FRACTAL IMAGE COMPRESSION USING PYRAMIDS

by

Huawu Lin

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Electrical and Computer Engineering
University of Toronto

©Copyright by Huawu Lin 1997
The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-27682-1
Abstract

Fractal image compression is an attractive technique for image coding because of its distinct features and the low bit-rate requirement. In this research, several techniques are introduced to improve the compression performance. Fractal image compression is based on the self-similarity search of the image. The encoding process is computationally intensive. In this thesis, a pyramidal framework is proposed to reduce the encoding complexity. The encoding complexity is reduced by as much as two orders of magnitude. Because the domain-range matching is independent, parallel methods are proposed to further speed up the encoding process.

As with any lossy compression schemes, one of the challenges is either to maximize the image quality at a fixed bit rate, or to minimize the rate required for a given quality. To fulfill this purpose, the constant contractive factor in conventional fractal image compression is extended to the case of nonlinear contractive functions, which lead to significantly better reconstructed images and faster decoding than the conventional fractal method. Furthermore, as digital images and video products are designed for human viewing, human visual system (HVS) models are exploited to faithfully reproduce perceptually important information and eliminate the information that the visual system cannot perceive. Based on the human visual system’s nonlinear response to luminance and the visual masking effects, a perceptually appropriate metric is defined. The psychophysical raw data on the visual contrast threshold is first interpolated as a function of background luminance and visual angle, and is then used as an error upper bound for perceptually based fractal image compression. The perceptually based method produces visually better reconstructed images than the conventional method and the JPEG standard. To reduce the domain search complexity, the pyramidal search method is also extended for perceptually based fractal image compression.

Practical fractal image compression is a block coding. At very low bit rates, fractal method encoded images may exhibit blocking artifacts. Based on the Laplacian pyramidal representation of the image, the thesis presents a general post-processing method to remove the blocking effects.
Acknowledgments

I wish to express my sincere gratitude and appreciation to my thesis supervisor, Prof. A. N. Venetsanopoulos for his guidance, advice, encouragement and the financial support throughout the course of this research.

I deeply appreciate the invaluable comments and helpful suggestions from the following members of my thesis committee, Prof. F. R. Kschischang, Prof. S. Panchanathan, Prof. S. Pasupathy, Prof. M. L. G. Joy, Prof. S. A. Argyropoulos, Prof. D. Hatzinakos, and Prof. R. A. Ross.

I would like to thank Dr. John Ross at the University of Toronto Instructional and Research Computing (UTIRC) for many helpful guides to parallel programming. I wish to thank to Ido Rabinovitch, and Dimitrios Androutsos for improving the readability of my writing. My thanks are also extended to Mary Stathopoulos for her constant office support.

I gratefully acknowledge the financial support by the University of Toronto in the form of the Graduate Open Fellowship, by the Ontario Information Technology Research Centre (ITRC) in the form of ITRC scholarship.

Last but not least, I wish to thank my wife Eva and daughter Lucy who have, as always, been a source of encouragement, and have helped me to keep my priorities straight.
# Contents

1 Introduction
   1.1 Digital Image Compression and Techniques .......................... 1
   1.2 Fractal Image Compression ........................................... 7
   1.3 Problems and Our Approaches ....................................... 13
   1.4 Organization of the Thesis ......................................... 14

2 Fractal Image Compression and Extensions ............................... 16
   2.1 Mathematical Background ........................................... 16
      2.1.1 Binary Image Compression ...................................... 16
      2.1.2 Greyscale Image Compression .................................. 21
   2.2 Fractal Coding with Nonlinear Contractive Functions ............... 28
   2.3 Parallel Implementation of Fractal Image Encoding .................. 34
      2.3.1 Three Parallel Encoding Structures ............................ 34
      2.3.2 Parallel Implementation of the Domain Block Search .......... 34

3 Speeding up Fractal Image Encoding Using Pyramids .................... 39
   3.1 Introduction .......................................................... 39
   3.2 The Image Pyramidal Data Structure .................................. 40
   3.3 Fast Pyramidal Domain Block Search .................................. 43
   3.4 Determining the Thresholds .......................................... 46
   3.5 Computational Efficiency ............................................ 49
   3.6 Coding Results ....................................................... 51
# Perceptually Based Fractal Image Compression

4.1 Introduction ........................................... 57
4.2 The Human Visual System (HVS) Model .................. 58
4.3 A Perceptually Meaningful Image Metric ............... 59
4.4 The Visual Contrast Threshold .......................... 61
4.5 Results and Comparisons ................................ 64

# A Pyramidal Algorithm for Perceptually Based Fractal Image Compression

5.1 Introduction ........................................... 72
5.2 Fast Pyramidal Domain Block Search .................... 73
5.3 Determining the Thresholds ............................. 75
5.4 Computational Efficiency ................................ 82
5.5 Coding Results ........................................ 83

# Post-processing and Conclusions

6.1 Removing the Blocking Effects Using Laplacian Pyramidal Filtering .... 87
  6.1.1 Introduction ........................................ 87
  6.1.2 Pyramidal Generation for Quadtree Blocks ......... 88
  6.1.3 The Laplacian Pyramidal Filtering.................. 90
  6.1.4 Experimental Results ................................ 92
  6.2 Conclusions ........................................... 95
  6.3 Recommendations ..................................... 96

Appendix A The Coarse Pyramidal Level Thresholds ........... 99
Appendix B Bivariate Probability Density Function of $X_n$ and $X_{n+1}$ .... 102
Appendix C Probability Density Function of $E^{(m)}$ ........... 105
Appendix D Relationship Between Correlation Coefficients $\rho_E$ and $\rho$ .... 109
Appendix E Probability Density Function of $X_n^{(k)}$ ........... 111
Appendix F Probability Density Function of $E^{(k)}$ ........... 114
Appendix G Relationship Between Correlation Coefficients $\rho_{L_n}$ and $\rho_{L_n'}$ .... 116
List of Figures

2.1 Binary contractive mapping ................................................................. 18
2.2 A 512 × 512 binary fern image ......................................................... 20
2.3 Local self-similarity. Two similar blocks are highlighted. ................. 23
2.4 A contractive mapping w from domain D to range R .......................... 23
2.5 Encoding a range block ................................................................. 24
2.6 Eight symmetrical mappings ........................................................... 25
2.7 Nonlinear contractive mapping ......................................................... 29
2.8 Original 512 × 512 Lenna image ..................................................... 31
2.9 Range blocks .................................................................................. 31
2.10 Nonlinear contractive functions encoded image, 0.1997 bpp, 31.05 dB ... 32
2.11 Constant contractive factors encoded image, 0.2001 bpp, 30.45 dB ....... 32
2.12 Fractal decoding: PSNR as a function of the number of iterations ..... 33
2.13 Partition the domain locations into n tiles. Each point represents the upper left corner of a domain block. ..................................................... 35
2.14 Parallel domain block searches ........................................................ 36
2.15 Execution time as a function of the number of processors .................. 37
2.16 Speed-up achieved by KSRl ............................................................. 37
2.17 Original 512 × 512 Peppers image .................................................. 38
2.18 Parallel encoded image, 0.2404 bpp, 31.72 dB .................................. 38
3.1 A pyramid of Lenna image .............................................................. 44
3.2 Range and domain blocks on a two-level pyramid ............................. 45
3.3 Refinement of fractal codes from coarse to fine pyramidal levels ......... 47
3.4 Matching error (D_i - R_i) distribution ............................................. 48
Promising domain locations in each pyramidal level for a 32 x 32 range block, where \( k \) is the pyramidal level index for the range block. "*" is the closest matched domain block location.

Full search encoded image, 0.2317 bpp, 30.91 dB

Pyramidal search encoded image, 0.2316 bpp, 30.62 dB

Full search and pyramidal search time as a function of the search step size

Full search encoded image, 0.2378 bpp, 31.12 dB

Pyramidal search encoded image, 0.2375 bpp, 30.57 dB

Parameters involved in the calculation of visual angle

Blackwell's visual contrast threshold

The least-squares surface fitting of Blackwell's visual contrast threshold

Threshold surface of the Lenna image

Perceptually based fractal image compression, CR = 11.01, PSNR = 35.5 dB

Conventional fractal image compression, CR = 9.00, PSNR = 36.4 dB

JPEG image compression, CR = 8.17, PSNR = 37.8 dB

Perceptually based fractal image compression, CR = 33.64, PSNR = 30.9 dB

Conventional fractal image compression, CR = 33.61, PSNR = 31.3 dB

Original 512 x 512 Chart image (Courtesy of IMAX Corp.)

Encoded image, CR = 8.00, PSNR = 36.0 dB

Bivariate Laplacian PDF part I: The amplitude of the impulse function

Bivariate Laplacian PDF part II: Bivariate double-sided exponential function

Full search encoded image, CR = 26.3, PSNR = 30.5 dB

Pyramidal search encoded image, CR = 26.3, PSNR = 30.2 dB

Full search encoded image, CR = 24.6, PSNR = 30.5 dB

Pyramidal search encoded image, CR = 25.3, PSNR = 30.2 dB

Quadtree partition of image into range blocks

Four blocks of a quadtree at pyramidal level \( k \) and inner boundary conditions

An enlarged part of the fractal encoded image, 0.1515 bpp

After post-processing

Nonlinear contractive functions encoded image, 0.1515 bpp, 30.2 dB
6.6 After post-processing ........................................ 93
6.7 Conventional $L_2$ norm fractal encoded image, 0.1635 bpp, 30.0 dB .... 94
6.8 After post-processing ........................................ 94
# List of Tables

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>IFS code for the fern image</td>
<td>20</td>
</tr>
<tr>
<td>2.2</td>
<td>PSNR as a function of the number of iterations</td>
<td>33</td>
</tr>
<tr>
<td>3.1</td>
<td>Experimental results for the Lenna image</td>
<td>52</td>
</tr>
<tr>
<td>3.2</td>
<td>Experimental results for the Peppers image</td>
<td>52</td>
</tr>
<tr>
<td>4.1</td>
<td>Comparison of results for the 512 × 512 Lenna image</td>
<td>66</td>
</tr>
<tr>
<td>5.1</td>
<td>Experimental results for the Lenna image</td>
<td>83</td>
</tr>
<tr>
<td>5.2</td>
<td>Experimental results for the Peppers image</td>
<td>84</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Digital Image Compression and Techniques

Digital image technology is transforming our expectations of computers and communications. The current emphasis is on bringing a new visual dimension to communications. While it may be true that an image can be worth a thousand words, it also imposes enormous demands on data handling, storage, and transmission. The reason for the interest is clear: representing images in digital form allows visual information to be easily manipulated in useful ways. This fact, combined with the development of image processing hardware and software, has resulted in the use of digital imaging systems in such diverse fields as broadcasting, facsimile transmission, computer communications, teleconferencing and desktop multimedia publishing. However, one major problem with digital images is the large number of bits required to represent them. For example, a single digital image of size $512 \times 512$ with 24 bits/pixel requires 768 Kbytes of computer storage. Transmission of the image using a 19200 bits/s modem takes approximately 5.5 minutes, which is unacceptable for most applications.

