GEOMETRICAL INTERPRETATION OF IFS BASED IMAGE CODING

JULIEN SIGNES
CCETT, 4, rue du Clos Courtel, 35 510 Cesson-Sévigné - FRANCE

Abstract

In most classical coding methods, a global optimization is carried out to ensure a given bit-rate or a given quality. In IFS based coding, some heuristic choices are made at each step of the algorithm: No method has, for instance, been proposed to optimize the domain blocks position, or the quantization of the transformations parameters. In this article we first analyze the limits of the existing algorithms, then define a new geometrical interpretation of the classical Jacquin-like method. Through the above analysis, we will answer some of these important questions.

1. INTRODUCTION

Michael Barnsley [1] was the first to propose a fractal based coding scheme for images. In most classical coding schemes, we consider the local redundancy in the image in order to reduce the volume of data. The basic idea of iterated function system (IFS) based coding is to take advantage of the similarities between different parts of the image at different scales. Jacquin was the first in 1989 [2] to propose a fully automatic algorithm for achieving this goal.

The method consists in partitioning the image support, $\mathcal{I}$, in range regions [*], $\mathcal{R}_k$. For each region $\mathcal{R}_k$, a "corresponding" domain region $\mathcal{D}_k$ is searched in the image. "Corresponding" means that under a contractive transform W_k, the luminance $\mathbf{D}_k$ of the region $\mathcal{D}_k$ is similar to the luminance $\mathbf{R}_k$ of the region $\mathcal{R}_k$ according to a certain distance $\mathbf{d}$. This transformation

[*]Initially, these regions were fixed size blocks, but we will consider any kind of regions.

1. **Spatial sub-sampling:** The domain region is usually bigger in surface than its corresponding range region, so it has to be sub-sampled for matching $\mathcal{R}_k$. We will denote this sub-sampled version of the luminance of $\mathcal{D}_k$ by $\mathbf{D}_k^{'}$, and the corresponding transform S_k.

2. **Spatial Isometry:** The domain region is then transformed by one of the eight 2D isometries that leave the square unchanged. We denote the new luminance vector obtained after applying the isometry O_k by $\mathbf{D}_k^{''}$.

3. **Luminance transform:** We then apply a contractive affine transform L_k on the luminance vector $\mathbf{D}_k^{''}$ to obtain the final luminance vector $\mathbf{R}_k^{'} = \alpha_k \mathbf{D}_k^{''} + \beta_k \mathbf{1}$. $\mathbf{1}$ denotes the constant vector containing only values 1.

The algorithm searches a pool of domain regions for the best fitting $\mathcal{D}_k$ minimizing the distance $\mathbf{d}(\mathbf{R}_k, \mathbf{R}_k^{'})$. The Banach fixed point theorem shows that an image $\mathbf{I}$ can then be coded by the transform $T = \bigcup_k W_k$.

2. LIMITS OF THE EXISTING ALGORITHMS

At this stage several unsolved problems arise:

- How should we choose the geometrical range partition for an optimal coding? Many attempts have been made to address the above problem, including fixed block partitions, quad-tree partitions [3], triangle partitions [4], polygonal partitions [5], and H-V partitions [3]. All the above methods except H-V are not theoretically justified. Hence the less expensive (in terms of complexity and bit rate) one, the fixed size block or the quad tree for adaptivity, seems to be the most reasonable choice. The H-V partition consists in choosing range and domain rectangles that are either "smooth" or contain an edge in their diagonal. In this way the content of a range region is explicitly chosen so that it can be easily matched to a domain region. However, none of the above methods ensures that for a highly textured region of reasonable size (for data compression), a correct matching domain region can be found. On the contrary, the numerical experiments we carried out indicate that most of the above variations of the Jacquin algorithm behave poorly on high frequencies areas (see Fig. 1)

- How should we quantize the parameters α_k and β_k of the image? Y. Fisher carried out a complete numerical experiment of the Jacquin algorithm in [6] to solve this problem. However only uniform scalar quantizations have been tried and no theoretical justification of the results have been stated.

