Fast Decoding Algorithms for Fractal Image Compression

Institut für Informatik — Report 86[1]

Raouf Hamzaoui
Institut für Informatik, Universität Freiburg
Am Flughafen 17, 79110 Freiburg, Germany
Email: hamzaoui@informatik.uni-freiburg.de.

March 18, 1997

[1]This paper will be presented at the conference Fractals in Engineering, Arcachon, June 1997

Abstract

A new decoding technique based on an iterative method with pixel updating is proposed for fractal image compression. We prove that the new iterative method converges in the most general case of variable shape segmentations, unconstrained domains, and use of pixel shuffling. We show that in some important cases the new method has a greater rate of convergence than the conventional method. Furthermore, it is indicated how standard iterative methods can be efficiently implemented in fractal image compression. Finally, experimental results confirming the superiority of our technique are presented.

1 Introduction

Fractal image compression [1, 2, 3] suffers from long encoding times. However, the decoding is both simple and fast. This makes it a good candidate for storage and retrieval applications where the encoding is performed once using special hardware, while the decoding is to be repeated several times in software by the user. One such application is CD-ROM encyclopedia for personal computers. So it is worth asking if one can further simplify the decoding algorithm for those environments.

The conventional decoding algorithm can be described as follows [1]. First, the original image $\mathbf{x}^* \in \mathbf{R}^N$ is stored implicitly as the unique fixed point $\mathbf{x}_T$ of a contractive affine mapping T defined by $T(\mathbf{x}) = A\mathbf{x} + \mathbf{b}$, where A is a real $N \times N$ sparse matrix and $\mathbf{b}$ is a real column vector of dimension N. The reconstruction of the fixed point $\mathbf{x}_T$ by the decoder proceeds by iteratively applying T to any arbitrary initial image $\mathbf{x}^{(0)}$. Banach's fixed point theorem ensures the convergence of the sequence of iterates $\{T^{(k)}(\mathbf{x}^{(0)})\}_k$ to the fixed point $\mathbf{x}_T = (I - A)^{-1}\mathbf{b}$, where I is the identity matrix of order N. Here a vector $T^{(k)}(\mathbf{x}^{(0)})$ which we will also denote by $\mathbf{x}^{(k)}$ is defined by the recurrence relation

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

The fixed point $\mathbf{x}_T$ is also known as the attractor of T. The encoding consists of finding a contractive affine mapping T such that the attractor $\mathbf{x}_T$ and $\mathbf{x}^*$ are as close as possible. The usual approach to handle this problem is by trying to minimize the distance between $\mathbf{x}^*$ and $T(\mathbf{x}^*)$. The success of this approach has been attributed to the Collage theorem [4] [1], a corollary of Banach's fixed point theorem which states that

$$\|\mathbf{x}_T - \mathbf{x}^*\| \leq \frac{1}{1 - \|A\|}\|\mathbf{x}^* - T(\mathbf{x}^*)\|.$$

In this paper we present a new decoding procedure for fractal image compression observed independently in [6, 7, 8]. It is based on an iteration technique in the spirit of the Gauss-Seidel method providing a faster convergence and relieving memory requirements at the decoding. We extend here the description of our initial algorithm to more general encoding schemes,

[1] Inconsistencies in the application of the theorem are raised in [5].

allowing variable shape partitionings and unconstrained domains with pixel shuffling. Furthermore, we provide details and proofs that were omitted in our original publication, and discuss the implementation of classic iterative methods in fractal image coding.

The rest of the paper is organized as follows. In Section 2 we present definitions and notations that will be needed throughout the text. We also discuss the convergence of the conventional iterative method. In Section 3 we explain our decoding algorithm and prove its convergence. Moreover, we argue that two standard iterative methods can be efficiently used in the decoding. Then the computational complexity of the various methods is compared. Next, it is shown that under certain fair conditions the new method has a greater rate of convergence than the conventional method. Section 4 presents experimental results. Finally, we summarize and provide suggestions for future work.

2 Notations and definitions

We begin this section with some notations and definitions. The first two definitions are borrowed from [9].

