Retrieving Faces by the PIFS Fractal Code

Sharat Chandran
Computer Science & Engg. Dept.
IIT Powai
Mumbai, INDIA 400076
sharat@umiacs.umd.edu

S. Kar
Computer Science and Engg. Dept.
IIT, Powai
Mumbai, INDIA 400076
kar@cse.iitb.ac.in

Abstract

The use of fractals in computer graphics and vision in modeling unstructured images, and compressing images, is well known. However, the use of fractals for indexing images in content based retrieval of video and static images has not received as much attention.

In this paper, we provide the theoretical arguments to justify their use in image retrieval. Our web-enabled implementation on the FERET database shows encouraging results.

1. Introduction

Given a library of reference images: $I_1, I_2, \ldots, I_n$, and a query image Q, we want to preprocess the reference images to produce indices such that we can find the 'closest' image I_i to Q.

In this paper, we consider the indexing problem for a class of images where it is possible to state fairly accurately the notion of a background and a foreground. Our experiments revolve around an important subset of this class, namely, photographs of humans (such as those used in corporate identity cards, or those clicked by an automatic teller machine camera). Unlike images generated under structured lighting conditions (such as those of nuts and bolts in factory plants), faces with facial and tonsural hair growth have a predominant texture. Traditional segmentation based techniques do not work well in such cases, and many interesting [2, 12] approaches fail.

Fractals are important mathematical entities that have the ability to represent natural unstructured entities such as face, hair, and trees against the background in a photograph. Fractal descriptors are also compact, and therefore, have been used for compression. Indeed, the fractal subdivision method of chopping an image may be viewed as an automatic segmentation algorithm.

The biggest impediment in using fractal descriptors for indexing is the one-to-many relationship between an image and fractal descriptors. Many descriptors can converge (using the fractal paradigm, made precise in Section 3) to the same image. In this paper, we study the use of fractal indices for general image indexing, and exemplify it with faces as the domain. Note that no assumption is made of "zeroing background" unlike approaches such as the venerable eigenfaces [11].

1.1. Our Contributions

- **Theory:** Although it has been well known that there are fixed points for fractal descriptors, the inverse problem has not been studied. We prove the existence of fractal descriptors W which have the following property: Given two images I_1 and I_2 there exists W_1 and W_2 such that if the images are "close," then so are the (compact) descriptors. This result is *different* from the proof in [1].
- **Implementation:** Using canonical descriptors, we have developed a fractal based image indexing system that works (See Section 4) well. The sample domain of our work are in facial images from the FERET [10] database, but we have shown it to work in other domains also.

The rest of this paper is organized as follows. In the next section, we summarize previous work in this area. In Section 3 we provide the theoretical background for this work. In Section 4 our implementation is discussed along with sample results. Final remarks are made in Section 5.

2. Related Work

Although the Iterated Function System (IFS) structure of fractal object representation has been utilized by many researchers for the purpose of image compression, there have been very few attempts directed towards object indexing or recognition. In [4] the authors have presented a somewhat restricted recognition scheme applicable to the specific domain of L-System fractals and tested their technique on binary synthetic plant images generated by the L-System.

In [8] a recognition method is suggested which (i) works on binary images, and (ii) which is based on applying the reference set of Partitioned Iterated Function System (PIFS) codes on the query object and finding out the code which produces minimum change. The query is then recognized to be the object corresponding to that PIFS code, if the change found out is less than a threshold. This technique is not very interesting for the indexing problem because the comparison happens in the image domain, and not in the domain of indices.

In [7] the authors present a technique of indexing and content-based retrieval by a set of PIFS fractal parameters without dealing with the theory of proximity of PIFS codes for visually similar images as established in this paper. In addition, the time required for indexing and retrieval of an image by the method suggested in [7] is larger as compared to that of our method.

The system proposed in [3] has the interesting property of being invariant under two classes of pixel intensity transformations: illumination or color alterations. The system can be used both by sketches, and the query by example paradigms. As seen in Section 4 and Figure 4, our system is more tolerant to semantic content changes.

In [13], a joint fractal coding technique is used for image retrieval, and compared with wavelet coding. They conclude that wavelet transform approach performs more effectively in content-based similarity comparison on those images which contain strong texture features, whereas fractal coding approach performs relatively more uniformly well for various type of images. Unlike our experiments on faces, the conclusions are drawn on synthetic Broadatz texture images. The use of a joint fractal coding is different from our canonical coding.

