An Efficient Algorithm for Fractal Analysis of Textures

Alceu Ferraz Costa, Gabriel Humpire-Mamani, Agma Juci Machado Traina
Department of Computer Science
University of São Paulo, USP
São Carlos, Brazil
{alceufc,ghumpire,agma}@icmc.usp.br

***Abstract*—In this paper we propose a new and efficient texture feature extraction method: the Segmentation-based Fractal Texture Analysis, or SFTA. The extraction algorithm consists in decomposing the input image into a set of binary images from which the fractal dimensions of the resulting regions are computed in order to describe segmented texture patterns. The decomposition of the input image is achieved by the Two-Threshold Binary Decomposition (TTBD) algorithm, which we also propose in this work. We evaluated SFTA for the tasks of content-based image retrieval (CBIR) and image classification, comparing its performance to that of other widely employed feature extraction methods such as Haralick and Gabor filter banks. SFTA achieved higher precision and accuracy for CBIR and image classification. Additionally, SFTA was at least 3.7 times faster than Gabor and 1.6 times faster than Haralick with respect to feature extraction time.**

***Keywords*-Fractal analysis; texture; feature extraction; content based image retrieval; image classification; image processing;**

I. Introduction

Automated analysis of images has many applications, ranging from image classification to image retrieval. For instance, features automatically extracted from images are used in content-based image retrieval (CBIR) to find and organize visual information [1].

Due to the semantic gap problem [2], which corresponds to the difference between the human user image perception and what automatically extracted features convey, an important aspect of the feature extraction task is to obtain a set of features (i.e. a feature vector) that is able to succinctly and efficiently represent the visual content of an image.

In many applications, texture features can be used to address the semantic gap problem. For example, it can be employed to describe terrain surfaces in satellite images and organ's tissues in the medical imaging domain. As a result, most of the research in texture analysis area is dedicated to improve the discriminatory ability of the features extracted from the image [3].

Texture analysis, however, is usually a very time-consuming process. For instance, in order to accurately capture the textural characteristics of an image, algorithms that rely on filter banks [4] or co-occurrence gray level matrices (GLCMs) [5] have to consider multiple orientations and scales. In scenarios where a big volume of images is involved, the computation cost problem can be critical.

In order to deal with the computational cost problem in texture analysis, we propose a new and efficient feature extraction algorithm: the Segmentation-based Fractal Texture Analysis (SFTA). The extraction algorithm consists in decomposing the input image into a set of binary images from which the fractal dimensions of the regions' borders are computed from to describe the segmented texture patterns. In order to decompose the input image, a new algorithm, named Two-Threshold Binary Decomposition (TTBD) is also proposed.

We evaluated the proposed SFTA feature extraction method for three different datasets. The first dataset corresponds to regions of interest (ROIs) of lung computed tomography (CT) scans. The two remaining datasets, *KTH-TIPS* [6] and *Textured Surfaces* [7] are publicly available and widely employed to evaluate texture analysis methods [8], [9], [10].

We compared SFTA to widely employed texture extraction methods, such as Haralick descriptors [5], Gabor filter banks [4] and others. In our experiments SFTA has shown superior performance with respect to image classification accuracy and CBIR precision. Furthermore, SFTA is at least 3.7 times faster than Gabor and 1.6 times faster than Haralick with respect to feature extraction time.

This paper is organized as follows. Section II discusses related works and previous researches in texture analysis. Section III details our proposed feature extraction method. Experiments, discussions and interpretation are presented in section IV. Finally, conclusions are provided in section V. The symbols used throughout the paper are listed in Table I.

II. Related work

Texture plays an important role in image analysis and understanding. Its role in domain-specific applications, such as in remote sensing, quality control and medical imaging is particularly vital due to its close relation to the underlying semantics in these cases [1]. Additionally, texture information can be employed in image segmentation, by classifying image pixels with basis on surrounding texture information.

Texture features are intended to capture the granularity and repetitive patterns of regions within an image. From a statistical point of view, textures can be seen as complicated pictorial patterns from which sets of statistics can be obtained for characterization purposes [11]. Examples of such statistics

TABLE I
TABLE OF SYMBOLS

Symbol	Definition
I	Grayscale image.
I_b	Binary image.
Δ	Border image.
n_l	Gray level range.
T	Set of threshold values.
n_t	Number of thresholds.
D	Fractal dimension.
D_0	Haussdorf fractal dimension.
ϵ	Box size in the box counting algorithm.
V_{SFTA}	SFTA Feature vector.

than can be extracted from an image or image region are mean gray level, standard deviation and entropy, among others.

Another widely employed approach to characterize texture consists in computing gray level co-occurrence matrices (GLCM) by counting the number of occurrences of gray levels at a given displacement and angle. Statistics such as contrast, energy, entropy are computed from the GLCM to obtain texture features as proposed by Haralick et al. in [5].

Filter bank-based methods provide another approach for texture analysis, and are employed both in classification and segmentation [8], [9]. Gabor filters [4], [12] is a popular example of a filter bank-based method for its invariance with respect to scale, rotation and displacement. Each Gabor filter is represented as a Gaussian function modulated by a complex sinusoidal signal. Considering that θ is the filter orientation, σ the standard deviation and λ the wavelength of the sinusoid, Equation 1 shows the formal representation of a Gabor filter:

$$g(x,y) = \frac{1}{2\pi\sigma_x\sigma_y}\exp\left(-\frac{1}{2}A + \frac{2\pi i R_1}{\lambda}\right) \tag{1}$$

where,

$$A = \left(\frac{R_1^2}{\sigma_x^2} + \frac{R_2^2}{\sigma_y^2}\right) \text{ and } \begin{bmatrix} R_1 \\ R_2 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{2}$$

The pixel position in the filter is represented as (x, y).