Definition 1 *Let $A = (a_{u,v})$ and $B = (b_{u,v})$ be two $n \times n$ matrices. Then, $A \geq B (> B)$ if $a_{u,v} \geq b_{u,v} (> b_{u,v})$ for all $1 \leq u \leq n,\ \ 1 \leq v \leq n$. If O is the null matrix and $A \geq O (> O)$, we say that A is a nonnegative (positive) matrix. Finally, $|A|$ denotes the matrix with entries $|a_{u,v}|$.*

Definition 2 *For $n \times n$ real 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 $M_2 \geq O$.*

Definition 3 *The spectral radius $\rho(A)$ of a matrix A is the largest absolute value of the eigenvalues of A. The rate of convergence of an iterative method with iteration matrix A is $-\ln \rho(A)$.*

Definition 4 *An image of size $N = 2^m \times 2^m$ is a real valued function f defined on $\mathcal{I} = \{1, \ldots, 2^m\} \times \{1, \ldots, 2^m\}$. The elements of $\mathcal{I}$ are called pixels. An image piece of size n is the restriction of f to a subset $\mathcal{B} \subset \mathcal{I}$ having n elements. Now let ψ be a one-to-one indexing mapping from $\mathcal{I}$ to $\{1, \ldots, N\}$.*

Through ψ we identify $\mathcal{B}$ with the subset $B = \psi(\mathcal{B}) \subset \{1, \ldots, N\}$. When there is no ambiguity on the function f, both subsets $\mathcal{B}$ and B will also be called image pieces. To each image piece $B = \{b_1, \ldots, b_n\}$ we associate an $n-$dimensional vector $\mathbf{x}_B$ such that

$$\mathbf{x}_B = (f\psi^{-1}(b_1), \ldots, f\psi^{-1}(b_n))^T.$$

This vector $\mathbf{x}_B$ will be also called an image piece. Finally, two image pieces $\mathcal{B}_1$ and $\mathcal{B}_2$ are said to be overlapping if $\mathcal{B}_1 \bigcap \mathcal{B}_2 \neq \emptyset$.

In the encoding step, an image of size N is partitioned into n_R nonoverlapping image pieces R_i of size n_i, $n_{min} \leq n_i \leq n_{max}$, called ranges. It is always possible to assume that ψ is chosen such that for $R_i = \{(r_i)_1, \ldots, (r_i)_{n_i}\}$ we have

$$(r_i)_{k+1} = (r_i)_k + 1, \quad \text{for all } i \in \{1, \ldots, n_R\} \text{ and } k \in \{1, \ldots, n_i - 1\}$$

and

$$(r_i)_{n_i} + 1 = (r_{i+1})_1, \text{ for all } i \in \{1, \ldots, n_R - 1\}.$$

Then each range $\mathbf{x}_{R_i}$, $i = 1, \ldots, n_R$, is approximated by the linear combination

$$\widehat{\mathbf{x}_{R_i}} = s_i S_i P_i \mathbf{x}_{D_i} + o_i \mathbf{1}_{n_i}, \tag{2}$$

where

- $\mathbf{1}_{n_i}$ is the constant vector $\mathbf{1}_{n_i} = (1, \ldots, 1)^T \in \mathbf{R}^{n_i}$.
- s_i and o_i are scalar quantization coefficients called scaling factor and offset, respectively.
- $D_i = \{(d_i)_1, \ldots, (d_i)_{m_i n_i}\}$ is an image piece of size $m_i n_i$ called domain, where $m_{min} \leq m_i \leq m_{max}$.
- P_i is a permutation matrix of order $m_i n_i$.
- S_i is an $n_i \times m_i n_i$ matrix defined by

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

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

For each range R_i, the domain D_i is chosen from a large set of available domains called *domain pool*. The matrix P_i shuffles the components of the domain $\mathbf{x}_{D_i}$. The matrix S_i shrinks the size of the vector $P_i \mathbf{x}_{D_i}$ to the size of the range by averaging m_i consecutive components. Finding an optimal domain from the domain pool is the most time consuming part in the encoding. We now define for each $i \in \{1, \dots, n_R\}$ a mapping