3. Theoretical Basis

In this section, we first introduce a few definitions, lemmas and theorems. The notion of complete and compact metric spaces, and the Hausdorff metric h are as formulated in [1]. These are presented in the context of any set, and therefore are applicable to images when viewed as sets.

Definition 3.1 *A transformation $w : \Re^2 \rightarrow \Re^2$ of the form $w(x, y) = (ax + by + e, cx + dy + f)$, where a, b, c, d, e and f are real numbers is called a 2-D affine linear transformation. The numbers a, b, c, d, e, and f are called the parameters of the transformation.*

Definition 3.2 *A transformation $w : X \rightarrow X$ on a metric space (X, d) is called contraction mapping if there is a constant $0 \leq s < 1$ such that $d(w(x), w(y)) \leq sd(x, y), \forall x, y \in X$. Any such number s is called a contractivity factor (CF) for w.*

Definition 3.3 *An iterated function system (IFS) consists of a complete metric space (X, d) together with a finite set of contractive mappings $w_n : X \rightarrow X$, with respective contractivity factors $s_n, n = 1, 2, 3, ..., N$. The notation used for the IFS code is $W = \{X; w_n, n = 1, 2, ..., N\}$ and its CF is $s = \max\{s_n : n = 1, 2, ..., N\}$. The set of parametric values for all the individual maps taken together is called the parameter of W.*

Table 1 shows the IFS code for a ‘fern’ leaf using the notations of affine transformation as in Definition 3.1. IFS codes are compact and converge to an image (also known as a fixed point) from which they are produced (see Figure 1(a)).

a	b	c	d	e	f
0.00	0.00	0.00	0.16	0.00	0.00
0.85	0.04	-0.04	0.85	0.00	1.60
0.20	-0.26	0.23	0.22	0.00	1.60
-0.15	0.28	0.26	0.24	0.00	0.44

Table 1. The IFS code for a fern: A few numbers generate a nice image.

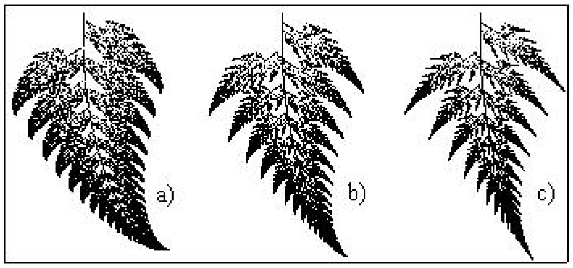


Figure 1. (a) An image obtained as the fixed point of the code in Table 1. (b) Another image obtained from the same code with the parameter a(1) changed to 0.80. (c) Same as (b) with a(1)=0.75.

IFS codes are also well behaved as illustrated in Figure 1. This result may be stated formally as follows. Let (P, d_p) and (X, d) be metric spaces. Let $w : P \times X \rightarrow X$ be a family of contraction mappings on X with contractivity factor $0 \leq s < 1$. That is, for each $p \in P$, $w(p, .)$ is a contraction mapping on X. For each fixed $x \in X$, let w be continuous on p. Then the fixed point of w depends continuously on p. That is $x_f : P \rightarrow X$ is continuous.

Unfortunately, IFS codes are not unique as shown in Figure 2.

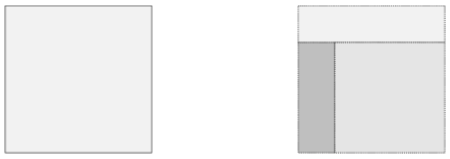


Figure 2. Two different contractive mapping may be obtained for a square, shown on the left. The obvious first one is to use four squares each of one fourth area that tile the rectangle. The second mapping is to use three rectangles shown in the picture.

Whereas IFS deals with images that are self similar, the generalized notion of the Partitioned IFS is important for most images (anisotropic and non homogeneously scaled) that one deals with in real life. The key notion (formalized below) is to look at subsets of the image, rather than the entire image.

Definition 3.4 *Let (X, d) be a compact metric space. Let D be a nonempty subset of X. Let $w : D \to X$ and let s be a real number with $0 \leq s < 1$. If $d(w(x), w(y)) \leq sd(x, y), \forall x, y \in D$, then w is called a local contraction mapping on (X, d) with CF s. If there is a set of such w's written as w_i with CFs s_i, for $i = 1, 2, ..., N$, then $\{w_i : D_i \to X; i = 1, 2, ..., N\}$ is called a partitioned iterated function system (PIFS). The number $s = \max\{s_i : i = 1, 2, ..., N\}$ is the CF of the PIFS.*

3.1. Is Indexing by PIFS correct?

All the above notions are self contained and apply to any set. These definitions are mapped to gray level images (for example, see [5, 6]).

