THE HONG KONG
POLYTECHNIC UNIVERSITY
香港理工大學

電子及資訊工程學系

DEPARTMENT
OF
ELECTRONIC AND INFORMATION
ENGINEERING

Fractal-based Techniques and Their Applications

A thesis submitted in partial fulfillment of the requirements for
the Degree of Doctor of Philosophy

Student Name:	LAI Cheung-Ming
Supervisor Name:	Dr. Kenneth K. M. Lam
Co-supervisor Name:	Prof. Siu Wan Chi
Date:	March 2005

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____(Signed)

LAI Cheung-Ming (Name of student)

Abstract

The aim of this research is to develop efficient algorithms for fractal image coding, which can be applied in digital image compression, image magnification and image denoising. Fractal image coding can provide a highly reconstructed image quality with a high compression ratio, is independent of resolution, and has a fast decoding process. The problem with fractal coding is its high computational complexity in the encoding process. Most of the encoding time is spent on finding the best-matched domain block from a large domain pool to represent an input range block with respect to contrast and intensity offset, as well as the isometric transformations. The objectives of this research are to investigate and develop efficient techniques for fractal image coding, fractal-based image magnification and denoising.

In this thesis, four efficient fractal image coding algorithms have been proposed. The first algorithm is based on new feature vectors and the property of zero contrast. The proposed feature vectors can provide a better representation of image blocks, and thus result in a more efficient search of the domain block using the *k*-d tree scheme. The second algorithm is an efficient windowing scheme for fractal image coding based on the local variances method. In this method, windows covering those domain blocks whose variances are higher than that of the range block are considered according to a mathematical model. The exhaustive search algorithm can obtain the optimal result by searching all the blocks within the domain pool, but this process requires a high computational cost, which limits its practical application. A single kick-out condition is proposed which can avoid a large number of range-domain block matches when finding

the best-matched domain block. An efficient method for zero contrast prediction is also proposed, which can determine whether the contrast factor for a domain block is zero or not, and compute the corresponding difference between the range block and the transformed domain block efficiently and exactly. The fourth method is another fast full search fractal image-coding algorithm, which uses the angle between an input range block and a reference domain block to determine a tighter decision boundary for eliminating the searching space in the domain pool. The encoding time can be further decreased when more reference domain vectors are used.

These efficient algorithms have been further investigated to extend their applicability to image magnification and denoising. In this thesis, an efficient image-magnification algorithm based on the Iterated Function System (IFS) is proposed. This IFS-based image-magnification method employs the self-similarity property instead of the conventional interpolation approach. This self-similarity makes it possible to generate images of higher resolution. To further improve the quality of the high-resolution images, the error image or residual errors are considered. In addition, our algorithm can combine with other magnification algorithms. We have also derived a new fractal-based image denoising method, which employs the decoupling property of the fractal code instead of the conventional fractal coding using the contrast scaling and offset parameters. In order to improve the visual quality of a denoised image, a range-block partitioning scheme is used to generate a set of overlapping sub-images. These sub-images are then averaged to produce an optimal denoised image.

List of Publications

The following technical papers have been published or accepted for publication based on the result generated from this work.

Journal Papers (Accepted)

1. Cheung-Ming Lai, Kin-Man Lam and Wan-Chi Siu, "A Fast Fractal Image Coding based on Kick-out and Zero Contrast Conditions", IEEE Transactions on Image Processing, Vol. 12, No. 11, pp. 1398-1403, November 2003.

2. Cheung-Ming Lai, Kin-Man Lam and Wan-Chi Siu, "An Improved Searching Scheme for Fractal Image Coding", Electronics Letters, Vol. 38, No. 25, pp. 1653-1654, 5 December 2002

Journal Papers (Submitted)

1. Cheung-Ming Lai, Kin-Man Lam, Yuk-Hee Chan and Wan-Chi Siu, “An Efficient Image Magnification Algorithm based on Iterated Function Systems”, submitted to Image and Vision Computing.

2. Cheung-Ming Lai, Kin-Man Lam and Wan-Chi Siu, “Adaptive Fractal-based Image Denoising”, submitted to Electronics Letters.

3. Cheung-Ming Lai, Kin-Man Lam and Wan-Chi Siu, “Fast Fractal Image Coding Using Geometric Inequality”, submitted to Journal of Electronic Imaging.

Conference Papers

1. Cheung-Ming Lai, Kin-Man Lam and Wan-Chi Siu, "Fast Fractal Image Compression Using Feature Vector Matching", in Proceedings, 2nd International Conference on Image and Graphics, pp. 146-153, 16-18 August 2002, Hefei, China

2. Cheung-Ming Lai, Kin-Man Lam and Wan-Chi Siu, “An Efficient Algorithm for Fractal Image Coding using Kick-out and Zero Contrast Conditions”, in Proceedings, IEEE International Symposium on Circuits and systems (ISCAS2003), Vol. II, pp. 480-483, May 2003, Bangkok, Thailand.

3. Cheung-Ming Lai and Kin-Man Lam, “A Non-symmetric Window Search Scheme for Fractal Image Coding”, 4th ACM Postgraduate Research Day 2003, pp. 222-225, 23 January 2003, Hong Kong.

4. Cheung-Ming Lai, Kin-Man Lam, Yuk-Hee Chan and Wan-Chi Siu, “An Efficient Fractal-based Algorithm for Image Magnification”, 2004 International Symposium on Intelligent Multimedia, Video & Speech Processing (ISIMP2004), pp. 571-574, October 2004, Hong Kong.

5. Cheung-Ming Lai, Kin-Man Lam and Wan-Chi Siu, “An Efficient Algorithm for Fractal Image Coding using Kick-out and Zero Contrast Conditions”, IEEE Signal Processing Postgraduate Forum 2004, 8 May 2004, Hong Kong.

6. Cheung-Ming Lai, Kin-Man Lam and Wan-Chi Siu, “An Edge-preserved Image Denoising Technique based on Iterated Function Systems”, accepted to appear in Proceedings, Visual Communications and Image Processing 2005, Beijing, China, 12-15 July, 2005

Acknowledgements

I would like to take this opportunity to express my sincere gratitude to my Chief Supervisor, Dr. K. M. Lam, as well as my Co-supervisor, Professor W. C. Siu, from the Department of Electronic and Information Engineering of The Hong Kong Polytechnic University, for their patience and guidance throughout my research studies. My Ph.D. studies would have never been completed without their supervision. Their immense enthusiasm for research is greatly appreciated. Both their professional advice and plentiful experience help me to further improve myself.

I am also thankful to all the members of the DSP Research Laboratory, especially for Dr. W P. Chow, Mr. Alex Lin and Mr. K. W. Wong for their supportive and constructive comments, that will never be forgotten. Their expert knowledge and friendly encouragement have helped me to overcome a lot of difficulties during my research study. I would also like to thank the Centre for Multimedia Signal Processing of the Department of Electronic and Information Engineering for generous support over the past four years.

Last but not least, without the patience and forbearance of my family, the preparation of this research work would have been impossible. I appreciate their constant and continuous support and understanding.

Statements of Originality

The following contributions reported in this thesis are claimed to be original.

1. A new fast algorithm for fractal image compression based on new feature vectors and the property of zero contrast is proposed. The proposed feature vectors can provide better representation of image blocks, resulting in a more efficient search of the domain block.
2. A fast algorithm for fractal image coding based on a single kick-out condition and the zero contrast prediction is presented. The single kick-out condition can avoid a large number of range-domain block matching when finding the best matched domain block.
3. An efficient method for zero contrast prediction is also proposed, which can determine whether the contrast factor for a domain block is zero or not, and compute the corresponding difference between the range block and the transformed domain block efficiently and exactly.
4. A method of modeling local variances based on a symmetrical window is derived.
5. A non-symmetric windowing scheme for fractal image coding based on the local variances method is proposed. The original local variances method uses a symmetric search window to search the best-matched domain block in the domain pool for each range block. In our method, a non-symmetric search window is used in such a way that windows cover those domain blocks where variances are higher than that of the range block are considered.
6. A new representation of range-domain matching function is derived, which shows the relationship between the range-domain block matching error and the angle

between the range and domain block vectors. An inequality in terms of the sine of the angle has been proposed. This inequality can reject impossible domain block candidates efficiently in searching for the best-matched domain block

7. An image magnification algorithm that can achieve a high subjective and objective quality is proposed. This algorithm improves the quality of a magnified image based on a set of high-resolution images generated using IFS codes. A pixel-averaging scheme and an error compensation scheme are proposed to form a magnified image which can preserve its high-frequency information and remove blocky artifacts. In addition, this algorithm can combine with other magnification algorithms, such as IEUF, to further improve the quality of a magnified image
8. A super-high-resolution image magnification algorithm with high visual quality is proposed. In order to extract the information from the original image as much as possible, we propose generating more super-high-resolution images by adopting more range block sizes in the encoding process. Typically, the range block sizes to be adopted can be adjusted from 3×3 to 8×8.
9. An efficient image denoising algorithm is proposed which employs the decoupling property of the fractal code instead of the conventional fractal code using the contrast scaling and offset parameters. This decoupling property makes it possible to denoise images more efficiently and flexibly. To improve the image quality, a range-block partitioning scheme is used to generate a set of overlapping sub-images. The range blocks of these sub-images are coded and represented by their range-block mean values and contrast scaling parameters for removing the noise. These sub-images are then averaged to obtain an optimal denoised image.

Table of Contents

Chapter 1. Introduction 1

1.1 Overview of some Efficient Image Coding Techniques 1

1.1.1 Why Image Compression? 1

1.1.2 Vector quantization 2

1.1.3 JPEG (Joint Photographic Experts Group) 3

1.1.4 JPEG2000 3

1.1.5 Fractal Image Coding 4

1.2 Motivation of Fractal Image Coding 5

1.3 Organization of the thesis 7

Chapter 2. Review of Fractal Coding 10

2.1 History and Development of Fractal Image Coding 10

2.2 Basic Theory 11

2.2.1 Iterated Function Systems (IFS) 12

2.2.2 Partitioned Iterated Function Systems (PIFS) 13

2.2.3 Collage Theorem 14

2.3 Overview of Fractal Image encoding 15

2.3.1 Range Block 16

2.3.2 Domain Blocks 17

2.3.3 Construction of Fractal Codes 21

2.3.4 Image Partitioning 25

2.3.5 Coefficient Quantization 27

2.4 Fractal Image Decoding 27

2.5 Review of Some Fractal Image Coding Algorithms 32
2.5.1 Heavy Brute Force 33
2.5.2 Block classification 33
2.5.3 Domain pool reduction 34
2.5.4 Fast Full Search Algorithms 36
2.5.5 Region-based coding 37
2.6 Review of Some Applications of Fractal Image Coding 38
2.6.1 Fractal-based Image Magnification 39
2.6.2 Fractal-based Image Denoising 42
2.7 Conclusions 43
Chapter 3. Fast Fractal Image Compression Using Feature Vector Matching 44
3.1 Introduction 44
3.2 Proposed Algorithm 45
3.2.1 Fast computation of the feature vectors 49
3.2.2 An efficient fractal coding for low complexity range block 50
3.2.3 Increase of zero contrast cases 52
3.3 Experimental Results 53
3.4 Conclusions 58
Chapter 4. A Non-Symmetric Window Search Scheme For Fractal Image Coding .. 59
4.1 Introduction 59
4.2 Improved Searching Scheme 60
4.3 Experimental Results 62
4.4 Conclusions 66

Chapter 5. A Fast Fractal Image Coding based on Kick-out and Zero Contrast Conditions 67
5.1 Introduction 67
5.2 Proposed Algorithm 68
5.2.1 The Kick-out Condition 69
5.2.2 Fast Error Calculation using Zero Contrast Prediction 70
5.2.3 Combining Other Approaches 71
5.3 Experimental Results 72
5.4 Conclusions 75
Chapter 6. Fast Fractal Image Coding Using Geometric Inequality 79
6.1 Introduction 79
6.2 The Proposed Algorithm 80
6.3 Experimental Results 85
6.4 Conclusions 89
6.5 A Summary of Our Proposed Fractal Image Coding Techniques 89
Chapter 7. An Efficient Image Magnification Algorithm based on Iterated Function Systems 92
7.1 Introduction 92
7.2 Our Proposed Algorithms 93
7.2.1 Adaptive Overlapped Range Block Partitioning 93
7.2.2 Pixel Averaging 96
7.2.3 Error Compensation 97
7.2.4 Combined with Other Algorithms 99

7.3 Experimental Results 99
7.4 Conclusions 108
Chapter 8. Adaptive Fractal-based Image Denoising 109
8.1 Introduction 109
8.2 Proposed Algorithm 111
8.2.1 The Basic Idea of Fractal-based Image Denoising 111
8.2.2 Overlapped Range Block Partitioning Scheme 113
8.2.3 The Best Denoised Image 114
8.3 Experimental Results 115
8.4 Conclusions 118
Chapter 9. Conclusions and Future Work 129
9.1 Conclusion on the current works 129
9.2 Future Work 131
9.2.1 Human Face Image Compression Using Fractal Coding 131
9.2.2 Fractal Coding Of Video Sequences 132
9.2.3 An Efficient Fractal-based Super High Resolution Image Reconstruction using a Single Image 132

List of Figures

Figure 2-1 An example of self-similarity: the small square is similar to the larger square 11

Figure 2-2: The test image Lena of size 512×512 with range block of size 8×8 17

Figure 2-3 The domain pool construction process. 18

Figure 2-4. Eight isometric operations..... 20

Figure 2-5 Diagrams for different partitioning schemes: (a) Fixed Block Partition, (b) Quad-tree partition and (c) HV partition 26

Figure 2-6. Lena of size 256×256 under different iterations with a constant initial image of value 0. 29

Figure 2-7. Lena of size 256×256 under different iterations with a constant initial image of value 128 30

Figure 2-8. Lena of size 256×256 under different iterations with original image as initial image..... 31

Figure 2-9 (a) The fractal image decoding process and (b) the conventional fractal image magnification approach. 40

Figure 2-10 A magnified image of size 512×512 generated by the conventional fractal image magnification technique, with PSNR equal to 31.40dB..... 41

Figure 3-1 Horizontal features and vertical feature for (a) 16×16, (b) 8×8, and (c) 4×4 blocks 48

Figure 3-2 A 8×8 block undergoes 180° rotation..... 51

Figure 3-3 PSNR vs. Compression Ratio for the “Lena” image 54

Figure 3-4 PSNR vs. Compression Ratio for the “Peppers” image. 54

Figure 3-5 Compression ratio vs. PSNR for Lena in different threshold values..... 57

Figure 4-1. The distribution of the best-matched domain blocks..... 61

Figure 4-2 Search windows for (a) local variance method, and (b) our proposed scheme. 62

Figure 6-1. The angle between a range block and a domain block. 81

Figure 6-2 P and Q denote the maximum and minimum range-domain block matching errors, respectively. 82

Figure 6-3. Preprocessing process. 84

Figure 6-4. Number of reference vectors versus the encoding speedup for our proposed algorithm with λ=2 using the image Lena of size 512×512. 86

Figure 7-1 (a) A low-resolution image whose range blocks are extracted at the position (3,1), and (b) the corresponding magnified image $M_{(3,1)}$ with a range block size of 4×4. 96

Figure 7-2 The error compensation process. 98

Figure 7-3 The magnified image quality against the range block of size 3×3 to 8×8 for different test images with magnification factor of 2. 100

Figure 7-4. The PSNR versus the number of pixels being averaged for the test images (a) Lena and (b) Boat..... 102

Figure 7-5 A comparison of average run-time required by our algorithm using the *k*-d tree search with different domain grid sizes versus the average PSNR drop when compared to the full search scheme. 105

Figure 7-6 (a) The original high-resolution image “Lena”, and the magnified images of Lena based on (b) the conventional fractal magnification; (c) the bicubic spline

interpolation; (d) the IEUF; (e) the pixel-averaging scheme; (f) the pixel-averaging scheme with error compensation; and (g) the pixel-averaging scheme with error compensation combined with IEUF. 106

Figure 7-7 (a) The original high-resolution image "Baboon", and the magnified images of Lena based on (b) the conventional fractal magnification; (c) the bicubic spline interpolation; (d) the IEUF; (e) the pixel-averaging scheme; (f) the pixel-averaging scheme with error compensation; and (g) the pixel-averaging scheme with error compensation combined with IEUF. 107

Figure 8-1. The first range blocks (represented by thick line) of size $n_r \times n_r$ (n_r = 4) overlapped sub-images obtained from a noise image. The starting point coordinates of the respective sub-images are (i, j), where $0 \leq x,y \leq n_r - 1$, and the sub-images are denoted as $f_{N(i,j)}$. 119

Figure 8-2 The denoising performance obtained using both the range-block mean and the contrast scaling, and using the range-block mean only under different noise levels. 120

Figure 8-3. The PSNR versus the number of pixels being averaged for different test images when the standard deviation is 10. 121

Figure 8-4 The PSNR versus the number of pixels being averaged for different test images when the standard deviation is 30. 122

Figure 8-5 The denoising performances of our algorithm and the FID in terms of PSNR under different noise levels σ. 125

Figure 8-6 The visual qualities of our proposed algorithm and the FID method based on the test image "Barbara" under different noise levels σ. 126

Figure 8-7 The visual qualities of our proposed algorithm and the FID method based on the test image “Lena” under different noise levels σ. 127
Figure 8-8 The visual qualities of our proposed algorithm and the FID method based on the test image “Boat” under different noise levels σ. 128
Figure 9-1 Traditional Fractal magnification Algorithm with magnification factor α=8135
Figure 9-2 Bicubic Spline Interpolation with magnification factor α=8 136
Figure 9-3 Proposed Algorithm with magnification factor α=8 137
Figure 9-4 Traditional Fractal magnification Algorithm with magnification factor α=8. 138
Figure 9-5 Bicubic Spline Interpolation with magnification factor α=8. 139
Figure 9-6 Proposed Algorithm with magnifiocatin factor α=8 140

List of Tables

Table 2-1. Decoded image quality using different initial images. 32

Table 3-1 Performances of different algorithms based on the image “Lena” 56

Table 3-2 Performances of different algorithms based on the image “Peppers” 56

Table 5-1 Comparison of the coding results using domain grid of one and three level quadtree partitioning (16×16, 8×8 and 4×4)..... 76

Table 5-2 comparison of computational complexity for the full search and our proposed algorithm..... 76

Table 5-3 Comparison of the coding results using domain grid of one and 8×8 range block..... 77

Table 5-4 Adaptive search algorithm [75]: the encoding results using the test image lena of size 256×256 with range block size 4×4 and domain grid 4×4. The CPU is PII233MHz..... 77

Table 5-5 Our proposed algorithm: the encoding results using the test image lena of size 256×256 with range block size 4×4 and domain grid 4×4. The CPU is PIII 500MHz. 78

Table 5-6 Tree search algorithm [119]: the encoding results using the test image lena of size 256×256 with range block size 4×4 and domain grid 2×2. The machine is sun sparcstation 10 model 30 78

Table 5-7 Our proposed algorithm: the encoding using the test image lena of size 256×256 with range block size 4×4 and domain grid 2×2. The CPU is PIII 500MHz. 78

Table 6-1 Comparison of the coding results with different parameters using (a) Lena, (b) Peppers, (c) Boat and (d) Goldhill of size 512×512, domain grid of two and three level quadtree partitioning (16×16, 8×8 and 4×4). 87

Table 7-1 Performances in terms of the PSNR of our proposed algorithm and other algorithms based on different test images using a full search. 103

Table 7-2 Performances in terms of the PSNR of our proposed algorithm and other algorithms based on different test images using the *k*-d tree. 103

Chapter 1. Introduction

The objective of this chapter is to introduce the general concepts of some well-known still image compression techniques, which include Vector Quantization (VQ), JPEG (Joint Photographic Experts Group), JPEG2000 and Fractal image coding. The advantages and potential for image processing of fractal image coding over other image coding techniques will be discussed.

1.1 Overview of some Efficient Image Coding Techniques

1.1.1 Why Image Compression?

Image compression is the process of converting an original digital image into smaller sizes in order to efficiency achieve for storage and transmission. As one of the enabling technologies of the multimedia revolution, image compression is a key to rapid progress being made in information technology. It would not be practical to put images alone on websites without compression. Image compression algorithms are used to reduce the number of bits required to represent an image. Compression is the process of representing image information in a compact form. After compression, image data take up less storage space and require less bandwidth to be transmitted over the Internet. Although increasing the bandwidth is a possible solution, the relatively high cost makes this less attractive. Therefore, compression is a necessary and essential method for creating image with manageable and transmittable sizes.

Fox example, a single image captured by a digital camera with 6M Pixels 24 bits/pixel of true color, will produce a file containing more than 24 megabytes of data. At least 17 floppy disks are required to store such an image. This image requires about 50 minutes for transmission by a typical transmission line (64k bit/second ISDN). Hence, the need to find efficient compression and coding techniques urges on the development some of image compression standards.

Many image coding algorithms have been devised to represent an image with some information loss which cannot be perceived by human eyes. Thus, lossy coding has received many research interests in the field of image compression and can be classified into four main categories: (i) Vector Quantization (VQ) [1-23], (ii) JPEG (Joint Photographic Experts Group) [24-27], (iii) JPEG 2000 [28-32] and (iv) fractal image coding [11, 33-81]

1.1.2 Vector quantization

Vector Quantization (VQ) is a well-known Image compression technique, which has been widely applied in image coding. VQ is based on the fact that grouping the source outputs together will yield more efficient compression. In other words, it encodes a sequence of samples instead of the individual samples. In Vector Quantization, the input image to be encoded is divided into a set of non-overlapping image blocks. The encoder takes an input image block and outputs the index of the corresponding codeword that gives the minimum distortion. In this case, the minimum distortion is found by evaluating the Euclidean distance between the input image block and each codeword in the codebook. Once the best-matched codeword is found, the index of that codeword is sent through a channel. Compression is achieved by transmitting the indices associated to

the codewords instead of the codewords because of the far fewer bits required for the indices. The codebook is generated from a set of training images.

1.1.3 JPEG (Joint Photographic Experts Group)

JPEG is currently the most widely used image compression method. In JPEG coding, an image to be encoded is first divided into a set of 8 by 8 pixel image blocks, and then the discrete cosine transform (DCT) is performed on each block. A quantizer rounds off the DCT coefficients according to the quantization matrix. This process produces the lossy coding of the original image but allows for a large compression ratio. Because adjacent image pixels are highly correlated, the DCT technique can achieve data compression because most of the signal energy is concentrated in the lower spatial frequencies. After quantization, all of the quantized coefficients are ordered into the zig-zag sequence. JPEG compression technique uses entropy coding technique on these quantized DCT coefficients. The entropy coding achieves additional compression losslessly. In the decompression process, JPEG recovers the quantized DCT coefficients from the compressed data stream, and takes the inverse transforms.

1.1.4 JPEG2000

JPEG 2000 was implemented as an entirely new way of image compression based on the wavelet transforms, in contrast to the DCT used in the original JPEG standard. Wavelet Transform is more flexible in representing images and is able to take into account Human Visual System characteristics. It also has good energy compaction

capabilities, which results in a high compression ratio. This standard will include many modern features including improved low bit-rate compression performance, lossless and lossy compression, etc. In JPEG2000 encoder, the discrete wavelet transform is first applied on the source image. The wavelet coefficients are then quantized and entropy coded, then forming the output bitstream. In the decoding process, the bitstream is first entropy decoded, dequantized and inverse discrete wavelet transformed, thus resulting in the reconstructed image data. JPEG2000 can achieve both lossy and lossless coding of an image. This depends on the wavelet kernels used and the quantization applied

1.1.5 Fractal Image Coding

Fractal theory was first presented by Barnsley [33], and is based on a mathematical theory called Iterated Function Systems (IFS). Jacquin [45, 46] proposed the first practical fractal image compression scheme that relies on the assumption that image redundancy could be efficiently exploited through self-transformability on a block-wise basis. Fractal image compression is based on the representation of an image by a set of iterated contractive transformations for which the reconstructed image is an approximate fixed point and close to the original image. In the fractal image coding scheme [45, 46], an image to be encoded is partitioned into a set of non-overlapping square blocks called range blocks. Each range block is represented by a transformed larger domain block of the same image with respect to contrast and intensity offset, as well as the isometric transformations. By minimizing the difference between the range block and the transformed domain block, the corresponding contractive affine transformation can be found for the range block. After finding the best-matched domain blocks for all range

blocks, the corresponding contractive affine transformation parameters are stored for each range block instead of their pixel intensities.

1.2 Motivation of Fractal Image Coding

A significant amount of research work has been done on fractal image compression recently[11, 35-81]. Fractal image coding can provide a highly reconstructed image quality with a high compression ratio, is independent of resolution, and has a fast decoding process. Fractal encoding is essentially a process to find a contractive affine transformation for each range block from the domain pool. We only need to store the location of the best-matched domain block, and perform isometric operation as well as the contrast scaling and luminance offset parameters at the encoding process. It is obvious that we can save quite a lot of bits by doing this instead of their pixel intensities. So the most valuable advantage of fractal compression is the ability to achieve a high compression ratio. The reconstructed fractal image can retain edge sharpness well and can generate textures that are quite often subjectively acceptable.

Both JPEG and JPEG2000 are symmetrical compression methods which require equal processing capability for compression and decompression of an image. However, the VQ and fractal image coding techniques can achieve very fast and simple decoding process. Fractal image compression can be seen as a type of vector quantization with a codebook that is extracted from the original image itself. One of the differences is that no codebook is required for fractal image coding in the decoding process.

Another advantage of fractal coding over other image coding techniques is that it is resolution-independent in the decoding process. After fractal image coding, the output of

the coder is an Iterated Function Systems (IFS), which approximates the image as a fixed point of a contractive transformation. This IFS code can be used to reconstruct the image at any level of resolution. These magnified images are suitable for graphics systems that are typically composed of devices of different resolutions. This type of feature is not exploited by traditional image coders, and this is the main motivation for fractal image coding.

In addition, fractal image coding can also be applied to denoise image. The idea of fractal image denoising is that range blocks can be described by a small number of contractive affine transformations based on the self-similarity structures across scales. However, it is impossible to approximate any random noisy structures occurring in an image [3]. Therefore, the conventional fractal image coding technique can be applied to remove the noise in a low-noise condition.

The problem with fractal coding is its high computational complexity in the encoding process. Most of the encoding time is spent on finding the best matched domain block from a large domain pool to represent an input range block with respect to contrast and intensity offset, as well as the isometric transformations. By minimizing the difference between the range block and the transformed domain block, the corresponding iterated contractive transformation can be found for the range block. The exhaustive search algorithm can obtain the optimal result by searching exhaustively all the blocks within the domain pool, but this process requires a high computational cost and limits its practical applications. Consequently, many fast algorithms, such as Fisher's canonical classification [34], local variances [56], etc. have been proposed to reduce the required computation.

The objectives of this research are to investigate and develop efficient techniques for fractal image coding and their applications for image processing. We have also devised a number of efficient algorithms for fractal image compression, image magnification and image denoising.

1.3 Organization of the thesis

The rest of this thesis will give an overview of existing techniques for fractal image coding, as well as for fractal-based image magnification and fractal-based image denoising.

Chapter 2 describes the principles of fractal image coding. We make a brief review on some well-known fractal image coding techniques, which include fast full search, domain pool reduction search, etc.

In Chapter 3, we propose a fast fractal image coding algorithm using a feature vector which can have a better representation of the image blocks. This can make the search of the corresponding domain block much more efficient and accurate than that of other existing fractal algorithms. We have also proposed the use of zero contrast condition to reduce the computational complexity and further improve the compression efficiency of the algorithm.

In Chapter 4, a fast algorithm for fractal image coding based on a single kick-out condition and the zero contrast prediction is presented. The single kick-out condition can avoid a large number of range-domain block matches when finding the best matched domain block. An efficient method for zero contrast prediction is also proposed, which can determine whether the contrast factor for a domain block is zero or not, and compute

the corresponding difference between the range block and the transformed domain block efficiently and exactly.

In Chapter 5, we devise an equation to model local variances method based on a symmetrical window. From the devised equation, we propose using a non-symmetric window to search for the best-matched domain block based on the local variances method.

Chapter 6 presents an algorithm to reduce the computational cost of range-domain block matching in the fractal image encoding process based on geometric inequalities. Our approach uses the angle between an input range block and a reference domain block to determine a tighter decision boundary for eliminating the searching space in the domain pool. The encoding time can be further decreased when more reference domain vectors are used.

Chapter 7 presents an efficient image magnification algorithm based on the Iterated Function System (IFS). This IFS-based image magnification method employs the self-similarity property instead of the conventional interpolation approach. This self-similarity makes it possible to generate images of higher resolution. To further improve the quality of the high-resolution images, the error image or residual errors are considered. In addition, our algorithm can combine with other magnification algorithms.

In Chapter 8, a new fractal-based image denoising method is proposed. This method employs the decoupling property of the fractal code instead of the conventional fractal coding using the contrast scaling and offset parameters. In order to improve the visual quality of a denoised image, a range-block partitioning scheme is used to generate a set of

overlapping sub-images. These sub-images are then averaged to produce an optimal denoised image.

We give the conclusions of our work in Chapter 9, where some suggestions for further development can also be found.

Chapter 2. Review of Fractal Coding

The objective of this chapter is to introduce the general concepts of Fractal Image Coding and its applications, such as Fractal-based image denoising and image magnification. The state-of-the art technology for fractal image coding and its applications will be presented. An overview of the techniques for fractal image coding, fractal-based image denosing and , fractal-based image magnification proposed and developed in this thesis will also be given.

2.1 History and Development of Fractal Image Coding

The term "fractal" was first used by Benoit Mandelbrot in 1975. Fractals are shapes that exhibit the property of self-similarity, where same parts of shape are presented recursively in different scale and position. Figure 2.1 shows an example of self-similarity of test image Lena. If an image is truly fractal in nature, the image can be reconstructed by using a set of affine transformations of an arbitrary input image. The basic idea of fractal image compression came from a mathematical theory called Iterated Function Systems (IFS). Michael Barnsley wrote a popular book called *Fractals Everywhere* [33] that presents the mathematics of IFS, which is a new tool for modeling techniques in computer graphics, called the Collage Theorem. The Collage Theorem states what conditions an IFS must satisfy in order to represent an image.

Figure 2-1 An example of self-similarity: the small square is similar to the larger square

The basic idea of fractal image compression using Partitioned Iterated Function Systems (PIFSs) was first proposed in 1989 by Jacquin. In his PhD thesis, Jacquin developed the necessary mathematical foundations and implemented the new approach in software, a description of which appears in his paper in [45]. The algorithm was not sophisticated and is expensive computations in the encoding process, but it was fully automatic. All contemporary fractal image compression schemes are based upon Jacquin's approach.

2.2 Basic Theory

The essence of most fractal-based image coding methods is to approximate each segment of an image by applying a contractive affine transformation on some bigger segment in the same image. One can then reconstruct the image with some error by using

only the parameters of the contractive affine transformation [45]. In this method, most of information in the image is basically encoded by coding relations among different segment of the image. In this session, the mathematical theory of IFS, PIFS and Collage Theorem are briefly introduced.

2.2.1 Iterated Function Systems (IFS)

Iterated Function Systems is the foundation for Fractal Image Compression. The basic idea of an Iterated Function System is to create a finite set of contractive affine transformations based on what image one desires to create. If these transformations are contractive, applying the IFS to an arbitrary seed image will eventually produce an attractor of that transformation [34, 45].

An affine transformation $\tau : R^n \to R^n$ can always be written as $\tau = Ax + b$, where $A \in R^{n \times n}$ and $b \in R^n$ is an offset vector. For example, an affine transformation defined on gray scale image can be of the form

$$\tau_i \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} p_i \\ q_i \\ o_i \end{bmatrix} \qquad (2.1)$$

where a_i, b_i, c_i, d_i controls the rotation and skewing of the image, while p_i, q_i controls the translation; s_i controls contrast scaling and o_i controls luminance offset of the transformation, respectively. An affine transformation is contractive if and only if the distance between any two points in the image plane becomes smaller after transformation.