$$T_i : \mathbf{R}^N \to \mathbf{R}^N$$

such that

$$T_i(\mathbf{x}) = s_i G_i S_i P_i F_i \mathbf{x} + o_i G_i \mathbf{1}_{n_i},$$

where

- The $m_i n_i \times N$ matrix $F_i = (f^i_{u,v})$ is defined by

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

- The $N \times n_i$ matrix $G_i = (g^i_{u,v})$ is defined by

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

The operator $T(\mathbf{x}) = A\mathbf{x} + \mathbf{b}$ of the previous section is equal to $\sum_{i=1}^{n_R} T_i$. It is not difficult to see that T can model the most general encoding schemes known in the literature, where image pieces may have arbitrary shapes and sizes, domains of the same size may be overlapping, and pixel shuffling is permitted. We point out at this stage that the sparse matrix $A = \sum_{i=1}^{n_R} s_i G_i S_i P_i F_i$ is not explicitly stored at the decoder. For any image $\mathbf{x}^{(k)}$ the next iterate $T(\mathbf{x}^{(k)})$ is obtained by applying formula (2) successively to the domains $\mathbf{x}^{(k)}_{D_i}$,

$i = 1, \ldots, n_R$. The matrix A has been introduced only to help the mathematical treatment of the problem.

It is a well known fact that the convergence of the iterative method (1) is guaranteed if all scaling factors are bounded by one in absolute value. This is a simple consequence of Banach's fixed point theorem and the following proposition.

Proposition 5 *If $|s_i| < 1$ for all $i = 1, \ldots, n_R$, then T is a contraction in the L_∞ norm.*

Proof. First, observe that A is such that each row has m_i components equal to s_i/m_i and $N - m_i$ components equal to zero. Due to this simple structure we have

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

This shows that T is a contraction with contractivity $\max_{1 \leq i \leq n_R} |s_i|$. $\square$

It has been pointed out by several authors [3, 10] that the decoding may converge even if there exists $i \in \{1, \ldots, n_R\}$ such that $|s_i| \geq 1$. Indeed, a necessary and sufficient condition for the convergence of the iterative method (1) is $\rho(A) < 1$. Unfortunately, the computation of $\rho(A)$ is not an easy task. This has been done so far only in some special cases [10]. Thus to control the convergence of the decoding a priori, it is common use in fractal coding to restrict the value of the scaling factors to the interval $(-1, 1)$. In the following, this assumption will also be made.

3 New algorithms for the decoding

3.1 The new scheme

We introduce in this section our new decoding technique. Let $\mathbf{x}^{(k)}$ be the decoded image at iteration step k. In conventional decoding it is necessary to keep all the components of the image $\mathbf{x}^{(k)}$ until we achieve the computation of $\mathbf{x}^{(k+1)}$. We propose instead, as in the Gauss-Seidel method, to use the latest estimates of the components $x_u^{(k+1)}$ as soon as available. Thus one immediate advantage is that we do not need the simultaneous storage of $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(k+1)}$. Our technique can be loosely described by saying that the domains $\mathbf{x}_{D_i}^{(k)}$ are updated progressively at the same iteration k. We now give a more precise description of the new iteration technique. Let the matrix A be expressed as the matrix sum $A = L + U$, 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 new decoding scheme corresponds to the iterative method

$$\mathbf{x}^{(k+1)} = L\mathbf{x}^{(k+1)} + U\mathbf{x}^{(k)} + \mathbf{b}, \tag{3}$$

or equivalently

$$\mathbf{x}^{(k+1)} = (I - L)^{-1}U\mathbf{x}^{(k)} + (I - L)^{-1}\mathbf{b}. \tag{4}$$

Note that the method proposed in [8] was slightly different as a newly calculated component of a range was not allowed to update a component of the same range. We now give the main result of the paper.

Theorem 6 *If $|s_i| < 1$, for all $1 \leq i \leq n_R$, then the iterative method (4) converges to $\mathbf{x}_T$, the limit vector of method (1), for any starting vector $\mathbf{x}^{(0)}$.*

