??, ??, 1–15 (??)

Fast Iterative Methods for Fractal Image Compression

RAOUF HAMZAOUI

hamzaoui@informatik.uni-leipzig.de

Institut für Informatik, Universität Leipzig, Augustusplatz 10–11, 04109 Leipzig, Germany

Received ??; Revised ??

Editors: ??

Abstract. In fractal image compression, a digital image is approximated by the unique fixed point of a contractive affine mapping. The decoding consists of iterating the affine mapping starting from an arbitrary image until convergence to the fixed point. We show that the decoding can be accelerated by using the new pixel intensities of an image iterate as soon as they become available. We provide a mathematical formulation of the proposed decoding and prove its convergence in a general setting. We show that under some mild restrictions the *asymptotic rate of convergence* of the proposed method is greater than or equal to that of the conventional method. We also discuss the use of standard iterative methods for the decoding. Finally, we show that the convergence of the proposed method can be enhanced by an ordering technique.

Keywords: Image coding, linear iterative methods, regular splittings, ordering

1. Introduction

The basic principle of fractal image compression [1, 2] is to store an image f, seen as an element of a complete metric space (X, d), as a contractive mapping $T : X \to X$ whose unique fixed point $f_T = T(f_T)$ approximates f. The code for the image is an efficient binary representation of the mapping T. The decoding is based on the contraction mapping principle, which states that f_T can be obtained as a limit of the sequence of iterates $\{f^{(k)}\}_k$, where $f^{(k+1)} = T(f^{(k)})$ and the starting point $f^{(0)}$ is arbitrary. The mapping T is determined by making the original image f and the *collage* $T(f)$ close enough. This approach is motivated by the Collage Theorem [3], which states that

$$d(f_T, f) \leq \frac{1}{1 - s(T)} d(f, T(f))$$

where $s(T)$ is the contraction factor of T.

Fractal image compression suffers from long encoding times [4]. However, the decoding is simple and relatively fast. This makes fractal image compression suitable for storage and retrieval applications in which the encoding is done once using special hardware, whereas the decoding is to be repeated several times in software by the user. One such application is CD-ROM encyclopedia for personal computers. Thus, it is worth asking if one can further accelerate the decoding for those practical environments.

The main purpose of this work is to analyze theoretically and numerically a decoding algorithm for fractal image compression first proposed in [5], used in [6], and rediscovered independently in [7, 8, 9]. Compared to the conventional algorithm,

the proposed algorithm allows faster convergence and relieves memory requirements.

The paper is organized as follows. In Section 1.1, we provide a mathematical formulation for the encoding and decoding problems in fractal image compression, which encompasses most of the schemes proposed in the literature. In Section 1.2, we describe Fisher's quadtree coder [10], which we used for most of our numerical results. In Section 2.1, we explain the proposed decoding procedure and prove its convergence. We also prove that under certain mild conditions the *asymptotic rate of convergence* of the proposed method is greater than or equal to that of the conventional method. Experimental results confirming the superiority of the method conclude the section. In Section 2.2, we discuss other fast iterative methods for the decoding. In Section 2.3, we use an ordering technique to further enhance the speed of convergence of the proposed method.

1.1. A linear algebraic framework

As suggested by Lundheim [11], we use a linear algebraic framework for fractal image compression. This will allow us to rigorously treat the convergence properties of the decoding.

We consider a monochrome digital image of size $N_h \times N_v$ as a real valued function f defined on $\mathcal{X} = \{0,\ldots,N_h-1\}\times\{0,\ldots,N_v-1\}$, $N_h, N_v \geq 1$. Thus, the intensity or gray value of pixel $(i,j) \in \mathcal{X}$ is $f(i,j)$. An *image piece* is the restriction $f_{/\mathcal{B}}$ of f to a nonempty subset $\mathcal{B} \subset \mathcal{X}$ called *support*.

Let $N = N_h N_v$ and let ψ be a one-to-one mapping from $\mathcal{X}$ to $\{1,\ldots,N\}$ called *ordering mapping*. For a given ψ, we associate to the image piece $f_{/\mathcal{B}}$ an $n-$dimensional vector

$$\mathbf{x}_{f_{/\mathcal{B}}} = (f\psi^{-1}(b_1),\ldots,f\psi^{-1}(b_n))^T.$$

Here, T is the transpose symbol and $\psi(\mathcal{B}) = \{b_1,\ldots,b_n\} \subset \{1,\ldots,N\}$, where $b_1 < b_2 < \ldots < b_n$. Conversely, any $N-$dimensional vector $\mathbf{x}$ can be seen as the unique image f such that $\mathbf{x}_f = \mathbf{x}$.

Let f be an image defined on $\mathcal{X}$. Let τ be a one-to-one transformation (permutation) on $\mathcal{B} \subset \mathcal{X}$. The image of $f_{/\mathcal{B}}$ under τ, $\tau f_{/\mathcal{B}}$ is the function defined on $\mathcal{B}$ by

$$(\tau f_{/\mathcal{B}})(i,j) = f\tau^{-1}(i,j).$$

Throughout the paper O denotes the zero matrix and $\mathbf{1}$ is a vector whose components are all equal to one. In both cases, the dimension will be clear from the context.

For a given original image $f : \mathcal{X} \to \mathbb{R}$ of size $N_h \times N_v = N$, the encoder finds a partition of the image support $\mathcal{X}$ into n_R disjoint image supports $\mathcal{R}_i$, $i = 1,\ldots,n_R$ called *ranges* such that each range $\mathcal{R}_i$ of size n_i (the number of pixels in $\mathcal{R}_i$) is *encoded* by :

- an image support $\mathcal{D}_i \subset \mathcal{X}$ of size $m_i n_i$ called *domain*,
- a permutation matrix P_i of order $m_i n_i$,
- a real number s_i called *scaling factor*,
- and a real number o_i called *offset*,

which are determined such that the range vector $\mathbf{x}_{f_{/\mathcal{R}_i}}$ is well approximated by the vector

$$s_i E_i P_i \mathbf{x}_{f_{/\mathcal{D}_i}} + o_i \mathbf{1}. \tag{1}$$

Here, E_i is an $n_i \times m_i n_i$ *downsampling matrix* defined by

$$E_i = \frac{1}{m_i}\begin{pmatrix} \mathbf{1} & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{1} \end{pmatrix},$$

where $\mathbf{1} = (1\ldots1)$ and $\mathbf{0} = (0\ldots0)$ are $1 \times m_i$ submatrices.

