A Review Fractal Image Compression

Manjot Kaur, Gagan kumar

Department of Computer Science and Engineering, Modern Institute of engineering & technology, Mohri, Kurukshetra , Haryana, India

Abstract: The demand for images, video sequences and computer animations has increased drastically over the years. This has resulted in image and video compression becoming an important issue in reducing the cost of data storage and transmission. JPEG is currently the accepted industry standard for still image compression, but alternative methods are also being explored. Fractal Image Compression (FIC) is one of them. This scheme works encoding by partitioning an image into blocks and using Contractive Mapping to map range blocks to domains. The encoding step in fractal image compression has high computational complexity whereas, decoding step involves starting from all zeros image to achieve final image which is same as original image by applying self Transformations.

Keywords: Image processing, Image Compression, Digital Image

I. INTRODUCTION

With the advance of the information age the need for mass information storage and fast communication links grows. Storing images in less memory leads to a direct reduction in storage cost and faster data transmissions. These facts justify the efforts, of private companies and universities, on new image compression algorithms.

Images are stored on computers as collections of bits (a bit is a binary unit of information which can answer “yes” or “no” questions) representing pixels or points forming the picture elements. Since the human eye can process large amounts of information (some 8 million bits), many pixels are required to store moderate quality images. These bits provide the “yes” and “no” answers to the 8 million questions that determine the image. Most data contains some amount of redundancy, which can sometimes be removed for storage and replaced for recovery, but this redundancy does not lead to high compression ratios. An image can be changed in many ways that are either not detectable by the human eye or do not contribute to the degradation of the image.

The standard methods of image compression come in several varieties. The current most popular method relies on eliminating high frequency components of the signal by storing only the low frequency components (Discrete Cosine Transform Algorithm). This method is used on JPEG (still images), MPEG (motion video images), H.261 (Video Telephony on ISDN lines), and H.263 (Video Telephony on PSTN lines) compression algorithms.

Fractal Compression was first promoted by M.Barnsley, who founded a company based on fractal image compression technology but who has not released details of his scheme. The first public scheme was due to E.Jacobs and R.Boss of the Naval Ocean Systems Center in San Diego who used regular partitioning and classification of curve segments in order to compress random fractal curves (such as political boundaries) in two dimensions. A doctoral student of Barnsley’s, A. Jacquin, was the first to publish a similar fractal image compression scheme [2].

Fractal compression is a lossy image compression method using fractals to achieve high levels of compression. The method is best suited for photographs of natural scenes (trees, mountains, ferns, clouds). The fractal compression technique relies on the fact that in certain images, parts of the image resemble other parts of the same image. Fractal algorithms convert these parts, or more precisely, geometric shapes into mathematical data called "fractal codes" which are used to recreate the encoded image. Fractal compression differs from pixel-based compression schemes such as JPEG, GIF and MPEG since no pixels are saved. Once an image has been converted into fractal code its relationship to a specific resolution has been lost; it becomes resolution independent. The image can be recreated to fill any screen size without the introduction of image artifacts or loss of sharpness that occurs in pixel-based compression schemes.

II. WHAT IS FRACTAL IMAGE COMPRESSION?

Imagine a special type of photocopying machine that reduces the image to be copied by half and reproduces it three times on the copy (see Figure 2). What happens when we feed the output of this machine back as input? Figure 3 shows several iterations of this process on several input images. We can observe that all the copies seem to converge to the same final image, the one in 3(c). Since the copying machine reduces the input image, any initial image placed on the coping machine will be reduced to a point as we repeatedly run the machine; in fact, it is only the position and the orientation of the copies that determines what the final image looks like.