Fortunately, digital images generally contain a significant amount of redundancy. For example: (1) Still images contain spatial redundancy, which is due to the correlation between neighbouring pixels; (2) Colour images contain spectral redundancy, which is due to the correlation between different colour components (R, G, B); (3) Video contains
temporal redundancy, which is due to the correlation between different frames. Image compression aims to reduce the number of bits required to represent an image by removing these redundancies. On the other hand, as digital images and video products are designed mainly for human viewing, human visual system (HVS) models can be exploited to better reproduce perceptually important information at the expense of less important aspects of the image. Image compression techniques can be categorized as lossless or lossy depending on whether they are reversible or irreversible. The following is an overview of some popular compression techniques.

**Predictive Coding - DPCM** [1]
Differential pulse code modulation (DPCM) achieves compression by exploiting local correlation in the image spatial domain. In this technique, a prediction of a current pixel is formed from neighbouring, already encoded pixel elements. The prediction error is then quantized and encoded. For highly correlated images, a well designed predictor will produce a prediction error that has small dynamic range. The variance reduction is the basis of the DPCM coding method. Since the prediction difference has a smaller variance than the original image, if we reduce the number of bits available to encode each pixel, then the DPCM quantization error will be less than that of PCM. Under the absence of the prediction error quantizer, DPCM coding can be made lossless. Among various encoding schemes, DPCM has the lowest complexity, and can provide compression ratios of about 2:1 to 3:1 with minor distortion. DPCM is extensively used in telecommunications for speech coding, and in visual communications for image coding, such as in digital cameras.

**Transform Coding** [2]
Image transforms perform linear operations, which convert image data into a set of transform coefficients. The transform coefficients can be converted back to the original image by applying the corresponding inverse transform. Transform coding is not applied to the entire image, but rather over fixed blocks, typically of size 8 x 8 or 16 x 16. Orthonormal transformations, such as the Karhunen-Loeve transform (KLT), the discrete cosine transform (DCT), the Walsh-Hadamard transform (WHT), and the discrete Fourier
transform (DFT), etc., have been found useful for image coding since they often offer an energy compact representation of the underlying data. Studies based on natural images have shown that in the transform domain, most of the energy of an image is concentrated in the low frequency region. Consequently, zero and small-valued, high frequency coefficients are discarded and the rest of the coefficients are scaled and quantized to form a compact representation of the original image. For stationary image data, KLT is optimal in terms of energy compactness. The KLT basis functions are image-dependent and require to estimate the autocorrelation matrix of the source. In particular the rows of the KLT are the eigenvectors of the this matrix. However, this transformation has no fast algorithms in general. As an alternative, DCT has fast algorithm implementation and provides a performance close to that of the KLT, when the pixel-to-pixel correlation is high.

DCT has been used in image compression standards, such as JPEG (the Joint Photographic Experts Group) [3] from the CCITT (the International Consultative Committee for Telephone and Telegraph) and ISO (International Standardization Organization) for still images, H.261 [4] from the CCITT and H.263 [5] from ITU (International Telecommunication Union) for video teleconferencing, and MPEG-1 and MPEG-2 (the Moving Picture Experts Group) [6] from ISO for video. In the generic video source coding standards, H.261 could deliver px64 Kbits/s. The intended applications were videophone and videoconferencing. The coding algorithm of H.263 is similar to that used by H.261, however with some improvements and changes for better performance and error recovery. MPEG-1 dealt with audio and video coding at a bit rate up to about 1.5 Mbits/s, using a progressive format, and was mainly dedicated to storage applications. MPEG-2 was optimized to enable digital broadcasting for both normal definition television at a bit rate of about 4-9 Mbits/s, and high definition television (HDTV) at 15-25 Mbits/s in an interlaced format. The JPEG still image compression standard consists of several parts, including both lossless and lossy encoding. The lossless compression uses the predictive/adaptive model described earlier in this section, with a Huffman code output stage. The JPEG lossy compression algorithm operates in three successive steps, i.e., DCT transformation, coefficient quantization and entropy encoding. A production-quality JPEG compressor targets low compression ratios from 4:1 to 6:1 [7].
Since compression is obtained mainly by discarding high frequency coefficients, at low bit rates (at a compression ratio of about 20:1 or higher), the image loses its edge detail and also its blocking artifacts become visible. To improve the performance, the new JPEG-2000 standard [8] is under proposal. The new standard will cover bi-level, greylevel and colour images. It will provide a lower bit-rate mode of operation than the existing JPEG standard.

Vector Quantization [9]
The basic idea of vector quantization (VQ) originates from the early finding of Shannon that more efficient coding can always be achieved by quantizing the output of a source as vectors rather than scalars. The image is first partitioned into small blocks, typically 4 × 4, then each block is compared with a set of representative vectors. The representative vector is called a codeword and the set is called a codebook. The codebook is generally generated by using a training set of images that are representatives of the image to be encoded. For the closest match, an index corresponding to the chosen vector is transmitted to the receiver, which has access to the same codebook. A lookup table can be used by the receiver to reconstruct the encoded vector. Compression is achieved when the number of codebook entries is less than the total number of possible input vectors. Apparently, the codebook search in the encoding stage is computationally intensive, while the table lookup search for decoding is simple. The performance of VQ depends on the codebook. Obviously, to encode a particular image, the optimal codebook would be generated by using the image itself as the training set. If the images to be encoded differ greatly in terms of edges, lines, textures, etc., the performance of VQ will be substantially degraded. On average, VQ gives a better performance than transform coding, especially at bit rates of about 0.25 bpp and lower.

Wavelets Coding [10, 11]
Wavelets refer to a set of orthonormal basis functions, which are derived from translations and dilations of a single function called the "mother" wavelet. The basic idea of the discrete wavelet transform (DWT) is to decompose the input signal into a coarse approximation signal, which is equivalent to lowpass filtering and sub-sampling, and an
additional detail signal, which is equivalent to highpass filtering and sub-sampling. The procedure is then repeated a number of times by decomposing the approximation into a coarser approximation and another detail signal, and so on. Consequently, the DWT decomposes the input signal into one lowpass approximation and a set of bandpass frequency channels. Compression is obtained by keeping only the most relevant coefficients within each channel. Reconstruction is the inverse of the decomposition process. After up-sampling, each channel signal passes through its synthesis filter. Summation of the resulting individual channels leads to the reconstruction of the image. Compared with DCT, which exploits the low frequency aspects at the expense of high frequency aspects, DWT has both local spatial and frequency domain properties due to the compact support of its base function. At low bit rates (about 0.25 bpp), wavelet coding gives a better performance than transform coding. The technique is exempt from blocking effects, which may occur with blocking encoding schemes. Instead, it produces a ringing effect along the edges due to the Gibbs phenomenon.

**Model Based Coding [12]**

In model based coding, the 2-D perspective projection of a 3-D scene is expressed by a semantic model. Image analysis techniques can be used to generate feature parameters, which, when transmitted to the decoder, can be used for the image synthesis [13]. The analysis process is extremely difficult for natural scenes due to the arbitrary contents. On the other hand, the synthesis is relatively easier because image synthesis techniques for computer graphics have already addressed the problem. Currently, most research is concentrated on the head-and-shoulder image as is the case with video conferencing. A wire frame model is commonly used to render the shape of the subject, in conjunction with texture mapping. On the other hand, a fractal model can be used to render the outdoor natural scenes, such as mountains and landscapes [14]. For example, the texture area can be reconstructed with an overall similarity to the original one by generating a statistical fractal surface from the fractal dimension parameter [15]. As the research topic of this thesis, fractal method is discussed in the next section and throughout the thesis. To encode image sequences, changing parameters and locations are sent to the decoder as updates. Provided that the subject changes and facial motions are not too drastic, lifelike
image synthesis can be obtained. Despite its vast complexity, the model based coding appears very attractive for very low video bit rates from 10 Kbits/s to 64 Kbits/s, as in the case of MPEG-4 [16, 17]. The main applications include videophone, video database retrieval, games, etc. Recently, a new member of the MPEG family, MPEG-7 (formally called 'Multimedia Content Description Interface') [18], is under development. MPEG-7 is a content representation standard for information search. It allows fast and efficient search for the audiovisual information.

**Entropy Coding [19]**

Among various coding techniques, encoders often generate a set of code parameters which have nonuniform probability distributions. For example, the histograms of DPCM prediction error usually show a high peak at zero, and low frequency transform coefficients tend to contain most of the signal energy. By taking advantage of the nonuniform distribution of the underlying data, entropy coding such as Huffman coding, run-length coding and arithmetic coding can achieve further data compression. In Huffman coding, symbols with a lower probability of appearance are encoded with a codeword using more bits. Symbols with a higher probability of appearance are represented with a codeword using fewer bits. For highly correlated symbol streams, Huffman coding is often combined with run-length coding. Run-length coding takes advantage of the fact that several nearby symbols in the data will, statistically, tend to have the same value. This form of redundancy can be reduced by grouping symbols of the identical value into an intermediate symbol, which consists of one symbol and one run-length integer. The intermediate symbols are then Huffman encoded. In Huffman coding, there is a one-to-one correspondence between the codeword blocks and the source sequence blocks. In comparison, arithmetic coding is a tree code, where a codeword is assigned to the entire input string. It works by relating a source output string to an interval between 0 and 1. As the message becomes longer, the interval needed to represent it becomes smaller, and the number of bits needed to specify that interval increases. Successive symbols in the message reduce the interval in accordance with the probability of that symbol. The more likely symbols reduce less the range, and thus add fewer bits to the encoded message. The performance of arithmetic coding is at least as
good as that of the Huffman code since for each Huffman code there exists a corresponding arithmetic code with the same efficiency [20]. Due to its capability of using a fractional bit, arithmetic coding usually outperforms the Huffman code. Entropy coding is noiseless coding, i.e., the original data can be exactly recovered from the compressed data.

1.2 Fractal Image Compression

During the last decade, fractal geometry has captured increasing attention and interest. Fractal, a term coined by Mandelbrot, refers to a rough or fragmented geometric shape, each part of which is at least approximately a reduced size copy of the whole. Fractal objects are generally self-similar and independent of scale. There are many mathematical structures that are fractals, e.g., Mandelbrot set, Sierpinski triangle, Koch snowflake, Peano curve, etc. The Mandelbrot set is described by a very simple equation yet it exhibits an infinite variety of details. This may be considered as one form of compression: the equation itself can be represented by a few bits of information or implemented in a very short computer program, but the resultant image would need a large amount of bits, described as a set of pixels. Mandelbrot did not actually consider fractals for compression, but he showed that they could be used as an approximate description of many real world objects such as coastlines, mountains, trees, clouds, turbulence, that do not correspond to simple geometric shapes. Such objects reveal more details whenever you look closer at them. Mandelbrot's book The Fractal Geometry of Nature [21], first published in 1977, attracted a wide-ranging attention. The images generated by fractal modelling were very realistic and the fractal technique is now commonly used to generate computer graphics.