In the image analysis paradigm, fractal dimension measurements can be used to estimate and quantify the complexity of the shape or texture of objects [13], [14]. Fractal geometry involves various approaches to define fractional dimensions, where the most common is the Hausdorff's dimension. Considering an object that possesses an Euclidean dimension E, the Hausdorff's fractal dimension D_0 can be computed by the following expression:

$$D_0 = \lim_{\epsilon \to 0} \frac{\log N(\epsilon)}{\log \epsilon^{-1}} \tag{3}$$

where $N(\epsilon)$ is the counting of hyper-cubes of dimension E and length ϵ that fill the object.

If we consider an object represented by a binary image I_b, an approximation D for D_0 can be obtained through the box counting algorithm [15]. Without loss of generality, the algorithm for the 2D case can be described as follows. Initially, the image is divided into a grid composed of squares of size $\epsilon \times \epsilon$. The next step consists in counting the number $\bar{N}(\epsilon)$ of squares of size $\epsilon \times \epsilon$ that contains at least one pixel of the object. By varying the value ϵ, it is possible to create a $\log \bar{N}(\epsilon)$ vs $\log \epsilon^{-1}$ curve. Finally, this curve is approximated by a straight line using a line fitting method (e.g. least squares fitting). The fractal dimension D corresponds to the slope of this line.

In [16] a feature extraction method termed Fast Fractal Stack (FFS) is proposed. FFS has shown to be effective for the task of lung disease classification in computed tomography (CT) scans, that are images in which texture information has a close relation to the application semantics. The FFS extraction algorithm computes fractal measurements from a set of binary images obtained from the input grayscale image using the binary stack decomposition algorithm [11].

The binary stack decomposition algorithm models the input grayscale image I as a 2D function $I(x, y)$, where $I(x, y) \in \{1, 2, \cdots, n_l\}$. $I(x, y)$ is called the grayscale value or intensity of the pixel at position (x, y). This input image is decomposed by applying successive thresholding operations. When an image $I(x, y)$ is thresholded by a value $t \in \{1, \cdots, n_l\}$, a corresponding binary image is obtained. That is:

$$I_b(x,y) = \begin{cases} 1 \text{ if } I(x,y) \geq t \\ 0, \text{ otherwise.} \end{cases} \tag{4}$$

where $I_b(x, y)$ denotes the binary image obtained with the threshold t.

For a given original image, there are n_l potentially different binary images. This set of binary images is referred to as a binary image stack. However, to avoid extracting redundant information, FFS algorithm does not employ all possible values of thresholds to obtain the set of binary images. Rather, the strategy of selecting equally spaced thresholds is adopted:

$$t_i = \left\lfloor \frac{n_l}{n_t + 1} \cdot i \right\rfloor, \quad i = 1, 2, \cdots, n_t \tag{5}$$

where n_t is a user defined parameter that corresponds to the number of threshold values. In the experiments, FFS empirically set n_t equal to 8.

The disadvantage of FFS approach is that information about the input image, such as the gray level distribution, is not considered for selecting the set of thresholds. Additionally, despite FFS being able to describe texture information in lung CT images, our experiments have shown that it does not perform well for texture datasets such *KTH-TIPS* [6] and *Textured Surfaces*, [7].

III. PROPOSED METHOD

Our SFTA extraction algorithm can be divided into two main parts: first we decompose the input grayscale image

into a set of binary images. To decompose the input image we employ a new technique named Two-Threshold Binary Decomposition (TTBD) that we also propose in this work. Then, for each resulting binary image, we compute the fractal dimension from its regions' boundaries. Additionally, we compute the regions' mean gray level and size (pixel counting).

A. Two-Threshold Binary Decomposition

The Two-Threshold Binary Decomposition (TTBD) takes as input a grayscale image $I(x, y)$ and returns a set of binary images. The first step of TTBD consists in computing a set T of threshold values. In the binary stack decomposition described in section II, the set of threshold values is obtained by selecting equally spaced gray level values. TTBD adopts a different strategy that uses the input image gray level distribution information to compute the set of thresholds. This is achieved by employing the multi-level Otsu algorithm [17].

The multi-level Otsu algorithm consists in finding the threshold that minimizes the input image intra-class variance. Then, recursively, the Otsu algorithm is applied to each image region until the desired number of thresholds n_t is obtained, where n_t is a user defined parameter.

The next step of the TTBD algorithm consists in decomposing the input grayscale image $I(x, y)$ into a set of binary images. This is achieved by selecting pairs of thresholds from T and applying a two-threshold segmentation as follows:

$$I_b(x, y) = \begin{cases} 1 \text{ if } t_\ell < I(x, y) \leq t_u \\ 0, \text{ otherwise.} \end{cases} \quad (6)$$

where t_ℓ and t_u denote, respectively, lower and upper threshold values.