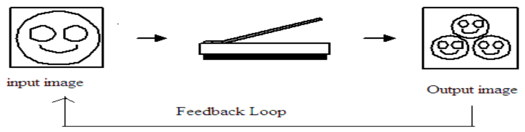


Figure 1: A copy machine that makes three reduced copies of the input image

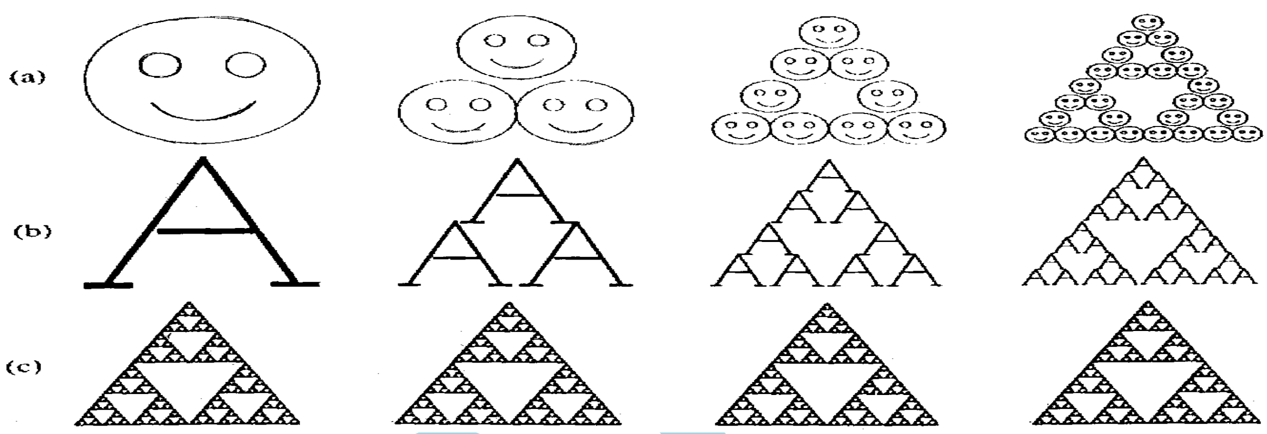
The way the input image is transformed determines the final result when running the copy machine in a feedback loop. However we must constrain these transformations, with the limitation that the transformations must be contractive, that is, a given transformation applied to any two points in the input image must bring them closer in the copy. This technical condition is quite logical, since if points in the copy were spread out the final image would have to be of infinite size. Except for this condition the transformation can have any form.

In practice, choosing transformations of the for

$$t_i\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \end{bmatrix} \qquad (1)$$

is sufficient to generate interesting transformations called affine transformations of the plane. Each can skew, stretch, rotate, scale and translate an input image.

A common feature of these transformations that run in a loop back mode is that for a given initial image each image is formed from a transformed (and reduced) copies of itself, and hence it must have detail at every scale. That is, the images are fractals. This and more information about the various ways of generating such fractals can be found in books by Barnsley and Yuval Fisher [4].



Initial copy First Copy Second copy Third Copy

Figure 2. The first three copies generated on the copying machine figure 2

Barnsley suggested that perhaps storing images as collections of transformations could lead to image compression. His argument went as follows: the image in Figure 3 looks complicated yet it is generated from only 4 affine transformations.

Each transformation *wi* is defined by 6 numbers, *ai, bi, ci, di, ei,* and *fi* , see eq(1), which do not require much memory to store on a computer (4 transformations x 6 numbers / transformations x 32 bits /number = 768 bits). Storing the image as a collection of pixels, however required much more memory (at least 65,536 bits for the resolution shown in Figure 3). So if we wish to store a picture of a fern, then we can do it by storing the numbers that define the affine transformations and simply generate the fern whenever we want to see it. Now suppose that we were given any arbitrary image, say a face. If a small number of affine transformations could generate that face, then it too could be stored compactly. The trick is finding those numbers.

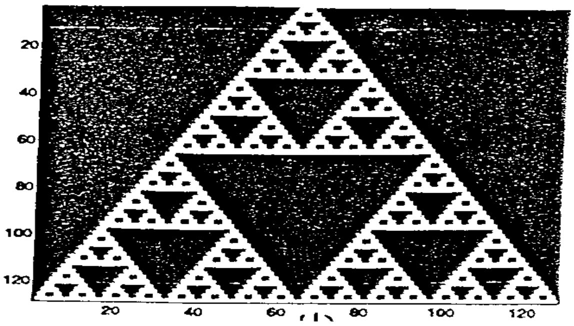


Figure 3: Fractal Fern

III. CONTRACTIVE TRANSFORMATIONS

A transformation *w* is said to be contractive if for any two points *P1, P2*, the distance

d(t(p1),t(p2)) < sd(p1,p2),

for some *s* < 1, where d = distance. This formula says the application of a contractive map always brings points closer together (by some factor less than 1).

