

# Image Compression Based on Fractal Properties: Fractal Image Compression Optimization Methods

Corina Săraru\*

Faculty of Mathematics and Computer Science Department of Computer Science,  
University of Pitești, Ale. Dimitrie Cantemir, nr.2, Bl. P8, Ap. 14, Pitești, Argeș, ROMANIA  
(Received 3 January, 2008)

The Fractal Image Compression (FIC) involves the idea of resolution independence of the compression rate. Though, the amount of time spent to apply such a method makes it difficult to be implemented in practice. There are several methods which optimize the process. Among them, one can find interesting the compression using Kohonen self-organizing neural networks, hash like compression and compression made using the "yardstick travelling" method. These methods will be reviewed here.

**PACS numbers:** 07.05.Pj, 07.05.Mh

**Keywords:** fractal image compression, neural networks

## 1. Introduction

The most popular way to compress an image is the JPEG (Joint Photographic Experts Group) standard; the mathematics behind it is done through the Discrete Cosine Transform. The method consists in eliminating atomic components of the image (pixels) which have high frequency in some areas. If too many pixels are eliminated, the image becomes visibly modified.

Also, the JPEG standard is resolution dependent. This means that zooming in a part of an image (this implies multiplying pixels) will create inaccurate portions of the image as the zoom process increases rate.

Inspired by Mandelbrot's *The Fractal Geometry of Nature*, M. Barnsley emphasized the possibility of compressing images through the use of fractals. His idea was based on using IFS (Iterated Function System) for describing the image that was to be compressed as a fractal.

Arnaud Jacquin, one of Barnsley's PhD students, had the idea of partitioning the image in ranges for which to find a different IFS (that is, to find similar parts of the image within itself and not necessarily resembling to the whole one). Therefore, every part of the image would have been better represented and compressed. The instrument he used is known under the name of PIFS (Partitioned Iterated Function System).

## 2. Representation of an image

For representing an image from a mathematical point of view, one needs to consider the description of the pixels composing the image. For a black-and-white image, the description will contain the coordinates of pixels in the image, and the intensity of gray they own. The natural extension to colored images implies that

the color of the pixel is described instead of the gray value. This can be achieved by using the RGB (Red Green Blue) system and representing the three values that compose the color.

Considering the next mapping defined by

$$f : [a, b] \times [a, b] \rightarrow \{0, 1, \dots, N - 1\},$$

$$[a, b] \subset \mathbb{R}, f(x, y) = z \tag{1}$$

as the mathematical representation of an image, then  $z$  is the gray level of the current pixel, while  $x$  and  $y$  are its coordinates in the image.

If the domain of the mapping is discrete, the image is called digital.

$N$  is considered to be 256.

In practice, an image is represented by a matrix:

$$(f(x_i, y_j))_{i=\overline{1, n}, j=\overline{1, m}}, n, m \in \mathbb{Z}^+ \tag{2}$$

## 3. The algorithm of fractal image compression

The algorithm of fractal image compression assumes the use of Arquin's idea.

The image will be divided into a partition of blocks called range blocks and denoted by  $R_i (i \in \{1, \dots, M\})$  is the number of the current block and  $M$  is the number of blocks of the partition):

$$\bigcup_i R_i = [a, b]^2, R_i \cap R_j = \phi, \forall i \neq j \tag{3}$$

Then, another block taken from the image is considered (called domain block and denoted by  $D_j, j \in \{1, \dots, K\})$  such as, for some affine transformation  $\tilde{T}_i$  the relation

$$R_i = \tilde{T}_i(D_j) \tag{4}$$

\*E-mail: corina\_sararu@yahoo.com

is valid.

$\tilde{T}_i$  may have the following expression:

$$\tilde{T}_i(x, y) = A_i(x, y)^T + b_i \quad (5)$$

where  $A_i$  is a non-singular matrix and  $b_i$  is a bi-dimensional not null vector.