Barnsley and his coworkers at the Georgia Institute of Technology were among the first to recognize the potential interest of the fractal method for image compression. Barnsley developed the theory of iterated function system (IFS), a concept first introduced by Hutchinson in 1981 [22]. After the publication of Barnsley's book Fractals Everywhere [23] in 1988, and his paper in the January 1988 issue of Byte magazine [24], fractal compression gained considerable popularity. The interest in this technique was aroused by the large compression ratio claimed by Barnsley, up to 10000 to 1 [24]. Together with Alan Sloan, Barnsley founded Iterated Systems, Inc. Unfortunately, the
large compression ratios could only be obtained for specially constructed images, and through human intervention. For example, a person determines that a branch of a tree can be viewed as a reduced copy of the whole tree, possibly distorted in a certain way. The person guides the compression process by segmenting the original image in such a way that each segment looks like a reduced copy of the whole image and the union of all the segments approximates the original image as closely as possible. The process is sometimes known as the "graduate student algorithm," which consists of giving a graduate student an office and a workstation, locking the door, waiting until the student has found a good IFS for the image, and opening the door. It was not possible to completely automate the compression process, even with a supercomputer. Thus, IFS-based compression turned out to be impractical. A breakthrough was made in 1988 by Jacquin [25, 26, 27], one of Barnsley's Ph.D. students. Instead of trying to find an IFS for a whole image, Jacquin had the idea of partitioning the image into non-overlapping range blocks and finding a local iterated function system (LIFS) for each range block. This approach transformed the encoding problem into a manageable task, which could be automated. LIFS is also named partitioned iterated function system (PIFS), and Barnsley uses the term "fractal transform." They all actually refer to the same technique.

Fractal image compression is based on the observation that all real world images are rich in affine redundancy. That is, under suitable affine transformation, large blocks, known as domain blocks, of the image look like small blocks, known as range blocks. In other words, a range block can be approximated by an affine mapping of a well chosen domain block. Since the mapping shrinks the domain block, it is called a contractive mapping. Images with global self-similarity can be encoded with extreme efficiency by solving the inverse problems [28, 29, 30]. However, general images are rarely globally self-similar. This property only exists locally among different small parts of the image. Thus, practical fractal coding is a block coding technique. First, the image is partitioned into non-overlapping range blocks. Similarly, the same image is divided into a set of overlapping domain blocks, which are larger in size than range blocks (in order to meet the contractive condition). The encoding of a range block amounts to finding an affine transform of the domain block which best fits in terms of the used image metric. The parameters of the affine transform, which include rotation/reflection, scaling, shift and the
location of the chosen domain block, constitute a fractal code of the range block. These affine maps usually give a compact representation of the original image. In general, ranges and domains can have arbitrary shapes instead of a square shape. Fisher and Menlove’s horizontal and vertical (HV) partition scheme used rectangles [31]. Davoine et al. presented a scheme based on triangles and quadrilaterals [32]. Reusens studied the issue of partitioning complexity and compared performances of partitions using, respectively, squares, rectangles and polygons [33]. Monro et al. extended the affine form of the fractal approximation to a second order polynomial [34]. A high order approximation improves the encoding fidelity but reduces the compression ratio because there are more parameters to encode. Morgan et al. applied shape recognition to fractal coding [35]. The technique is based on edge detection using Sobel operators to identify the high frequency components of an image, and using this information to guide the domain block search. Obrador et al. studied texture image compression using the fractal method [36].

To improve the encoding performance, other methods have been combined with fractal coding. In particular, Lin considered fractal coding as generalized predictive coding [37]. Kim and Hamzaoui et al. enhanced fractal coding by vector quantization [38, 39]. Thao proposed a hybrid fractal-DCT coding method [40]. Akujuobi et al. made a comparative study of fractals and wavelets based on their properties: self-similarity, geometrical objects representation, existence of scaling function, affine transformation, and the duality principle [41]. Considering the self-similarity observed in each subband, Belloulata et al. proposed to use the fractal method to encode the coefficients of the wavelet transform [42]. In their scheme, the original image is first decomposed into subbands containing information in different spatial directions and different scales, using finite impulse response (FIR) filters. Subbands are encoded using the fractal method, with range and domain blocks representing horizontal or vertical directionalities. The block sizes are defined according to the correlation length in each subband. Wall et al. suggested that a neural network paradigm, known as frequency sensitive competitive learning, can be employed to assist the encoder in locating fractal self-similarity searches [43]. Franich et al.’s method related the fractal coding to an object-based system [44]. Ida et al.’s work showed that fractal coding can be used for image segmentation [45]. For fractal colour image compression, the experiments of Hurtgen et al. indicated that encoding of the RGB
components resulted in a slightly better reconstruction quality compared to the YUV components [46]. Zhang et al.'s results showed that the use of vector distortion measurement would lead to 1.5 times compression improvement compared with separate fractal coding in the RGB domain [47]. All these modifications give significantly better results than JPEG and some wavelet methods, yet their main drawback is slower encoding.

Promising performances provided by fractal still image compression methods spurred researchers on to applying fractal theory to video coding. Beaumont suggested a direct extension of the 2-D approach to 3-D data volumes [48]. The encoded image, using this method, may have blocking artifacts in the quick moving areas. In order to reduce the encoding computation time, Li et al. proposed a 3-D approach without a domain block search but with increased contractive mapping complexity by using quadratic form [49]. Reusens presented a scheme where sequence volume is adaptively segmented along an octree structure and 3-D blocks are encoded either by contractive mapping or 3-D temporal block matching [50]. Lazar et al. followed the same approach, but allowing only contractive mapping [51]. Their results indicate that average compression ratios, ranging from 40 to 77, can be obtained with subjective reconstruction video-conferencing quality. Hurtgen et al. introduced a 2-D approach, where foreground regions are coded by intraframe fractal coding, while background regions are coded by a temporal DPCM-loop [52]. Kim et al. proposed a fractal video encoding method using extended circular prediction mapping, where each range block is approximated by a domain block in the adjacent frame [53]. Thus, the domain-range mapping is similar to the block matching algorithm in the motion compensation. Barthel et al.’s scheme combined the wavelet with the fractal method for video coding [54]. In paper [55], Monro et al. presented results of fractal colour video coding with a compression ratio of 220 to 1.

Fractal decoding is based on the fixed-point theorem [56]. The conventional greyscale image decoding method starts with an arbitrary initial image and successively applies the affine maps until the image converges. Kang et al. introduced a fast decoding method that uses the distribution of the domain location, and it uses the image, updated by block transformation, as an initial image [57]. A number of orthogonal basis decoding methods have eliminated the need for iterations [58, 59]. For fractal video decompression [60], the
current frame $I_n$ is derived from an iteration of the fractal code $W$ on the last frame $I_{n-1}$, i.e., $I_n = W(I_{n-1})$.

Fractal image compression has three distinct performance advantages over DCT-based techniques:

1. Resolution independence: Since the affine maps used to encode the image do not depend on its resolution, fractal encoded images are scaleable and resolution independent, allowing a fractal reconstructed image to be displayed at any resolution without file conversions. You can zoom in infinitely on the fractal image without obvious loss of detail, although after a point, that detail is mathematically predicted. It also provides an enhancing technique where the output of the compressed image has a higher resolution than the input. At a higher resolution, the image obtained through fractal decoding is much more natural-looking [61, 62]. The decoding process creates artificial detail which was not present in the original image, but which looks as if we had really zoomed in the original image. This enhancement is a useful feature, but it does have certain limits. If we try to zoom in at a huge scale factor, we will not end up seeing the molecules, but rather, we will be looking at details which are completely artificial.

2. Decompression speed: DCT based compression algorithms are symmetric, with decompression being the reverse of compression and requiring the same computational effort. This is the case when an image is transmitted from one person to another as in image interchange applications. By contrast, fractal compression is an asymmetrical scheme. Encoding is computationally intensive, while decoding is simpler and faster than DCT based methods [61, 63]. The low computation of decompression makes it possible to play back full-screen movies with synchronized sound at 30 frames per second on a standard 80486 using software alone [64]. Asymmetrical coding is mainly used where an image is compressed once and decompressed many times. This is the case when an image is transmitted to many different users, such as video-on-demand, CD-ROM storage and other image-distribution applications, where each one needs to decompress that same image.

3. High coding efficiency: Fractal coding can achieve its highest compression ratio for images which are inherently self-similar or self-affine. For general images, the coding methods have better compression efficiency than the JPEG standard both at perceptually
lossless quality level (about 0.5-1 bpp) and at the low bit rates (about 0.2 bpp). A fractal video encoder also has a better encoding efficiency than MPEG and wavelets [65].

The applications of data compression are primarily in the storage and transmission of information. The choice among the compression methods depends on the applications required. With the adoption of the DCT-based standards, such as JPEG and MPEG, other compression methods require better performances or lower costs to win an audience.

Fractal image compression is asymmetric. The encoding is computationally intensive, while decoding is simple. In storage applications, the requirements are less stringent than in transmission applications because much of the preprocessing can be done off-line. Fractal software video decompression with existing hardware is much cheaper than the standards [63] and could be used for quick access of the compressed image data base. For example, the "Microsoft Encarta" multimedia CD-ROM has used fractal image compression [56]. In transmission applications, compression techniques are constrained by real-time and on-line considerations that limit the size and complexity of the hardware.

Field programmable gate array (FPGA) chips, digital signal processor (DSP) chips and other very large-scale integrated circuit (VLSI) chips may be programmed or designed for real-time applications. For the HDTV broadcasting application, the use of fractal technique means a complex transmitter and simple receivers. Barnsley and Sloan's company, Iterated Systems Inc., sells fractal based software and hardware products. In 1994, Iterated Systems Inc. reported that their real-time fractal based communication system was used to transmit reconnaissance type video over SIGCGARS radio and cellular phone networks [66]. New products also incorporated image searching ability into the fractal compression [65]. However, they have not made public the details of their technology. In particular, the fractal image format (FIF) used by Iterated Systems products has not been publicly described. Possibly because of this, and of the patents attached to the method, fractal image compression is not yet used in practice as much as other techniques. However, fractal compression is still a subject of active research, and the technique has already demonstrated its superiority at least for applications where a very high compression efficiency is required, and it is feasible to carry out a large number of computations for each image to be compressed.
1.3 Problems and Our Approaches

In the following, we describe three problems for fractal image compression and present our approaches toward finding solutions.

1. During the encoding process, the fractal compression algorithm has to search through the domain block pool to find the best matched domain block under an affine transform. This kind of self-similarity search is very computationally intensive as compared to the JPEG algorithm. The high encoding complexity is the main drawback of the fractal image compression. In this thesis, we first propose three parallel implementation structures to speed up the process [67, 68]. However, its complexity remains the same. We next propose a fast encoding algorithm based on the pyramidal representation of the image [69, 70, 71, 72]. Assuming that the distribution of the matching error is described by an independent, identically distributed (i.i.d.) Laplacian random process, we derive the threshold sequence for the objective function in each pyramidal level. The search is first carried out on an initial level of the pyramid. This initial search increases the encoding speed significantly, because not only is the number of domain blocks to be searched, reduced, but also the data within each domain block is only $1/4^k$ of that in the finest level, where $k$ is the pyramidal level. Only a few of the fractal codes from promising domain blocks in the coarse level are then refined through the pyramid to the finest level with little computational effort. The algorithm is quasi-optimal in terms of minimizing the mean square error. Computational efficiency depends on the depth of the pyramid and the search step size, and could be improved up to two orders of magnitude compared with the full search of the original image.

