–" means "not available". Boldface print is used to mark the best results.

Method	DT-7		DT-8		DT-9		DT-50		DT-SIR
Classifier	1NN	SVM	1NN	SVM	1NN	SVM	1NN	SVM	1NN
[37]	–	–	70.00	80.00	–	–	–	–	–
[9]	92.30	–	–	–	–	–	81.00	–	60.00
[17]	–	–	–	–	95.60^M	–	99.00^M	–	–
3D-OTF	**96.11**	**98.37**	**95.80**	**99.50**	**96.32**	**97.23**	**99.25**	**87.10**	**67.45**



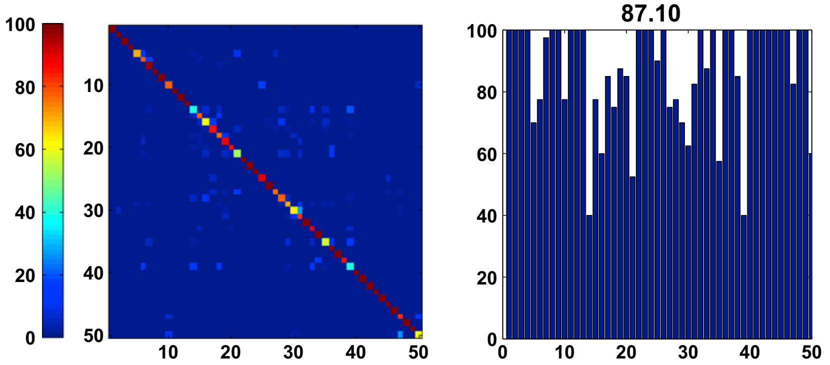


Fig. 14. Confusion matrices for DT-50 for classification. The upper is using a NN classifier, and the lower is using an SVM classifier.

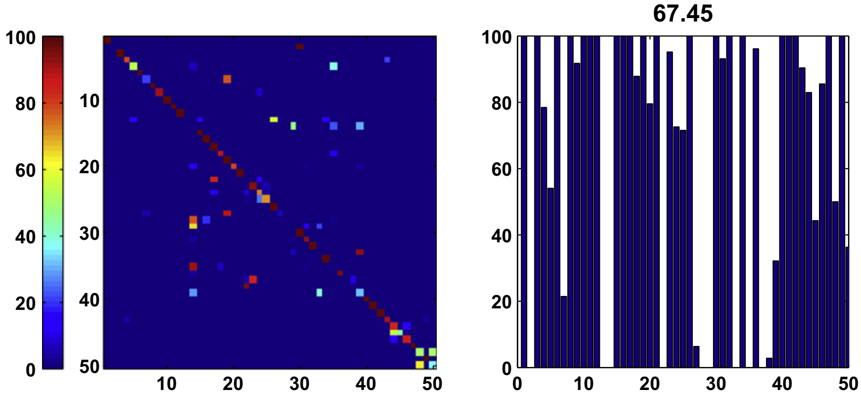
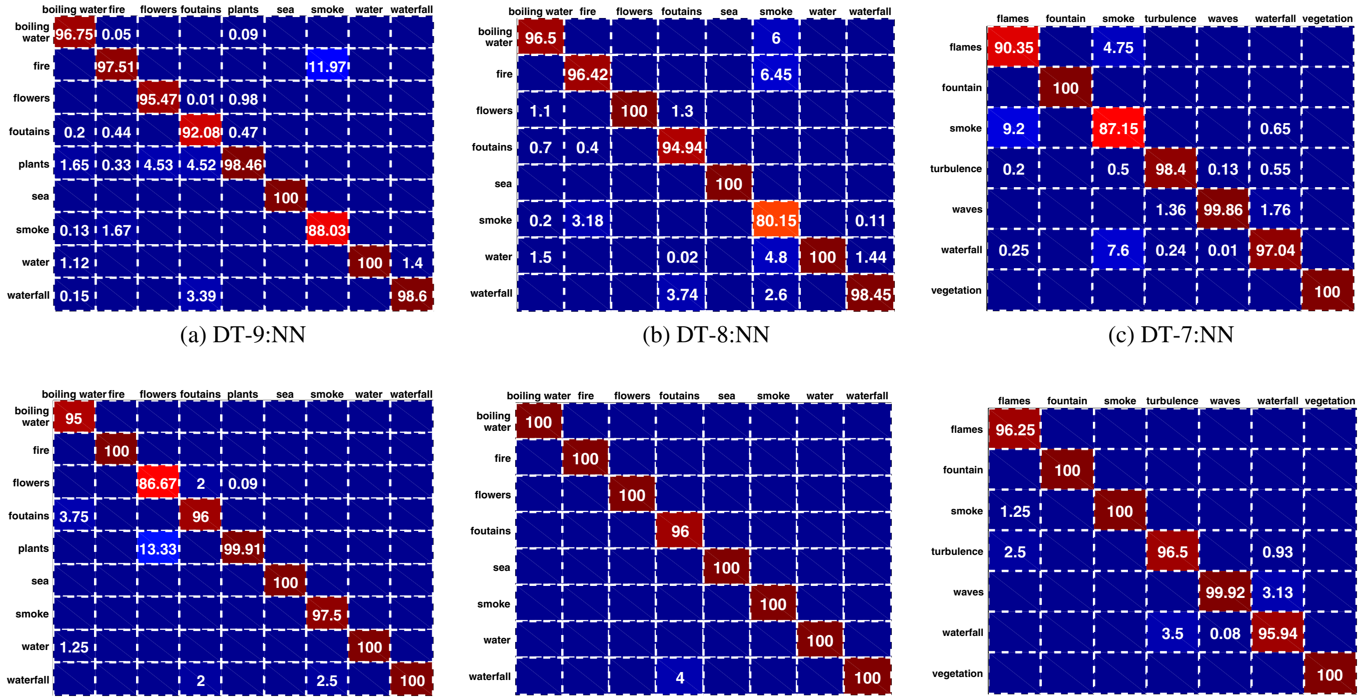