For  $T_i$  it is necessary to determine the transformation

$$T_i(f)(x, y) = c_i f(\tilde{T}_i^{-1}(x, y)) + o_i, \forall (x, y) \in R_i \quad (6)$$

which maps the domain block  $D_j$  into the range block  $R_i$ ; the difference is emphasized through contrast ( $c_i$ ) and brightness ( $o_i$ ).

In order to determine the similarity between the two types of blocks, it is necessary to emphasize the notion of distance between two images (the blocks can be viewed then as smaller images).

For two digital images  $f$  and  $g$  one can use in this purpose the following formula:

$$d(f, g) = \left( \sum_{i=1}^n \sum_{j=1}^m (f(x_i, y_j) - g(x_i, y_j))^2 \right)^{\frac{1}{2}} \quad (7)$$

If  $f \in \mathbf{F}$  is a function representing an image (also called image function) and if  $T_f$  is a contraction on the space of the image functions  $\mathbf{F}$ , such that  $T_f(f) = f$  then  $\exists n : d(T_f^{(n)}(g), f) < \varepsilon$ , where  $\varepsilon > 0, g \in \mathbf{F}$  ([1]).

That means that if one applies the  $T_f$  transformation to some image  $g$  enough times, then one can bring it close to  $f$  with accuracy  $\varepsilon$ . When using range blocks to divide the entire image, the relation above can be adapted in this manner (using the Collage theorem):

$$d(f(x, y), c_k f(\tilde{T}_k^{-1}(x, y)) + o_k) < \varepsilon, \quad \forall (x, y) \in R_k \quad (8)$$

The fractal image compression algorithm implies the next steps:

1. Partition the image into range blocks:

$$\bigcup_i R_i = [a, b]^2, R_i \cap R_j = \emptyset, \forall i \neq j \quad (9)$$

2. Construct the domain blocks of different sizes:

$$\bigcup_i D_i \subseteq [a, b]^2 \quad (10)$$

3. - For each range block, search a domain, an affine transformation, a contrast and brightness range such that the transformation defined above as  $T_i(f)(x, y)$  is a contraction and that the next relation is true:

$$d(f(x, y), c_i f(\tilde{T}_i^{-1}(x, y)) + o_i) < \varepsilon, \quad \forall (x, y) \in R_i \quad (11)$$

- If  $\exists i$  such that for  $R_i$  it is impossible to determine the above information, then repeat the algorithm from step 1, considering  $R_i$  as the entire image.

Due to the large number of comparisons needed to find the right parameters of compression, such algorithm has a high time cost and therefore is not very practical. That is why many optimization methods have been developed.

Three of them are reviewed below:

#### 4. Kohonen optimization

A Kohonen network is an unsupervised artificial neural network which takes as input a number of  $m$  vectors and tries to map them to a number of  $p$  output vectors. Usually,  $m$  is much larger than  $p$ ; this type of network is used to reduce dimensionality of a problem, while preserving the topological properties of the input data. In this purpose, two vectors will be used:

- $r_i \in \mathbb{R}^n$ , representing the intensity of the pixels in the range blocks  $R_i$ ;
- $d_j \in \mathbb{R}^n$ , representing the intensity of the pixels in the domain blocks  $D_j$ ;

Observation: One of the instruments used in this method is the  $\phi$  operator defined below ([1]):

Let  $C = (1/\sqrt{n}, \dots, 1/\sqrt{n}), C \in \mathbb{R}^n, \Omega$  be a linear envelope of  $C$  and

$O : \mathbb{R}^n \rightarrow \Omega^\perp$  be an operator of orthogonal projection.

Then, for  $Z = (z_1, z_2, \dots, z_n) \in \mathbb{R}^n \setminus \Omega, \phi(Z) := \frac{OZ}{\|OZ\|}$

The training of the network involves clustering the sets  $\pm \phi(d_j)$ .

The weight vectors  $w_i$  of the network will be close to the geometric center of the clusters.