A PIFS $\{\mathbf{w}_i\}$ for an image Ψ is defined by $\mathbf{w}_i(x, y, \Psi) = (w_i(x, y), v_i(\Psi))$, where v_i maps the intensity and is of the form $v_i(x) = s_i x + o_i$. s_i performs a contrast scaling and o_i a luminance shift.

Definition 3.5 *The parameters of w_i along with the s_i and o_i factors of v_i underlying $\mathbf{w}_i$ are called the parameters of $\mathbf{w}_i$. The parameters of all the $\mathbf{w}_i$'s constituting a PIFS code are called the parameters of the code.*

Definition 3.6 *d_w is a metric to compute the distance between two PIFS codes by summing up the term-wise absolute differences between the parameters of the two codes.*

The entire theoretical framework discussed so far is needed for formally stating the following lemma which makes indexing by PIFS possible:

Lemma 3.1 *Given two images I_1 and I_2 with distance $d(I_1, I_2)$ under the metric d, and the PIFS code $\mathbf{W}_1$ (with parameter $p1$) approximating I_1 with error ϵ_1, there exists a PIFS $\mathbf{W}_2$ (with parameter $p2$) encoding I_2 with error ϵ_2, such that*

$$d_w(\mathbf{W}_1, \mathbf{W}_2) \leq k_1 d(I_1, I_2), \tag{1}$$

and

$$d(I_1, I_2) \to 0 \wedge \epsilon_1 \to 0 \Rightarrow \epsilon_2 \to 0 \wedge d_w(\mathbf{W}_1, \mathbf{W}_2) \to 0, \tag{2}$$

where k_1 is a constant.

Lemma 3.1 establishes the continuity of the parameters of the PIFS codes over the *encoded* images, under a metric d_w chosen to measure the distance in the PIFS code domain. Figure 3 depicts three faces; the first two faces are of the same person whereas the third face is of a different person. There exists PIFS codes of these three faces where the distance of the code for the first face from that of the second is small because of the continuity of the parameters of the PIFS codes over the encoded images; whereas the distance of the code for the first face from that of the third is large.

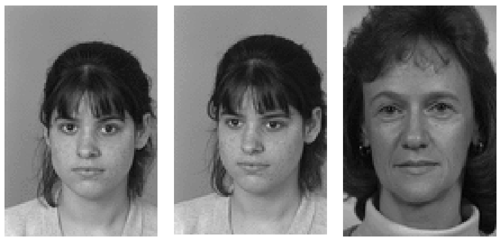


Figure 3. Not all PIFS codes generated for the first two figures are close to each other (when compared to that of the third), but there do exist canonical codes which satisfy this property.

It is important to note that for a given image, it is possible to generate more than one PIFS code; but for the purpose of object recognition, it is enough if we obtain a *canonical* PIFS code in a manner that adheres to the proof in [6]. A collection of several canonical PIFS codes generated for a set of reference images constitute the database for carrying out object indexing.

4. Results

We have tested the PIFS based object recognition method on more than 1300 human face images. On our system, one can present a query, for example, by selecting randomly generated images from the server using a standard browser.

Once a query is submitted, by clicking with a mouse on an image on the screen (which forms part of our database), top matching several images are displayed. The exact number of retrieved images is decided by a dynamic thresholding algorithm applied on the matching score. The retrieval time is about 2 seconds (on the server software running on a low end workstation) for about 1000 images. For even larger databases, a standard multilevel indexing approach may be used to prune the database. Figure 4 shows the result of sample queries. The leftmost column in this image depicts the query images. The top 5 matched images (the query itself is always the highest match) are included in the row corresponding to the query, in descending order of matching score.

The graph in Figure 5, based on 50 random queries drawn on a subset of the FERET database, depicts the error in our system. The graph shows the fraction of inappropriate results (meaning a different (as judged by a human observer) person's face is retrieved). A value of 0 indicates that all retrieved results matched the query.

In summary, Figure 5 depicts the performance of our system quantitatively and Figure 4 qualitatively. This data shows that our system satisfies the level of accuracy on mugshot-like images as suggested in [9].

More objective results are reflected in the graphs in Figure 6 and Figure 7 on the FERET database. Figure 6 corresponds to an experiment conducted on 50 random queries where the intensity values of the original images were changed by 20% and 40%. Figure 7 corresponds to an experiment based on 70 random queries where the pose angle of the subject was varied.

