Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2008, Article ID 691924, 15 pages
doi:10.1155/2008/691924

Research Article

Learning How to Extract Rotation-Invariant and Scale-Invariant Features from Texture Images

Javier A. Montoya-Zegarra,[1,2] João Paulo Papa,[2] Neucimar J. Leite,[2] Ricardo da Silva Torres,[2] and Alexandre X. Falcão[2]

[1] *Computer Engineering Department, Faculty of Engineering, San Pablo Catholic University, Av. Salaverry 301, Vallecito, Arequipa, Peru*
[2] *Institute of Computing, The State University of Campinas, 13083-970 Campinas, SP, Brazil*

Correspondence should be addressed to Javier A. Montoya-Zegarra, jmontoyaz@gmail.com

Received 2 October 2007; Revised 1 January 2008; Accepted 7 March 2008

Recommended by C. Charrier

Learning how to extract texture features from noncontrolled environments characterized by distorted images is a still-open task. By using a new rotation-invariant and scale-invariant image descriptor based on steerable pyramid decomposition, and a novel multiclass recognition method based on optimum-path forest, a new texture recognition system is proposed. By combining the discriminating power of our image descriptor and classifier, our system uses small-size feature vectors to characterize texture images without compromising overall classification rates. State-of-the-art recognition results are further presented on the Brodatz data set. High classification rates demonstrate the superiority of the proposed system.

1. INTRODUCTION

An important low-level image feature used in human perception as well as in recognition is texture. In fact, the study of texture has found several applications ranging from texture segmentation [1] to texture classification [2], synthesis [3, 4], and image retrieval [5, 6].

Although various authors have attempted to define what texture is [7, 8], there still does not exist a commonly accepted definition. However, the basic property presented in every texture consists in a small elementary pattern repeated periodically or quasiperiodically in a given region (pixel neighborhood) [9, 10]. The repetition of those image patterns generates some visual cues, which can be identified, for example, as being directional or nondirectional, smooth or rough, coarse or fine, uniform or nonuniform [11, 12]. Figures 1–4 show some examples of these types of visual cues. Note that each texture can be associated with one or more visual cues.

Further, texture images are typically classified as being either natural or artificial. Natural textures are related to nonman-made objects and among others they include, for example, brick, grass, sand, and wood patterns. On the other side, artificial textures are related to man-made objects such as architectural, fabric, and metal patterns.

Regardless of its classification type, texture images may be characterized by their variations in scale or directionality. Scale variations imply that textures may look quite different when varying the number of scales. This effect is analogous to increase or decrease the image resolution. The bigger or the smaller the scales are, the more different the images are. This characteristic is related to the *coarseness* presented in texture images and can be understood as the spatial repetition period of the local pattern [13]. Finer texture images are characterized by small repetition periods, whereas coarse textures present larger repetition periods. In addition, oriented textures may present different principal directions as the images rotate. This happens because textures are not always captured from the same viewpoint.

On the other hand, work on texture characterization can be divided into four major categories [1, 14]: structural, statistical, model-based, and spectral. For structural methods, texture images can be thought as being a set of primitives with geometrical properties. Their objective is therefore to find the primitive elements as well as the formal rules of their spatial placements. Example of this kind of

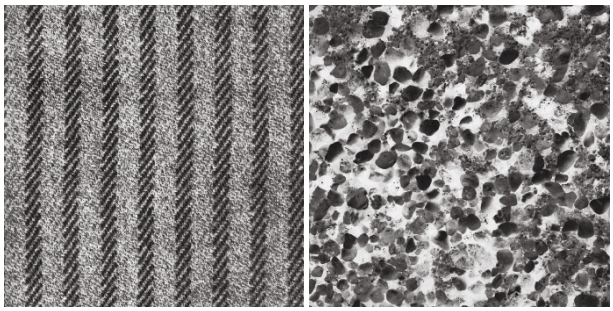

Figure 1: Directional versus nondirectional visual cues.

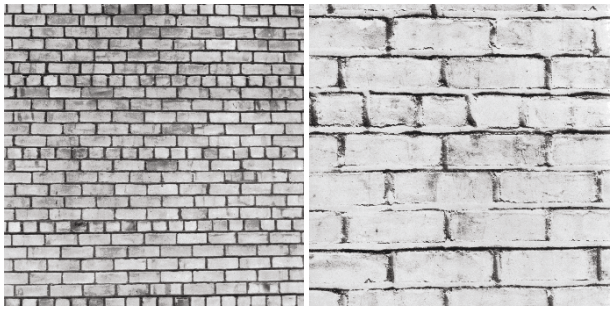

Figure 2: Smooth versus rough visual cues.

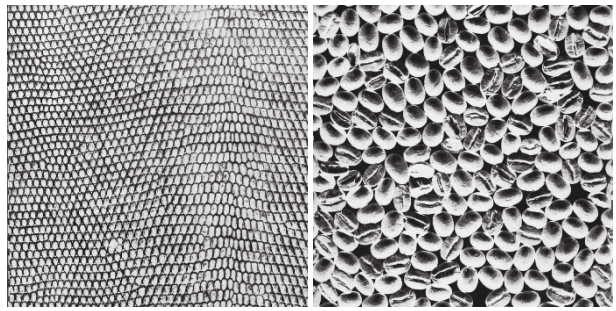

Figure 3: Fine versus coarse visual cues.

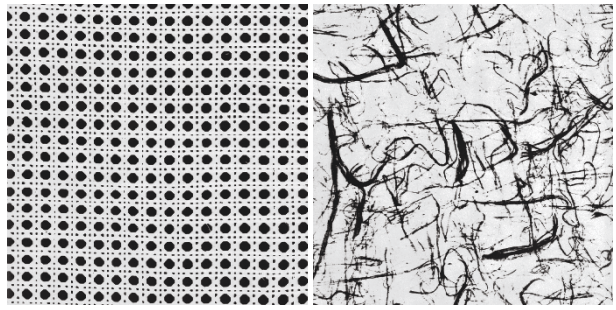

Figure 4: Uniform versus nonuniform visual cues.

methods can be found in the works of Julesz [15] and Tüceryan [16]. In addition, statistical methods study the spatial gray-level distribution in the textural patterns, so that statistical operations can be performed in the distributions of the local features computed at each pixel in the image. Statistical methods include among others the gray-level co-occurrence matrix [17], second-order spatial averages, and the autocorrelation function [18]. Further, the objective of the model-based methods is to capture the process that generated the texture patterns. Popular approaches in this category include Markov random fields [19, 20], fractal [21], and autoregressive models [22]. Finally, spectral methods perform frequency analysis in the image signals to reveal specific features. Examples of this may include Law's [23, 24] and Gabor's filters [25].

Although many of these techniques obtained good results, most of them have not been widely evaluated in *noncontrolled* environments, which may be characterized by texture images having (1) small interclass variations, that is, textures belonging to different classes may appear quite similar, especially in terms of their global patterns (coarseness, smoothness, etc.) and the patterns may present (2) image distortions such as rotations or scales. In this sense, texture pattern recognition is a still-open task. The next challenge in texture classification should be, therefore, to achieve rotation-invariant and scale-invariant feature representations for *noncontrolled* environments.

Some of these challenges are faced in this work. More specifically, we focus on feature representation and recognition. In feature representation, we wish to emphasize some open questions, such as how to model the texture images so that the relevant information is captured despite of the image distortions, and how to keep low-dimensional feature vectors so that texture recognition applications are facilitated, where data storage capacity is a limitation. In feature recognition, we wish to choose a technique that handles multiple nonseparable classes with minimal computational time and supervision. To deal with the challenges in feature extraction, we propose a new texture image descriptor based on steerable pyramid decomposition, which encodes the relevant texture information in small-size feature vectors including rotation-invariant and scale-invariant characterizations. To address the feature recognition requirements, we are using a novel multiclass object recognition method based on the optimum-path forest [26].

Roughly speaking, a steerable pyramid is a method by which images are decomposed into a set of multiscale, and multiorientation image subbands, where the basis functions are directional derivative operators [27]. Our motivation in using steerable pyramids relies on that, unlike other image decomposition methods, the feature coefficients are less affected by image distortions. Furthermore, the optimum-path forest classifier is a recent approach that handles nonseparable classes without the necessity of using boosting procedures to increase its performance, resulting thus in a faster and more accurate classifier for object recognition. By combining the discriminating power of our image descriptor and classifier, our system uses small-size feature vectors to characterize texture images without compromising overall classification rates. In this way, texture classification applications, where data storage capacity is a limitation, are further facilitated.

A previous version of our texture descriptor has been proposed for texture recognition, using only rotation-

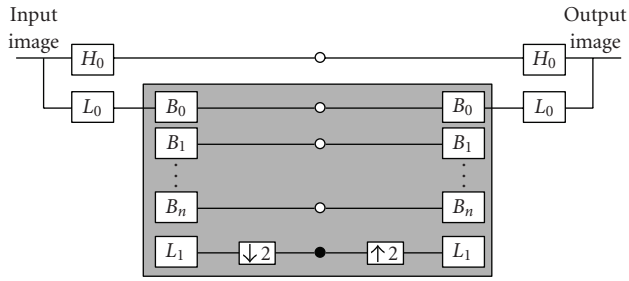

Figure 5: First-level steerable pyramid decomposition using n oriented bandpass filters.

invariant properties [28]. In the present work, the proposed descriptor has not only rotation-invariant properties, but also scale-invariant properties. The descriptor with both properties was previously evaluated for content-based image retrieval [29], but this is the first time it is being demonstrated for texture recognition. The optimum-path forest classifier was first presented in [30] and first evaluated for texture recognition in [28]. Improvements in its learning algorithm and evaluation with several data sets have been made in [26] for other properties rather than texture. The present work is using this most recent version of the optimum-path forest classifier for texture recognition. We are providing more details about the methods, more data sets, and a more in deep analysis of the results: rotation- and scale-invariance analyses, accuracy of classification with different descriptors, and the mean computational time of the proposed system.