Now let $\psi(\mathcal{R}_i) = \{(r_i)_1,\ldots,(r_i)_{n_i}\}$ and $\psi(\mathcal{D}_i) = \{(d_i)_1,\ldots,(d_i)_{m_i n_i}\}$. Consider the $m_i n_i \times N$ matrix $C_i = (c^i_{u,v})$

$$c^i_{u,v} = \begin{cases} 1, & \text{if } (u,v) = (k,(d_i)_k), \quad k = 1,\ldots,m_i n_i; \\ 0, & \text{otherwise,} \end{cases}$$

and the $N \times n_i$ matrix $G_i = (g^i_{u,v})$

$$g^i_{u,v} = \begin{cases} 1, & \text{if } (u,v) = ((r_i)_k,k), \quad k = 1,\ldots,n_i; \\ 0, & \text{otherwise.} \end{cases}$$

Then each encoding of a range $\mathcal{R}_i$ yields a mapping $T_i : \mathbb{R}^N \to \mathbb{R}^N$ such that

$$T_i(\mathbf{x}) = s_i G_i E_i P_i C_i \mathbf{x} + o_i G_i \mathbf{1}.$$

The code for the original image f (or equivalently for the vector $\mathbf{x}_f$) is given by the affine mapping $T : \mathbb{R}^N \to \mathbb{R}^N$ defined by

$$T(\mathbf{x}) = \sum_{i=1}^{n_R} T_i(\mathbf{x})$$

$$= A\mathbf{x} + \mathbf{b},$$

where $A = \sum_{i=1}^{n_R} s_i G_i E_i P_i C_i$ is a real $N \times N$ matrix and $\mathbf{b} = \sum_{i=1}^{n_R} o_i G_i \mathbf{1}$ is a real column vector of dimension N. Note that each row of the matrix A has m_i components (for an $i \in \{1, \ldots, n_R\}$) equal to s_i/m_i and $N - m_i$ components equal to zero.

When T is a contraction, the contraction mapping principle ensures that the fixed point $\mathbf{x}_T = T(\mathbf{x}_T)$ can be obtained as limit of the sequence of iterates $\{\mathbf{x}^{(k)}\}_k$, where $\mathbf{x}^{(0)}$ is *any* initial vector and $\mathbf{x}^{(k)}$ is defined by the linear iterative method

$$\mathbf{x}^{(k+1)} = T(\mathbf{x}^{(k)}) = A\mathbf{x}^{(k)} + \mathbf{b}. \qquad (2)$$

The computation of $\mathbf{x}_T$ by the iterative method (2) is called conventional decoding. Making the vector in (1) close to $\mathbf{x}_{f_{/\mathcal{R}_i}}$ for all $i = 1, \ldots, n_R$ guarantees that $T(\mathbf{x}_f)$ is close to $\mathbf{x}_f$ as required by the Collage Theorem.

An essential property of the decoding algorithm is the convergence of the sequence of images to the fixed point for any starting image. More precisely

Definition 1. Let $A = (a_{u,v})$ be an $N \times N$ matrix and $\mathbf{b}$ be a vector of dimension N. The linear iterative method $\mathbf{x}^{(k+1)} = A\mathbf{x}^{(k)} + \mathbf{b}$ is convergent if for all $\mathbf{x}^{(0)}$ the sequence $\{\mathbf{x}^{(k)}\}_k$ converges to a limit that is independent of $\mathbf{x}^{(0)}$.

A necessary and sufficient condition for the convergence of the iterative method (2) is $\rho(A) < 1$, where $\rho(A)$ is the spectral radius of A [12]. The following proposition gives a practical condition to ensure the convergence of the decoding.

Proposition 1. *Let T be defined as before. If $|s_i| < 1$ for all $i = 1, \ldots, n_R$, then the iterative method (2) is convergent.*

Proof: We have

$$\begin{aligned} \|A\|_\infty &= \max_{1 \le u \le N} \sum_{v=1}^{N} |a_{u,v}| \\ &= \max_{1 \le i \le n_R} m_i |s_i/m_i| \\ &= \max_{1 \le i \le n_R} |s_i| \\ &< 1. \end{aligned}$$

Hence T is a contraction for the L_∞ norm, which ensures the convergence of the iterative method. □

1.2. Fisher's quadtree scheme

An efficient fractal image compression scheme should be able to find the mapping T in reasonable time. Moreover, the binary representation of T should be compact. This is satisfied by Fisher's quadtree scheme [10], which hierarchically partitions the support of the image into square blocks (ranges) and approximates the range vectors by domain vectors in a least squares sense. In this section, we briefly describe Fisher's encoding and decoding algorithms. More details can be found in [10].

The encoding: Fix a maximum block size and a minimum block size. Let t be a root mean square tolerance value. Let s_{max} be a positive number. Start from an initial partition of the image support into square blocks of maximum size. Then, until all blocks in the partition are encoded do the following:

1. Let $\mathcal{B}$ be a nonencoded $2^n \times 2^n$ square block in the partition. For each $2^{n+1} \times 2^{n+1}$ square block $\mathcal{D}_p$ (domain) of the image support whose upper-left pixel is situated on a grid with a fixed horizontal and vertical spacing of l pixels, and for each permutation τ_k, $k = 1, \ldots, 8$ find the real numbers $s_{p,k}$ and $o_{p,k}$ solutions of the least squares problem

$$\min_{s,o \in \mathbb{R}} \|R_n(f_{/\mathcal{B}}) - (sE_{n+1}R_{n+1}(\tau_k f_{/\mathcal{D}_p}) + o\mathbf{1})\|_2.$$

Here, the permutations τ_k correspond geometrically to the eight isometries of the square (the identity, three rotations, and four reflections), the operator R_n maps an image piece $f_{/\mathcal{B}}$ defined on a $2^n \times 2^n$ square block $\mathcal{B} = \{i, i+1, \ldots, i+2^n -$

Fig. 1. The 512 × 512 Boat image and a three-level quadtree partitioning of its support.

$1\} \times \{j, j+1, \ldots, j+2^n-1\}$ to the column vector

$$R_n(f_{/\mathcal{B}}) = \begin{pmatrix} f(i,j) \\ f(i+1,j) \\ \vdots \\ f(i+2^n-1,j) \\ f(i,j+1) \\ f(i+1,j+1) \\ \vdots \\ f(i+2^n-2,j+2^n-1) \\ f(i+2^n-1,j+2^n-1) \end{pmatrix},$$

and the matrix E_{n+1} reduces the size of vector $R_{n+1}(\tau_k f_{/\mathcal{D}_p})$ to $2^n \times 2^n$ by averaging components corresponding to groups of four neighboring pixels.

2. Uniformly quantize $s_{p,k}$ in $[-s_{max}, s_{max}]$ and $o_{p,k}$, yielding a scaling factor $\hat{s}_{p,k}$ and an offset $\hat{o}_{p,k}$.

3. Find a domain $\mathcal{D}^*$ and a permutation τ^* that minimize the root mean square error

$$\frac{1}{2^n}\|R_n(f_{/\mathcal{B}}) - (\hat{s}_{p,k}E_{n+1}R_{n+1}(\tau_k f_{/\mathcal{D}_p}) + \hat{o}_{p,k}\mathbf{1})\|_2.$$

4. If the minimum root mean square error is less than t or if the size of $\mathcal{B}$ is equal to the minimum block size, then consider $\mathcal{B}$ as encoded and store the position of $\mathcal{D}^*$, the index of τ^*, and the indices of the corresponding scaling factor and offset. Otherwise, add the four quadrants of $\mathcal{B}$ to the partition and remove $\mathcal{B}$ from the partition.

