Fractal Transforms and Feature Invariance

Ben A.M. Schouten and Paul M. de Zeeuw
Centre for Mathematics and Computer Sciences (CWI)
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
{B.A.M.Schouten, Paul.de.Zeeuw}@cwi.nl

Abstract

In this paper, fractal transforms are employed with the aim of image recognition. It is known that such transforms are highly sensitive to distortions like a small shift of an image. However, by using features based on statistics kept during the actual decomposition we can derive features from fractal transforms which are invariant to perturbations like rotation, translation, folding or contrast scaling. Further, we introduce a feature invariance measure which reveals the degree of invariance of a feature with respect to a database. The features and the way their invariance is measured, appear well-suited for the application to images of textures.

1 Introduction

Fractals can be generated by Iterated Function Systems (IFS) [2]. In most cases, the function system, to generate the fractal, consists of a limited number of functions $\mathbb{R}^n \rightarrow \mathbb{R}^n$. The domain for the functions in the system is some fixed part of $\mathbb{R}^n$. Simple variations like rotations of the fractal, lead to simple variations in the parameters of the function system. Chang [3] pays attention to the relationship of the fractal parameters (of the IFS) and some more complicated variations, like resize and relocation of the fractal. This is only done for binary deterministic fractals. The fractal transform of a *natural image* consists of a

Table 1. Difference between PIFS and IFS

	Domain	Range	# Functions
IFS	Whole Image	Part of Image	Limited
PIFS	Part of Image	Part of Image	Numerous

Partial Iterated Function System (PIFS). At the encoding, the image is subdivided into ranges, which form an non-overlapping cover of the image, see Figures 1, 2 and 3. For

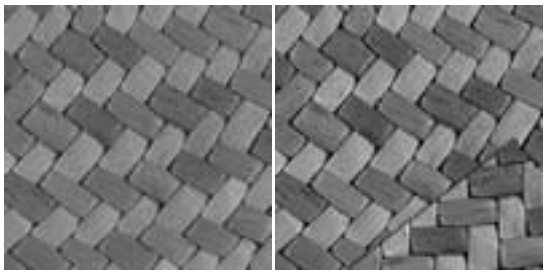

Figure 1. Fabric and fabric with a fold.

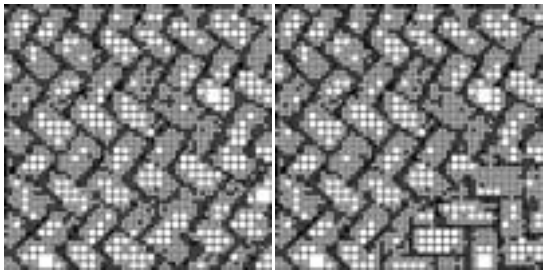

Figure 2. Quad-tree structure.

each range-block the encoder searches a different domain and a different function (affine transformation) to create the PIFS. The essential difference between IFS and PIFS, is the number of transformations in the system and the choice of the domain-blocks, see Table 1. Each function out of the system acts, by contrast scaling and luminance offset on the gray values of a local area of the image.

This paper is concerned with the use of fractal transformations as feature extractors [1, 6, 7, 8, 11, 12]. One of the reasons we want to investigate this problem is that many databases suffer from duplications. Often, similar images can be found in the database, mostly under some slightly different variations, like rotations, zooms, small translations etc. In the field of textile, for example, cloth may be presented in different folds but one still wants to recognize the texture.

The assumption in this paper is that though perturbations

Proceedings of the International Conference on Pattern Recognition (ICPR'00)
1051-4651/00 $10.00 @ 2000 IEEE

like rotation and folding may produce quite different fractal transforms, the impact on well-chosen statistics of the transform remains limited.

We present a (computable) measure for the invariance of a feature with respect to a perturbation. Because two images can generally be expected to be neither identical nor completely dissimilar, invariance is often up to a degree.

The organization of the paper is as follows: in Section 2 the basics of fractal coding schemes and fractal feature extraction are presented, followed by a description of four statistical features in Section 3. In Section 4 we subject images from a database to different types of perturbations and we demonstrate our new method which identifies the perturbed images with their originals in the database. We introduce feature invariance measures. In Section 5 conclusions are summarized.