Proof. Suppose that the iteration scheme (4) is not convergent. Then $\rho((I - L)^{-1}U) \geq 1$. Thus there exists an eigenvalue λ of $(I - L)^{-1}U$ such that $|\lambda| \geq 1$. Let $Q = I - \frac{1}{\lambda}U - L$. Then Q is singular since if $\mathbf{x}$ is an eigenvector of $(I - L)^{-1}U$ associated to the eigenvalue λ, we have $Q\mathbf{x} = \mathbf{0}$. Let us now show that Q is strictly diagonally dominant. Since $|s_i| < 1$ for all $i \in \{1, \ldots, n_R\}$, and due to the structure of A, we have

$$\begin{aligned} 1 &> \sum_{v} |a_{u,v}| \\ &= \sum_{v<u} |l_{u,v}| + \sum_{v \geq u} |u_{u,v}| \\ &\geq \sum_{v<u} |l_{u,v}| + \frac{1}{|\lambda|} \sum_{v \geq u} |u_{u,v}| \\ &= \sum_{v \neq u} |-\frac{1}{\lambda}u_{u,v} - l_{u,v}| + \frac{1}{|\lambda|}|u_{u,u}|. \end{aligned}$$

Thus

$$\begin{aligned} |q_{u,u}| &= |1 - \frac{1}{\lambda} u_{u,u} - l_{u,u}| \\ &\geq 1 - \frac{1}{|\lambda|} |u_{u,u}| \\ &> \sum_{v \neq u} |-\frac{1}{\lambda} u_{u,v} - l_{u,v}| \\ &= \sum_{v \neq u} |q_{u,v}|. \end{aligned}$$

But this is a contradiction since by Theorem 1.8 of [9] a matrix Q cannot be strictly diagonally dominant and singular. The fact that the limit vector is $\mathbf{x}_T$ is trivial from (3). □

3.2 Related methods

We show in this section how well known iterative methods can be efficiently applied in the computation of the fixed point $\mathbf{x}_T = (I - A)^{-1}\mathbf{b}$. Let D be the diagonal matrix obtained from A by setting all off-diagonal entries of A to zero. Then U can be expressed as the matrix sum $U = D + F$, where F is strictly upper triangular. The *point Jacobi* method associated with the solution of the problem $(I - A)\mathbf{x} = \mathbf{b}$ is given by the iterative method

$$\mathbf{x}^{(k+1)} = (I - D)^{-1}(A - D)\mathbf{x}^{(k)} + (I - D)^{-1}\mathbf{b}, \tag{5}$$

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

$$\mathbf{x}^{(k+1)} = (I - D - L)^{-1}F\mathbf{x}^{(k)} + (I - D - L)^{-1}\mathbf{b}. \tag{6}$$

In the case of encodings with constrained scaling factors, both methods converge.

Proposition 7 *If $|s_i| < 1$, for all $1 \leq i \leq n_R$, then both the point Jacobi method (5) and the point Gauss-Seidel method (6) are convergent for any initial vector $\mathbf{x}^{(0)}$.*

Proof. The result follows from Theorem 3.4. in [9] since $I - A$ is clearly strictly diagonally dominant. $\square$

The following proposition gives a relation between the various methods presented so far.

Proposition 8 *If $a_{u,u} = 0$ for all $u = 1, \ldots, N$, then the conventional method (1) reduces to the point Jacobi method and the iterative method (4) reduces to the point Gauss-Seidel method.*

Proof. If $a_{u,u} = 0$ for all $u = 1, \ldots, N$, then D is the null matrix which proves our claim. $\square$

A sufficient condition for the diagonal matrix D to vanish is that for each range R_i, either $s_i = 0$ or $R_i \bigcap D_i = \emptyset$. However, this is unlikely to happen. First, zero scaling factors are an exception. Second, it has been observed that domains overlapping the ranges are the most frequently chosen domains in the encoding [3].

We now compare the computational complexity of the proposed methods. Obviously, our iterative method (4) requires in the general case for each iteration the same amount of arithmetical operations as the conventional method (1). For the other methods we have the following result.

Proposition 9 *Both the point Jacobi method (5) and the point Gauss-Seidel method (6) need at most n_R more arithmetical operations per iteration step than the conventional method (1).*

Proof. We need only show that the result holds for one of the methods (5) or (6), since both have the same complexity. Now let us write (5) in the equivalent form

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

If $a_{u,u} = 0$, then the same number of arithmetical operations is required for the computation of $x_u^{(k+1)}$ by methods (5) and (1). Otherwise, we have for (5) one addition less, but one subtraction and one division more. We conclude the proof by remarking that the subtraction $1 - a_{u,u}$ needs to be computed only once for all $u \in R_i,\ i \in \{1, \ldots, n_R\}$, where R_i is a range. $\square$

Note that even though the two methods (5) and (6) do not require many more arithmetical operations than our proposed method (4), they demand for all $u = 1, \ldots, N$ testing if $a_{u,u} = 0$. This leads to an undesirable additional computational work. For this reason, we will not consider these methods any further in this paper.