IV. CONTRACTIVE MAPPING FIXED POINT THEOREM

This theorem says something that is intuitively obvious: if a transformation is contractive then when applied repeatedly starting with any initial point, we converge to a unique fixed point.

If *X* is a complete metric space and $W: X \rightarrow X$ is contractive, then *W* has a unique fixed point |*W*|.

This simple looking theorem tells us how we can expect a collection of transformations to define an image.

V. ITERATED FUNCTION SYSTEMS (IFS)

IFS is the term originally devised by Michael Barnsley [2] for a collection of contraction mappings over a complete metric space, typically compact subsets of R^n. Systems of contraction mappings had been considered previously by a number authors for various purposes. The Barnsley showed how such systems of mappings with associated probabilities could be used to construct fractal sets and measures: the former from a geometric measure theory setting and the latter from a probabilistic setting.

Just as important, however, was the fact that the Barnsley paper was the first to suggest that IFS could be used to approximate natural objects. This was the seed of the **inverse problem of fractal approximation**: Given a "target" set, S, for example, a leaf, can we find an IFS with attractor A that approximates S to a reasonable degree.

In a subsequent paper, Barnsley showed how the inverse problem of fractal approximation was could be reformulated by means of the infamous "Collage Theorem": instead of trying to find IFS whose attractor A would match the target S (a very tedious and difficult problem), one can look for IFS that maps A as close as possible to itself.

VI. PIFS FRACTAL IMAGE ENCODING

The generic type of PIFS fractal encoding for gray images is introduced [3], which follows the next steps.

1. The gray image to be encoded is partitioned into non-overlapping range blocks denoted by R_i of size N X N, and is partitioned into overlapping domain blocks denoted by D_i of size 2N X 2N. R_i is an encoding cell. All domain blocks form the domain pool S_D.

2. According to the least-square error criteria, R_i searches for its best matching block in S_D. Namely, D_i approximates to R_i through contraction affine transform ω_i - ω_i is composed of three transformations. The first is geometric scaling mapping making R_i and D_i have same space size. The second is reflection or rotation transformations. Saupe [39] indicates that without the second transformations, better quality of the reconstructed image can also be obtained by reducing the domain blocks' partitioning step, and the compression ratio is improved due to never saving the transformations information. The third is gray level transform G defined as follows:

$\hat{R}_i = G(D_i) = s_i . D_i + \sigma_i$

Where s_i is scaling coefficient and σ_i is luminance offset. Δ is the least-square error between $\hat{R}_i$ and R_i . L is the matching error threshold. If $\Delta < L$, this domain block is marked as a matching block of R_i. Then we select another domain block in S_D and do step 2 repeatedly, until all domain blocks in S_D have been compared with R_i. The domain block with minimum Δ is regarded as the best matching block of R_i. If the matching block doesn't exist, R_i will be divided into 4 equal sized sub-images, and they are encoded respectively [32].

3. Save s_i, σ_i and the relative positions r_x , r_y between R_i and its best matching block as encoding information.

Repeat the above step to encode other range blocks.

The image compression scheme describe later can be said to be fractal in several senses. The scheme will encode an image as a collection of transforms that are very similar to the copy machine metaphor. Just as the fern has detail at every scale, so does the image reconstructed from the transforms. The decoded image has no natural size, it can be decoded at any size. The extra detail needed for decoding at larger sizes is generated automatically by the encoding transforms. One may wonder if this detail is "real"; we could decode an image of a person increasing the size, with every iteration, and eventually see skin cells or perhaps atoms. The answer is, of course, no.