2 Fractal feature extraction

2.1 Fractal image coding

For completeness we give a brief description of fractal image compression (FIC) [4, 5]. Most of the feature extraction methods are based on the parameters used in FIC.
A given image is partitioned into non-overlapping range blocks, see Figures 2 and 3. The fractal encoder searches for parts called domain-blocks (which can be larger and overlapping) in the same image that look similar under some fixed number of affine transformations. Such an affine transformation can be written as:

$$t_i(\vec{x}) = A_i\,\vec{x} + \vec{o}, A_i \equiv \begin{pmatrix} a_{11} & b_{12} & 0 \\ c_{21} & d_{22} & 0 \\ 0 & 0 & u_i \end{pmatrix}, \qquad \text{(1a)}$$

$$\vec{x} \equiv \begin{pmatrix} x \\ y \\ f(x,y) \end{pmatrix}, \vec{o} \equiv \begin{pmatrix} e_x \\ f_x \\ o_i \end{pmatrix}. \qquad \text{(1b)}$$

Index i indicates the range-blocks within the image, $f(x, y)$ denotes the gray value at position (x, y). u_i is the contrast scaling and o_i is the luminance offset. The u_i and o_i are used to match the gray values of the domain with the gray values of the range-block, within the limits of an imposed accuracy ϵ. In practice 8 different degrees of freedom are used for A, composed of 4 rotations over $0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}$, in combination with a possible flip of the domain-block along the second diagonal. In the code only the parameters of the transformations are stored. Usually, a domain-block has twice the size of a range-block.

The contractive nature of the transformations t_i makes the fractal encoder work. The transformation $T = \bigcup_{i=1}^{N} t_i$ (where N is the total number of range blocks in the image) has a fixed point which approximates the original image. It can be restored by iterating T in the decoding phase starting with an arbitrary given image. The code is producing detail at every iteration step.

2.2 Features and invariances

Most of the fractal feature extractors use the parameters, discussed in the previous section, to describe the image or object [1, 7, 9]. Kouzani et al. [6] uses only the e_x, f_x parameters; Baldoni et al. [1] and Vissac et al. [12] use simplified or altered fractal schemes, only inspired by the original compression scheme. Other researchers use the behavior under decompression as a feature [8, 11].

There is a major drawback in using fractal transformations for feature extraction. The same image (attractor) can be the result of two totally different fractal transformations, making it hard to compare two images. This occurs for instance when an image is slightly translated. Invariance to small translations can be achieved by input image shifting [11]. The features can also be made invariant to scale and rotation [8, 11]. Marie-Julie [7] uses multi-resolution or multi-compression schemes in which several domain partitions are used for one image. However, all the above methods are computationally expensive. We proposed statistical analysis of the fractal parameters [9], assuming that well-chosen statistics of the different fractal transforms remain invariant. We strive for invariance with respect to folds and gloss as well (in the context of textile). In the literature no such invariances are found.

3 The features

3.1 Introduction

As stated in the introduction we are interested in how several statistical aspects of the matching process within the fractal coding, alter by certain perturbations of the image. Here we give an outline of the features we employ, see also [9].

Most of the existing fractal coding schemes use a quad-tree structure as a subdivision of the image, see Figure 2. For a given accuracy ϵ (see Section 2.1), the algorithm finds a matching domain-block for the range-block in question. This is called a *success*. If there is no satisfactory match, the range-block splits into four equal parts. In this way several *depths i* of the quad-tree are created, containing range-blocks of the same size, see Figure 3.