The ranges are the blocks in the final partition. **The decoding**: The data structure used by the decoder is a linked list of *range transformations*. Each transformation specifies the position of a range, the location of the domain, the permutation, the scaling factor, and the offset (see [2] p. 279 for more details). Let k_{max} be a maximum iteration step. Let $f^{(0)}$ be an arbitrary initial image. Denote the ranges by $\mathcal{R}_i$, $i = 1, \ldots, n_R$. Denote the domain, the permutation, the scaling factor, and the offset used for the encoding of $\mathcal{R}_i$ by $\mathcal{D}_{(i)}, \tau_{(i)}, s_{(i)}$, and $o_{(i)}$, respectively. Then compute the sequence of image iterates $f^{(k)}$ as follows:
For $k = 0, \ldots, k_{max} - 1$
For $i = 1, \ldots, n_R$

$$R_n(f^{(k+1)}_{/\mathcal{R}_i}) = s_{(i)}E_{n+1}R_{n+1}(\tau_{(i)}f^{(k)}_{/\mathcal{D}_{(i)}}) + o_{(i)}\mathbf{1}.$$

Figure 1 shows for the 512 × 512 Boat image a quadtree partition generated by Fisher's quadtree scheme. The minimum block size is 8 × 8, the maximum block size is 32 × 32, the tolerance value t is equal to 8, $s_{max} = 0.99$, and the grid spacing

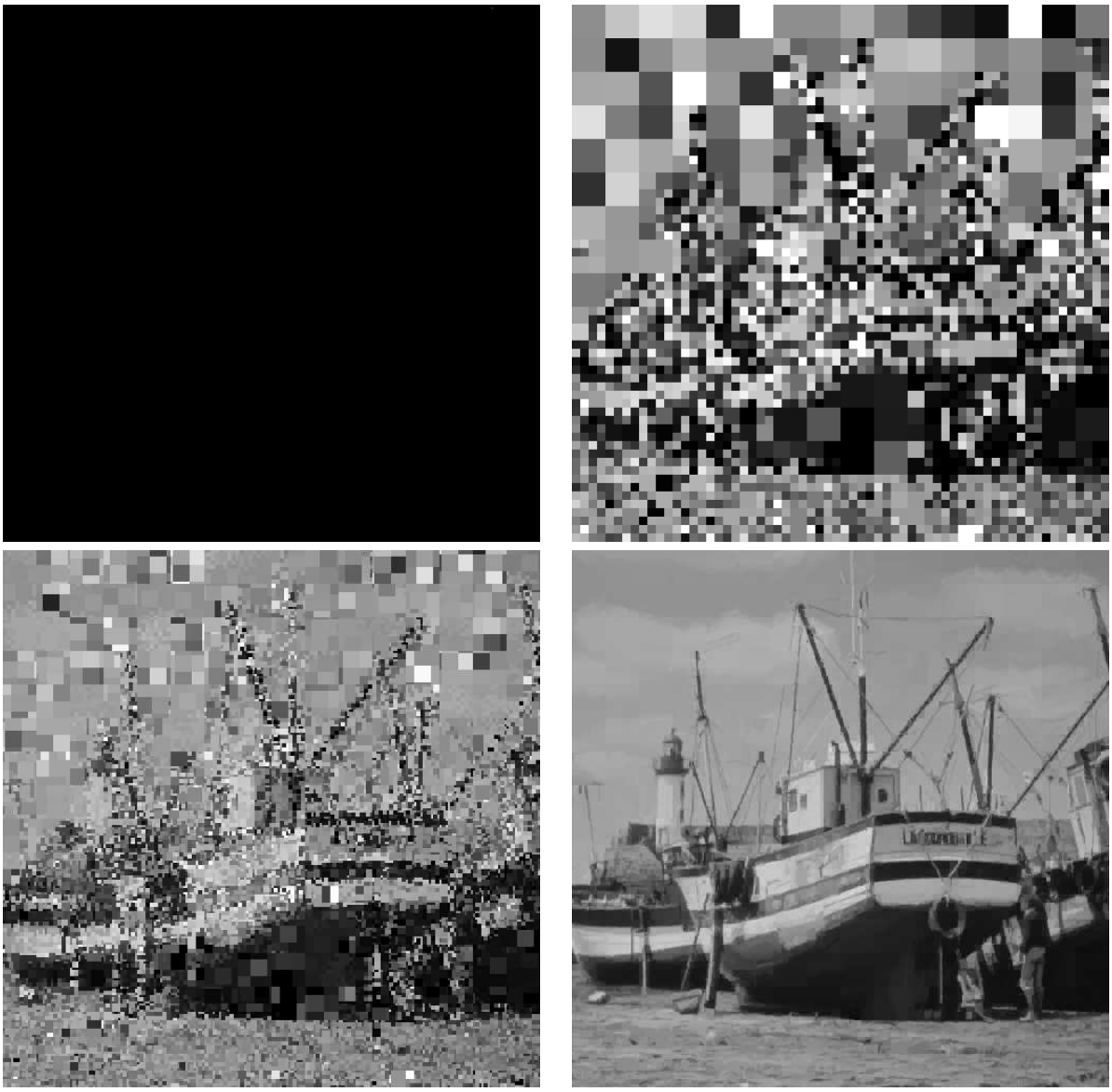

Fig. 2. An initial black image (upper left), and the first (upper right), the second (lower left), and tenth iterate (lower right).

l is equal to 4. There are 2^5 and 2^7 quantization levels for $\hat{s}_{p,k}$ and $\hat{o}_{p,k}$, respectively. Thus, the code for T consists of the bits for the quadtree partition and for each range:

- 14 bits to specify the domain position.
- 3 bits for the permutation.
- 5 bits for the scaling factor.
- 7 bits for the offset.

Figure 2 illustrates the convergence of the decoding procedure. The iteration is started from a black image $f^{(0)}$, where $f^{(0)}(i,j) = 0$ for all $i,j = 0,\ldots,511$.

2. Results and Discussion

We first introduce notation and recall some basic results of the theory of linear iterative methods. Let $A = (a_{u,v})$ and $B = (b_{u,v})$ be two $N \times M$ matrices. Then, $A \geq B (> B)$ if $a_{u,v} \geq b_{u,v} (> b_{u,v})$ for all $1 \leq u \leq N,\ 1 \leq v \leq M$. We also use these notations to compare N dimensional vectors by seeing them as $N \times 1$ matrices. Finally, $|A|$ denotes the matrix with entries $|a_{u,v}|$.

Let A be a matrix with spectral radius $\rho(A) < 1$. Then the asymptotic rate of convergence or simply the rate of convergence of the iterative method

$$\mathbf{x}^{(k+1)} = A\mathbf{x}^{(k)} + \mathbf{b}$$

is $-\ln \rho(A)$.

Definition 2. For $N \times N$ matrices A, M_1, and M_2, $A = M_1 - M_2$ is a regular splitting of the matrix A if M_1 is nonsingular with $M_1^{-1} \geq O$, and $M_2 \geq O$.