Many other questions remain to be answered, but we will restrict ourselves to these two and try to analyze the algorithm in such a way that theoretical answers can be given.

a

b

Fig. 1 Lenna 512×512 coded with a classical quadtree fractal algorithm at 0.21 bpp (a),and with a JPEG algorithm at the same rate (b): On (a), one can notice the poor result on the feathers. On (b), block effects are more visible but high frequencies are better preserved

3. A GEOMETRICAL ANALYSIS

In this section we will assume that a pool $\left\{\mathcal{D}''_{k,i}\right\}_{i\in\{0\cdots Q_k\}}$ of Q_k already spatially transformed acceptable domain regions is available. The considered range region $\mathcal{R}_k$ contains N_R pixels. The algorithm is equivalent to the following minimization problem:

$$\min_{i\in\{0\cdots Q_k\},|\alpha_k|<1,\beta_k\in[0,L]} \mathbf{d}(\mathbf{R}_k, \alpha_k \mathbf{D}''_{k,i} + \beta_k \mathbf{1})$$

L denotes the maximum value of the luminance. This minimization is equivalent to the geometrical projection of the points $\mathbf{R}_{k,i}$ of $\Re^{N_R}$ on the two dimensional subspaces generated by the vectors $\mathbf{1}$ and $\mathbf{D}''_{k,i}$. In fact, the points belong to the restriction of $\Re^{N_R}$ to $[0,L]^{N_R}$. Since $|\alpha_k| < 1$ and $\beta_k \in [0,L]$, the projection space is a two dimensional "band" $B_{k,i}$. These bands are all "turning" around the vector $\mathbf{1}$ and intersect the origin O. The Fig. 2 illustrates the model in the very simple case where the range region is a three-pixels triangle.

