Technical Report

TR-2007-010

Inpainting Through Fractal Image Encoding

by

Ying Wai (Daniel) Fan, James G. Nagy

Mathematics and Computer Science

EMORY UNIVERSITY

INPAINTING THROUGH FRACTAL IMAGE ENCODING

YING WAI (DANIEL) FAN* AND JAMES G. NAGY†

Abstract. Computational techniques that attempt to reconstruct missing pixels in an image are broadly referred to as *digital image inpainting*. Approaches have been developed based on continuation of isophotes, curvature-driven diffusion, as well as approaches that require decomposing the image into, for example, cartoon (smooth) and texture parts. The purpose of this paper is to propose a novel approach to inpainting based on fractal image encoding. The basic idea is to exploit self similarity in the image using fractal image encoding schemes. A modification of the encoding algorithm is needed to take into account missing data. Then, by applying the standard decoding algorithm, the built in self similarity property of the fractal code reconstructs the missing pixels. Several examples are presented to illustrate the effectiveness of the proposed inpainting method.

Key words. fractal image encoding, fractal image compression, inpainting, fixed point iteration

AMS Subject Classifications: 65F20, 65F30

1. Introduction. When a painting has been damaged by weathering, aging, or vandalism, artists are often employed to restore the painting. The process they use to fill in the damaged regions is called *inpainting*. This is clearly a subjective process; the artist must determine how to "best" fill in the missing information, and different artists may produce different restorations. Determining what is the "correct" restoration is not possible without having access to the original painting.

Recently, scientists have extended the concept of inpainting to the digital age. In 2000, Bertalmio, Sapiro, Caselles and Ballester [5] introduced the problem of filling in missing pixels of a digital image through mathematical and computational techniques, and called the process *digital image inpainting*. Since this original work, many different approaches have been proposed for digital image inpainting. The original approach by Bertalmio et al. is based on continuation of isophotes into the inpainting regions. Chan and Shen [7] apply curvature-driven diffusion in the inpainting region, and also do additional denoising on the undamaged regions using total variation.

As with the classical problem of inpainting, determining what is the "correct" or "best" restoration is not possible without having access to the original painting. Thus, it seems especially important to have a variety of fundamentally different techniques for the digital image inpainting problem, providing the user with different interpretations of how to fill in the missing data. The two approaches described in the previous paragraph produce good results on smooth images, but may have difficulties when the image has a lot of texture. Approaches that can also deal with the texture of the image have been developed. Bertalmio, Vese, Sapiro and Osher [6] decompose an image into cartoon (smooth) part and texture part, then recover the inpainting region in both parts with variational techniques and texture synthesis, respectively, and then recombine both parts. Elad, Starck, Querre and Donoho [10] treat an image as a sparse combination of many different components. Applying morphological component analysis with a predetermined dictionary of components, they compute the proportion of each component in the image and then reconstruct the whole image from these components.

*Department of Mathematics and Computer Science, Emory University. Email: yfan@mathcs.emory.edu.

†Department of Mathematics and Computer Science, Emory University. Email: nagy@mathcs.emory.edu. Research supported in part by the NSF under grant DMS-05-11454 and by an Emory University Research Committee grant.

In this paper we propose a novel, fundamentally different approach to digital image inpainting that is based on *fractal image encoding*. The basic idea is to produce a *fractal code* of the damaged image using known (undamaged) pixels, and then to use a decoding algorithm to reconstruct the image. During the decoding process, the damaged pixels are filled in. Since the fractal encoding/decoding process is based on exploiting self similarity in the image, the inpainting information is extracted from the known pixels to determine appropriate values for the unknown pixels. This paper is outlined as follows. In section 2 we provide a brief introduction to the method of fractal image encoding. In section 3 we describe our approach to inpainting through a modified fractal image encoding scheme. Experimental results are presented in section 4, and concluding remarks are given in section 5.

2. Fractal Image Encoding. Fractal image encoding, more widely known as fractal image compression, is a lossy image compression scheme that represents an image by a contractive mapping. In this section we introduce the basic idea behind fractal image encoding, and refer the interested reader to [2], [18] and [3] for more complete presentations.