Theorem 1. *(Theorem 3.15 in [12] and Corollary p. 125 in [13]). Let $A = M_1 - N_1 = M_2 - N_2$ be two regular splittings of A, where A is nonsingular and $A^{-1} \geq O$. If $N_2 \geq N_1$ then*

$$\rho(M_2^{-1}N_2) \geq \rho(M_1^{-1}N_1).$$

If in addition $A^{-1} > O$ and neither N_1 nor $N_2 - N_1$ is the null matrix, then the above inequality is strict.

2.1. The proposed decoding procedure

In this section, we present the proposed decoding method, analyze its convergence and compare its rate of convergence to that of the conventional method.

The conventional decoding method (2) can be written

$$x_u^{(k+1)} = \sum_{v=1}^{N} a_{u,v} x_v^{(k)} + b_u,\ u = 1, \ldots, N. \quad (3)$$

Thus, it is necessary at each step to keep all the components (pixel intensities) of the $k-$th image iterate $\mathbf{x}^{(k)}$ until we finish the computation of $\mathbf{x}^{(k+1)}$. We propose instead, as in the Gauss-Seidel method, to use the new components $x_1^{(k+1)}, \ldots, x_{u-1}^{(k+1)}$ instead of the old components $x_1^{(k)}, \ldots, x_{u-1}^{(k)}$ for the computation of $x_u^{(k+1)}$. In this way, we have for $u = 2, \ldots, N$

$$x_u^{(k+1)} = \sum_{v=1}^{u-1} a_{u,v} x_v^{(k+1)} + \sum_{v=u}^{N} a_{u,v} x_v^{(k)} + b_u. \quad (4)$$

One straightforward advantage of the proposed method over the conventional method is that it has less storage requirements. Indeed, in (4) we do not need the simultaneous storage of $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(k+1)}$ as in conventional decoding (3).

We now give a matrix form of (4), which will be useful in the analysis of the convergence properties. Let the matrix A be expressed as the matrix sum

$$A = L + U, \quad (5)$$

where the matrix $L = (l_{u,v})$ is strictly lower triangular defined by

$$l_{u,v} = \begin{cases} 0, & \text{if } v \geq u; \\ a_{u,v}, & \text{otherwise.} \end{cases}$$

Then the proposed decoding scheme corresponds to the iterative method

$$\mathbf{x}^{(k+1)} = L\mathbf{x}^{(k+1)} + U\mathbf{x}^{(k)} + \mathbf{b}, \quad (6)$$

or equivalently

$$\mathbf{x}^{(k+1)} = (I - L)^{-1}U\mathbf{x}^{(k)} + (I - L)^{-1}\mathbf{b}. \quad (7)$$

We now have

Theorem 2. *If $|s_i| < 1$ for all $1 \leq i \leq n_R$, then the iterative method (7) is convergent. Moreover, method (7) and the conventional method (2) have the same limit vector $\mathbf{x}_T$.*

Proof: From (6), it is trivial that if method (7) converges, then the limit vector is $\mathbf{x}_T$. We now show that the infinity norm of the matrix $(I - L)^{-1}U$ of the iterative method (7) is strictly smaller than one, which ensures convergence. This can be done exactly as in the proof of the convergence of the Gauss-Seidel method under the assumption of strict diagonal dominance in [14] p. 578. Nevertheless, we give here the proof for completeness. Since $|A| = |L| + |U|$, we have

$$\begin{aligned} |U|\mathbf{1} &= (|A| - |L|)\mathbf{1} \\ &\leq (||A||_\infty I - |L|)\mathbf{1}. \end{aligned}$$

Thus

$$\begin{aligned} |(I-L)^{-1}U|\mathbf{1} &\leq |(I-L)^{-1}||U|\mathbf{1} \\ &\leq (I-|L|)^{-1}|U|\mathbf{1} \\ &\leq (I-|L|)^{-1}(\|A\|_\infty I-|L|)\mathbf{1} \\ &= (I-|L|)^{-1}(I-|L| \\ &\quad +(\|A\|_\infty-1)I)\mathbf{1} \\ &= (I+(\|A\|_\infty-1)(I-|L|)^{-1}))\mathbf{1}. \end{aligned}$$

But $\|A\|_\infty < 1$ from Proposition 1 and $(I-|L|)^{-1} = I+|L|+\cdots+|L|^{N-1} \geq I$. Hence

$$\begin{aligned} |(I-L)^{-1}U|\mathbf{1} &\leq (I+(\|A\|_\infty-1)I)\mathbf{1} \\ &= \|A\|_\infty\mathbf{1}, \end{aligned}$$

which gives

$$\|(I-L)^{-1}U\|_\infty = \||(I-L)^{-1}U|\|_\infty \leq \|A\|_\infty < 1.$$

□

2.1.1. Rate of convergence We now compare the decoding speed of the two methods. The speed of an iterative method is determined by the following criteria:

- The number of operations per iteration step.
- The number of iteration steps needed for convergence.

Clearly, the number of operations per iteration step is the same in conventional decoding and in the proposed decoding (compare equations (3) and (4)). The second criterion is commonly measured by the rate of convergence of the method [12]. In the following, we compare the rates of convergence, or equivalently the spectral radii, of the iteration matrices of methods (2) and (7). We begin with a preliminary result.

Lemma 1. *Let* $0 \leq s_i < 1$ *for all* $i \in \{1,\ldots,n_R\}$ *or* $-1 < s_i \leq 0$ *for all* $i \in \{1,\ldots,n_R\}$. *Then* $I-|A|$ *is nonsingular. Moreover,* $(I-|A|)^{-1} \geq O$

Proof: If λ is an eigenvalue of $I-|A|$, then $1-\lambda$ is an eigenvalue of $|A|$. But since all scaling factors have the same sign, we have $\rho(|A|) = \rho(A) < 1$. Thus $\lambda \neq 0$, which proves that $I-|A|$ is nonsingular. We have for $k > 0$,

$$(I-|A|)^{-1}-(I+|A|+\cdots+|A|^k) = (I-|A|)^{-1}|A|^{k+1}.$$

Thus

$$\begin{aligned} \|(I-|A|)^{-1}-(I+|A|+\cdots+|A|^k)\| \leq \\ \|(I-|A|)^{-1}\| \cdot \||A|^{k+1}\|. \end{aligned}$$

Since $\rho(|A|) < 1$, the sequence $\{\||A|^k\|\}_k$ converges to zero. Thus, by letting k tend to ∞ in the above inequality, we obtain

$$(I-|A|)^{-1} = \lim_{k\to\infty} I+|A|+\cdots+|A|^k,$$

which shows that $(I-|A|)^{-1} \geq O$. □

We now prove that if all scaling factors have the same sign, then the rate of convergence of the proposed method is greater than or equal to that of the conventional method.

Proposition 2. *Suppose that* $0 \leq s_i < 1$ *for all* $i \in \{1,\ldots,n_R\}$ *or* $-1 < s_i \leq 0$ *for all* $i \in \{1,\ldots,n_R\}$. *Then* $\rho((I-L)^{-1}U) \leq \rho(A)$.