2. As with any lossy compression schemes, one of the challenges is either to maximize the image quality at a fixed bit rate, or to minimize the rate required for a given quality. Our second objective targets to improve the coding fidelity at a given bit rate. To fulfill this purpose, the constant contractive factor in conventional fractal image compression is extended to the case of nonlinear contractive functions [73, 74]. Furthermore, HVS model is incorporated into the fractal image compression [75, 76, 77]. Compared with transform coding, fractal coding is performed in the spatial domain. According to the collage theorem [56], the reconstruction error is directly related to the metric used in the encoding process. The conventional fractal encoding uses a mean
square error metric as an optimization criterion, which correlates poorly with the human visual response to errors. As digital images and video products are designed for human viewing, human visual system models can be exploited to faithfully reproduce perceptually important information and eliminate the information that the visual system cannot see. Specifically, when a compressed image cannot be distinguished visually from the original under certain viewing conditions, the compression is said to be perceptually lossless. According to Weber's law, it is the image contrast, and not the linear difference, that determines the visibility of luminance. In this research, we introduce a perceptually meaningful distortion measure based on the human visual system's nonlinear response to luminance and the visual masking effects. Blackwell's psychophysical raw data [78] on the contrast threshold is first interpolated as a function of background luminance and visual angle, and is then used as an error upper bound for perceptually based image compression. Experimental results show that both at the low bit rate and high bit rate, the perceptually based fractal method produces a better reconstructed image than JPEG. To reduce the domain search complexity, we also extend the pyramidal search method for perpetually based fractal image compression [79].

3. The third problem is common in block coding schemes such as transform coding and vector quantization. At the very low bit rate (about 0.2 bpp), the fractal method encoded images often have blocking artifacts. Based on the Laplacian representation, the thesis presents a multichannel filtering method to remove the blocking effects [80, 81]. At the root of the quadtree, four children blocks are first decomposed into a set of bandpass channels, i.e., Laplacian pyramids. Each channel is then filtered at the block boundary. Finally, the image is reconstructed from these channel components. Experimental results show that the algorithm yields superior images compared to those obtained by simple lowpass filtering.

1.4 Organization of the Thesis

This thesis consists of six chapters and seven appendices which are organized in the following way.

The mathematical background and modifications of the basic algorithm are given in Chapter 2. In particular, the concept of contractive mapping, fixed point theorem and
collage theorem are introduced both for binary and greyscale images. Section 2 provides a modification of the basic algorithm by introducing nonlinear contractive functions, which lead to significantly better reconstructed images and faster decoding than the conventional fractal method. The last section describes three parallel structures for fast encoding.

Chapter 3 is devoted to the fast pyramidal algorithm. In this chapter, we define the pyramidal framework for fast block matching, ranging from the definition of fractal parameter propagation rule, thresholds for the determination of promising matching locations to computational efficiency analysis. The comparison of results with a full search method is tabulated, and compressed sample images are included.

Chapter 4 covers perceptually based fractal image compression. It starts with a review of the HVS model. Then, a perceptually appropriate metric is defined based on the HVS nonlinear and masking effect. Next, visual contrast thresholds are derived based on the early psychophysical data. Finally, the gain factor is introduced as a control parameter of compression quality. The comparisons of the results with the conventional fractal method and the JPEG standard are presented.

Chapter 5 presents an extension of the pyramidal algorithm to the perceptually based compression method. The parameter propagation rule is redefined. The threshold sequence is also rederived based on the more accurate image model, i.e., Markov random process. The derived mathematical results such as the threshold sequence, the bivariate Laplacian probability model and the correlation model etc. can also be used in other fields, such as multiresolution image processing, signal estimation and statistical model-based coding.

Chapter 6 contains the post-processing method and the conclusions of the thesis. Our postprocessing philosophy is that it should be matched to the partition scheme (i.e., quadtree, in our case), and the human visual response to different frequency bands, so as to provide the best overall performance. Based on the Laplacian pyramidal representation, a multichannel filter is designed to remove the blocking effects.

The last sections summarize the contributions of the research and present some possible extensions and ideas for future research. Proofs of mathematical results are given in detail in appendices that support the discussion at the corresponding parts of the thesis.
Chapter 2

Fractal Image Compression and Extensions

2.1 Mathematical Background

The use of fractal geometry has largely been in the domain of computer graphics. Many examples of synthetically generated models of complex natural scenes such as mountains and landscapes are given in Mandelbrot's book "The Fractal Geometry of Nature" [21]. Fractal geometry, however, can be effectively applied to the image processing and analysis, as well. The application of fractal models to image compression has been advocated by Barnsley et al. For historical reasons, fractal coding was first applied to binary images. In the following, an introduction to binary image compression is also given to provide some useful concepts and insights.

2.1.1 Binary Image Compression

A metric space \((X, d)\) is a set of points \(X\), along with a function \(d\) that takes two elements of the set and gives the distance between them. It is a complete metric space if the metric space does not have "holes" or values missing. Let \((X, d)\) be a complete metric space, \(X \subset R^2\), where \(R^2\) denotes the Euclidean plane. A binary image can be represented by a compact (i.e., closed and bounded) subset of \(X\). For example, \(X\) can be a rectangular...
subset of $R^2$, corresponding to an image window. $d(x, y)$ is the Euclidean distance defined for $x, y \in X$. A concept central to fractal coding is the contractive mapping.

**Definition 2.1** A map $w: X \rightarrow X$ on a metric space $(X, d)$ is said to be contractive if there exists a constant $0 \leq s < 1$, such that

$$d(w(x), w(y)) \leq s \ d(x, y)$$

(2.1)

for all $x, y \in X$, where $s$ is called the contractive factor of the map. Equation (2.1) means that the distance between $w(x)$ and $w(y)$ is no greater than the distance between $x$ and $y$. Similarly, a set of contractive maps can be defined $w_i: X \rightarrow X$, with contractive factors $0 \leq s_i < 1$, $i = 1, ..., N$. This set of maps with its contractive factor $s = \max\{ s_i, i = 1, ..., N \}$ is called an iteration function system (IFS), denoted by $\{X, w_i: i = 1, ..., N\}$. In the 2-D Euclidean plane, the special case map $w_i$ takes the form of an affine transformation:

$$w_i \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \end{bmatrix}$$

(2.2)

or $w_i(X) = A_i X + T_i$, where

$$A_i = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix}$$

(2.3)

represents affine scaling (shearing) and rotation, and

$$T_i = \begin{bmatrix} e_i \\ f_i \end{bmatrix}$$

(2.4)

is a translation vector. Thus, each $w_i$ is described by six numbers and is contractive if and only if $|\text{det}(A_i)| < 1$. Figure 2.1 shows a contractive affine map from domain $D$ to range $R$, which shrinks the area on which it operates.

By $H(X)$ we denote the set of all non-empty compact subsets of $X$. We can consider $H(X)$ as a collection of all black-and-white pictures, where a subset of the plane is represented by a picture that is black at the points of the subset and white elsewhere. The Hausdorff distance between sets $A$ and $B$ in $H(X)$ is defined by
Figure 2.1 Binary contractive mapping

\[ h(A, B) = \max\{\max_{x \in A} \min_{y \in B} d(x, y), \max_{x \in B} \min_{y \in A} d(x, y)\} \quad (2.5) \]

The union of the maps defined by the IFS constitutes a contractive map on the metric space \((H(X), h)\) according to the following theorem.

**Theorem 2.1** (Fixed point theorem for binary images) [56]

Let \(\{X, w_i : i = 1, ..., N\}\) be an IFS on a complete metric space \((X, d)\) with a contractive factor \(s\). Define \(W: H(X) \to H(X)\) by

\[ W(B) = \bigcup_{i=1}^{N} w_i(B) \quad (2.6) \]

for all \(B \in H(X)\), where the union can be explained as that the whole image is an assembly of reduced copies of its whole self. Then \(W\) is a contractive mapping on \((H(X), h)\), that is

\[ h(W(B), W(C)) \leq s \cdot h(B, C) \quad (2.7) \]

for all \(B, C \in H(X)\). The map \(W\) has a unique fixed point, known as the IFS attractor, \(A \in H(X)\), which obeys

\[ A = W(A) = \bigcup_{i=1}^{N} w_i(A) \quad (2.8) \]

and is given by
\[ A = \lim_{n \to \infty} W^n(A_0) \]  

for any \( A_0 \in H(X) \), where "\( \cdot^n \)" denotes the \( n \)-th iteration of the map \( W \), with the definition of \( W^n(A_0) = A_0 \), and \( W^{n+1}(A_0) = W(W^n(A_0)) \). That any non-empty set \( A_0 \in H(X) \) converges under iterations to an attractor \( A \) is important. It implies that the binary image \( A \) is defined by \( W \) only. Given a target image \( T \), the problem is to construct a set of contractive affine transforms \( w_i \), whose attractor \( A \) is close to \( T \) in the Hausdorff metric. This topic has become known as the "inverse problem." Barnsley's collage theorem provides an important guideline in solving the inverse problem [28].

**Theorem 2.2 (Collage theorem for binary images) [56]**

Let \( (X, d) \) be a complete metric space, \( T \in H(X) \) a given binary image, and \( \varepsilon \geq 0 \). Choose an IFS \( \{ X, w_i : i = 1, \ldots, N \} \) with the contractive factor \( 0 \leq s < 1 \), such that

\[ h(T, \bigcup_{i=1}^{N} w_i(T)) \leq \varepsilon \]  

(2.10)

where \( h \) is the Hausdorff metric. Then

\[ h(T, A) \leq \frac{\varepsilon}{1 - s} \]  

(2.11)

where \( A \) is the attractor of the IFS.

The theorem can be used in the following way. Given a binary image \( T \), one draws an outline of \( T \) and covers it as \( \varepsilon \) close by the union of smaller affine copies of the outline. For each one of the smaller copies, there is a unique affine map \( w_i \) from the outline onto it. The affine maps determined this way are the IFS codes. The collage theorem states that the attractor \( A \) of the IFS will be close to \( T \) in the sense of the Hausdorff metric (i.e., \( h(T, A) \leq \varepsilon/(1 - s)) \). Essentially, the collage theorem states that the more accurately the image is covered by contractive affine maps of itself, then the more accurately the IFS encodes the image. If a good collage has been found and the total number of bits used to represent the affine transformations is smaller than the total number of bits in the image, then encoding the mapping coefficients requires fewer bits than enumerating all pixel values. Generally speaking, fractal compression is lossy.
because the attractor of the IFS is a close approximation to the original image but not necessarily equal to it. Clearly, the images which are self-affine can be accurately represented with proper IFS. Other objects require an approximate solution in the form of an attractor which is sufficiently close to the target image.