3.3 Rate of convergence

In this section we compare the rate of convergence of methods (1) and (4). We need first some preliminary results.

Lemma 10 *The matrix $(I - |L|)^{-1}$ is nonnegative.*

Proof. Since $|L|$ is strictly lower triangular, it is nilpotent. Thus there exists $k = N > 0$ such that $|L|^k = 0$. From the equality

$$(I - |L|)(I + |L| + \cdots + |L|^{k-1}) = I,$$

we obtain $(I - |L|)^{-1} = I + |L| + \cdots + |L|^{k-1} \geq O$. $\square$

Lemma 11 *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, its inverse is nonnegative.*

Proof. If λ is an eigenvalue of $I - |A|$, then $1 - \lambda$ is an eigenvalue of $|A|$. But $\rho(|A|) = \rho(A) < 1$. 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

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

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 $(I - |A|)^{-1} \geq O$. $\square$

We are now ready to prove that if all scaling factors have the same sign, then the iterative method (4) has a greater rate of convergence than the conventional method.

Proposition 12 *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 $M = I - |A| = (I - |L|) - |U|$. From Lemma 10, one can see that these two splittings of M are regular. Furthermore, $|A| \geq |U|$, and by Lemma 11, $M^{-1} \geq O$. The inequality $\rho(|A|) \geq \rho((I - |L|)^{-1}|U|)$ follows then from Corollary 5.6. p. 125 in [11]. We complete the proof by noting that $\rho(A) = \rho(|A|)$ and $\rho((I-L)^{-1}U) \leq \rho((I - |L|)^{-1}|U|)$. The last inequality being due to

$$\begin{aligned} |(I-L)^{-1}U| &\leq |(I-L)^{-1}||U| \\ &= |I + L + \cdots + L^{N-1}||U| \\ &\leq (I + |L| + \cdots + |L|^{N-1})|U| \\ &= (I - |L|)^{-1}|U|. \end{aligned}$$

and Theorem 2.8. in [9]. □

If the scaling factors are not of the same sign, we cannot ensure the inequality in Proposition 12. We argue that this is not a severe limitation for two reasons. First, it has been observed that restricting the scaling factors to be positive does not degrade the rate-distortion performance significantly [12]. Actually, this was the choice of Jacquin in his original work [1], and seems also to be preferred by Iterated Systems, Inc. [13]. Second, we have noticed in our simulations that our decoding method converged faster even if the scaling factors were not all of the same sign (see next section).

As already mentioned, it may happen that the conventional method (1) converges even if some of the scaling factors are larger than one in absolute value. How will method (4) behave in those cases?

Proposition 13 *If the conventional method (1) converges and all scaling factors are of the same sign, then the new method (4) converges also. Moreover, method (4) has a greater rate of convergence than method (1).*

Proof. In the proof of Proposition 12, the assumption that the scaling factors are bounded by one in absolute value, was needed only to ensure that $\rho(|A|) < 1$. Now, method (1) converges if and only if $\rho(A) < 1$. When in addition all scaling factors are of the same sign, we have $\rho(|A|) = \rho(A) < 1$. Hence we can use the proof of Proposition 12 to obtain the inequality $\rho((I-L)^{-1}U) \leq \rho(A)$, which gives the desired result. □

If in Proposition 13 we drop the condition that all the scaling factors are of the same sign, then the convergence of method (4) cannot be guaranteed any more as shown by the following example. Let

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

Then $\rho(A) = \frac{1}{2} < 1$. 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