Fig. 15. Confusion matrix for DT-SIR for classification using NN classifier.

of clusters of these local descriptors. The third method, the MFS method by Xu et al. [47], derives the MFSs of simple local measurements (the local density function of the intensity, image gradient and image Laplacian). The texture descriptor is a combination of the three MFSs. The results on the UIUC dataset using the SVM classifier for the (H+L)(S+R) method is from [24]. The other results are obtained from our implementations. We denote our approach as OTF method. Fig. 9 shows the classification rate vs. the number of training samples on the UIUC dataset. Fig. 10 shows the classification percentage vs. the index of classes on the UIUC dataset based on 20 training samples. Figs. 11 and 12 show the results of the UMD dataset using the same experimental evaluation.



(a) DT-9:NN

	boiling water	fire	flowers	foutains	plants	sea	smoke	water	waterfall
boiling water	96.75	0.05			0.09				
fire		97.51					11.97		
flowers			95.47	0.01	0.98				
foutains	0.2	0.44		92.08	0.47				
plants	1.65	0.33	4.53	4.52	98.46				
sea						100			
smoke	0.13	1.67					88.03		
water	1.12							100	1.4
waterfall	0.15			3.39					98.6

(b) DT-8:NN

	boiling water	fire	flowers	foutains	sea	smoke	water	waterfall
boiling water	96.5					6		
fire		96.42				6.45		
flowers	1.1		100	1.3				
foutains	0.7	0.4		94.94				
sea					100			
smoke	0.2	3.18				80.15		0.11
water	1.5			0.02		4.8	100	1.44
waterfall				3.74		2.6		98.45

(c) DT-7:NN

	flames	fountain	smoke	turbulence	waves	waterfall	vegetation
flames	90.35		4.75				
fountain		100					
smoke	9.2		87.15			0.65	
turbulence	0.2		0.5	98.4	0.13	0.55	
waves				1.36	99.86	1.76	
waterfall	0.25		7.6	0.24	0.01	97.04	
vegetation							100

(d) DT-9:SVM

	boiling water	fire	flowers	foutains	plants	sea	smoke	water	waterfall
boiling water	95								
fire		100							
flowers			86.67	2	0.09				
foutains	3.75			96					
plants			13.33		99.91				
sea						100			
smoke							97.5		
water	1.25							100	
waterfall				2			2.5		100

(e) DT-8:SVM

	boiling water	fire	flowers	foutains	sea	smoke	water	waterfall
boiling water	100							
fire		100						
flowers			100					
foutains				96				
sea					100			
smoke						100		
water							100	
waterfall				4				100

(f) DT-7:SVM

	flames	fountain	smoke	turbulence	waves	waterfall	vegetation
flames	96.25						
fountain		100					
smoke	1.25		100				
turbulence	2.5			96.5		0.93	
waves					99.92	3.13	
waterfall				3.5	0.08	95.94	
vegetation							100

Fig. 16. Confusion matrices for UCLA DT-9, DT-8 and DT-7 for classifications, the upper is using NN classifier, and the lower is using SVM classifier.