Proof: Define the matrix $M = I-|A| = (I-|L|)-|U|$. Clearly, these two splittings of M are regular. Furthermore, $|A| \geq |U|$, and by Lemma 1, $M^{-1} \geq O$. The inequality $\rho(|A|) \geq \rho((I-|L|)^{-1}|U|)$ follows then from Theorem 1. We complete the proof by noting that $\rho(A) = \rho(|A|)$ and $\rho((I-L)^{-1}U) \leq \rho((I-|L|)^{-1}|U|)$. □

If the scaling factors are not of the same sign, we cannot ensure the inequality in Proposition 2. We argue that this is not a severe limitation for two reasons. First, Saupe [15] observed that restricting the scaling factors to be positive does not degrade the rate-distortion performance significantly. In fact, this was the choice of Jacquin in his original work [1], and seems also to be preferred by Iterated Systems, Inc. [16]. Second, our numerical results show that the proposed decoding method converges faster even when the scaling factors do not all have the same sign (see next section).

It is possible that the conventional method converges even when some of the scaling factors are larger than one in absolute value. In such cases, if we assume that all scaling factors have the same

sign, then, as easily derived from the proof of Proposition 2, the proposed method is also convergent, and its rate of convergence is greater than or equal to that of the conventional method. However, if we drop the condition that all the scaling factors have the same sign, then the convergence of method (7) cannot be guaranteed any more as shown by the following example. Let

$$A = \frac{1}{2}\begin{pmatrix} 5 & 5 & 0 & 0 \\ 0 & 0 & 5 & 5 \\ -4 & -4 & 0 & 0 \\ 0 & 0 & -4 & -4 \end{pmatrix}.$$

The matrix corresponds to a 2×2 image partitioned in two ranges of size 2. The domain is the image support and the scaling factors are 5 and -4. We have $\rho(A) = \frac{1}{2} < 1$. But

$$(I - L)^{-1}U = \begin{pmatrix} \frac{5}{2} & \frac{5}{2} & 0 & 0 \\ 0 & 0 & \frac{5}{2} & \frac{5}{2} \\ -5 & -5 & -5 & -5 \\ 10 & 10 & 10 & 8 \end{pmatrix}$$

and

$$\rho((I - L)^{-1}U) = \frac{1}{4}(11 + \sqrt{41}) > 1.$$

2.1.2. Experimental results The tables in this section show the root mean square (rms) error (square root of the average of the squared error) between the fixed point $\mathbf{x}_T$ and the iterate $\mathbf{x}^{(k)}$ as a function of the iteration step k. The iteration was always started from the image vector $\mathbf{x}^{(0)} = (0, \ldots, 0)^T$.

The first two tables provide results for encodings with Fisher's quadtree scheme [10] (see Section 1.2). Table 1 shows results for the 512×512 Bridge image. We used a uniform partition into 4096 square ranges of size 8×8. We allowed only nonnegative scaling factors. The grid spacing was equal to 2, yielding 62001 domains of size 16×16.

Table 2 shows results for the 512×512 Lenna image for a quadtree partition with three range sizes: 16×16, 8×8, and 4×4. The grid spacing was equal to four. Thus, we had 16129 domains of size 8×8, 15625 domains of size 16×16, and 14641 domains of size 32×32. The rms tolerance value t was equal to eight. Both positive and negative scaling factors were allowed.

Table 1. Convergence of the decoding for the two methods measured by the root mean square error (rms_k) between the k-th iterate and the fixed point for the 512×512 Bridge image. Only nonnegative scaling factors are used.

Iteration k	Conventional rms_k	Conventional $\frac{rms_k}{rms_{k-1}}$	Proposed method rms_k	Proposed method $\frac{rms_k}{rms_{k-1}}$
1	82.821		61.368	
2	50.460	0.61	26.650	0.43
3	30.195	0.60	10.313	0.39
4	17.470	0.58	3.666	0.36
5	9.801	0.56	1.270	0.35
6	5.395	0.55	0.441	0.35
7	2.937	0.54	0.151	0.34
8	1.590	0.54	0.051	0.34
9	0.858	0.54	0.017	0.34
10	0.462	0.54	0.006	0.34
11	0.249	0.54	0.002	0.34
12	0.134	0.54	0.001	0.34

Table 2. Convergence of the decoding for the two methods measured by the root mean square error between an iterate and the fixed point for the 512×512 Lenna image. Both positive and negative scaling factors are allowed.

Iteration k	Conventional rms_k	Conventional $\frac{rms_k}{rms_{k-1}}$	Proposed method rms_k	Proposed method $\frac{rms_k}{rms_{k-1}}$
1	72.393		58.546	
2	42.530	0.59	24.600	0.42
3	25.455	0.60	8.933	0.36
4	12.560	0.49	2.511	0.28
5	5.390	0.43	0.632	0.25
6	2.139	0.40	0.152	0.24
7	0.834	0.39	0.040	0.26
8	0.332	0.40	0.011	0.29
9	0.126	0.38	0.003	0.29
10	0.047	0.37	0.001	0.29

Table 3 compares the decoding convergence of the two methods for the 512×512 Barbara image. Here, we encoded the image with the scheme of Ruhl *et al.* [17], which generates ranges having more irregular shapes (Figure 3).

In all cases, the proposed decoding scheme converged faster than the conventional one. We observed similar results for all other tested images.

2.2. Standard methods

In this section, we discuss the use of standard iterative methods for the decoding and compare them to the proposed method. We assume throughout

Table 3. Convergence of the decoding for the two methods measured by the root mean square error between an iterate and the fixed point for the 512×512 Barbara image. Both positive and negative scaling factors are allowed.

Iteration step	Conventional	Proposed method
1	63.995	44.340
2	33.111	15.051
3	18.252	4.464
4	8.470	1.229
5	3.561	0.321
6	1.373	0.080
7	0.501	0.018
8	0.162	0.004

this section that

$$0 \leq s_i < 1 \text{ for all } i \in \{1, \ldots, n_R\}.$$

The decoding problem consists of finding the fixed point $\mathbf{x}_T = A\mathbf{x}_T + \mathbf{b}$, which is the solution of the equation

$$(I - A)\mathbf{x} = \mathbf{b}. \tag{8}$$

Using the notation of Section 2.1, if the matrix U in (5) is expressed as the matrix sum $U = D + F$, where D is the diagonal matrix obtained from A by setting the nondiagonal entries to zero, and F is strictly upper triangular, then the point Jacobi method associated to the solution of equation (8) is given by the iterative method

$$\mathbf{x}^{(k+1)} = (I-D)^{-1}(A-D)\mathbf{x}^{(k)} + (I-D)^{-1}\mathbf{b}, \tag{9}$$

and the point Gauss-Seidel method associated to the same equation is given by the iterative method

$$\mathbf{x}^{(k+1)} = (I - D - L)^{-1}F\mathbf{x}^{(k)} + (I - D - L)^{-1}\mathbf{b}. \tag{10}$$

When $0 \leq s_i < 1$ for all $i \in \{1, \ldots, n_R\}$, both methods are convergent because $I - A$ is strictly diagonally dominant [12].