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

4 Experimental results

The tables presented in this section show the root mean square error between the attractor $\mathbf{x}_T$ and the iterate $\mathbf{x}^{(k)}$ as a function of the number of iteration steps k, starting from an initial black image $\mathbf{x}^{(0)} = (0, \ldots, 0)^T$ for both the conventional decoding and the new method (4). In a practical implementation, since $\mathbf{x}_T$ is not available, one may decide to stop iterating if

$$\frac{\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\|}{\|\mathbf{x}^{(k)}\|} < \epsilon \text{ or } k > k_0,$$

where ϵ is a tolerance bound and k_0 is a maximum iteration step. The first tables provide results for images encoded with Fisher's quadtree code [14]. The scaling factors and the offsets were uniformly quantized with 5 bits and 7 bits, respectively. Domains at each level of the quadtree had 4 times the size of the ranges. They were selected as subsquares of the image whose upper-left corners were positioned on a lattice with a fixed vertical and horizontal spacing equal to d. Pixel shuffling was allowed and three domain classes out of 72 were searched (see [14]). Table 1 shows the results for the

Iteration	Conventional	New method
1	82.82092	61.36793
2	50.46009	26.65044
3	30.19527	10.31304
4	17.47030	3.66579
5	9.80071	1.26984
6	5.39547	0.44142
7	2.93693	0.15110
8	1.58974	0.05089
9	0.85756	0.01709
10	0.46205	0.00574
11	0.24900	0.00193
12	0.13424	0.00065

Table 1: Convergence of the decoding for the two methods measured by the root mean square error between an iterate and the attractor for the 512 $\times$ 512 Bridge image. Only nonnegative scaling factors were used.

512 $\times$ 512 Bridge image. A uniform partitioning with 4096 ranges of size 8 $\times$ 8 was employed. Only nonnegative scaling factors were allowed. The domain pool step size d was equal to 2. Thus, there were 62001 overlapping domains of size 16 $\times$ 16. Note that a domain here was not necessarily a union of ranges. Table 2 shows the results for the 512 $\times$ 512 Lenna image for a three level quadtree segmentation and a root mean square tolerance value of 8. The domain pool step size was equal to 8. Thus, there were 4096 nonoverlapping domains of size 8 $\times$ 8, 3969 overlapping domains of size 16 $\times$ 16 and 3721 overlapping domains of size 32 $\times$ 32. Both positive and negative scaling factors were allowed. Table 3 compares the convergence of the decoding for the two methods for the 512 $\times$ 512 Barbara image. Here we used the code of [15] in the encoding. With this scheme each range is a union of edge connected small square image pieces (see Figure 1). In all cases our new decoding scheme converged faster than the conventional one. The same observation held for all other tested images.

Iteration	Conventional	New method
1	69.93440	55.51971
2	39.89977	22.44756
3	23.04110	8.77134
4	13.03152	3.11604
5	6.96367	1.02256
6	3.62978	0.32492
7	1.87708	0.09921
8	0.95866	0.02951
9	0.48477	0.00867
10	0.24328	0.00253
11	0.12125	0.00074
12	0.06008	0.00021

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

5 Conclusion and future work

We have introduced a fast decoding technique for fractal image compression that needs only half the storage requirements of the conventional decoding method. Convergence of the new method was proved in the most general case of variable size and shape segmentations where searching, unconstrained domains and pixel shuffling were allowed. If all scaling factors are of the same sign, it is shown that the new iterative method has a greater rate of convergence than the standard decoding technique. Experimental results on a large set of test images showed that the decoding was faster even without this restriction.

Other methods have been previously proposed to accelerate the decoding procedure. They are based on a hierarchical interpretation of the fractal code [16, 17]. However, these methods were described for special cases where, particularly, each domain was a union of ranges. This may be a serious restriction since unconstrained domain pools are known to provide the best rate-distortion performances [14, 15, 18, 19]. On the other hand, the method in [17] prescribes the usual iterations at, however, a lower resolution, and may