2.1. Representing an image by a contractive mapping. A function $f : \mathbb{R}^N \to \mathbb{R}^N$ is a contractive mapping if there exist $0 \le c < 1$ such that

$$\|f(\mathbf{x}) - f(\mathbf{y})\| \le c\|\mathbf{x} - \mathbf{y}\|, \tag{2.1}$$

for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$. By the Banach Fixed Point Theorem, if f is a contractive mapping, then the fixed point iteration $\mathbf{x}_{k+1} = f(\mathbf{x}_k)$ converges to the unique fixed point $\mathbf{x}_* = f(\mathbf{x}_*)$ for any initial guess $\mathbf{x}_0$.

Suppose we are given an $n \times n$ image X and let[1] $\mathbf{x} = \mathtt{vec}(X) \in \mathbb{R}^{n^2} = \mathbb{R}^N$. Suppose further that we are able to find a contractive mapping f with the property that the fixed point, $\mathbf{x}_* = f(\mathbf{x}_*)$, is a good approximation of $\mathbf{x}$ (ideally, $\mathbf{x}_* = \mathbf{x}$); see Fig. 2.1. Then the contractive mapping, f, can be used to represent the image $\mathbf{x}$. That is, we need only store the parameters that define f, and $\mathbf{x}$ can be discarded. Note that if the storage of parameters defining f is significantly less than the storage of pixels in the image, $\mathbf{x}$, then the encoding acts as a compression mechanism. To reconstruct (i.e., decode) the image $\mathbf{x}$ from f, we simply use the fixed point iteration $\mathbf{x}_{k+1} = f(\mathbf{x}_k)$ for an arbitrary initial guess $\mathbf{x}_0$.

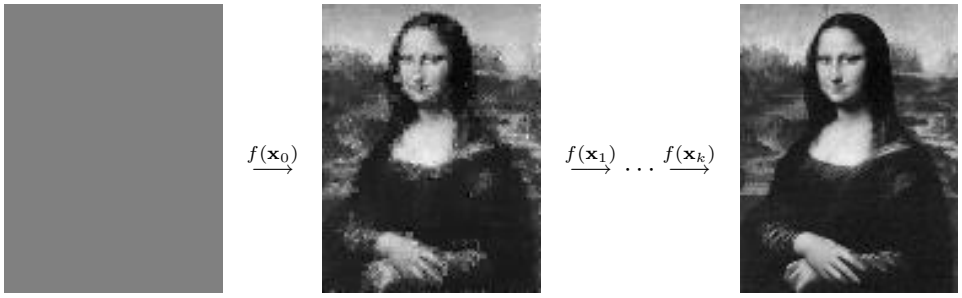

FIG. 2.1. *Illustration of convergence of a contractive map, with an arbitrary initial guess* $\mathbf{x}_0$.

[1]The `vec` operator transforms an $n \times n$ image X into a vector of length $N = n^2$ by stacking columns of X. Other orderings (e.g., lexicographic) can be used.

There are two theorems that govern the approximation of $\mathbf{x}_*$ to $\mathbf{x}$.

THEOREM 2.1 (Collage Theorem [4]). *If f is a contractive mapping with contraction constant c and fixed point $\mathbf{x}_*$, then for any $\mathbf{x}$,*

$$\|\mathbf{x}_* - \mathbf{x}\| \leq \frac{1}{1-c}\|f(\mathbf{x}) - \mathbf{x}\| \tag{2.2}$$

THEOREM 2.2 (Anti-Collage Theorem [20]). *Assume the conditions of Theorem 2.1, then for any $\mathbf{x}$,*

$$\|\mathbf{x}_* - \mathbf{x}\| \geq \frac{1}{1+c}\|f(\mathbf{x}) - \mathbf{x}\| \tag{2.3}$$

By these two theorems, we can find the contractive mapping f for a given image $\mathbf{x}$ such that

$$\mathbf{x}_* \approx \mathbf{x}, \tag{2.4}$$

by making sure that

$$f(\mathbf{x}) \approx \mathbf{x}. \tag{2.5}$$

All fractal image encoding algorithms achieve (2.4) by ensuring (2.5), as the latter is more tractable.