Table 2
Results in leave-one-group-out test (%) on DynTex dataset.

	LBP-TOP [50]	3D-OTF
Non-weighting	95.71	95.89
Best-weighting	97.14	96.70

From Fig. 9–12, it is seen that our method clearly outperformed the VG-fractal method and the MFS method on both datasets. Also our method obtained better results than the (H+L)(S+R) method. We emphasize that heavy clustering is needed in both, the VG-fractal method and the (H+L)(S+R) method, which is very computationally expensive. In contrast, our approach is much simpler and efficient without requiring clustering.

4.2. Dynamic texture

There are three public dynamic texture datasets that have been widely used: the UCLA dataset [10], the DynTex dataset [34] and the DynTex++ dataset [17]. We applied our dynamic texture descriptor for dynamic texture classification on these three datasets and compared the results with those from a few state-of-the-art dynamic texture classification approaches.

4.2.1. UCLA dataset

A popular dynamic texture benchmark for performance evaluation is the UCLA dataset (*e.g.* [9,17,38,37,45]). The original UCLA dataset consists of 50 dynamic textures. Each dynamic texture is given in terms of four grayscale image sequences captured from the same viewpoint, resulting in a total of 200 sequences, each of which consists of 75 frames of size 110∗160. The literature does not agree on a ground truth regarding the classification of the UCLA dataset. In [9,17,37] the following five classifications, termed DT-50, DT-SIR, DT-9, DT-8 and DT-7, were considered:

1. **DT-50** [9,17]. All 50 classes are used for classification.
2. **DT-SIR** (Shift-invariant recognition) [9]. Each of the original 200 video sequences is spatially cut into non-overlapping, left and right halves resulting in a total of 400 sequences. The ‘‘shift-

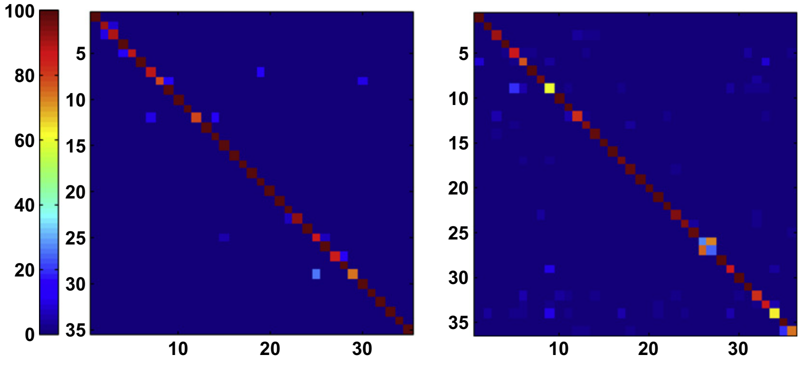


Fig. 17. Confusion matrices by our method 3D-OTF on the DynTex (left) and DynTex++(right) datasets.