The outline of this work is as follows. In Section 2, we briefly review the fundamentals of the steerable pyramid decomposition. Section 3 describes how texture images are characterized to obtain rotation-invariant and scale-invariant representations. Section 4 describes the optimum-path forest classifier method. The experimental setup conducted in our study is presented in Section 5. In Section 6, experimental results on several data sets are presented and used to demonstrate the recognition accuracy of our system. Comparisons with state-of-the-art texture feature representations and classifiers are further discussed. Finally, some conclusions are drawn in Section 7.

2. STEERABLE PYRAMID DECOMPOSITION

The steerable pyramid decomposition is a linear multiresolution image decomposition method by which an image is subdivided into a collection of subbands localized at different scales and orientations [27]. Using a high- and low-pass filter (H_0, L_0) the input image is initially decomposed into two subbands: a high- and a low-pass subband, respectively. Further, the low-pass subband is decomposed into K-oriented band-pass portions $B_0, \ldots, B_{K-1}$, and into a low-pass subband L_1. The decomposition is done recursively by subsampling the lower low-pass subband (L_S) by a factor of 2 along the rows and columns. Each recursive step captures different directional information at a given scale. Considering the polar-separability of the filters in the Fourier domain, the first low- and high-pass filters, are defined as [31]

$$L_0(r) = \frac{L(r/2)}{2}, \qquad H_0(r) = H\left(\frac{r}{2}\right), \tag{1}$$

where r, θ are the polar frequency coordinates. The raised cosine low- and high-pass transfer functions denoted as L, H, respectively, are computed as follows:

$$L(r) = \begin{cases} 2 & r \le \frac{\pi}{4}, \\ 2\cos\left(\frac{\pi}{2}\log_2\left(\frac{4r}{\pi}\right)\right) & \frac{\pi}{4} < r < \frac{\pi}{2}, \\ 0 & r \ge \frac{\pi}{2}, \end{cases} \tag{2}$$

$$B_k(r,\theta) = H(r)G_k(\theta), \quad k \in [0, K-1].$$

$B_k(r,\theta)$ represents the kth directional bandpass filter used in the iterative stages, with radial and angular parameters, defined as

$$H(r) = \begin{cases} 1 & r \ge \frac{\pi}{2}, \\ \cos\left(\frac{\pi}{2}\log_2\left(\frac{2r}{\pi}\right)\right) & \frac{\pi}{4} < r < \frac{\pi}{2}, \\ 0 & r \le \frac{\pi}{4} \end{cases} \tag{3}$$

$$G_k(\theta) = \begin{cases} \alpha_K \left(cos\left(\theta - \frac{\pi k}{K}\right)\right)^{K-1} & \left|\theta - \frac{\pi k}{K}\right| < \frac{\pi}{2}, \\ 0 & \text{otherwise}, \end{cases}$$

where $\alpha_k = 2^{(k-1)}((K-1)!/\sqrt{K[2(K-1)]!})$.

Figure 5 depicts a steerable pyramid decomposition using only one scale and n orientations.

3. TEXTURE FEATURE REPRESENTATION

This section describes the proposed modification of steerable pyramid decomposition to obtain rotation-invariant and scale-invariant representations, used further to characterize the texture images.

3.1. Texture representation

Roughly speaking, texture images can be seen as a set of basic repetitive primitives characterized by their spatial homogeneity [32]. By applying statistical measures, this information is extracted and used to capture the relevant image content into feature vectors. More precisely, by considering the presence of homogeneous regions in texture images, we use the mean (μ_{mn}) and standard deviation (σ_{mn}) of the energy distribution of the filtered images (S_{mn}). Given an image $I(x,y)$, its steerable pyramid decomposition is defined as

$$S_{mn}(x,y) = \sum_{x_1}\sum_{y_1} I(x_1,y_1)B_{mn}(x-x_1, y-y_1), \tag{4}$$

where B_{mn} denotes the directional bandpass filters at stage $m = 0, 1, \ldots, S-1$ and orientation $n = 0, 1, \ldots, K-1$. The energy distribution ($E(m,n)$) of the filtered images at scale m and at orientation n is defined as

$$E(m,n) = \sum_x \sum_y |S_{mn}(x,y)|. \tag{5}$$

Additionally, the mean (μ_{mn}) and standard deviation (σ_{mn}) of the energy distributions are found as follows:

$$\mu_{mn} = \frac{1}{MN} E_{mn}(x,y),$$
$$\sigma_{mn} = \sqrt{\frac{1}{MN} \sum_x \sum_y (|S_{mn}(x,y)| - \mu_{mn})^2}, \tag{6}$$

where M and N denote the height and width of the input image, respectively. The corresponding feature vector ($\vec{f}$) is defined by using the mean and standard deviation as feature elements. It is denoted as

$$\vec{f} = [\mu_{00}, \sigma_{00}, \mu_{01}, \sigma_{01}, \ldots, \mu_{S-1K-1}, \sigma_{S-1K-1}]. \tag{7}$$

The dimensionality of the feature vectors depends on the number of scales (S) and on the number of orientations (K) considered during image decomposition. The feature vector dimensionality is computed by multiplying the number of scales and orientations by factor of 2 ($2 \times S \times K$). This factor corresponds to the mean and standard deviation computed in each filtered image.

3.2. Rotation-invariant representation

Rotation-invariant representation is achieved by computing the dominant orientation of the texture images followed by feature alignment. The *dominant orientation* (DO) is defined as the orientation with the highest total energy across the different scales considered during image decomposition [33]. It is computed by finding the highest accumulated energy for the K different orientations considered during image decomposition:

$$\mathrm{DO}_i = \max\left\{E_0^{(R)}, E_1^{(R)}, \ldots, E_{K-1}^{(R)}\right\}, \tag{8}$$

where i is the index where the dominant orientation appeared and

$$E_n^{(R)} = \sum_{m=0}^{S-1} E(m,n), \quad n = 0, 1, \ldots, K-1. \tag{9}$$

Note that each $E_n^{(R)}$ covers a set of filtered images at different scales but at same orientation.

Finally, rotation-invariance is obtained by shifting circularly feature elements within the same scales, so that first elements at each scale correspond to dominant orientations. As an example, let $\vec{f}$ be a feature vector obtained by using a pyramid decomposition with $S = 2$ scales and $K = 3$ orientations:

$$\vec{f} = [\mu_{00}, \sigma_{00}, \mu_{01}, \sigma_{01}, \mu_{02}, \sigma_{02}; \mu_{10}, \sigma_{10}, \mu_{11}, \sigma_{11}, \mu_{12}, \sigma_{12}]. \tag{10}$$

Now suppose that the dominant orientation appears at index $i = 1$ ($\mathrm{DO}_{i=1}$), thus the rotation-invariant feature vector, after feature alignment, is represented as follows:

$$\vec{f}^R = [\mu_{01}, \sigma_{01}, \mu_{02}, \sigma_{02}, \mu_{00}, \sigma_{00}; \mu_{11}, \sigma_{11}, \mu_{12}, \sigma_{12}, \mu_{10}, \sigma_{10}]. \tag{11}$$

3.3. Scale-invariant representation

Similarly, scale-invariant representation is achieved by finding the scale with the highest total energy across the different orientations (*dominant scale*). For this purpose, the dominant scale (DS) at index i is computed as follows:

$$\mathrm{DS}_i = \max\left\{E_0^{(S)}, E_1^{(S)}, \ldots, E_{S-1}^{(S)}\right\}, \tag{12}$$

where $E_m^{(S)}$ denotes the accumulated energies across the S different scales:

$$E_m^{(S)} = \sum_{n=0}^{K-1} E(m,n), \quad m = 0, 1, \ldots, S-1. \tag{13}$$

Note that each $E_m^{(S)}$ covers a set of filtered images at different orientations for each scale. As an example, let $\vec{f}$ be, again, the feature vector obtained by using a pyramid decomposition with $S = 2$ scales and $K = 3$ orientations:

$$\vec{f} = [\mu_{00}, \sigma_{00}, \mu_{01}, \sigma_{01}, \mu_{02}, \sigma_{02}\mu_{10}, \sigma_{10}, \mu_{11}, \sigma_{11}, \mu_{12}, \sigma_{12}]. \tag{14}$$

By supposing that the dominant scale was found at index $i = 1$ (second scale in the image decomposition), its scale-invariant version, after feature alignment, is defined as

$$\vec{f}^S = [\mu_{10}, \sigma_{10}, \mu_{11}, \sigma_{11}, \mu_{12}, \sigma_{12}; \mu_{00}, \sigma_{00}, \mu_{01}, \sigma_{01}, \mu_{02}, \sigma_{02}]. \tag{15}$$

For both rotation-invariant and scale-invariant representations, the feature alignment process is based on the assumption that to classify textures, images should be rotated so that their dominant orientations/scales are the same. Further, it has been proved that image rotation in spatial domain is equivalent to circular shift of feature vector elements [34].

4. TEXTURE FEATURE RECOGNITION

This section aims to describe the most recent version of the optimum-path forest (OPF) classifier [26], which is an important part of the texture recognition system proposed in this work. Previous works have demonstrated that OPF can be more effective and much faster than artificial neural networks [35] and support vector machines [26, 30, 35]. The OPF approach works by modeling the patterns as being nodes of a graph in the feature space, where every pair of

INPUT: *A λ -labeled training set Z_1, prototypes $\Omega \subset Z_1$ and the pair (v, d) for feature vector and distance computations.*
OUTPUT: *Optimum-path forest P, cost map C and label map L.*
AUXILIARY: *Priority queue Q and cost variable* cst.

1. For each $s \in Z_1 \setminus \Omega$, set $C(s) \leftarrow +\infty$.
2. For each $s \in \Omega$, do
3. $\quad C(s) \leftarrow 0$, $P(s) \leftarrow$ nil, $L(s) \leftarrow \lambda(s)$, and insert s in Q.
4. While Q is not empty, do
5. $\quad$ Remove from Q a sample s such that $C(s)$ is minimum.
6. $\quad$ For each $t \in Z_1$ such that $t \neq s$ and $C(t) > C(s)$, do
7. $\quad\quad$ Compute cst $\leftarrow \max\{C(s), d(s,t)\}$.
8. $\quad\quad$ If cst $< C(t)$, then
9. $\quad\quad\quad$ If $C(t) \neq +\infty$, then remove t from Q.
10. $\quad\quad\quad$ $P(t) \leftarrow s$, $L(t) \leftarrow L(s)$, $C(t) \leftarrow$ cst,
11. $\quad\quad\quad$ and insert t in Q.