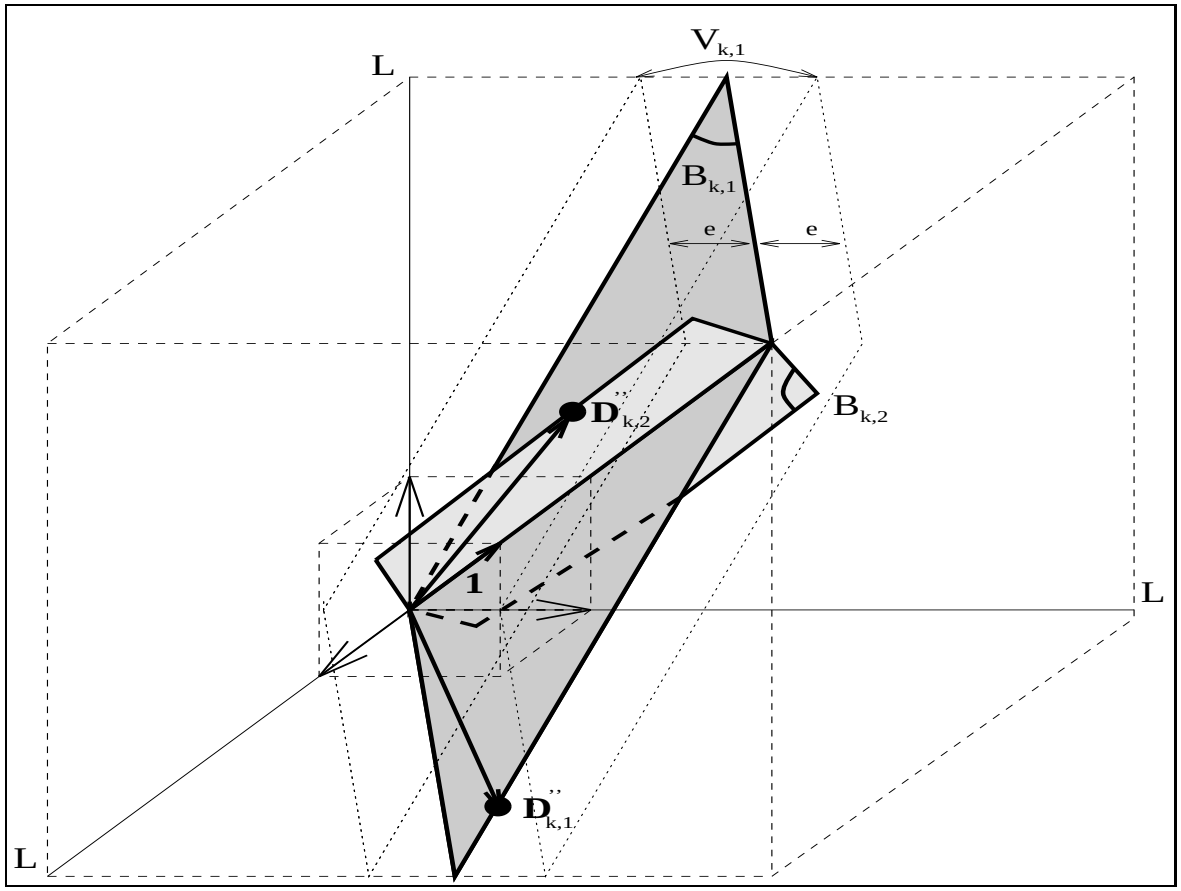

Fig. 2 The geometrical model for $N_R = 3$ and for a pool of two domain regions. $\mathbf{D}''_{k,1}$ generates the largest occupation volume $V_{k,1}$. $\mathbf{D}''_{k,1}$ is "near" the segment generated by $\mathbf{1}$ and has a smaller occupation volume $V_{k,2}$

4. SPACE FILLING

When solving the above problem for a highly textured range region, no assumption can be made on its content, therefore the vector $\mathbf{R}_k$ should be considered as a random point in $K = [0, L]^{N_R}$. A "good" pool of domain regions should hence generate bands $B_{k,i}$ that "occupy" as much space as possible: Given a tolerance error e, the volume V_k generated by the balls of radius e centered on the points of the bands $B_{k,i}$ should occupy the whole set K. We can estimate an upper bound of this volume V_k by assuming that all individual occupation volumes $V_{k,i}$ do not intersect, and that $\mathbf{d} = L_\infty$. Given that the bands $B_{k,i}$ have a maximum width of $\sqrt{N_R}L$ (to fit inside the set K), the volume to estimate is enclosed in a N_R dimensional rectangle parallepiped with two "sides" of size $\sqrt{N_R}L$ and $(N_R - 2)$ of size e. Hence,

$$V_{k,i} < (2e)^{N_R-2} N_R L^2 \quad and,$$

for the global occupation volume V_k:

$$V_k < Q_k (2e)^{N_R-2} N_R L^2$$

An upper bound of the probability P_k of "acceptance" of the region $\mathcal{R}_k$ is hence given by:

$$P_k = \frac{V_k}{V_t} < Q_k N_R \left(\frac{2e}{L}\right)^{N_R-2}$$

In a usual case where $N_R = 8 \times 8 = 64$, $L = 256$, $Q_k = 256$, and with an acceptance error of $e = 8$, we have $P_k < \frac{1}{2^{234}}$!!!. If we wanted this probability to become more "reasonable", we would like to find $P_k < 1$ (which is always true, of course, but intuitively we understand that in this case, P_k is closer to one), then in the above usual case $Q_k = 2^{242}$ domain blocks would be necessary (remember that Q_k is here the total number of domain blocks after the subsampling and the geometrical transformations).

This shows that it is "statistically" impossible to find a matching domain block for a given highly textured range block of size 8×8. If we want to ensure a given reasonable error e, we must reduce the size of the blocks. We also understand why fractal schemes behave so poorly in high frequency areas, as we noticed in our numerical experiments.

In the next sections, we will analyze how we can try to better this acceptance probability.

5. OPTIMAL PARAMETERS QUANTIZATION

In the previous section, we have carried out a volume estimation assuming that all bands $B_{k,i}$ had an empty intersection, and that all points of each band were represented. These two hypotheses can lead to an upper bound of the volume V_k but don't stand if we want to find an optimal quantization for α_k and β_k. We have to take into account the fact that all bands intersect at the origin O and include the segment generated by vector $\mathbf{1}$. This means that "flat" blocks are over represented, whereas some areas of textured blocks in our space K are not in the occupation volume. In order to occupy as much volume as possible, the geometrical model then leads to the following conclusions:

1. For the higher values of the parameters α_k and β_k a finer quantization should be used. Hence a uniform quantization is not optimal.