Figure 2.2 shows the 512 × 512 binary fern image. It can be identified by four self-similar components: the major central branch, the major lower left leaf, the major lower right leaf, and the bottom part of the stem. Table 2.1 lists the parameters of the four affine transformations. The transformation $w_4$ squashes the fern flat to yield the stem. Assume that each coefficient uses 32 bits. Then the fern image can be encoded using 768 bits. The compression ratio for the 512 × 512 binary image is 341.3.

![Figure 2.2 A 512 × 512 binary fern image](image)

<table>
<thead>
<tr>
<th>$w$</th>
<th>$a$</th>
<th>$b$</th>
<th>$c$</th>
<th>$d$</th>
<th>$e$</th>
<th>$f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.85</td>
<td>0.04</td>
<td>-0.04</td>
<td>0.85</td>
<td>0.0</td>
<td>0.16</td>
</tr>
<tr>
<td>2</td>
<td>0.2</td>
<td>-0.26</td>
<td>0.23</td>
<td>0.22</td>
<td>0.0</td>
<td>0.16</td>
</tr>
<tr>
<td>3</td>
<td>-0.15</td>
<td>0.28</td>
<td>0.26</td>
<td>0.24</td>
<td>0.0</td>
<td>0.044</td>
</tr>
<tr>
<td>4</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.16</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>
An extension of the iterated function concept is the *fractal transform*. The fractal transform is a local iterated function system (LiFS) in which the maps \( w_1, w_2, \ldots, w_N \) are not applied to the whole image but are restricted to local domains. The \( w_i \) is specified not only by an affine map, but also by the domain to which it is applied. Obviously, the fractal transform is not a transform in the same sense as a Fourier transform or a discrete cosine transform. The fixed point theorem and collage theorem still hold. Such a system allows for simple encoding of more general binary shapes. Furthermore, it has been extended to compress greyscale images.

### 2.1.2 Greyscale Image Compression

One greyscale fractal image compression approach is to consider the greyscale image as a normalized Borel measure on its support, and to construct a Markov operator which has an invariant measure \([82]\). However, this approach can only be realized manually. From now on, we focus on applying the greyscale fractal transform for the purpose of automatic compression of greyscale images.

Let \( F \) be the space of a real world images, consisting of all real valued functions \( z = f(x, y) \) with its support \( S^2 \), \( f: S^2 \to I \), where \( I = [a, b] \subset R \) is a real interval that represents the grey levels in images. In order to form a complete metric space, we define two metrics \([83]\):

1. Supremum metric:

\[
d_{\sup}(f_1, f_2) = \sup |f_1(x, y) - f_2(x, y)|
\]  

(2.12)

With this metric, a map is contractive if it contracts in the \( z \) direction, i.e., \( |w(z_i(x, y)) - w(z_2(x, y))| \leq s \ |z_i(x, y) - z_2(x, y)| \).

2. \( L_2 \) metric:

\[
d = \left[ \int_{S^2} [f_1(x, y) - f_2(x, y)]^2 dx dy \right]^{\frac{1}{2}}
\]  

(2.13)

where \( f_1, f_2 \in F \).

This metric has more complicated contractivity requirements. In order to describe the contractivity of a map in the scene of the \( L_2 \) norm, we need the concept of the eventual contractive map \([84]\). A map \( W: F \to F \) is eventually contractive if there is a positive
integer \( n \) so that \( W^on \) is contractive. Unlike the supremum metric, the condition \( |s| < 1 \) is not sufficient to ensure contractivity for the \( L_2 \) metric, but it is a sufficient condition to ensure eventual contractivity. This metric is particularly useful since it can be easily minimized by a regression of the algorithm parameters.

**Theorem 2.3** (Fixed point theorem for greyscale images) [56]

Let \( W: F \rightarrow F \) denote a contractive transformation defined on the complete metric image space \( (F, d) \). That is, for all \( f_1, f_2 \in F \), \( d(W(f_1), W(f_2)) \leq s \, d(f_1, f_2) \), where \( 0 \leq s < 1 \) is the contractive factor of \( W \). In this case, there exists a unique image \( f \in F \) such that \( W(f) = f \). Moreover, for any initial image \( f_0 \in F \), the fixed point is the limit:

\[
f = \lim_{n \to \infty} W^{on}(f_0).
\]  

Let \( f_{\text{orig}} \in F \) be an original image to encode. The goal of fractal encoding is to construct a contractive transform \( W: F \rightarrow F \), for which \( f_{\text{orig}} \) is an approximate fixed point. The encoding error bound between \( f_{\text{orig}} \) and \( f \) is given by the following theorem.

**Theorem 2.4** (Collage theorem for greyscale images) [56]

Let \( W: F \rightarrow F \) be a contractive mapping with contractive factor \( s \). Let \( f_{\text{orig}} \) be an image to encode. Then

\[
d(f_{\text{orig}}, f) \leq \frac{1}{1 - s} \, d(f_{\text{orig}}, W(f_{\text{orig}}))
\]  

From the collage theorem we have two observations:

1. In order for \( f \) to be close to \( f_{\text{orig}} \), the mapping \( W \) should minimize \( d(f_{\text{orig}}, W(f_{\text{orig}})) \).

2. In order to have a small error bound and keep the map \( W \) contractive, \( s \) should be less than 1.

Images with global self-similarity can be coded efficiently by solving the inverse problem. However, natural images are rarely globally self-similar. The property only exists locally among different parts of the image. Figure 2.3 shows an image with local self-similarity. Thus, practical fractal coding is a block coding technique. First, the support \( S^2 \) is partitioned into \( Nr \) nonoverlapping range blocks \( R_1, R_2, \ldots, R_{Nr} \). Similarly, the same image support is partitioned into a set of overlapping domain blocks \( D_1, D_2, \ldots, \)
Figure 2.3 Local self-similarity. Two similar blocks are highlighted.

Figure 2.4 A contractive mapping $\psi$ from domain $D$ to range $R$.
\( D_{Ni} \), where \( D_i \) can be anywhere in the image, and is larger in size than range blocks in order to meet the contractivity condition. Usually, \( D_i \) is chosen to be twice the size of \( R_i \).

The contractive transformation \( W \) is defined blockwise:

\[
W = \bigcup_{i=1}^{N_r} w_i
\]

where \( w_i: f|_{D_i} \rightarrow f|_{R_i} \) is a contractive map from the domain block \( D_i \) to the range block \( R_i \), as shown in Figure 2.4. When \( w_i \) is applied to the domain block, the result is an image on the range block. We have

\[
W(f) = \bigcup_{i=1}^{N_r} W(f)|_{R_i} = \bigcup_{i=1}^{N_r} w_i(f|_{D_i})
\]  

(2.17)

Thus, the construction of the fractal code \( W \) for \( f_{\text{range}} \) can be done by defining the \( w_i \) separately for each \( R_i \). The block encoding of \( f_{\text{range}} \) amounts to finding a transformation \( w_i \) from a domain \( D_i \) to the range \( R_i \), such that the transformed domain block \( w_i(f_{\text{range}}|_{D_i}) \) is a close approximation to the original range block \( f_{\text{range}}|_{R_i} \) (i.e., \( d(f_{\text{range}}|_{R_i}, w_i(f_{\text{range}}|_{D_i})) \) is as small as possible). The process is shown in Figure 2.5.

Notice that \( w_i \) is in effect a three-dimensional transform since

1. It spatially shrinks the domain support to the range support.
2. It contracts the intensity of the domain image.
Let us denote
\[ w_i(x, y, z) = (G_i(x, y), I_i(z)): D_i \times I \rightarrow R_i \times I \] (2.18)

where \( G_i \) is a geometric part which maps the domain to the range, and \( I_i \) is an intensity part which changes the pixel intensity values. The geometric part first spatially contracts the domain size \( D_i \) to the range size \( R_i \). We denote this mapping as
\[ h_i(x, y): D_i \rightarrow R_i \] (2.19)

Since there are eight ways to map one square to another, as shown in Figure 2.6, the geometric part then performs one of the symmetrical mappings.

![Figure 2.6 Eight symmetrical mappings](image)

We denote this mapping as
\[ \theta_i(x, y): R_i \rightarrow R_i \] (2.20)

The total geometric mapping can be written as
\[ G_i(x, y) = \theta_i(x, y) \circ h_i(x, y): D_i \rightarrow R_i, \ \forall (x, y) \in D_i \] (2.21)

or:
After the support $D_i$ is transformed into $R_i$, the intensity of the pixel will be changed accordingly. We denote the intensity mapping caused by $G_i$ as

$$g_i(f): I \rightarrow I$$

(2.23)

The pixel value after the intensity mapping is taken as the local average of those from the domain block before the geometric mapping. Then

$$u(x, y) = g_i(f|_{D_i}) = g_i(f(G_i^{-1}(x, y))), \forall (x, y) \in R_i$$

(2.24)

Further intensity mappings are now performed according to

$$m_i(f): I \rightarrow I$$

(2.25)

where $m_i$ is defined as a contractive affine transformation, consisting of contrast scaling and intensity shifting.

$$\nu(x, y) = m_i[u(x, y)] = s, u(x, y) + t_i$$

(2.26)

The total intensity mapping can be written as

$$I_i(f) = m_i \circ g_i(f): I \rightarrow I$$

(2.27)

or:

$$I_i(f|_{D_i}) = m_i \circ g_i(f|_{D_i}) = \nu(x, y) = s, u(x, y) + t_i$$

(2.28)

where $I_i(f|_{D_i})$ is contractive (i.e., $d(I_i(f_1), I_i(f_2)) \leq s_i d(f_1, f_2)$), allowing us to use the contractive fixed point theorem. For a point $(x, y)$ with intensity $z$ belonging to domain $D_i$, the contractive mapping $w_i$ can also be expressed in the matrix form as
\[
\begin{bmatrix}
    x \\
    y \\
    z
\end{bmatrix}
= \begin{bmatrix}
    a_i & b_i & 0 \\
    c_i & d_i & 0 \\
    0 & 0 & s_i
\end{bmatrix} \begin{bmatrix}
    x - D_x \\
    y - D_y \\
    g_i(z)
\end{bmatrix} + \begin{bmatrix}
    R_x \\
    R_y \\
    t_i
\end{bmatrix}
\] (2.29)

where \((a_i, b_i, c_i, d_i, D_x, D_y, R_x, R_y)\) represents the geometric part, and \((s_i, t_i, g_i)\) represents the intensity part. In practice, the encoder does not need to explicitly determine the constants \((a_i, b_i, c_i, d_i)\). They are implicitly defined by the relative size, orientation and position of the domain with respect to the range. In particular, if the encoder only looks for domain blocks that are twice as large as the range block, the scaling factor which would normally be derived from the \((a_i, b_i, c_i, d_i)\) is equal to 0.5. Similarly, if domains and the ranges are restricted to squares, there are only 8 possible orientations of the domain relative to the range. Thus 3 bits are sufficient to encode the orientation index. Parameters \(s_i\) and \(t_i\) are chosen so that the \(w_i\)-transformed domain block is as close as possible to a given range block. Specifically, let

\[E = \frac{1}{MN} \sum_{(x, y) \in R} \left[ s_i u(x, y) + t_i - f(x, y) \right]^2\] (2.30)

where \(M \times N\) is the size of the range block. The minimum \(E\) occurs when the partial derivatives of \(E\) with respect to \(s_i\) and \(t_i\) are both zero. The solution of the two linear equations is given by

\[s_i = \frac{MN \sum_{(x, y) \in R} f u - (\sum_{(x, y) \in R} f) (\sum_{(x, y) \in R} u)}{MN \sum_{(x, y) \in R} u^2 - (\sum_{(x, y) \in R} u)^2}\] (2.31)

\[t_i = \frac{1}{MN} \left( \sum_{(x, y) \in R} f - s_i \sum_{(x, y) \in R} u \right)\] (2.32)

The \(w_i\) is fully specified by five numbers:

1. \(\theta_i\): the index of symmetries.
2. \(D_x\): the \(x\) coordinate of domain block \(D_x\).
3. \(D_y\): the \(y\) coordinate of domain block \(D_y\).
4. \( s_i \): the contractive factor.
5. \( t_i \): the grey level shift.