Algorithm 1: OPF algorithm.

nodes is connected by an arc (complete graph). This classifier creates a discrete optimal partition of the feature space such that any unknown sample can be classified according to this partition. This partition is an optimum-path forest computed in $\mathbb{R}^n$ by the image foresting transform (IFT) algorithm [36]. The OPF classifier extends the IFT from the image domain to the feature space, where the samples may be images, contours, or any other abstract entities.

Let Z_1, Z_2, and Z_3 be, respectively, the training, evaluation, and test sets with $|Z_1|$, $|Z_2|$, and $|Z_3|$ samples, respectively. Let $\lambda(s)$ be the function that assigns the correct label i, $i = 1, 2, \ldots, c$, from class i to any sample $s \in Z_1 \cup Z_2 \cup Z_3$. Z_1 and Z_2 are labeled sets used to the design of the classifier and the unseen set Z_3 is used to compute the final accuracy of the classifier. Let $\Omega \subset Z_1$ be a set of prototypes of all classes (i.e., key samples that best represent the classes). Let v be an algorithm which extracts n attributes (texture properties) from any sample $s \in Z_1 \cup Z_2 \cup Z_3$ and returns a vector $\vec{v}(s) \in \mathbb{R}^n$. The distance $d(s,t)$ between two samples, s and t, is the one between their feature vectors $\vec{v}(s)$ and $\vec{v}(t)$ (e.g., Euclidean or any other valid metric).

Let (Z_1, A) be a complete graph whose the nodes are the samples in Z_1. We define a path as being a sequence of distinct samples $\pi = \langle s_1, s_2, \ldots, s_k \rangle$, where $(s_i, s_{i+1}) \in A$ for $1 \leq i \leq k-1$. A path is said to be *trivial* if $\pi = \langle s_1 \rangle$. We assign to each path π a cost $f(\pi)$ given by a path-cost function f. A path π is said to be optimum if $f(\pi) \leq f(\pi')$ for any other path π', where π and π' end at the same sample s_k. We also denote by $\pi \cdot \langle s,t \rangle$ the concatenation of a path π with terminus at s and an arc (s,t). The OPF algorithm uses the path-cost function $f_{\max}$, for the reason explained in Section 4.1,

$$f_{\max}(\langle s \rangle) = \begin{cases} 0 & \text{if } s \in \Omega, \\ +\infty & \text{otherwise,} \end{cases} \tag{16}$$

$$f_{\max}(\pi \cdot \langle s,t \rangle) = \max\{f_{\max}(\pi), d(s,t)\}.$$

We can observe that $f_{\max}(\pi)$ computes the maximum distance between adjacent samples in π, when π is not a trivial path.

The OPF algorithm assigns one optimum path $P^*(s)$ from Ω to every sample $s \in Z_1$, forming an optimum-path forest P (a function with no cycles which assigns to each $s \in Z_1 \setminus \Omega$, its predecessor $P(s)$ in $P^*(s)$, or a marker *nil* when $s \in \Omega$). Let $R(s) \in \Omega$ be the root of $P^*(s)$ which can be reached from $P(s)$. The OPF algorithm computes for each $s \in Z_1$, the cost $C(s)$ of $P^*(s)$, the label $L(s) = \lambda(R(s))$, and the predecessor $P(s)$, as follows.

Lines 1–3 initialize maps and insert prototypes in Q. The main loop computes an optimum path from Ω to every sample s in a nondecreasing order of cost (lines 4–11). At each iteration, a path of minimum cost $C(s)$ is obtained in P when we remove its last node s from Q (line 5). Lines 8–11 evaluate if the path that reaches an adjacent node t through s is cheaper than the current path with terminus t and update the position of t in Q, $C(t)$, $L(t)$, and $P(t)$ accordingly. The label $L(s)$ may be different from $\lambda(s)$, leading to classification errors in Z_1. The training finds prototypes with zero classification errors in Z_1, as follows.

4.1. Training phase

We say that Ω^* is an optimum set of prototypes when Algorithm 1 propagates the labels $L(s) = \lambda(s)$ for every $s \in Z_1 \cdot \Omega^*$ can be found by exploiting the theoretical relation between minimum-spanning tree (MST) and optimum-path tree for $f_{\max}$ [37]. The training essentially consists of finding Ω^* and an OPF classifier rooted at Ω^*.

By computing an MST in the complete graph (Z_1, A), we obtain a connected acyclic graph whose nodes are all samples in Z_1 and the arcs are undirected and weighted by the distance d between the adjacent sample feature vectors. This spanning tree is optimum in the sense that the sum

INPUT: *Training and evaluation sets, Z_1 and Z_2, labeled by λ, number T of iterations, and the pair (v, d) for feature vector and distance computations.*
OUTPUT: *Learning curve $\mathcal{L}$ and the best OPF classifier, represented by the predecessor map P, cost map C, and label map L.*
AUXILIARY: *False positive and false negative arrays,* FP *and* FN, *of sizes c, and list* LM *of misclassified samples.*

1. For each iteration $I = 1, 2, \ldots, T$, do
2. LM $\leftarrow \varnothing$
3. Compute $\Omega^* \subset Z_1$ as in Section 4.1 and P, L, C
4. by Algorithm 1.
5. For each class $i = 1, 2, \ldots, c$, do
6. FP$(i) \leftarrow 0$ and FN$(i) \leftarrow 0$.
7. For each sample $t \in Z_2$, do
8. Find $s^* \in Z_1$ that satisfies (17).
9. If $L(s^*) \neq \lambda(t)$, then
10. FP$(L(s^*)) \leftarrow$ FP$(L(s^*)) + 1$.
11. FN$(\lambda(t)) \leftarrow$ FN$(\lambda(t)) + 1$.
12. LM $\leftarrow$ LM $\cup\, t$.
13. Compute $\mathcal{L}(I)$ by (20) and save P, L, and C.
14. While LM $\neq \varnothing$
15. LM $\leftarrow$ LM $\setminus\, t$
16. Replace t by randomly objects of the same class
17. in Z_1, except the prototypes.
18. Select the instance P, L, C of highest accuracy.

Algorithm 2: Learning algorithm.

of its arc weights is minimum as compared to any other spanning tree in the complete graph. In the MST, every pair of samples is connected by a single path which is optimum according to $f_{\max}$. The optimum prototypes are the closest elements in the MST with different labels in Z_1. By removing the arcs between different classes, their adjacent samples become prototypes in Ω^* and Algorithm 1 can compute an optimum-path forest with zero classification errors in Z_1.

It is not difficult to see that the optimum paths between classes should pass through the same removed arcs of the minimum-spanning tree. The choice of prototypes as described above blocks these passages, avoiding samples of any given class to be reached by optimum paths from prototypes of other classes. Given that several methods for graph-based clustering are based on MST, the relation between MST and minimum-cost path tree for $f_{\max}$ [37] makes interesting connections among the supervised OPF classifier, these unsupervised approaches, and the previous works on watershed-based/fuzzy-connected segmentations [36, 38–43].

4.2. Classification

For any sample $t \in Z_3$, the OPF considers all arcs connecting t with samples $s \in Z_1$, as if t was part of the graph. Considering all possible paths from Ω^* to t, we find the optimum path $P^*(t)$ from Ω^* and label t with the class $\lambda(R(t))$ of its most strongly connected prototype $R(t) \in \Omega^*$. This path can be identified incrementally by evaluating the optimum cost $C(t)$ as

$$C(t) = \min\{\max\{C(s), d(s,t)\}\}, \quad \forall s \in Z_1. \tag{17}$$

Let the node $s^* \in Z_1$ be the one that satisfies the above equation (i.e., the predecessor $P(t)$ in the optimum path $P^*(t)$). Given that $L(s^*) = \lambda(R(t))$, the classification simply assigns $L(s^*)$ to t. An error occurs when $L(s^*) \neq \lambda(t)$.

4.3. Learning algorithm

The performance of the OPF classifier improves when the closest samples from different classes are included in Z_1, given that the prototypes will come from them, working as sentinels on the boundaries between classes. On the other hand, the computational time and storage cost increase with the size of the training set. This section then describes how to improve the OPF performance without increasing the number of samples in Z_1.

Algorithm 2 is a simple learning procedure, but very effective. In each iteration, a set Z_1 is used for training and the classification is performed on Z_2. The best prototypes are assumed to be among the misclassified samples of Z_2. So, the algorithm randomly replaces misclassified samples of Z_2 by nonprototypes samples of Z_1, and training and classification are repeated during a few iterations. The algorithm outputs a *learning curve*, which reports the accuracy values of each

OPF's instance during learning, and the instance with the highest accuracy (which is usually the last one).

Lines 2–6 perform variable initialization and training on Z_1. The classification on Z_2 is performed in lines 7–12, updating the arrays of false positive and false negative for accuracy computation (line 13). Misclassified samples of Z_2 are stored in a list LM in line 12, and they are replaced by nonprototype samples of Z_1 in lines 14–17. The OPF instance with the highest accuracy is then selected in line 18.