Note that the conventional method (2) (the proposed method (7)) reduces to the point Jacobi method (the point Gauss-Seidel method) if and only if $a_{u,u} = 0$ for all $u = 1, \ldots, N$. Experiments with the quadtree scheme of Fisher [10] and the scheme in [17] show, however, that there always exists $u \in \{1, \ldots, N\}$ such that $a_{u,u} \neq 0$, that is, there always exists at least one range in which a pixel (i, j) is encoded by a group of domain pixels that include pixel (i, j) itself.

For methods (9) and (10) to be useful in fractal image compression, they should provide a computational gain over the conventional decoding method. For the rate of convergence we have

Proposition 3.

$$\rho((I - D)^{-1}(A - D)) \leq \rho(A),$$

and either

$$0 = \rho((I - D - L)^{-1}F) = \rho((I - D)^{-1}(A - D))$$

or

$$0 < \rho((I - D - L)^{-1}F) < \rho((I - D)^{-1}(A - D)).$$

Proof: We have $I - A = (I - D) - (A - D)$, and these two splittings are regular. Moreover, $(I - A)^{-1} \geq O$ from Lemma 1 and $A - D \leq A$. Thus, by Theorem 1 we obtain the first inequality. The rest of the proof follows from the Stein-Rosenberg Theorem [12]. □

We now compare the number of arithmetical operations involved in the methods.

Proposition 4. *Both the point Jacobi method (9) and the point Gauss-Seidel method (10) need at most n_R more arithmetical operations per iteration step than the conventional method (2).*

Proof: Clearly, methods (9) and (10) need the same number of arithmetical operations. Thus, it is sufficient to show that the result holds for method (9). Let us write (9) in the equivalent form

$$(1 - a_{u,u})x_u^{(k+1)} = \sum_{v=1,\ v \neq u}^{N} a_{u,v}x_v^{(k)} + b_u.$$

First, recall that for a fixed u, the entry $a_{u,v}$ is equal either to zero or to s_i/m_i (see Section 1.1). If $a_{u,u} = 0$, then the same number of arithmetical operations is required to compute $x_u^{(k+1)}$ by methods (9) and (2). Otherwise, we have for (9) one addition less, but one subtraction and one division more. We conclude by observing that the subtraction $1 - a_{u,u}$ has to be computed only once for all indices u corresponding to the pixels in the same range. □

Even though the two methods (9) and (10) do not require many more arithmetical operations

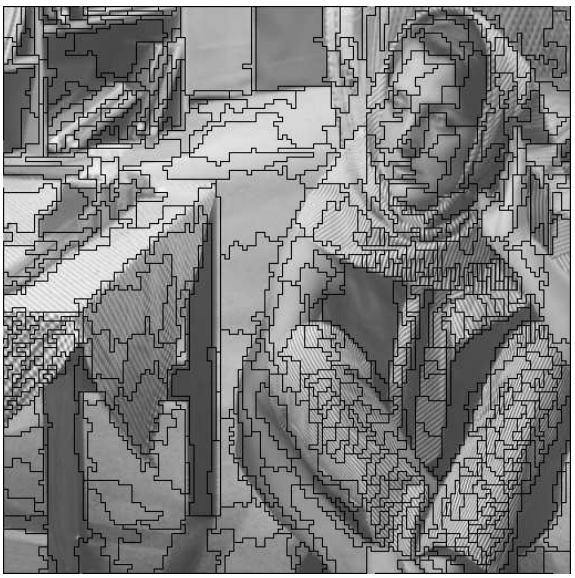

Fig. 3. Partition of the 512 × 512 Barbara image into 600 ranges of various sizes and shapes.

than the conventional method (2), they demand for all $u = 1, \ldots, N$ testing if $a_{u,u} = 0$. But since the matrix A is not explicitly available at the decoder, this leads to an undesirable additional computational work, which is not shared by the proposed method (7). Nevertheless, we implemented the Gauss-Seidel method and compared its L_2 convergence to that of method (7). Numerical results for several images showed that the methods had almost identical behavior.

One may be tempted to try the sophisticated nonstationary iterative methods [18]. Unfortunately, since in fractal coding the iteration matrix $I - A$ is nonsymmetric, a large number of these techniques are not appropriate. Furthermore, unlike the proposed method, those which could be used such as conjugate gradient on the normal equations (CGNE and CGNR) or generalized minimal residual (GMRES) require more arithmetic operations and storage per iteration step than the conventional method. Thus, since in fractal image compression the decoding typically converges in a few iterations (see numerical results), the possible reduction in iteration steps will be severely penalized by the extra costs.

Øien et *al.* [19] and Baharav et *al.* [20] proposed other efficient methods for fast decoding. However, these methods were described for special cases where, particularly, each domain used in the code has to be a union of ranges. This can be a serious restriction because unconstrained domains provide a better rate-distortion performance [10, 17, 21]. Furthermore, the scheme in [20] uses the conventional method at a lower image resolution and interpolates the obtained fixed point to the real size. Thus, this technique can be easily combined with the method proposed in this paper.

2.3. Ordered decoding algorithm

With the conventional method (3), changing the order in which the pixel intensities are computed has no effect on the image iterates $\{\mathbf{x}^{(k)}\}_k$. This is clearly not true for the proposed method (see equation (4)). In this section, we propose to apply the proposed method starting with the pixels in the ranges with highest frequency [22, 9]. The frequency of an image support is defined as follows:

Definition 3. Let $\mathcal{D}_1, \ldots, \mathcal{D}_{n_R}$ be the domains used in the decoding. Let $\mathcal{B}$ be a nonempty subset of the image support. Then the frequency of $\mathcal{B}$ is

$$\frac{\sum_{i=1}^{n_R} card(\mathcal{B} \bigcap \mathcal{D}_i)}{card(\mathcal{B})},$$

where $card(\mathcal{B})$ is the number of pixels in $\mathcal{B}$.

The proposed ordered decoding is illustrated in Figure 4.

The expected gain with the new ordering is twofold. First, this strategy increases the number of pixels that are used in their updated form, yielding, hopefully, a faster convergence. Second, because of the sorting of the frequencies, the decoder can identify the ranges that are not covered by the domains, that is, ranges with zero frequency. The pixels intensities in these ranges, as pointed out in [23, 24] in a different context, need only be computed once during the decoding. The following proposition justifies this observation.