The algorithm contains two main steps :

1. Training step:

- Initialize the weight vectors  $w_i$  randomly.  
Observation: for obtaining good results, one can use for initialization some subsets of  $\{\phi(d_j)\}_{j \in \{1, \dots, K\}}$ .
- Search the nearest weight vector for  $\phi(d_j)$ .  
Observation: this step involves choosing some vector  $\phi(d_j)$  and searching among the weight vectors  $w_i$ , one that minimizes the distance to the chosen domain vector.
- Improve the weight vectors, by granting different influence power to  $\phi(d_j)$ :

$$w_i = w_i + \gamma(\phi(d_j) - w_i), 0 < \gamma < 1 \quad (12)$$

In this way the results of the training algorithm are the corresponding clusters for each domain vector. The following step of the algorithm gives the cluster index for each  $\phi(d_j)$ , the index of this vector in the cluster and the distance from this vector to the range vector  $\phi(r_k)$ .

2. Choosing the nearest domain vector for each range vector:
  - for each  $k \in \{1, \dots, M\}$  do
    - compute the distance from each domain vector to the weight vector (which is the center or close to the center of the cluster);
    - find the nearest domain vector for  $\phi(r_k)$  in the cluster in which it belongs.

## 5. Hash-like fractal image compression

Feature vector methods are another approach used in fractal image compression optimization. The idea is the following: for each block there exists a vector recording its characteristics, and hence, the problem is translated from the world of finding similarities between blocks in that of finding similarities between feature vectors. One of the ideas of representing a block involves building the feature vector from a set of pixels selected from the block. The distance between the pixels should be as large as possible. Therefore, a good idea would be ([2]) to consider the four pixels from the corner of the block and, optionally, one from the middle.

The proposed algorithm in [2] searches for the nearest domain block to a range through the help of their features, which are synthesized in hash tables.

Every range has a hash key set which represents it (stores its characteristics, like a fingerprint) and among the values of it, the algorithm searches for one close to some value in the hash table of the domain block. If such value is found, then the current range block will be compressed using the corresponding domain. If not, then the algorithm uses an escape

procedure, which compresses the block through a classic algorithm (Discrete Cosine Transformation, respectively).

## 6. The yardstick travelling technique

The yardstick travelling method is proposed by Walach and Karnin and compresses every line in an image using the following idea:

Considering a scan line of pixels (which means representation of the number of the pixels depending of their intensity), one can try to approximate it with a set of straight lines called yardsticks.

The number of the points obtained in this manner is less than the number of pixels, but for a sufficiently small dimension of the yardstick, the approximation could be successfully used to compress the current line. The method's efficiency depends ([3]) on the fractal dimension of the image, as the number of steps necessary for compression is given by:

$$N = fy^{-d} \quad (13)$$

where  $f$  is a proportionality factor (usually representing the amount of data samples used in the process),  $y$  is the length of the "yardstick" and  $d$  is the fractal dimension.

## 7. Conclusion

The importance of fractal image compression as an instrument for optimizing storage and transportation methods is more and more recognized among specialists.

That is why the improvement of this type of compression is looked upon as a useful domain to study, in the present and also in the future.

Although, there is much to be done, the above mentioned methods represent an important step in the development of the image compression domain.

---

## References

- [1] R.V. Lototskiy. Images Fractal Compression Optimization by Means of Artificial Kohonen Neural Networks. *Journal of Automation and Information Sciences*. **35**(1), (2003).
- [2] K. Koroutchev, J. R. Dorronsoro. Hash-Like Fractal Image Compression with Linear Execution Time. In: *Iberian Conference on Pattern Recognition and Image Analysis*, 395-402 (2003).
- [3] E. Walach, E. Karnin. A Fractal Based Approach to Image Compression. *ICASSP*. **1**, 529-532 (1986).
- [4] V. Britanak, R. L. de Queiroz, R.D. Dony, Guojun Lu, S. Mann, T. Nguyen, G. Schuller, I. Selesnick, T. Tran, J. Walker, X. Wu. *The Transform and Data Compression Handbook*. (CRC Press LLC, Boca Raton, Florida, 2001).