The accuracy $\mathcal{L}(I)$ of a given iteration I, $I = 1, 2, \ldots, T$, is measured by taking into account that the classes may have different sizes in Z_2 (similar definition is applied for Z_3). Let $NZ_2(i)$, $i = 1, 2, \ldots, c$, be the number of samples in Z_2 from each class i. We define

$$e_{i,1} = \frac{\mathrm{FP}(i)}{|Z_2| - |NZ_2(i)|}, \qquad e_{i,2} = \frac{\mathrm{FN}(i)}{|NZ_2(i)|}, \quad i = 1, \ldots, c, \tag{18}$$

where $\mathrm{FP}(i)$ and $\mathrm{FN}(i)$ are the false positives and false negatives, respectively. That is, $\mathrm{FP}(i)$ is the number of samples from other classes that were classified as being from the class i in Z_2, and $\mathrm{FN}(i)$ is the number of samples from the class i that were incorrectly classified as being from other classes in Z_2. The errors $e_{i,1}$ and $e_{i,2}$ are used to define

$$E(i) = e_{i,1} + e_{i,2}, \tag{19}$$

where $E(i)$ is the partial sum error of class i. Finally, the accuracy $\mathcal{L}(I)$ of the classification is written as

$$\mathcal{L}(I) = \frac{2c - \sum_{i=1}^{c} E(i)}{2c} = 1 - \frac{\sum_{i=1}^{c} E(i)}{2c}. \tag{20}$$

5. EXPERIMENTS

5.1. Data sets

To evaluate the accuracy of our system, thirteen texture images obtained from the standard Brodatz database were selected. Before being digitized, each of the 512×512 texture images was rotated at different degrees [44]. Figure 6 displays the nonrotated version of each of the texture images.

From this database, three different image data sets were generated: *nondistorted*, *rotated-set*, and *scaled-set*. The nondistorted image data set was constructed from texture patterns at 0 degrees. Each texture image was partitioned into sixteen 128×128 nonoverlapping subimages. Thus, this data set comprises 208 (13×16) different images. Images belonging to this data set will be used in the learning stage of our classifier. Note that in previous works related to texture recognition [45, 46], rotated or scaled-versions of the patterns were included in both the training and classification phases [47]. However, more recently works suggest that the recognition algorithms should perform well, even by having during the training phase nondistorted training samples, which means patterns without rotations or scales [48, 49].

The second image data set referred to as *rotated-set* was generated to evaluate the rotation-invariance capabilities of our approach. It is further subdivided into two data sets: *rotated-set A* and *rotated-set B*. The *rotated image data set A* was generated by selecting the four 128×128 innermost subimages from texture images at 0, 30, 60, and 120 degrees. A total number of 208 images were generated ($13 \times 4 \times 4$). In addition, in the case of the *rotated image data set B*, we selected the four 128×128 innermost subimages of the rotated image textures (512×512) at 0, 30, 60, 90, 120, 150, and 200 degrees. This led to 364 ($13 \times 4 \times 7$) data set images. The first data set was initially used to test our system under the presence of few texture oriented images, whereas the second one was used to show how our systems performs by increasing the number of texture oriented images.

On the other side, the *scaled image data set* was partitioned into two data sets: *scaled-set A* and *scaled-set B*. In the *scaled-set A*, the 512×512 nonrotated textures were first partitioned into four 256×256 nonoverlapping subimages. Each partitioned subimage was further scaled by using four different factors, ranging from 0.6 to 0.9 with 0.1 interval. This led to 208 ($13 \times 4 \times 4$) scaled images. To generate the *scaled-set B*, each of the four partitioned subimages was scaled by using seven different factors, ranging from 0.6 to 1.2 with 0.1 interval. In this way, 364 ($13 \times 4 \times 7$) scaled images were generated.

5.2. Similarity measure for classification

Similarity between images is obtained by computing the distance of their corresponding feature vectors (recall Section 3). The smaller the distance, the more similar the images. Given the query image (i), and the target image (j) in the data set, the distance between the two patterns is defined as [50]

$$d(i, j) = \sum_{m} \sum_{n} d_{mn}(i, j), \tag{21}$$

where

$$d_{mn}(i, j) = \left| \frac{\mu_{mn}^{i} - \mu_{mn}^{j}}{\alpha(\mu_{mn})} \right| + \left| \frac{\sigma_{mn}^{i} - \sigma_{mn}^{j}}{\alpha(\sigma_{mn})} \right|, \tag{22}$$

$\alpha(\mu_{mn})$ and $\alpha(\sigma_{mn})$ denote the standard deviations of the respective features over the entire data set. They are used for feature normalization purposes.

6. EXPERIMENTAL RESULTS

Three series of experiments were conducted to demonstrate the discriminating power of our system for recognizing texture patterns. By considering that a recognition system is comprised of two mainly parts (feature extraction module as well as feature recognizer module), each of those parts was evaluated.

In the first series of experiments (Section 6.1), we first evaluated the effectiveness of the proposed rotation-invariant feature representation against two other approaches: the conventional pyramid decomposition [51] and with a recent proposal based on Gabor wavelets [33]. To evaluate the effectiveness of the feature recognizer module, we compared the

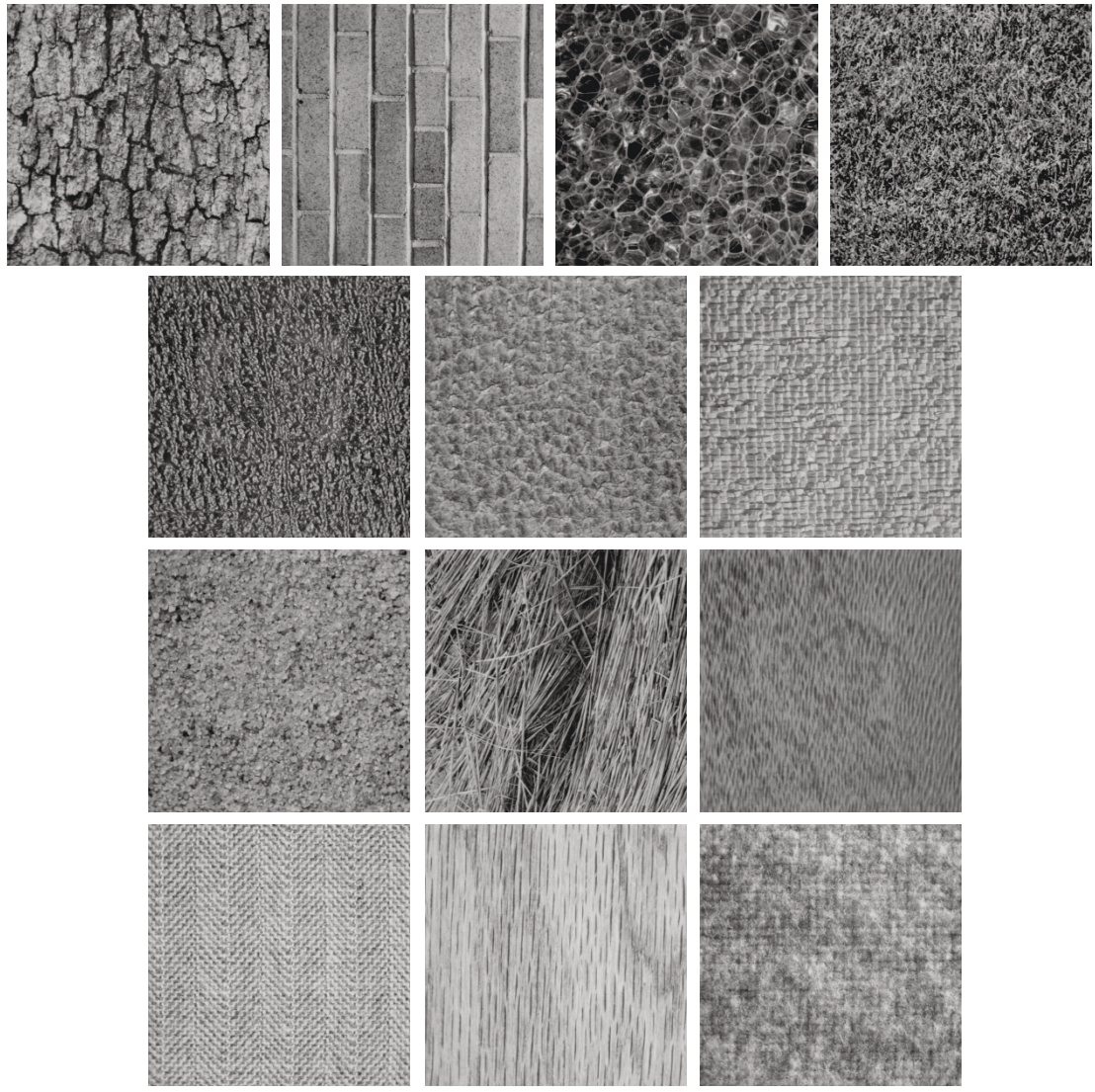

Figure 6: Texture images from the Brodatz data set used in our experiments. From left to right, and from top to bottom, they include Bark, Brick, Bubbles, Grass, Leather, Pigskin, Raffia, Sand, Straw, Water, Weave, Wood, and Wool.

recognition accuracy of the novel OPF multiclass classifier against the well-known support vector machines technique. For those purposes, we used the *rotated image data sets A* and *B*.

The second series of experiments (Section 6.2) are used to evaluate the scale-invariant properties in our feature extraction module. Effectiveness of the multiclass recognition method under the presence of scale-invariant features are further discussed. Again, we used the conventional steerable pyramid decomposition [51] and the Gabor wavelets [50] as references for comparing the scale-invariant properties of our method. SVMs are used for evaluating the classification accuracy of our feature recognizer module. Further, *scaled image data sets A* and *B* were used in this set of experiments.

In both series of experiments, we used steerable pyramids having different decomposition levels ($S = 2, 3, 4$) at several orientations ($K = 4, 5, 6, 7, 8$). Our experiments agree with [52] in that the most relevant textural information in images is contained in the first two levels of decomposition, since little recognition improvement is achieved by varying the number of scales during image decomposition. Therefore, we focus our discussions on image decompositions having ($S = 2, 3$) scales.

Given that the performance of the OPF classifier can increase using a third set in a learning algorithm (Section 6.3), we also employed this same procedure to the SVM approach. The constraints in lines 16-17 of Algorithm 2 refer to keep the prototypes out of the sample interchanging process between Z_1 and Z_2 for the OPF. We do the same with the support vectors in SVM. However, they may be selected for interchanging in future iterations if they are no longer prototypes or support vectors. For SVM, we use the LibSVM package [53] with radial basis function (RBF) kernel, parameter optimization, and the one-versus-one strategy for the multiclass problem to implement line 3.