We now introduce several feature histograms. Let L be the integer signifying the maximum depth imposed in the (fractal) decomposition with quad-tree refinement, likewise l signifies the minimum depth. A domain $\Omega_{l,L;k}$ is defined as:

$$\Omega_{l,L;k} \equiv \left\{(i,j) \in \mathbb{N}^2 \mid l \leq i \leq L, 1 \leq j \leq k\right\} \qquad (2)$$

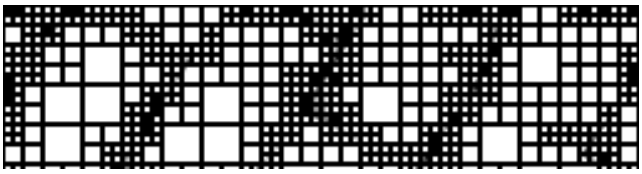

Figure 3. Detail of Figure 2, four depths i of the quad-tree are shown

where i is associated with the depth in the quad-tree structure and k is the chosen number of *feature-bins*, see Section 3.2. A histogram h on $\Omega_{l,L;k}$ is defined as a function

$$h: \quad \Omega_{l,L;k} \to \mathbb{R}, \quad \text{with} \quad h \geq 0. \tag{3}$$

If $(i,j) \in \Omega_{l,L;k}$ then $h_{ij} = h(i,j)$ is called the *value* of h at (i,j). A histogram h on $\Omega_{l,L;k}$ is called a *(weighted) quad-tree feature histogram* if it satisfies the following additional requirements:

$$h_{ij} = w_i v_{ij}, \tag{4}$$

where

$$\sum_{i=l}^{L} w_i \leq 1; \quad w_i \geq 0 \tag{5}$$

and

$$v_{ij} \geq 0; \quad \sum_{j=1}^{k} v_{ij} = 1, \quad \forall i \in \mathbb{N} \quad \text{with} \quad l \leq i \leq L. \tag{6}$$

It follows at once that a (weighted) quad-tree feature histogram h satisfies:

$$\sum_{i=l}^{L} \sum_{j=1}^{k} h_{ij} \leq 1. \tag{7}$$

Requirement (6) can be interpreted as that at each depth i we have k bins v_{ij} of which the contents add up to 1. Requirement (5) can be interpreted as weighing the contents of the bins depending on depth i. For an interpretation of the bins see Section 3.2 (next).

3.2 Description of the feature-bins

We use four different fractal image features to recognize a texture. The first feature is determined by the success rate (see Section 3.1) of the fractal decomposition at each depth in the quad-tree. The three different fractal features that follow relate to typical perceptual aspects of texture. Their definition involves the first feature.

1. *Coarseness Feature.*

   At each level i in the quad-tree we record the fraction w_i of the images area that is matched by the fractal decomposition (success). These fractions are the weights in (4). In case that an image has been fully resolved by fractal decomposition then the $\leq$ in both (5) and (7) turn into equal-signs. The w_i together ($l \leq i \leq L$) constitute a quad-tree feature histogram with $k = 1$ bins.

2. *Uniformity Feature.*

   When a range-block is approximated by a domain-block the spatial distance between the upper left corners can be calculated and divided by the maximum distance in the image. By splitting the interval $[0,1]$ into 8 equal intervals the above fractions are distributed over $k = 8$ different bins. Intuitively, the feature is able to detect whether one or more different textures are present in the image.

3. *Symmetry Feature.*

   The eight degrees of freedom in the affine transformations, see Section 2.1, yield $k = 8$ different feature-bins, constituted of rotations over $0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}$ with a possible flip along the second diagonal. Intuitively, this feature is able to determine whether a texture has one or more dominating directions.

4. *Contrast Feature.*

   To match the gray values of the range-blocks by the gray values of the domain-blocks, a scaling factor u_i is used, see Section 2.1:

   $$t_i \circ f(x,y) = u_i \cdot f(x,y) + o_i, \quad 0 < ||u_i|| < 1.$$

   The range of this scaling factor is divided into 8 intervals, which leads to $k = 8$ feature-bins for this feature. Intuitively, the feature relates to the homogeneity of the gray values within the image.

Figure 4 gives examples of typical quad-tree feature histograms.

4 Experiments and results

4.1 Introduction

Fractal feature extractors have been shown before to be effective for indexing multimedia database consisting of texture images [7, 9].