Unfortunately finding a contractive mapping f that satisfies the above properties, for an arbitrary image $\mathbf{x}$, is nontrivial. According to [17], finding the optimal f among all possible contractive mappings is an NP-hard problem. In order to make the search for f computationally tractable, it is necessary to assume f has a certain canonical form. The typical approach, proposed by Jacquin [15], consists of approximating the image X using a set of affine transformations.

B_Γ^J	Get block of size $2^J \times 2^J$ from location Γ
$(B_\Gamma^J)^*$	Put block of size $2^J \times 2^J$ to location Γ
L_P	Apply isometry P to a block
A^k	Average and subsample a block k times
1	$2^N \times 2^N$ square matrix of all 1's
$\mathcal{D}$	Domain Pool
$\mathcal{R}$	Set of range blocks
g_Γ	Gain coefficient for block/subtree Γ
h_Γ	DC offset for block Γ

TABLE 2.1
Basic Notation

2.2. Jacquin's fractal image encoding scheme. The basic idea behind the encoding scheme proposed by Jacquin is to discover self similarity in the image by examining and comparing (via affine transformations) various subblocks in the image. For example, a region in the image containing a vertical line (and nothing else) is very similar to a region that contains only a (possibly different length) horizontal line. To obtain a mathematical description of the encoding process, we need to introduce some notation; in this paper we use the same notation as Davis [9] which is summarized in Table 2.1.

We first need operators that "grab" and "put" subblocks of an image. Let $B^J_{K,L}$ be the get-block operator, which extracts a subblock of size $2^J \times 2^J$ with lower left corner located at the (K, L) pixel entry in the image. Let $(B^J_{K,L})^*$ be the put-block operator, which inserts a subblock of size $2^J \times 2^J$ into a $2^N \times 2^N$ image of all zeros, with the lower left corner of the subblock positioned at the (K, L) pixel entry in the image. For convenience, we use Γ as an abbreviation of (K, L).

Suppose a $2^N \times 2^N$ image X is partitioned it into non-overlapping *range blocks* of size $2^R \times 2^R$.Thus the collection of range blocks are

$$\{B^R_\Gamma X | \Gamma \in \mathcal{R}\} \text{ where } \mathcal{R} = \{(2^R m, 2^R n) | m, n \in \mathbb{Z} \text{ and } 1 \le m, n < 2^{N-R}\}.$$

We extract subblocks larger than the range blocks from X to form a pool of *domain blocks*. Thus the collection of domain blocks is

$$\{B^D_\Gamma X | \Gamma \in \mathcal{D}\} \text{ where } \mathcal{D} = \{(m, n) | m, n \in \mathbb{Z} \text{ and } 1 \le m, n < 2^N - 2^D\}.$$

Usually, D is taken to be $R + 1$, so each domain block is 4 times bigger than a range block. Note that the domain blocks are allowed to be overlapping.

Let A be a downsampling operator which produces a $2^{(J-1)} \times 2^{(J-1)}$ block from a $2^J \times 2^J$ block. Specifically we set

$$\begin{aligned}(AB^J_\Gamma X)(k, l) =& \frac{1}{4}[(B^J_\Gamma X)(2(k-1)+1), 2(l-1)+1) + (B^J_\Gamma X)(2k, 2(l-1)+1) \\ &+ (B^J_\Gamma X)(2(k-1)+1, 2l) + (B^J_\Gamma X)(2k, 2l)].\end{aligned}$$

Let L_k, $1 \le k \le 8$ be an operator that maps a block to one of its eight isometries obtained from rotations (multiples of 90 degrees) and reflections (horizontal, vertical, diagonal and anti-diagonal) of the block.