The set of binary images is obtained by applying the two-threshold segmentation (Equation 6) to the input image using all pairs of contiguous thresholds from $T \cup \{n_l\}$ and all pairs of thresholds $\{t, n_l\}, t \in T$, where n_l corresponds to the maximum possible gray level in $I(x, y)$. Thus, the number of resulting binary images is $2n_t$. Figure 1 illustrates the decomposition of a region taken from a satellite image using the TTBD algorithm. The resulting set of binary images was obtained using $n_t = 8$.

An important property of the TTBD is that the set the binary images obtained is a superset of all binary images that would be obtained by applying a one threshold segmentation (Equation 4) using the thresholds computed with the multi-level Otsu algorithm.

The rationale for using pairs of thresholds to compute the set of binary images is to segment objects that otherwise would not be segmented by regular threshold segmentation. This is especially true for objects and structures whose gray level lies in the middle ranges of the input image histogram. This is illustrated using a terrazzo floor texture in Figure 2. The chips (stones) that compose the surface can be roughly divided into three classes with respect to brightness: light, medium and dark. By using pairs of threshold values it is possible to extract region information from the medium brightness chips that would not be segmented by a single threshold value.

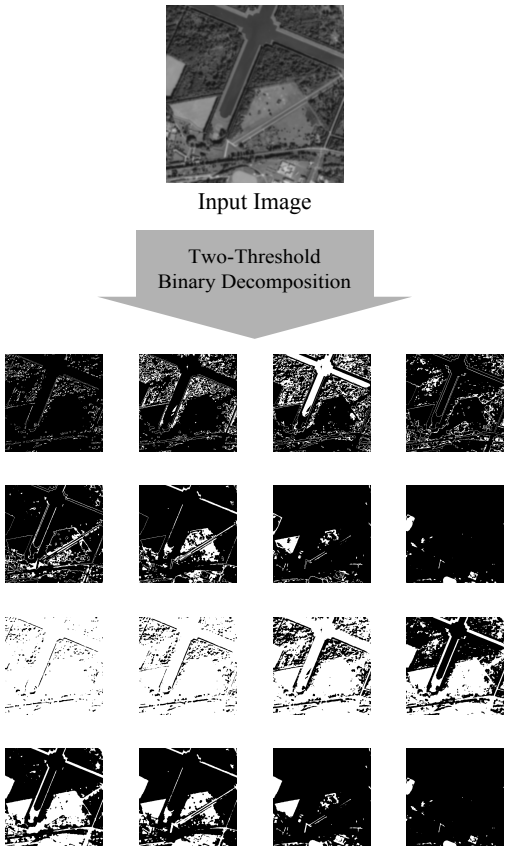

Fig. 1. Decomposition of a region taken from a satellite image using the TTBD algorithm. The resulting set of binary images was obtained using $n_t = 8$.

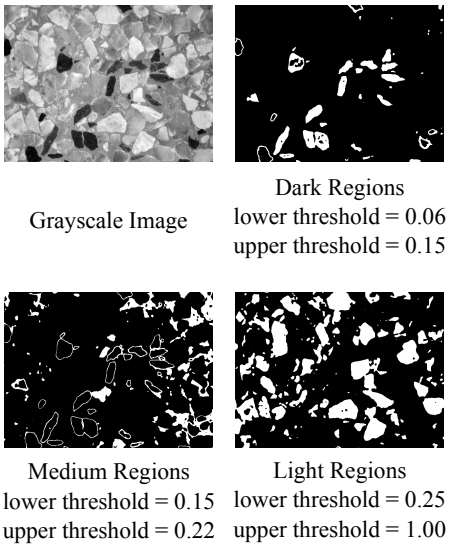

Fig. 2. Texture and the corresponding binary images obtained by two-threshold segmentation as described in Equation 6. Gray level values are in interval $[0, 1]$. By using pairs of thresholds, the TTBD algorithm is able to extract region information from the medium brightness chips that would not be segmented by a single threshold value. Texture surface photography from the *Textured Surfaces* dataset [7].

B. SFTA extraction algorithm

After applying the Two Threshold Binary Decomposition to the input gray level image, the SFTA feature vector is constructed as the resulting binary images' size, mean gray level and boundaries' fractal dimension. The fractal measurements are employed to describe the boundary complexity of objects and structures segmented in the input image. The regions' boundaries of a binary image $I_b(x, y)$ are represented as a border image denoted by $\Delta(x, y)$ and computed as follows:

$$\Delta(x,y) = \begin{cases} 1 \text{ if } \exists(x',y') \in N_8[(x,y)] : \\ \quad I_b(x',y') = 0 \ \wedge \\ \quad I_b(x,y) = 1, \\ 0, \text{ otherwise.} \end{cases} \quad (7)$$

where $N_8[(x, y)]$ is the set of pixels that are 8-connected to (x, y). $\Delta(x, y)$ takes the value 1 if the pixel at position (x, y) in the corresponding binary image $I_b(x, y)$ has the value 1 and having at least one neighboring pixel with value 0. Otherwise, $\Delta(x, y)$ takes the value 0. Hence, one can realize that the resulting borders are one-pixel wide. The fractal dimension D is computed from each border image using the box counting algorithm described in section II.

The mean gray level and size (pixel count) complement the information extracted from each binary image without significantly increasing the computation time. Thus, the SFTA feature vector dimensionality corresponds to the number of binary images obtained by TTBD multiplied by three, since the following measurements are computed from each binary image: fractal dimension, mean gray level and size. Figure 3 illustrates SFTA extraction algorithm. Figure 4 illustrates the three measurements that are extracted from each binary image.