An iterative function system, $\boldsymbol{\tau} = \{\tau_i\}$, is a finite set of affine transformations that composes a single contractive transformation:

$$\boldsymbol{\tau} = \bigcup_{i=1}^{n_r} \tau_i, \qquad \tau_i : R^3 \to R^3 \tag{2.2}$$

$$d\left(\boldsymbol{\tau}(f_o), \boldsymbol{\tau}(f_1)\right) \le s d(f_0, f_1), \qquad 0 \le s < 1 \tag{2.3}$$

where the τ_i is the affine transformation, f_0 and f_1 are any two images at the same dimension. The symbol *d* denotes a distance metric, normally the root mean square that measures the distance between two images. Because the contractivity factor *s* lie in the range [0, 1], the transformation τ makes these two images to become more alike. The Banach's fixed point theorem ensures that when applied to any input image f_0, the IFS have the following remarkable properties: (i) The dynamical system $\{\boldsymbol{\tau}(f_0)\}$ has a unique attractor: $f_\infty = \lim_{n\to\infty} \boldsymbol{\tau}^n(f_0)$, where *n* is the number of iterations. (ii) The unique attractor is a fixed point image of transformation $\boldsymbol{\tau} : \boldsymbol{\tau}(f_\infty) = f_\infty$. That is, starting with any image f_0 and repeatedly applying τ, after certain iterations, the process will converge to a unique attractor image which is independent of the input image f_0.

2.2.2 Partitioned Iterated Function Systems (PIFS)

Theoretically, each image has a unique fixed point. However, it is impossible to find a unique fixed point for a whole image in practice. Therefore, the image should be partitioned as different parts, and the fixed points for the corresponding parts should be obtained through different affine transformations. The above IFS will be different from different parts of the image, and therefore it is called a partitioned iteration function

system (PIFS). This is contrast to IFS that approximates each part of the image by applying a contractive affine transformation on another part of the image. In PIFS, the image to be encoded is partitioned into range blocks $\Re$. Then, each range block $R_i \in \Re$ is approximated by a transformed version of a bigger domain block D_j.

The process of encoding images requires us to find a collection of contractive affine transformations $\tau = \bigcup \tau_i$ such that βf is the fixed point of the map τ. The fixed-point equation suggests that we partition f into pieces to which we apply the affine transformation τ_i to get back the original image f [33]. But finding an exact set of transformations is impossible since arbitrary images are not self-similar. Therefore, we find another image βf with $d_{rms}(f, \beta f)$ less than or equal to some specified tolerance error. The encoder finds the contractive transformation τ_i that minimize the RMS error, $d_{rms}(f, \beta f)$.

In the decoding process, an arbitrary image f_0 is used as a seed image for iterating through the collection of affine transformations. On the first iteration, $f_1 = \tau(f_0)$, and on the second iteration, $f_2 = \tau(f_1) = \tau(\tau(f_0))$, etc. This process can be repeated until the attractor resembles the original image.

2.2.3 Collage Theorem

Collage theorem states what conditions an IFS must satisfy in order to represent an image. The IFSs are good for generating natural looking images. However, the inverse problem is more difficult and challenging; given an arbitrary image f_{orig} to be encoded, find its corresponding IFS to produce another image similar to the original image f_{orig}. In

other words, find a finite set of maps $\{\tau_i\}$ such that $d\left(f_{orig}, f_{\infty}\right)$ is as small as possible. However, finding such a transformation τ is a very complex problem. This problem is solvable by the collage theorem [1]:

$$d\left(f_{orig}, f_{\infty}\right) \leq \frac{1}{1-s} d\left(f_{orig}, \boldsymbol{\tau}\left(f_{\infty}\right)\right) \tag{2.4}$$

The collage theorem [33] suggests a possible approach to this problem, i.e., the problem of "finding a τ whose fixed point is close to the image" can be changed to "finding a $\boldsymbol{\tau}$ mapping the image close to itself" to guarantee that the fixed point of this transformation is close to the original image. One thus needs only to minimize the so-called collage distance $d\left(f_{orig}, \boldsymbol{\tau}\left(f_{\infty}\right)\right)$ instead of $d\left(f_{orig}, f_{\infty}\right)$ to encode a give image f_{orig}. It should be pointed out that the transformation τ found in this way generally is not optimal, but it is a compromise that one has to make in order to solve the above minimization problem.

2.3 Overview of Fractal Image encoding

A significant amount of research work has been done on fractal image compression recently [11, 34-69, 71-118]. Jacquin[5] proposed the first practical fractal image compression scheme that relies on the assumption that image redundancy could be efficiently exploited through self-transformability on a block-wise basis. Fractal image compression is based on the representation of an image by a set of iterated contractive transformations for which the reconstructed image is an approximate fixed point and close to the original image.

2.3.1 Range Block

In the fractal image compression scheme, an image, f_{orig}, of size $m_x \times m_y$ is partitioned into two basic block units: the range blocks and the domain blocks. The range blocks $\Re$ are a set of non-overlapping image blocks of size $k=n_r \times n_r$, which are denoted as $\{R_i\}_{i=1}^{N_R} = \{r_{i1}, r_{i2}, \ldots, r_{ik}\}_{i=1}^{N_R}$. The number of range blocks is $N_R = \left(\frac{m_x}{n_r}\right) \times \left(\frac{m_y}{n_r}\right)$, and image f_{orig} is a union of $\{R_i\}_{i=1}^{N_R}$:

$$f_{orig} = \bigcup_{i=1}^{N_R} R_i \tag{2.5}$$

Note that the range block is the primary coding unit in fractal image coding and is encoded one by one in raster order from left to right, top to bottom. Figure 2.2 shows a test image Lena of size 512×512 with range block of size 8×8.

Figure 2-2: The test image Lena of size 512×512 with range block of size 8×8

2.3.2 Domain Blocks

Overlapping image blocks of f_{orig} with a size larger than that of the range blocks are called domain blocks. These domain blocks can be obtained by sliding a window of size $l = n_d \times n_d$, where $n_d > n_r$, throughout the image to construct the domain pool. The size of a domain block is usually four times that of a range block, i.e. $l = 2n_r \times 2n_r$. To encode a range block *R*, each of the blocks in the domain pool is scaled to the size of the

range block, and is then compared to R with respect to intensity offset o and contrast scaling s parameters, as well as the isometric transformations. The set of contracted domain blocks is denoted as $\{D_i\}_{i=1}^{N_D} = \{d_{i1}, d_{i2} ..., d_{il}\}_{i=1}^{N_D}$, where N_D is the number of domain blocks in the domain pool.

As described above, for a range block, a best-matched domain block and its IFS need to be found so that range block and domain block become the best pair with minimum error. The domain block pool consists of $2n_r \times 2n_r$ squares from the original image and is a set of domain blocks. It can be generated by sliding $2n_r \times 2n_r$ window within the original image by skipping δ pixels from left to right, top to bottom as shown in Figure 2.3.

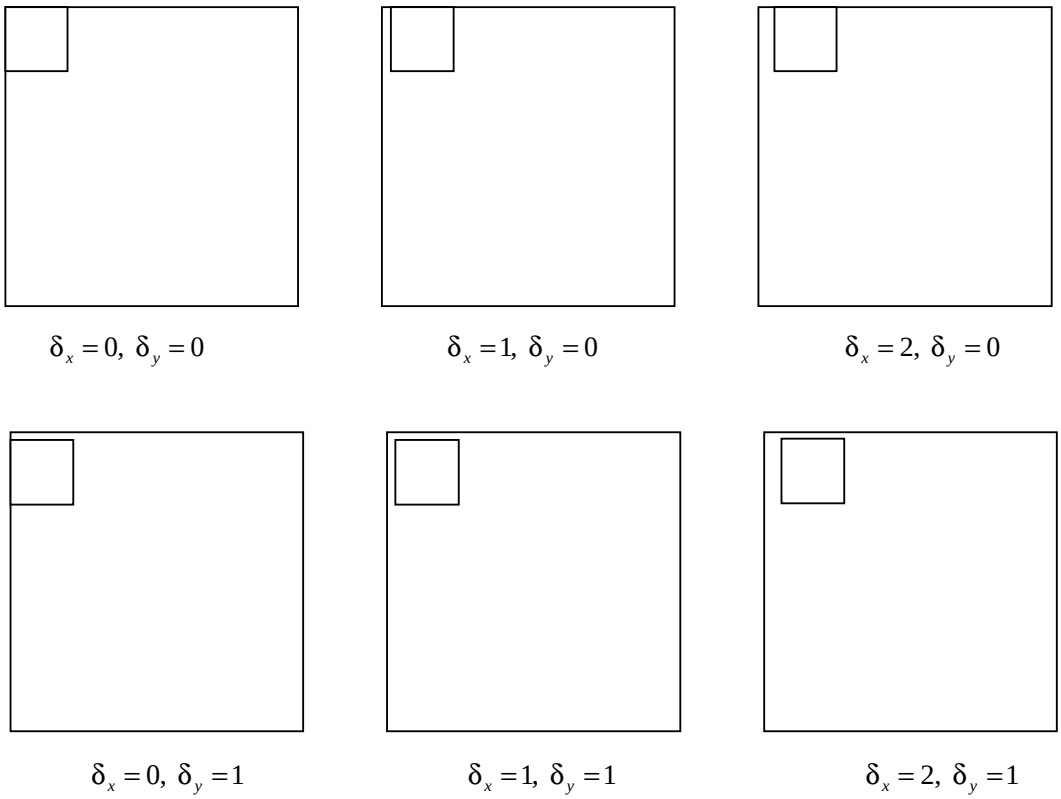


Figure 2-3 The domain pool construction process.

If the image is $m_x \times m_y$, then there are $\left(\frac{m_x - 2n_r}{\delta} + 1\right) * \left(\frac{m_x - 2n_r}{\delta} + 1\right)$ domain blocks in the domain pool. For example, if the image of size 256×256 and range block of size 4×4, the domain grid size δ is 1, then there are $\left(\frac{256 - 2\times 4}{1} + 1\right) * \left(\frac{256 - 2\times 4}{1} + 1\right)$=62,001 domain blocks in the domain pool. Here we take notation of the domain block pool as $\Omega = \{D_j\}$. Comparing the range block with all domain blocks among the domain block pool one by one, find the best-matched domain block. To increase the probability of finding a best-matched domain blocks for a range block, the domain pool is enlarged by including all eight isometric operations of the original domain blocks, as shown in Figure 2.4. The transformed domain blocks are obtained by rotating original blocks by 90, 180 and 270 degrees and flipping them vertically. In other words, there are $\left(\frac{256 - 2\times 4}{1} + 1\right) \times \left(\frac{256 - 2\times 4}{1} + 1\right) \times 8 = 496,008$ domain blocks in the domain pool.

0: No transformation

1: Reflection in the y axis

2: Reflection in the x axis

3: 180° Rotation

4: Reflection in the line y = x

5: Rotate 90° counter-clockwise

6: Rotate 90° clockwise

7: Reflection in the line y = –x

Figure 2-4. Eight isometric operations.

2.3.3 Construction of Fractal Codes

After both range blocks and domain blocks are defined, a contractive affine transformation is defined on each range block R_i mapping a domain block to each range block. The action of this affine transformation can be seen to consist of three parts: the first part is to extract an approximate domain block from the domain pool, the second part is to transform the selected domain block, and the last part places the transformed domain block at the range block location and compare the range-domain matching error. Mathematically, for each range block R, an affine transformation is constructed in the following way:

The collage error minimization is carried out by exhaustive searching the domain pool to find the best-matched domain block with an isometric operation and corresponding parameters, contrast scaling s and luminance offset o to minimize the distance. The domain block with its corresponding parameters and isometric operation that gives the minimum distance is stored. The corresponding parameters for the affine transformation τ_i are determined by minimizing the following equation:

$$E(R,D) = \sum_{i=1}^{k} (r_i - sd_i - o)^2 \qquad \text{where } o, s \in \boldsymbol{R} \tag{2.6}$$

where r_i and d_i are the pixels in the range block R and the transformed domain blocks D, respectively and they are both ordered lexicographically. The contrast scaling s and luminance offset o parameters can be calculated explicitly by selecting their corresponding partial derivatives of (2.6) to zero, i.e.,

$$\frac{\partial}{\partial s}\left[\sum_{i=1}^{k}\left(r_i - s.d_i - o\right)^2\right] = 0 \tag{2.7}$$

$$\frac{\partial}{\partial o}\left[\sum_{i=1}^{k}\left(r_i - s.d_i - o\right)^2\right] = 0 \tag{2.8}$$

To find the minimum of the error function the partial derivatives of $E(R, D_i)$ must be set to zero. By solving these simultaneous equations in s and o, we can obtain the explicitly expression for the optimal parameters s and o as follows:

$$s = \frac{k\left(\sum_{i=1}^{k} r_i d_i\right) - \left(\sum_{i=1}^{k} r_i\right)\left(\sum_{i=1}^{k} d_i\right)}{k\left(\sum_{i=1}^{k} d_i^2\right) - \left(\sum_{i=1}^{k} d_i\right)^2} \tag{2.9}$$

$$o = \frac{1}{k}\left[\left(\sum_{i=1}^{k} r_i\right) - s\left(\sum_{i=1}^{k} d_i\right)\right] \tag{2.10}$$

If $k\left(\sum_{i=1}^{k} d_i^2\right) - \left(\sum_{i=1}^{k} d_i\right)^2 = 0$, then $s=0$ and $o = \frac{1}{k}\sum_{i=1}^{k} r_i$. The equations (2.9) and (2.10) can be substituted into equation (2.6) to obtain the following expression:

$$E\left(R, D\right) = \frac{1}{k}\left[\sum_{i=1}^{k} r_i^2 + s\left(s\sum_{i=1}^{k} d_i^2 - 2\sum_{i=1}^{k} r_i d_i + 2o\sum_{i=1}^{k} d_i\right) + o\left(ko - 2\sum_{i=1}^{k} r_i\right)\right] \tag{2.11}$$

The contrast scaling s and offset parameters o serve to minimize the square error between range block R and the transformation domain block D. The contrast factor should be $-1 < s < 1$ to ensure the contractivity of the transformation. The domain block which results in the smallest difference with equation (2.11) is then chosen as the best-matched domain block, and the corresponding parameters τ_i for the transformations are encoded and stored.

The above encoding process is repeated to encode all of the range block one-by-one in raster order and the corresponding contractive affine transformations are encoded and stored $\{\tau_i \mid i = 1,2,\ldots,N_R\}$. The fractal code used to represent an image is then the union of the parameters τi of all the range blocks as follows:

$$\tau = \mathbf{r}_{i=1}^{N_r} \tau_i \tag{2.12}$$

2.3.3.1 Convolution Method

Equation (2.6) can be rewritten to a more compact format as

$$E(R, D_i) = \|R - (s.D_i + oI)\|^2, \quad \text{where } o, s \in \Re, \tag{2.13}$$

where D_i is the transformed domain block under isometric transformation, I denotes a unity vector of dimension k, respectively. For a given range block and the corresponding domain block, these two parameters are given as follows:

$$s = \frac{\langle R, D\rangle - \frac{1}{k}\langle R, I\rangle\langle D, I\rangle}{\|D\|^2 - \frac{1}{k}\langle D, I\rangle^2}, \text{ and} \tag{2.14}$$

$$o = \frac{1}{k}\left(\langle R, I\rangle - s\langle D, I\rangle\right) \tag{2.15}$$

The $\|\ \|$ is the two-norm and $\langle .,. \rangle$ is inner product.

$$E(R, D) = s\left(s\|D\|^2 + 2.o\left(\langle D,1\rangle - \langle R, D\rangle\right)\right) + o\left(o.k - 2\langle R,1\rangle\right) + \|R\|^2 \tag{2.16}$$

The range-domain matching error can be regarded as a function of $\langle R, D\rangle$, $\|D\|^2$, $\|R\|^2$, $\langle R,1\rangle$ and $\langle D,1\rangle$. The calculation of the inner products $\langle R, D\rangle$ dominant the computational cost in the encoding process.

2.3.3.2 Øien's Method

In order to reduce the computational cost, equation (2.13) can be further simplified. As advocated by Øien *et al* [58] and by Tong *et al* [73], the DC component of all image blocks is subtracted. The full search problem becomes

$$E(R,D) = \left\| s\left(D - \bar{d}I\right) - \left(R - \bar{r}I\right) \right\|^2 \tag{2.17}$$

The contrast scaling can be rewritten as follows:

$$s = \frac{\left\langle R - \bar{r}, D - \bar{d} \right\rangle}{\left\| D - \bar{d} \right\|^2} \tag{2.18}$$

In other words, the optimal scaling factor between a range block and a domain block is their inner product divided by the domain block sum-of-squares. This value is calculated for all candidate domain blocks, under all eight symmetry operations, in search of the minimum error. The representation of a range block can be rewritten as follows:

$$\begin{aligned} R &= s.D_i + oI \\ &= s\left(.D_i - \bar{d}\right) + s\bar{d} + oI \\ &= s\left(.D_i - \bar{d}\right) + s\bar{d} + \left(\bar{r} - s\bar{d}\right)I \\ &= s\left(.D_i - \bar{d}\right) + s\bar{d} - s\bar{d} + \bar{r}I \\ &= s\left(.D_i - \bar{d}\right) + \bar{r}I \end{aligned} \tag{2.19}$$

Note that the fractal parameters are now the parameters s and range block mean $\bar{r}$, instead of the conventional contrast scaling s and offset parameter o. The range-domain matching error becomes:

$$E(R,D) = s\left\| D - \bar{d}I \right\|^2 + \left\| R - \bar{r}I \right\|^2 - 2.s\left\langle R - \bar{r}I, D - \bar{d}I \right\rangle \tag{2.20}$$

The computational complexity of equation (2.20) is smaller than equation (2.16).

2.3.4 Image Partitioning

In designing fractal image compression, we should consider the partition of the image. Image partitioning is about dividing the image into sections that are more appropriate for the application to work on. The partition for image compression is performed to improve the rate distortion characteristic of the encoder. Three types of partitioning method will be described in the follow: Fixed Size Block Partitioning [45], Quad-tree partitioning [34] and Horizontal-Vertical partitioning [34, 95]. Figure 2.5 is an illustration of these methods.

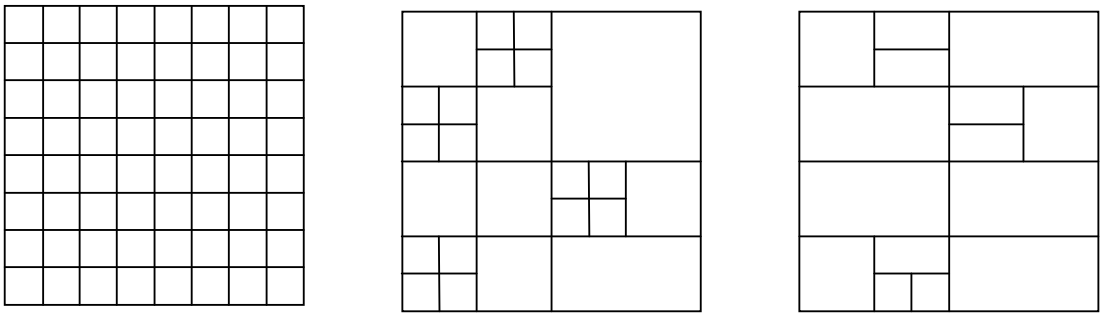
2.3.4.1 Fixed Size Block Partitioning

In Fixed Size Block Partitioning scheme [45], the original image is only partitioned into a fixed size of range blocks. No splitting will be carried out to the range blocks no matter how large the distance is between a range block and a transformed domain block. The size of the range blocks is an important parameter in the encoding scheme. An encoding with small range blocks results in a high signal to noise ratio but a low compression ratio. With a larger size of range blocks, the compression ratio is increased but the details in the range block are lost. To capture fine details while preserving a high compression ratio, the quad-tree partitioning scheme is introduced.

2.3.4.2 Quad-tree Partitioning

The Quadtree partitioning scheme [34] is the most common scheme for fractal image coding. In this scheme, the original image to be encoded is first divided into 4 range blocks. Each of these quarters, in turn can be divided up into 4 more equal sized range blocks. This goes on until suitable mapping of the range blocks can be found, or a

maximum depth is reached. An adaptive partitioning of an image may hold strong advantages over encoding range blocks of fixed size. There may be homogeneous image regions in which a sufficient collage error can be attained using large blocks, while in high contrast regions smaller block sizes may be required to arrive at the desired quality.



(a) (b) (c)

Figure 2-5 Diagrams for different partitioning schemes: (a) Fixed Block Partition, (b) Quad-tree partition and (c) HV partition

2.3.4.3 HV Partitioning

The horizontal-vertical (HV) partitioning scheme produces a tree-structured partition of an image. This partitioning scheme is similar with quad-tree scheme. Instead of recursively splitting quadrants, however, each image block is split into two by a horizontal or vertical line. In the encoding process, the range block is split into two smaller blocks, either horizontally or vertically if the matching error is larger than a pr-defined threshold. In such a way, splitting positions may be constructed so that boundaries tend to fall along prominent edges. Both the domain blocks created in this fashion are recursively split in this manner until the matched domain block is found. The

advantage of the HV method is that larger domain blocks are likely because at each recursion the domain blocks are halved in size instead of quartered.

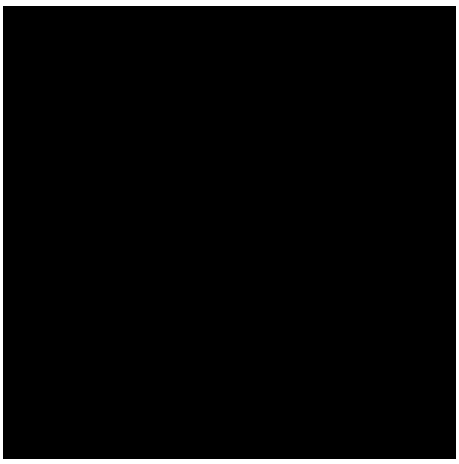
2.3.5 Coefficient Quantization

Once the best-matched domain block and the affine transformations are obtained for each range block, the compression ratio can be further increased by reducing the number of bits used to represent the parameters in the affine transformations. There are five parameters for each transformation τ_i: the isometric operation φ_i, the location of the best-matched domain $\left(p_x, p_y\right)$, the contrast scaling parameters s_i and the luminance offset o_i. It is reasonable to store the isometric operations used as three bits, since there are eight possible operations. The number of bits required to store the location of the best-matched domain $\left(p_x, p_y\right)$ depends on the domain grid δ and the corresponding equations are $\log_2\left(m_x\right)-\log_2\left(\delta\right)$ and $\log_2\left(m_y\right)-\log_2\left(\delta\right)$, respectively. There are 5 bits and 7 bits needed to quantize the contrast scaling s_i and luminance offset o_i parameters, respectively, as suggested by Fisher [36].

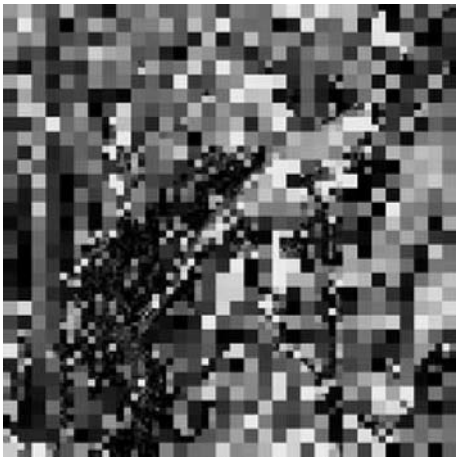
2.4 Fractal Image Decoding

At the decoding phase, the contractive transformation parameters τ are recursively applied to an arbitrary initial image. The output of each step is used as the input for the next iteration and the output will converge to a unique fixed point that is independent of the initial input image. The fixed point is known as the attractor of the IFS. The attractor

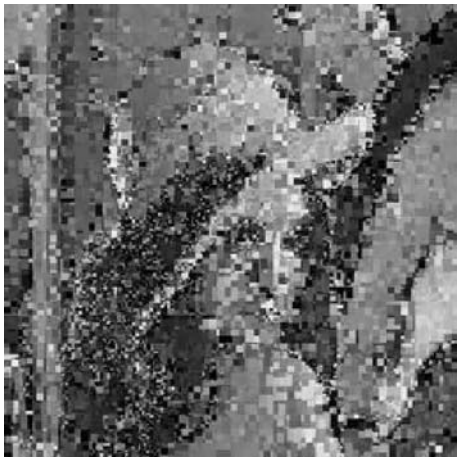
has details at every scale and is referred to as a fractal. Figure 2.6 illustrates the use of an initial image of constant gray level 0. After two iterations, the original image has been recognizable, and the output converges to the original image after eight iterations. Figures 2.7 and 2.8 illustrate the decoding processes using another constant image of value 128 and original image as initial image. Table 2.1 tabulates the PSNR after each of the iteration with different initial images. The results show that the result converges and have the same PSNR after 8 iterations.



(a) Initial Image



(b) After one iteration

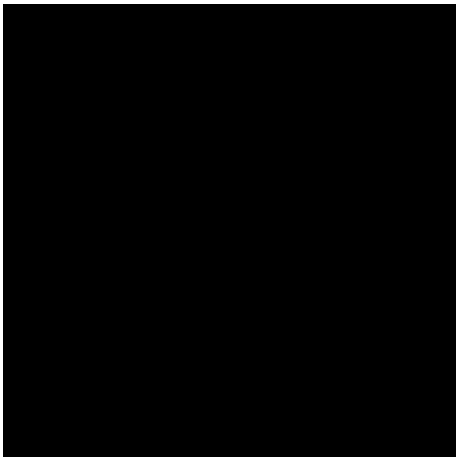


(c) After two iterations

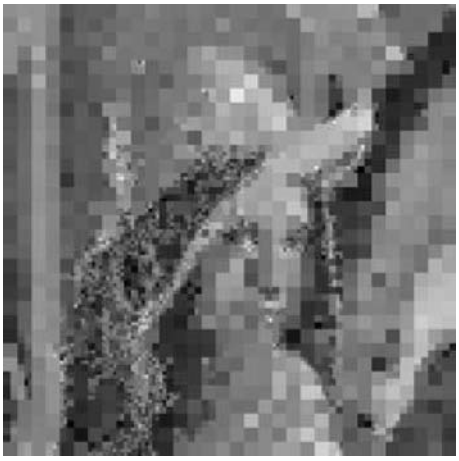
(d) After three iterations

(e) After eight iterations

Figure 2-6. Lena of size 256×256 under different iterations with a constant initial image of value 0.



(a) Initial Image



(b) After one iteration

(c) After two iterations

(d) After three iterations

(e) After eight iterations

Figure 2-7. Lena of size 256×256 under different iterations with a constant initial image of value 128

(a) Initial Image

(b) After one iteration

(c) After two iterations

(d) After three iterations

(e) After eight iterations

Figure 2-8. Lena of size 256×256 under different iterations with original image as initial image

No. of Iteration	Initial Image 0	Initial Image 128	Initial Image Orig. Image
1	10.6 dB	19.5 dB	33.02 dB
2	14.7 dB	23.78 dB	32.76 dB
3	19.35 dB	28.01 dB	32.66 dB
4	24.51 dB	31.16 dB	32.67 dB
5	31.93 dB	32.41 dB	32.67 dB
6	32.55 dB	32.65 dB	32.67 dB
7	32.65 dB	32.67 dB	32.67 dB
8	32.67 dB	32.67 dB	32.67 dB
9	32.67 dB	32.67 dB	32.67 dB
10	32.67 dB	32.67 dB	32.67 dB

Table 2-1. Decoded image quality using different initial images.

2.5 Review of Some Fractal Image Coding Algorithms

A significant amount of research work has been done on fractal image compression recently. Fractal image coding can provide a highly reconstructed image quality with a high compression ratio, is independent of resolution, and has a fast decoding process. The problem with fractal coding is its high computational complexity in the encoding process. Most of the encoding time is spent on finding the best matched domain block from a large domain pool to represent an input range block with respect to contrast and intensity

offset, as well as the isometry transformations. By minimizing the difference between the range block and the transformed domain block, the corresponding iterated contractive transformation τ_i can be found for the range block. The exhaustive search algorithm can obtain the optimal result by searching exhaustively all the blocks within the domain pool, but this process requires a high computational cost and limits its practical applications.

2.5.1 Heavy Brute Force

The original images of size 256×256 shown in Figure 2.2, are based on a uniform partitioning scheme of range block of size 8×8 and produces 1024 range blocks. If the domain blocks are restricted to twice the size of range blocks, the domain pool contains 8×(256-8+1)×(256-8+1)=496,008 domain blocks. In total, 496,008×1024≈508M possible range-domain matching are required. As the size of the domain pool is very large, the search of the optimal domain block for a range block is computationally intensive. Many fast fractal algorithms [11, 35-81, 119, 120] have been proposed to speed up this searching process.

2.5.2 Block classification

Block classification [34, 65, 79, 86] is the most widely used technique to speed up the encoding process. In the block classification scheme, the original image is first partitioned into non-overlapping range blocks and overlapping domain blocks. All the domain blocks are divided into several classes based on some predefined features. Only those domain blocks and range blocks with the same feature take part in the domain-

range matching process. Block classification can lead to a substantial computational cost saving without much fidelity loss.

Fisher *et al.* [34] proposed a well-known quadrant variance classification scheme, in which an image block is divided into four subblocks. The average pixel intensity A_i and the variance V_i, where i=1,…,4, for each of the four subblocks are computed. Based on these techniques, the domain blocks are classified into 3 major classes, each of which is further divided into 24 subclasses. The range blocks are also categorized using the same method. When searching for a matching domain block, only those of the same category as the range block are considered. However, the search complexity of this algorithm is still linearly proportional to the domain pool size, and the number of image blocks in the subclasses may not be even. It is possible that one subclass has more image blocks than another. In addition, it is difficult to extend the search to any other neighboring classes in case a domain block sufficiently similar to a range block cannot be found in one subclass.

2.5.3 Domain pool reduction

One of the simplest ways to speed-up the encoding process is to decrease the size of the domain pool in order to reduce the number of domain blocks to be searched. Lee *et al.* [56] proposed using local variances as the feature to speed up the matching of a domain block and a range block. This method arranges the domain blocks in order according to the magnitudes of their corresponding variances. With an input range block, its corresponding variance is computed and the domain block with the closest variance to this range block is identified. The best-matched domain block is then searched within a

search window centred at the identified range block. In [56], the window covers 4.7% of the total domain blocks in its upper and lower parts with respect to its center.

In [76], Wan *et al.* proposed an efficient fractal coding method that takes advantage of the correlation between range blocks. In this method, four candidate domain blocks, mapped by the previous neighboring blocks of the input range block, are searched. If one of the four candidates whose range-domain matching error is smaller than the pre-defined error threshold, then this domain block is selected to match the current range block without any other searching in the domain pool. Otherwise, a fast fractal coding method is performed to find the best-matched domain block to match the input range block.

In [121], Saupe proposed a fast fractal image compression scheme, where an image block is normalized to have zero mean and unity variance before encoding. This makes it possible to determine the block of minimum distortion without considering the intensity offset and contrast parameters between the two image blocks. Therefore, the search for an optimal domain block is reduced to a nearest neighbor search in the Euclidean space. Since all range blocks are compared against the same domain pool, the use of the k-d tree can achieve a significant speedup.

In [43], C. He *et al.* proposed a new variance-based scheme to speed up the fractal image encoding process. Firstly, range blocks are grouped into two classes according to the variability of their intensities. If the variance of a range block is below an a priori fixed threshold, the block is regarded as a shade block, otherwise as a non-shade block. After classification, all intensity values in a shade range block are replaced by the mean of the range block. In addition, they found that that the variance of a range block and the best-matched domain block cannot differ greatly, hence the best-matched block to a

given range block should be the neighbor of the range block in the sense of intensity variances.

Saupe *et al.* [70] proposed another domain pool reduction algorithm to speed-up the encoding process. In this method, an adjustable number of domain blocks with low intensity variance are excluded from the domain pool. Thus, a large fraction of low variance blocks is discarded and will affect only a few range blocks. This paper indicates that the best-matched domain blocks are located mostly along edges and in regions of high contrast of the image. The goal is to store this bitmap which contained the high variance domain blocks efficiently. Then the number of bits required to identify a particular used domain block is reduced. If the structure of the bitmap is strong, then these savings are greater than the overhead necessary for coding the bitmap and an overall reduced file size for the code can be achieved. Therefore, The WBS storage scheme is adopted to encode the bitmap to reduce the bit rate.

2.5.4 Fast Full Search Algorithms

Fast Full Search Algorithms [47, 54, 67, 68, 75] for fractal image coding can achieve the same reconstructed image quality as the exhaustive search, and can greatly reduce the required computation or runtime. Fractal image coding via fast convolution [68, 85] is one of these fast full search methods. In this method, the domain block that yield the minimum error is obtained rather than an acceptable but suboptimal one. The domain blocks are typically defined by down-filtering the image to half its resolution. The inner products are the finite impulse response (FIR) of the down-filtered domain block with respect to the range blocks. Thus, the cross-correlation of the range block with

the domain block is required. This discrete two-dimensional convolution can be carried out more efficiently in the frequency domain when the range block is not too small.