Proposition 5. *Let $A = (a_{u,v})$ be the $N \times N$ matrix and $\mathbf{b} = (b_u)$ be the vector of dimension N associated to the affine mapping T providing the code of an image f. Suppose that $\rho(A) < 1$. Let*

$\mathcal{R}_1$ 2	$\mathcal{R}_2$ 2	$\mathcal{R}_3$ 6	$\mathcal{R}_4$ 6
$\mathcal{R}_5$ 2	$\mathcal{R}_6$ 2	$\mathcal{R}_7$ 6	$\mathcal{R}_8$ 6
$\mathcal{R}_9$ 0	$\mathcal{R}_{10}$ 8	$\mathcal{R}_{11}$ 8	$\mathcal{R}_{12}$ 0
$\mathcal{R}_{13}$ 0	$\mathcal{R}_{14}$ 8	$\mathcal{R}_{15}$ 8	$\mathcal{R}_{16}$ 0

Fig. 4. Image partition into 16 ranges. The range frequency is indicated on each range. With the proposed technique, the ranges are decoded in the following order: $\mathcal{R}_{10}, \mathcal{R}_{11}, \mathcal{R}_{14}, \mathcal{R}_{15}, \mathcal{R}_3, \mathcal{R}_4, \mathcal{R}_7, \mathcal{R}_8, \mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_5, \mathcal{R}_6, \mathcal{R}_9, \mathcal{R}_{12}, \mathcal{R}_{13}, \mathcal{R}_{16}$.

$\mathbf{x}_T$ *denote the fixed point of* T. *Now suppose that* A *has zero entries at the* p *columns* $j_1, \ldots, j_p$. *Let* $\{i_1, \ldots, i_{N-p}\} = \{1, \ldots, N\} \backslash \{j_1, \ldots, j_p\}$. *Call* A' *the* $(N-p) \times (N-p)$ *matrix obtained from* A *by discarding both columns and rows* $j_1, \ldots, j_p$. *Call* T' *the operator defined by* $T'(\mathbf{x}) = A'\mathbf{x} + \mathbf{b}'$, *where* $b'_u = b_{i_u}$, $u = 1, \ldots, N-p$. *Then we have:*
1. $\rho(A) = \rho(A')$.
2. Let $\mathbf{x}_{T'} = (x'_1, \ldots, x'_{N-p})^T$ *denote the fixed point of* T'. *Then* $\mathbf{x}_T = (x_1, \ldots, x_N)^T$ *is given by*

$$x_{i_u} = x'_u \text{ for } u = 1, \ldots, N-p$$

and

$$x_{j_k} = \sum_{v \in \{1,\ldots,N-p\}} a_{j_k, i_v} x'_v + b_{j_k} \text{ for } k = 1, \ldots, p.$$

Proof: 1. Without loss of generality we can assume that the first p columns are null. Thus, A can be written in the form

$$A = \begin{pmatrix} 0 & \ldots & 0 & a_{1,p+1} & \ldots & a_{1,N} \\ 0 & \ldots & 0 & a_{2,p+1} & \ldots & a_{2,N} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \ldots & 0 & a_{N,p+1} & \ldots & a_{N,N} \end{pmatrix},$$

which has the block matrix form

$$A = \begin{pmatrix} O_1 & A_1 \\ O_2 & A' \end{pmatrix},$$

where O_1 and O_2 are zero matrices of size $p \times p$ and $(N-p) \times p$, respectively. Thus, $\det(\lambda I - A) = \lambda^p \det(\lambda I - A')$, which shows that A and A' have the same spectral radius. 2. One simply checks that $A\mathbf{x}_T + b = \mathbf{x}_T$. □

Since pixels with zero frequency correspond to zero columns in the matrix A, the proposition implies that if the decoding is convergent, then one can find the fixed point by iteratively computing the intensities of the pixels with nonzero frequency until convergence, and then using the result to compute the intensities of the pixels with zero frequency in one final step. Clearly, this property also holds for the proposed decoding.

The pseudo-code of the ordered decoding algorithm is as follows:

1. Compute the frequency of each range.
2. Order the ranges according to decreasing frequency.
3. Until convergence is achieved, at each iteration, compute (according to equation (4)) the intensity of pixels in ranges with nonzero frequency in the above order.
4. Compute the intensity of pixels in ranges with zero frequency.

Note that the additional step needed for the computation and the sorting of the range frequencies is negligible because it is done once.

Alternatively, one could use the frequency of each single pixel and decode according to decreasing pixel frequency. But in this case, the sorting costs may become too expensive. Moreover, for an efficient implementation, the decoder would need a data structure that has much more storage requirements than the original one.

2.3.1. Experimental results In this section, we compare the following methods:

1. Conventional decoding (Fisher's quadtree scheme).
2. Proposed decoding with the ordering of the original data structure (natural ordering).
3. Proposed decoding with the ordering based on the range frequencies.

In our implementation of the frequency-based decoding, we used as a data structure for the range

Table 4. Convergence of the decoding for the three methods for the 512 × 512 Lenna image. The values given in the table are the root mean square errors between the fixed point and successive iterates of an initial black image.

Iteration step	1	2	3
1	73.145	59.668	46.834
2	42.252	25.161	17.106
3	24.606	9.524	4.762
4	14.041	2.337	0.923
5	4.559	0.381	0.147
6	1.324	0.074	0.029
7	0.373	0.017	0.007
8	0.108	0.004	0.002

These labels are used in the table above:
1 : Conventional decoding
2 : Proposed decoding with the natural ordering
3 : Proposed decoding with the frequency-based ordering

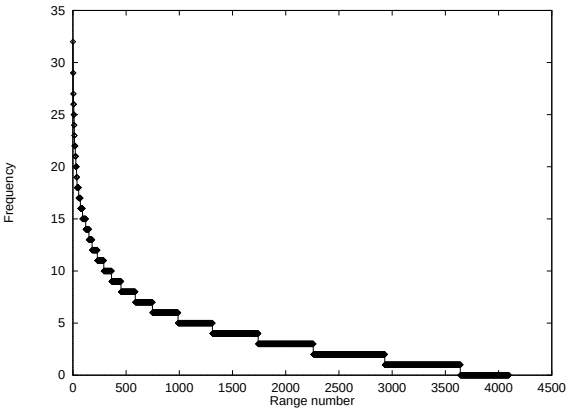

Fig. 5. Range frequency for the 512 × 512 Lenna image. Ranges are numbered according to decreasing frequency. The partition contains 455 ranges with zero frequency.

transformations an array instead of a linked list and ordered the frequencies of the ranges with the quicksort algorithm.

Table 4 shows results for the 512 × 512 Lenna image, which was encoded with a uniform partition in square ranges of size 8 × 8. The maximum scaling factor was equal to 0.99. The domains were the square blocks of size 16 × 16 obtained with a grid spacing of 8 pixels. Figure 5 gives statistics on the frequency of each range block. Figure 6 illustrates the range frequencies.