Algorithm 1 summarizes SFTA feature extraction process. V_{SFTA} denotes the resulting feature vector. In line 1 the set of thresholds is computed using the multi-level Otsu algorithm. In line 2 all pairs of contiguous thresholds in T are added to T_A. In line 3 the pairs of thresholds $\{t_i, n_l\}$, where n_l corresponds to the maximum gray level in $I(x, y)$, are added to T_B. Line 5 iterates over all pairs of threholds in $T_A \cup T_B$. Each pair of thresholds is used in line 6 to compute the two threshold segmentation as described in Equation 6. Lines 7-10 corresponds to fractal dimension, mean gray level and region area computation.

Since fractal dimension can be efficiently computed in linear time by the box counting algorithm proposed in [18], SFTA extraction algorithm asymptotic complexity is $O(N \cdot |T|)$, where N is the number of pixels in the grayscale image I, and $|T|$ is the number of different thresholds resulting from the multi-level Otsu algorithm.

IV. Experiments

We evaluated our proposed method (SFTA) for the tasks of CBIR and image classification using three different datasets, *Lung CT ROIs*, *KTH-TIPS* and *Textured Surfaces*. The SFTA performance was compared to that of the following feature vectors: FFS, Haralick, Gabor, Histogram, Basic Texture and

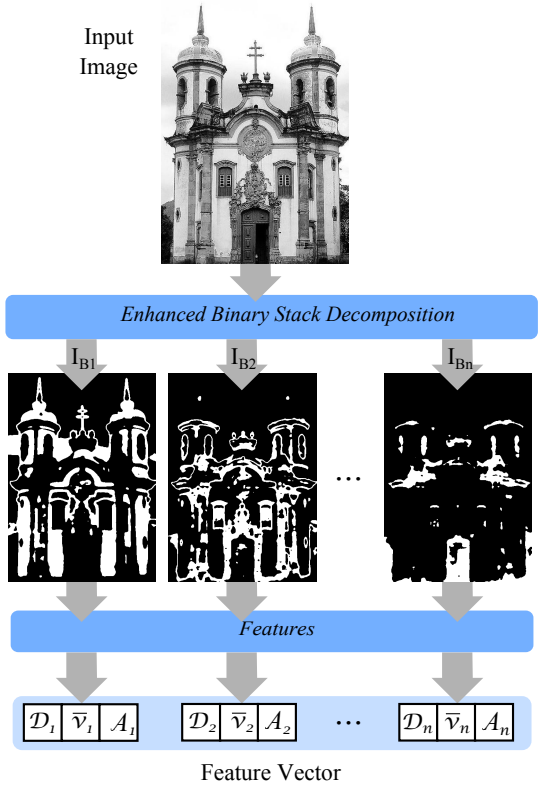

Fig. 3. SFTA extraction diagram taking as input a grayscale image. First the input image is decomposed into a set of binary image by the TTBD algorithm. Then, the SFTA feature vector is constructed as the resulting binary images' size, mean gray level and boundaries' fractal dimension.

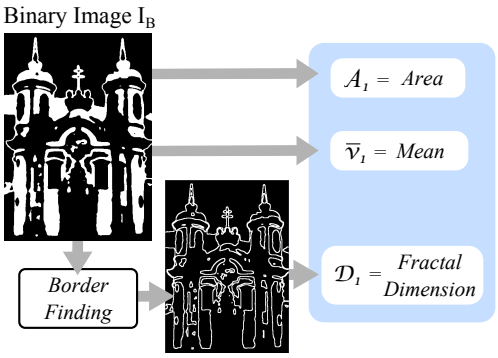

Fig. 4. Features extracted from each binary image resulting from the TTBD. Area and mean gray level are computed directly from the binary image. Fractal dimension is computed from the border image (Equation 7).

Combined. The extraction process for each feature vector is summarized in Table II. All extraction algorithms were implemented in Matlab.

SFTA requires the user to set the parameter n_t that defines the number of thresholds that will be employed in the input image decomposition. In section IV-E we provide an analysis showing how SFTA performance for image classification changes for different number of thresholds. Considering experimental results, we have set the n_t parameter to 8.

To evaluate the extraction algorithms for the task of CBIR, we employed precision and recall (P&R) curves. A rule of

TABLE II
FEATURE EXTRACTION METHODS USED IN THE EXPERIMENTS.

Method Name	Description	Number of Components
SFTA	SFTA feature vector using 8 thresholds.	48
FFS	FFS feature vector using 8 thresholds.	8
Haralick	Variance, entropy, uniformity, homogeneity, 3rd order moment, inverse of variance and step statistics from the image's GLCMs. The GLCMs are computed using 4 angles, 5 displacements and 8 gray levels.	140
Gabor	Mean and standard deviation of the response of each filter from Gabor filter bank. The Gabor filter bank is generated using 6 orientations and 4 scales.	48
Histogram	16 bin histogram computed by re-quantizing the input image to 16 gray levels.	16
Basic Texture	Mean, contrast, skewness, kurtosis, entropy and standard deviation of the input image's gray level distribution.	6
Combined	Combination of Haralick, Histogram, Basic Texture and Zernike moments [19].	418

Algorithm 1 SFTA extraction algorithm.

Require: Grayscale image I and number of thresholds n_t.
Ensure: Feature vector V_{SFTA}.