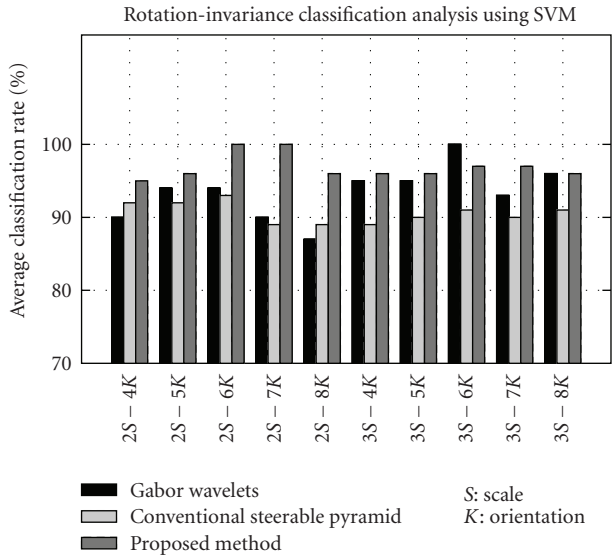

FIGURE 7: Classification accuracy comparison using the SVM classifier obtained in rotated data set A using ($S = 2, 3$) scales with ($K = 4, 5, 6, 7, 8$) orientations for Gabor wavelets, conventional steerable pyramid decomposition, and our method.

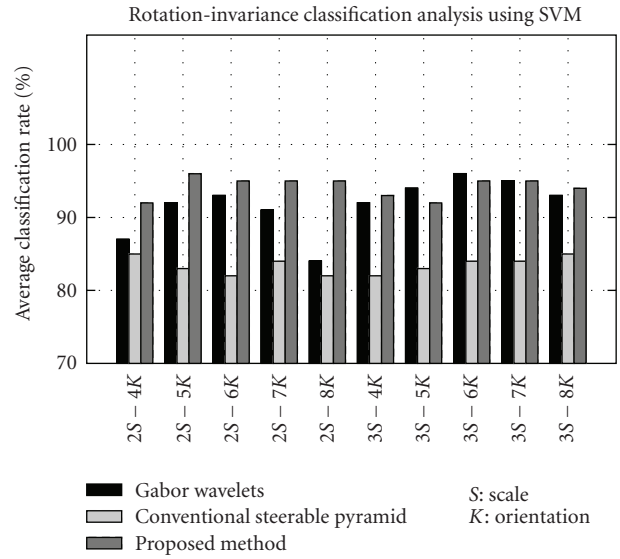

FIGURE 8: Classification accuracy comparison using the SVM classifier obtained in rotated data set B using ($S = 2, 3$) scales with ($K = 4, 5, 6, 7, 8$) orientations for Gabor wavelets, conventional steerable pyramid decomposition, and our method.

The experiments evaluate the accuracy on Z_3 and the computational time of each classifier, OPF and SVM. In all experiments, the data sets were divided into three parts: a training set Z_1 with 20% of the samples, an evaluation set Z_2 with 30% of the samples, and a test set Z_3 with 50% of the samples. These samples were randomly selected and each experiment was repeated 10 times with different sets Z_1, Z_2, and Z_3 to compute the mean accuracy.

Recall that an important motivation in our study is to use small-size feature vectors, in order to (1) show that the recognition accuracy of our approach is not compromised, and (2) facilitate texture recognition applications where data storage capacity is a limitation.

6.1. Effectiveness of the rotation invariance representation

To analyze the texture characterization capabilities of our feature extraction method against the conventional pyramid decomposition and the Gabor wavelets, we used Gaussian kernel support vector machines (SVMs) as texture classification mechanisms (note that the SVM parameters were optimized by using the cross-validation method).

Figure 7 compares the recognition accuracy obtained by those three methods in the *rotated data set A*, whereas Figure 8 depicts the recognition accuracy obtained in the *rotated data set B*. From both Figures, it can be seen that our image descriptor outperforms mostly the other two approaches, regardless of the number of scales or orientations considered during feature vector extraction.

In the case of the *rotated data set A*, the higher classification accuracies achieved by our method were obtained by using 7 orientations, which corresponds to image rotations in steps of 25.71°. By considering two and three decomposition levels ($S = 2, 3$), those accuracies are, respectively, 100% and 97.31%. The equivalent classification accuracies obtained by the Gabor wavelets are 90.36% and 93.90% ($S = 2, 3$; $K = 7$), whereas for the conventional steerable pyramid those accuracies are 89.67% and 90.36%. Note that the classification accuracies obtained by using $K = 6, 7, 8$ orientations are very close to each other. Therefore, to guarantee low-dimensionality feature vectors, we set $S = 2$ and $K = 6$ as the most appropriate parameter combinations for our rotation-invariant image descriptor.

In the case of the *rotated-set B*, the higher classification accuracies achieved by our descriptor were again obtained by using 7 orientations. Classification rates of 95.86% and 95.73% correspond respectively to feature vectors with $S = 2, 3$ scales and $K = 7$ orientations. Further, it is found that both Gabor wavelets and conventional steerable pyramid decomposition present lower classification rates, being, respectively, 91.05%, 95.35% for the first method and 84.22%, 84.23% for the second one. As stated in the results obtained in *rotated data set A*, the classification accuracies are very close to each other, when using $K = 6, 7$ or $K = 8$ orientations. From those results, we can reinforce that the most appropriate parameter settings for our descriptor are $S = 2$ scales and $K = 6$ orientations.

Furthermore, from the bar graphs shown in Figures 7 and 8, the highest classification rate obtained by the Gabor method is as good as the one obtained by our descriptor. However, this rate is obtained at $S = 3$ scales, whereas our proposed descriptor achieves the same performance using only $S = 2$ scales. In this sense, an important advantage of our method is its high performance rate at low-size feature vectors.

Our objective now is to demonstrate the recognition improvement of our novel classifier over the SVM approach.

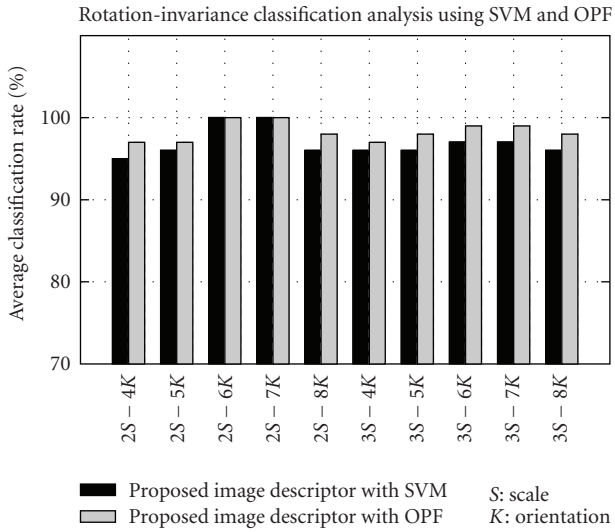

Figure 9: Classification accuracy comparison using the SVM classifier obtained in rotated data set A using ($S = 2, 3$) scales with ($K = 4, 5, 6, 7, 8$) orientations for Gabor wavelets, conventional steerable pyramid decomposition, and our method.

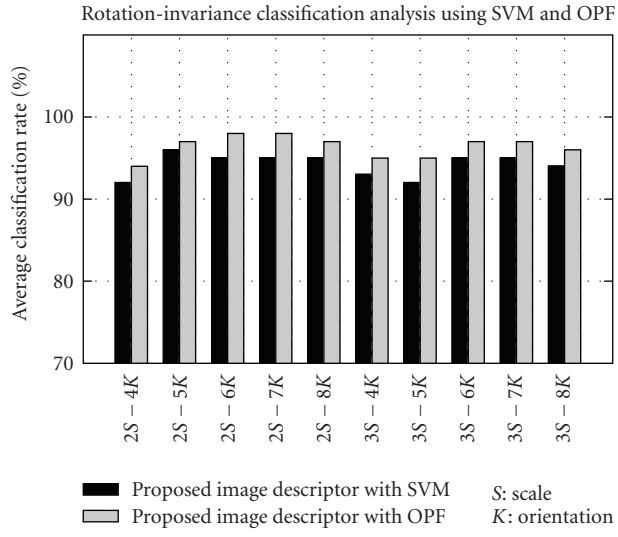

Figure 10: Classification accuracy comparison using the SVM classifier obtained in rotated data set B using ($S = 2, 3$) scales with ($K = 4, 5, 6, 7, 8$) orientations for Gabor wavelets, conventional steerable pyramid decomposition, and our method.

From Figure 9 it can be seen, that for almost all feature extraction configurations, the recognition rates of the OPF classifier are higher than those of the SVM classifier. The latter method presents the same recognition rates as the ones of the OPF classifier when using $S = 2$ scales and $K = 6, 7$ orientations. In the case of the image *rotated data set B*, our classifier yields better recognition rates for all feature extraction configurations (see Figure 10). By considering that it was found that the most appropriate parameter settings for our descriptor are $S = 2$ scales

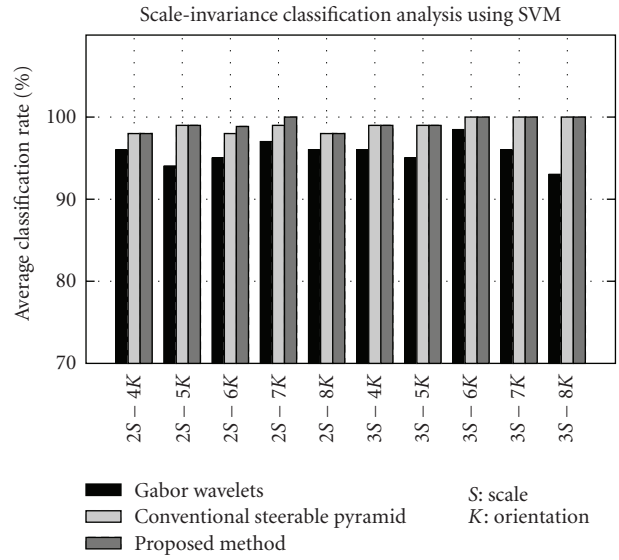

Figure 11: Classification accuracy comparison using the SVM classifier obtained in scaled data set A using ($S = 2, 3$) scales with ($K = 4, 5, 6, 7, 8$) orientations for Gabor wavelets, conventional steerable pyramid decomposition, and our method.

and $K = 6$ orientations, it is worth to mention that, by using this configuration, the recognition accuracy obtained by the OPF classifier is 98.49% in comparison with the corresponding accuracy of 95.48% obtained by the SVM classifier.