Our other experiments indicate that when there are translations (of upto 20%), rotation (few degrees) as well as intensity fluctuation (upto 2%) with respect to the query, the responses are always correct (the top few). Other examples are Figure 8(a) and Figure 8(b) which show invariance to content and scale changes (upto 10%).

5. Conclusion

We have described an image indexing scheme based on PIFS fractal codes of gray level images. The parameters of our canonical PIFS code are continuous over the encoded images. By measuring the distance between the parameters of the reference PIFS codes kept in a library and the parameters of the code of a query image, and by checking whether the minimum of such distances is less than an algorithmically determined threshold, we can index the query image.

It is important to note that the PIFS code for an image, and the image itself do not have a one-to-one correspondence. However, for indexing, it is enough to be able to extract a canonical PIFS for a given image. Our recognition method can index objects which are unstructured in general. For faces, this means that we do not need to "know" the background unlike the eigenfaces approach which uses, for example in FERET database, the location of eyes. Our approach is computationally less intensive as compared to many other methods.

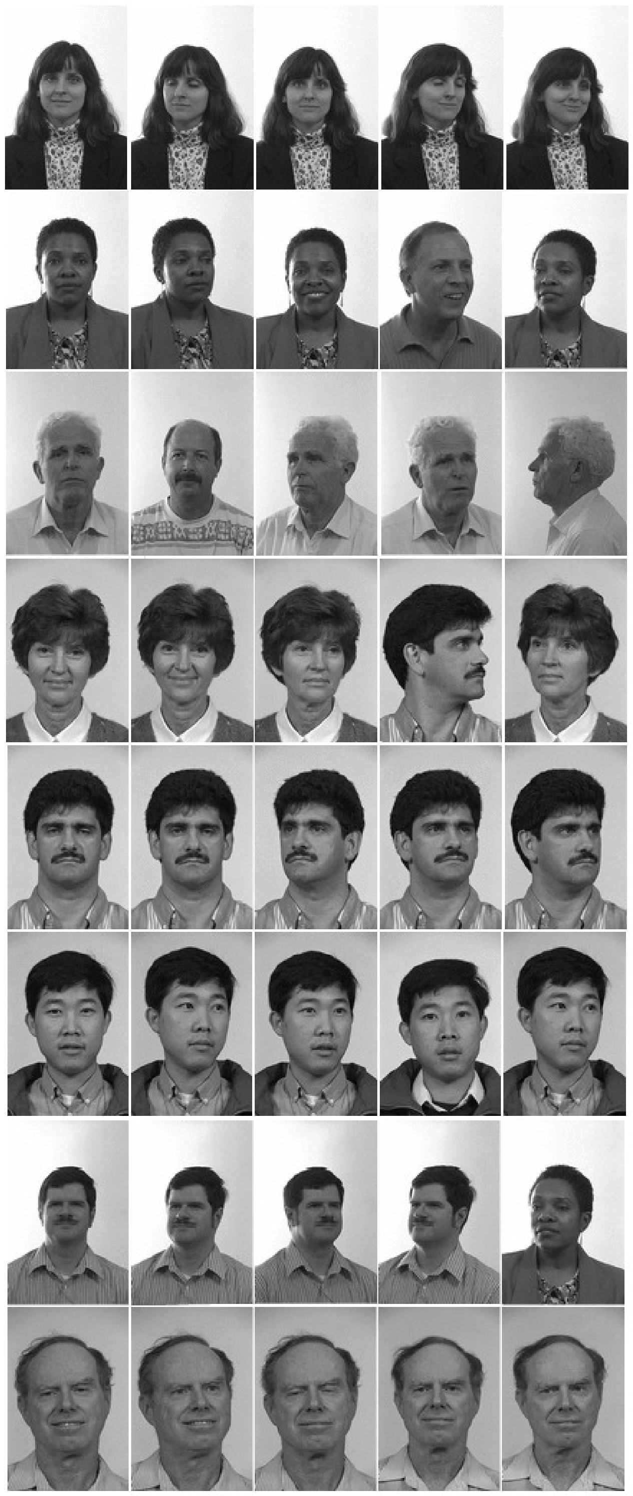


Figure 4. Sample queries and retrieved results from our system on the FERET database.