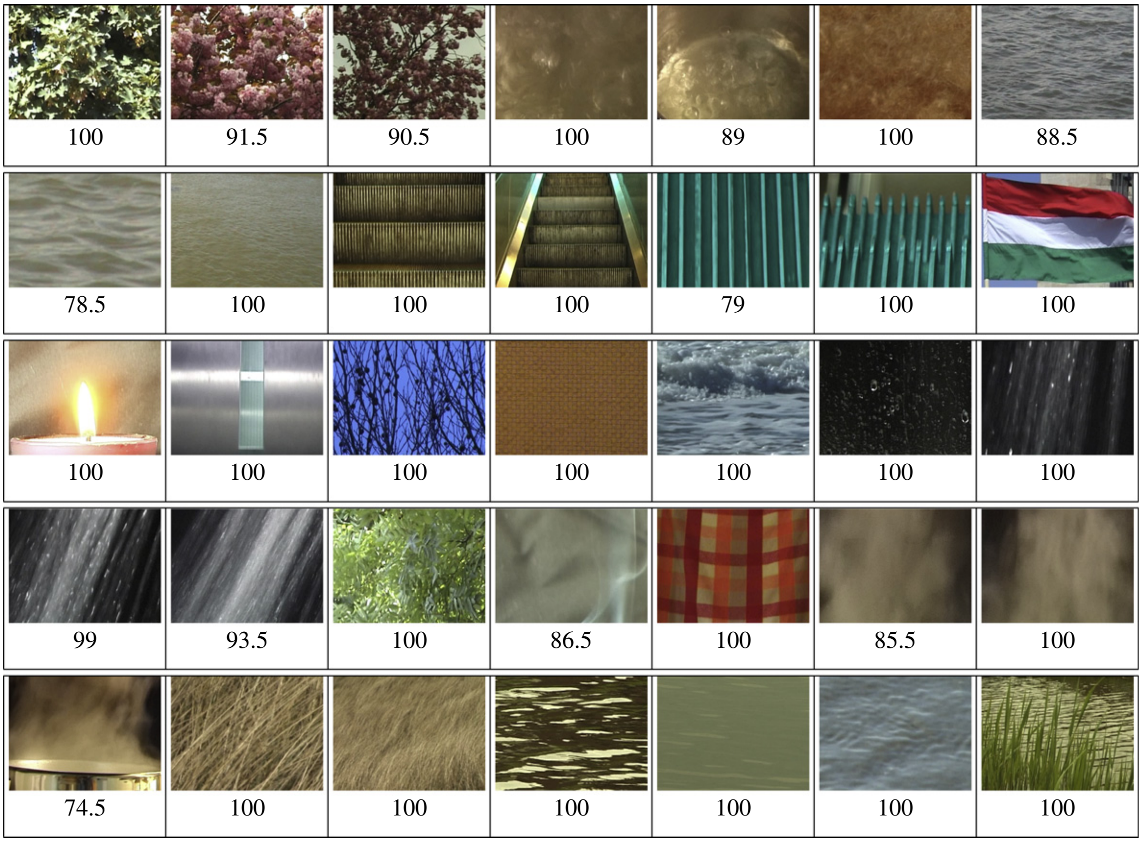


Fig. 18. Classification rate (in %) for the different classes of the DynTex dataset.

Table 3
Results (%) on DynTex++dataset.

Method	DL-PEGASOS [17]	3D-OTF
Classification rate	63.70%	89.17%

invariant recognition" was used to eliminate the effects due to biases in identical viewpoint selection. Nearest-neighbor classification was applied in the recognition process.

3. **DT-9** [17]. The dataset is divided into 9 classes: boiling water (8), fire (8), flowers (12), fountains (20), plant (108), sea (12), smoke (4), water (12) and waterfall (16), where the number in parentheses denotes the number of elements of each class. Sample frames are shown in Fig. 13. In our experiments we used the original images of size 110*160.
4. **DT-8** [37]. This dataset is obtained from DT-9 by discarding the large class "plants", and considering only the eight other classes.
5. **DT-7** [9] The original sequences in the dataset are split spatially into left and right halves resulting in 400 sequences, which were classified into seven semantic categories: flames (16), fountain (8), smoke (8), turbulence (40), waves (24), waterfall (64) and vegetation (240).

We compared our method using both NN(Nearest-neighbor) and SVM classifiers to the methods in [9,17] and [37] on the five categorizations (DT-7, DT-8, DT-9, DT-50 and DT-SIR). See Table 1 for a comparison of these methods. As can be seen from Table 1 our method outperformed the other three state-of-the-art methods. We also included the so-called *confusion matrix* to show the details