2. The parameters should never take a zero value. The initial value should be at least e.

3. To ensure an acceptance error e, the distance between two quantized points on a given projection band $B_{k,i}$ should lie between e and $2e$ (depending on which distance is used).

The Fig. 3.a illustrates the case where a uniform quantization has been applied. ith shows the over representation of constant blocks, and the weakness on highly textured blocks (near the boundaries of the space K). In Fig. 3.b, non uniform quantization has been applied, and better domain blocks have been chosen (see next section). These ideas have not yet been implemented.

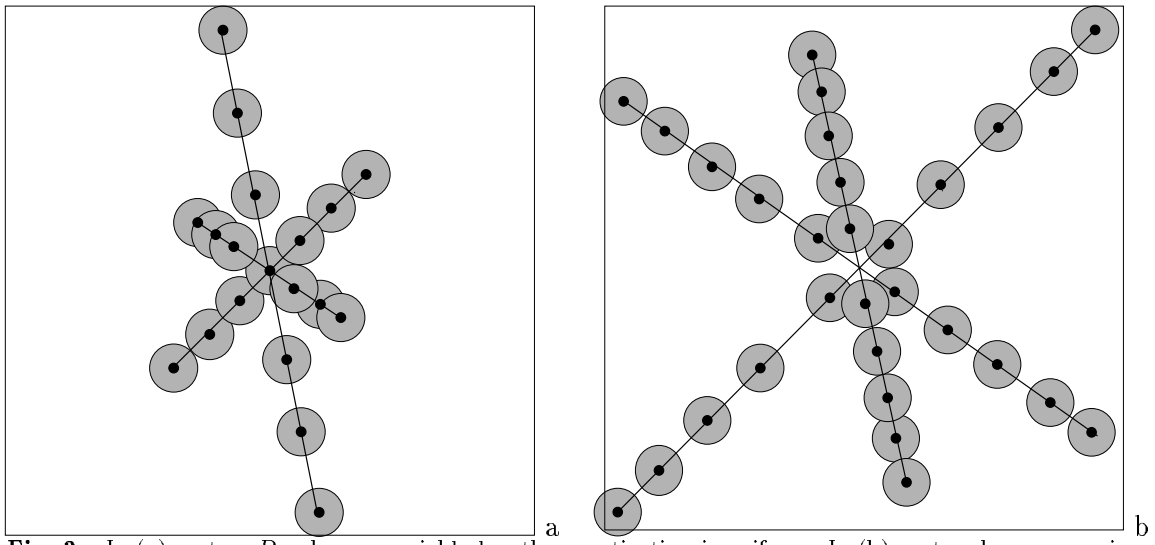

Fig. 3 In (a) vectors $D_{k,i}$ have a variable length, quantization is uniform. In (b) vectors have approximatively the same length, and quantization is not uniform

The Fig. 4 shows the distribution of the α parameters obtained with the image *Lenna* 512×512. As expected the most probable values are ±1, which indicates that the boundaries of the space K are not accurately represented. We are now able to estimate the number of necessary bits to encode the transformations parameters: if we want to ensure an error of $e = 8$, with a segment of length $L = \sqrt{N_R}L = 255 * 8$ to be covered, we find that at least 128 points are necessary, which means we need 7 bits to represent each parameter β_k. This is exactly the result found numerically in [6].

6. OPTIMAL DOMAIN REGIONS

6.1 Geometrical Criteria For An Optimal Domain Pool

Choosing an optimal domain pool is essential for minimizing the global collage error. Since full search complexity is too high, and no assumption can be made on the content of a textured range region, the only reasonable method seems to be a fixed path search, as it is done in most publications. However, the volume criterion suggests the following:

1. The luminance vectors $\mathbf{D}''_{k,i}$ should not be near the segment generated by the vector $\mathbf{1}$. This means the distance between the domain vector and this segment should be as large as possible.