6.2. Effectiveness of the scale invariance representation

Figures 11 and 12 display the classification accuracy of our scale-invariant image descriptor against the conventional pyramid decomposition and the Gabor wavelets in the *scaled image data sets A* and *B*, respectively. Those Figures demonstrate the classification accuracy improvement of our image descriptor over both methods.

From Figure 11 it can be noticed that by using just $S = 2$ scales and $K = 7$ orientations, our feature extraction algorithm achieves a classification rate of 100%. This same rate is achieved by the other two methods, but at the cost of having larger image feature vectors. To obtain a classification rate of 100%, both pyramid decomposition and Gabor wavelets need at least $S = 3$ scales. Recalling Section 3.1, the feature vector dimensionality is obtained by multiplying the number of scales and orientations by a factor of 2, since we considered the mean and standard deviation as feature components. In this way, the dimensionality of our feature vectors is of size of 28 ($2 \times 2 \times 7$) elements, in comparison with a size of 42 ($2 \times 3 \times 7$) elements of their analogous feature vectors. By considering that the typical storage space of a float number is equal to 8 bytes, each of our feature vectors requires only 224 bytes to be stored, in comparison with the 336 bytes required for their analogous feature vectors. In this way, our image descriptor requires 66.7% less storage space than the one belonging to the compared descriptors.

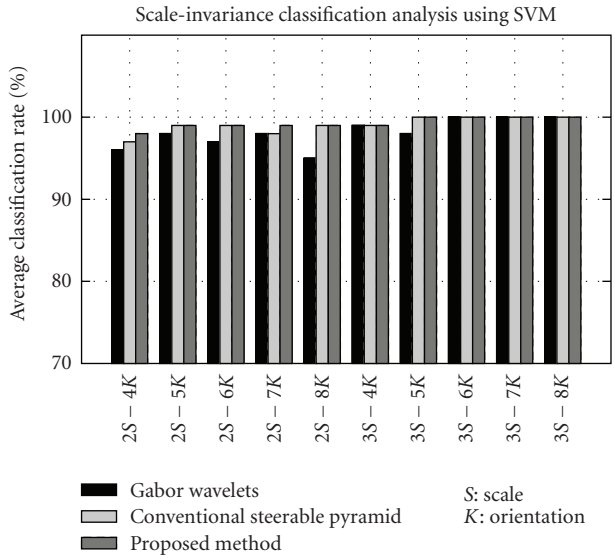

Figure 12: Classification accuracy comparison using the SVM classifier obtained in scaled data set B using ($S = 2, 3$) scales with ($K = 4, 5, 6, 7, 8$) orientations for Gabor wavelets, conventional steerable pyramid decomposition, and our method.

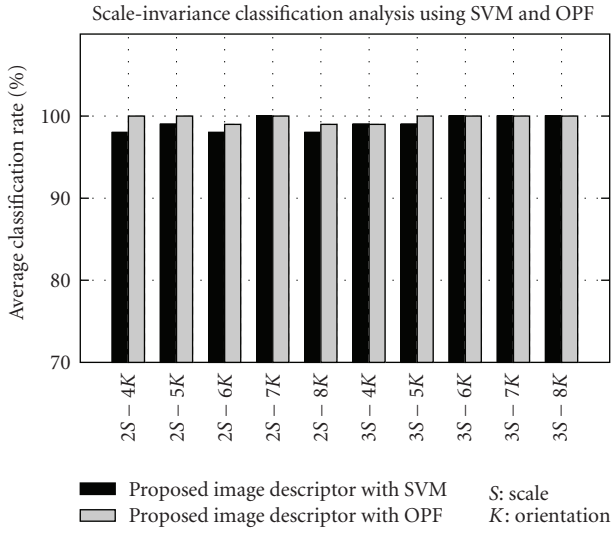

Figure 13: Classification accuracy comparison using the SVM classifier obtained in scaled data set A using ($S = 2, 3$) scales with ($K = 4, 5, 6, 7, 8$) orientations for Gabor wavelets, conventional steerable pyramid decomposition, and our method.

By analyzing the classification accuracies depicted in Figure 12, we can notice again that our image features perform better than the other two methods. However, the main difference between the results presented in Figures 11 and 12 is that in the case of the *scale data set B* all methods achieved higher classification rates. The reason for this lies in the tested texture data set, which has more discriminative samples to be used during the training phase of the classifier. This can be thought of as having sufficient discriminatory training data regardless of the testing data size.

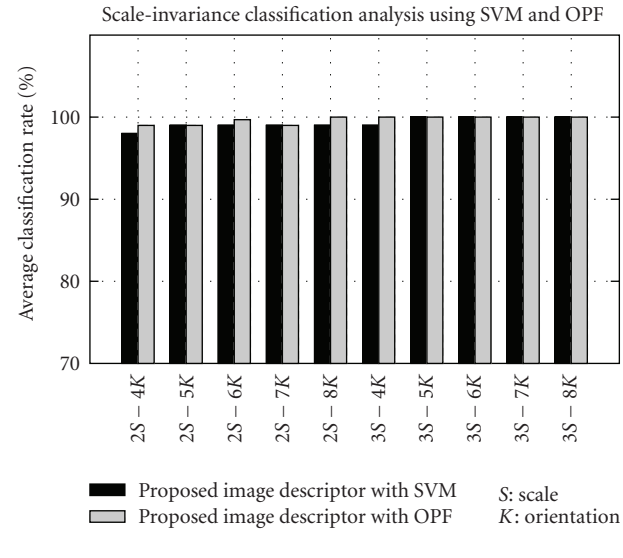

Figure 14: Classification accuracy comparison using the SVM classifier obtained in scaled data set B using ($S = 2, 3$) scales with ($K = 4, 5, 6, 7, 8$) orientations for Gabor wavelets, conventional steerable pyramid decomposition, and our method.

Another important question that arises now is to know how our feature classifier performs in both scaled data sets. To answer this question, we compared in Figures 13 and 14 the classification accuracies of the OPF against those obtained with SVMs. From Figure 13 we can see that by using a 16 dimension feature vector ($S = 2$ scales and $K = 4$ orientations) the OPF achieves a classification accuracy of 100%, which increases in turn the corresponding accuracy of the SVM up to 2%. Although this difference may appear despicable, note that the SVMs achieved the same accuracy when using a 28 dimension feature vector ($S = 2$ scales and $K = 7$ orientations). Thus, our recognition system requires almost only the half feature vector dimensionality to obtain a complete recognition. In the case of the *scaled data set B*, the OPF achieved a 100% classification rate by using 24 ($S = 2$ scales and $K = 6$ orientations) dimension feature vectors. The SVMs achieved, in turn, this accuracy by using 30 ($S = 3$ scales and $K = 5$ orientations) dimension feature vectors. By considering again an 8 byte storage space for a float number, our recognizer uses 192 (8×24) bytes to classify texture images in an efficient manner, whereas by using SVMs 240 (8×30) bytes are needed.

6.3. Summary of the results

In this subsection, we provide a summary of our experimental results. For notation purposes, we will denote our image descriptor, the Gabor wavelets, and the conventional pyramid decomposition descriptors as ID1, ID2, and ID3, respectively. The summary of our results for the *rotated data sets A* and *B* is provided in Tables 1 and 2. Table 1 compares for each rotated data set the mean recognition rates obtained by the three texture image descriptors using different scales ($S = 2, 3$) and different orientations ($K = 4, 5, 6, 7, 8$). In this

Table 1: Mean recognition rates for the three different texture image descriptors using Gaussian-kernel support vector machines as classifiers in the rotated data sets A and B.

Rotated data set	Scales (S)/ orientations (K)	ID1	ID2	ID3
A	($S = 2$; $K = 4,5,6,7,8$)	**95.89**%	93.19%	93.19%
A	($S = 3$; $K = 4,5,6,7,8$)	**97.99**%	92.36%	97.92%
B	($S = 2$; $K = 4,5,6,7,8$)	**92.30**%	85.30%	91.29%
B	($S = 3$; $K = 4,5,6,7,8$)	**96.70**%	85.76%	96.67%

Table 2: Mean recognition rates for the proposed rotation-invariant texture image descriptor using both OPF and SVM classifiers in the rotated datasets A and B.

Rotated data set	Scales (S)/ orientations (K)	OPF	SVM
A	($S = 2$; $K = 4,5,6,7,8$)	**98.89**%	95.89%
A	($S = 3$; $K = 4,5,6,7,8$)	**98.61**%	97.99%
B	($S = 2$; $K = 4,5,6,7,8$)	**97.35**%	92.30%
B	($S = 3$; $K = 4,5,6,7,8$)	**96.74**%	96.70%

Table 3: Mean recognition rates for the three different texture image descriptors using Gaussian-kernel support vector machines as classifiers in the scaled data sets A and B.

Scaled data set	Scales (S)/ orientations (K)	ID1	ID2	ID3
A	($S = 2$; $K = 4,5,6,7,8$)	**98.78**%	93.19%	97.04%
A	($S = 3$; $K = 4,5,6,7,8$)	**99.67**%	99.66%	96.05%
B	($S = 2$; $K = 4,5,6,7,8$)	**99.35**%	97.12%	99.90%
B	($S = 3$; $K = 4,5,6,7,8$)	**99.95**%	99.83%	99.44%

Table 4: Mean recognition rates for the proposed scale-invariant texture image descriptor using both OPF and SVM classifiers in the scaled data sets A and B.