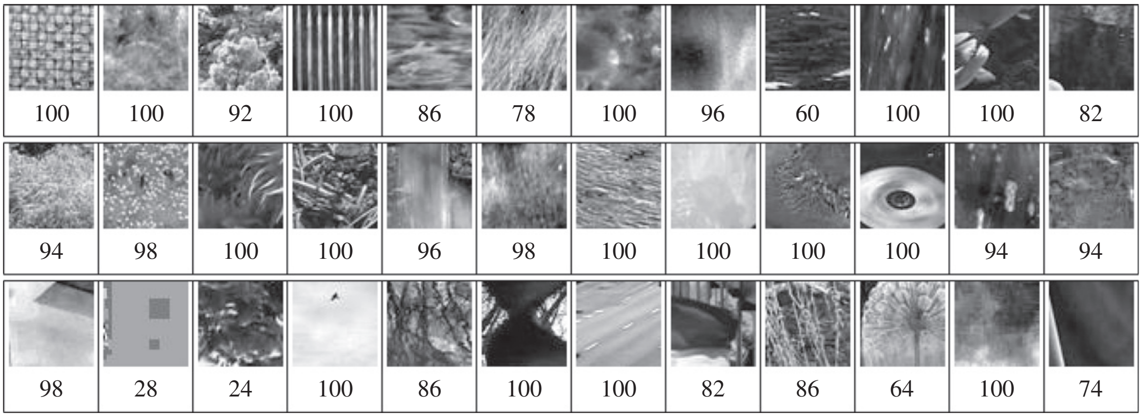


Fig. 19. Classification rate (in %) for the different classes of the DynTex++dataset.

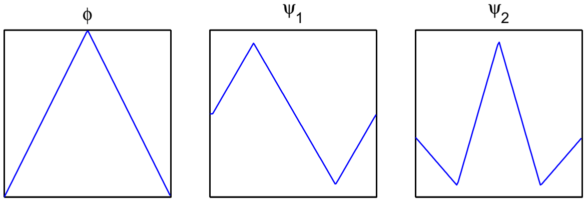


Fig. A.20. Piecewise linear wavelet frame system [8].

of the performance of the proposed method for each class. Each column of the confusion matrix represents the instances in a predicted class and each row represents the instances in an actual class. The confusion matrices of the proposed method for DT-50, DT-SIR, DT-9, DT-8 and DT-7 are shown in Figs. 14–16 respectively.

4.2.2. DynTex dataset

The DynTex dataset [34] consists of various kinds of videos of dynamic texture, including struggling flames, whelming waves, sparse curling smoke, dense swaying branches, and so on. The sequences are in color and of dimension 400 × 300 in space and consisting of 250 frames (over 10 s) de-interlaced with a spatio-temporal median filter.

The DynTex dataset has been used in [14,17,50] with different experimental configurations. Here we follow the settings in [50], and we compare to the method described there, which achieved very good recognition performance using the so-called LBP-TOP [50] method. This method in essence extends the so-called 2D LBP descriptor (a qualitative local statistical descriptor, that codes for a point which pixels in its neighborhood have larger value and which have smaller value than the point) to the spatio-temporal domain by applying the LBP descriptors in three orthogonal planes. The classification was implemented using the leave-one-group-out scheme. Table 2 reports the average performance over 2000 runs. It can be seen from Table 2 that our method performs better than the method in [50] when not using weighting, but performs worse when weighting is used. The confusion matrix is shown in Fig. 17, and the classification rate for individual classes is shown in Fig. 18.

4.2.3. DynTex++dataset

The DynTex++dataset [17] provides a rich and reasonable benchmark for dynamic texture recognition. This challenging dynamic texture dataset contains 36 classes of dynamic textures, each of which contains 100 sequences of size 50 × 50 × 50. The DL-PEGASOS method proposed in [17] is chosen for comparison, which is based on the maximum margin distance learning (MMDL) method. Good performance is obtained on the UCLA dataset and the DynTex++dataset by learning class-independent and class-dependent weights. We used the experimental setting in [17]. SVM was used as a classifier, with 50% of the dataset used for training and the rest for testing. Table 3 summarizes the comparison. Our 3D-OTF descriptor obtained an average recognition rate of 89.17%, which is noticeably better than the 63.7% achieved by the method in [17]. The confusion matrix is shown in Fig. 17, and the classification rate on each class of the DynTex++dataset is shown in Fig. 19.