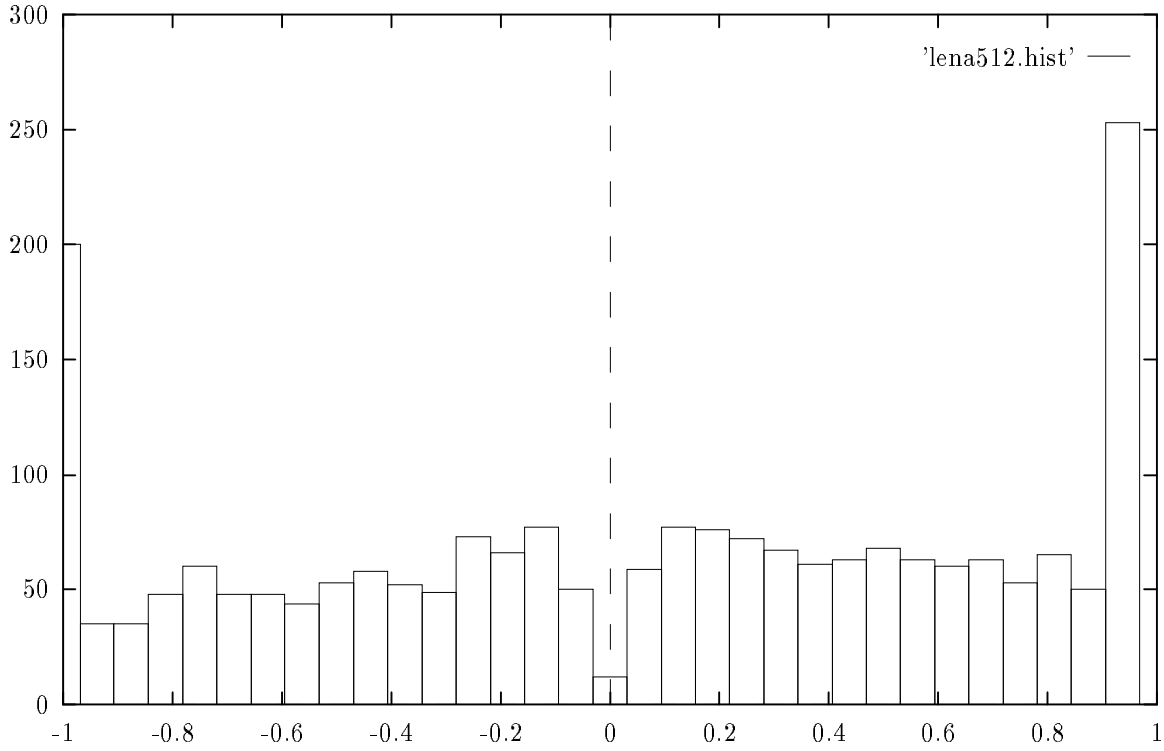

Fig. 4 Histogram of α values for Lenna 512×512

2. The vectors $D''_{k,i}$ should have at least one high value (but not all, to respect the above condition).

3. The vectors $D''_{k,i}$ should not be "too" correlated. We cannot impose an orthogonality condition since we usually take a larger pool than the dimension of the space (N_R). A way to ensure this would consist in using a generalized notion of angle (see next section) between different domain vectors $\mathbf{D}''_{k,i}$

6.2 Comparing Vectors Of The Space $\Re^{\mathrm{NR}}$

Let us consider a domain vector $\mathbf{D} = \left(\begin{array}{c} d_1 \cdots d_N \end{array} \right)$ in K. We would like to choose $\mathbf{D}$ such as the distance $d(\mathbf{1}, \mathbf{D})$ between $\mathbf{D}$ and the segment generated by vector $\mathbf{1}$ is maximal within our space K. We can calculate easily the distance in the N_R dimensional space between a point M and a straight line generated by a vector $\vec{x}$ and a point O:

$$d((O, \vec{x}), M) = \| \, O\vec{M} \, \|^2 - \left(\frac{O\vec{M}.\vec{x}}{\| \, \vec{x} \, \|} \right)^2$$

We can then estimate this distance in our case where $\vec{x} = \mathbf{1} = (1 \cdots 1)$, and $M = \mathbf{D}$. We find after some calculations:

$$d(\mathbf{1}, \mathbf{D}) = N_R \sigma(\mathbf{D})^2,$$

$\sigma(\mathbf{D})^2$ being the variance of the vector $\mathbf{D}$.