Having defined this notation, we are ready to discuss fractal encoding. The codebook $\mathcal{C}$ of the encoding consists of isometries of domain blocks shrunken to the size of range blocks:

$$\mathcal{C} = \{L_k A^{D-R} B^D_\Gamma X | \Gamma \in \mathcal{D}, 1 \le k \le 8\}.$$

The contrast of each codeword is adjusted by a gain coefficient g and the codeword is then added by a multiple of the block $\mathbf{1}$ of all 1's. For each range block $B^R_\Gamma X$, we have

$$B^R_\Gamma X \approx g_\Gamma L_{P(\Gamma)} A^{D-R} B^D_{\Pi(\Gamma)} X + h_\Gamma B^R_\Gamma \mathbf{1}, \tag{2.6}$$

where $P : \mathcal{R} \to \{1, \ldots, 8\}$ assigns each range block an isometry index and $\Pi : \mathcal{R} \to \mathcal{D}$ assigns a domain block to each range block. The parameters g_Γ, h_Γ, $\Pi(\Gamma)$ and $P(\Gamma)$ are chosen such that the right hand side of (2.6) is a good approximation to the left hand side. We also constrain $|g_\Gamma| < 1$ for convergence in the decoding process. Then the collection of $\{(g_\Gamma, h_\Gamma, \Pi(\Gamma), P(\Gamma)) | \Gamma \in \mathcal{R}\}$ is the fractal code of X.

Now, we consider the decoding process. From (2.6), we have

$$\begin{aligned} X &= \sum_{\Gamma\in\mathcal{R}} (B_\Gamma^R)^*(B_\Gamma^R)X \\ &\approx \sum_{\Gamma\in\mathcal{R}} (B_\Gamma^R)^*[g_\Gamma L_{P(\Gamma)} A^{D-R} B_{\Pi(\Gamma)}^D X + h_\Gamma B_\Gamma^R \mathbf{1}] \\ &= \sum_{\Gamma\in\mathcal{R}} g_\Gamma (B_\Gamma^R)^* L_{P(\Gamma)} A^{D-R} B_{\Pi(\Gamma)}^D X + h_\Gamma (B_\Gamma^R)^* B_\Gamma^R \mathbf{1} \\ &= GX + H \\ &= F(X), \end{aligned} \tag{2.7}$$

where G is a linear operator, H is a constant image and F maps images to images. We can guarantee F to be contractive by constraining $|g_\Gamma| < 1$ for all $\Gamma \in \mathcal{R}$. An example of $(B_\Gamma^R)^*[g_\Gamma L_{P(\Gamma)} A^{D-R} B_{\Pi(\Gamma)}^D X + h_\Gamma B_\Gamma^R \mathbf{1}]$ is shown in Fig. 2.2.

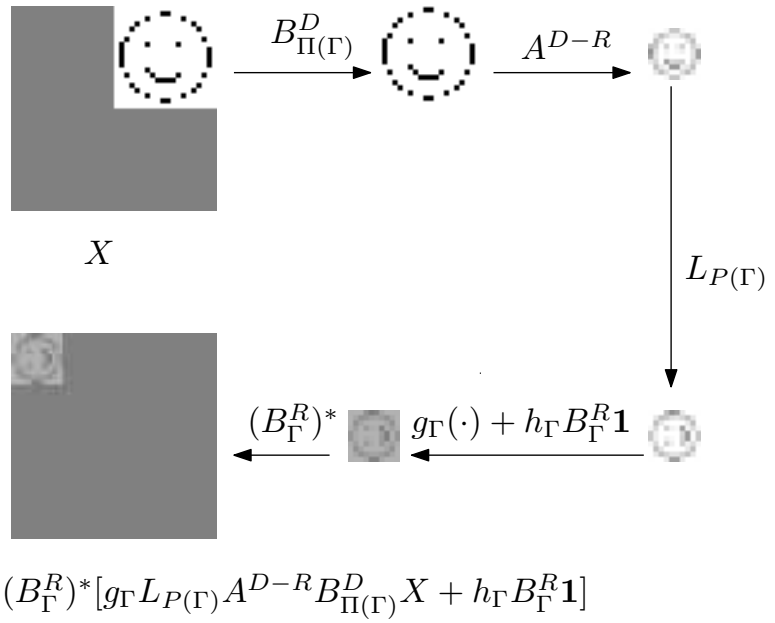

FIG. 2.2. *Illustration of the operations that compute* $Y = (B_\Gamma^R)^*[g_\Gamma L_{P(\Gamma)} A^{D-R} B_{\Pi(\Gamma)}^D X + h_\Gamma B_\Gamma^R \mathbf{1}]$