5. Summary and conclusions

In this paper, we proposed a new texture descriptor, which applies the global MFS to local gradient orientation histograms. The proposed descriptor has strong robustness to both local and global illumination changes and is robust to many geometric changes. Locally, robustness to illumination changes and geometric variations is achieved by using templates of local gradient orientation histograms; robustness to local scale changes is achieved by using scale-invariant image gradient fields. Globally, the multi-fractal spectrum [47] and its sparse approximation in a wavelet frame system are employed to obtain further robustness to global environmental changes. Our texture description is rather efficient and simple to compute without feature detection and clustering. Experiments on static and dynamic texture classifications showed that our approach performed well. In future research, we would like to investigate how to apply the proposed framework to other recognition tasks including object recognition and scene understanding.

Acknowledgments

Y. Xu was partially supported by Program for New Century Excellent Talents in University (NCET-10-0368), the Fundamental Research Funds for the Central Universities (SCUT 2009ZZ0052) and National Nature Science Foundations of China 60603022 and 61070091. Cornelia Fermüller gratefully acknowledges the support of the European Union under the Cognitive Systems Program (project POETICON++) and the National Science Foundation under the Cyberphysical Systems Program.

Appendix A. Wavelet frame system

Instead of directly using the MFS vector as the texture descriptor, we decompose it under a shift-invariant wavelet frame system and only take the leading wavelet coefficients (coefficients with large magnitude). The reason for doing so is to further increase the robustness of the resulting texture descriptor by removing insignificant coefficients which are sensitive to environmental changes. In this section, we give a brief review on wavelet frame systems. For an in-depth theoretical analysis and practical implementation, see for example [2,8].

A wavelet frame system is a redundant system that generalizes the orthonormal wavelet basis (see [8] for more details). Wavelet tight frames have greater flexibility than orthonormal bases by sacrificing orthonormality and linear independence, but they have the same efficient decomposition and reconstruction algorithms as orthonormal wavelet bases. The filters used in wavelet frame systems have many attractive properties, not present in those used in orthonormal wavelet systems: *e.g.*, symmetry (anti-symmetry), smoothness,and shorter support. These nice properties make wavelet frame systems ideal for building a descriptors with strong robustness.

An MRA-based wavelet frame system is based on a single scaling function $\phi \in L^2(\mathbb{R})$ and several wavelet functions $\{\psi_1, \ldots, \psi_r\} \subset L^2(\mathbb{R})$ that satisfy the following refinable equation:

$$\phi(t) = \sqrt{2}\sum_k h_0(k)\phi(2t-k); \psi_\ell(t) = \sqrt{2}\sum_k h_\ell(k)\phi(2t-k), \ell = 1, 2, \ldots, r.$$

Let $\phi_k(t) = \phi(t-k)$ and $\psi_{k,j,\ell} = \psi_\ell(2^j t - k)$. Then for any square integrable function $f \in L^2(\mathbb{R})$, we have a multi-scale representation of f as follows:

$$f = \sum_{k=-\infty}^{\infty} c_k\phi_k(t) + \sum_{\ell=1}^{r}\sum_{j=0}^{\infty}\sum_{k=-\infty}^{\infty} d_{k,j,\ell}\psi_{k,j,\ell}, \tag{A.1}$$

where $c_k = \int_{\mathbb{R}} f(t)\phi_k(t)dt$ and $d_{k,j,\ell} = \int_{\mathbb{R}} f(t)\psi_{k,j,\ell}(t)dt$. Eq. (A.1) is called the perfect reconstruction property of wavelet tight frames. The coefficients $\{c_k\}$ and $\{d_{k,j,\ell}\}$ are called low-pass and high-pass wavelet coefficients respectively. The wavelet coefficients can be efficiently calculated by a so-called cascade algorithm (see e.g. [29]). In this paper, we use the piece-wise linear wavelet frame developed in [8]:

$$h_0 = \frac{1}{4}[1,2,1]; \quad h_1 = \frac{\sqrt{2}}{4}[1,0,-1]; \quad h_2 = \frac{1}{4}[-1,2,-1].$$

See Fig. A.20 for the corresponding ϕ and ψ_1, ψ_2. We follow [2] for a discrete implementation of the multi-scale tight frame decomposition without downsampling. For convenience of notation, we denote such a linear frame decomposition by a rectangular matrix A of size $m \times n$ with $m > n$. Thus, given any signal $\mathbf{f} \in \mathbb{R}^n$, the discrete version of (A.1) is expressed as follows:

$$\mathbf{f} = A^T\mathbf{w} = A^T(A\mathbf{f}),$$

where $\mathbf{w} \in \mathbb{R}^m$ is the wavelet coefficient vector of $\mathbf{f}$. It is noted that we have $A^TA = I$ but $AA^T \neq I$ unless the tight framelet system degenerates to an orthonormal wavelet system.