In [75], Truong *et al.* proposed a fast full search coding in the frequency domain. In fractal image coding, it is necessary to consider the eight orientations of the allocated domain block for each input range block. That means it requires eight separate mean square error (MSE) computations for between the input range block and selected domain block. These eight operations are very high computational processes. Therefore, this paper shows that the redundancies in these eight orientations can be reduced to two in frequency domain after Discrete Cosine Transform DCT.

2.5.5 Region-based coding

This category consists of so-called region-based fractal coders. For high image-adaptivity and better rate-distortion performance, all published region-based fractal coders (RBFC) [35, 40, 72, 88, 122] take the split-and-merge approach to further increase the image quality and compression. In [35], a region based coding of still image based on fractal is proposed. Firstly, the image to be encoded is segmented into different semantically-meaningful regions. In order to assure the encoded region can be decoded in the receiver, both range and domain blocks must be located within the same region. In practice, the boundary domain block does not fully enclose the range block segment. Therefore, extrapolation is required during encoding. These regions are then encoded one-by-one independently. This kind of encoding scheme permits new functionalities at the receiver. Moreover, the region of interest can be transmitted first in a transmission channel with limited bandwidth. It also can achieve better image quality than standard

coding since regional boundaries usually contain higher variances that are difficult to encode.

The region-based fractal coder using heuristic search is proposed by Thomas and Deravi [72]. This method starts by choosing one of the range blocks in uniform partition as a seed, and attempts to search for the best-matched domain block. In a recursive manner, the algorithm extends the region from the seed in all four principal directions of the newly extended range block using the same coefficients as the original seed range block. The extension of the domain block region is simply under the same direction corresponding to that of range block extension to reduce the computational complexity.

In [122], Saupe *et al.* devised an adaptive partitions fractal coder that finds the image partition by a split and merge process, yielding range blocks as unions of edge-connected small square blocks. This fractal coder has a better rate-distortion performance and subjective quality than the widely used quadtree-based fractal coders.

2.6 Review of Some Applications of Fractal Image Coding

Fractal image coding has already proven to be a reliable technique that exploits the self-similarity present in an image by representing it as a collection of affine contractive transformations [45]. In the same way, it is possible to utilize the same collection of transformations as features which can be applied to image magnification [123-132], image denoising [133-136], watermarking [137-141], face recognition [142-149], and content-based image retrieval [150-158], etc. In this thesis, fractal-based image magnification and denoising will be investigated.

2.6.1 Fractal-based Image Magnification

One of the advantages of fractal theory is that it is resolution-independent. After fractal image coding, the output of the coder is an IFS, which approximates the image as a fixed point of a contractive transformation. This IFS code can be used to reconstruct the image at any level of resolution. These magnified images are suitable for graphics systems that are typically composed of devices of different resolutions. However, the conventional fractal image magnification method [34] causes serious blocky artifacts due to the independent and lossy coding of the range blocks.

The main idea of fractal image magnification is based on the resolution-independent property of fractal. If we assume that the fractal coding is a fractal process, the fractal code's attractor is a fractal object. By iterating the transformation τ on an arbitrary initial image f_o, a fractal image $\hat{f}$ will be obtained after a number of iterations. This process is independent of the resolution of the final image. According to the fractal theory, the fractal code τ has no intrinsic size, and can be used to construct an image at any level of resolution when the coding error is rather small. These magnified images are suitable for graphics systems that are typically composed of devices of different resolutions. However, the conventional fractal image magnification method [34] causes serious blocky artifacts due to the independent and lossy coding of the range blocks. A number of fractal image magnification approaches have been proposed [123-127]. Chung *et al.* [127] has proposed to use an additional enhancement layer (IEUF) to produce a magnified image without any blocky artifacts. However, that method cannot enhance the high frequency components efficiently.

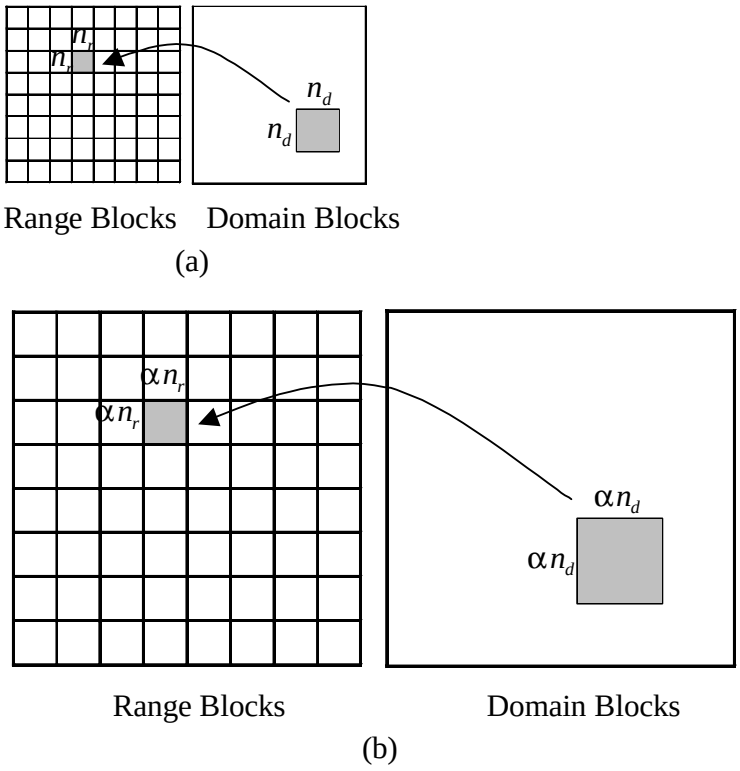


Figure 2-9 (a) The fractal image decoding process and (b) the conventional fractal image magnification approach.

In this thesis, the fractal image magnification algorithm is implemented by using the IFS in equation (2.12) with the magnification factor α a multiple of 2. To achieve a magnification factor of α, the sizes of the range blocks and domain blocks are increased from n_r to αn_r and from n_d to αn_d or $2\alpha n_r$, respectively. Therefore, the location of the best-matched magnified domain block for each magnified range block can be obtained by multiplying the corresponding location in the original image by the factor α. Figure 2.9 illustrates the basic idea the conventional fractal image magnification approach. Figure 2.10 shows a magnified image generated by using this conventional fractal image magnification algorithm.

Figure 2-10 A magnified image of size 512×512 generated by the conventional fractal image magnification technique, with PSNR equal to 31.40dB.

As fractal coding is a lossy process, it is impossible to obtain all the IFS with zero collage error. Therefore, a magnified image will have a visually poor quality. Actually, the fractal-based magnification method causes serious blocky artifacts. The collage error should be taken into account, as this error will also be magnified during the decoding process when performing image magnification. Consequently, this error should be considered in order to achieve a better performance than that of the classical interpolation approaches and conventional fractal image magnification algorithms.

In [127], Chung *et al.* proposed a fractal-based enlargement technique which can preserve the high-frequency contents of a magnified image by introducing an enhancement layer. In this method, a best-matched domain block D_i for a range block R_i of size $n_r \times n_r$ is obtained and the corresponding optimized affine transformations are s_i and o_i and φ_I. The corresponding reconstructed range block $\hat{R}_i$ is given as follows:

$$\hat{R}_i = s_i.\varphi_i\left(D_i\right) + o_i I \tag{2.21}$$

Assume that there exists an ideal domain block D_i^* for R_i, such that the original range block can be obtained with the same affine transformation as follows:

$$R_i = s_i.\varphi_i\left(D_i^*\right) + o_i I \tag{2.22}$$

Then, the ideal domain block can be obtained by using the following equation:

$$\varphi_i\left(D_i^*\right) = \frac{R_i - o_i I}{s_i} \tag{2.23}$$

The difference E_i between the ideal domain block $\varphi_i\left(D_i^*\right)$ and the best-matched domain block in the domain pool $\varphi_i\left(D_i\right)$ is given as follows:

$$E_i = \frac{R_i - o_i I}{s_i} - \varphi_i\left(D_i^*\right) \tag{2.24}$$

The difference is an image block of size $n_r \times n_r$ and its magnified version will be used as an enhancement layer for image magnification.

2.6.2 Fractal-based Image Denoising

One of the basic properties of fractal image coding is the notion of self-similarity in the different scales of a given image. In image coding, a range block can be approximated by a larger domain block from elsewhere in the image after applying decimation and a non-linear intensity transformation. The computed fractal image can be recovered up to a small distortion by an iteration procedure at the decoder. The idea of fractal image denoising is that range blocks can be described by a small number of contractive affine transformations based on the self-similarity structures across scales. However, it is

impossible to approximate any random noisy structures occurring in an image [134, 135]. Therefore, the conventional fractal image coding technique can be applied to remove the noise in a low-noise condition. However, when the noise level is significant, the denoised images will lose their high frequency contents and suffer from blocky artifacts.

A recent fractal image denoising (FID) technique was proposed by Ghazel *et al* [135], which can produce a denoised image efficiently under a heavy noisy condition. The basic idea of the method is to estimate the fractal code of a noise-free image from the corresponding noisy image based on a statistical approach. This fractal-based image denoising technique (FID) can achieve a gain in PSNR of 1.19dB on average when compared to the well-known Lee's filter [159] under the conditions that the standard deviation σ of the noise is larger than 20.

2.7 Conclusions

Throughout this chapter, we have given a precise description on how fractal compression works. Details of the fractal encoding and decoding were also discussed. Fractal image coding is an important step for many applications, such as image magnification, image denoising, watermarking, etc. However, the fractal encoding process suffers from high computational complexity. A significant amount of research work has been done on fractal image compression recently to speed-up the encoding, including domain pool reduction techniques, block classification schemes, and fast full search algorithms, etc. In addition, the various applications of fractal image coding have also been studied.

Chapter 3. Fast Fractal Image Compression Using Feature Vector Matching

3.1 Introduction

This chapter presents a fast fractal coding algorithm based on Saupe's coding scheme [121]. The scheme partitions an image into image blocks of different sizes using quadtree partitioning. In our algorithm, each of the image blocks is normalized to have zero mean and unity variance. Instead of subsampling the image blocks, the means and/or variances of each of their rows and columns are computed and used to form a feature vector. The features to be extracted depend on the size of the image block, so that the vector dimension is a constant and the feature vector itself is a good representation of the image block. In addition, the corresponding feature vectors of the eight symmetrical orientations of the image block can be derived without any computation. The *k*-d tree structure is used to partition the feature vectors of the domain blocks. To reduce computation required for the construction of the *k*-d tree, those feature vectors under the isometric transformations are not included. An efficient encoding approach for simple range blocks is proposed, which encodes the mean values of the range blocks without performing any domain block matching. This can lead to an improvement in encoding time and an increase in compression ratio, while maintaining comparable image quality. Moreover, during the range-domain matching process, a simple but very efficient coding method is proposed by exploiting the property of zero contrast value, which can further improve the encoding time and compression ratio

especially in high compression ratio. Experimental results show that the run-time of our proposed algorithm is over 200 times faster than that of the exhaustive search with comparable picture quality.

This chapter is organized as follows. Section 3.2 presents our new fractal image compression algorithm based on feature vector matching. In Section 3.3, we compare the performance of our proposed fast algorithm with the full search, Fisher [34], local variances [56], and Saupe [121] methods. Finally, conclusions are given in Section 3.4.

3.2 Proposed Algorithm

An image is partitioned into a collection of image blocks by using quadtree partition, and each of the partitioned image blocks are normalized to have zero mean and unity variance. Similar to Saupe's approach, a new feature vector is used to represent an image block. A feature vector is composed of the variances and/or means of each of its rows and columns, which depends on the size of the image block. The extracted means and/or variances from a normalized image block are then concatenated to form a feature vector. This set of means and variances should provide a much better representation of the image blocks, so the search for a best matching domain block will become more accurate as compared to other approaches using a single mean and variance. If the two feature vectors are similar, this implies that the two corresponding image blocks should be similar. For image blocks of 16×16, the variances of every two successive columns and rows are extracted to form a feature vector. The variance of each column and each row are extracted for image blocks of size 8×8, and both the means and variances for the columns and rows are extracted for image blocks of size 4×4. Therefore, all the feature

vectors also have a size of 16, irrespective of the size of an image block. A partitioned block size of 4×4 means that the image block contains a lot of details, so we use relatively more data to represent it. For an image block of 16×16, it should contain fewer details and a relatively smaller amount of data can be used for its representation. The use of the feature vector can also reduce the required computation in the search process, as the size of the feature vector is less than or equal to the size of the image blocks. In addition, as all the feature vectors are of the same size, this facilitates the construction of the *k*-d tree for efficient search.

An $n_r \times n_r$ image block $\boldsymbol{B}=\{b_{ij} \mid i=1,\ldots,n;\ j=1,\ldots,n;\ B \in R^k\}$ is a *k*-dimensional vector, with $k = n_r \times n_r$. The mean, μ, and variance, σ, of this image block are extracted according to the following equations:

$$\mu = \frac{1}{k} \sum_{i=1}^{n_r} \sum_{j=1}^{n_r} b_{ij} \tag{3.1}$$

$$\sigma^2 = \sum_{i=1}^{n_r} \sum_{j=1}^{n_r} \left(b_{ij} - \mu \right)^2 \tag{3.2}$$

This image is then normalized to have zero mean and unity variance as follows:

$$\phi(\boldsymbol{B}) = \frac{1}{\sigma} (\boldsymbol{B} - \mu \boldsymbol{I}) \tag{3.3}$$

where $\boldsymbol{B}$ is the original image block, represented in a vector form, $\phi(\boldsymbol{B})$ is the normalized image block, and $\boldsymbol{I}$ denotes a vector of size n_r^2 with all components equal to one. The normalized image block has the property that the minimum distortion between two image blocks can be determined without considering the offset and contrast factors.

In our algorithm, the variances and/or means of the columns and rows are used to represent an image block. In the following equations, ξ represents the feature vector,

while *M* and *V* represent the mean and variance, respectively. The superscripts "h" and "v" denote "horizontal" the corresponding feature being extracted "horizontal" and "vertically", respectively. For image blocks of size 16×16 and 8×8, only the variances are considered which are sufficient to represent the blocks. The computation of the variances are given as follows:

Horizontal features: $$M_i^h = \frac{1}{n_r}\sum_{j=1}^{n_r}\phi\left(b_{ij}\right) \quad \text{where } j=1,..,n_r \tag{3.4}$$

$$V_i^h = \frac{1}{n_r - 1}\sum_{j=1}^{n_r}\left(\phi\left(b_{ij}\right) - M_i^h\right)^2 \quad \text{where } j=1,..,n_r \tag{3.5}$$

Vertical features: $$M_j^v = \frac{1}{n_r}\sum_{i=1}^{n_r}\phi\left(b_{ij}\right) \quad \text{where } i=1,..,n_r \tag{3.6}$$

$$V_j^v = \frac{1}{n_r - 1}\sum_{j=1}^{n_r}\left(\phi\left(b_{ij}\right) - M_j^v\right)^2 \quad \text{where } i=1,..,n_r \tag{3.7}$$

As mentioned previously, the feature vectors of 4×4 blocks are composed of the means and variances, as given below:

$$\xi^{4\times4} = \left(M_1^h\, V_1^h, ..., M_4^h\, V_4^h\; M_1^v\, V_1^v, ..., M_4^v\, V_4^v\right)^T \tag{3.8}$$

For 8×8 block, the feature vector is composed of its variances only as represented below:

$$\xi^{8\times8} = \left(V_1^h\, V_2^h, ..., V_8^h\;\; V_1^v V_2^v, ..., V_8^v\right)^T \tag{3.9}$$

For 16×16 block, the variances of every two columns and every two rows are extracted to form the feature vector.

Horizontal features: $$M_i^h = \frac{1}{2n_r}\sum_{j=1}^{n_r}\left[\phi\left(b_{ij}\right) + \phi\left(b_{(i+1)j}\right)\right] \tag{3.10}$$

where $j=1,..,n_r$

$$V_i^h = \frac{1}{2n_r - 1}\sum_{j=1}^{n_r}\left[\left(\phi\left(b_{ij}\right) - M_i^h\right)^2 + \left(\phi\left(b_{(i+1)j}\right) - M_i^h\right)^2\right] \quad (3.11)$$

where j=1,3,..,n_r

Vertical features: $$M_j^v = \frac{1}{2n_r}\sum_{i=1}^{n_r}\left[\phi\left(b_{ij}\right) + \phi\left(b_{i(j+1)}\right)\right] \quad (3.12)$$

where i=1,..,n_r

$$V_j^h = \frac{1}{2n_r - 1}\sum_{i=1}^{n_r}\left[\left(\phi\left(b_{ij}\right) - M_j^v\right)^2 + \left(\phi\left(b_{i(j+1)}\right) - M_i^v\right)^2\right] \quad (3.13)$$

where i=1,3,..,n_r -1

The corresponding feature vector is represented as follows:

$$\xi^{16\times16} = \left(V_1^h\ V_2^h, ..., V_8^h\ \ V_1^v V_2^v, ..., V_8^v\right)^T \quad (3.14)$$

Figures 3.1 illustrate the features horizontally and vertically to form a feature vector for different block sizes.

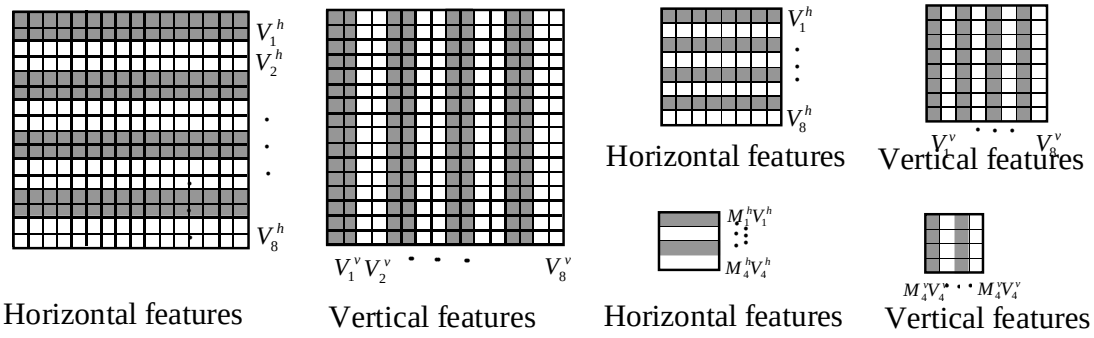


Figure 3-1 Horizontal features and vertical feature for (a) 16×16, (b) 8×8, and (c) 4×4 blocks

3.2.1 Fast computation of the feature vectors

In our approach, a feature vector is computed for each image block, as well as for its eight symmetrical orientations. In this Section, we will show that the feature vectors of the corresponding eight transformations can be obtained from that of the original image block without additional computation. A feature vector is represented as $\xi = \left(\gamma_1^h \gamma_2^h, ..., \gamma_{n_r}^h \gamma_1^v \gamma_2^v, ..., \gamma_{n_r}^v\right)^T$, where γ is the variances for image blocks of size 8×8 and 16×16 block. For 4×4 block, both the means and variances are considered, so $\gamma_i = (M_i V_i)$. The corresponding feature vectors under the eight symmetrical orientations are obtained by the following transformations:

1. Rotation about the center 0°:

$$\tau_0\left(\gamma_i^h\right) = \gamma_i^h, \tau_0\left(\gamma_i^v\right) = \gamma_i^v \tag{3.15}$$

2. Rotation about the center 90°:

$$\tau_1\left(\gamma_i^h\right) = \gamma_{n-1-i}^h, \tau_1\left(\gamma_i^v\right) = \gamma_i^h \tag{3.16}$$

3. Rotation about the center 180°:

$$\tau_2\left(\gamma_i^h\right) = \gamma_{n-1-i}^h, \tau_2\left(\gamma_i^v\right) = \gamma_{n-1-i}^v \tag{3.17}$$

4. Rotation about the center 270°:

$$\tau_3\left(\gamma_i^h\right) = \gamma_i^v, \tau_3\left(\gamma_i^v\right) = \gamma_{n-1-i}^h \tag{3.18}$$

5. Reflection at the vertical axis:

$$\tau_4\left(\gamma_i^h\right) = \gamma_i^v, \tau_4\left(\gamma_i^v\right) = \gamma_{n-1-i}^v \tag{3.19}$$

6. Reflection at the horizontal axis:

$$\tau_5\left(\gamma_i^h\right) = \gamma_{n-1-i}^h, \tau_5\left(\gamma_i^v\right) = \gamma_i^v \tag{3.20}$$

7. Reflection at the first diagonal:

$$\tau_6\left(\gamma_i^h\right)=\gamma_i^v,\tau_6\left(\gamma_i^v\right)=\gamma_i^h \tag{3.21}$$

8. Reflection at the second diagonal:

$$\tau_7\left(\gamma_i^h\right)=\gamma_{n-1-i}^v,\tau_7\left(\gamma_i^v\right)=\gamma_{n-1-i}^h \tag{3.22}$$

where i=1,…,n, n=8 if 8×8 or16×16 block, n=4 for 4×4 block.

The computations required for extracting the feature vectors of the eight orientations can be reduced by applying the above equations. The candidate domain blocks are extracted from the domain pool and the corresponding feature vectors for the eight transformations are generated. In the searching stage, the *k*-d tree is used to find a matching domain block for each range block. The *k*-d tree is a data structure capable of efficiently indexing a multi-dimensional data space. Figure 3.2 shows an example for fast calculation for isometric transformation.

3.2.2 An efficient fractal coding for low complexity range block

Conventional approaches search a domain block with minimum error from the candidate domain pool for an input range block in terms of the contrast and offset parameters by using equation (2.13). However, when the contrast factor is equal to zero, the domain becomes irrelevant. In other words, no domain block or transformation information needs to be encoded in this case. This can happen only when the range block has a fairly constant gray level intensity. The detection of this zero contrast condition can help improve coding efficiency and reduce computation.

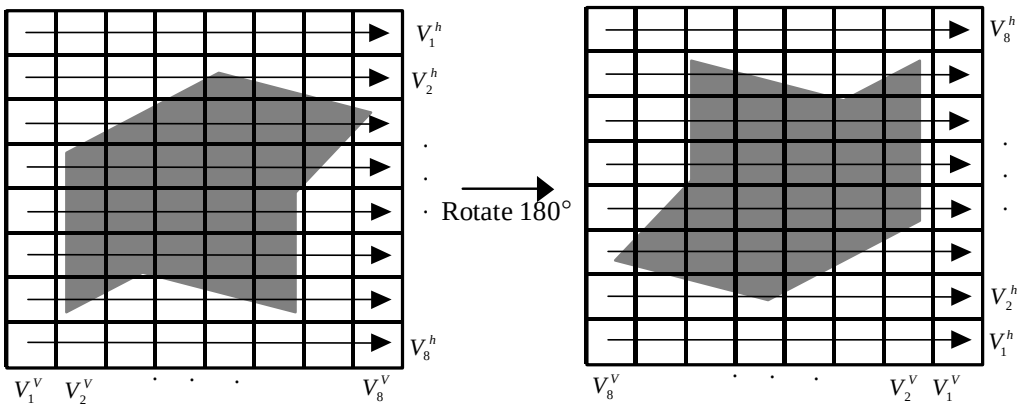


Figure 3-2 A 8×8 block undergoes 180° rotation.

In our algorithm, we also incorporate the detection of zero contrast condition. If the input range block is of a smooth area, it implies that the best matching domain block should also be smooth. Consequently, we can embed the domain block into the offset part by setting the contrast scaling *s* to zero. Therefore, we seek to minimize the distortion using the following equation:

$$E(R,D) = \left\| \mathrm{r_i} - o\,\mathrm{I} \right\|^2 \tag{3.23}$$

It is obvious that the minimum errors occur when the offset *o* is the mean value of the smooth range block. Moreover, the distortion $E(R,D)$ is equal to the variance of the range block. Both the offset and the distortion have been calculated. Therefore, when the variance of the range block is lower than a threshold, the contrast factor is set to zero, and only the mean of the input range block is encoded [45, 46]. This approach can achieve a high compression ratio with high image quality. In addition, the search of the best matching domain block is no longer performed, so the encoding process is speeded

up. If the variance of the range block is higher than the threshold, the *k*-d tree is then used to search for the best matching domain block.

3.2.3 Increase of zero contrast cases

Edge areas are encoded by using the conventional approaches to find a best matching domain block with the contrast scaling and offset parameters. The candidates retrieved from the domain pool using the *k*-d tree scheme are close to the input range block. To exploit the possibility of using the zero contrast condition, instead of searching a domain block with the minimum error criterion, we seek to find a domain block which results in zero contrast while the error is less than a threshold. If this condition is fulfilled, the range block is coded as the zero contrast case. Experimental results show that this proposed search approach further speeds up the encoding process and increases the compression ratio, especially in high compression ratio.

3.3 Experimental Results

In the experiment, a three-level quadtree partition scheme with range block sizes of 4×4, 8×8 and 16×16 pixels, and search grid of two were used. The size of a domain block is four times of a range block. A domain block is scaled to the size of a range block by averaging 2×2 pixels before the error is computed. Our proposed algorithm is compared with the exhaustive search, Fisher [34], Saupe [121] and local variances [56] methods in terms of the runtime and the peak-signal-noise ratio (PSNR). The PSNR is given as:

$$PSNR = 10\log_{10}\left(\frac{255\times 255}{\frac{1}{m_x m_y}\sum_{i=0}^{m_x-1}\sum_{j=0}^{m_y-1}\left(f_{orig}(i,j)-\hat{f}(i,j)\right)^2}\right) dB \qquad (3.24)$$

where f_{orig} and $\hat{f}$ represent the original image and the reconstructed image with size $m_x \times m_y$, respectively, and i, j denote the pixel coordinates in the image block. The experiments were conducted on a Pentium III 500 processor, using monochrome images of size 512×512 with 256 gray levels.

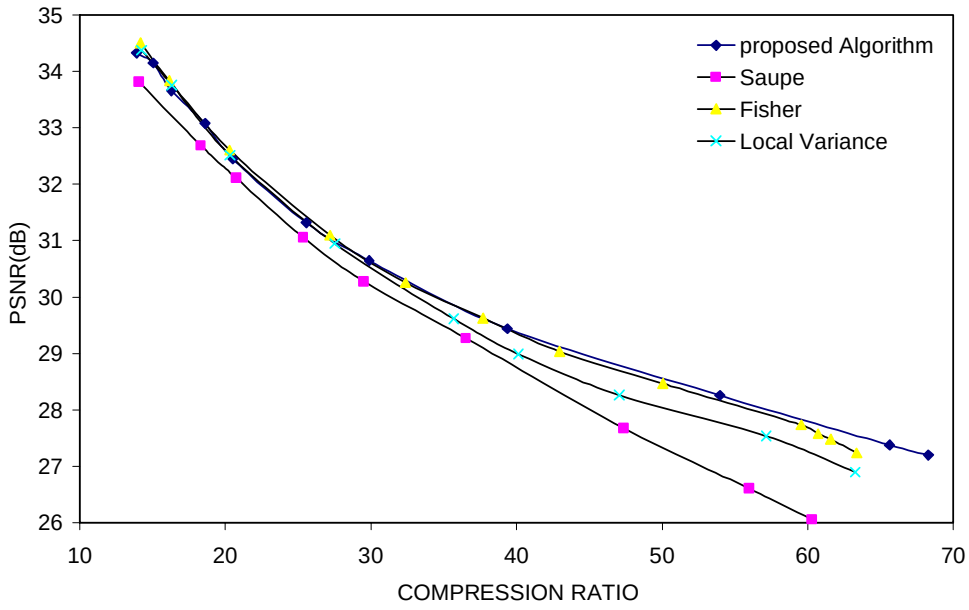


Figure 3-3 PSNR vs. Compression Ratio for the “Lena” image

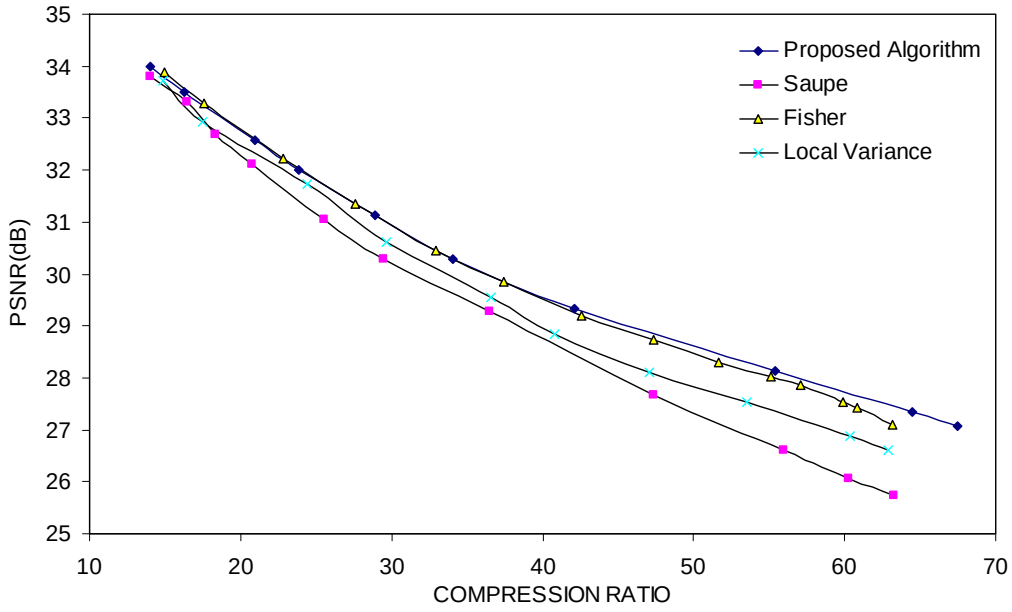


Figure 3-4 PSNR vs. Compression Ratio for the “Peppers” image.

In our proposed approach, the mean and variance of each range block are computed once, and stored in the normalization process, according to equations (3.1)and (3.2). The search for the best matching domain block is skipped if the variance of the range block is small. In this case, as the domain block becomes irrelevant, the contrast factor is set to zero and only the mean of the range block value is coded. If the variance is higher than a threshold, the range block is coded using equation to find the best matching domain block by mean of the *k*-d tree structure. In the *k*-d tree, each image block can be viewed as a *k*-dimensional tree data structure in spatial domain. Using *k*-d tree structure, the decoding time can be improved while the reconstructed image quality remains relatively constant, as we can directly retrieve the best domain blocks for the encoding of a given range block. Both our proposed algorithm and Saupe's algorithm also use a threshold ε to find the nearest neighbors, which is set to 3, and 30 of (1+ε) approximate nearest neighbors are returned.

The performance of the different fractal compression algorithms are compared at two fixed compression ratios: 20.90 and 37.32 for the image "Lena", and 20.20 and 34.20 for the image "Peppers". Our proposed algorithm is compared with (1) exhaustive search, (2) local variances method, (3) Saupe's method and (4) Fisher's method. The local variances method uses a search window size equal to 4.7% of the domain pool size. For Saupe's method, the image blocks are subsampled so that the dimension of the feature vectors is equal to 16. Tables 3.1 and 3.2 tabulate the compression ratios, PSNR, and runtimes for the respective approaches using the two images.

Table 3-1 Performances of different algorithms based on the image “Lena”

	Full Search	Local Variance	Fisher	Saupe	Proposed Algorithm
CR	20.92	20.85	20.90	20.87	20.98
PSNR(dB)	32.82	32.45	32.44	32.11	32.33
Time(s)	2467	227	72	14	12
CR	37.32	37.38	37.37	37.32	37.37
PSNR(dB)	30.07	29.58	29.69	29.21	29.74
Time(s)	1890	173	51	9	8

Table 3-2 Performances of different algorithms based on the image “Peppers”

	Full Search	Local Variance	Fisher	Saupe	Proposed Algorithm
CR	20.23	20.25	20.25	20.24	20.22
PSNR(dB)	32.85	32.52	32.67	32.22	32.69
Time(s)	2546	235	76	13	12
CR	34.12	34.26	34.27	34.27	34.21
PSNR(dB)	30.50	30.13	30.27	29.50	30.35
Time(s)	1985	282	58	9	9

The experimental results show that our proposed method is faster than the other fast algorithms at almost the same PSNR and compression ratio. In Table 3.1, when the compression ratio is 20.9, our proposed approach achieves a PSNR of 32.33dB and requires only 12 seconds needed for encoding. There are 471 range blocks coded using the mean values only, and 30 range blocks are obtained with zero contrast values. That means that only a small number of range-domain matching have to be carried out in the proposed algorithm. For these range blocks, we only need to encode the contrast value with 5 bits and offset value of 7 bits without any additional bit required. The runtime of our proposed algorithm is reduced by approximately 200 times in comparison with the full search with a slight degradation of 0.5dB in image quality. The simulation results also show that the performance of the proposed algorithm is superior to that of Fisher’s, local variances and Saupe approaches. Our proposed approach achieves a precision

similar to that of the local variances and Fisher approaches, but requires less encoding time. At comparable compression ratio and image quality, our approach can be over 12 times and 6 times faster than that of the local variances and the Fisher methods.