$T \leftarrow$ **MultiLevelOtsu**(I, n_t)
$T_A \leftarrow \{\{t_i, t_{i+1}\} : t_i, t_{i+1} \in T, i \in [1..|T| - 1]\}$
$T_B \leftarrow \{\{t_i, n_l\} : t_i \in T, i \in [1..|T|]\}$
$i \leftarrow 0$
for $\{\{t_\ell, t_u\} : \{t_\ell, t_u\} \in T_A \cup T_B\}$ **do**
  $I_b \leftarrow$ **TwoThresholdSegmentation**(I, t_ℓ, t_u)
  $\Delta(x, y) \leftarrow$ **FindBorders**(I_b)
  $V_{\text{SFTA}}[i] \leftarrow$ **BoxCounting**(Δ)
  $V_{\text{SFTA}}[i+1] \leftarrow$ **MeanGrayLevel**(I, I_b)
  $V_{\text{SFTA}}[i+2] \leftarrow$ **PixelCount**(I_b)
  $i \leftarrow i + 3$
end for
return V_{SFTA}

thumb to understand P&R curves is: the closer the curve is to the top of the graph, the better the technique's retrieval performance is [20]. In P&R curves, precision and recall are defined as follows:

$$\text{precision} = \frac{\text{number of relevant images retrieved}}{\text{number of images retrieved}} \quad (8)$$

$$\text{recall} = \frac{\text{number of relevant images retrieved}}{\text{number of relevant images in dataset}} \quad (9)$$

Each image was used as query center, returning the k most similar images using k-nearest neighbor (kNN) queries and the Euclidean distance function. An image was considered relevant when its class was the same as that of the query image. P&R curves were built by averaging the resulting curve of each query.

As for the classification experiments, we make use of an SVM (Support Vector Machine) classifier, built on a polynomial kernel using the SMO (Sequential Minimal Optimization) algorithm [21] We chose SVM due to its effectiveness and wide exploitation in texture classification [8], [9].

A. CBIR, Lung CT ROIs Dataset

The *Lung CT ROIs* dataset is composed of 3258 ROIs from lung computed tomography (CT) scans. Each ROI corresponds to an image of size 64 × 64 pixels classified either as normal or as one of the following lung disease patterns: consolidation, emphysema, thickening, honeycombing and ground glass. This dataset was used in [16] to evaluate the FFS extraction algorithm. Figure 6 shows a sample ROI from each class of the *Lung CT ROIs* dataset.

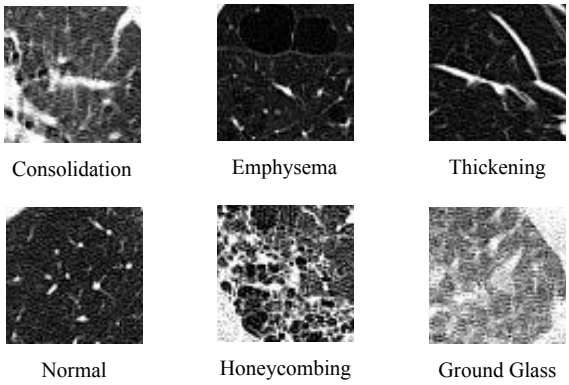

Fig. 5. Sample ROI from each class of the *Lung CT ROIs* dataset

Figure 6 shows the precision and recall graph obtained for the *Lung CT ROIs* dataset. SFTA corresponds to the black curve with $*$ marks, presenting the highest image retrieval precision for all recall levels. Gabor starts as the second best feature vector but its precision degrades for recall levels higher than 0.15 and is surpassed by Histogram. A possible explanation for this result is the problem of the curse of dimensionality. That is, the high dimensionality of the Gabor feature vector contributes to reduce its precision.

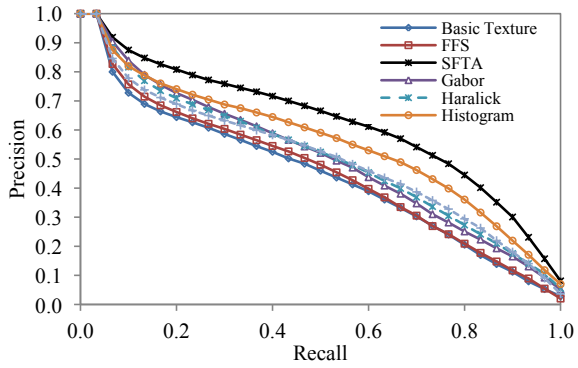

Fig. 6. Precision and recall curves obtained for the *Lung CT ROIs* dataset. SFTA corresponds to the black curve with $*$ marks, presenting the highest image retrieval precision for all recall levels.

B. CBIR, KTH-TIPS Dataset

The *KTH-TIPS* (Textures under varying Illumination, Pose and Scale) is a publicly available dataset[1] composed of 810 grayscale images [6]. The dataset is divided into 10 texture classes captured at different scales, illumination directions and poses. Examples of texture classes from this dataset include aluminum foil, brown bread, corduroy, cotton and sponge. Figure 7 shows an image sample from each texture class.

Figure 8 shows the precision and recall curves obtained for the dataset *KTH-TIPS*. SFTA and Gabor presented similar precision. For recall levels below 0.3 Gabor has slightly higher precision while SFTA is superior for higher recall levels. The Combined feature vector starts the curve with competitive performance when compared to SFTA and Gabor, but its precision rapidly decreases for recall higher than 0.15.