We do not include the range locations \( R_x \) and \( R_y \) because they are uniquely determined by the position of \( R \) in the encoding sequence. The above encoding procedure shows that the affine maps are contractive in both spatial and intensity dimensions. The contraction in intensity (reduction of contrast) is essential to ensure convergence of the decoding process. The spatial contraction is useful to create detail in the image at different scales, and so to get a much better approximation of the original image. Without spatial contraction, the decoding process would still converge, but to a blocky image without any contrast inside each block. With spatial contraction, the contrast across range blocks, which is initially provided by the brightness offset components of the affine maps, is propagated within each range to smaller and smaller scales after each iteration. With the above algorithm, the fractal codes of an image consist of a list of affine maps, each map being restricted to a specific domain. Since each map is contractive in the intensity dimension, we can apply the fixed point theorem to decode the image. Start with any image \( f_0 \) and successively compute \( W(f_0) \), \( W(W(f_0)) \), and so on, until the sequence converges to the attractor \( f \). Convergence is generally obtained in 8 to 10 iterations.

### 2.2 Fractal Coding with Nonlinear Contractive Functions

Contractive factor \( s_i \), given above, is the result of a linear regression. However, it does not necessarily satisfy the contractive condition of \( |s_i| < 1 \). When the slope of the range block is larger than that of the domain block, \( |s_i| \) is larger than 1. In this case, the fractal transform may be divergent, resulting in a white spot or black hole in the corresponding range block. Even if the decoding is eventual contractive, it would be very slow since it would require about twice as many iterations to arrive at the contraction. Jacobs et al. have noticed this problem [84]. When \( |s_i| > 1 \), the contractive factor is truncated to the allowed maximum value \( s_{\text{max}} \). This treatment leads to an increase of the error \( E \) and thus results in loss of fidelity in the reconstructed image. Instead of choosing a constant \( s \) as a contractive factor, a contractive function \( s(x, y) \) is defined.

**Definition 2.2** Let \( W \) be a mapping on the image metric space \((F, d)\). For any \( f_1(x, y), f_2(x, y) \in F \), if there exists a function \( 0 \leq |s(x, y)| < 1 \) and \( d(W(f_1(x, y)), W(f_2(x, y))) \leq ")
\[ |s(x, y)| d(f_1(x, y), f_2(x, y)), \] then \( W \) is called a contractive mapping and \( s(x, y) \) is called a contractive function.

A cluster of planes which are not parallel to the \( Z \) axis has the general form \( z = ax + by + c \). The contractive factor functions are then formed by limiting the dynamic range of these planes to \((-1, 1)\) with the following mapping:

\[
s(x, y, a, b, c) = \pm \frac{1}{1 + e^{ax + by + c}}
\]  

(2.33)

where \( a, b \) and \( c \) are the parameters to be optimized. This sigmoidal function is commonly used in the backpropagation algorithm of neural networks [85]. \( s(x, y) \) takes positive values when the angle between a domain normal direction and a range normal direction is less than \( \pi/2 \), and negative values, if the angle is larger than \( \pi/2 \). Figure 2.7 shows a domain block that is mapped into a steeper range block under a nonlinear contractive function.

![Figure 2.7 Nonlinear contractive mapping](attachment:image.png)

The optimization objective function can be written as

\[
E = \frac{1}{MN} \sum_{(x, y) \in R} \sum [s_i(x, y, a, b, c) u(x, y) + t_i - f(x, y)]^2
\]  

(2.34)

To find the closest matched domain block for a given range block, fractal coding has to search through the domain block pool. The variable metric optimization [86] is used to find \((a_i, b_i, c_i, t_i)\). Besides these four parameters, the fractal code also includes the sign (+ or -) of the equation (2.33), \( \theta \), \( D_x \) and \( D_y \).

In general, the performance of a compression scheme can be described by the coding
efficiency, the quality of reconstructed images and the computational complexity. Coding efficiency is often given in terms of the compression ratio (CR), which is defined as

\[ CR = \frac{\text{Number of bits in original image}}{\text{Number of bits in compressed image}} \] (2.35)

For an 8-bit original image, the bit rate in terms of bits per pixel (bpp) is related to CR as

\[ \text{bpp} = \frac{8}{CR} \] (2.36)

Ideally, the quality of the reconstructed image should be evaluated by subjective tests. In practice, the peak signal-to-noise ratio (PSNR) is used to measure the difference between two images although it is a poor indication of the perceived quality. The PSNR is defined as (in dB)

\[ \text{PSNR} = 10 \log_{10} \left( \frac{B^2}{\text{MSE}} \right) \] (2.37)

where \( B \) is the largest signal amplitude (255 for an 8-bit image), and \( \text{MSE} \) is the Mean-Squared Error. For an image of size \( N \times N \), \( \text{MSE} \) is given by

\[ \text{MSE} = \frac{1}{N^2} \sum_{x=1}^{N} \sum_{y=1}^{N} [f(x, y) - \hat{f}(x, y)]^2 \] (2.38)

where \( f \) is the original image and \( \hat{f} \) is the reconstructed image.

Experiments for the nonlinear contractive function based method in this section have been conducted. Figure 2.8 shows the original 512 x 512 Lenna image. Quadtree partition is used for range blocks. The initial range block size is 64 x 64. The root-mean-square (RMS) error was determined for each range block. Blocks which had an error exceeding 8.5 and were larger than 8 x 8 in size, were split. Figure 2.9 shows range blocks given by quadtree partition for our algorithm. To reduce the dynamic range of the coefficients, all range blocks are classified into the positive class or the negative class, according to the sign of equation (2.33). Each map is first uniformly quantized with parameters of \( D_x \): 6 bits, \( D_y \): 6 bits, \( a \): 4 bits, \( b \): 4 bits, \( c \): 6 bits, \( t \): 7 bits, \( \theta \): 3 bits, and the sign in (2.33):
Figure 2.8 Original $512 \times 512$ Lenna image

Figure 2.9 Range blocks
Figure 2.10 Nonlinear contractive functions encoded image, 0.1997 bpp, 31.05 dB

Figure 2.11 Constant contractive factors encoded image, 0.2001 bpp, 30.45 dB
1 bits. The fractal codes are then encoded by Huffman encoding. Figure 2.10 shows the reconstructed image by nonlinear contractive functions at bit rate 0.1997 bpp and PSNR = 31.05 dB. Figure 2.11 is the result by constant contractive factors at 0.2001 bpp and PSNR = 30.45 dB. It is clear that our approach gives a visually better reconstructed image.

Table 2.2 PSNR as a function of the number of iterations

<table>
<thead>
<tr>
<th>Number of Iterations</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR Nonlinear</td>
<td>19.91</td>
<td>25.04</td>
<td>29.19</td>
<td>30.66</td>
<td>31.01</td>
<td>31.05</td>
<td>31.05</td>
<td>31.05</td>
<td>31.05</td>
<td>31.05</td>
</tr>
<tr>
<td>PSNR L₂ Norm</td>
<td>18.03</td>
<td>21.00</td>
<td>23.66</td>
<td>26.64</td>
<td>29.13</td>
<td>30.25</td>
<td>30.45</td>
<td>30.46</td>
<td>30.45</td>
<td>30.35</td>
</tr>
</tbody>
</table>

Table 2.2 lists PSNR as a function of the number of iterations and the data is plotted in Figure 2.12. The number of iterations using the nonlinear method is about half the number of that using the conventional L₂ norm method. This improvement results in a faster decoding process. However, the nonlinear optimization used in the encoding process is time-consuming. Fortunately, we have proposed a new algorithm for fast fractal encoding which will be presented in the next chapter.

Figure 2.12 Fractal decoding: PSNR as a function of the number of iterations
2.3 Parallel Implementation of Fractal Image Encoding

2.3.1 Three Parallel Encoding Structures
Fractal image encoding is a self-similarity search process between range and domain blocks. Full search through the image is computational-demanding. For example, according to Sloan, complex colour images require 100 hours to encode using a MASSCOMP 5600 work station [87]. Fortunately, in fractal coding each block is encoded independently, and in addition, the domain block searches are also independent. These properties make the following three parallel implementations possible.

1. All range blocks are encoded in parallel.
2. The domain block searches for each range block are implemented in parallel.
3. Nested parallel pattern: encoding different range blocks and searching the domain blocks for each range block are both parallelized.

Since the range blocks are nonstationary, the computational load for each processor in scheme (1) is more likely imbalanced than that in scheme (2). In this section, we implement the second scheme on a KSR1 (Kendall Square Research) parallel computer.

2.3.2 Parallel Implementation of the Domain Block Search
Assume that a range block has the size of \( r \times r \) and its corresponding domain blocks have the size of \( 2r \times 2r \). Starting from the origin of the image, shift the domain block along row and column directions, we consider all possible locations of the domain blocks, as shown in Figure 2.13. Each domain block has eight symmetrical mappings. For an image of size \( N \times N \), the total number of domain blocks is

\[
N_d = 8 \left( \frac{N - 2r}{h} + 1 \right)^2
\]

(2.39)

where \( h \) is the step size of the domain location shift in the original image. For \( N = 512 \), \( r = 16 \), \( h = 8 \), we have \( N_d = 29768 \). Serial search through the large number of the domain blocks is time-consuming. Since matching a range block to different domain blocks is independent, the search process is readily parallelized.
Figure 2.13 Partition the domain locations into \( n \) tiles. Each point represents the upper left corner of a domain block.

The KSR1 is a parallel computer which can be configured with up to 1088 processors. Individual processors are called cells in KSR terminology. The University of Toronto KSR1 is equipped with 32 cells, giving it a maximum performance rating of 1.28 GFLOPS (Giga-floating-point operations per second). Parallelism is available to users at two levels:

1. Process level parallelism. This is the traditional UNIX support for multiple processes, with parallelism arising from the fact that different processes can be executed concurrently on different cells.