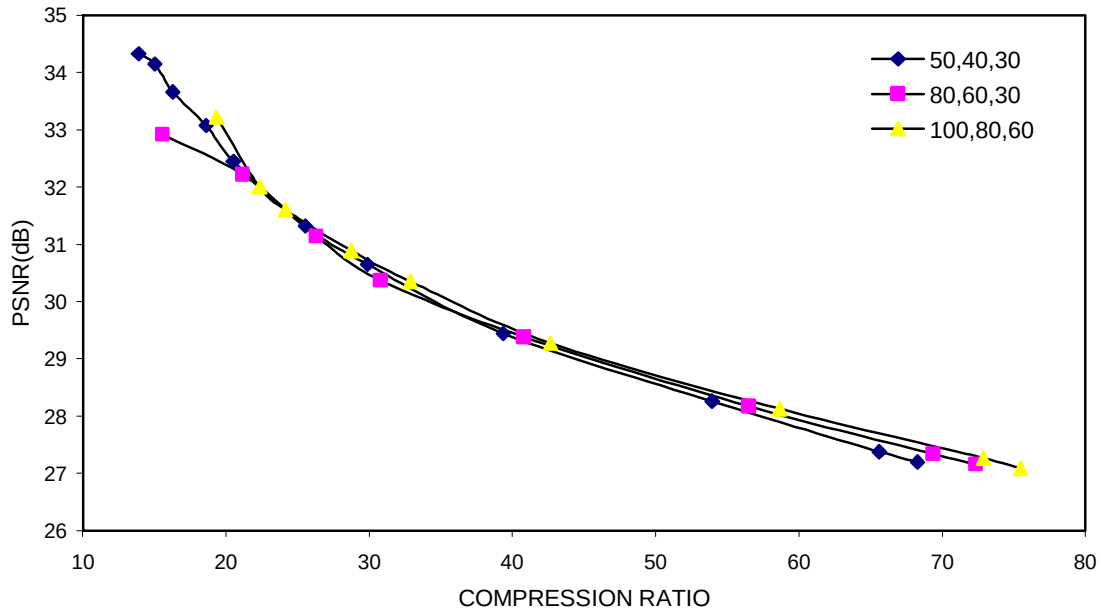


Figure 3-5 Compression ratio vs. PSNR for Lena in different threshold values.

Figures 3.3 and 3.4 illustrate the PSNR against the compression ratio of the respective fractal compression algorithms. Simulation results show that our approach achieves a better PSNR and a shorter encoding time than the Saupe algorithm at the same compression ratio. Experimental results also show that the new algorithm is faster than the other fast algorithms for fractal compression. Moreover, at higher compression ratios, our approach becomes much superior due to the fact that more range blocks will be encoded with the zero contrast condition.

The effect of the variance threshold in the classification of low complexity range block is also investigated. VAR_TH0, VAR_TH1 and VAR_TH2 are denoted as variance

thresholds for image blocks of sizes 16×16, 8×8 and 4×4, respectively. Three sets of thresholds are tested in the experiment: (VAR_TH0, VAR_TH1, VAR_TH2) = (50,40,30), (80,50,30) and (100,800,60). From Figure 3.5, the first set of threshold can achieve a higher PSNR, as the range blocks are coded with smaller distortion.

3.4 Conclusions

We have presented a new fast algorithm for fractal image compression based on new feature vectors and the property of zero contrast. The proposed feature vectors can provide better representation of image blocks, so result in a more efficient search of the domain block. The zero contrast condition is considered and used in encoding. To further reduce the bit rate, the smooth range blocks are coded using their mean values only. This new algorithm is compared with the exhaustive search, local variances approach, Fisher algorithm, and Saupe algorithm. Experimental results show that this new algorithm outperforms the other four algorithms in terms of runtime. In addition, our proposed approach can achieve a speed up of 200 when compared with the full search algorithm.

Chapter 4. A Non-Symmetric Window Search Scheme For Fractal Image Coding

4.1 Introduction

One drawback of fractal image coding is the intensive computation required for searching the best-matched image blocks in a large domain pool for a range block in encoding process. Different fast fractal coding algorithms have been proposed to reduce the searching space in the domain pool. In this chapter, we propose using a non-symmetric window to search for the best-matched domain block based on the local variances method [56]. Experimental results show that, by reducing the window size by half, our proposed scheme can achieve a speedup of 50% with similar PSNR and compression ratio, as compared to the local variances method. With the same window size, our method can have an improvement of PSNR by about 0.25 dB, as well as a higher compression ratio.

This chapter is organized as follows. We present our proposed non-symmetric windowing scheme based on the local variances method in Section 4.2. In Section 4.3, we compare the performance of our proposed fast algorithm with that using a symmetric search window. Finally, conclusions are given in Section 4.4.

4.2 Improved Searching Scheme

We simplify equation (2.13) by removing the mean value of each range block and each transformed domain block. The error function between the range block and the transformed domain block can be rewritten as follows:

$$E(R,D) = \left\| R - \bar{r}I \right\|^2 - s^2 \left\| D - \bar{d}I \right\| \tag{4.1}$$

$$\text{where } s = \frac{\left\langle R - \bar{r}I, D - \bar{d}I \right\rangle}{\left\| D - \bar{d}I \right\|^2},$$

$\| \ \|$ is the two-norm and $\langle .,. \rangle$ represents the inner product. As both the range blocks and domain blocks are demeaned, these two terms, $\|R - \bar{r}I\|^2$ and $\|D - \bar{d}I\|^2$, represent the variances of the range block and the domain block, respectively. Note that the error function depends only on the contrast *s*, and s^2 is less than 1. From equation (4.1), the value of the error function depends on the variances of the range block and the domain block. This value will be at a minimum when $\|R - \bar{r}I\|^2 = s^2 \|D - \bar{d}I\|^2$. As s^2 is less then 1, $\|D - \bar{d}I\|^2$ should be larger than $\|R - \bar{r}I\|^2$ in order to achieve the minimum. In the local variances method, the best-matched domain block is searched in the neighborhood of the domain block with the closest variance to that of the range block. However, a better match should always happen when $\|D - \bar{d}I\|^2$ is larger than $\|R - \bar{r}I\|^2$. Hence, similar search results should be obtained if the search is conducted in the direction where $\|D - \bar{d}I\|^2$ is increasing. In other words, only those domain blocks with variances higher than that of the range block will be considered when searching for the best matched range block, instead of considering both lower and higher variances as in [56]. Figure 4.1 shows the distribution of locating the best matched domain block away from the position where the

variances of the range block and the domain block are closest. It is obvious that the distribution of the best-matched domain blocks with respect to the point of closest variance is non-symmetric. Figure 4.2 illustrates the difference in the search window between the local variances method and our proposed method. Points *A*, *B* and C represent number of best-matched domain blocks within the searching window of size 0-5%, 0- (-5)% and 5-10%, respectively. In our proposed algorithm, points *A* and *C* are chosen instead of *A* and *B* which are chosen uses Local Variance method. Obviously, the total number of best-matched domain blocks inside our proposed searching window is more than that of the Local Variance method. Therefore, our algorithm can achieve better decoded image quality with higher compression ratio.

Figure 4-1. The distribution of the best-matched domain blocks.

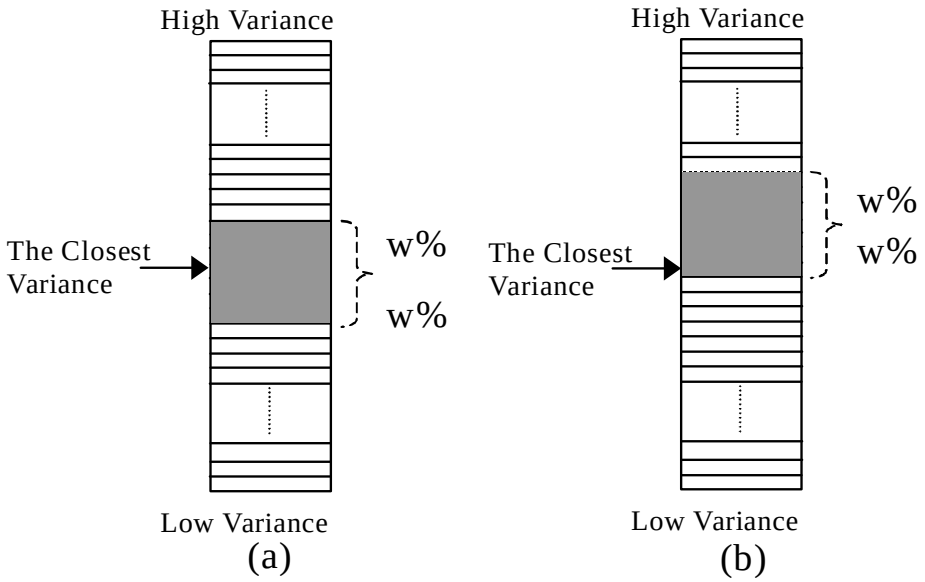


Figure 4-2 Search windows for (a) local variance method, and (b) our proposed scheme.

4.3 Experimental Results

In the experiment, a three-level quadtree partition scheme with range block sizes of 4×4, 8×8 and 16×16 pixels with different error thresholds for the quadtree splitting process and a domain grid of two were used. Four test images: Lena, peppers, Boat and Goldhill, of size 512×512 were used to evaluate the performance of the proposed algorithm and the local variances method. The PSNR is given as follows:

$$PSNR = 10\log_{10}\left(\frac{255\times 255}{\frac{1}{m_x m_y}\sum_{i=0}^{m_x-1}\sum_{j=0}^{m_y-1}\left(f_{Orig}(i,j)-\hat{f}(i,j)\right)^2}\right) dB \qquad (4.2)$$

where f_{orig} and $\hat{f}$ represent the original image and the reconstructed image with size $m_x \times m_y$, respectively, and i, j denote the pixel coordinates in the image.

The computer used is a Pentium III 500MHz. In our method, different sizes of the search window are considered, and the windows cover those domain blocks that have a larger variance than that of the range block. The larger the search window is, the better the decoded image quality and the compression ratio are. Table 4.1 shows the performances of our proposed scheme in terms of the encoding time, PSNR and compression ratio for the four different images. The error thresholds used to form the quadtree are 36, 49, 64, 81 and 100. We found that the performance of the local variances method can be improved by using our proposed non-symmetric searching window. In addition, our algorithm does not impose any additional computational complexity on the local variances method. Experimental results confirm that our scheme achieves better performance in all aspects compared to the local variances method with the same window size. From the experimental results, our proposed scheme can improve the decoded image quality by about 0.25 dB and also increase the compression ratio slightly. The encoding time can be reduced by more than half if only the upper half of the window is used while the decoded image quality and compression ratio are only very slightly decreased.

Table 4.1. A comparison of coding performances of the local variances method and our proposed scheme for the images (a) “Lena”, (b) “Peppers”, (c) “Boat” and (d) “Goldhill”, with error thresholds of 36, 49, 64, 81 and 100. The numbers inside the brackets represent the window size being used. The largest window is 2×4.7%, which means that 2×4.7% of the total domain blocks are covered by the window.

Error Threshold		Algorithms			
		Local Variamces	Propsoed (1*4.7%)	Proposed (1.5*4.7%)	Proposed (2*4.7%)
36	Time (s)	1125	570	835	1097
	PSNR (dB)	34.86	34.85	35.00	35.11
	CR	11.92	11.67	11.91	12.07
49	Time(s)	1017	514	751	985
	PSNR (dB)	33.55	34.09	34.21	34.31
	CR	16.07	13.88	14.14	14.35
64	Time(s)	940	478	702	918
	PSNR (dB)	33.55	33.58	33.67	33.77
	CR	16.07	15.70	16.03	16.13
81	Time(s)	874	446	655	852
	PSNR (dB)	32.87	32.98	33.07	33.08
	CR	18.11	17.57	18.05	18.31
100	Time(s)	810	413	609	794
	PSNR (dB	32.12	32.08	32.32	32.39
	CR	20.32	20.05	20.40	20.69

(a)Lena of size 512×512

Error Threshold		Algorithms			
		Local Variamces	Propsoed (1*4.7%)	Proposed (1.5*4.7%)	Proposed (2*4.7%)
36	Time (s)	1172	591	861	1122
	PSNR (dB)	34.47	34.35	34.44	34.50
	CR	11.74	11.49	11.86	12.10
49	Time(s)	1021	516	751	977
	PSNR (dB)	33.68	33.61	33.70	33.75
	CR	14.57	15.24	14.73	15.06
64	Time(s)	921	467	680	886
	PSNR (dB)	33.02	32.99	33.10	33.16
	CR	17.40	16.93	17.46	17.80
81	Time(s)	850	433	633	824
	PSNR (dB)	32.41	32.40	32.49	32.58
	CR	19.87	19.24	19.80	20.16
100	Time(s)	788	402	589	769
	PSNR (dB	31.87	31.84	31.94	32.01
	CR	22.77	22.13	22.77	23.08

(b)Peppers of size 512×512

Error Threshold		Algorithms			
		Local Variamces	Propsoed (1*4.7%)	Proposed (1.5*4.7%)	Proposed (2*4.7%)
36	Time (s)	1472	739	1087	1441
	PSNR (dB)	34.84	34.76	34.95	35.12
	CR	7.93	7.81	7.90	7.99
49	Time(s)	1389	699	1023	1352
	PSNR (dB)	34.26	34.19	34.34	34.50
	CR	8.68	8.51	8.69	8.81
64	Time(s)	1285	649	951	1253
	PSNR (dB)	33.37	33.42	33.57	33.71
	CR	9.83	9.58	9.79	9.96
81	Time(s)	1202	606	887	1160
	PSNR (dB)	32.73	32.77	32.90	32.99
	CR	11.01	10.75	11.04	11.31
100	Time(s)	1108	560	821	1072
	PSNR (dB	31.95	32.01	32.09	32.18
	CR	12.62	12.23	12.62	12.86

(c) Boat of size 512×512

Error Threshold		Algorithms			
		Local Variamces	Propsoed (1*4.7%)	Proposed (1.5*4.7%)	Proposed (2*4.7%)
36	Time (s)	1668	849	1239	1620
	PSNR (dB)	33.71	33.65	33.78	33.89
	CR	6.78	6.66	6.80	6.88
49	Time(s)	1466	748	1094	1425
	PSNR (dB)	32.78	32.79	32.90	33.00
	CR	8.17	8.00	8.14	8.26
64	Time(s)	1303	671	973	1266
	PSNR (dB)	31.94	32.02	32.11	32.18
	CR	9.75	9.47	9.69	9.83
81	Time(s)	1166	596	870	1130
	PSNR (dB)	31.27	31.30	31.41	31.46
	CR	11.68	11.38	11.62	11.84
100	Time(s)	1045	537	778	1017
	PSNR (dB	30.53	30.59	30.63	30.72
	CR	13.86	13.47	13.89	14.09

(d) Goldhill of size 512×512

4.4 Conclusions

In this chapter, an efficient windowing scheme for fractal image coding based on the local variances method is proposed. The original local variances method uses a symmetric search window to search for the best-matched domain block in the domain pool for each range block. In our method, a non-symmetric search window is used in such a way that windows cover those domain blocks where variances are higher than that of the range block are considered. Experimental results show that this new scheme can achieve better decoded image quality and further increase the compression ratio when compared to the local variances method with the same window size.

Chapter 5. A Fast Fractal Image Coding based on Kick-out and Zero Contrast Conditions

5.1 Introduction

The problem with fractal coding is the high computational complexity in its encoding process. Most of the encoding time is spent on finding the best matched domain block from a large domain pool to represent an input range block with respect to contrast and intensity offset, as well as the isometric transformations. The exhaustive search algorithm can obtain the optimal result by searching exhaustively all the blocks within the domain pool, but this process requires a high computational cost, which limits its practical application. To solve this problem, extensive research on fast fractal image encoding algorithms [11, 35-81, 119] has been carried out. In particular, Bani-Eqbal [119] proposed a tree search method, which devised an incremental procedure to bind the domain block pixels, and then arranged the domain blocks in a tree structure to direct the search. In [73], an adaptive search algorithm based on an adaptive necessary condition to reduce the computational complexity was proposed. However, these techniques can reduce the required computation only at the expense of additional memory and degradation of the reconstructed image quality. So, some efficient algorithms [68, 75, 85, 120] have been developed to alleviate the computation burden, while the image quality can be maintained at the level of that of the full search. However, these algorithms first need to transform the image blocks into the frequency domain. In other words, pre-processing is required.

In this chapter, we propose an efficient algorithm based on a single kick-out condition and zero contrast prediction, which can greatly reduce the required computation as compared to the exhaustive search, while maintaining the same reconstructed image quality. Our proposed kick-out condition can determine efficiently whether a domain block is a good representation of a range block, and so excessive computation can be avoided in the early stage. With zero contrast prediction, the computation involved is further reduced. In addition, the algorithm does not need any pre-processing and extra memory for its implementation. Our proposed approach can also be combined with other fast encoding methods, such as the DCT Inner Product [75] method, to further speed up the encoding time. Experimental results show that the runtime can be reduced by about 75% when our algorithm is combined with the DCT Inner Product algorithm [75].

This chapter is organized as follows. We present our new fractal image compression algorithm based on a kick-out condition and zero contrast prediction in Section 5.2. In Section 5.3, we compare the performance of our proposed fast algorithm with the full search, the adaptive search scheme [73], tree search scheme [119], DCT Inner Product algorithm [75], and our algorithm combined with the DCT algorithm. Finally, conclusions are given in Section 5.4.

5.2 Proposed Algorithm

The principal of our proposed algorithm is to bypass those domain blocks that satisfy a kick-out condition, so they will no longer be considered and no further computations will be needed. In our algorithm, we first convert the full search equation (2.13) from two parameters, i.e. the contrast s and the offset o, to a function which only contains the contrast s. Based on this formulation, we can successively eliminate the

search space in the domain pool and thus decreasing the computation required to compare a range block and a transformed domain block.

5.2.1 The Kick-out Condition

From equation (2.13), the error function can be further simplified as follows:

$$\begin{aligned} E(R,D) &= \left[\|R\|^2 - \tfrac{1}{k}\langle R,I\rangle^2\right] - \frac{\left[\langle R,D\rangle - \tfrac{1}{k}\langle R,I\rangle\langle D,I\rangle\right]^2}{\left[\|D\|^2 - \tfrac{1}{k}\langle D,I\rangle^2\right]} \\ &= \left[\|R\|^2 - \tfrac{1}{k}\langle R,I\rangle^2\right] - \frac{\left[\langle R,D\rangle - \tfrac{1}{k}\langle R,I\rangle\langle D,I\rangle\right]^2}{\left[\|D\|^2 - \tfrac{1}{k}\langle D,I\rangle^2\right]^2}\left[\|D\|^2 - \tfrac{1}{k}\langle D,I\rangle^2\right] \end{aligned} \tag{5.1}$$

As $s = \dfrac{\langle R,D\rangle - \frac{1}{k}\langle R,I\rangle\langle D,I\rangle}{\|D\|^2 - \frac{1}{k}\langle D,I\rangle^2}$, the error function $E(R,D)$ can be written as follows:

$$E(R,D) = A - s^2B \tag{5.2}$$

where $A = \|R\|^2 - \frac{1}{k}\langle R,I\rangle^2$ and $B = \|D\|^2 - \frac{1}{k}\langle D,I\rangle^2$.

The coefficient s is limited to the range (-1,1) to ensure convergence in the decoding process. If A is greater than B, the maximum error occurs when s=0 while minimum error occurs when s=1. Therefore, we have:

If A-$B \geq 0$, then

(1). The maximum error occurs when s=0,

$$E_{\max} = A - s^2B = A \tag{5.3}$$

(2). The minimum error occurs when s=±1,

$$E_{\min} = A - s^2B = A - B \tag{5.4}$$

This means that, in finding the best matched domain block, the search is performed only if the minimum error $E_{\min}$ for the domain block under consideration is less than the current minimum error $d_{\min}$. Thus, we propose the kick-out condition as follows:

$$E_{\min} = A - B \geq d_{\min}. \qquad (5.5)$$

Based on (5.5), we propose a fast search algorithm which can reject dissimilar domain blocks efficiently for a given range block. In our algorithm, we select the domain blocks from left to right and top to bottom. The matching errors of the first domain block D_1 with each of the eight isometric operations are calculated. The one with the minimum error is considered to be the initial best matched domain block. The current minimum distance $d_{\min}$ is then set to this minimum distortion, say $E(R,D_1)$, and the search proceeds in a raster scan order. To determine whether the next candidate domain block D_2 is closer to R than the current best match D_1, we compute $E_{\min}(R,D_2)$ and compare it to $d_{\min}$. If $E_{\min}(R,D_2)$ is larger than or equal to $d_{\min}$, it also means that the condition $E(R,D_2) \geq d_{\min}$ is always guaranteed. The domain block D_2 is therefore rejected. Otherwise, the actual distortion $E(R,D_2)$ is calculated for the domain block D_2 with the eight isometric operations, and is compared to $d_{\min}$. If all the matching errors between the transformed domain blocks of D_2 and R are larger than the current minimum error $d_{\min}$, D_2 is rejected for the same reason mentioned above. Otherwise, $d_{\min}$ is replaced by $E(R,D_2)$ and the current best matched domain block is set to D_2 with the corresponding isometric operation. This process is repeated for all the domain blocks D_i in the domain pool to find the best matched one for an input range block. Based on this kick-out condition, the required computation for searching the best matched domain block will be greatly reduced.

5.2.2 Fast Error Calculation using Zero Contrast Prediction

In the implementation, the contrast factor s is encoded using 5 bits. Therefore, any value of the contrast s falling within (–0.03125, 0.03125) will be set to zero after

quantization. With (5.2), as $|s|<1$, this means that the zero contrast condition will happen only when $A<B$:

$$E(R,D) = A - s^2 B \geq 0$$

or
$$\sqrt{\frac{A}{B}} \geq |s|. \tag{5.6}$$

The contrast factor s is quantized to 0 if the absolute value of s is less than 0.03125, i.e.:

$$0.03125 > \sqrt{\frac{A}{B}} \geq |s|. \tag{5.7}$$

When s is set to zero, the corresponding error is given as follows:

$$E(R,D)=A. \tag{5.8}$$

In this case, the range-domain block matching error can be obtained without performing any calculation, and their errors can be represented by the constant value A.

5.2.3 Combining Other Approaches

Our proposed algorithm can combine with other fast fractal algorithms to further improve their speed. One example is the DCT Inner Product [75] approach, which allows the computation of two inner products only for the eight isometric operations of a domain block. However, the whole domain pool still has to be considered in order to obtain the best matched domain block. This DCT approach can combine with our algorithm to further improve its speed. In encoding an image, the single kick-out condition (5.5) will be checked to reject those dissimilar domain blocks. Zero contrast prediction (5.7) is then used to determine whether the contrast factor is zero or not, and the corresponding error function can be computed without performing the range-domain block matching. Therefore, the required runtime for the algorithm can be further reduced.

5.3 Experimental Results

In the experiments, we first compared the performance of our proposed schemes with the full search scheme. A three-level quadtree partition scheme with range block sizes of 4×4, 8×8 and 16×16 pixels, and a search grid of one are used. Three popular 512×512 images, Lena, Boat and Goldhill, are used to evaluate the performance of the proposed algorithm and other algorithms. The computer used is a Pentium III 500MHz. The runtimes (in seconds) for our proposed algorithm and the full search are listed in Table 5.1. We measured the runtimes of our algorithm based on (i) the single kick-out condition (i.e. case 1), (ii) the zero contrast prediction (i.e. case 2), and (iii) the combination of both conditions (i.e. cases 1 and 2). Experimental results show that, considering both conditions, our proposed algorithm can reduce the required computation by about 50% as compared to the exhaustive search method. In other words, a large number of domain blocks are rejected for performing the range-domain block matching by the kick-out condition, and a number of the error functions are obtained based on the zero contrast prediction.

The required computation of our algorithm is a function of the number of range-domain block matching removed by the kick-out condition, and the number of error functions computed by zero contrast prediction. However, these numbers are image-dependent. Table 5.2 tabulates the percentages of the kick-out condition and zero contrast prediction that occurred in encoding the three images. Experimental results show that about 50% of the domain blocks are rejected by the kick-out condition, while 6% of the remaining range-domain block matching can use zero contrast prediction to compute the corresponding error functions. In (5.2), the kick-out condition has not considered the

effect of quantizing the luminance offset, so we cannot guarantee that the optimal domain block will be obtained. Therefore, in order to obtain the best domain block for representing a range block, we set a tolerance of 10% more of the current minimum error $d_{\min}$ when considering whether a domain block is to be rejected or not. With this setting, we found that the reconstructed image quality will be equal to that of the exhaustive search.

The performance of our algorithm combined with the DCT Inner Product approach was also investigated. The size of the range blocks is set to 8×8 only. We combined the DCT approach with the single kick-out condition and zero contrast prediction. These combined algorithms were compared with the baseline method and the DCT Inner Product method in terms of the encoding time and PSNR. The experimental results are tabulated in Table 3, which shows that the runtime of the combined algorithm is about 25% of the baseline approach and 50% of the DCT approach. Furthermore, the PSNR based on our algorithm is the same as that of the baseline method.

Our algorithm was also compared to the adaptive search algorithm [73] and the tree search algorithm [119]. In [75], 2 bits and 6 bits were used to represent the contrast scaling factor and the range mean. In our algorithm, the corresponding numbers of bits used for these two factors are 5 bits and 7 bits, respectively. Experimental results show that the adaptive search scheme suffers from a significant drop in PSNR at a high speedup, although its compression ratio is slightly higher due to a smaller number of bits being used to represent the two factors. In our proposed algorithm, only those domain blocks satisfying the condition $A–B<d_{\min}$ will be considered as candidate blocks. From Table 5.5, our proposed algorithm can achieve a speedup of 1.9 without any loss of image

quality when compared to the conventional full search scheme. In addition, the encoding time required by our algorithm can be further reduced by limiting the search range in the domain pool. If these domain blocks satisfying the condition $0<A–B<d_{min}$ are considered as candidate blocks, the speedup of our algorithm will be significantly increased. From Tables 5.4 and 5.5, we can see that our algorithm can achieve a speedup of 15.2 with a PSNR drop of 1.17dB. At a comparable speedup of 16.25, the PSNR drop based on [73] is 1.75dB.

Tables 5.6 and 5.7 illustrate the performances of the tree search algorithm [119] and our algorithm. When the speedup is 25, the corresponding relative drops in PSNR are 4.2% and 4%. At a higher speedup of 58, the relative drops become 7.18% and 7.88%, respectively. Apparently, the tree search algorithm shows a slightly better performance level at a very high speedup. However, the additional memory requirement of our algorithm is small, and it can achieve the same PSNR and compression ratio (CR) as the full search with a speedup of 1.8. According to Table 6 in [119], the tree search algorithm has a large memory requirement. To encode an image of size 256×256, 16MB of memory space will be required to store the leave nodes of the tree for range block size of 4×4 with eight isometric operations and 2×2 domain grid, and 1.5MB will be required if pixel average is considered. The experimental results shown in Tables 5.6 and 5.7 were obtained without using the quadtree partitioning scheme, which is normally used in fractal image coding. If the quadtree partitioning scheme is adopted, the memory requirement of the tree search algorithm will be increased significantly.

5.4 Conclusions

In this chapter, we have proposed a single kick-out condition and the use of zero contrast prediction to speed up the encoding process. The efficiency of the kick-out condition depends on how quickly the global minimum error is detected. Once this global error is found, most of the remaining domain blocks will be rejected and range-domain block matching will not be performed. Experimental results show that the runtime of our algorithm is about 50% of the exhaustive search. Our algorithm can also be combined with other fast fractal coding algorithms, such as the DCT Inner Product, to further improve the speed. The combined algorithm can reduce the required computation by about 75% as compared to the baseline approach. In addition, our algorithm was also compared with an adaptive search algorithm and a tree search algorithm.

Table 5-1 Comparison of the coding results using domain grid of one and three level quadtree partitioning (16×16, 8×8 and 4×4).

Test Images		Algorithms	
		Full Search	Proposed Algorithm (Case 1 & 2)
Lena	Time (s)	5940	3242
	PSNR (dB)	34.91	34.91
	CR	15.50	15.50
Boat	Time(s)	8463	3666
	PSNR (dB)	34.91	34.91
	CR	9.71	9.71
Goldhill	Time(s)	8976	5755
	PSNR (dB)	33.36	33.36
	CR	9.11	9.11

Table 5-2 comparison of computational complexity for the full search and our proposed algorithm.

	Lena	Boat	Goldhill
Full Search	100%	100%	100%
Kick-out	50%	54%	45%
Zero Contrast	3%	3%	2%

Table 5-3 Comparison of the coding results using domain grid of one and 8×8 range block.

Test Images		Algorithms			
		Baseline	DCT	DCT + Proposed Algorithm case (1)	DCT + Proposed Algorithm cases (1) & (2)
Lena	Time (s)	5340	2473	1540	1342
	PSNR (dB)	31.14	31.14	31.14	31.14
Boat	Time (s)	5340	2473	1571	1376
	PSNR (dB)	29.21	29.21	29.21	29.21
Goldhill	Time (s)	5340	2473	1625	1501
	PSNR (dB)	29.68	29.68	29.68	29.68

Table 5-4 Adaptive search algorithm [75]: the encoding results using the test image lena of size 256×256 with range block size 4×4 and domain grid 4×4. The CPU is PII233MHz.

T_0	T_1	Encoding Time (sec)	PSNR(dB)	Speedup	PSNR Drop(dB)
Full Search		65	31.01	1	0
0	*std* (*R*)/2	26	31.00	2.5	-0.01
0	*std* (*R*)/4	20	30.79	3.25	-0.22
0	*std* (*R*)/8	13	30.28	5	-0.73
3	*std* (*R*)/8	7	30.14	9.28	-0.87
3	*std* (*R*)/16	5	29.36	13	-1.75
3	*std* (*R*)/32	4	28.15	16.25	-2.86

Table 5-5 Our proposed algorithm: the encoding results using the test image lena of size 256×256 with range block size 4×4 and domain grid 4×4. The CPU is PIII 500MHz.

	Encoding Time (sec)	PSNR(dB)	Speedup	PSNR Drop(dB)
Full Search	38	31.70	1	0
A-B<d_{min}	20	31.70	1.9	0
0<A-B<d_{min}	2.5	30.53	15.2	-1.17
0<A-<0.4*d_{min}	1.2	29.93	31.7	-1.77

Table 5-6 Tree search algorithm [119]: the encoding results using the test image lena of size 256×256 with range block size 4×4 and domain grid 2×2. The machine is sun sparcstation 10 model 30

Lena	PSNR(dB)	Time(sec)	Speedup	PSNR Drop(dB)
Full Serach	31.61	8750	1	0
Tree β=20	30.29	340	25	-1.32
Tree β=100	29.34	150	58	-2.27

Table 5-7 Our proposed algorithm: the encoding using the test image lena of size 256×256 with range block size 4×4 and domain grid 2×2. The CPU is PIII 500MHz.

Lena	PSNR(dB)	Time(sec)	Speedup	PSNR Drop(dB)
Full Serach	33.52	586	1	0
A-B<d_{min}	33.52	322	1.8	0
0<A-<0.6*d_{min}	32.17	23	25	-1.35
0<A-<0.1*d_{min}	30.88	10	58	-2.64

Chapter 6. Fast Fractal Image Coding Using Geometric Inequality

6.1 Introduction

Fractal image compression has recently attracted considerable attention, as it can provide a high compression ratio with an extremely fast decoding process.In fractal image coding scheme, equation (2.13) can be further simplified by removing the mean value of each range block and each transformed domain block. Therefore, equation (2.13) is converted from a two-parameter function into a function that only involves the contrast parameter for optimization. The error function between the range block and the transformed domain block can be rewritten as follows:

$$E\left(R, D_i\right) = \left\|R - \overline{r}I\right\|^2 - s_i^{\,2}\left\|D_i - \overline{d}_i I\right\|^2, \tag{6.1}$$

where $\overline{r}$ and $\overline{d}_i$ are the mean value intensity of the range block and the contracted domain block, respectively. $\|\ \|$ is the two-norm and $\langle .,. \rangle$ represents the inner product.