Here we demonstrate that the features as described above keep images distinguishable after they have been altered by either rotation, translation, brightness/contrast adjustment

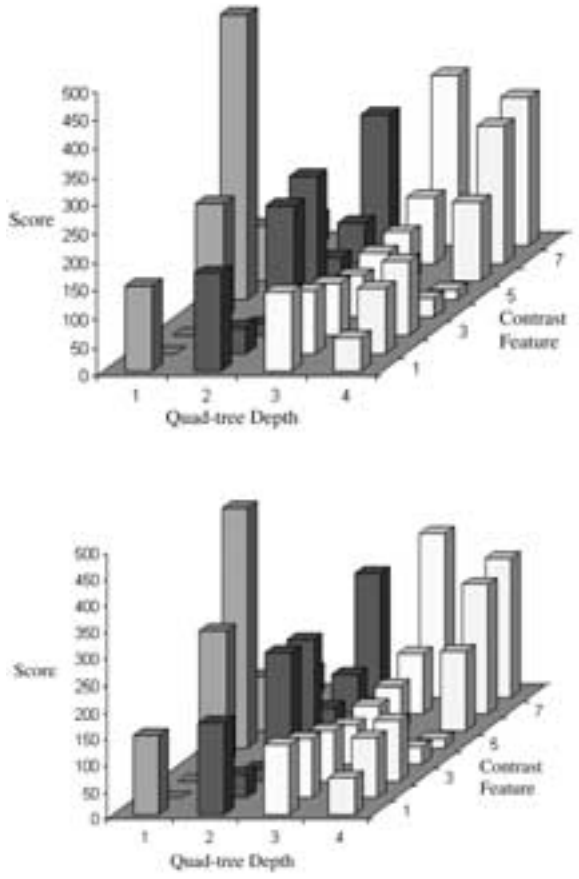

Figure 4. Two-dimensional quad-tree feature histograms (contrast feature) of a texture image from the database (top) and of an image of the same texture but folded (below).

or folding. Below, we describe the details of the numerical experiment: the method of comparison, the test-case and the presentation of results.

4.2 Method

Each perturbed image (total of $4N$) is presented to the database for a match. We introduce an invariance measure for features w.r.t. a database D. Let a database D count N images q:

$$q_i \in D, i = 1, \ldots, N.$$

Let $p^{(n)}$ be a perturbation: an operator that perturbs an image q into an image $p^{(n)}(q)$. The collection of all perturbations of the images, is denoted by P:

$$p^{(n)} \in P, n = 1, \ldots, M.$$

Before we match images according to their features, we have to define a metric on our feature space. A quad-tree feature histogram can be interpreted as a point in $\mathbb{R}^n$ with $n \equiv k(L - l + 1)$. The distance d between two quad-tree feature histograms is defined as the 2-norm of their distance in $\mathbb{R}^n$. For ease of notation we identify the image with its histogram. So $d(q_i, q_j)$ denotes the distance between the features (histograms) of image q_i and q_j.

We now define the *feature invariance measure* (FIM) with respect to the database D and perturbation $p \in P$:

$$\mu(D, p) = \sum_{i=1}^{N} \frac{d(p(q_i), q_i)}{\sum_{j=1}^{N} d(p(q_i), q_j)} \in \mathbb{R}. \qquad (8)$$

Obviously $\mu \geq 0$ by definition of d. Typically $\mu \approx 0$ if a feature is invariant to a perturbation p. If it turns out that $\mu \approx 1$, then apparently no invariance is observed (this is the value that can be expected "at random"). The larger μ, the less invariant is the feature for the specified perturbation p. An important advantage of the FIM is its (expected) insensibility with respect to the *dimension* of the database D. This is expected because (8) is equivalent to:

$$\mu_i = \frac{d(p(q_i), q_i)}{\overline{d(p(q_i), q_j)}^{j}}, i = 1, \ldots, N, \qquad (9a)$$

$$\mu(D, p) = \overline{\mu_i}^{i}, \qquad (9b)$$

where $\overline{\mu_i}^{i}$ is the average of μ_i over $i = 1, \ldots, N$, etc.

4.3 Test-case

In our experiments we had the following configuration:

1. $M = 4$, that is 4 perturbations, namely: rotation, translation, brightness/contrast adjustment (b.c.a.), folds.
2. $N = 52$, that is we have 52 textures in the database.