Comparing (2.7) and (2.5) and by Theorems 2.1 and 2.2, we know that the fixed point X_* of F is a good approximation to X. So to decode the fractal code to get back an approximation to X, we simply start from an arbitrary image X_0, then repeatedly apply F to it to get an image sequence $\{X_k\}$, such that $X_k = F(X_{k-1})$. We stop the iteration when convergence is attained. The last iterate $X_K \approx X_*$ is what we call the decoded image. In practice, 10 iterations are needed for convergence, and on a standard laptop computer it takes less than a second to complete the decoding process. The encoding takes longer, but usually finishes within 10 seconds.

2.3. Remarks. We remark that instead of using the domain pool of all possible domain block, we can use the disjoint domain pool ($\mathcal{D} = \{(2^D m, 2^D n) | 0 \le m, n \le 2^{N-D}\}$) or half-overlapping domain pool ($\mathcal{D} = \{(2^{D-1} m, 2^{D-1} n) | 0 \le m, n \le$

$2^{N-D+1}\}$). In these cases, we reduce the computational load at the expense of a possible larger encoding error. More sophisticated, adaptive partitioning schemes, such as quadtrees, could be used [11].

3. Modified encoding schemes for inpainting. During the fractal image encoding, an image is represented by a contractive mapping that needs less storage than the original image. This means that there is redundant information in the image. If we can find the fractal code of an image without using all pixels, say by skipping unknown pixels, then we can solve the inpainting problem by decoding the fractal code.

3.1. Encoding only on known pixels. Let $\bar{X}$ be an image with unknown pixels. Let $\bar{A}$ be a downsampling operator that produces a $2^{(J-1)} \times 2^{(J-1)}$ block from a $2^J \times 2^J$ block and takes into account missing pixels. Specifically, it averages only the known pixels in every 2×2 block to produce a new pixel and if all 4 pixels are unknown then the new pixel is marked as unknown.

The codebook $\mathcal{C}$ is now

$$\mathcal{C} = \{L_k \bar{A}^{D-R} B_\Gamma^D \bar{X} | \Gamma \in \mathcal{D}, 1 \le k \le 8\}.$$

For each range block $B_\Gamma^R \bar{X}$, we have

$$B_\Gamma^R \bar{X} \approx g_\Gamma L_{P(\Gamma)} \bar{A}^{D-R} B_{\Pi(\Gamma)}^D \bar{X} + h_\Gamma B_\Gamma^R \mathbf{1}. \tag{3.1}$$

The parameters g_Γ, h_Γ, $\Pi(\Gamma)$ and $P(\Gamma)$ are chosen such that the right hand side of (3.1) is a good approximation to the left hand side. Since there may be unknown pixels on both sides of (3.1), we only consider in the approximation those pixel positions at which the pixels are known on both sides. Again, we also constrain $|g_\Gamma| < 1$ for convergence in the decoding process. Then the collection of $\{(g_\Gamma, h_\Gamma, \Pi(\Gamma), P(\Gamma)) | \Gamma \in \mathcal{R}\}$ is the fractal code of $\bar{X}$.

3.2. Decoding only on unknown pixels. The decoding can be done as in Section 2.2. From the fractal code $\{(g_\Gamma, h_\Gamma, \Pi(\Gamma), P(\Gamma)) | \Gamma \in \mathcal{R}\}$, we construct the contractive mapping F, then we iterate from an arbitrary image X_0 to get the fixed point X_* of F. The fixed point X_* is a good approximation to $\bar{X}$ with no unknown pixels. This is how we inpaint $\bar{X}$.

It is noted in [13] that the decoding of the fractal code of a noisy image produces a denoised image. Thus, if the original damaged image $\bar{X}$ is noisy, the decoding in Section 2.2 would produce both an inpainting and a denoising effect to $\bar{X}$.