The application of the full search fractal image encoding is limited because of its high computational cost, but a high compression ratio can be achieved. Therefore, many fast fractal image-coding algorithms [11, 35-81, 119, 120] have been developed for reducing the computational cost. In this chapter, a new algorithm is presented which can significantly reduce the computations. This new approach uses a set of reference vectors to confine the searching space. Our algorithm can progressively reduce the searching space during the matching process and so can save more computations while the

corresponding best-matched domain blocks are obtained. Experimental results demonstrate that the encoding time of the new algorithm can be reduced significantly, while the encoding quality remains the same as the full-search scheme.

This chapter is organized as follows. A new algorithm will be presented which can significantly reduce the computations. This new approach uses a set of reference vectors to confine the searching space. Our algorithm can progressively reduce the searching space during the matching process and so can save more computations while the corresponding best-matched domain blocks are obtained. Experimental results demonstrate that the encoding time of the new algorithm can be reduced significantly, while the encoding quality remains the same as the full-search scheme.

6.2 The Proposed Algorithm

In this chapter, an inequality is set up to determine whether or not a range-domain block matching is required by utilizing the property of the "sine law". In our approach, both the range block and the domain block are viewed as a vector. Using the relationship between inner product and the cosine of the angle between the two vectors, we have:

$$\left\langle R-\bar{r}I, D-\bar{d}I \right\rangle = \left\| R-\bar{r}I \right\| \left\| D-\bar{d}I \right\| \cos\theta \ , \ \text{where } 0 \le \theta \le \pi. \qquad (6.2)$$

As shown in Figure 6.1, θ is the angle between the range block vector and the domain block vector. We can express the matching distortion in terms of the range block variance and the sinusoidal function. Equation (6.1) can be rewritten as:

$$E\left(R, D_i\right) = \left\| R-\bar{r}I \right\|^2 \sin^2\theta \ , \ \text{where } 0 \le \theta \le \pi/2. \qquad (6.3)$$

It is known that the period of $\sin^2\theta$ is π, and we consider its range between 0 and π. According to the supplement identity $\sin(\pi-\theta) = \sin\theta$, this implies that $\sin^2(\pi-\theta) = \sin^2\theta$, and the graph of the function $\sin^2\theta$ is symmetrical at the angle $\pi/2$. Therefore, we can consider the interval from 0 to $\pi/2$ only for the function $\sin^2\theta$.

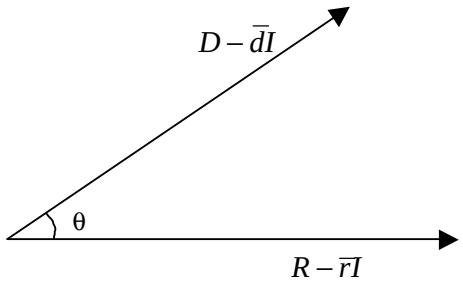


Figure 6-1. The angle between a range block and a domain block.

Figure 6.2 illustrates the effect of $\sin^2\theta$ on the range-domain block matching error in equation (6.3). The points P and Q represent the maximum and the minimum matching errors, respectively. With the derived result, we propose an efficient fractal coding algorithm by using the sine law identity. The objective of the fractal coder is to find the best-matched domain block to an input range block with respect to the contrast scaling *s* and offset *o* parameters, as well as its isometric operations, in the domain pool for representing the range block. In this section, we will establish an inequality based on equation (6.3) to reduce the search space within the domain pool, while the optimal domain block for each range block can still be found.

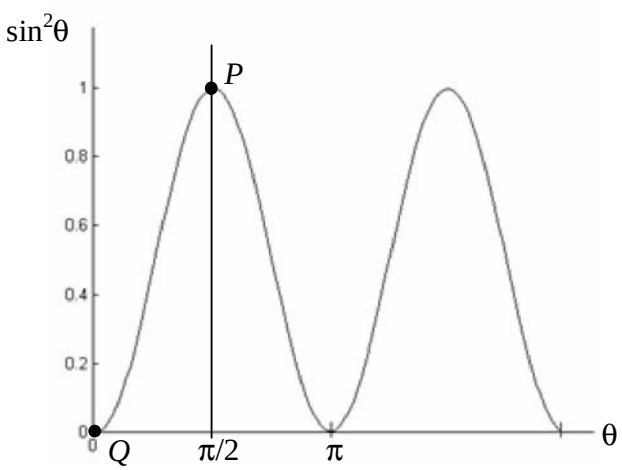


Figure 6-2 P and Q denote the maximum and minimum range-domain block matching errors, respectively.

At first, a reference domain block *A* is selected arbitrarily from the domain pool. Let θ, θ_1 and θ_2 be the angles between $R-\bar{r}I$ and $D_i-\bar{d}_iI$, *A* and $D_i-\bar{d}_iI$, and *A* and $R-\bar{r}I$, respectively, where D_i and $\bar{d}_i$ represent the domain block under consideration and its corresponding mean value. The trihedral theorem [23] states that "of the three plane angles that form a trihedral angle, any of two add together greater the third". According to the statement, we have $\theta+\theta_1\geq\theta_2$ and $\theta+\theta_2\geq\theta_1$ hold. It implies that $\theta\geq|\theta_1-\theta_2|$, i.e. the angle θ is bounded by $|\theta_1-\theta_2|$. As $\sin^2\theta$ is a monotonic increasing function between $0\leq\theta\leq\pi/2$, so $\sin\theta\geq\sin(\theta_1-\theta_2)$ and $\sin^2\theta\geq\sin^2(\theta_1-\theta_2)$. The minimum range-domain block matching error ($E^i_{\min}$) can be definded as follows:

$$E^i_{\min}=\left\|R-\bar{r}I\right\|^2\sin^2(\theta_1-\theta_2) \tag{6.4}$$

By expanding $\sin(\theta_1-\theta_2)$,

$$\sin(\theta_1-\theta_2)=\sin\theta_1\cos\theta_2-\cos\theta_1\sin\theta_2, \tag{6.5}$$

The minimum range-domain block matching error ($E_{\min}^{i}$) can be written as:

$$E_{\min}^{i} = \left\| R - \bar{r}I \right\|^2 \left(\sin\theta_1 \cos\theta_2 - \cos\theta_1 \sin\theta_2 \right)^2 . \qquad (6.6)$$

This $E_{\min}^{i}$ represents the minimum range-domain block matching error for the domain block D_i. In the range-domain block matching process, the range block is compared to each of the domain blocks one by one. If $E_{\min}^{i}$ is smaller than the current minimum range-domain block error, $E_{\min}$, of those domain blocks that have been compared, the exact error for the D_i will be computed. If this error is less than $E_{\min}$, $E_{\min}$ will be replaced by this new error. However, if $E_{\min}^{i}$ is higher than $E_{\min}$, D_i can be eliminated since it cannot be closer to R than the "so far" nearest domain block.

To speed up the algorithm, a pre-processing step, as shown in Figure 6.3, is performed. A reference vector (A) is selected arbitrarily in the domain pool. Prior to the encoding process, the angle between the reference block A and each domain block is calculated, and the corresponding $\sin\theta_1$ and $\cos\theta_1$ are stored. These tables for $\sin\theta_1$ and $\cos\theta_1$ will be used repeatedly during encoding. For an input range block R, the corresponding $\sin\theta_2$ and $\cos\theta_2$ between R and the reference block A are calculated, and the computation of $\left\| R - \bar{r}I \right\|^2 \left(\sin\theta_1 \cos\theta_2 - \cos\theta_1 \sin\theta_2 \right)^2$ is therefore simple.

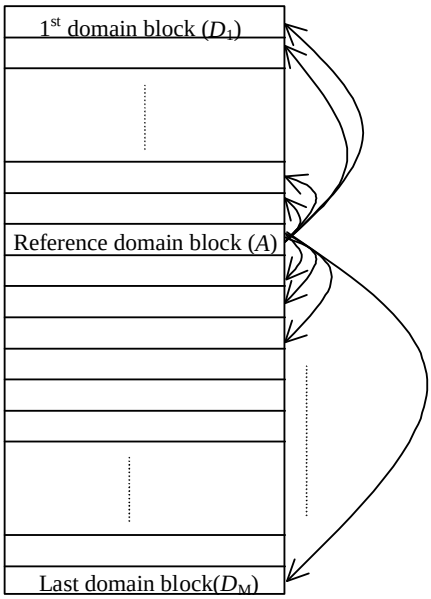


Figure 6-3. Pre-processing process.

Based on (6.6), candidates in the domain pool can be rejected progressively during the encoding process. Our proposed algorithm can speed up the encoding process of finding the best-matched domain block by eliminating impossible candidate domain blocks before their actual errors are calculated. If a domain block has a minimum error that exceeds the current minimum value of $E_{\min}$, the domain block cannot be the best-matched domain block, and the actual error need not be calculated. This encoding scheme is efficient, and can achieve the same image quality as conventional full-search fractal image coding. More reference vectors can be used to further improve the encoding speed, but at the expense of more memory. With M reference vectors and N domain blocks in the domain pool, the additional memory requirement is $2NM$.

6.3 Experimental Results

In the experiments, a three-level quadtree partitioning scheme with range block sizes of 4×4, 8×8 and 16×16 pixels, and a search grid of two are used. We evaluated the performance of our fast algorithm based on four 512×512 gray scale images: Lena, Peppers, Boat and Goldhill. The experiments were conducted on a Pentium III 500 processor with 256MB memory.

With a minimum error $E_{\min}$, the encoding speed can be further increased if a weighting parameter λ (≥ 1) is used in $E(R, D_i) \geq \lambda E_{\min}$. With a larger value of λ, more domain blocks will be rejected in the encoding. For the test image Lena of size 512×512, when λ=2, the speedup of the encoding process is 2.48 when compared to the full search. When λ=6, our proposed algorithm gives a speedup of 6.34 in the encoding process with almost the same reconstructed image quality as the full-search fractal image coding. From the simulation results shown in Figure 6.4, we can observe that the speedup becomes steady when more than 10 reference vectors are used for the domain grid of 2 with the three-level quadtree partitioning scheme. The computational requirement of the proposed algorithm using 10 reference vectors is reduced substantially, compared to that using only one reference vector. The percentage of reduction in encoding time increases from 17% to 60% when the number of reference vectors used is increased from 1 to 10. The PSNR and compression ratio (CR) are the same as that of the conventional exhaustive search approach. In order to reject the dissimilar domain block candidates in the domain pool efficiently, the distance within these reference vectors should be as far as possible and the searching space can be further reduced significantly.

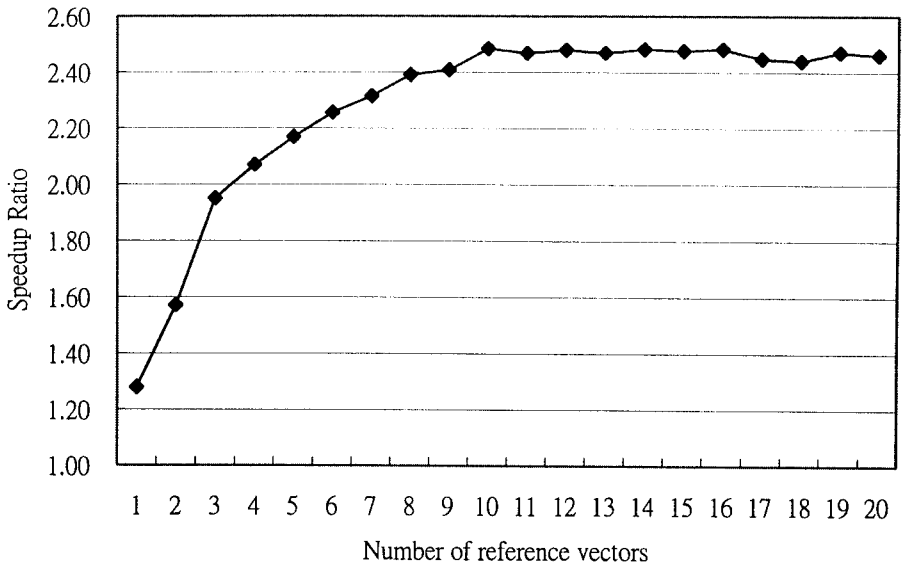


Figure 6-4. Number of reference vectors versus the encoding speedup for our proposed algorithm with λ=2 using the image Lena of size 512×512.

As shown in Table 6.1, the computational cost of the proposed algorithm can be reduced substantially for all the test images when compared to the exhaustive search. Our proposed algorithm requires less than 50% of the average computation of the conventional full search algorithm when λ is set at 2 and 10 reference vectors are used. If a larger value of λ is used, the search space will be further reduced. When λ=6, our proposed algorithm can achieve an average speedup of 5 in the encoding process with almost the same result as the conventional fractal image coding. In this case, our algorithm is able to eliminate about 80% of domain blocks from consideration.

Table 6-1 Comparison of the coding results with different parameters using (a) Lena, (b) Peppers, (c) Boat and (d) Goldhill of size 512×512, domain grid of two and three level quadtree partitioning (16×16, 8×8 and 4×4).

	PSNR(dB)	CR	Time(s)	Speedup	PSNR Loss	CR Loss
F. S.	32.74	22.31	2303	1.00	0.00	0.00
λ=1	32.74	22.31	1427	1.61	0.00	0.00
λ=2	32.74	22.31	927	2.48	0.00	0.00
λ=3	32.73	22.27	665	3.46	0.01	0.04
λ=4	32.72	22.19	516	4.47	0.01	0.13
λ=5	32.70	22.12	423	5.44	0.04	0.19
λ=6	32.68	21.98	363	6.34	0.05	0.34
λ=7	32.63	21.87	320	7.20	0.11	0.44
λ=8	32.64	21.77	288	7.99	0.10	0.54
λ=9	32.58	21.69	265	8.70	0.16	0.62
λ=10	32.56	21.57	247	9.32	0.17	0.74

(a)

	PSNR(dB)	CR	Time(s)	Speedup	PSNR Loss	CR Loss
F. S.	32.22	25.07	2207	1.00	0.00	0.00
λ=1	32.22	25.07	1371	1.61	0.00	0.00
λ=2	32.22	25.07	911	2.42	0.00	0.00
λ=3	32.22	25.02	664	3.32	0.00	0.05
λ=4	32.21	24.89	516	4.28	0.01	0.19
λ=5	32.19	24.78	422	5.23	0.04	0.29
λ=6	32.18	24.62	358	6.17	0.04	0.45
λ=7	32.15	24.49	312	7.07	0.08	0.58
λ=8	32.16	24.31	280	7.88	0.07	0.76
λ=9	32.18	24.16	254	8.69	0.05	0.91
λ=10	32.15	23.92	235	9.38	0.08	1.15

(b)

	PSNR(dB)	CR	Time(s)	Speedup	PSNR Loss	CR Loss
F. S.	32.48	14.07	3101	1	0.00	0.00
λ=1	32.48	14.07	2393	1.30	0.00	0.00
λ=2	32.48	14.07	1715	1.81	0.00	0.00
λ=3	32.47	14.05	1347	2.30	0.01	0.02
λ=4	32.46	14.03	1111	2.79	0.02	0.04
λ=5	32.43	13.96	945	3.28	0.05	0.11
λ=6	32.41	13.93	824	3.76	0.07	0.13
λ=7	32.38	13.88	730	4.25	0.10	0.19
λ=8	32.31	13.83	657	4.72	0.17	0.24
λ=9	32.27	13.75	600	5.17	0.21	0.32
λ=10	32.24	13.69	554	5.60	0.24	0.38

(c)

	PSNR(dB)	CR	Time(s)	Speedup	PSNR Loss	CR Loss
F. S.	30.93	15.46	2898	1.00	0.00	0.00
λ=1	30.93	15.46	2390	1.21	0.00	0.00
λ=2	30.93	15.46	1713	1.69	0.00	0.00
λ=3	30.92	15.45	1326	2.18	0.01	0.01
λ=4	30.92	15.38	1085	2.67	0.01	0.08
λ=5	30.90	15.36	922	3.14	0.03	0.10
λ=6	30.87	15.32	802	3.61	0.06	0.14
λ=7	30.84	15.29	714	4.06	0.09	0.17
λ=8	30.79	15.22	647	4.48	0.14	0.24
λ=9	30.76	15.13	593	4.88	0.17	0.33
λ=10	30.72	15.05	550	5.27	0.21	0.41

(d)

6.4 Conclusions

A fast full-search fractal image-coding algorithm is proposed in this chapter. Our algorithm is based on the relationship between the range-domain block matching error and the angle between the range and domain block vectors. An inequality in terms of the sine of the angle has been proposed. This inequality can reject impossible domain block candidates efficiently in searching for the best-matched domain block. Based on the experimental results, our algorithm can reduce more than 80% of the required computations, as compared to the full-search algorithm, with only a slight quality degradation in the reconstructed image.

6.5 A Summary of Our Proposed Fractal Image Coding Techniques

Four efficient fractal image coding algorithms have been proposed. The first algorithm is based on new feature vectors and the property of zero contrast. The proposed feature vectors can provide a better representation of image blocks, and thus result in a more efficient search of the domain block using the *k*-d tree scheme. The *k*-d tree is a data structure capable of efficiently indexing a multi-dimensional data space. Using *k*-d tree structure, the encoding time can be improved while the reconstructed image quality remains relatively constant. The main drawback of this coding technique is the huge amount of memory needed to construct the *k*-d tree structure for storing the domain blocks. Additional memory of $(n_r \times n_r)N_D$ bytes is required to store the pixel intensity for each normalized domain block in the domain pool.

The second algorithm is an efficient windowing scheme for fractal image coding based on the local variances method. In this method, windows covering those domain blocks whose variances are higher than that of the range block are considered according to a mathematical model. In other words, only those domain blocks with variances higher than that of the range block will be considered when searching for the best matched range block, instead of considering both lower and higher variances. However, there are still lots of best-matched domain blocks distributed far away from the searching window, so that this windowing approach cannot achieve a high image quality when compared to the full search method. Moreover, additional memory of N_D bytes is required to store the variance of each domain block in the domain pool.

The exhaustive search algorithm can obtain the optimal result by searching all the blocks within the domain pool, but this process requires a high computational cost, which limits its practical application. We propose an efficient algorithm based on a single kick-out condition and zero contrast prediction, which can greatly reduce the required computation as compared to the exhaustive search, while maintaining the same reconstructed image quality. In addition, the algorithm does not need any pre-processing and extra memory for its implementation. Experimental results show that the runtime of our algorithm is about 50% of the exhaustive search. Our algorithm can also be combined with other fast fractal coding algorithms, such as the DCT Inner Product, to further improve the speed. The combined algorithm can reduce the required computation by about 75% as compared to the baseline approach.

The fourth method is another fast full search fractal image-coding algorithm, which uses the angle between the two vectors representing the input range block and a

reference domain block to determine a tighter decision boundary for eliminating the searching space in the domain pool. The encoding time can be further decreased when more reference domain vectors are used. Our algorithm can reduce more than 80% of the required computations, as compared to the full-search algorithm, with only a slight quality degradation in the reconstructed image. With M reference vectors and N_D domain blocks in the domain pool, the additional memory requirement is $2N_DM$ bytes.

These efficient algorithms have been further investigated to extend their applicability to image magnification and image denoising as described in Chapter 7 and Chapter 8.

Chapter 7. An Efficient Image Magnification Algorithm based on Iterated Function Systems

7.1 Introduction

Digital image magnification techniques have an increasing demand for developing product applications, such as for digital cameras. A magnification algorithm produces an output image that has a greater size than that of the input image while preserving as much visual content of the original image as possible. Enhancing the resolution of an image is complicated due to the localized high-frequency nature of the pixel intensities across edges. Conventionally, the amplitude of an unknown or missing pixel in a high-resolution image is estimated from the corresponding neighboring pixels in its low-resolution version. The bilinear and bicubic spline interpolation methods are the simplest forms in this class. Unfortunately, these methods are unable to capture sudden changes around edges, so visually unacceptable artifacts such as blurring or ringing will appear. Various techniques have been proposed to improve the visual quality of magnified images.

Fractal image magnification is one of the techniques with the most potential, which can achieve a high-resolution enhancement for the sharpness across edges. This is because the fractal approach yields resolution independent of the image decompression process. After fractal image coding, the output of the coder is an IFS, which approximates the image as a fixed point of a contractive transformation. This IFS code

can be used to reconstruct the image at any level of resolution. These magnified images are suitable for graphics systems that are typically composed of devices of different resolutions. However, the conventional fractal image magnification method [34] causes serious blocky artifacts due to the independent and lossy coding of the range blocks. A number of fractal image magnification approaches have been proposed [123, 125-127, 129, 131, 132, 160]. Chung *et al.* [127] have proposed to use an additional enhancement layer (IEUF) to produce a magnified image without any blocky artifacts. However, that method cannot enhance the high frequency components efficiently. In this chapter, we propose an efficient algorithm based on the fractal properties for image magnification to deal with this problem. In order to speed up the encoding process, the *k*-d tree structure is used to direct the search. This can lead to a significant improvement in encoding time while maintaining comparable magnified image quality.

This chapter is organized as follows. We will present the details of our proposed image magnification algorithm based on the fractal properties in Section 7.2. In Section 7.3, we compare the performance of our proposed algorithm with the bicubic spline interpolation, conventional fractal image magnification [34], and IEUF [127] methods. Finally, conclusions are given in Section 7.4.

7.2 Our Proposed Algorithms

7.2.1 Adaptive Overlapped Range Block Partitioning

An efficient image magnification algorithm based on the Iterated Function System with error compensation is proposed in this chapter. Our proposed algorithm can not only

maintain the high frequency information to preserve the edges, but can also effectively remove blocky artifacts from the magnified images. In a fractal image coding scheme, Reusens [82] uses an overlapped range block partitioning scheme to construct an image with good visual quality. In this section, we describe an extension of the overlapped range block technique to image magnification.

In order to preserve the visual details in a magnified image without introducing visible artifacts, the original low-resolution image f_{orig} of size $m_x \times m_y$, to be magnified, is partitioned into a set of overlapping sub-images. Let $f_{(i,j)}$ denote an image whose non-overlapping range blocks are extracted from the original image f_{orig} starting at location (i, j), i.e.

$$f_{(i,j)} = \mathop{\mathbf{r}}_{p=0}^{\left(N_{R_x}^i - 1\right)} \mathop{\mathbf{r}}_{q=0}^{\left(N_{R_y}^j - 1\right)} \left[R\left(pn_r + i, qn_r + j \right) \right], \qquad i, j = 0, \ldots, n_r - 1, \tag{7.1}$$

where $N_{R_x}^i = \left\lfloor \dfrac{m_x - i}{n_r} \right\rfloor$ and $N_{R_y}^j = \left\lfloor \dfrac{m_y - j}{n_r} \right\rfloor$, which represent the number of range blocks in the horizontal and vertical directions, respectively. In other words, there are $n_r \times n_r$ possible images for $f_{(i,j)}$, which are extracted from the original image f_{orig} from i to $\left[m_x - 1 - \left((n_r - i) \bmod n_r \right) \right]$ in the horizontal direction and from j to $\left[m_y - 1 - \left((n_r - j) \bmod n_r \right) \right]$ in the vertical direction. Hence, there are $N_{R_x}^i \times N_{R_y}^j$ range blocks, where $0 \le i, j \le n_r - 1$. Obviously, when i=0 and j=0, it is easily seen that $f_{(0,0)}$ is equivalent to the original image f_{orig}. Otherwise, $f_{(i,j)}$ is a sub-image of f_{orig}. Fractal

image coding is then applied to each of the images $f_{(i,j)}$ independently to obtain the corresponding IFS codes $\boldsymbol{\tau}_{(i,j)}$. The IFS codes can be represented as follows:

$$\boldsymbol{\tau}_{(i,j)} = \mathop{\mathbf{r}}_{p=0}^{\left(N_{R_x}^i - 1\right)} \mathop{\mathbf{r}}_{q=0}^{\left(N_{R_y}^j - 1\right)} \left[\tau\left(p,q\right)\right] \qquad i, j = 0, \ldots, n_r - 1 \tag{7.2}$$

To magnify an image with a factor α, each of the IFS codes $\boldsymbol{\tau}_{(i,j)}$ is applied to an arbitrary image where the sizes of the range blocks and domain blocks are increased from n_r to αn_r and from n_d to αn_d or $2\alpha n_r$, respectively. The magnified image $M_{(i,j)}$ obtained from $f_{(i,j)}$ is defined as follows:

$$M_{(i,j)} = \mathop{\mathbf{r}}_{p=0}^{\left(N_{R_x}^i - 1\right)} \mathop{\mathbf{r}}_{q=0}^{\left(N_{R_y}^j - 1\right)} \left[R\left(\alpha p n_r + \alpha i, \alpha q n_r + \alpha j\right)\right] \qquad i, j = 0, \ldots, n_r - 1 \tag{7.3}$$

From equation (7.3), we can see that $n_r \times n_r$ magnified images, $M_{(i,j)}$, are constructed based on the respective IFS codes $\boldsymbol{\tau}_{(i,j)}$. Figure 7.1 shows the example of a magnified image. Figure 7.1(a) shows the low-resolution image $f_{(3,1)}$ whose range blocks of size 4×4 are extracted from the original low-resolution image f_{orig} of size 256×256 at location (3,1), while Figure 7.1(b) is the corresponding magnified image $M_{(3,1)}$ with a magnification factor α=2.

(a) (b)

Figure 7-1 (a) A low-resolution image whose range blocks are extracted at the position (3,1), and (b) the corresponding magnified image $M_{(3,1)}$ with a range block size of 4×4.

7.2.2 Pixel Averaging

A set of magnified high-resolution images $M_{(i,j)}$ $(0 \le i, j \le n_r - 1)$ is constructed by iterating the corresponding IFS code in the decoding process. $M_{0,0}$ is equivalent to reconstructing an image by the conventional fractal image magnification algorithm. The image qualities of these *N* magnified images suffer from blocky artifacts and the loss of high-frequency details. In our proposed algorithm, an optimal high-resolution image M^{opt} is obtained by the following way:

$$M^{opt}(x,y) = \frac{1}{p}\sum_{s=0}^{p} M_s(x,y) \tag{7.4}$$

where p ($1 \leq p \leq k$) denotes the number of pixels to be averaged at a pixel location (x, y). The optimal pixel value at (x, y) in M^{opt} is obtained by averaging p pixels from the corresponding p of the magnified images $M_s = M_{(i,j)}$ where the index $s = i+j \cdot n_r$. The pixel value at (x, y) in each of the magnified image $M_{(i,j)}$ can be considered as a random variable with the mean equal to the optimal value at that position. Consequently, averaging the corresponding pixel values of the magnified images $M_{(i,j)}$ should produce an optimal magnified image. This should also give the best subjective and objective quality. From Figure 7.4, we can see that the PSNR of a magnified image increases dramatically when more corresponding pixels are averaged. An optimal magnified image M^{opt} with the highest PSNR is obtained by averaging k magnified images.

7.2.3 Error Compensation

The goal of this section is to present a method to further improve the visual quality of a magnified image generated as described in Section 7.2.2 by adding a residual image. In this error compensation scheme, the magnified high-resolution image M^{opt} constructed as described in Section 7.2.2 is down-scaled to the size of the original low-resolution image, which is then subtracted from the original image f_{orig} to give the residual image that contains the high-frequency details. This residual image is then magnified by using the bicubic spline interpolation method, and is added to the magnified image M^{opt} to form an image of better quality M^{opt+s}, i.e.

$$M^{opt+s} = M^{opt} + 2\uparrow\left[f_{orig} - 2\downarrow\left[M^{opt} \right] \right] \tag{7.5}$$

where $2\uparrow[.]$ and $2\downarrow$ [.] represent the 2× up- and down-scaling operations, respectively. This error compensation scheme can significantly enhance the subjective and objective qualities of the magnified image. In magnification of the residual image to a larger size, a search window of size 4×4 is used in order to deal with the misalignment issue during interpolation. Figure 7.2 illustrates the details of this error compensation process.

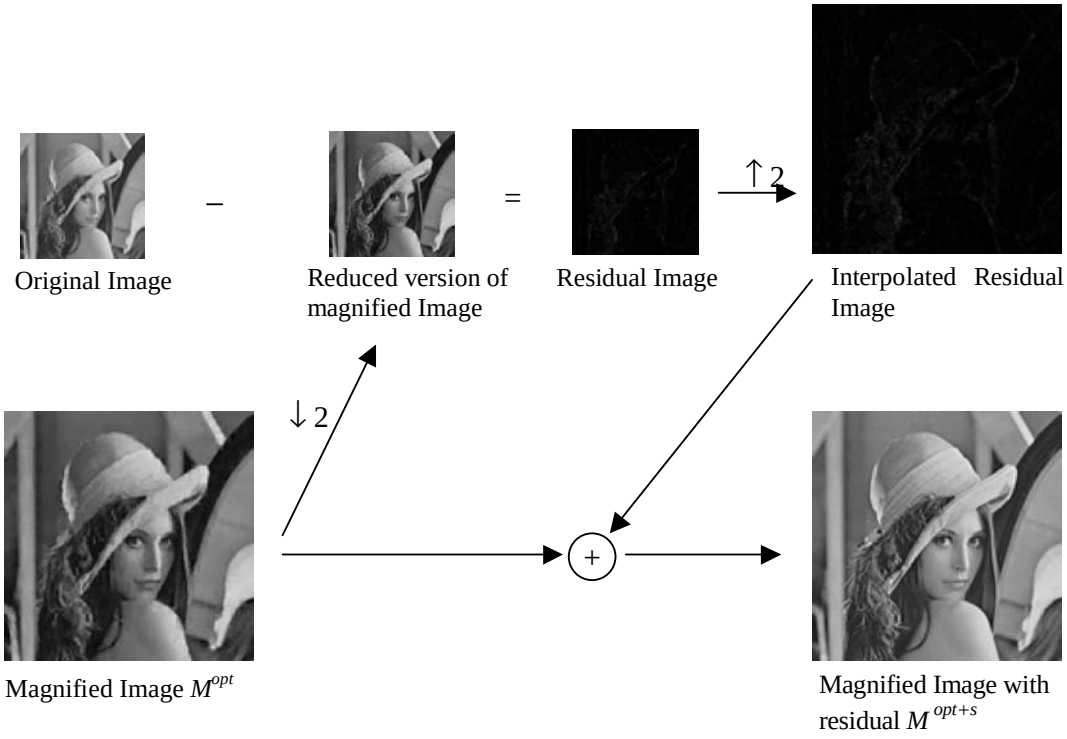


Figure 7-2 The error compensation process.

7.2.4 Combined with Other Algorithms

Chung *et al*. [127] proposed a fractal-based enlargement technique which can preserve the high-frequency contents of a magnified image by introducing an enhancement layer. This technique can be easily adapted to our proposed algorithm to further improve the quality of the magnified image. Firstly, a set of magnified images $M^{e}_{(i,j)}$ is generated with the enhancement layer using the adaptive overlapped range block partitioning scheme, as described in Section 7.2.2. Then, the optimal pixel selection scheme is used to obtain an optimal magnified image M^{e+opt}. Finally, the error compensation process is applied to M^{e+opt} to further improve the quality and obtain the magnified image $M^{e+opt+s}$.

7.3 Experimental Results

In the experiments, our algorithm was compared to the conventional fractal method [34], the well-known bicubic spline interpolation method, and the IEUF [127]. The PSNRs of our algorithm were measured based on (i) the pixel-averaging scheme, (ii) the pixel-averaging scheme with error compensation, and (iii) the combination of both schemes with IEUF. The experiments were conducted on a Pentium IV 2.4GHz processor, and five different 512×512 images with 256 gray levels – Lena, Boat, Goldhill, Couple and Baboon – were used to evaluate the respective performances of the different image magnification algorithms. To measure the performances, we first reduce the size of the high-resolution images by a factor α. These down-scaled images will be considered as the original low-resolution images, which are then magnified by the same

factor α by means of the different magnification algorithms. In the experiment, the value of the magnification factor α is set at 2. The PSNR of these magnified images are then computed with respect to their original high-resolution images.

Figure 7.3 plots the PSNR of the magnified images against different range block sizes varying from 3×3 to 8×8 for the different test images. Experimental results show that a range block size of 4×4 gives the best image quality based on the fractal image magnification scheme.