The computation of μ requires the evaluation of N^2 distances d for each perturbation. The actual images we used are extracted from the VisTex database (MIT), supplemented with images from the Brodatz collection. It contains gray-scale images of fabrics (originals). For an illustration, see Figures 1, 5 and 6. The images consist of 512×512 pixels and 256 gray levels. Four quad-tree levels were taking into account, varying in size from 2×2 to 64×64. For our experiments we used the original fractal coding algorithm of Fisher [4].

4.4 Results

In Table 2 we present the results for the feature invariance measure (8) with respect to the example database of Section 4.3. We use the four features explained (and la-

Table 2. Feature invariance measure

Feat.	Rotat.	Transl.	B.c.a.	Fold
1	$2.4_{\mathrm{E}}-3$	$5.3_{\mathrm{E}}-2$	$2.4_{\mathrm{E}}-1$	$2.3_{\mathrm{E}}-2$
2	$6.2_{\mathrm{E}}-2$	$1.4_{\mathrm{E}}-1$	$3.0_{\mathrm{E}}-1$	$9.6_{\mathrm{E}}-2$
3	$9.7_{\mathrm{E}}-2$	$1.5_{\mathrm{E}}-1$	$2.8_{\mathrm{E}}-1$	$1.4_{\mathrm{E}}-1$
4	$2.8_{\mathrm{E}}-2$	$1.2_{\mathrm{E}}-1$	$2.8_{\mathrm{E}}-1$	$7.3_{\mathrm{E}}-2$

beled 1–4) in Section 3.2. The set P of perturbations consists of rotation ($90°$), translation, brightness/contrast adjustment and folding. We observe an excellent invariance if images are perturbed by either rotation, folding or translation and a moderate invariance if images are perturbed by brightness/contrast adjustment. Table 3 shows how well a particular feature matches a perturbed image with its original in the database. We observe how some of the features

Table 3. Percentage of successful queries

Feat.	Rotat.	Transl.	B.c.a.	Fold
1	100.0	84.6	48.1	98.1
2	100.0	92.3	51.9	100.0
3	96.2	94.2	63.5	94.2
4	100.0	98.1	67.3	100.0

score an optimal performance of 100% for some of the perturbations. However, the results of Table 3 are expected to depend on the dimension of the database in contrast with the results of Table 2.

5 Discussion and future research

We introduced features based on statistics stemming from fractal decompositions of images. Further we introduced a feature invariance measure which reveals the degree of invariance of a feature with respect to a database and to a perturbation. For an example database the results for this measure show that features are highly invariant to perturbations by rotation, folding and translation and moderately invariant to brightness/contrast adjustment. Feature 4, the 'Contrast Feature' appears to be the most promising (see Table 3). As was conjectured before, the statistics of the fractal transform hardly change under a perturbation of the image. This fact is noteworthy. Remember that although the same image can result in two totally different fractal transform; the statistics of both transforms is shown to remain about the same.

If the features point images numerically out as close, then the images are visually close. As an example see Figure 5, all features of the left image point out that the right image is within a close range of similarity, which to the opinion of the authors is in accordance with visual perception.

The features didn't show any contradictory results; when an unperturbed (original) image was presented as a query for the database, all four features recognized the image without fault. As expected, the different features also ex-

Figure 5. Confusingly similar images.

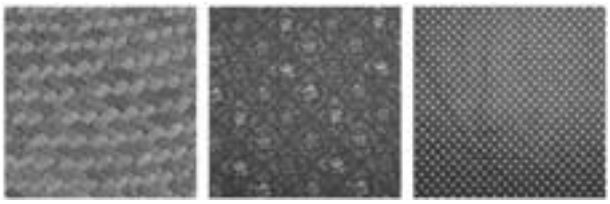

Figure 6. Different features express different aspects of similarity.

press different aspects of the texture present in the image. Take a look at Figure 6 for instance. For the eye of the beholder the left and middle image are similar when the direction of the texture is taken into account. The left and the right image are similar when a scale aspect of the image is taken into account. The features seem to have this power of discernment.

Perturbing an image by a small shift does have a huge impact on the fractal transform of the image in question. For an example let us consider Figure 5. The content of these images change significantly with a small shift. In many fractal feature-applications images are scanned several times over with small shifts to remedy this. The results of our method show that irrespective of the shift of the image, still the correct image is recognized.