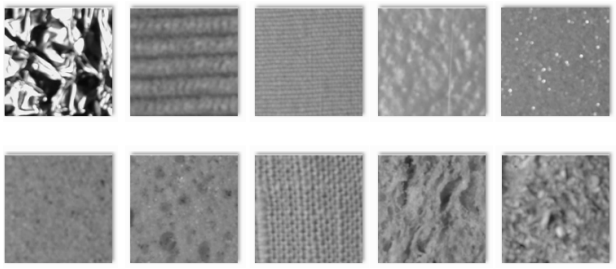

Fig. 7. Sample images from the *KTH-TIPS* dataset.

C. CBIR, Textured Surfaces Dataset

The *Textured Surfaces Dataset* [7] includes surfaces composed of materials such as wood, marble and fur under varying viewpoints, scales and illumination conditions. The dataset is publicly available[2] and consists of 1,000 images of size 640 $\times$ 480 pixels comprising 40 samples of 25 different textures. Figure 9 shows images samples from the dataset.

The precision and recall graph obtained for the *Textured Surfaces* dataset is shown in Figure 10. SFTA presented the highest precision except for recall levels higher than 0.75, when Combined showed slightly higher performance. Gabor precision was similar to that of SFTA for recall bellow 0.1 but then presented a sharp decrease.

[1]http://www.nada.kth.se/cvap/databases/kth-tips
[2]http://www-cvr.ai.uiuc.edu/ponce_grp/data

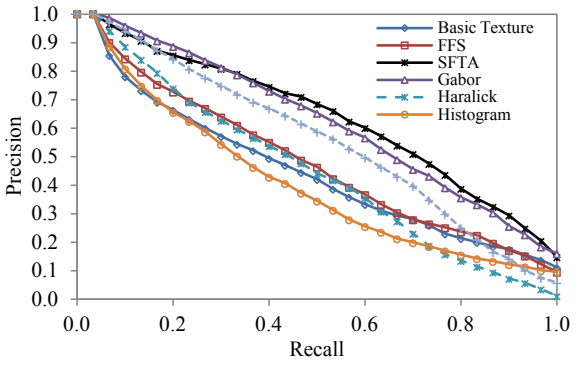

Fig. 8. Precision and recall curves obtained for the *KTH-TIPS* dataset.

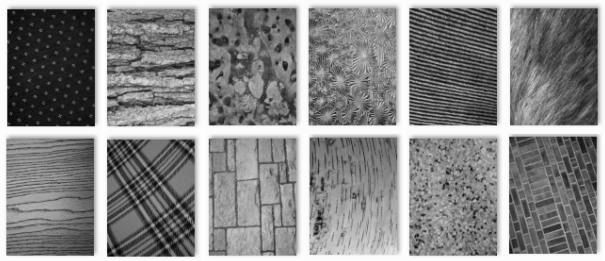

Fig. 9. Sample images from the *Textured Surfaces Dataset*.

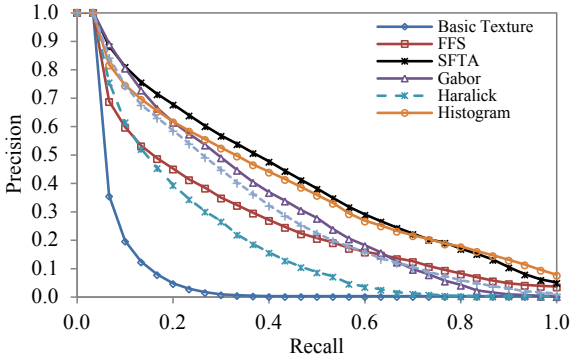

Fig. 10. Precision and recall curves obtained for the *Textured Surfaces* dataset. SFTA (black curve with $*$ marks) presented the highest precision except for recall levels higher than 0.75.

D. Image Classification and Feature Extraction Time

In this section we describe experiments performed to evaluate SFTA for the task of image classification. For this purpose, we employed the datasets described in sections IV-A to IV-C. We applied an SVM classifier with a polynomial kernel using the SMO algorithm to compare SFTA accuracy with the other extractors. The best SVM parameters for each dataset were established by 10-fold cross validation.

Table III shows classification accuracy with standard deviation values indicated between parentheses. The symbol $*$ highlights methods that obtained the best results for each dataset using a two-tailed t-Student test with p = 0.05. For the datasets *Lung CT ROIs* and *Textured Surfaces* SFTA presented the highest classification accuracy. SFTA also obtained the best classification accuracy for the dataset *KTH-TIPS* tying with Haralick, Combined and Gabor.

TABLE III
IMAGE CLASSIFICATION ACCURACY (%).

Extractor	Dataset		
	Lung CT ROIs	KTH-TIPS	Textured Surfaces
SFTA	* 88.83 (1.2)	* 95.19 (2.4)	* 90.80 (3.3)
FFS	84.36 (2.3)	73.46 (4.1)	75.30 (3.9)
Haralick	83.03 (2.2)	* 93.83 (3.6)	82.40 (4.1)
Histogram	77.72 (1.9)	70.00 (5.7)	75.60 (3.6)
Combined	78.24 (2.3)	* 94.57 (3.2)	86.90 (4.3)
Gabor	86.43 (1.4)	* 95.19 (2.3)	87.70 (3.4)
Basic Texture	67.65 (3.0)	72.83 (5.3)	13.20 (2.9)

The symbol $*$ highlights methods that obtained the best results for each dataset using a two-tailed t-Student test with p = 0.05.