References

[1] J. Aloimonos, Shape from texture, Biological Cybernetics 58 (1988) 345–360.

[2] J. Cai, R.H. Chan, Z. Shen, A framelet-based image inpainting algorithm, Applied and Computational Harmonic Analysis 24 (2) (2008) 131–149.

[3] A. Chan, N. Vasconcelos, Classifying video with kernel dynamic textures, CVPR (2007).

[4] Y.W. Chen, C.J. Lin, Combining SVMs with Various Feature Selection Strategies, Feature Extraction, Foundations and Applications, Springer, 2006.

[5] D. Chetverikov, R. Péteri, A brief survey of dynamic texture description and recognition, in: Proc. 4th Int. Conf. on Computer Recognition Systems, Springer Advances in Soft Computing, 2005, pp. 17–26.

[6] A. Conci, L.H. Monterio, Multifractal characterization of texture-based segmentation, ICIP (2000) 792–795.

[7] K. Dana, S. Nayar, Histogram model for 3d textures, CVPR (1998) 618–624.

[8] I. Daubechies, B. Han, A. Ron, Z. Shen, Framelets: MRA-based constructions of wavelet frames, Applied and Computational Harmonic Analysis 14 (2003) 1–46.

[9] K.G. Derpanis, R.P. Wildes, Dynamic texture recognition based on distributions of spacetime oriented structure, CVPR (2010).

[10] G. Doretto, A. Chiuso, Y.N. Wu, S. Soatto, Dynamic texture, IJCV (2003).

[11] A. Efros, T. Leung, Texture synthesis by non-parametric sampling, ICCV (1999) 1039–1046.

[12] K.J. Falconer, Techniques in Fractal Geometry, John Wiley, 1997.

[13] S. Fazekas, D. Chetverikov, Analysis and performance evaluation of optical flow features for dynamic texture recognition, in: Signal Processing: Image Communication, Special Issue on Content-Based Multimedia Indexing and Retrieval, vol. 22 (7–8), 2007, pp. 680–691,

[14] S. Fazekas, D. Chetverikov, Normal versus complete flow in dynamic texture recognition: a comparative study, in: Workshop on Texture Analysis and Synthesis, 2005.

[15] D.A. Forsyth, J. Ponce, Computer Vision: A Modern Approach, Prentice Hall, 2002.

[16] J. Garding, T. Lindeberg, Direct computation of shape cues using scale-adapted spatial derivative operators, IJCV 17 (2) (1996) 163–191.

[17] B. Ghanem, N. Ahuja, Maximum margin distance learning for dynamic texture recognition, ECCV (2010).

[18] B. Ghanem, N. Ahuja, Phase based modelling of dynamic textures, ICCV (2007).

[19] E. Hayman, B. Caputo, M. Fritz, J.O. Eklundh, On the significance of real-world conditions for material classification, ECCV (2004) 253–266.

[20] D.J. Heeger, J.R. Bergen, Pyramid based texture analysis/synthesis, Computer Graphics Proceedings (1995) 229–238.

[21] B. Julesz, Texture and visual perception, Science America 212 (1965) 38–48.

[22] C. Kervrann, F. Heitz, A Markov random field model-based approach to unsupervised texture segmentation using local and global spatial statistics, IEEE Transactions on Image Processing 4 (6) (1995) 856–862.

[23] S.M. Konishi, A.L. Yuille, Statistical cues for domain specific image segmentation with performance analysis, CVPR (2000) 125–1132.

[24] S. Lazebnik, Local semi-local and global models for texture, object and scene recognition, Ph.D. Dissertation, University of Illinois at Urbana-Champaign, 2006.

[25] T. Leung, J. Malik, Representing and recognizing the visual appearance of materials using three-dimensional texons, IJCV 43 (1) (2001) 29–44.