To give a good estimation of the convergence speed, pixel intensities were computed in double precision. However, in some implementations, the pixel intensities are rounded to integers at each

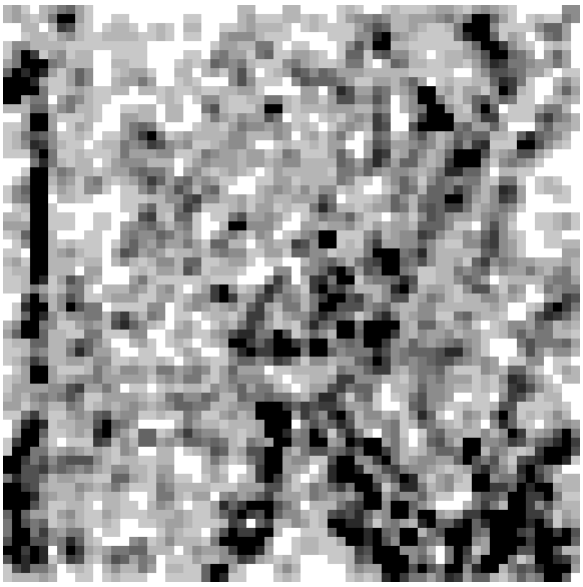

Fig. 6. Gray scale image illustrating the range frequencies in the 512 × 512 Lenna image. The intensity decreases with increasing range frequency. White areas correspond to the 455 ranges with zero frequencies.

Table 5. Decoding speed of the three methods for the 512 × 512 Couple image encoded at 0.59 bpp. The CPU time is given in seconds. The last column indicates the iteration step at which convergence occurred.

Decoding method	Time (sec)	Iteration
Conventional	1.80	10
Proposed (natural ordering)	1.31	8
Proposed (frequency ordering)	1.38	7

iteration to save memory. In this situation also, the proposed methods converged faster than the conventional method. Figure 7 shows for the original monochrome 8 bits per pixel (bpp) 512 × 512 Couple image the second iterate of each decoding scheme when rounding was used. The maximum range size was 16 × 16 and the minimum size was 4 × 4. The grid spacing was equal to 2, and the rms tolerance was equal to 12.

Table 5 compares the decoding speed of the methods. CPU times were measured on an SGI O2 running an MIPS R5000 180 MHz processor. The decoding was stopped when the rms error between two successive image iterates was less than 0.2. Compared to conventional decoding, the proposed decoding with the natural ordering saved 512 × 512 × sizeof(unsigned char) bytes of space

Fig. 7. The fixed point image (upper left), and the second iterate with respectively the conventional decoding (upper right), the proposed method with the natural ordering (lower left), and the proposed method with the ordering based on range frequencies (lower right). The partition consists of 4879 ranges, 190 of which with zero frequency.

because only one image array was used at each iteration. On the other hand, the proposed decoding with the range frequency-based ordering saved the same amount of space, but it required in addition 512 × 512 × sizeof(int) bytes to store the pixel frequencies (we needed the pixel frequencies for an efficient computation of the range frequencies), 4879 × sizeof(int) bytes for the range frequencies, and 4879 × sizeof(int) bytes for an index table used for sorting without moves of the items in the transformations array.

The experimental results confirm that convergence with frequency-based decoding is faster than convergence with the natural ordering, which itself is faster than convergence with conventional decoding. Moreover, the decoding time of the two proposed methods was shorter than that of conventional decoding. However, because of the preprocessing step, the decoding time of the frequency-based decoding was slightly longer than that of the proposed decoding with the natural ordering. Nevertheless, one may still prefer the frequency-based ordering because it provides at each iteration step a better approximation of the original image.

References

1. A. E. Jacquin, "Image coding based on a fractal theory of iterated contractive image transformations," IEEE Trans. Image Processing 1 (1992) 18–30.
2. Y. Fisher (ed.), *Fractal Image Compression — Theory and Application,* Springer-Verlag, New York, 1994.
3. M. Barnsley, *Fractals Everywhere,* Academic Press, San Diego, 1988.
4. D. Saupe, R. Hamzaoui, "Complexity reduction methods for fractal image compression," in: Proc. of I.M.A. Conf. on Image Processing: Mathematical Methods and Applications, Sept. 1994, J.M. Blackledge (ed.), pp. 211–229, Oxford University Press, 1997.
5. H. Kaouri, "Fractal coding of still images," in: Proc. IEE 6th International Conference on Digital Processing of Signals in Communications, pp. 235–239, 1991.
6. M. Nelson, J.-L. Gailly, *The Data Compression Book, 2nd Edition*, M & T books, New York, 1995.
7. R. Hamzaoui, "Decoding algorithm for fractal image compression," Electronics Letters 32,14 (1996) 1273–1274.
8. R. Hamzaoui, "Fast decoding algorithms for fractal image compression," in: Proc. of the Conference Fractals in Engineering, Arcachon, June 1997.
9. H.-S Kang, S.-D. Kim, "Fractal decoding algorithm for fast convergence," Optical Engineering 35,11 (1996) 3191–3198.
10. Y. Fisher, "Fractal image compression with quadtrees," in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.
11. L. M. Lundheim, "A discrete framework for fractal signal modelling," in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.
12. R. S. Varga, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, N. J., 1962.
13. D. M. Young, *Iterative Solution of Large Linear Systems,* Academic Press, New York, 1971.
14. J. Stoer, R. Bulirsch, *An Introduction to Numerical Analysis,* Springer Verlag, New York, 1992.
15. D. Saupe, "The futility of square isometries in fractal image compression," in: Proc. ICIP-96 IEEE Int. Conf. on Image Processing, Vol. I, pp. 161–164, Lausanne, Sept. 1996.
16. N. Lu, *Fractal Imaging,* Academic Press, 1997.
17. M. Ruhl, H. Hartenstein, D. Saupe, "Adaptive partitionings for fractal image compression," in: Proc. ICIP-1997 IEEE International Conference on Image Processing, Volume II, pp. 310–313, Santa Barbara, California, Oct. 1997.
18. R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, H. van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods,* SIAM, Philadelphia, PA, 1994.
19. G. E. Øien, S. Lepsøy, "A class of fractal image coders with fast decoder convergence," in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.
20. Baharav, Z., Malah, D., Karnin, E., "Hierarchical interpretation of fractal image coding and its applications," in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.
21. Y. Fisher, S. Menlove, "Fractal encoding with HV partitions," in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.
22. R. Hamzaoui, "Ordered decoding algorithm for fractal image compression," in: Proc. PCS'97 Picture Coding Symposium, pp. 91–95, Berlin, Sept. 1997.
23. B. Hürtgen, S. F. Simon, "On the problem of convergence in fractal coding schemes," in: Proc. ICIP-94 IEEE Int. Conf. on Image Processing, Vol. 3, pp. 103–106, Austin, Texas, 1994.
24. J. Domaszewicz, V. A. Vaishampayan, "Graph-theoretical analysis of the fractal transform," in: Proc. ICASSP-1995 IEEE Int. Conf. on Acoustics, Speech and Signal Processing, Vol. 4, pp. 2559–2562, Detroit, 1995.

Raouf Hamzaoui is a researcher at the University of Leipzig, Germany. He received the Maîtrise de

mathématiques from the Faculty of Sciences of Tunis, Tunisia, in 1986, the M.Sc. degree in mathematics from the University of Montreal, Canada, in 1993, and the Dr. Rer. nat. degree from the Faculty of Applied Sciences of the University of Freiburg, Germany, in 1997. His research interests include digital image processing and numerical methods.