2. Thread level parallelism. A thread is a single sequential flow of execution within a process, and a single process can have an arbitrary number of threads. Threads share a common memory space and can be created and destroyed dynamically as needed. Most of KSR's parallelism is built on p-threads, or POSIX threads (based on IEEE POSIX P1003.3a).
For our domain searches, let us partition the iteration space of the domain blocks into \( n \) tiles, where \( n \) equals the number of the threads, as shown in Figure 2.13. The thread level parallelism is achieved by a simple parallel C (SPC) interface to the KSR parallel run time library, PRESTO, which maps tiles to threads. Threads are then scheduled on to the cells by KSR OS to cause parallel executions. In our case, each tile is executed by a single processor cell. The process is shown in Figure 2.14. Since the number of the domain blocks \( N_i \) is very large while \( n \) is small (1 - 24), the work load is in a good balance. The speedup of the parallel execution is

\[
\text{speedup} = \frac{T_1}{T_n}
\]

(2.40)

where \( T_1 \) is the serial execution time using a single cell and \( T_n \) is the parallel execution time using \( n \) cells.

Figure 2.15 shows the encoding time as a function of the number of processors. On the curve, we see that doubling the number of processors gives a processing time approximately reduced by a factor of two. Figure 2.16 is the speedup curve achieved by KSR1. Figures 2.17 and 2.18 show the original 512 x 512 Peppers image and its encoding results at 0.2404 bpp with PSNR = 31.72, respectively. Although the speedup is significant, the encoding process is time-consuming. To reduce the computational complexity, we have proposed a pyramidal algorithm for the fast fractal encoding, which could speed up the search process up to two orders of magnitude compared with the brute force search technique. This fast encoding algorithm is discussed in the next chapter.
Figure 2.15 Execution time as a function of the number of processors

Figure 2.16 Speed-up achieved by KSR1
Figure 2.17 Original 512 x 512 Peppers image

Figure 2.18 Parallel encoded image, 0.2404 bpp, 31.72 dB
Chapter 3

Speeding Up Fractal Image Encoding Using Pyramids

3.1 Introduction

As was discussed in the previous chapters, the high encoding complexity is the major drawback of fractal image compression. Attempts have been made to speed up the encoding process [88, 89, 90, 91, 92, 93]. One approach used to reduce the search complexity is to use a classification scheme, as in classified vector quantization (CVQ) [94]. Both the blocks to be encoded (i.e., range blocks) and the blocks to be searched (i.e., domain blocks) are classified into shade, mid-range, and edge classes. For a given range block, the closest matched domain block under contractive mapping is found by searching within the same class domain blocks. However, a classification scheme can give meaningful results only for small blocks, typically $4 \times 4$. Jacquin noticed that most of the artifacts visible in decoded images are rooted in wrong block classification and inaccurate block analysis for large blocks [26]. Furthermore, as there are only three classes, the computational savings are relatively small. Lepsøy et al. made an extension by using the 'clustering' method [88]. The LBG iterative procedure for vector quantization code book training [95] is used to find cluster centres which are sensitive to the initialization of cluster centres. The complexity was reduced by a factor of 7.4 in their example run. Other classification schemes are also possible. For example, Fisher used ordering of the first and
second order moments from four quadrants of a block as a criterion for classification [89]. However, the low order moments are not unique features of images (i.e., two visually different images may have the same second order moments [96]). Hurtgen et al. applied the locality of the domain block to reduce the search complexity [90]. Their approach is based on the assumption that close-by domain blocks are more likely to provide a good match than faraway ones, so that the domain blocks close to the range block may take a smaller step size for a fine search, while domain blocks faraway from the range block may take a larger step size for a coarse search. However, there is a disagreement about the use of close-by domain blocks. Fisher showed that for a fixed range, the domain is equally likely to be anywhere [89]. Departing from the above algorithms, Dudbridge presented a non-search method, which is an IFS-based solution to the image encoding problem [92]. The method retains the advantage of linear complexity, which may prove useful in real-time applications, but for a given bit rate, the encoded images are usually much inferior in quality to those of search-based methods. Dudbridge's method provides a tradeoff between the reproduction quality and the compression complexity. A parallel method can also be used to speed up the encoding (see Section 2.3.2), as each range block is encoded independently and the domain block searches are also independent.

In this chapter, we propose a fast encoding scheme based on pyramidal image representation. The search is first carried out on an initial coarse level of the pyramid. This initial search increases the encoding speed significantly, because not only is the number of the domain blocks to be searched reduced, but also the data within each domain block is only $1/4^k$ of that in the finest level, where $k$ is the pyramidal level. Consequently, only a small number of the fractal codes from the promising domain blocks in the coarse level are refined through the pyramid to the finest level, with little computational effort.

### 3.2 The Image Pyramidal Data Structure

Pyramidal image models employ several copies of the same image at different resolutions. The technique has also appeared under the names multi-grid method, multi-resolution analysis, multi-level approach, and hierarchical representation [97, 98, 99, 100]. Let $I(x, y)$ be the original image of size $2^M \times 2^M$. An image pyramid is a set of image arrays $I_k(x, y)$,
$k = 0, 1, ..., M$, each having size $2^k \times 2^k$. The pyramid is formed by lowpass filtering and subsampling of the original image. The lowpass filter is called the pyramid-generating kernel. When the generating kernel is symmetric, its convolution with an image is the same as the local averaging operation. The pixel $I_k(x, y)$ at level $k$ of a pyramid is given by

$$I_k(x, y) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} w(i, j) I_{k-1}(2x + i - c, 2y + j - c) \quad (3.1)$$

for $0 \leq x, y \leq 2^k - 1$, where $w(i, j)$ is a kernel of size $N \times N$, and $c = \lfloor N/2 \rfloor$ is the centre coordinate of the kernel. Notice that the size of the image has been reduced by half in each dimension, yielding a resultant image four times smaller than the input image. Iterative applications of the same filtering and subsampling process yield a multi-resolution representation of the image.

The generating kernel can be odd or even. Its properties have been studied by Burt [101] and Meer et al. [102]. The following constraints are often applied:

1. **Normalization:**

$$\sum_{i=0}^{N-1} \sum_{j=0}^{N-1} w(i, j) = 1 \quad (3.2)$$

This constraint guarantees that the reduced image maintains the same average intensity as the original image.

2. **Symmetry:**

$$w(i, j) = w(N - 1 - i, j) = w(i, N - 1 - j) = w(N - 1 - i, N - 1 - j) \quad (3.3)$$

for all $i$ and $j$. Thus, the neighbour pixels affect the centre pixel symmetrically.

3. **Unimodality:**

$$0 \leq w(i, j) \leq w(p, q) \quad (3.4)$$

for $i \leq p < N/2$ and $j \leq q < N/2$. The constraint implies that the larger weights will be at the centre of the mask.

4. **Equal contribution to the next level:**

The total contribution of a pixel at level $k + 1$ to level $k$ is the sum of all weights
which are multiplied to that pixel during the calculation of the level $k$. In order to avoid distortion of the signal, the weights are arranged so that each pixel at level $k + 1$ contributes an equal amount ($= 1/4$) to the pixel at level $k$.

$$
\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} w(2x + i, 2y + j) = \frac{1}{4}
$$

(3.5)

for $i, j = 0, 1$, where $w(i, j) = 0$ for $i, j > N - 1$.

5. Separability:

$$
w(i, j) = w_1(i) \cdot w_2(j)
$$

(3.6)

The separable kernel allows the 2-D filtering operation to be implemented efficiently as two 1-D filtering operations. The particular case

$$
w_1(i) = w_2(i) = w_0(i)
$$

(3.7)

is commonly assumed when working with symmetric weights.

With the odd size kernels, the $5 \times 5$ separable mask is used by Burt, yielding the so-called Gaussian pyramid. Using the above constraints, we have

$$
w_0 = (c, b, a, b, c)
$$

(3.8)

where

$$
b = \frac{1}{4} \text{ and } c = \frac{1}{4} - \frac{a}{2}
$$

(3.9)

Different values of $a$ give different masks. In particular, the Gaussian-like function $w_0 = (0.05, 0.25, 0.4, 0.25, 0.05)$ will be used in multichannel filtering in Chapter 6.

With the even size kernels, the $4 \times 4$ mask is widely used. The one-dimensional weight is

$$
w_0 = (b, a, a, b)
$$

(3.10)

where

$$
a = \frac{1}{2} - b \text{ and } a \geq b
$$

(3.11)
In particular, when \( w_0 = (0, 0.5, 0.5, 0) \), the actual \( 2 \times 2 \) kernel is given by

\[
\begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0.25 & 0.25 & 0 \\
0 & 0.25 & 0.25 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\] (3.12)

which produces an equal contribution and non-overlapping pyramid. In this case, (3.1) can be simplified as

\[
I_k(x, y) = \frac{1}{4} \sum_{i=0}^{1} \sum_{j=0}^{1} I_{k+1}(2x + i, 2y + j)
\] (3.13)

Because each pixel at level \( k \) is the \( 2 \times 2 \) local arithmetic mean of the pixels at the level \( k + 1 \), the pyramid is called a mean pyramid. We will use the mean pyramid in the next section for fast fractal encoding. The coarsest level \( (k = 0) \) image has size \( 1 \) and represents the average grey level of the original image. The finest level image \( I_M \) is the original image of size \( 2^M \times 2^M \). As the number of the levels decreases, the image details are gradually suppressed and spurious low spatial frequency components are introduced due to the aliasing effect. Figure 3.1 shows a 4-level pyramid of the Lenna image.

Because the pyramidal structures offer an abstraction from image details, they have been proven to be very efficient in certain kinds of image analysis \([103]\), motion estimation \([97]\) and image compression applications \([101]\).

### 3.3 Fast Pyramidal Domain Block Search

In this section, the pyramidal framework for fractal image compression is first introduced. The promising location matrix is then defined to guide the search process. The encoding parameter propagation rules are given at the end.

Notice that the contracted domain block image \( u(x, y) \) in (2.24) corresponds to the block in \((M - 1)\)th level \( I_{M-1} \) of the pyramid. When the range blocks are of size \( 2^m \times 2^n \), the optimization objective function (2.30) for the best matched domain block search can be rewritten as
Figure 3.1 A pyramid of the Lenna image
\[ E = \frac{1}{4^m} \sum_{x=0}^{2^m-1} \sum_{y=0}^{2^m-1} \left[ D(x, y, s, t) - R(x, y) \right]^2 \]  

(3.14)

where \( D(x, y, s, t) = s I_{M}(x, y) + t \) is an affine function of the scaled domain block, and \( R(x, y) = I(x, y) \) is the range block to be encoded.

\[ E^k = \frac{1}{4^k} \sum_{x=0}^{2^k-1} \sum_{y=0}^{2^k-1} \left[ D^k(x, y, s^k, t^k) - R^k(x, y) \right]^2 \]  

(3.15)

for \( k_0 \leq k \leq m \). Therefore, at pyramidal level \( k \), the search amounts to finding the best matched domain block of size \( 2^k \times 2^k \) in the image of the size \( 2^{M-m-k} \times 2^{M-m-k} \).