Scaled data set	Scales (S)/ orientations (K)	OPF	SVM
A	($S = 2$; $K = 4,5,6,7,8$)	**99.03**%	98.78%
A	($S = 3$; $K = 4,5,6,7,8$)	**99.89**%	99.67%
B	($S = 2$; $K = 4,5,6,7,8$)	**99.58**%	99.35%
B	($S = 3$; $K = 4,5,6,7,8$)	**100**%	99.95%

set of experiments, we used Gaussian-kernel support vector machines (SVMs) as texture classification mechanisms. From our results, it can be noticed that our texture image descriptor performs better regardless of the data set used, or the image decomposition parameters considered during feature extraction (number of scales and orientations). Furthermore, as it can be seen in Table 2, the OPF classifier improves the recognition accuracies obtained by the SVM classifier in all of our experiments. The summarized results for the *scale data sets A* and *B* are presented in Tables 3 and 4. As we can see, our proposed recognition system performs again better than the previously mentioned approaches in both feature extraction and classification tasks.

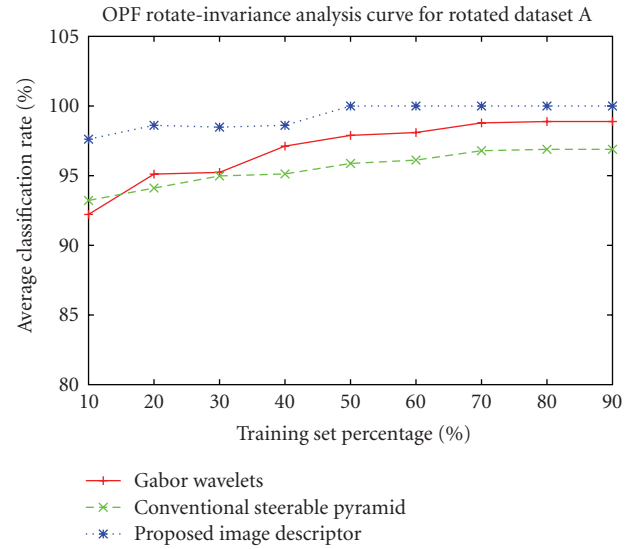

Figure 15: Average classification accuracy versus number of training samples in rotated data set A.

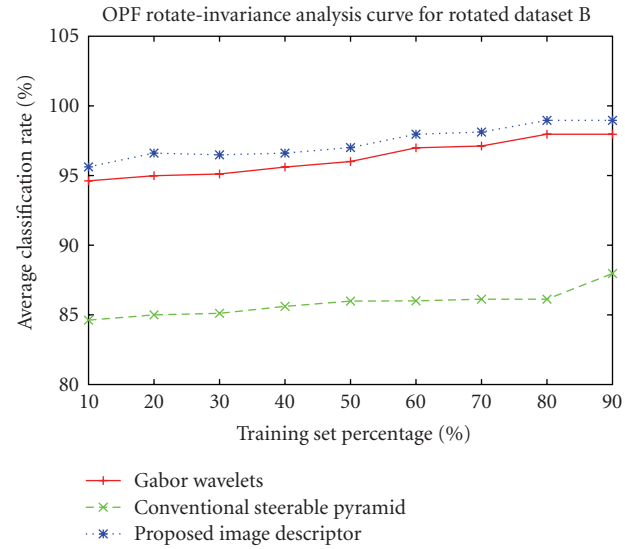

Figure 16: Average classification accuracy versus number of training samples in rotated data set B.

6.4. Training sample classification rates

The achieved performances of our feature classifier using different number of training samples are shown graphically in Figures 15–18. The y-axis denotes the achieved average classification rate, whereas the x-axis represents the number of training samples considered. Each unique line belongs to each of the evaluated image descriptors (Gabor wavelets, conventional steerable pyramid decomposition, and our method). From those Figures we can see that almost all image descriptors attain reasonably good results even by using small-dimensional feature vectors (85%+). However, the superiority of our system can be clearly seen. Note that in the case of the *rotated data sets A* and *B* (Figures 15

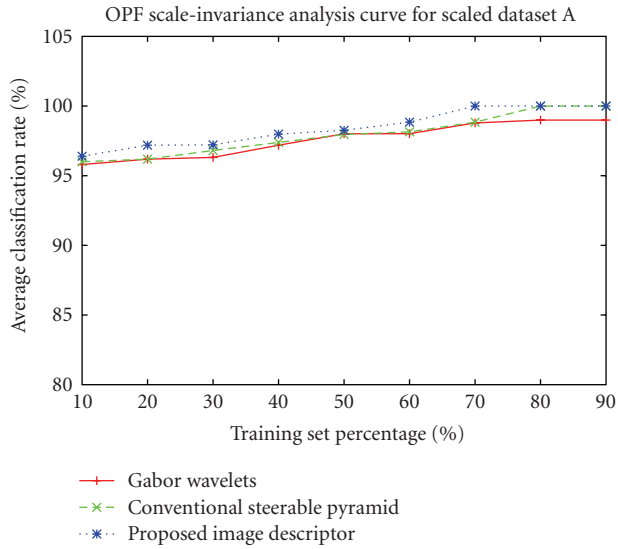

Figure 17: Average classification accuracy versus number of training samples in scaled data set A.

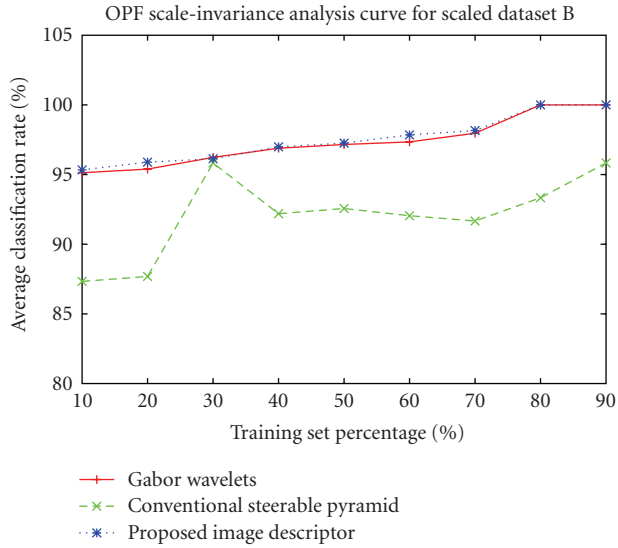

Figure 18: Average classification accuracy versus number of training samples in scaled data set B.

and 16, resp.) our system remained with high accuracies above 97% and 95%, respectively. The analogous accuracies of Gabor wavelets in both data sets have not reached the rates of our descriptor in any number of training samples used. At the same time, our improvements over the conventional steerable pyramid decomposition are notorious. In contrast, in the case of the *scale data set A*, we can see that the accuracies of the image descriptors are very close to each other. However, our system achieved a 100% classification accuracy by using less training samples as the other two methods (Figure 17). Moreover, it can be seen from Figure 18 that by increasing the number of training samples, the conventional steerable pyramid decomposition does not improve much and in some cases it even deteriorates its classification accuracies. The curves in this figure show that the average classification accuracies between our proposed image descriptor and the Gabor wavelets are almost the same. However, the accuracies of our method are still higher. Finally, by analyzing all those results, we can see clearly that our method provides a significant improvement over the other approaches.

Table 5: Execution times of the OPF and SVM approaches in seconds.

data set	OPF	SVM
Rotated data set A	**0.0260**	2.916
Rotated data set B	**0.0480**	6.256
Scaled data set A	**0.0388**	4.877
Scaled data set B	**0.0487**	6.151

6.5. Recognition processing time

We also computed the recognition processing time for the classifiers in the evaluated data sets. Note that for computing the processing time, we considered both training and classification times together. Table 5 displays those values in seconds.

As we can see, the OPF algorithm is extremely faster than the SVM classifier. For the *rotated data sets A* and *B* as well as for the *scaled data sets A* and *B*, the OPF classifier was 112.15, 130.33, 125.69, and 126.30 times faster, respectively. The SVM algorithm had a slow performance due to the fact of the optimization procedure implemented in the libSVM [53]. However, by removing the optimization procedures, this processing time could be decreased. In turn, this could produce lower classification rates.

7. CONCLUSIONS

A novel texture classification system was proposed in this work. Its main features are (1) a new rotation-invariant and scale-invariant image descriptor, as well as (2) a recent multiclass recognition method based on optimum-path forest. The proposed image descriptor exploits the discriminatory properties of the steerable pyramid decomposition for texture characterization. By finding either the *dominant orientation* or *dominant scale* value presented in the texture images, the feature elements are aligned according to this value. By doing this, a more reliable feature extraction process can be performed, since corresponding feature elements of distinct feature vectors coincide with images either at the same orientations or at the same scales. In addition, our system adopted a recent approach for pattern classification based on optimum-path forest, which finds prototypes with zero classification errors in the training set and learns from errors in an evaluation set, without increasing the training set size. By combining the discriminating power of our image descriptor and classifier, our system uses small size feature vectors to characterize texture images without compromising overall classification rates, being ideally for

real-time applications or for applications where data storage capacity is a limitation.

State-of-the-art results on four image data sets derived from the standard Brodatz database were further discussed. For the rotation-invariance evaluation, our method obtained a mean classification rate of 98.89% in comparison with a mean accuracy of 95.89% obtained by using SVMs in the *rotated data set A*. In the case of the *rotated data set B,* those rates are 97.35% and 92.30%, respectively. Concerning the scale-invariance evaluation, our system improves classification rates from 98.78% to 99.03% in the case of the *scaled data set A*, whereas in the *scaled data set B* those rates are improved from 99.35% to 99.58%.

Further, the OPF multiclass classifier outperformed the SVM in the four data sets. It is a new promising graph tool for pattern recognition, which differs from traditional approaches in that it does not use the idea of feature space geometry, therefore, better results in overlapped databases are achieved.

ACKNOWLEDGMENTS

The authors would like to thank CNPq (Grants 302427/2004-0, 134990/2005-6, 477039/2006-5, and 311309/2006-2), Webmaps II CNPq project, FAPESP (Grant 03/14096-8), Microsoft Tablet PC Technology and Higher Education project, as well as CAPES/COFECUB (Grant 392/08) for their financial support. They would also like to thank the anonymous reviewers for their comments.

REFERENCES

[1] T. R. Reed and J. M. H. Dubuf, "A review of recent texture segmentation and feature extraction techniques," *CVGIP: Image Understanding*, vol. 57, no. 3, pp. 359–372, 1993.