Here, we propose another kind of decoding scheme specific for the inpainting problem when X is not noisy. Instead of starting from an arbitrary image, we take X_0 to be the original damaged image $\bar{X}$. As some pixels are unknown in X_0, we need to use $\bar{A}$ instead of A in the iteration. Since some pixels are already known, we keep these unchanged for each iteration and update those pixels that are originally unknown. That is

$$\hat{X}_{k+1} = \bar{F}(X_k) = \sum_{\Gamma \in \mathcal{R}} g_\Gamma (B_\Gamma^R)^* L_{P(\Gamma)} \bar{A}^{D-R} B_{\Pi(\Gamma)}^D X_k + h_\Gamma (B_\Gamma^R)^* B_\Gamma^R \mathbf{1}$$

$$X_{k+1}(i,j) = \begin{cases} \hat{X}_{k+1}(i,j) & \text{if pixel at } (i,j) \text{ is unknown in } \bar{X} \\ X_k(i,j) & \text{otherwise} \end{cases}.$$

This new decoding scheme should give better inpainting results than the original scheme as it retains all known pixels in $\bar{X}$.

4. Experimental results. There are many different implementations for fractal coding [16, 8, 12, 14, 21]. We use the code written by Mario Polvere [16]. Polvere uses several techniques to speed up the fractal image encoding, such as classification of the domain blocks. He uses quadtree partitioning to obtain the domain block pool. We modify his code to cater to unknown pixels as discussed in Section 3.

We test our inpainting scheme on 512×512 Barbara and Goldhill images (top row of Fig. 4.1). We discard every 20th pixel in the horizontal and vertical directions to get the second row of Fig. 4.1. We then fractally encode the two "damaged" images and decode only on the unknown pixels to get the third row of Fig. 4.1. The resulting images are visually indistinguishable from the original images. The root-mean-square (RMS) errors and the peak signal-to-noise ratios (PSNR) between the original and the inpainted images are shown in Table 4.1. The PSNR between images X and Y is defined as

$$PSNR = 10 \log_{10} \frac{512 \cdot 512 \cdot 255^2}{\sum_{i,j=1}^{512} (X_{i,j} - Y_{i,j})^2}.$$

	Barbara	Goldhill
RMS error	6.04	4.27
PSNR	32.5	35.5

TABLE 4.1

RMS errors and PSNR between the original and the inpainted images with "grid" inpainting region

In the second experiment, we simulate scratches on the images (second row of Fig. 4.2) and apply our inpainting methods on them. The resulting images are shown in the third row of Fig. 4.2 and the RMS errors and PSNR's are shown in Table 4.2.

	Barbara	Goldhill
RMS error	4.15	2.56
PSNR	35.8	40.0

TABLE 4.2

RMS errors and PSNR between the original and the inpainted images with "scratch" inpainting region

5. Comparison with Other Inpainting Methods. Our fractal imaging encoding approach is similar to the variational methods in that it exploits the inherent physical properties of the image; it is similar to the component analysis method in that a reconstruction of the image is needed. The advantage over the variational methods is that we don't need a complicated physical model of the image and the advantage over the component analysis method is that we do not need a prior dictionary of components. Moreover, once the fractal code is obtained, the image is also compressed and the decoding can be deterred until it is needed, say after it is sent over the network to the end-user. Also, fractal coded images are "scaleless" as they can be zoomed in greatly without jagged edges.

6. Potential Future Research. In the SIAM Conference on Imaging Science 2006, Simon K. Alexander and Edward R. Vrscay suggest associating more than one domain block with each range block to get stronger denoising effect during the

decoding. Similarly, our fractal inpainting approach can be extended to use more than one domain blocks for each range block to get better inpainting effect.

There has been interest in relating fractal image encoding with a fractal transform [1, 9, 19]. We believe a similar inpainting approach exists in the wavelet domain. In particular, we might consider the wavelet interpretation of an image to develop further modifications of our inpainting method.