[26] T. Lindeberg, Automatic scale selection as a pre-processing stage for interpreting the visual world, FSPIPA 130 (1999) 9–23.

[27] D. Lowe, Distinctive image features from scale invariant keypoints, IJCV 60 (2) (2004) 91–110.

[28] J. Malik, R. Rosenholtz, Computing local surface orientation and shape from texture for curved surfaces, IJCV 23 (2) (1997) 49–168.

[29] S. Mallat, A Wavelet Tour of Singapore Processing, third ed., The Sparse Way, Academic Press, 2008.

[30] B.B. Mandelbrot, The Fractal Geometry of Nature, Freeman, San Francisco, CA, 1982.

[31] B.S. Manjunath, J.R. Ohm, V.V. Vasudevan, A. Yamada, Color and texture descriptors, IEEE Transactions on Circuits and Systems for Video Technology 11 (6) (2001) 703–715.

[32] F. Mindru, T. Tuytelaars, L. Van Gool, T. Moons, Moment invariants for recognition under changing viewpoint and illumination, CVIU 94 (1–3) (2004) 3–27.

[33] R.C. Nelson, R. Polana, Qualitative recognition of motion using temporal texture, Computer Vision, Graphics, and Image Processing: Image Understanding 56 (1) (1992) 78–89.

[34] R. Péteri, S. Fazekas, M.J. Huiskes, DynTex: a comprehensive database of dynamic texture, Pattern Recognition Letters 31 (12) (2010) 1627–1632.

[35] M. Pontil, A. Verri, Support vector machines for 3D object recognition, PAMI 20 (6) (1998) 637–646.

[36] J. Portilla, E.P. Simoncelli, A parametric texture model based on joint statistics of complex wavelet coefficients, IJCV 40 (1) (2000) 49–71.

[37] A. Ravichandran, R. Chaudhry, R. Vidal, View-invariant dynamic texture recognition using a bag of dynamical systems, CVPR (2009).

[38] P. Saisan, G. Doretto, Y. Wu, S. Soatto, Dynamic texture recognition, CVPR II (2001) 58–63.

[39] B. Scholkopf, A. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond, MIT Press, Cambridge, MA, 2002.

[40] P. Scovanner, S. Ali, M. Shah, A 3-dimensional SIFT descriptor and its application to action recognition, ACM Multimedia (2007).

[41] V. Tresp, A. Schwaighofer, Scalable kernel systems, in: Proceedings of ICANN 2001, Lecture Notes in Computer Science, vol. 2130, Springer Verlag, 2001, pp. 285–291.

[42] M. Varma, R. Garg, Locally invariant fractal features for statistical texture classification, ICCV (2007).

[43] M. Varma, A. Zisserman, Classifying images of materials: achieving viewpoint and illumination independence, ECCV (2002) 255–271.

[44] R.P. Wildes, S.R. Bergen, Qualitative spatiotemporal analysis using an oriented energy representation, ECCV (2000) 768–784.

[45] F. Woolfe, A. Fitzgibbon, Shift-invariant dynamic texture recognition, ECCV II (2006) 549–562.

[46] J. Wu, M.J. Chantler, Combining gradient and albedo for rotation invariant classification of 2D surface texture, ICCV (2003) 848–855.

[47] Y. Xu, H. Ji, C. Fermuller, Viewpoint invariant texture description using fractal analysis, IJCV 83 (1) (2009) 85–100.

[48] Y. Xu, X. Yang, H.B. Ling, H. Ji, A new texture descriptor using multifractal analysis in multi-orientation wavelet pyramid, CVPR (2010).

[49] J. Zhang, M. Marszalek, S. Lazebnik, C. Schmid, Local features and kernels for classification of texture and object categories: a comprehensive study, IJCV 73 (2) (2007) 213–238.

[50] G. Zhao, M. Pietikäinen, Dynamic texture recognition using local binary patterns with an application to facial expression, PAMI 29 (6) (2007) 915–928.

[51] S.C. Zhu, Y. Wu, D. Mumford, Filters, random fields and maximum entropy (FRAME): towards a unified theory for texture modeling, IJCV 27 (2) (1998) 107–126.

[52] http://www.cfar.umd.edu/ fer/website-texture/texture.htm