[2] M. Unser, "Texture classification and segmentation using wavelet frames," *IEEE Transactions on Image Processing*, vol. 4, no. 11, pp. 1549–1560, 1995.

[3] B. Balas, "Attentive texture similarity as a categorization task: comparing texture synthesis models," *Pattern Recognition Society*, vol. 41, no. 3, pp. 972–982, 2008.

[4] F. Wu, C. Zhang, and J. He, "An evolutionary system for near-regular texture synthesis," *Pattern Recognition*, vol. 40, no. 8, pp. 2271–2282, 2007.

[5] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1349–1380, 2000.

[6] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma, "A survey of content-based image retrieval with high-level semantics," *Pattern Recognition*, vol. 40, no. 1, pp. 262–282, 2007.

[7] R. M. Pickett, "Visual analyses of texture in the detection and recognition of objects," in *Picture Processing and Psychopictorics*, B. C Lipkin and A. Rosenfeld, Eds., pp. 289–308, Academic Press, New York, NY, USA, 1970.

[8] J. K. Hawkins, "Textural properties for pattern recognition," in *Picture Processing and Psychopictorics*, B. C. Lipkin and A. Rosenfeld, Eds., pp. 347–370, Academic Press, New York, NY, USA, 1970.

[9] A. K. Jain and K. Karu, "Learning texture discrimination masks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 2, pp. 195–205, 1996.

[10] B. Jähne, *Digital Image Processing*, Springer, London, UK, 5th edition, 2002.

[11] J. Wu, *Rotation invariant classification of 3D surface texture using photometric stereo*, Ph.D. thesis, Department of Computer Science, School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, UK, 2003.

[12] Jiahua Wu and M. J. Chantler, "Combining gradient and albedo data for rotation invariant classification of 3D surface texture," in *Proceedings of the 9th IEEE International Conference on Computer Vision (ICCV '03)*, vol. 2, pp. 848–855, Nice, France, October 2003.

[13] W. K. Pratt, *Digital Image Processing: PIKS Inside*, John Wiley & Sons, Los Altos, Calif, USA, 3rd edition, 2001.

[14] G. L. Gimel'farb and A. K. Jain, "On retrieving textured images from an image database," *Pattern Recognition*, vol. 29, no. 9, pp. 1461–1483, 1996.

[15] B. Julesz, "Texton gradients: the texton theory revisited," *Biological Cybernetics*, vol. 54, no. 4-5, pp. 245–251, 1986.

[16] M. Tüceryan and A. K. Jain, "Texture segmentation using Voronoi polygons," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 2, pp. 211–216, 1990.

[17] R. M. Haralick, "Statistical and structural approaches to texture," *Proceedings of the IEEE*, vol. 67, no. 5, pp. 786–804, 1979.

[18] K. R. Castleman, *Digital Image Processing*, Prentice-Hall, Englewood-Cliffs, NJ, USA, 1996.

[19] G. R. Cross and A. K. Jain, "Markov random field texture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, no. 1, pp. 25–39, 1983.

[20] R. C. Dubes and A. K. Jain, "Random field models in image analysis," *Journal of Applied Statistics*, vol. 16, no. 2, pp. 131–164, 1989.

[21] A. P. Pentland, "Fractal-based description of natural scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 6, pp. 661–674, 1984.

[22] J. Mao and A. K. Jain, "Texture classification and segmentation using multiresolution simultaneous autoregressive models," *Pattern Recognition*, vol. 25, no. 2, pp. 173–188, 1992.

[23] A. C. Bovik, M. Clark, and W. S. Geisler, "Multichannel texture analysis using localized spatial filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 1, pp. 55–73, 1990.

[24] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognition*, vol. 24, no. 12, pp. 1167–1186, 1991.

[25] K. Laws, *Textured image segmentation*, Ph.D. thesis, Department of Electrical Engineering, University of Southern California, Los Angeles, Calif, USA, 1980.

[26] J. P. Papa, A. X. Falcão, C. T. N. Suzuki, and N. D. A. Mascarenhas, "A discrete approach for supervised pattern recognition," in *Proceedings of the 12th International Workshop on Combinatorial Image Analysis (IWCIA '08)*, Buffalo, NY, USA, April 2008.

[27] W. T. Freeman and E. H. Adelson, "The design and use of steerable filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 9, pp. 891–906, 1991.

[28] J. A. Montoya-Zegarra, J. P. Papa, N. J. Leite, R. da Silva Torres, and A. X. Falcão, "Rotation-invariant texture recognition," in *Proceedings of the 3rd International Symposium on Advances in Visual Computing (ISVC '07)*, vol. 4842 of *Lecture Notes in Computer Science*, pp. 193–204, Springer, Lake Tahoe, Nev, USA, November 2007.

[29] J. A. Montoya-Zegarra, N. J. Leite, and R. da Silva Torres, "Rotation-invariant and scale-invariant steerable pyramid

decomposition for texture image Retrieval," in *Proceedings of the 20th Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI '07)*, pp. 121–128, IEEE Computer Society, Belo Horizonte, MG, Brazil, October 2007.

[30] J. P. Papa, A. X. Falcão, P. A. V. Miranda, C. T. N. Suzuki, and N. D. A. Mascarenhas, "Design of robust pattern classifiers based on optimum-path forests," in *Proceedings of the 8th International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing (ISMM '07)*, pp. 337–348, MCT/INPE, Rio de Janeiro, Brazil, October 2007.

[31] J. Portilla and E. P. Simoncelli, "Parametric texture model based on joint statistics of complex wavelet coefficients," *International Journal of Computer Vision*, vol. 40, no. 1, pp. 49–70, 2000.

[32] A. del Bimbo, *Visual Information Retrieval*, Morgan Kaufmann, San Francisco, Calif, USA, 1st edition, 1999.

[33] S. Arivazhagan, L. Ganesan, and S. P. Priyal, "Texture classification using Gabor wavelets based rotation invariant features," *Pattern Recognition Letters*, vol. 27, no. 16, pp. 1976–1982, 2006.

[34] D. Zhang, A. Wong, M. Indrawan, and G. Lu, "Content based image retrieval using Gabor texture features," in *Proceedings of the 1st IEEE Pacific-Rim Conference on Multimedia (PCM '00)*, pp. 392–395, Sydney, Australia, December 2000.

[35] J. P. Papa, A. X. Falcão, P. A. V. Miranda, C. T. N. Suzuki, and N. D. A. Mascarenhas, "A new pattern classifier based on optimum path forest," Tech. Rep. IC-07-13, Institute of Computing, State University of Campinas, São Paulo, Brazil, May 2007.

[36] A. X. Falcão, J. Stolfi, and R. de Alencar Lotufo, "The image foresting transform: theory, algorithms, and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 19–29, 2004.

[37] C. Allène, J. Y. Audibert, M. Couprie, J. Cousty, and R. Keriven, "Some links between min-cuts, optimal spanning forests and watersheds," in *Proceedings of the 8th International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing (ISMM '07)*, pp. 253–264, MCT/INPE, Rio de Janeiro, Brazil, October 2007.

[38] P. K. Saha and J. K. Udupa, "Relative fuzzy connectedness among multiple objects: theory, algorithms, and applications in image segmentation," *Computer Vision and Image Understanding*, vol. 82, no. 1, pp. 42–56, 2001.

[39] S. Beucher and F. Meyer, "The morphological approach to segmentation: the watershed," in *Mathematical Morphology in Image Processing*, pp. 433–481, Marcel Dekker, New York, NY, USA, 1993.

[40] L. Vincent and P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 6, pp. 583–598, 1991.

[41] R. Lotufo and A. X. Falcão, "The ordered queue and the optimality of the watershed approaches," in *Proceedings of the International Symposium on Mathematical Morphology (ISMM '00)*, vol. 18, pp. 341–350, Kluwer Academic Publishers, Palo Alto, Calif, USA, June 2000.

[42] R. Audigier and R. Lotufo, "Seed-relative segmentation robustness of watershed and fuzzy connectedness approaches," in *Proceedings of the 20th Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI '07)*, pp. 61–68, IEEE CPS, Belo Horizonte, MG, Brazil, October 2007.

[43] R. Audigier and R. Lotufo, "Watershed by image foresting transform, tie-zone, and theoretical relationship with other watershed definitions," in *Proceedings of the 8th International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing (ISMM '07)*, pp. 277–288, MCT/INPE, Rio de Janeiro, Brazil, October 2007.

[44] University of Southern California Signal, Institute I.P., "Rotated textures," March 2007, http://sipi.usc.edu/services/database/Database.html.

[45] T. N. Tan, "Rotation invariant texture features and their use in automatic script identification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 7, pp. 751–756, 1998.

[46] G. M. Haley and B. S. Manjunath, "Rotation-invariant texture classification using a complete space-frequency model," *IEEE Transactions on Image Processing*, vol. 8, no. 2, pp. 255–269, 1999.

[47] F. Lahajnar and S. Kovačič, "Rotation-invariant texture classification," *Pattern Recognition Letters*, vol. 24, no. 9-10, pp. 1151–1161, 2003.

[48] M. Pietikäinen, T. Ojala, and Z. Xu, "Rotation-invariant texture classification using feature distributions," *Pattern Recognition*, vol. 33, no. 1, pp. 43–52, 2000.

[49] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[50] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837–842, 1996.

[51] E. P. Simoncelli and W. T. Freeman, "The steerable pyramid: a flexible architecture for multi-scale derivative computation," in *Proceedings of the 2nd IEEE International Conference on Image Processing (ICIP '95)*, vol. 3, pp. 444–447, Washington, DC, USA, October 1995.

[52] M. N. Do and M. Vetterli, "Wavelet-based texture retrieval using generalized Gaussian density and Kullback-Leibler distance," *IEEE Transactions on Image Processing*, vol. 11, no. 2, pp. 146–158, 2002.

[53] C. C. Chang and C.J. Lin, "LIBSVM: a library for support vector machines," 2001, http://www.csie.ntu.edu.tw/~cjlin/libsvm/.