In Figure 11 we show a plot of extraction time (vertical axis, log scale) versus classification accuracy (horizontal axis) for each dataset. Extraction time was obtained by running each extraction method in a computer with an Intel i7 2.66GHz processor, 8GB of RAM running Windows 7 64-bit OS. All extraction algorithms were implemented in Matlab.

For all datasets SFTA achieved the highest classification accuracy. Additionally, SFTA extraction time was always lower than Haralick, Gabor and Combined. FFS, Histogram and Basic Texture had lower extraction times when compared to SFTA but presented a significantly inferior classification accuracy. Combined presented the highest extraction time which can be attributed to the Zernike moments' extraction. Hence, we can conclude that the Combined feature vector is unsuitable for texture description.

When compared to Haralick, SFTA was 6.5, 1.6, 2.2 times faster for the datasets *Lung CT ROIs*, *KTH-TIPS* and Textured Surfaces, respectively. Haralick's high extraction time for dataset *Lung CT ROIs* can be regarded to the fact that the GLCMs dimensions from which the statistics described in Table II were computed are significantly higher than the ROIs' dimensions (64×64 pixels). Results from Figure 11 also show that SFTA was at least 3.7 times faster than Gabor (dataset *Textured Surfaces*) and up to 17 times faster for dataset *Lung CT ROIs*.

E. Analysis of SFTA performance for different number of thresholds

A central part of the SFTA algorithm is the decomposition of the input image into a set of binary images. This is achieved by the Two Threshold Binary Decomposition (TTBD) technique, presented in section III-A. TTBD algorithm takes as input parameter the number of thresholds employed to decompose the image. In this section we provide an analysis showing how SFTA performance changes for different number of thresholds.

In Figure 12 we show a plot of the classification accuracy (vertical axis) of the SFTA feature vector obtained employing different number of thresholds (horizontal axis). Each curve corresponds to one of the three datasets discussed in sections IV-A to IV-C. As for section IV-D, classification accuracy was obtained using an SVM classifier.

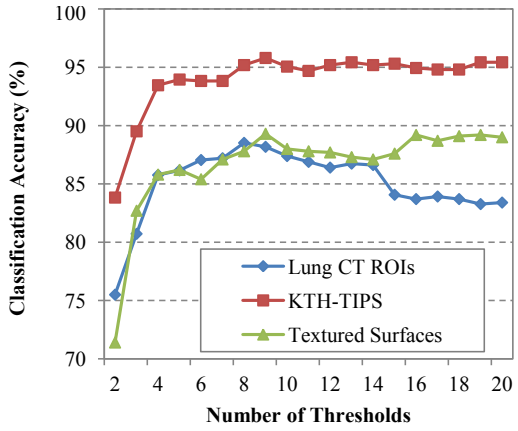

Fig. 12. Classification accuracy of the SFTA feature vector obtained employing different number of threshold values. Each curve corresponds to one of the three datasets presented in sections IV-A to IV-C.

For all datasets, classification accuracy increases until the number of thresholds is 8 or 9. After this point, we did not observe any significant increase in accuracy. A possible explanation for this behavior is that after the number of thresholds is higher than 8 or 9, no additional texture patterns are identified by the TTBD algorithm. Additionally, for dataset *Lung CT ROIs* we observed a decline in accuracy when the number of thresholds is higher than 14.

V. CONCLUSIONS

In this paper, we proposed a new feature extraction algorithm, the Segmentation based Fractal Texture Analisys (SFTA). The SFTA is aimed both at medical imaging and general domain texture feature extraction. We also proposed the Two-Threshold Binary Decomposition algorithm, that is employed by SFTA to partition the input grayscale image into a set of binary images that are used to characterize textural patterns.

We have evaluated SFTA both for the task of content-based image retrieval (CBIR) and image classification. We compared SFTA performance against other texture feature extraction methods such as Haralick and Gabor filter banks. Three datasets were considered in our experiments. The first consisted of regions of interest taken from lung computed tomography scans. The two remaining datasets are publicly available and widely used for evaluating texture analysis algorithms. Experimental results showed that SFTA performance was superior to that of Gabor and Haralick, achieving higher