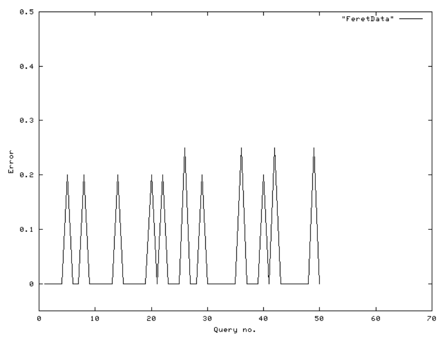


Figure 5. The error in our system. A number close to zero is good.

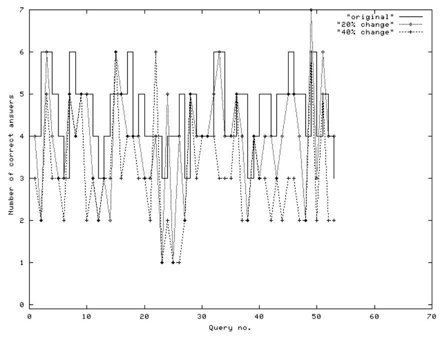


Figure 6. The number of correct results in our system when the database is modified to contain synthetically generated images. The '0%' trend corresponds to queries when images are not modified. The '20%' and '40%' trend curves do not shift too much (average of 0.85 and 1.5) from the '0%' curve, a reflection of robustness with respect to intensity variation.

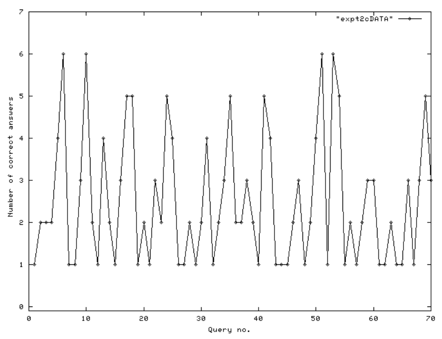
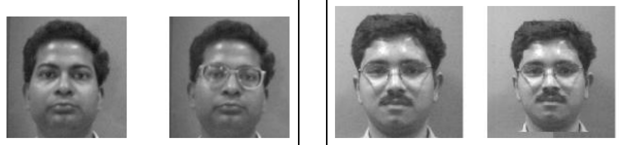


Figure 7. The number of correct results when the database is modified to contain pose variant images (like the first two images of Figure 3).



(a) Two images of one person with and without spectacles.

(b) Two images of the same person acquired at different scales.

Figure 8. Experiment for content change and scale change.

References

[1] M. Barnsley. *Fractals Everywhere*. Academic Press Professional, 1993.

[2] C. Carson, M. Thomas, S. Belongie, J. Hellerstein, and J. Malik. Blobworld: a system for region-based image indexing and retrieval. In *Third International Conference on Visual Information Systems*, pages 509–516, 1999.

[3] M. Distasi, M. Nappi, and M. Tucci. Using fractal encoding for image indexing. In *Universit di Salerno Proceedings of the 10th International Conference on Image Analysis and Processing: ICIAP*, 1999.

[4] D. Holliday and A. Samal. Object recognition using L-system fractals. *Pattern Recognition Letters*, 16:33–42, 1995.

[5] A. Jacquin. Image coding based on fractal theory of iterated contractive image transformations. *IEEE Transactions on Image Processing*, 1:18–30, 1992.

[6] S. Kar. *Indexing Images for Unstructured Object Recognition*. PhD thesis, Indian Institute of Technology, 2001.

[7] J. Marie-Julie and H. Essafi. Digital image indexing and retrieval by content using the Fractal transform for multimedia databases. In *Proceedings of the 4th International Forum on Research and Technology Advances in Digital Libraries*, pages 509–516, 1997.

[8] G. Neil and K. Curtis. Shape recognition using fractal geometry. *Pattern Recognition*, 30:1957–1969, 1997.

[9] A. Pentland and T.Choudhury. Face recognition for smart environments. *IEEE Computer*, pages 50–55, 2000.

[10] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss. The FERET evaluation methodology for face-recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1090–1104, 2000.

[11] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neurosciences*, pages 71–86, 1991.

[12] J. Wang, J. Li, and G. Wiederhold. Simplicity: Semantics-sensitive integrated matching for picture libraries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:947–963, 2001.

[13] A. Zhang, B. Cheng, R. Acharya, and R. Menon. Comparison of wavelet transforms and fractal coding in texture-based image retrieval. *In G.G. Grinstein and R.F. Erbacher, editors, Visual Data Exploration and Analysis III, volume 2656 of SPIE Proceedings*, pages 116–125, 1996.