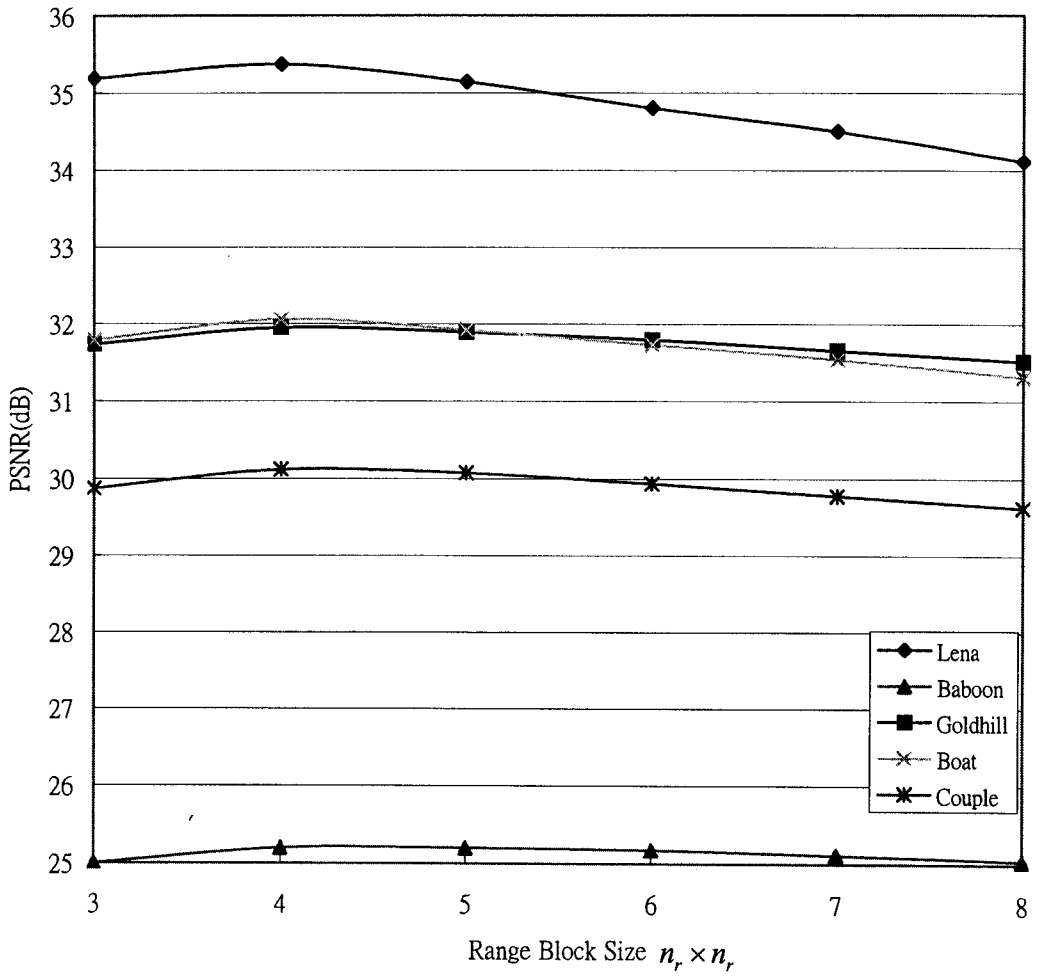
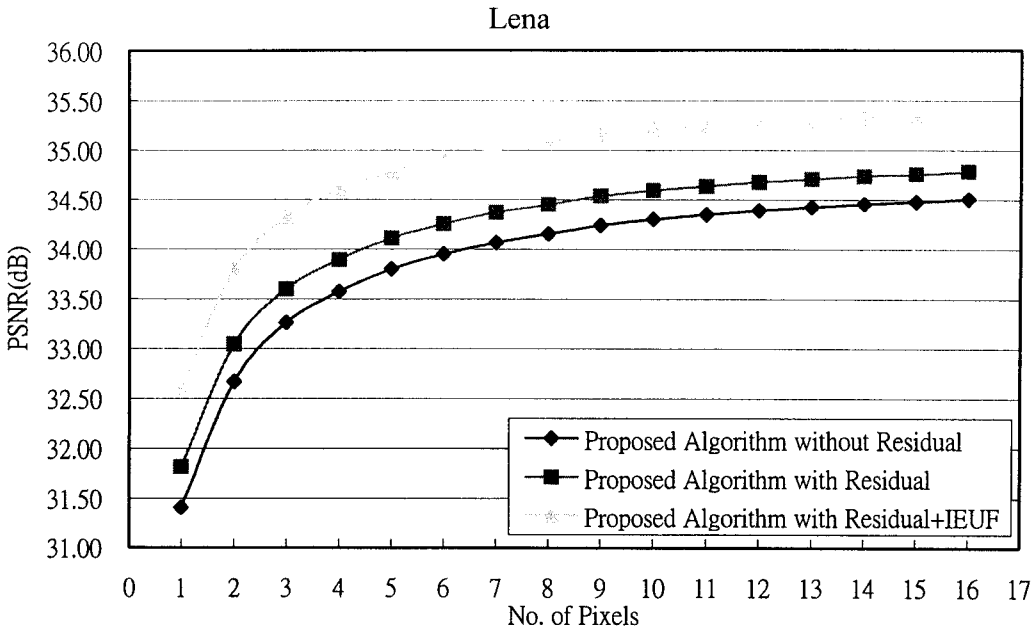
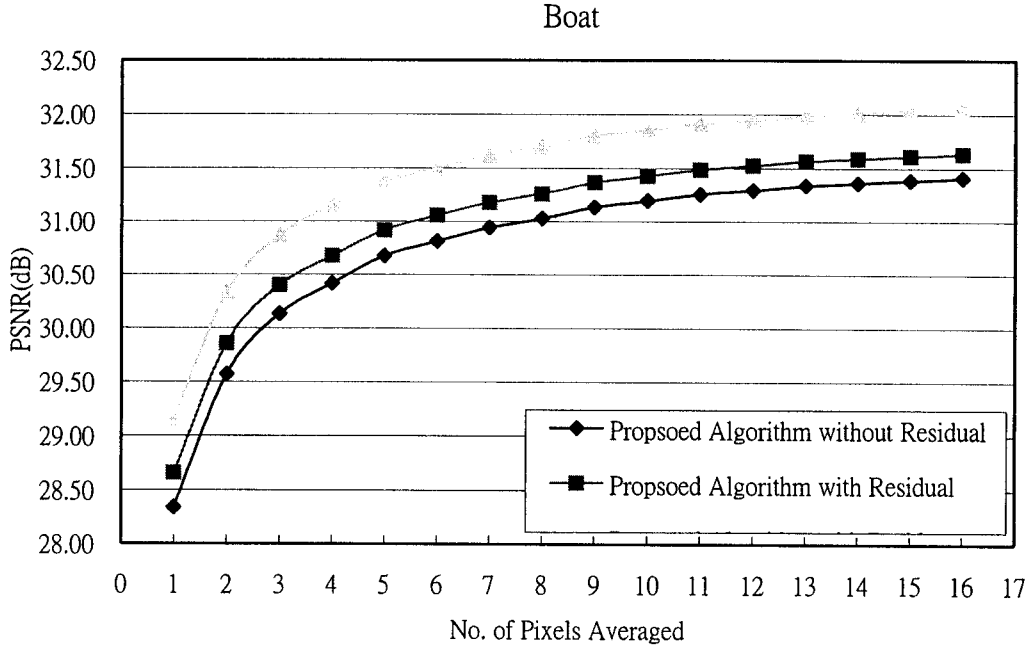


Figure 7-3 The magnified image quality against the range block of size 3×3 to 8×8 for different test images with magnification factor of 2.

The quality of a magnified image should improve if more pixel values are available for estimation at each position. Therefore, the quality of 4×4 is better than 3×3. However, when the range block size increases, the self-similarity between the blocks will decrease. This will degrade the quality of the magnified image. Consequently, a range block size of 4×4 can achieve the best performance. In the following experiments, we will use a range block size of 4×4 to compute the IFS code. Figure 7.4 shows the performance of our algorithm with the number of pixels being averaged when the range block size is 4×4. As explained in Section 7.2.2, the PSNR increases with the number of pixels being averaged. In general, our proposed algorithm with error compensation combined with IEUF achieves the best performance. Table 7.1 tabulates the respective performances of the different algorithms for different images. Our pixel-averaging scheme can achieve on average 2.6dB, 2.3dB, and 0.83dB increase in PSNR when compared to the conventional fractal scheme, the bicubic spline interpolation, and IEUF, respectively. The PSNR of the pixel-averaging scheme can be further improved by 0.22dB when the error compensation scheme is applied. This is due to the fact that the high-frequency components have been taken into account in the error compensation process. When our algorithm is combined with IEUF, a further improvement of 0.4dB is achieved.



(a)



(b)

Figure 7-4. The PSNR versus the number of pixels being averaged for the test images (a) Lena and (b) Boat.

Table 7-1 Performances in terms of the PSNR of our proposed algorithm and other algorithms based on different test images using a full search.

	Conventional Fractal Approach	Bicubic Spline Interpolation	IEUF	Proposed without residual	Proposed with residual	Proposed with residual+ IEUF
Lena	31.40	30.98	33.85	34.50	34.79	35.37
Baboon	23.00	23.88	24.08	24.66	24.86	25.20
Boat	28.34	28.45	30.36	31.41	31.63	32.06
Goldhill	29.05	29.46	30.59	31.49	31.65	31.95
Couple	26.85	27.35	28.59	29.57	29.78	30.11

Table 7-2 Performances in terms of the PSNR of our proposed algorithm and other algorithms based on different test images using the *k*-d tree.

	Conventional Fractal Approach	Bicubic Spline Interpolation	IEUF	Proposed without residual	Proposed with residual	Proposed with residual+ IEUF
Lena	31.15	30.98	33.78	34.32	34.66	35.33
Baboon	22.95	23.88	24.05	24.52	24.75	25.16
Boat	28.22	28.45	30.30	31.24	31.50	32.02
Goldhill	28.99	29.46	30.62	31.38	31.57	31.93
Couple	26.83	27.35	28.53	29.39	29.63	30.04

The main drawback of our proposed algorithm is its high computational complexity in the encoding phase to generate the IFS codes. The required encoding time is about 17,930s to generate 16 IFS codes for each 256×256 image. Therefore, the *k*-d tree is employed, which can greatly speed-up the encoding process. Table 7.2 tabulates the PSNR of the magnified images for the different test images using the *k*-d tree search. The average run-time is reduced to 198s at the expense of a small drop in quality in our proposed algorithm. From Table 7.2, we can see that the respective average drops in PSNR are 0.16dB, 0.12dB and 0.04dB for our proposed algorithms without residual, with residual and with residual combined with IEUF. In other words, using the *k*-d tree structure allows our algorithm with residual combined with IEUF to achieve a speed-up of 90 with only 0.04dB loss in quality when compared to the conventional full search scheme. This is mainly due to the use of the error compensation and enhancement layer technique, which can also compensate for the error caused by the *k*-d tree search. In additional, the average run-time can be further reduced by increasing the domain grid, with a slight drop in PSNR. Figure 7.5 shows the average run-time required by our algorithm with the use of the *k*-d tree search and different domain grid sizes versus the average PSNR drop when compared to the full search scheme.

Figures 7.6 and 7.7 illustrate the magnified images generated by using the different algorithms. It can be seen that the images magnified using the conventional fractal algorithm suffer from the blocky artifacts. The details of the images are blurred when the bicubic spline interpolation algorithm is used, as shown in Figures 7.6(c) and 7.7(c). The edges of the magnified images obtained by using our proposed algorithms are better than those obtained by the bicubic spline interpolation, conventional fractal algorithm and

IEUF. In other words, our algorithm has the best performance in terms of maintaining the high-frequency components and the texture properties of the images. In Figure 7.7, we can see that our magnified algorithm has the best performance in terms of preserving the fine details in the image, and provides the best subjective quality as well.

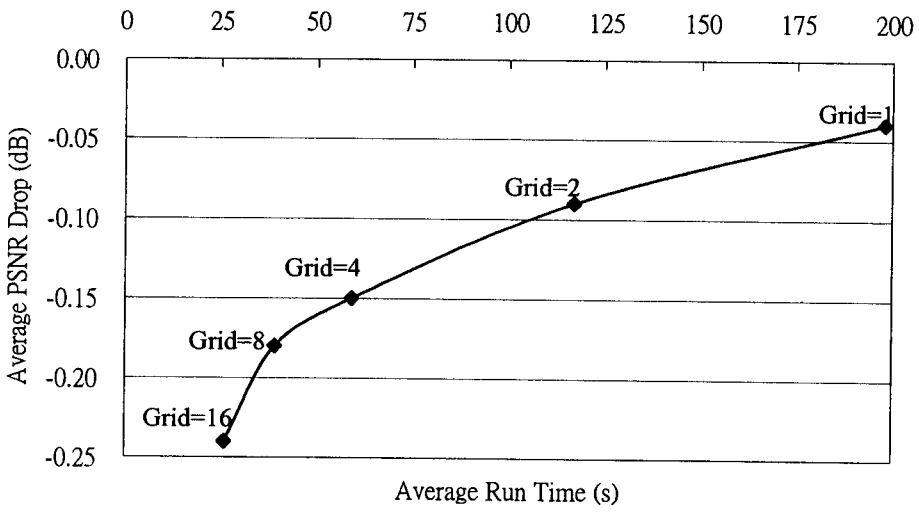


Figure 7-5 A comparison of average run-time required by our algorithm using the *k*-d tree search with different domain grid sizes versus the average PSNR drop when compared to the full search scheme.

(a)

(b) (c) (d)

(e) (f) (g)

Figure 7-6 (a) The original high-resolution image “Lena”, and the magnified images of Lena based on (b) the conventional fractal magnification; (c) the bicubic spline interpolation; (d) the IEUF; (e) the pixel-averaging scheme; (f) the pixel-averaging scheme with error compensation; and (g) the pixel-averaging scheme with error compensation combined with IEUF.

(a)

(b) (c) (d)

(e) (f) (g)

Figure 7-7 (a) The original high-resolution image “Baboon”, and the magnified images of Lena based on (b) the conventional fractal magnification; (c) the bicubic spline interpolation; (d) the IEUF; (e) the pixel-averaging scheme; (f) the pixel-averaging scheme with error compensation; and (g) the pixel-averaging scheme with error compensation combined with IEUF.

7.4 Conclusions

This chapter proposes an image magnification algorithm that can achieve a high subjective and objective quality. This algorithm improves the quality of a magnified image based on a set of high-resolution images generated using IFS codes. A pixel-averaging scheme and an error compensation scheme are proposed to form a magnified image which can preserve its high-frequency information and remove blocky artifacts. In addition, this algorithm can combine with other magnification algorithms, such as IEUF, to further improve the quality of a magnified image. Experimental results show that our magnification algorithm can achieve a significant improvement over the traditional methods in terms of objective and subjective qualities. In addition, the required run-time can be greatly reduced with the use of the *k*-d tree search, which will result in a slight degradation in quality.

Chapter 8. Adaptive Fractal-based Image Denoising

8.1 Introduction

One of the main challenges for image analysis is the reconstruction of an image with the existence of noises, which has been investigated intensively for many years [134-136, 159, 161]. This chapter aims to solve the problem of image denoising when an image is corrupted by additive white Gaussian noise (AWGN). This is a valid assumption when an image is captured by an imperfect image-capturing device, or distorted when transmitted over a network. The presence of noise can affect visual quality, as well as the processes of medical image analysis, object recognition, etc. Therefore, an efficient denoising algorithm is necessary in order to remove or alleviate the annoying effects.

In this chapter, we consider the typical problem of estimating a noise-free image f_{orig} from its noisy version f_N, which is the noise-free image degraded by AWGN. The problem is formally described as follows:

$$f_N = f_{orig} + G_N, \tag{8.1}$$

where f_{orig} is the original signal and G_N is AWGN. Thus, denoising is basically the methodology of estimating the signal f_{orig} from the Gaussian noise-corrupted signal f_N. Linear filtering techniques [159] have been widely used in many image processing applications, because of their simplicity to remove the additive white Gaussian noise. However, this kind of linear filter tends to blur the sharp edges, and destroy lines and other fine details in an image while failing to effectively remove heavy noise. A lot of

research work on noise reduction is based on wavelet thresholding [161]; a simple and very effective denoising method. The basic idea is to project a noisy image onto the wavelet kernels to generate a set of wavelet coefficients. Then, the small high-frequency coefficients can be set to zero, as they are most likely to be the noise. This thresholding technique can remove the noise efficiently and can preserve the fine details of the original image information. Other authors have combined wavelets with hidden Markov models [162]and spatially adaptive methods [163], or used other basis functions such as curvelets [164], so that a higher perceptual quality can be achieved when compared to the wavelet-based image denoising methods.

A recent fractal image denoising (FID) technique was proposed by Ghazel *et al.* [135], which can produce a denoised image efficiently under a heavy noise condition. The basic is to estimate the fractal code of a noise-free image from the corresponding noisy image based on a statistical approach. This fractal-based image denoising technique (FID) can achieve a gain in PSNR of 1.19dB on average when compared to the well-known Lee's filter [159] under the conditions that the standard deviation σ of the noise is larger than 20.

In this chapter, we propose a flexible and high-quality fractal-based image denoising algorithm, which can not only greatly reduce the additive white Gaussian noise (AWGN), but also maintain the high-frequency information so as to preserve the edges and effectively remove the blocky artifacts in a denoised image. Our proposed method can achieve increases of PSNR by 4.24dB and 1.09dB on average, when compared to the FID method under the low-noise ($\sigma < 20$) condition and the high-noise ($\sigma \geq 20$) condition, respectively.

This chapter is organized as follows. We will present the details of our new fractal-based image denoising algorithm in Section 8.2. In Section 8.3, we compare the performance of our proposed fractal-based image denosing algorithm with the recently proposed FID algorithm. Finally, conclusions are given in Section 8.4.

8.2 Proposed Algorithm

In this section, we present our efficient denoising algorithm based on the fractal property. Instead of using the conventional contrast scaling and offset parameters that are coupled to each other, our algorithm employs the decoupling property of the fractal code in which a range block is represented by its mean and the contrast scaling. A range-block partitioning scheme is used to generate a set of overlapping sub-images, and each of these sub-images is then represented by the mean values of the range blocks and the contrast scaling parameters for removing the noise in the image. The corresponding pixels in the respective sub-images are averaged to generate the best denoised image. Our proposed algorithm can not only remove the noise efficiently, but can also maintain the high-frequency contents. Therefore, the edges can be preserved in the denoised images without any blocky artifacts.

8.2.1 The Basic Idea of Fractal-based Image Denoising

One of the basic properties of fractal image coding is the notion of self-similarity in the different scales of a given image. In image coding, a range block can be approximated by a larger domain block from elsewhere in the image after applying decimation and a

non-linear intensity transformation. The computed fractal image can be recovered up to a small distortion by an iteration procedure at the decoder. The idea of the fractal image denoising is that range blocks can be described by a small number of contractive affine transformations based on the self-similarity structures across scales. However, it is impossible to approximate any random noisy structures occurring in an image [134, 135]. Therefore, the conventional fractal image coding technique can be applied to remove the noise in a low-noise condition. However, when the noise level is significant, the denoised images will lose their high frequency contents and suffer from blocky artifacts.

As shown in equation (2.13), a range block R is represented as a function of the best-matched domain block, and the contrast and offset parameters as follows:

$$R = sD_i + oI. \tag{8.2}$$

However, in encoding a noisy image, it is difficult to find the best-matched domain block corresponding to the noise-free version of a given range block. In other words, the contrast scaling s computed using equation (2.14), as well as the offset parameter o computed using equation (2.15), are unable to correctly represent the noise-free range block. The coupling effect between the contrast and offset parameters, as described in Section 2, makes it difficult to use fractal coding to remove the noise. In order to remove the noise in an image efficiently, equation (8.2) is decoupled into a DC term (the mean of a range block $\bar{r}$) and an AC term (the contrast scaling s), as proposed by Øien *et al.* [58] and Tong *et al.* [73]. This new representation of the range block R can be rewritten as follows by substituting equation (2.15) into equation(8.2):

$$\begin{aligned} R &= sD_i + oI \\ &= s\left(D_i - \bar{d}_i I\right) + s\bar{d}_i I + \left(\bar{r} - s\bar{d}_i\right) I, \\ &= s\left(D_i - \bar{d}_i I\right) + \bar{r} I \end{aligned} \tag{8.3}$$

where D_i is the best-matched domain block and $\bar{d}_i$ is the corresponding mean of the domain block. This formulation replaces o with the mean of the range block $\bar{r}$ that is independent of the contrast scaling s. The first term, $s(D_i - \bar{d}_i I)$, is usually small as $D_i \approx \bar{d}_i I$ and is sensitive to noise (due to the presence of D_i), while the second term, $\bar{r}I$, is less sensitive to noise (an average value is used) and has a larger magnitude when compared to the first term. Therefore, in the image denoising process, both the contrast scaling s and the mean of the range block $\bar{r}$ are used to reconstruct R when the noise level is low. However, when the noise level is high, the selected domain block D_i may not be the best-matched domain block of the corresponding noise-free range block. Thus, only the second term $\bar{r}I$ will be used in this condition. This scheme still suffers from the same problem of the conventional fractal-based image denoising; high-frequency content is lost and blocky artifacts appear. Therefore, we propose an Overlapped Range Block Partitioning Scheme to solve this problem. In our proposed denoising method, the threshold used to eliminate the noise is empirically determined by experiment.

8.2.2 Overlapped Range Block Partitioning Scheme

The fractal image coding scheme by Reusens [82] uses an overlapped range block partitioning scheme to construct an image with good visual quality. In this section, we will describe an extension of the overlapped range block technique for image denoising.

In order to remove the noise while preserve the visual details in a denoised image, the noisy image f_N of size $m_x \times m_y$ is partitioned into a set of overlapping sub-images. Let $f_{N(i,j)}$ denote a denoised image whose non-overlapping range blocks are extracted from the noisy image f_N starting at location (i, j), i.e.

$$f_{N(i,j)} = \sum_{p=0}^{\left(N_{R_x}^i - 1\right)} \sum_{q=0}^{\left(N_{R_y}^j - 1\right)} \left[R\left(pn_r + i, qn_r + j \right) \right], \quad i, j = 0, \ldots, n_r - 1, \tag{8.4}$$

where $N_{R_x}^i = \left\lfloor \frac{m_x - i}{n_r} \right\rfloor$ and $N_{R_y}^j = \left\lfloor \frac{m_y - j}{n_r} \right\rfloor$, which represent the number of range blocks in the horizontal and vertical directions, respectively. In other words, there are $n_r \times n_r$ possible images for $f_{N(i,j)}$, which are extracted from the noisy image f_N from i to $\left[m_x - 1 - \left((n_r - i) \bmod n_r \right) \right]$ in the horizontal direction and from j to $\left[m_y - 1 - \left((n_r - j) \bmod n_r \right) \right]$ in the vertical direction. Hence, there are $N_{R_x}^i \times N_{R_y}^j$ range blocks, where $0 \le i, j \le n_r - 1$. Obviously, when $i = 0$ and $j = 0$, it is easily seen that $f_{(0,0)}$ is equivalent to the original noisy image f_N. Otherwise, $f_{N(i,j)}$ is a sub-image of f_N. Fractal image coding is then applied to each of the images $f_{N(i,j)}$ independently to obtain the corresponding IFS codes $\boldsymbol{\tau}_{(i,j)}$. The IFS codes can be represented as follows:

$$\boldsymbol{\tau}_{(i,j)} = \sum_{p=0}^{\left(N_{R_x}^i - 1\right)} \sum_{q=0}^{\left(N_{R_y}^j - 1\right)} \left[\tau\left(p, q \right) \right] \quad i, j = 0, \ldots, n_r - 1. \tag{8.5}$$

The new transformation τ consists of the range block mean $\bar{r}$ instead of the offset parameter o, i.e. $\tau\left(p, q \right) = \left\{ s, \bar{r}, \iota, p_x, p_y \right\}$, where ι, p_x and p_y are the isometric operation, and the location of the best-matched domain block, respectively.

8.2.3 The Best Denoised Image

A set of denoised images $M_{(i,j)}$, where $0 \le i, j \le n_r - 1$, is constructed by iterating the corresponding IFS codes in the decoding process. $M_{(0,0)}$ is equivalent to reconstructing an image by the conventional fractal image denoising algorithm. The image qualities of

these N denoised images suffer from blocky artifacts and the loss of high-frequency details. In our proposed algorithm, the best denoised image M^{opt} is obtained by averaging those pixels at the same pixel position in the respective k images as follows:

$$M^{opt}(x, y) = \frac{1}{z} \sum_{s=0}^{z} M_s(x, y), \tag{8.6}$$

where z ($1 \leq z \leq k$) denotes the number of pixels to be averaged at a pixel location (x, y). The best pixel value at (x, y) in M^{opt} is obtained by averaging z pixels from the corresponding z denoised images $M_s = M_{(i,j)}$, where the index $s = i + j \cdot n_r$. For example, with reference to Figure 8-1, the values of z are equal to 1, 2, 4, 4, and 16 when the coordinates (x, y) are (0, 0), (1, 0), (3, 0), (0, 3) and (3, 3), respectively. In the inner part of the image M^{opt}, $n_r \times n_r$ pixels are involved. The pixel value at (x, y) in each of the denoised images $M_{(i,j)}$ can be considered as a random variable with the mean equal to the best value at that position. Consequently, averaging the corresponding pixel values of the denoised images $M_{(i,j)}$ should produce the best denoised image. This should also give both the best subjective and objective quality. From Figures 8-3 and 8-4, we can see that the PSNR of a denoised image increases dramatically when more corresponding pixels from the denoised images are averaged. The best denoised image M^{opt} with the highest PSNR is obtained by averaging all the magnified images available.

8.3 Experimental Results

In the experiments, a single partitioning scheme with range block size of 4×4 and a search grid of 4 is used. Eight popular 512×512 gray-level images were used to evaluate the performance of our proposed algorithm, which was also compared to the recently

proposed fractal-based image denoising algorithm, FID [135]. All these images were corrupted by AWGN with the standard deviation σ varying from 5 to 40. The performances of the different denoising algorithms were evaluated in terms of the Peak Signal-to-Noise Ratio (PSNR):

$$PSNR = 10\log_{10}\left(\frac{255\times 255}{\frac{1}{m_x m_y}\sum_{i=0}^{m_x-1}\sum_{j=0}^{m_y-1}\left(f_{Orig}(i,j)-\hat{f}(i,j)\right)^2}\right)dB \qquad (8.7)$$

where f_{orig} and $\hat{f}$ represent the original image and the denoised image of size $m_x \times m_y$, respectively.

The standard deviation σ of the noise is an important parameter, which affects the numerical quality and visual quality of the denoised images. This value can be estimated by examining the distribution of the local variances across the entire image [135]. In our algorithm, if σ is small**,** the best denoised image will be constructed from those respective denoised images generated by using both the first and second terms in equation (8.3). However, when σ is large, the best denoised image will be constructed by considering only the second term of equation (8.3), i.e. the means of the range blocks. Experiments with different values of σ were conducted in order to investigate the effect of σ on the PSNR of the denoised images. The image quality in terms of dB versus different levels of noise (σ) based on a range-block size of 4×4 is shown in Figure 8.2. The results show that the quality of a denoised image constructed by using both of the two terms in equation (8.3) starts to degrade noticeably when $\sigma \geq 20$. In these conditions, the quality of the denoised images using the mean of the range block only is better than that using both terms. In other words, the range mean and contrast scaling are used when the noise

standard deviation $\sigma < 20$, while only the means of the range blocks are used when $\sigma \geq 20$.

The numerical results of our denoising scheme and the FID scheme are illustrated in Figure 8.5. An additive white Gaussian noise (AWGN) is added to the images with the noise standard deviation changing from 5 to 40. For FID, the chosen range-block size is 8×8, the same as that used in [135]. Experimental results show that the noise is reduced effectively using our algorithm when compared to those denoised images obtained by the FID scheme. For low-noise ($\sigma < 20$) and high-noise ($\sigma \geq 20$) standard deviations, our proposed method can achieve, on average, 4.24dB and 1.09dB increase in PSNR, respectively, when compared to FID.

Our algorithm is always better than FID in terms of PSNR and the visual evaluation of image quality. Figures 8.6, 8.7 and 8.8 show the results of our proposed algorithm and the FID method. The FID eliminates most of the noise but, unfortunately, also over-smoothes the image, producing visually annoying artifacts on the denoised images. In Figures 8.6, 8.7 and 8.8, the first, second and third columns show the noisy images, the denoised images obtained using FID, and those obtained using our proposed algorithm, respectively. Our proposed algorithm outperforms the FID method in terms of image visual quality.

It is obvious that our proposed algorithm has a remarkable capability of noise reduction, as shown in the third column of Figures 8.6, 8.7 and 8.8. The edges of the denoised images in our proposed algorithm appear to be sharper than those in the FID method, as shown in the second column of Figures 8.6, 8.7 and 8.8. We can see that the FID method causes unsatisfactory behavior at the edges, in particular under heavy noise

conditions. For the Lena image, FID tends to blur the edges at the eyes. In contrast, the denoised images obtained using our proposed algorithm contains noticeably fewer visual artifacts. Similar results were observed based on the other test images.

8.4 Conclusions

In our algorithm, the noise in an image can be removed effectively while the edges can be preserved so as to provide fewer visual distortions and artifacts. Our algorithm employs the decoupling property of the fractal code instead of the conventional fractal code using the contrast scaling and offset parameters. This decoupling property makes it possible to denoise images more efficiently and flexibly. To improve the image quality, a range-block partitioning scheme is used to generate a set of overlapping sub-images. The range blocks of these sub-images are coded and represented by their range-block mean values and contrast scaling parameters for removing the noise. These sub-images are then averaged to obtain the best denoised image. A comparison of our proposed algorithm with a recently proposed technique for removing Gaussian noise from images validates the performance of our approach both quantitatively and visually.

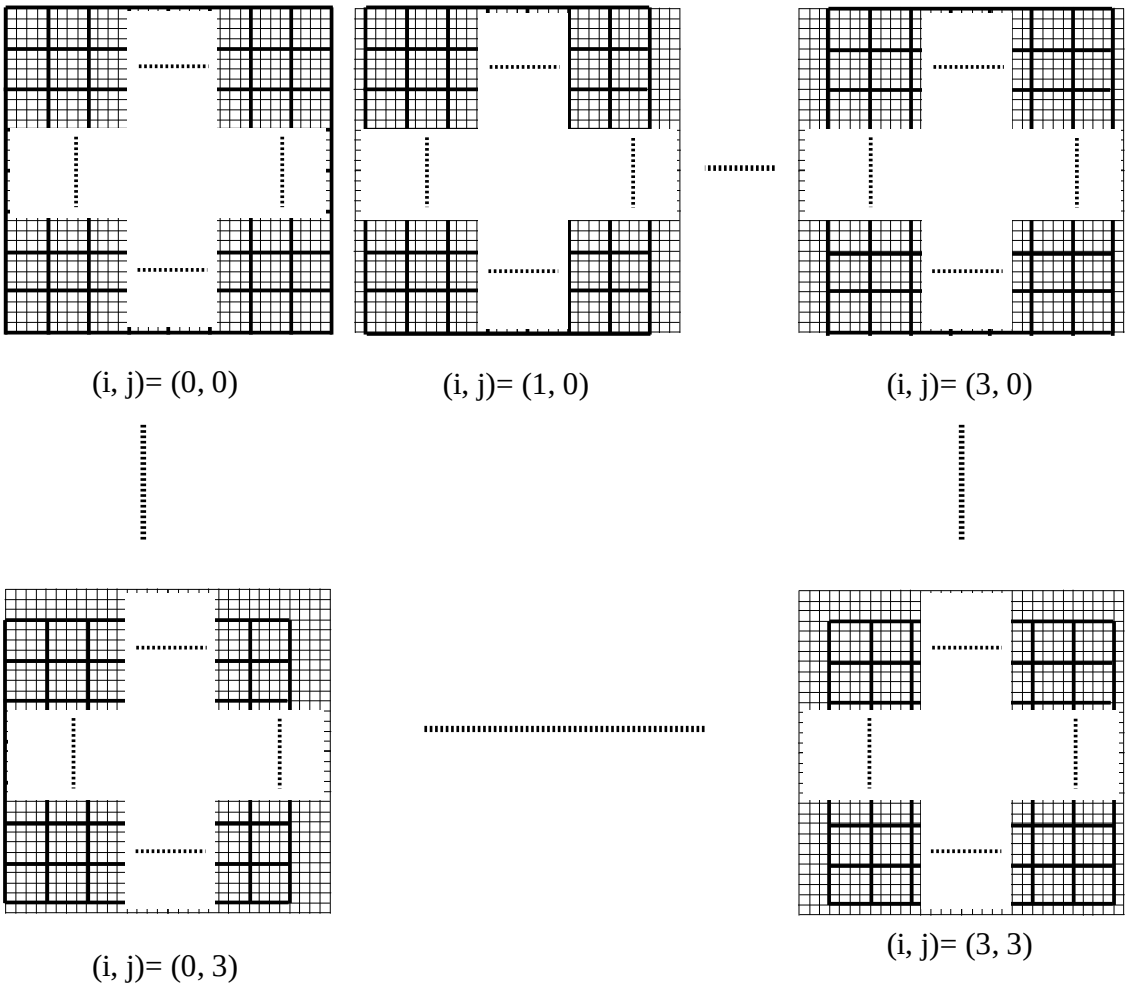


Figure 8-1. The first range blocks (represented by thick line) of size $n_r \times n_r$ (n_r = 4) overlapped sub-images obtained from a noise image. The starting point coordinates of the respective sub-images are (i, j), where $0 \leq x, y \leq n_r - 1$, and the sub-images are denoted as $f_{N(i,j)}$.