7. Acknowledgments. Most ideas in this research were inspired by lectures at the 2006 SIAM Conference on Imaging Science, especially the mini-symposium chaired by Michael Barnsley of the Australian National University. In subsequent email correspondence, Professor Barnsley has pointed us to valuable resources in consolidating the ideas of this research.

REFERENCES

[1] S. Alexander, D. of Applied Mathematics, and U. of Waterloo, *Two-and Three-dimensional Coding Schemes for Wavelet and Fractal-wavelet Image Compression*, PhD thesis, University of Waterloo, 2001.
[2] L. F. Anson, *Fractal image compression*, BYTE, (1993), pp. 195–202.
[3] M. Barnsley and L. Barnsley, *Fractal Transformations*, a collection of articles about fractals, companion to the documentary film, The Colours of Infinity (GordonFilms 1995), (2003).
[4] M. Barnsley, V. Ervin, D. Hardin, and J. Lancaster, *Solution of an inverse problem for fractals and other sets*, Proceedings of the National Academy of Sciences, 83 (1986), pp. 1975–1977.
[5] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, *Image inpainting*, in Siggraph 2000, Computer Graphics Proceedings, K. Akeley, ed., ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000, pp. 417–424.
[6] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, *Simultaneous structure and texture image inpainting*, IEEE Transactions on Image Processing, 12 (2003), pp. 882–889.
[7] T. Chan and J. Shen, *Nontexture inpainting by curvature driven diffusions (CDD)*, J. Visual Comm. Image Rep, 12 (2001), pp. 436–449.
[8] K.-L. Chung and C.-H. Hsu, *Novel prediction- and subblock-based algorithm for fractal image compression*, Chaos, Solitons and Fractals, 29 (2006), pp. 215–222.
[9] G. Davis, *A wavelet-based analysis of fractal image compression*, IEEE Transactions on Image Processing, 7 (1998), pp. 141–154.
[10] M. Elad, J.-L. Starck, P. Querre, and D. Donoho, *Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA)*, Applied and Computational Harmonic Analysis, 19 (2005), pp. 340–358.
[11] Y. Fisher, *Fractal Image Compression*, Springer-Verlag, 1994, ch. 3 Fractal Image Compression with Quadtrees, pp. 55–77.
[12] S. Furao and O. Hasegawa, *A fast no search fractal image coding method*, Signal Processing: Image Communication, 19 (2004), pp. 393–404.
[13] M. Ghazel, G. H. Freeman, and E. R. Vrscay, *Fractal image denoising*, IEEE Transactions on Image Processing, 12 (2003), pp. 1560–1578.
[14] C. He, X. Xu, and J. Yang, *Fast fractal image encoding using one-norm of normalised block*, Chaos, Solitons and Fractals, 27 (2006), pp. 1178–1186.
[15] A. Jacquin, *Image coding based on a fractal theory of iterated contractive image transformations*, Image Processing, IEEE Transactions on, 1 (1992), pp. 18–30.
[16] M. Polvere and M. Nappi, *Speed-up in fractal image coding: comparison of methods*, IEEE Transactions on Image Processing, 9 (2000), pp. 1002–1009.
[17] M. Ruhl and H. Hartenstein, *Optimal fractal coding is np-hard*, in Proceedings of the IEEE Data Compression Conference, 1997.
[18] E. Vrscay, *A Hitchhiker's Guide to*, Fractal-Based" Function Approximation and Image Compression", Department of Applied Mathematics, University of Waterloo, Ontario, Canada, (1995), pp. 1–20.
[19] ———, *From Fractal Image Compression to Fractal-based Methods in Mathematics*, IMA Volumes in Mathematics and its Applications, 132 (2002), pp. 65–106.
[20] E. R. Vrscay and D. Saupe, *Can one break the "collage barrier" in fractal image coding*, Springer-Verlag, London, 1999.

[21] C. Zhou, K. Meng, and Z. Qiu, *A fast fractal image compression algorithm based on average-variance function*, IEICE Transactions on Information and Systems, E89-D (2006), pp. 1303–1308.