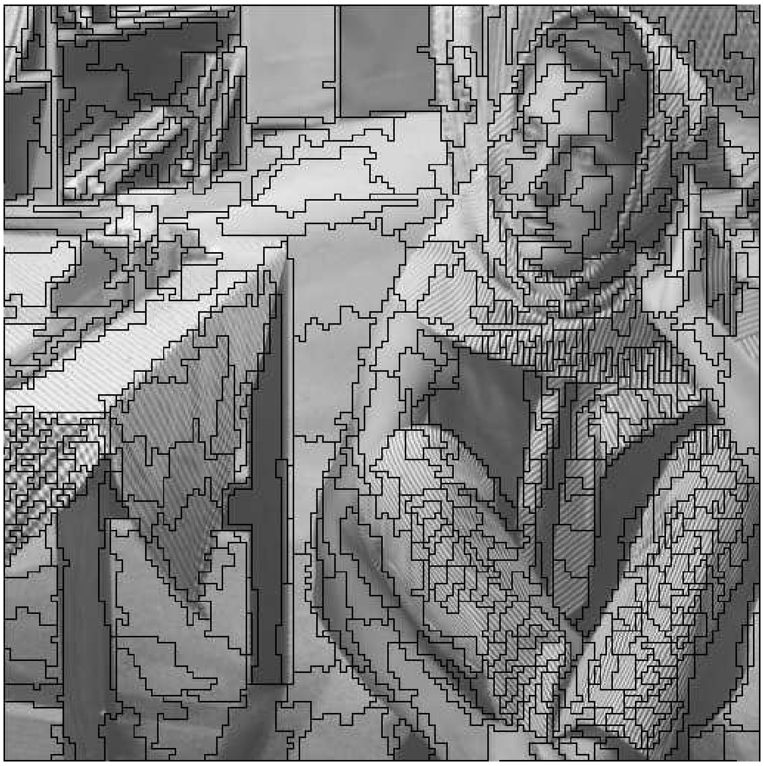

Figure 1: Partitioning of the 512 $\times$ 512 Barbara image in 600 ranges of various sizes and shapes.

be thus combined with our technique in a straightforward manner yielding a further improvement.

Whereas the order in which the pixels are decoded has no effect on the resulting image when the conventional method is used, it is clear that this will not be the case with our method. In [20] we propose a technique where the order is based on the frequency in the fractal code of the range to which the pixel belongs. By frequency of a range R in the fractal code we mean the number of times there existed $i \in \{1, \ldots, n_R\}$ such that $R \bigcap D_i \neq \emptyset$. Here $D_1, \ldots, D_{n_R}$ are the domains retained for the decoding. The expected gain is twofold. First, this strategy will increase the number of pixels that will be used in their updated form yielding, hopefully, a faster convergence. Second, as a consequence of the sorting of the range frequencies, the decoder can identify those ranges that are not present in the fractal code. These ranges,

Iteration	Conventional	New method
1	63.99485	44.34036
2	33.11056	15.05124
3	18.25223	4.46381
4	8.47017	1.22855
5	3.56080	0.32108
6	1.37339	0.07997
7	0.50095	0.01755
8	0.16188	0.00393
9	0.04962	0.00083
10	0.01662	0.00018
11	0.00571	0.00004
12	0.00201	0.00001

Table 3: Convergence of the decoding for the two methods measured by the root mean square error between an iterate and the attractor for the 512 $\times$ 512 Barbara image. Both positive and negative scaling factors were allowed.

as pointed out in [10, 21] in a different context, need only be decoded at the last iteration. Note that for arbitrary domain pools where a given domain is not necessarily a union of ranges, it may happen that a range gets a big frequency although only a small part of it is actually present in the fractal code. To deal with this, one may consider fractional range frequencies. For example, if only m pixels of the range of size n are covered by the domain D_i, then the frequency of this range will be increased by $\frac{m}{n}$. Another approach would be to compute the frequency in the fractal code of each pixel and to decode pixelwise. Table 4 compares the convergence of the decoding for the 512 $\times$ 512 Lenna image for the following three schemes.

- Scheme 1: The conventional decoding without pixel update.
- Scheme 2: The new decoding technique where the ordering is as in the encoding.
- Scheme 3: The new decoding technique with a range frequency based ordering.

A uniform partitioning in square ranges of size 8×8 was used. The domain pool consisted of square blocks of size 16×16 whose upper-left corners were positioned on a lattice with a vertical and horizontal spacing of 8 pixels. Figure 2 provides statistics on the frequency of each range in the fractal code. Even though the decoding with the frequency based ordering did not converge much faster than Scheme 2, it still may be preferred since at each step the iterates provided a better approximation. Similar results were obtained for all other tested images and domain pools.