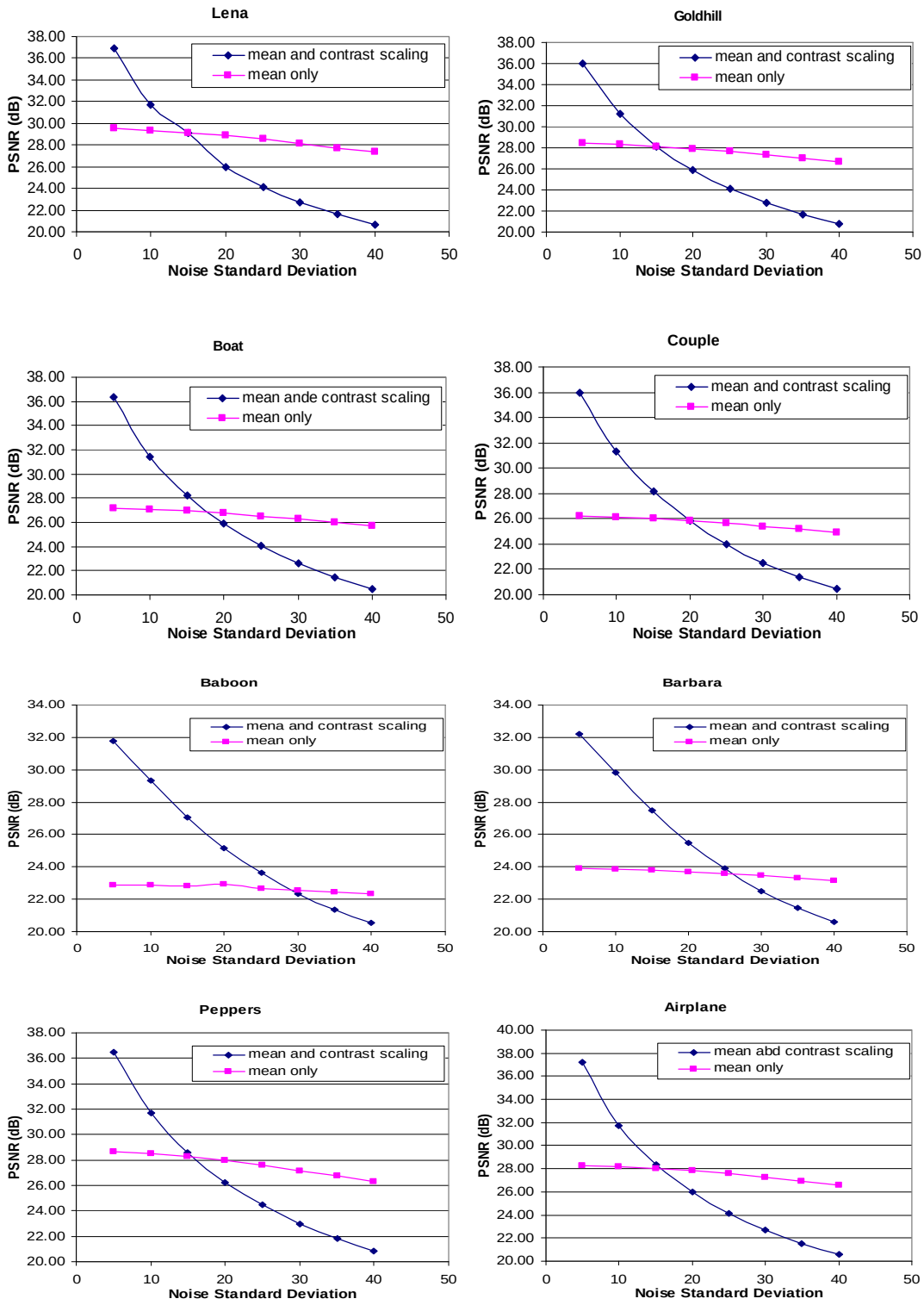


Figure 8-2 The denoising performance obtained using both the range-block mean and the contrast scaling, and using the range-block mean only under different noise levels.

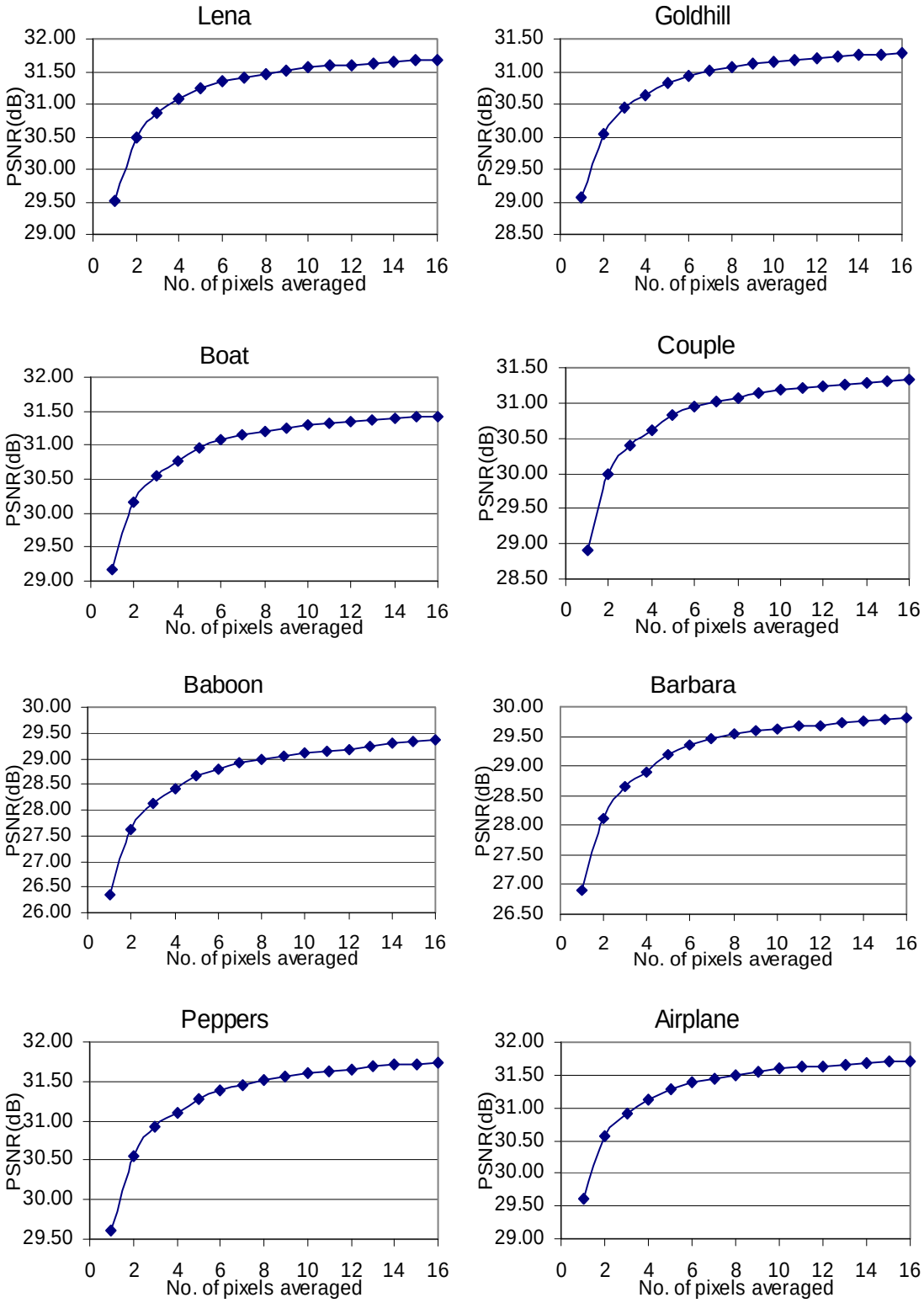


Figure 8-3. The PSNR versus the number of pixels being averaged for different test images when the standard deviation is 10.

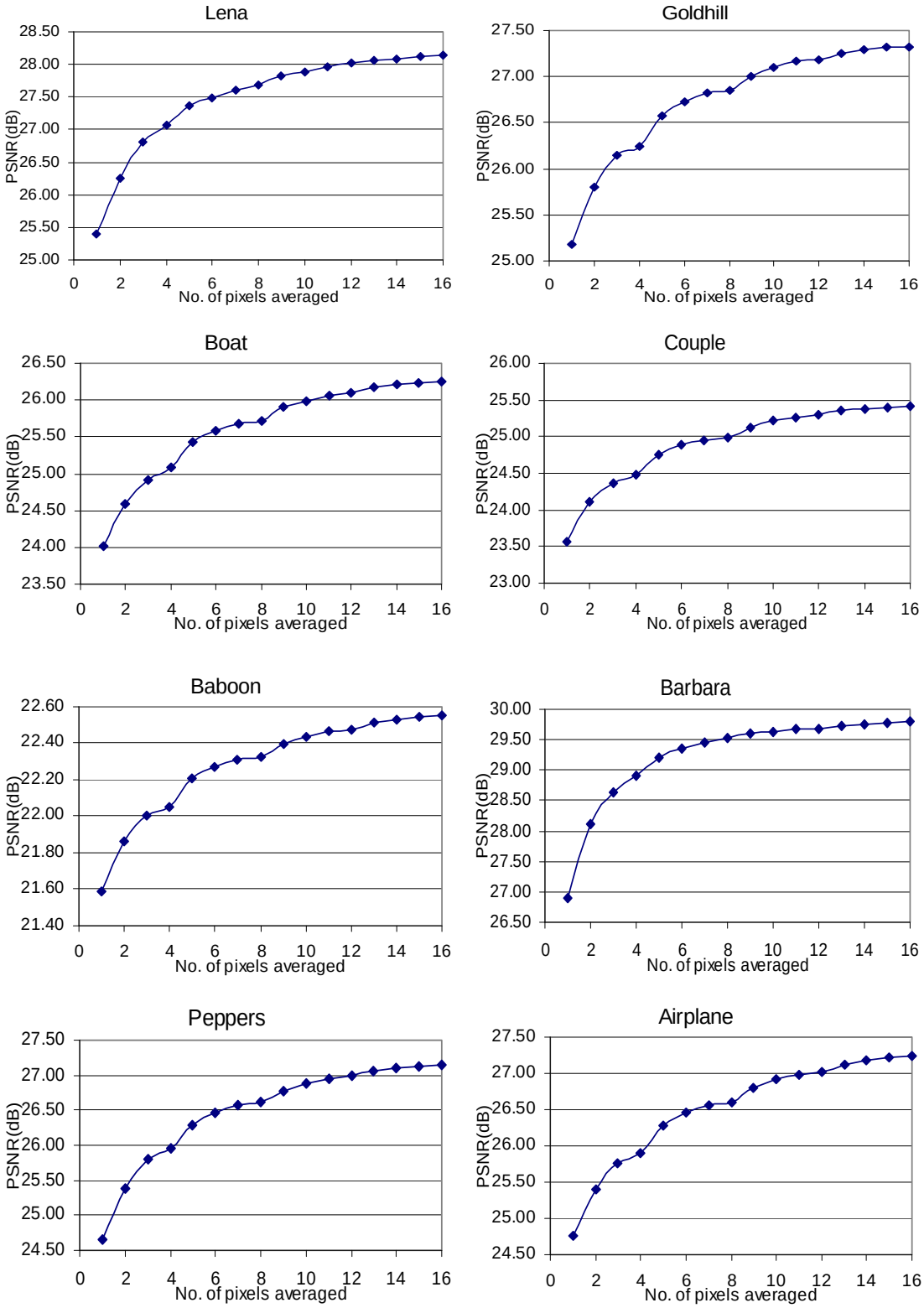
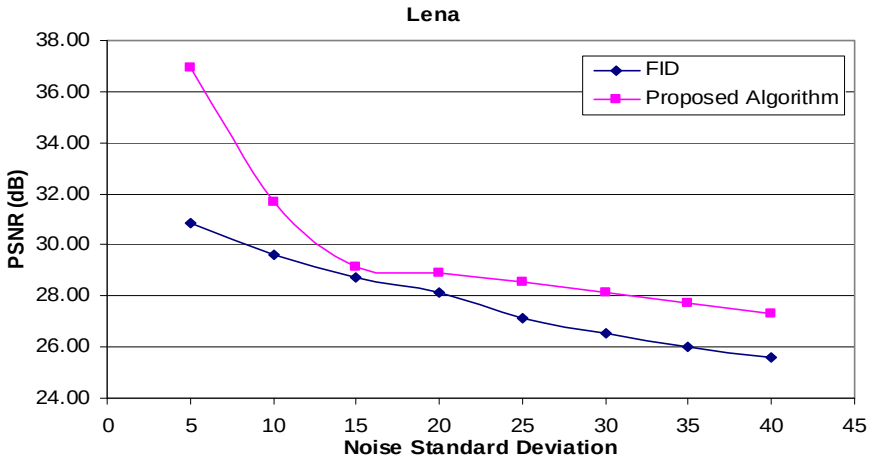
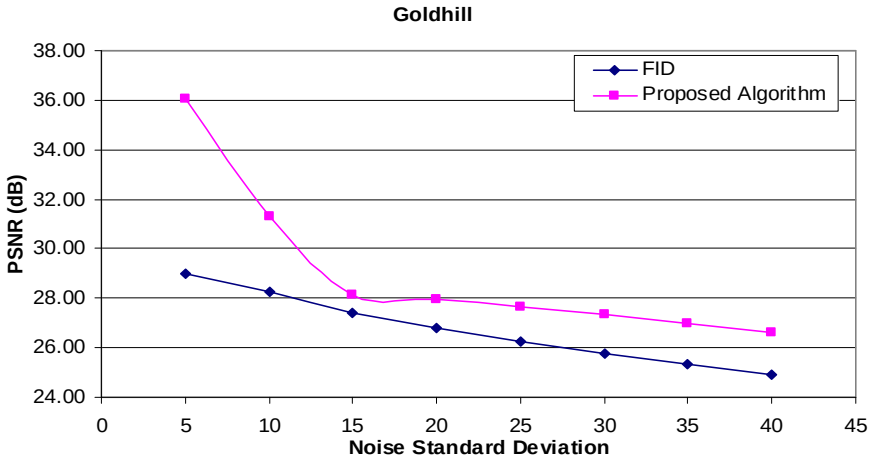


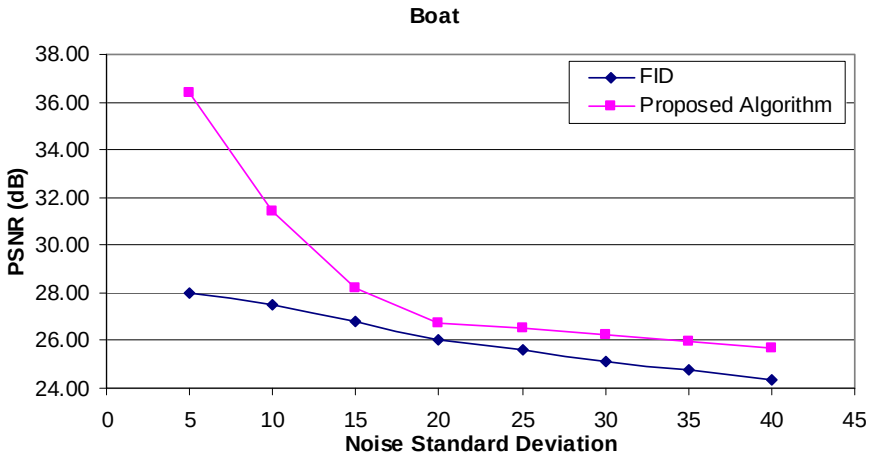
Figure 8-4 The PSNR versus the number of pixels being averaged for different test images when the standard deviation is 30.


Lena
38.00
36.00
34.00
32.00
30.00
28.00
26.00
24.00
PSNR (dB)
0
5
10
15
20
25
30
35
40
45
Noise Standard Deviation
FID
Proposed Algorithm

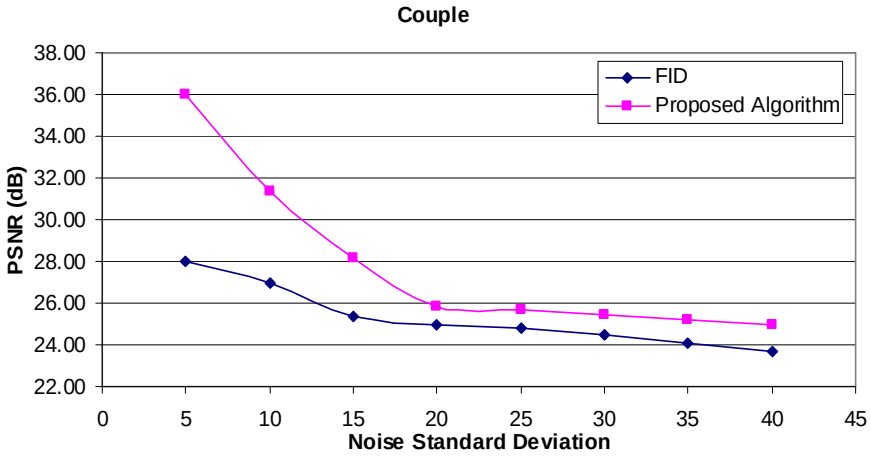
(a)


Goldhill
38.00
36.00
34.00
32.00
30.00
28.00
26.00
24.00
PSNR (dB)
0
5
10
15
20
25
30
35
40
45
Noise Standard Deviation
FID
Proposed Algorithm

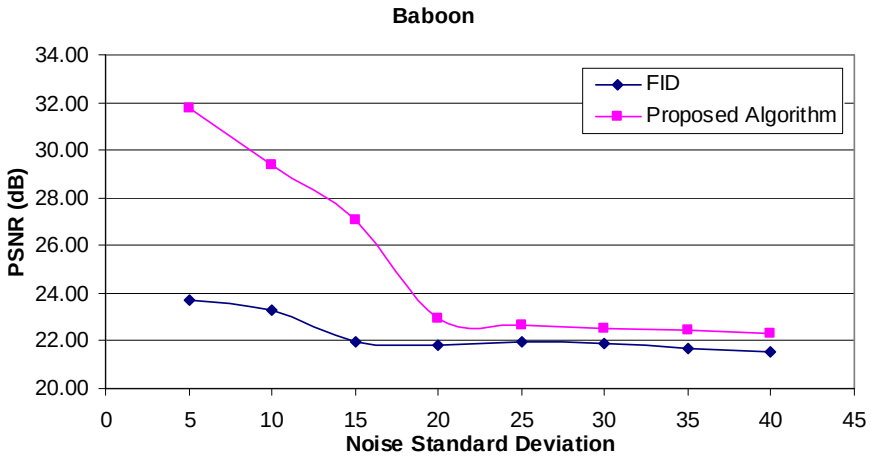
(b)


Boat
38.00
36.00
34.00
32.00
30.00
28.00
26.00
24.00
PSNR (dB)
0
5
10
15
20
25
30
35
40
45
Noise Standard Deviation
FID
Proposed Algorithm

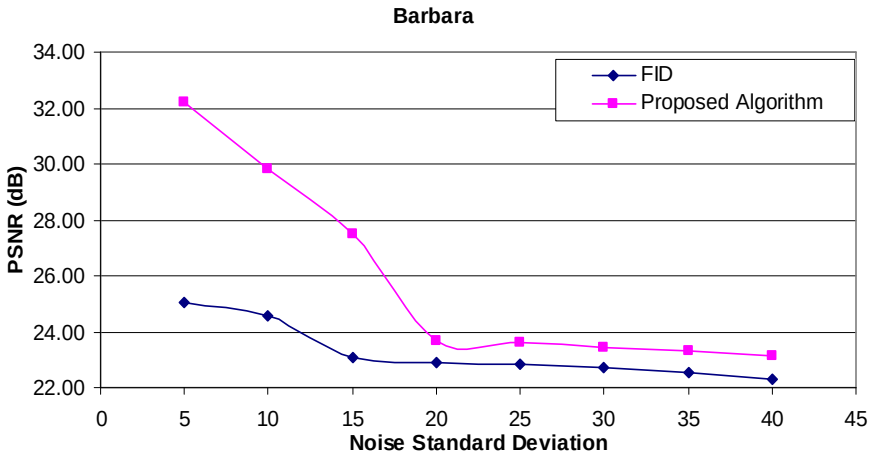
(c)


Couple
38.00
36.00
34.00
32.00
30.00
28.00
26.00
24.00
22.00
PSNR (dB)
0
5
10
15
20
25
30
35
40
45
Noise Standard Deviation
FID
Proposed Algorithm

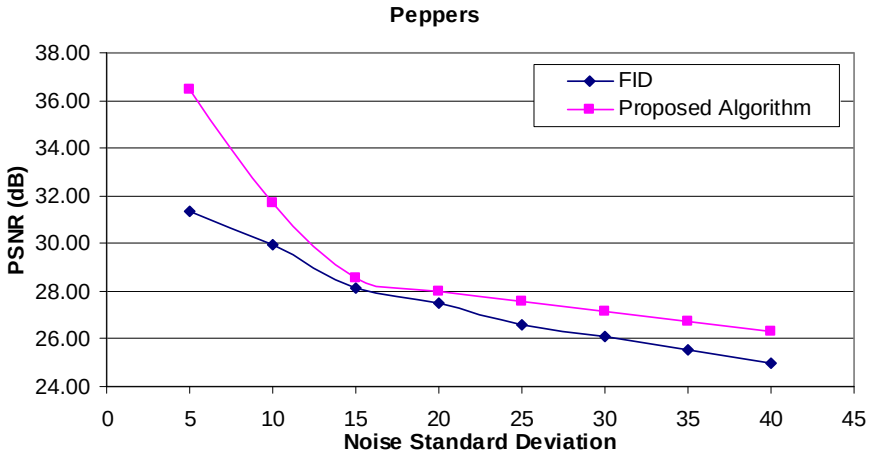
(d)


Baboon
34.00
32.00
30.00
28.00
26.00
24.00
22.00
20.00
PSNR (dB)
0
5
10
15
20
25
30
35
40
45
Noise Standard Deviation
FID
Proposed Algorithm

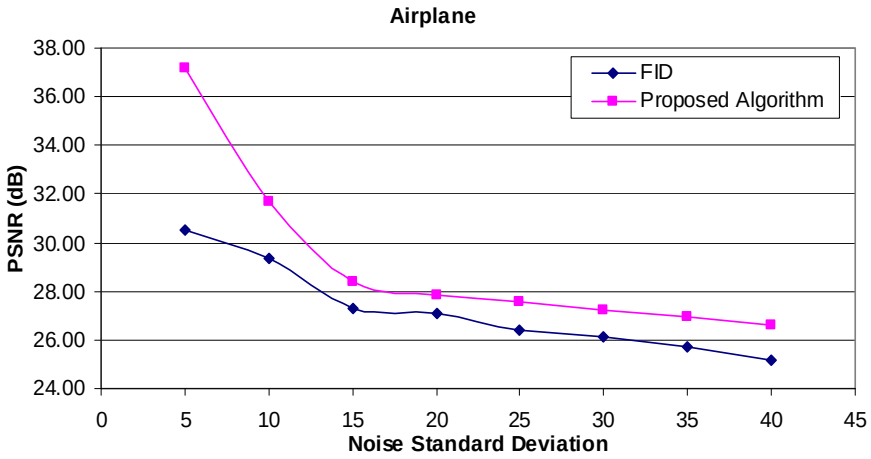
(e)


Barbara
34.00
32.00
30.00
28.00
26.00
24.00
22.00
PSNR (dB)
0
5
10
15
20
25
30
35
40
45
Noise Standard Deviation
FID
Proposed Algorithm

(f)



(g)



(h)

Figure 8-5 The denoising performances of our algorithm and the FID in terms of PSNR under different noise levels σ.

Noise Image with σ=10 FID Proposed Algorithm

Noise Image with σ=20 FID Proposed Algorithm

Noise Image with σ=30 FID Proposed Algorithm

Noise Image with σ=40 FID Proposed Algorithm

Figure 8-6 The visual qualities of our proposed algorithm and the FID method based on the test image "Barbara" under different noise levels σ.

Noise Image with σ=10 FID Proposed Algorithm

Noise Image with σ=20 FID Proposed Algorithm

Noise Image with σ=30 FID Proposed Algorithm

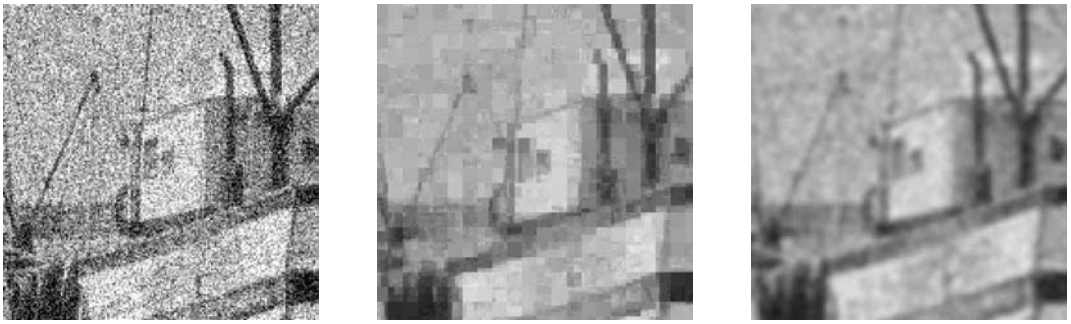
Noise Image with σ=40 FID Proposed Algorithm

Figure 8-7 The visual qualities of our proposed algorithm and the FID method based on the test image “Lena” under different noise levels σ.

Noise Image with σ=10 FID Proposed Algorithm

Noise Image with σ=20 FID Proposed Algorithm

Noise Image with σ=30 FID Proposed Algorithm



Noise Image with σ=40 FID Proposed Algorithm

Figure 8-8 The visual qualities of our proposed algorithm and the FID method based on the test image “Boat” under different noise levels σ.

Chapter 9. Conclusions and Future Work

9.1 Conclusion on the current works

In this thesis, we have provided an overview of fractal image coding and a variety of existing techniques to speed up the encoding process and some applications. Fractal image coding is an important step for many applications, such as image magnification, image denoising, etc. However, fractal encoding process suffers from high computational complexity. A significant amount of research work has been done on fractal image compression recently to speed-up the encoding, including domain pool reduction techniques, block classification schemes, fast full search algorithms, etc. In this thesis, we have devised four new efficient algorithms for fractal image compression and an algorithm for image magnification and one for image denoising.

Our first proposed algorithm uses a feature vector which can have a better representation of the image blocks is presented in Chapter3. Our algorithm can make the search of the corresponding domain block much more efficient and accurate than that of other existing fractal algorithms. We have also proposed the use of zero contrast condition to reduce the computational complexity and further improve the compression efficiency of the algorithm.

In Chapter 4, we have devised an equation to model the local variances method which uses a symmetrical window. From the devised equation, we proposed using a non-symmetric window to search for the best-matched domain block based on the local variances method.

The third approach is presented in Chapter 5 which is a fast algorithm based on a single kick-out condition and can achieve the same image quality as the full search. We found that almost all of the previous proposed algorithms will degrade the image quality when compared with the full search. Therefore, the efficiency of the kick-out condition depends on how quickly the global minimum error is detected. Once this global error is found, most of the remaining domain blocks will be rejected and range-domain block matching will not be performed.

A fast full-search fractal image-coding algorithm is presented in Chapter 6. Our algorithm is based on the relationship between the range-domain block matching error and the angle between the range and domain block vectors. An inequality in terms of the sine of the angle has been proposed. This inequality can reject impossible domain block candidates efficiently in searching for the best-matched domain block. Based on experimental results, our algorithm can reduce more than 80% of the required computations, as compared to the full-search algorithm, with only a slight quality degradation in the reconstructed image.

Chapter 7 presents an image magnification algorithm that can achieve a high subjective and objective quality. This algorithm improves the quality of a magnified image based on a set of high-resolution images generated using IFS codes. A pixel-averaging scheme and an error compensation scheme are proposed to form a magnified image which can preserve its high-frequency information and remove blocky artifacts. In addition, this algorithm can combine with other magnification algorithms, such as IEUF, to further improve the quality of a magnified image.

In Chapter 8, a new fractal-based image denoising method is presented which can preserve the edges and remove the blocky artifacts in a denoised image. Our algorithm employs the decoupling property of the fractal code instead of the conventional fractal code using the contrast scaling and offset parameters. This decoupling property makes it possible to denoise images more efficiently and flexibly. To improve the image quality, a range-block partitioning scheme is used to generate a set of overlapping sub-images. The range blocks of these sub-images are coded and represented by their range-block mean values and contrast scaling parameters for removing the noise. These sub-images are then averaged to obtain the best denoised image.

9.2 Future Work

9.2.1 Human Face Image Compression Using Fractal Coding

In the thesis, four efficient fractal coding algorithms based on feature vector matching, single kick-out and zero contrast conditions, and a non-symmetric search window have been devised. These efficient algorithms can be further investigated to extend to compress facial image. By considering the structure of a face image, the compression ratio and efficiency should be able to be further improved. The information about the face is the most important content to be considered, while the background is not important. Therefore, we can encode the background by boosting the use of zero contrast property and searching within a localized domain pool. This can speed up the encoding process and reduce the bit rate while maintaining a high reconstructed quality for the face region. Moreover, the encoding complexity can be further reduced by classifying the face region into two classes: edge block and smooth block. For encoding range blocks

containing edge, the search best domain blocks is limited to the edge domain blocks only. The coding of smooth blocks is performed in a similar manner. We believe that the face image coding technique based on zero contrast property and classified image block is one of the most efficient approaches to coding facial images.

9.2.2 Fractal Coding Of Video Sequences

Fractal-based image coding techniques can also be applied for coding video sequences. The domain and range blocks are classified based on the structure of the blocks. Then, the search of the best matched domain block is limited to those in the same class as of the range blocks. In addition, the position of the best-matched domain block in coding the previous frame will be used. This can further speed up the coding process. This approach is similar to the conventional motion estimation/compensation technique, but does not require encoding the residual errors. The complexity of the fractal image decoder is usually much lower than that of other decoder, such as those based on the transform methods. This makes this approach more suitable for broadcasting applications, where the video signals are compressed once by a powerful processor and the decompressed many times by many receivers.

9.2.3 An Efficient Fractal-based Super High Resolution Image Reconstruction using a Single Image

Super-High-Resolution (SHR) is becoming an increasingly important topic in digital image processing, in particular due to consumer digital photography becoming ever more popular. SHR has many areas of application, including enhancing the imagery for high-definition television (HDTV), medical imaging, aerial and satellite imaging,

remote sensing, surveillance systems, digital cameras, etc. High-resolution images can be obtained directly from high-precision optics and charge coupled devices (CCDs). However, due to hardware and cost limitations, imaging systems often provide us with only one or multiple low-resolution images. It has been known for some time that classical interpolation techniques such as linear and bicubic spline interpolations cannot produce high-resolution images of high quality, since these methods tend to blur and smooth the edges. The faithful reconstruction of edges in a magnified image is always important for human vision. Over the last two decades, research has been devoted to the problem of reconstructing a high-resolution image from a set of low-quality versions of a true image.