For two given domain vectors $\mathbf{D_1}$ and $\mathbf{D_2}$ a generalized cosine in the space $\Re^{N_R}$ between two vectors can be defined by:

$$\mathcal{COS}(\mathbf{D_1}, \mathbf{D_2}) = \frac{\mathbf{D_1}.\mathbf{D_2}}{\| \mathbf{D_1} \| \| \mathbf{D_2} \|}$$

The Schwarz inequality combined with the fact that all components of the domain vectors are positive shows that:

$$0 < \mathcal{COS}(\mathbf{D_1}, \mathbf{D_2}) < 1$$

When the two vectors are orthogonal, $\mathcal{COS}(\mathbf{D_1}, \mathbf{D_2}) = 0$, and $\mathcal{COS}(\mathbf{D_1}, \mathbf{D_2}) = 1$ in the case where the vectors are collinear.

6.3 Designing An Optimal Pool

According to the above observations, we would like to design a pool $\mathcal{P}$ of domain blocks (already down-sampled and spatially transformed) that would be used for any match between range and domain blocks. Of course if we want to keep an adaptive scheme (such as a quad tree algorithm for instance), with different sizes of range blocks, we would then have to design a pool $\mathcal{P}_d$ for each size d of the range block we consider.

We are able to build an optimal domain pool by applying the following method:

1. For a given size, select all possible blocks in the image, and keep those with a high variance (according to a certain threshold v_0). $\mathcal{P}_0$ will denote the first pool obtained in this way.
2. Take a starting domain block $\mathbf{D_0}$ from $\mathcal{P}_0$ and put it in $\mathcal{P}$
3. Find a block $\mathbf{D_1}$ in $\mathcal{P}_0$ such as $\mathcal{COS}(\mathbf{D_0}, \mathbf{D_1})$ is below a threshold c_0 for all $\mathbf{D_1}$ already in $\mathcal{P}$.
4. Add the block $\mathbf{D_1}$ in $\mathcal{P}$.
5. Go back to step 3 until the pool $\mathcal{P}$ has the desired size

Nothing guarantees in step 3 that such a block may be found. However, due to the enormous number of blocks in $\mathcal{P}_0$, this never happens in practice. If this was the case, for a given block $\mathbf{D_1}$ in $\mathcal{P}_0$, we would eliminate the blocks (usually one) that give a generalized cosine function above the threshold and reintroduce $\mathbf{D_1}$ in $\mathcal{P}$.

6.4 Other Conclusions

These results also explain why, as quoted in [7], IFS algorithms behave poorly in the case of compensation errors frames: In this case, most domain blocks will lie near to the origin 0, so our space K will be very poorly represented. Moreover, no specific scheme such as zero-length coding is used in the case of IFS based coding.

The geometrical model also shows that it is inefficient to add fixed vectors to the base (as suggested in [8], for instance), since all projection spaces would include the space generated by these fixed vectors. The use of a fixed vector, $\mathbf{1}$, is also questionable, for the same reason. A different strategy should be studied, where projection spaces have the "smallest" intersection.

7. NUMERICAL RESULTS

In this section, we would like to compare a classical quad-tree algorithm, in which domain blocks are chosen among the blocks of the partition, and the same algorithm but using our domain pool, designed with geometrical criteria.