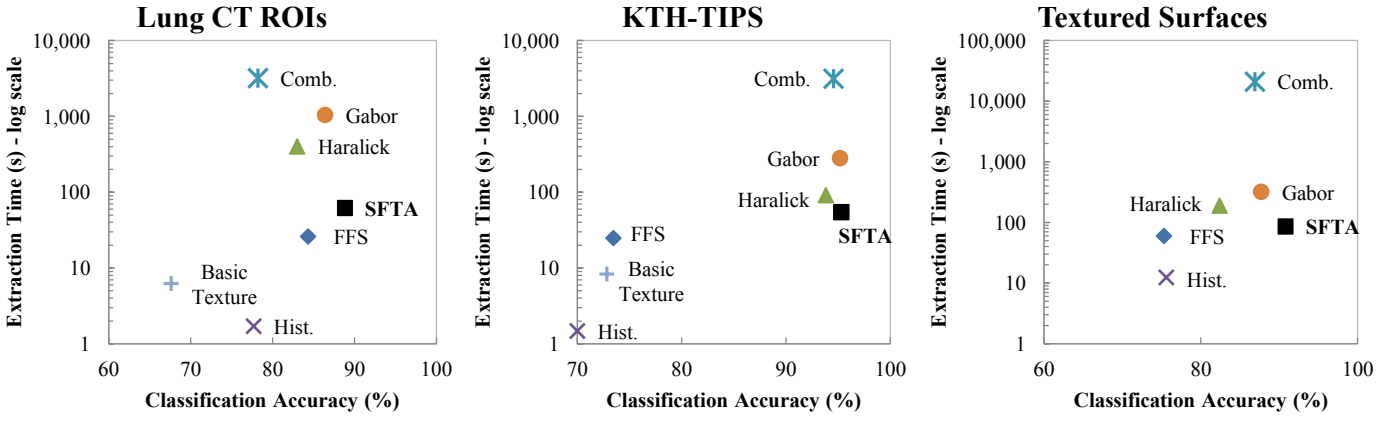

Fig. 11. Classification accuracy versus extraction time (log scale) for each extraction algorithm for datasets (a) *Lung CT ROIs* (b) *KTH-TIPS* and (c) *Textured Surfaces*. The proposed method (SFTA) is indicated by a boldface font and the symbol ■.

precision and accuracy for the task of CBIR and image classification.

Another important aspect of SFTA is that it is efficient. In our experiments, described in section IV-D, SFTA was at least 3.7 times faster than Gabor and 1.6 faster than Haralick with respect to extraction time, while always presenting equal or superior classification accuracy. This result can be attributed to the simple yet effective SFTA algorithm.

Acknowledgement

This research has been supported by FAPESP (São Paulo State Research Foundation), CNPq (Brazilian National Research Council), CAPES (Brazilian Coordination for Improvement of Higher Level Personnel) and Microsoft Research.

References

[1] R. Datta, D. Joshi, J. Li, and J. Z. Wang, "Image Retrieval: Ideas, Influences, and Trends of the New Age," *ACM Computing Surveys*, vol. 40, no. 2, pp. 1–60, 2008.

[2] T. M. Deserno, S. Antani, and R. Long, "Ontology of gaps in content-based image retrieval," *Journal of digital imaging*, vol. 22, no. 2, pp. 202–15, Apr. 2009.

[3] U. Kandaswamy, D. Adjeroh, and M. Lee, "Efficient Texture Analysis of SAR Imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 9, pp. 2075–2083, 2005.

[4] 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.

[5] R. M. Haralick, K. Shanmugam, and I. H. Dinstein, "Textural Features for Image Classification," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 3, no. 6, pp. 610–621, 1973.

[6] B. Caputo and E. Hayman, "Class-specific material categorisation," *Computer Vision*, pp. 1597–1604 Vol. 2, 2005.

[7] S. Lazebnik, C. Schmid, and J. Ponce, "A sparse texture representation using local affine regions," *Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1265–1278, Aug. 2005.

[8] M. Varma and A. Zisserman, "A statistical approach to material classification using image patch exemplars," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 11, pp. 2032–2047, Nov. 2009.

[9] J. Chen, S. Shan, C. He, G. Zhao, M. Pietikäinen, X. Chen, and W. Gao, "WLD: a robust local image descriptor." *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1705–20, Sep. 2010.

[10] T. Ahonen, J. Matas, and C. He, "Rotation invariant image description with local binary pattern histogram fourier features," *Image Analysis*, pp. 61–70, 2009.

[11] Y. Q. Chen and M. S. Nixon, "Statistical geometrical features for texture classification," *Pattern Recognition*, vol. 28, no. 4, pp. 537–552, 1995.

[12] X. Wang, X. Ding, and C. Liu, "Gabor filters-based feature extraction for character recognition," *Pattern Recognition*, vol. 38, no. 3, pp. 369–379, 2005.

[13] R. M. Rangayyan and T. M. Nguyen, "Fractal analysis of contours of breast masses in mammograms," *Journal of Digital Imaging*, vol. 20, no. 3, pp. 223–37, Sep. 2007.

[14] A. G. R. Balan, A. J. M. Traina, C. Traina Jr., and P. M. A. Marques, "Fractal analysis of image textures for indexing and retrieval by content," in *18th IEEE Symposium on Computer-Based Medical Systems (CBMS'05)*, 2005, pp. 581–586.

[15] M. Schroeder, *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise*. New York, NY, USA: W. H. Freeman, 1992.

[16] A. F. Costa, J. Tekli, and A. J. M. Traina, "Fast Fractal Stack: Fractal Analysis of Computed Tomography Scans of the Lung," in *MMAR'11:International ACM Workshop on Medical Multimedia Analysis and Retrieval*. Scottsdale, AZ, USA: ACM, 2011, pp. 13–18.

[17] P. Liao, T. Chen, and P. Chung, "A Fast Algorithm for Multilevel Thresholding," *Journal of Information Science and Engineering*, vol. 17, no. 5, pp. 713–727, 2001.

[18] C. Traina Jr., A. J. M. Traina, L. Wu, and C. Faloutsos, "Fast feature selection using fractal dimension," in *Brazilian Symposium on Databases (SBBD)*, João Pessoa, Brazil, 2000, pp. 158–171.

[19] A. Khotanzad and Y. H. Hong, "Invariant image recognition by Zernike moments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 5, pp. 489–497, May 1990.

[20] R. Baeza-Yates and B. Ribeiro-Neto, *Modern information retrieval*, 2nd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999.

[21] J. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods*. Cambridge, MA, USA: MIT Press, 1999, ch. 12, pp. 185–208.