The influence of folding the image to the statistics of the fractal transform appears small. Moreover, if a feature relates a folded image to the wrong original, then these two image are confusingly similar, as can be observed from the

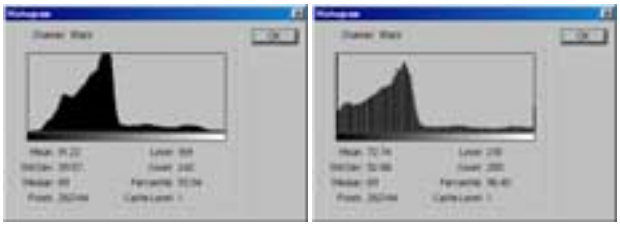

Figure 7. Gray-value histograms of a texture image from the database (left) and of the same image but with scaled contrast (right).

results. Hitherto, folding of the fabric has been digitally simulated, we plan to perform tests on real folded texture.

We also plan to perform tests on images perturbed by realistic gloss. In our experiments we merely used brightness/contrast adjustment of the gray-values instead. Let

$$f : \mathbb{N}^2 \rightarrow [0, ..., 255] \subset \mathbb{N}$$

represent the gray-values of the image. A contrast scaling on the gray values results in gray-values $f^{\text{scaled}} = cf(x, y)$, where c is constant over the entire image. As this scaling is global, one would not expect the search of range-blocks and matching domain-blocks to be disturbed. However, saturation may occur at either the lower or the higher end of the scale of gray values. This effect may disturb the similarity search, depending on the image. As an example we plot the histogram of gray values of an image and the histogram of the same image, but perturbed as a consequence of a contrast scaling, see Figure 7. It may well be that realistic gloss proves less troublesome, a topic for further investigation.

5.1 Future research

Presently we investigate whether, if an image has been altered by zooming in or out, the feature still recognizes the resulting image as similar, provided that we allow the histograms to translate along the axis of the quad-tree depth before comparison [10].

References

[1] M. Baldoni, C. Baroglio, D. Cavagnino, and L.Egidi. *Learning to Classify Images by Means of Iterated Function Systems*, pages 173–182. World Scientific Publishing Co. Pte. Ltd, Singapore, 1998.

[2] M. Barnsley. *Fractals Everywhere*. Academic Press, Inc., San Diego, USA, 1988.

[3] H.T. Chang, T.C. Chang, and K.L. Wen. Hierarchical fixed-point searchings method for resizing binary fractal image with ifs code modification. In *ICIP; Proceedings IEEE International Conference on Image Processing*, pages on CDROM, 1999.

[4] Y. Fisher. *Fractal Image Compression: Theory and Application*. Springer-Verlag, 1995.

[5] A. Jacquin. *A Fractal Theory of Iterated Markov Operators with Applications to Digital Image Coding*. Ph.D. Thesis, Georgia Institute of Technology, August 1989.

[6] A.Z. Kouzani, F. He, and K. Samut. Fractal face representation and recognition. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 1609–1613, 1997.

[7] J.M. Marie-Julie and H. Essafi. Image database indexing and retrieval using the fractal transform. In *ECMAST '97; Multimedia Applications, Services and Techniques*, pages 169–182. Springer, 1997.

[8] G. Neil and K.M. Curtis. Scale and rotationally invariant object recognition using fractal transformations. In *ICASSP; Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 6, pages 3458–3461, 1996.

[9] B.A.M. Schouten and P.M. de Zeeuw. Feature extraction using fractal codes. In *Visual 99; Visual Information and Information Systems, Third International Conference*, pages 483–492. Springer, 1999.

[10] B.A.M. Schouten and P.M. de Zeeuw. Scale and Fractal Transforms. In *ICIP-2000; Proceedings IEEE International Conference on Image Processing*, accepted for publication.

[11] T. Tan and H. Yan. Face recognition by fractal transformations. In *ICASSP; Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 6, pages 3537–3540, 1999.

[12] M. Vissac, J. Dugelay, and K. Rose. A fractals inspired approach to content-based image indexing. In *ICIP; Proceedings IEEE International Conference on Image Processing*, pages on CDROM, 1999.