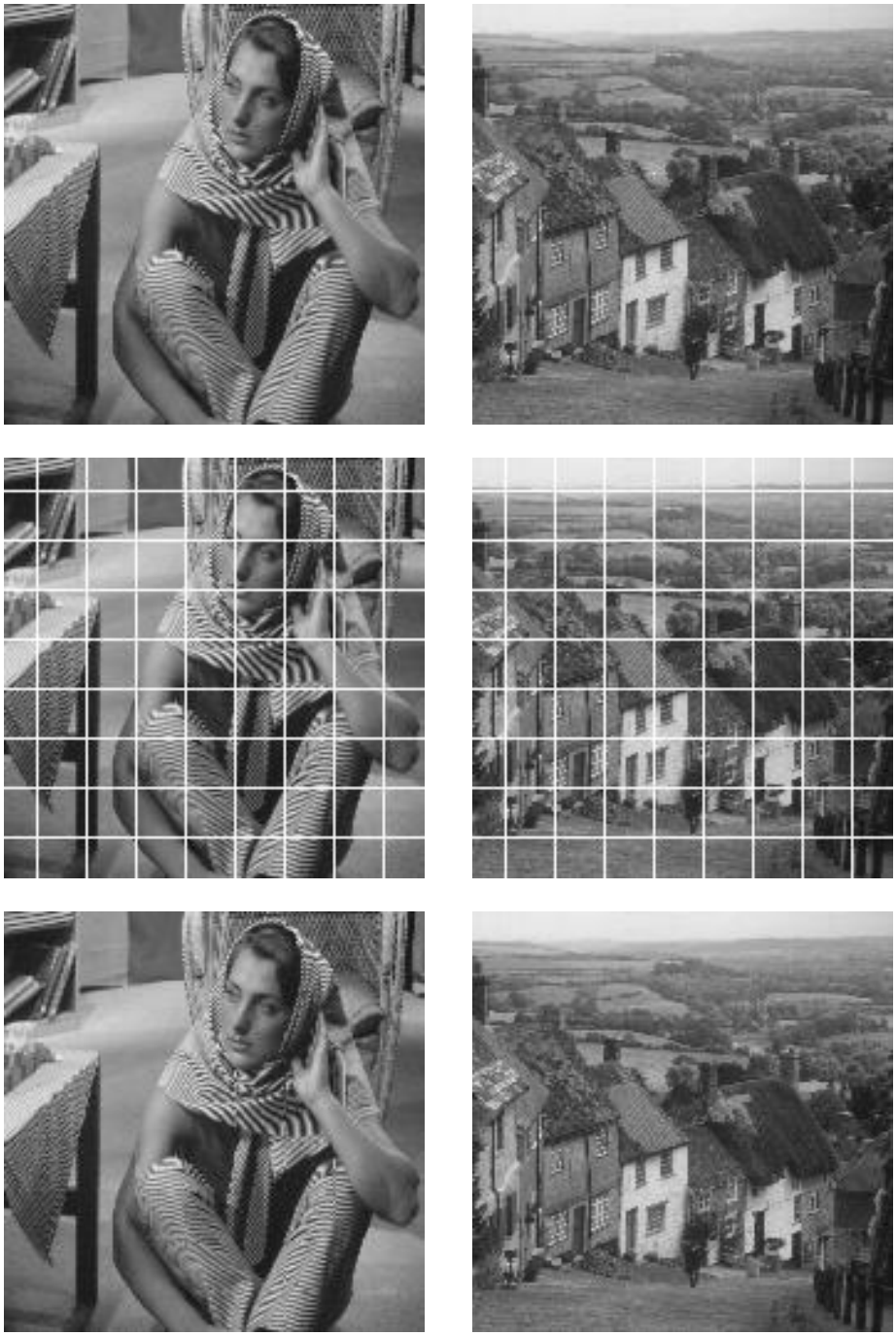

FIG. 4.1. *Inpainting results for Barbara and Goldhill with "grid" inpainting region. From top to bottom: the original images, images with pixels removed, the inpainted images.*

FIG. 4.2. *Inpainting results for Barbara and Goldhill with "scratch" inpainting region. From top to bottom: the original images, images with pixels removed, the inpainted images.*