Iteration	Scheme 1	Scheme 2	Scheme 3
1	73.1453	59.6675	46.8337
2	42.2518	25.1607	17.1060
3	24.6055	9.5244	4.7620
4	14.0406	2.3372	0.9230
5	4.5585	0.3805	0.1472
6	1.3235	0.0740	0.0287
7	0.3726	0.0168	0.0070
8	0.1080	0.0037	0.0020

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

Acknowledgment. I would like to thank Matthias Ruhl and Dietmar Saupe for making the C code of [15] available.

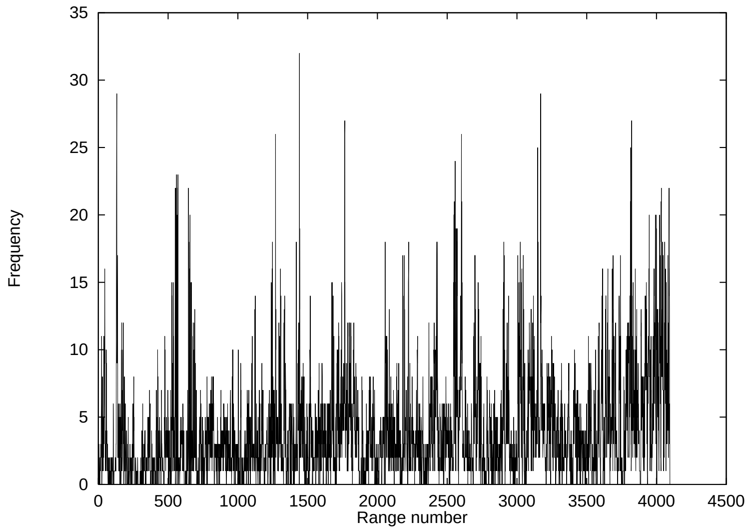

Figure 2: Range frequency in the fractal code.

References

[1] Jacquin, A. E., *Image coding based on a fractal theory of iterated contractive image transformations,* IEEE Trans. Image Processing 1 (1992) 18–30.

[2] Barnsley, M., Hurd, L., *Fractal Image Compression,* AK Peters, Wellesley, 1993.

[3] Fisher, Y., *Fractal Image Compression — Theory and Application,* Springer-Verlag, New York, 1994.

[4] Barnsley, M., *Fractals Everywhere,* Academic Press, San Diego, 1988.

[5] Bedford, T., Dekking, F. M., Keane, M. S., *Fractal image coding techniques and contraction operators,* Nieuw Arch. Wisk. (4) 10,3 (1992) 185–218.

[6] Kaouri, H., *Fractal coding of still images,* in: *IEE 6th International Conference on Digital Processing of Signals in Communications,* pp. 235–239, 1991.

[7] Nelson, M., Gailly, J.-L., *The Data Compression Book*, M & T books, New York, 1995.

[8] Hamzaoui, R., *Decoding algorithm for fractal image compression,* Electronics Letters 32,14 (1996) 1273–1274.

[9] Varga, R. S., *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, N. J., 1962.

[10] Hürtgen, B., Simon, S. F., *On the problem of convergence in fractal coding schemes,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Vol. 3, pp. 103–106, Austin, Texas, 1994.

[11] Young, D. M., *Iterative solution of large linear systems,* Academic Press, New York, 1971.

[12] Saupe, D., *The futility of square isometries in fractal image compression,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Vol. 1, Lausanne, Sept. 1996.

[13] Geronimo, J., Lu, N., *Lectures on fractal techniques in image compression,* in: *Image Tech Conference,* Atlanta, Georgia, March 1996.

[14] Fisher, Y., *Fractal image compression with quadtrees,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[15] Saupe, D., Ruhl, M., *Evolutionary fractal image compression,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Vol. 1, Lausanne, Sept. 1996.

[16] Øien, G. E., Lepsøy, S., *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.

[17] 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.

[18] Fisher, Y., Menlove, S., *Fractal encoding with HV partitions,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[19] Barthel, K. U., Voyé, T., Noll, P., *Improved fractal image coding,* in: *Proceedings of the International Picture Coding Symposium PCS'93,* Lausanne, March 1993.

[20] R. Hamzaoui, *Ordered decoding algorithm for fractal image compression*, submitted 1997.

[21] Domaszewicz, J., Vaishampayan, V. A., *Graph-theoretical analysis of the fractal transform,* in: *Proceedings of ICASSP-1995 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 4, D etroit, 1995.