We propose a super-high-resolution image reconstruction algorithm, which estimates the high-resolution data from a single image based on fractal theory. One of the advantages of fractal theory is that it is resolution-independent. After the fractal image coding, the output of the coder is an Iterated Function System (IFS), which approximates the image as a fixed point of a contractive transformation. This IFS code can be used to reconstruct the image at any level of resolution. In our proposed algorithm, a set of high-resolution images is generated from a low-resolution image using the fractal image coding technique, and is then combined to form a high-quality magnified image.

In order to construct a super-high-resolution image with high visual quality, it is necessary to extract as much information as possible from the original image. Therefore, we can generate more super-high-resolution images by adopting more range block sizes in the encoding process. Typically, the range block sizes to be adopted can be adjust from 3×3 to 8×8. Experimental results show that the use of range block sizes of 3×3 and 4×4

can produce magnified images with optimal visual quality when the magnification factor $\alpha = 8$. In others words, 25 super-high-resolution images are generated using traditional fractal magnification algorithm. A set of 25 super-high-resolution images is constructed by iterating the corresponding IFS code in the decoding process. In the experiments, the magnified images obtained from our scheme with the adaptive pixel averaging method can achieve a better quality visually and in terms of PSNR than both the traditional fractal image magnification algorithm and the bicubic spline interpolation scheme. Figures 9.1-9.6 show the example of magnified images using different magnification methods.

Figure 9-1 Traditional Fractal magnification Algorithm with magnification factor α=8

Figure 9-2 Bicubic Spline Interpolation with magnification factor α=8

Figure 9-3 Proposed Algorithm with magnification factor α=8

Figure 9-4 Traditional Fractal magnification Algorithm with magnification factor α=8.

Figure 9-5 Bicubic Spline Interpolation with magnification factor α=8.

Figure 9-6 Proposed algorithm with magnifiocatin factor α=8

References

[1] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. 28, pp. 84-95, 1980.

[2] C. M. Huang, G. S. Stiles, Q. B. and, and R. W. Harris., "Fast full search equivalent encoding algorithms for image compression using vector quantization," *IEEE Transactions on Image Processing*, vol. 40, pp. 576-579, 1992.

[3] G. Poggi, "Fast algorithm for full-search VQ encoding," *Electronics Letters*, vol. 29, pp. 1141-1142, 1993.

[4] C.-H. Lee and L.-H. Chen, "A fast search algorithm for vector quantization using mean pyramids of codewords," *IEEE Transactions on Communications*, vol. 43, pp. 1697-1702, 1995.

[5] K.-T. Lo and J. Feng, "Predictive mean search algorithms for fast VQ encoding of images," *IEEE Transactions on Consumer Electronics*, vol. 41, pp. 327-331, 1995.

[6] C. H. Hsieh, "Lossless Compression of VQ Index with Search-Order Coding," *IEEE Transactions on Image Processing*, vol. 44, pp. 1632-1628, 1996.

[7] S. C. Tai, C. C. Lai, and Y. C. Lin, "Two fast nearest neighbor searching algorithms for image vector quantization," *IEEE Transactions on Communications*, vol. 44, pp. 1623-1628, 1996.

[8] S. Baek, B. Jeon, and K.-M. Sung, "A fast encoding algorithm for vector quantization," *IEEE Signal Processing Letters*, vol. 4, pp. 325-327, 1997.

[9] N. M. Nasarabadi, "Image coding using vector quantization: A review," *IEEE Transactions on Communications*, vol. 36, pp. 957-971, 1998.

[10] K.-L. Chung, W.-M. Yan, and J.-G. Wu, "A simple improved full search for vector quantization based on Winograd's identity," *IEEE Signal Processing Letters*, vol. 7, pp. 342-344, 2000.

[11] R. Hamzaoui and D. Saupe, "Combining fractal image compression and vector quantization," *IEEE Transactions on Image Processing*, vol. 9, pp. 197-208, 2000.

[12] C.-H. Hsieh and Y.-J. Liu, "Fast search algorithms for vector quantization of images using multiple triangle inequalities and wavelet transform," *IEEE Transactions on Image Processing*, vol. 9, pp. 321-328, 2000.

[13] C. M. Huang, Q. Bi, G. S. S, and R. W. Harris, "Fast Full Search Equivalent Encoding Algorithms for Image Compression Using Vector Quantization," *IEEE Signal Processing Letters*, vol. 7, pp. 342-344, 2000.

[14] Y. C. Lin, "Optimal Feature-based Vector Quantization of Image Coding Using Integral Projections," *Journal of Information Science and Engineering*, vol. 16, pp. 661-668, 2000.

[15] J. McNames, "Rotated partial distance search for faster vector quantization encoding," *IEEE Signal Processing Letters*, vol. 7, pp. 244-246, 2000.

[16] K.-S. Wu and J.-C. Lin, "Fast VQ encoding by an efficient kick-out condition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, pp. 59-62, 2000.

[17] J. Mohorko, P. Planinsic, and Z. CuCej, "Fast algorithm for pyramid vector quantization," *IEEE Signal Processing Letters*, vol. 8, pp. 103-105, 2001.

[18] J. Mielikainen, "A novel full-search vector quantization algorithm based on the law of cosines," *IEEE Signal Processing Letters*, vol. 9, pp. 175-176, 2002.

[19] B. C. Song and J. B. Ra, "A fast search algorithm for vector quantization using L2-norm pyramid of codewords," *IEEE Transactions on Image Processing*, vol. 11, pp. 10-15, 2002.

[20] Z. Pan, K. Kotani, and T. Ohmi, "A hierarchical fast encoding algorithm for vector quantization with PSNR equivalent to full search,"IEEE International Symposium on Circuits and Systems, 2002.

[21] J. Z. C. Lai and Y.-C. Liaw, "Fast-searching algorithm for vector quantization using projection and triangular inequality," *IEEE Transactions on Image Processing*, vol. 13, pp. 1554-1558, 2004.

[22] Z. Pan, K. Kotani, and T. Ohmi, "A unified projection method for fast search of vector quantization," *IEEE Signal Processing Letters*, vol. 11, pp. 637-640, 2004.

[23] Z. Pan, K. Kotani, and T. Ohmi, "An improved full-search-equivalent vector quantization method using the law of cosines," *IEEE Signal Processing Letters*, vol. 11, pp. 247-250, 2004.

[24] M. Chang and G. G. Langdon, Jr., "Effect of coefficient coding on JPEG baseline image compression,"Data Compression Conference, 1991. DCC '91., 1991.

[25] J. In, S. Shirani, and F. Kossentini, "JPEG compliant efficient progressive image coding,"Acoustics, Speech, and Signal Processing, 1998. ICASSP '98. Proceedings of the 1998 IEEE International Conference on, 1998.

[26] G. K. Wallace, "The JPEG still picture compression standard," *Consumer Electronics, IEEE Transactions on*, vol. 38, pp. xviii-xxxiv, 1992.

[27] W. Xiang, S. S. Pietrobon, and S. A. Barbulescu, "Joint source-channel decoding of jpeg images using error resilient source codes,"Information Theory, 2003. Proceedings. IEEE International Symposium on, 2003.

[28] O. K. Al-Shaykh, I. Moccagatta, and H. Chen, "JPEG-2000: a new still image compression standard,"Signals, Systems & Computers, 1998. Conference Record of the Thirty-Second Asilomar Conference on, 1998.

[29] J. F. Barda, "JPEG 2000, the next millennium compression standard for still images,"Multimedia Computing and Systems, 1999. IEEE International Conference on, 1999.

[30] M. J. Gormish, D. Lee, and M. W. Marcellin, "JPEG 2000: overview, architecture, and applications,"Image Processing, 2000. Proceedings. 2000 International Conference on, 2000.

[31] M. W. Marcellin, M. J. Gormish, A. Bilgin, and M. P. Boliek, "An overview of JPEG-2000,"Data Compression Conference, 2000. Proceedings. DCC 2000, 2000.

[32] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG 2000 still image compression standard," *Signal Processing Magazine, IEEE*, vol. 18, pp. 36-58, 2001.

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

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

[35] K. Belloulata and J. Konrad, "Fractal image compression with region-based functionality," *IEEE Transactions on Image Processing*, vol. 11, pp. 351-362, 2002.

[36] J. Cardinal, "Fast fractal compression of greyscale images," *IEEE Transactions on Image Processing*, vol. 10, pp. 159-164, 2001.

[37] H. T. Chang and C. J. Kuo, "Iteration-free fractal image coding based on efficient domain pool design," *IEEE Transactions on Image Processing*, vol. 9, pp. 329-339, 2000.

[38] G. M. Davis, "A wavelet-based analysis of fractal image compression," *IEEE Transactions on Image Processing*, vol. 7, pp. 141-154, 1998.

[39] R. Hamzaoui, "Decoding algorithm for fractal image compression," *Electronics Letters*, vol. 32, pp. 1273-1274, 1996.

[40] H. Hartenstein, M. Ruhl, and D. Saupe, "Region-based fractal image compression," *IEEE Transactions on Image Processing*, vol. 9, pp. 1171-1184, 2000.

[41] C. He, S. X. Yang, and X. Huang, "Progressive decoding method for fractal image compression," *IEE Proceedings- Vision, Image and Signal Processing*, vol. 151, pp. 207-213, 2004.

[42] C. He, S. X. Yang, and X. Xu, "Fast fractal image compression based on one-norm of normalised block," *Electronics Letters*, vol. 40, pp. 1052-1053, 2004.

[43] C. He, S. X. Yang, and X. Huang, "Variance-based accelerating scheme for fractal image encoding," *Electronics Letters*, vol. 40, pp. 115-116, 2004.

[44] B. Hurtgen, "Contractivity of fractal transforms for image coding," *Electronics Letters*, vol. 29, pp. 1749-1750, 1993.

[45] A. E. Jacquin, "Image coding based on a fractal theory of iterated contractive image transformations," *IEEE Transactions on Image Processing*, vol. 1, pp. 18-30, 1992.

[46] A. E. Jacquin, "Fractal image coding: a review," *Proceedings of the IEEE*, vol. 81, pp. 1451-1465, 1993.

[47] J. H. Jeng, T. K. Truong, and J. R. Sheu, "Fast fractal image compression using the Hadamard transform," *IEE Proceedings-Vision, Image and Signal Processing*, vol. 147, pp. 571-574, 2000.

[48] J. H. Jeng and J. R. Shyu, "Fractal image compression with simple classification scheme in frequency domain," *Electronics Letters*, vol. 36, pp. 716-717, 2000.

[49] I. K. Kim and R.-H. Park, "Still image coding based on vector quantization and fractal approximation," *IEEE Transactions on Image Processing*, vol. 5, pp. 587-597, 1996.

[50] C.-S. Kim, R.-C. Kim, and S.-U. Lee, "A fractal vector quantizer for image coding," *IEEE Transactions on Image Processing*, vol. 7, pp. 1598-1602, 1998.

[51] S. Kumar and R. C. Jain, "Low complexity fractal-based image compression technique," *IEEE Transactions on Consumer Electronics*, vol. 43, pp. 987-993, 1997.

[52] S. Kumar, K. N. Rao, R. R. Mishra, and R. C. Jain, "An efficient bath fractal transform-based image coding technique," *IEEE Transactions on Consumer Electronics*, vol. 44, pp. 1298-1308, 1998.

[53] C.-M. Lai, K.-M. Lam, and W.-C. Sin, "Improved searching scheme for fractal image coding," *Electronics Letters*, vol. 38, pp. 1653-1654, 2002.

[54] C.-M. Lai, K.-M. Lam, and W.-C. Siu, "A fast fractal image coding based on kick-out and zero contrast conditions," *IEEE Transactions on Image Processing*, vol. 12, pp. 1398-1403, 2003.

[55] E. M. Lalitha and L. Satish, "Fractal image compression for classification of PD sources," *IEEE Transactions on Dielectrics and Electrical Insulation*, vol. 5, pp. 550-557, 1998.

[56] C. K. Lee and W. K. Lee, "Fast fractal image block coding based on local variances," *IEEE Transactions on Image Processing*, vol. 7, pp. 888-891, 1998.

[57] S.-M. Lee and S.-W. Ra, "Fast fractal encoding algorithm using variances of image blocks," *Electronics Letters*, vol. 34, pp. 971-973, 1998.

[58] G. E. Oien and S. Lepsoy, "Fractal-based image coding with fast decoder convergence," *Signal Processing*, vol. 40, pp. 105-117, 1994.

[59] J. Li and C.-C. J. Kuo, "Image compression with a hybrid wavelet-fractal coder," *IEEE Transactions on Image Processing*, vol. 8, pp. 868-874, 1999.

[60] J. Li, G. Chen, and Z. Chi, "A fuzzy image metric with application to fractal coding," *IEEE Transactions on Image Processing*, vol. 11, pp. 636-643, 2002.

[61] S. K. Mitra, C. A. Murthy, and M. K. Kundu, "Technique for fractal image compression using genetic algorithm," *IEEE Transactions on Image Processing*, vol. 7, pp. 586-593, 1998.

[62] D. M. Monro and F. Dudbridge, "Fractal block coding of images," *Electronics Letters*, vol. 28, pp. 1053-1055, 1992.

[63] Y. H. Moon, H. S. Kim, and J. H. Kim, "A fast fractal decoding algorithm based on the selection of an initial image," *IEEE Transactions on Image Processing*, vol. 9, pp. 941-945, 2000.

[64] B. Pesquet-Popescu and J. L. Vehel, "Stochastic fractal models for image processing," *IEEE Signal Processing Magazine*, vol. 19, pp. 48-62, 2002.

[65] M. Polvere and M. Nappi, "Speed-up in fractal image coding: comparison of methods," *IEEE Transactions on Image Processing*, vol. 9, pp. 1002-1009, 2000.

[66] D. C. Popescu, A. Dimca, and H. Yan, "A nonlinear model for fractal image coding," *IEEE Transactions on Image Processing*, vol. 6, pp. 373-382, 1997.

[67] M. Ramkumar and G. V. Anand, "An FFT-based technique for fast fractal image compression," *Signal Processing*, vol. 63, pp. 263-268, 1997.

[68] D. Saupe, "A new view of fractal image compression as convolution transform coding," *IEEE Signal Processing Letters*, vol. 3, pp. 193-195, 1996.

[69] D. Saupe and S. Jacob, "Variance-based quadtrees in fractal image compression," *Electronics Letters*, vol. 33, pp. 46-48, 1997.

[70] D. Saupe, "Lean domain pools for fractal image compression," *Journal of Electronic Imaging*, vol. 8, pp. 98-103, 1999.

[71] S. S. Selvi and A. Makur, "Variable dimension range and domain block-based fractal image coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 343-347, 2003.

[72] L. Thomas and F. Deravi, "Region-based fractal image compression using heuristic search," *IEEE Transactions on Image Processing*, vol. 4, pp. 832-838, 1995.

[73] C. S. Tong and M. Pi, "Fast fractal image encoding based on adaptive search," *IEEE Transactions on Image Processing*, vol. 10, pp. 1269-1277, 2001.

[74] C. S. Tong and M. Wong, "Adaptive approximate nearest neighbor search for fractal image compression," *IEEE Transactions on Image Processing*, vol. 11, pp. 605-615, 2002.

[75] T.-K. Truong, J.-H. Jeng, I. S. Reed, P. C. Lee, and A. Q. Li, "A fast encoding algorithm for fractal image compression using the DCT inner product," *IEEE Transactions on Image Processing*, vol. 9, pp. 529-535, 2000.

[76] C.-C. Wan and C.-H. Hsieh, "An efficient fractal image-coding method using interblock correlation search," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, pp. 257-261, 2001.

[77] Z. Wang, D. Zhang, and Y. Yu, "Hybrid image coding based on partial fractal mapping," *Signal Processing: Image Communication*, vol. 15, pp. 767-779, 2000.

[78] B. E. Wohlberg and G. de Jager, "Fast image domain fractal compression by DCT domain block matching," *Electronics Letters*, vol. 31, pp. 869-870, 1995.

[79] Y. Zhang and L. M. Po, "Speeding up fractal image encoding by wavelet-based block classification," *Electronics Letters*, vol. 32, pp. 2140-2141, 1996.

[80] Z.-M. Zhang and S.-L. Yu, "An improved zero-searching fractal image coding method," *IEEE Transactions on Consumer Electronics*, vol. 45, pp. 91-96, 1999.

[81] Y. Zhao and B. Yuan, "Image compression using fractals and discrete cosine transform," *Electronics Letters*, vol. 30, pp. 474-475, 1994.

[82] E. Reusens, "Overlapped Adaptive Partitioning for Image Coding Based on the Theory of Iterated Functions Systems,"IEEE International Conference on Acoustics, Speech and Signal Processing, 1994.

[83] B. Bani-Eqbal, "Combining tree and feature classification in fractal encoding of images,"Proceedings Data Compression Conference, 1996. DCC '96., 1996.

[84] M. Barakat and J.-L. Dugelay, "Image sequence coding using 3-D IFS,"Proceedings., International Conference on Image Processing, 1996., 1996.

[85] D. Saupe and H. Hartenstein, "Lossless acceleration of fractal image compression by fast convolution,"Proceedings., International Conference on Image Processing, 1996., 1996.

[86] B. Bani-Eqbal, "Fractal encoding of images by classified domain trees,"Sixth International Conference on Image Processing and Its Applications, 1997.,, 1997.

[87] D. Cai, "Adaptive block partitioning in fractal image coding,"Proceedings of IEEE TENCON '97. IEEE Region 10 Annual Conference. Speech and Image Technologies for Computing and Telecommunications'.,, 1997.

[88] Y.-C. Chang, B.-K. Shyu, and J.-S. Wang, "Region-based fractal image compression with quadtree segmentation,"Processing, 1997. ICASSP-97., 1997 IEEE International Conference on Acoustics, Speech, and Signal, 1997.

[89] M. Pi, J. Peng, and M. Ding, "Fractal approximation of image block in its neighbor and fractal coding,"1997 IEEE International Conference on Intelligent Processing Systems, 1997. ICIPS '97., 1997.

[90] M. Ruhl, H. Hartenstein, and D. Saupe, "Adaptive partitionings for fractal image compression,"Proceedings., International Conference on Image Processing, 1997., 1997.

[91] C. chunling, W. shaodi, and S. bingzhe, "A fractal image coding based on the quadtree,"Proceedings, 1998. ICSP '98. 1998 Fourth International Conference on Signal Processing, 1998.

[92] D. J. Hebert and E. Soundararajan, "Fast fractal image compression with triangular multiresolution block matching, "Proceedings. 1998 International Conference on Image Processing, 1998. ICIP 98., 1998.

[93] B. M., *Fractal Everywhere*. San Diego: Academic Press, 1998.

[94] W. Penghui and Z. Baoyu, "A new image compression method based on HV fractal and DCT,"Proceedings, 1998. ICCT '98. 1998 International Conference on Communication Technology, 1998.

[95] D. Saupe, M. Ruhl, R. Hamzaoui, L. Grandi, and D. Marini, "Optimal hierarchical partitions for fractal image compression, "Proceedings. 1998 International Conference on Image Processing, 1998. ICIP 98., 1998.

[96] M. Breazu, G. Toderean, D. Volovici, and M. Iridon, "Speeding up fractal image compression by working in Karhunen-Loeve transform space, "International Joint Conference on Neural Networks, 1999. IJCNN '99., 1999.

[97] H. Honda, M. Haseyama, and H. Kitajima, "Fractal interpolation for natural images,"Proceedings. 1999 International Conference on Image Processing, 1999. ICIP 99., 1999.

[98] I. Salih and S. H. Smith, "Encoding time reduction in fractal image compression,"Proceedings. DCC '99 Data Compression Conference, 1999., 1999.

[99] W. Xuejun, C. Lianyu, and C. Hexin, "A quadtree classified-based fractal image coding approach,"Fifth Asia-Pacific Conference on ... and Fourth Optoelectronics and Communications Conference Communications, 1999. APCC/OECC '99., 1999.

[100] R. Yuxuan and T. G. Nge, "An improved fractal image compression scheme embedding DCT encoder,"Seventh International Conference on (Conf. Publ. No. 465) Image Processing and Its Applications, 1999., 1999.

[101] D. Dasgupta, G. Hernandez, and F. Nino, "An evolutionary algorithm for fractal coding of binary images," *Evolutionary Computation, IEEE Transactions on*, vol. 4, pp. 172-181, 2000.

[102] R. Hamzaoui, D. Saupe, and M. Hiller, "Fast code enhancement with local search for fractal image compression, "Proceedings. 2000 International Conference on Image Processing, 2000., 2000.

[103] C. Tong and M. Wong, "Approximate nearest neighbour search for fractal image compression based on a new affine transform parametrization, "Proceedings. 15th International Conference on Pattern Recognition, 2000., 2000.

[104] W. M. Woon, A. T. S. Ho, T. Yu, S. C. Tam, S. C. Tan, and L. T. Yap, "Achieving high data compression of self-similar satellite images using fractal, "Proceedings. IGARSS 2000. IEEE 2000 International Geoscience and Remote Sensing Symposium, 2000., 2000.

[105] S. S. Koh and C. H. Kim, "Fractal image coding based on the accurate estimation of image paramters from noise image, "IEEE International Conference on Multimedia and Expo, 2001. ICME 2001., 2001.

[106] Y. Zhao and B. Yuan, "A novel scheme for fractal image coding, "Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing, 2001., 2001.

[107] K. M. Curtis, G. Neil, and V. Fotopoulos, "A hybrid fractal/DCT image compression method,"2002 14th International Conference on Digital Signal Processing, 2002. DSP 2002., 2002.

[108] J. Valantinas, N. Morkevicius, and T. Zumbakis, "Accelerating compression times in block based fractal image coding procedures, "Proceedings. The 20th Eurographics UK Conference, 2002., 2002.

[109] G. Farhadi, "A hybrid image compression scheme using block-based fractal coding and DCT,"4th EURASIP Conference focused on Video/Image Processing and Multimedia Communications, 2003., 2003.

[110] G. Farhadi, "An enhanced fractal image compression based on quadtree partition,"Proceedings of the 3rd International Symposium on Image and Signal Processing and Analysis, 2003. ISPA 2003., 2003.

[111] C.-M. Lai, K.-M. Lam, and W.-C. Siu, "An efficient algorithm for fractal image coding using kick-out and zero contrast conditions, "Proceedings of the 2003 International Symposium on Circuits and Systems, 2003. ISCAS '03., 2003.

[112] D. Loganathan, J. Amudha, and K. M. Mehata, "Classification and feature vector techniques to improve fractal image coding, "TENCON 2003. Conference on Convergent Technologies for Asia-Pacific Region, 2003.

[113] D. Suhanov and A. Suhanov, "Algorithm of resolution increasing of digital images using fractal-spectral representation, "The IEEE-Siberian Conference on Control and Communications, 2003. SIBCON 2003., 2003.

[114] Y.-G. Wu, M.-Z. Huang, and Y.-L. Wen, "Fractal image compression with variance and mean,"Proceedings. 2003 International Conference on Multimedia and Expo, 2003. ICME '03., 2003.

[115] B. Aoued, "Accelerating fractal image compression by domain pool reduction, adaptive partitioning and structural block classification, "IEEE/Sarnoff Symposium on Advances in Wired and Wireless Communication, 2004, 2004.

[116] L. Hu, Qi-Chen, and D. Zhang, "An image compression method based on fractal theory,"Proceedings. The 8th International Conference on Computer Supported Cooperative Work in Design, 2004., 2004.

[117] J. Lakkad and J. K. Ghosh, "Fractal feature for land use class of satellite images,"Proceedings Elmar 2004. 46th International Symposium Electronics in Marine, 2004., 2004.

[118] A. M. Ramirez, A. D. Sanchez, M. L. Aranda, and J. V. Pineda, "An architecture for fractal image compression using quad-tree multiresolution, "Proceedings of

the 2004 International Symposium on Circuits and Systems, 2004. ISCAS '04., 2004.

[119] B. Bani-Eqbal, "Enhancing the speed of fractal image compression," *Optical Engineering*, vol. 34, pp. 1705-1710, 1995.

[120] S. Lee and S. Ra, "An analysis of siometry transforms in frequency domain for the fast fractal coding," *IEEE Signal Processing Letter*, vol. 6, pp. 100-102, 1999.

[121] D. Saupe, "Accelerating fractal image compression by multi-dimensional nearest neighbor search,"Proceedings Data Compression Conference, 1995. DCC '95., 1995.

[122] D. Saupe and M. Ruhl, "Evolutionary fractal image compression, "Proceedings., International Conference on Image Processing, 1996., 1996.

[123] J. R. Price and M. H. Hayes, III, "Fractal interpolation of images and volumes,"Signals, Systems & Computers, 1998. Conference Record of the Thirty-Second Asilomar Conference on, 1998.

[124] J. R. Price and M. H. Hayes, III, "Resampling and reconstruction with fractal interpolation functions," *Signal Processing Letters, IEEE*, vol. 5, pp. 228-230, 1998.

[125] H. Honda, M. Haseyama, and H. Kitajima, "Fractal interpolation for natural images,"Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on, 1999.

[126] S. K. Mitra, C. A. Murthy, and M. K. Kundu, "Image compression and magnification using fractal technique, "TENCON '98. 1998 IEEE Region 10 International Conference on Global Connectivity in Energy, Computer, Communication and Control, 1998.

[127] K.-H. Chung, Y.-H. Fung, and Y.-H. Chan, "Image enlargement using fractal,"Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on, 2003.

[128] O. I. Craciunescu, S. K. Das, J. M. Poulson, and T. V. Samulski, "Three-dimensional tumor perfusion reconstruction using fractal interpolation functions," *Biomedical Engineering, IEEE Transactions on*, vol. 48, pp. 462-473, 2001.

[129] P. Motallebi and P. O. Ogunbona, "Edge image description using fractal interpolation, "TENCON '97. IEEE Region 10 Annual Conference. Speech and

Image Technologies for Computing and Telecommunications'., Proceedings of IEEE, 1997.

[130] D. Suhanov and A. Suhanov, "Algorithm of resolution increasing of digital images using fractal-spectral representation, "Control and Communications, 2003. SIBCON 2003. The IEEE-Siberian Conference on, 2003.

[131] Z. Baharav, D. Malah, and E. D. Karnin, "Hierarchical interpretation of fractal image coding and its applications, "International Conference on Digital Signal Processing 1993, 1993.

[132] L. Zhigang, L. Daizhi, and W. Renming, "An improved fractal-based interpolation method for terrain imagery zooming, "Signal Processing Proceedings, 2000. WCCC-ICSP 2000. 5th International Conference on, 2000.

[133] S. Bal and W. Kinsner, "Image denoising for reduced-search fractal block coding,"Electrical and Computer Engineering, 1996. Canadian Conference on, 1996.

[134] M. Ghazel, G. H. Freeman, and E. R. Vrscay, "Fractal-wavelet image denoising,"Image Processing. 2002. Proceedings. 2002 International Conference on, 2002.

[135] M. Ghazel, G. H. Freeman, and E. R. Vrscay, "Fractal image denoising," *Image Processing, IEEE Transactions on*, vol. 12, pp. 1560-1578, 2003.

[136] M. Haseyama, M. Takezawa, K. Kondo, and H. Kitajima, "An image restoration method using IFS,"Image Processing, 2000. Proceedings. 2000 International Conference on, 2000.

[137] P. Bas, J.-M. Chassery, and F. Davoine, "Using the fractal code to watermark images,"Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on, 1998.

[138] C. Hanqiang and Z. Guangxi, "A watermarking method based on fractal self-similarity,"Signal Processing Proceedings, 2000. WCCC-ICSP 2000. 5th International Conference on, 2000.

[139] D. Levicky and P. Foris, "Some modifications of fractal image coding and digital watermarking,"Industrial Technology, 2003 IEEE International Conference on, 2003.

[140] M. H. Pi and C.-H. Li, "A novel fractal watermarking technique, "Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference on, 2004.

[141] Z. Yao, "Fixed point in fractal image compression as watermarking, "Image Processing, 2003. Proceedings. 2003 International Conference on, 2003.

[142] S. Chandran and S. Kar, "Retrieving faces by the PIFS fractal code, "Applications of Computer Vision, 2002. (WACV 2002). Proceedings. Sixth IEEE Workshop on, 2002.

[143] H. Ebrahimpour-Komleh, V. Chandran, and S. Sridharan, "Robustness to expression variations in fractal-based face recognition, "Signal Processing and its Applications, Sixth International, Symposium on. 2001, 2001.

[144] H. Ebrahimpour-Komleh, V. Chandran, and S. Sridharan, "Face recognition using fractal codes,"Image Processing, 2001. Proceedings. 2001 International Conference on, 2001.

[145] A. Z. Kouzani, F. He, and K. Sammut, "Fractal face representation and recognition,"Systems, Man, and Cybernetics, 1997. 'Computational Cybernetics and Simulation'., 1997 IEEE International Conference on, 1997.

[146] A. Z. Kouzani, F. He, and K. Sammut, "Face image matching using fractal dimension,"Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on, 1999.

[147] K.-H. Lin, K.-M. Lam, and W.-C. Siu, "Locating the eye in human face images using fractal dimensions," *Vision, Image and Signal Processing, IEE Proceedings-*, vol. 148, pp. 413-421, 2001.

[148] T. Tan and H. Yan, "Face recognition by fractal transformations, "Acoustics, Speech, and Signal Processing, 1999. ICASSP '99. Proceedings., 1999 IEEE International Conference on, 1999.

[149] T. Tan and H. Yan, "Analysis of the contractivity factor in fractal based face recognition,"Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on, 1999.

[150] R. Distasi, M. Nappi, and M. Tucci, "Using fractal encoding for image indexing,"Image Analysis and Processing, 1999. Proceedings. International Conference on, 1999.

[151] R. Distasi, M. Nappi, and M. Tucci, "FIRE: fractal indexing with robust extensions for image databases," *Image Processing, IEEE Transactions on*, vol. 12, pp. 373-384, 2003.

[152] K. F Li, T. S. C. and, and K. H. Chen, "Fractal Image Process Based Image Comparison Search Engine," *Journal Of Information Science and Engineering*, vol. 19, pp. 1-14, 2003.

[153] A. Lasfar, S. Mouline, D. Aboutajdine, and H. Cherifi, "Content-based retrieval in fractal coded image databases," Pattern Recognition, 2000. Proceedings. 15th International Conference on, 2000.

[154] J. M. Marie-Julie and H. Essafi, "Digital image indexing and retrieval by content using the fractal transform for multimedia databases, "Research and Technology Advances in Digital Libraries, 1997. ADL '97. Proceedings., IEEE International Forum on, 1997.

[155] M. Nappi, G. Polese, and G. Tortora, "FIRST: Fractal Indexing and Retrieval SysTem for Image Databases," *Image and Vision Computing*, vol. 16, pp. 1019-1031, 1998.

[156] M. Pi, M. Mandal, and A. Basu, "Image retrieval based on histogram of new fractal parameters," Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on, 2003.

[157] M. Vissac, J.-L. Dugelay, and K. Rose, "A fractals-inspired approach to content-based image indexing,"Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on, 1999.

[158] Z. Wang, Z. C. and, and D. Feng, "Content-based image retrieval using block-constrained fractal coding and nona-tree decomposition," *IEE Proceedings - Vision, Image and Signal Processing*, vol. 147, pp. 9-15, 2000.

[159] J. S. Lee, "Digital image enhancement and noise filtering by use of local statistics," *IEEE Transactions on Pattern and Machine Intelligence*, vol. 2, pp. 165-168, 1980.

[160] S. K. Mitra, a. C. A. Murthy, and M. K. Kundu, "A technique for image magnification using partitioned iterative system," *Pattern Recognition*, vol. 33, pp. 1119-1133, 2000.

[161] D. L. Donoho and I. M. Johnstone, "Ideal spatial adaptation via wavelet shrinkage," *Biometrika*, vol. 81, pp. 425-455, 1994.

[162] M. Crouse, R. Nowak, and R. Baraniuk, "Wavelet-based statistical signal processing using hidden Markov models," *IEEE Transactions on Signal Processing*, vol. 46, pp. 886-902, 2000.

[163] S. G. Chang, B. Yu, and M. Vetterli, "Spatially Adaptive Wavelet Thresholding for Image Denoising and Compression," *IEEE Transactions on Image Processing*, vol. 11, pp. 670-684, 2002.

[164] J. L. Starck, E. J. Candes, and D. L. Donoho, "The Curvelet Transform for Image Denoising," *IEEE Transactions on Image Processing*, vol. 11, pp. 670-684, 2002.