Our experiments were carried out on 512×512 images, with range sizes of 16×16, 8×8 or 4×4, corresponding to domain blocks of 32×32, 16×16 or 8×8. In the following table the size of the pool for each size is given (starting with 8×8 blocks). For each experiment, the result is compared with te quad-tree method: As we can observe in the table 1, the

Table 1 The results of the "optimal pool" for various sizes of pools, compared with the result of the quad-tree method for the same compression ratio

Size	PSNR(db)	Compression	Quad-tree
(16,16,16)	28.44	54.0	27.6
(32,32,32)	28.76	50.9	27.9
(256,256,256)	29.80	39.9	28.6

optimal pool gives numerical results that are between .8 and 1.2 db above the classical quad-tre algorithm. Of course, the search of the optimal pool takes some time, so the encoding is slower. This is mostly due to the fact that the algorithm to design an optimal domain pool is not yet optimized.

Fig. 5 compares the classical quad-tree algorithm with the new one. We can notice that the coding results are also visually better in the case of the new algorithm. This is due to the use of domain blocks with a high variance, whereas a lot of flat blocks are used in the case of the quad-tree algorithm. Another main difference appears when comparing the bit streams of the two algorithms: In the new one, the positions of the domain blocks in the global pool are stored, and then only an index is necessary to adress them in each transform. In the case of the quad-tree algorithm, the position of the domain block in the quad-tree partition has to be given for each transformation, which is expensive in terms of bit rates.

Some efforts still remain to be done in order to find a faster algorithm, but these results look very promising for this preliminary stage. We can also think that a mixed appoach using global domain pools and local ones could be fruitful. Professor D. Saupe used a similar analysis and succeeded in reducing the complexity by projecting the problem in a one dimensional space [9]. It should also be possible to improve the occupation volume by using non affine luminance transforms. Further work remains to be done on this subject.

8. CONCLUSIONS

We have developed a geometrical analysis for Jacquin-like coding algorithms. The new model explained some numerical results of previous publications, and improved the coding results. A lot of work remains to be done in this relatively new field of image coding, including better analysis of contractivity factor and coding of high frequency regions.

9. ACKNOWLEDGMENT

The author thanks Henri Sanson (CCETT), Professor Henri Maître (ENST, Paris) and Professor Jacques Levy Vehel (INRIA, Rocquencourt) for useful discussions.

a

b

Fig. 5 Lena 512 × 512 coded with a compression factor of 50.9 and PSNR = 28.76dB with the new algorithm (a), and at PSNR = 27.9 dB with the classical quad-tree algorithm (b). The image (a) was generated with the use of only 32+32+32 = 96 domain vectors

10. REFERENCES

1. Michael F. Barnsley and Alan D. Sloan. A better way to compress images. *Byte*, pages 215–223, January 1988.
2. A. E. Jacquin. *A fractal theory of iterated Markov operators with applications to digital image coding*. PhD thesis, Georgia Institut of Technologie, 1989.
3. Y. Fisher, E.W. Jacobs, and R.D. Boss. Fractal image compression using iterated transforms. In James A. Storer, editor, *Image and Text Compression*, chapter 2, pages 35–61. Kluwer Academic Publishers, Norwell, Massachusetts, 1992.
4. F. Davoine, E. Bertin, and J-M. Chassery. From rigidity to adaptive tesselations for fractal image compression: Comparative studies. In *IEEE 8th Workshop on Image and Multidimensional Signal Processing*, September 1993.
5. Emmanuel Reussens. Partitioning complexity issue for iterated functions systems based image coding. *SIGNAL PROCESSING VII: Theories and Applications*, pages 171–174, 1994.
6. Y. Fisher, E.W. Jacobs, and R.D. Boss. Image compression: A study of the iterated transform method. *Signal Processing*, 29:251–263, 1992.
7. B. Hurtgen and P. Buttgen. fractal approach to low rate video coding. In *Proceedings of SPIE: Visual Communications and Image Processing*, volume SPIE Vol. 2094, pages 120–131, 1993.
8. Geir E. Øien, , Skjalg Lepsøy, and Tor A. Ramstad. Attractor image compression with a fast non iterative decoding algorithm. In *International Conference on Accoustics, Speech and Signal Processing*, pages 337–340, 1993.
9. Dietmar Saupe. Accelerating fractal image compression by multi dimensional nearest neighbor search. In *ASI NATO 95 on Fractal image encoding and analysis, Trondheim, to be published in a Springer-Verlag book*, 1995.