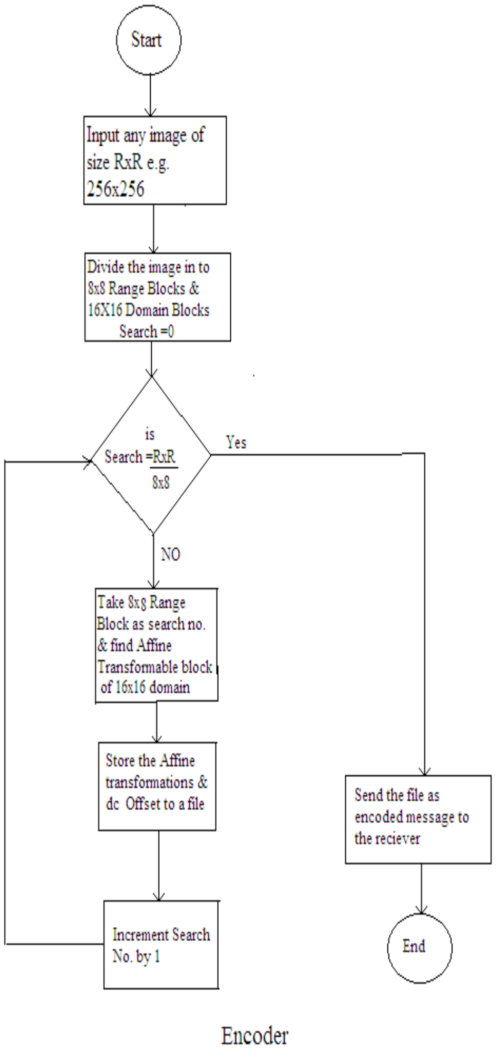

Start
Input any image of size RxR e.g. 256x256
Divide the image in to 8x8 Range Blocks & 16X16 Domain Blocks Search =0
is Search =RxR / 8x8
Yes
NO
Take 8x8 Range Block as search no. & find Affine Transformable block of 16x16 domain
Store the Affine transformations & dc Offset to a file
Increment Search No. by 1
Send the file as encoded message to the reciever
End
Encoder

Fig 5: Fractal Image Encoding Block diagram

VII. WHY THE NAME "FRACTAL"

The detail is not at all related to the actual detail present when the image was digitized; it is just the product of the encoding transforms which originally only encoded the large-scale features. However, in some cases the detail is realistic at low magnifications, and this can be useful in Security and Medical Imaging applications.

VIII. HOW MUCH COMPRESSION CAN FRACTAL ACHIEVE?

The compression ratio for the fractal scheme is hard to measure since the image can be decoded at any scale. For example, the decoded image in Figure 4 is decoded at 4 times its original size, so the full decoded image contains 16 times as many pixels and hence this compression ratio is 91.2:1. This many seem like cheating, but since the 4-times-later image has detail at every scale, it really is not.

ADVANTAGES

- High compression ratio
- High reconstruction quality

- Resolution independent for decoded image
- Magnify without losing details

DISADVANTAGES

- High computation load
- Long coding & decoding time
- Iterative decoding
- Difficult for hardware design

APPLICATIONS:

This type of compression can be applied in Medical Imaging, where doctors need to focus on image details, and in Surveillance Systems, when trying to get a clear picture of the intruder or the cause of the alarm. This is a clear advantage over the Discrete Cosine Transform Algorithms such as that used in JPEG or MPEG.

IX. CONCLUSION

Although fractal image coding is a relatively new technique, it has acquired a performance comparable with other methods such as JPEG or vector quantization. Furthermore, the field of research is far from being exhausted since there are many directions that have not yet been fully investigated (e.g., the use of non-affine transformations, the combination of fractal coding with other techniques and extensions to volume data and video frames). The main advantages of the fractal compression scheme are its ability to provide high compression ratios for a large class of images, the speed of its decoding process and its multi-resolution properties. However, to arrive at an optimal algorithm which can outperform traditional techniques, more attention needs to be devoted to the encoding process which still suffers from long computation times.

REFERENCE

[1] A. Jacquin, “Image coding based on a fractal theory of iterated contractive image transformations,” IEEE Trans. Image Processing, vol. 1, pp. 18–30, Jan. 1992.

[2] M. F. Barnsley and A. Jacquin, “Application of recurrent iterated function systems to images,” Proc. SPIE, vol. 1001, pp. 122–131, 1988.

[3] A. Jacquin, “Fractal image coding based on a theory of iterated contractive image transformations,” Proc. SPIE Conf. Visual Communications and Image Processing, pp. 227–239, 1990.

[4] Y. Fisher, “Fractal image compression with quadtrees,” in Fractal Compression: Theory and Application to Digital Images, Y. Fisher, Ed. New York: Springer-Verlag, 1994.

[5] D. Saupe, “Accelerating fractal image compression by multidimensional nearest neighbor search,” in Proc. Data Compression Conf., Snowbird, UT, Mar. 1995, pp. 222–231