Figure 3.2 shows a range block and its domain block in a two-level pyramid. For example, for an original image of size \( 512 \times 512 \) \( (M = 9) \) and a range block size \( 32 \times 32 \) \( (m = 5) \), the search at \( k_0 = 2 \) is confined to an image of size \( 32 \times 32 \), with a range block size of \( 4 \times 4 \). The \( k = k_0 \) level of the range block pyramid is said to be initial and...
every domain block location from the \((M - m + k_0 - 1)\)th level of the image pyramid needs to be tested. An additional feature of the algorithm is the optimization of parameters (fractal codes) \(P^k(\theta^k, D^k_x, D^k_y, s^k, t^k)\) at each pyramidal level.

The domain block locations where the match errors are below some predefined threshold are known as the promising locations. Now, generate a \(2^M \cdot m + k \times 2^M \cdot m + k\) promising locations matrix \(G\):

\[
(G)_{2p, 2q} = \begin{cases} 1, & \text{if } E^k(p, q) < T^k \\ 0, & \text{otherwise} \end{cases}
\]  

(3.16)

where \((p, q)\) is the upper left corner coordinates of the domain block and \(T^k\) is the threshold at level \(k\). Matrix \(G\) is used as a guide in the search for the domain locations at the next level \(k + 1\). Tests are to be performed only at locations \((i, j)\) for \((G)_{i, j} = 1\) and its neighbouring locations. The other parameters \(P^k\) of the promising locations are also propagated to \(P^{k + 1}\) for further refinement at level \(k + 1\). For the conventional \(L_2\) norm fractal coding, we have \(\theta^{k+1} = \theta^k\), \(D^{k+1}_x = 2D^k_x\) and \(D^{k+1}_y = 2D^k_y\). Parameters \(s^{k+1}\) and \(t^{k+1}\) need to be reevaluated by (2.31) and (2.32), respectively. In the case of the nonlinear contractive functions described in the Section 2.2, the initial parameters at level \(k + 1\) are: \(\alpha^{k+1} = \frac{1}{2}a^k\), \(b^{k+1} = \frac{1}{2}b^k\), \(c^{k+1} = c^k\) and \(t^{k+1} = t^k\). The \(\frac{1}{2}\) gain before the \(a^k\) and \(b^k\) is due to the resolution increase in the \(x\) and \(y\) directions. The algorithm provides a gradual refinement of the fractal code. The process is repeated recursively until the finest level \(m\) is reached as shown in Figure 3.3. The iteration is performed for the promising domain blocks. At the finest level, if there is more than one location \((p, q)\) such that \((G)_{p, q} = 1\), the parameters with the smallest match error are selected as the fractal code. Different thresholds are used to determine promising locations in the corresponding pyramidal levels. The next section shows how to estimate these thresholds under the given image model.

### 3.4 Determining the Thresholds

The goal of this section is to find an estimation of the threshold used at each pyramidal level. Let \(x_i\) denote the grey level difference of a pixel between an affine transformed domain block \(D_i\) and a range block \(R_i\) at the finest level \(m\), i.e., \(x_i = D_i - R_i\), for \(i = 0, 1, \ldots, 2^M - 1\).
The initial range \( R^{k_0} \) is refined to \( R^{k_0+1} \) and so on, optimizing \( P_i^{k_0} \) to \( P_i^{k_0+1} \) levels.

At the match location \((p', q')\), the correlation of \( x_i \) is very small [104]. Thus, we may consider \( x_i \) as independent, identically distributed (i.i.d.) random variables with an approximate Laplacian density function of the form:

\[
f(x) = \frac{\alpha}{2} e^{-\alpha |x|}
\]

where \( f(x) \) has mean \( \mu_0 = 0 \) and variance \( \sigma_0^2 = 2/\alpha^2 \). The histogram from our experimental data showed a reasonable approximation to the density function (see Figure 3.4). It can be shown, then, that the function of the random variable \( y_i = x_i^2 \) has a density function:

\[
f_y(y) = \frac{\alpha}{2\sqrt{y}} e^{-\alpha y^{\frac{1}{2}}} \quad (y > 0)
\]

Figure 3.3 Refinement of fractal codes from coarse to fine pyramidal levels

,..., \((2^m \times 2^m - 1)\). At the match location \((p', q')\), the correlation of \( x_i \) is very small [104]. Thus, we may consider \( x_i \) as independent, identically distributed (i.i.d.) random variables with an approximate Laplacian density function of the form:

\[
f(x) = \frac{\alpha}{2} e^{-\alpha |x|}
\]

where \( f(x) \) has mean \( \mu_0 = 0 \) and variance \( \sigma_0^2 = 2/\alpha^2 \). The histogram from our experimental data showed a reasonable approximation to the density function (see Figure 3.4). It can be shown, then, that the function of the random variable \( y_i = x_i^2 \) has a density function:

\[
f_y(y) = \frac{\alpha}{2\sqrt{y}} e^{-\alpha y^{\frac{1}{2}}} \quad (y > 0)
\]
It has a mean \( \mu_y = \sigma^2_0 \) and a variance \( \sigma^2_y = 5\sigma^4_0 \). The next step towards the goal is to find the distribution of the mismatch measure as in (3.14), which can be rewritten as

\[
E = \frac{1}{n} \sum_{i=0}^{n-1} x_i^2 = \frac{1}{n} \sum_{i=0}^{n-1} y_i \tag{3.19}
\]

where \( n = 2^m \times 2^m \). According to the central limit theorem [105], under certain conditions, the density of the sum of \( n \) independent random variables tends to a normal density as \( n \) increases. It follows, then, that \( E \) has approximate normal distribution with

\[
\mu_E = \sigma^2_0, \quad \sigma^2_E = \frac{5}{4^m} \sigma^4_0 \tag{3.20}
\]

Let \( P_a \) be the probability of finding the best match \( (p^*, q^*) \) (i.e., \( P(E < T^m) = P_a \)). The threshold will then be

\[
T^m = \mu_E + x_a \sigma_E = \sigma_0(1 + \frac{\sqrt{5}}{2^m} x_a) \tag{3.21}
\]

where \( x_a \) is the \( P_a \) point of standard normal distribution. For example, when \( P_a = 0.9 \), \( x_a = 1.28 \).

At a coarse level \( k \), it is shown (see Appendix A) that the thresholds are given by

![Figure 3.4. Matching error (\( D_i - R_i \)) distribution](image-url)
for \( k = k_0, \ldots, m - 1 \). From (3.22), the threshold is a monotonic increasing function of \( k \). That is, the finer levels have larger thresholds than the coarser levels. Since we have assumed that the encoding errors are not correlated, the theoretical threshold in (3.22) is smaller than the practical one. Therefore, the thresholds should be enlarged by multiplying scalars for practical applications.

### 3.5 Computational Efficiency

To encode a range block, the fractal method has to calculate the parameters \( s_i \) and \( t_i \) for every domain block. The calculation can be much simplified by omitting the repeated operations. Let us introduce the following notations:

\[
\begin{align*}
  P_1 &= C \sum_{(x, y) \in \cal R} I_{M-1}^1, \\
  P_2 &= C \sum_{(x, y) \in \cal R} I, \\
  P_3 &= C \sum_{(x, y) \in \cal R} I_{M-1}^2, \\
  P_4 &= C \sum_{(x, y) \in \cal R} I^2, \\
  P_5 &= C \sum_{(x, y) \in \cal R} I_{M-1} I
\end{align*}
\]

where \( C (= 1/4^m) \) is a constant related to the range block size. As a result, equations (2.31) and (2.32) can be respectively rewritten as

\[
\begin{align*}
  s_i &= \frac{P_5 - P_1 P_2}{P_3 - P_1^2}, \\
  t_i &= P_2 - s_i P_1
\end{align*}
\]

Because \( P_1 \) and \( P_3 \) are derived from domain blocks, they are calculated only once and used for encoding all range blocks. \( P_2 \) and \( P_4 \) are derived from range blocks, where \( P_2 \) is easily obtained from the initial range level. The main computational effort of matching a range to a domain block is to find the correlation term \( P_5 \). Hence, we assume that the computational cost is approximately proportional to the product of the number of domain
blocks searched and the number of pixels in each block. For an original image of size $2^M \times 2^M$ and range blocks of size $2^m \times 2^m$, with $D_i$ chosen in each dimension as twice the size of the $R_i$, the search domain image is $2^{M-1} \times 2^{M-1}$ with the contracted domain block of size $2^m \times 2^m$. The computational cost is

$$C_1 = 8 \left( \frac{2^{M-1} - 2^m}{h} + 1 \right)^2 2^{2m}$$

(2.25)

where $h$ is the step size of the domain block search which is performed in the contracted domain image $I_{M-1}$ of the pyramid. When a pyramidal search is applied, the computational effort for the algorithm is determined by the average number of the promising locations $n_p$ on every pyramidal level and the number of the shifts $n_i$ around each promising location. The computational effort can be computed according to

$$C_2 = 8 \left( \frac{2^{(M-1)m-k_i-k}}{h(k_0)} + 1 \right)^2 2^{2k_0} + n_p n_i \sum_{i=k_0-1}^{m-1} 2^{2i}$$

(3.26)

where the first term corresponds to the initial step of the algorithm which needs to test every domain block on the initial range pyramidal level $k_0$. The search step size $h(k_0)$ is related to the finest level step size $h$ as follows:

$$h(k_0) = \max \left( 1, \frac{h}{2^{(m-k_i)}} \right)$$

(3.27)

where we assume that only the integer search step is used in level $k_0$, although, in general, a search with sub-pixel accuracy is possible.

The number of operations required to create an image pyramid is proportional to the number of pixels, and is given by

$$C_3 = K_1 \sum_{i=M-1}^{m-1} 2^{2(M-i)}$$

(3.28)

Compared to the number of optimization operations required during the domain block search, this part can be neglected.

The benefit in computational saving, using pyramids, relative to the full search of the
original image is estimated as

\[ Q = \frac{C_1}{C_2 + C_3} = \frac{C_1}{C_2} \quad (3.29) \]

For a given image and range block size, the value of \( Q \) depends on the depth of the pyramid and the search step size. The deeper the pyramid, the bigger the \( Q \) value is, as there is less data in the coarser level of the pyramid. However, the depth of the pyramid is restricted by the size of the range block and minimum range block size (such as 2 \( \times \) 2). The smaller the step size, the bigger the \( Q \) value becomes, since there would be more domain blocks to be searched than those in the pyramid. For example, for an image of size 512 \( \times \) 512, a range block of 32 \( \times \) 32, when \( h = 2, n_p = 20, n_s = 16 \) and \( k_0 = 2 \), the computational saving factor will be 194. When computing the error (3.15) during encoding, uniform quantized \( s \) and \( t \) values are used to improve the fidelity. In equation (3.25), we have neglected the quantization operations of \( s \) and \( t \), while in (3.26), we have neither included quantization operations, nor the overhead for propagation of promising locations. Hence, the actual \( Q \) value is expected to be smaller than the theoretical value.

It is also apparent that the equation (3.29) is for a given block size. When an adaptive partition scheme such as quadtree partition is used, the encoded image may contain many different block sizes. The speed-up for encoding a full size image will be the weighted average of (3.29), where each weight is given by the percentage of the number of its corresponding block size in the image.

### 3.6 Coding Results

The above algorithm is implemented on the KSR1 computer. Quadtree partition is used for range blocks. The initial range block size is 64 \( \times \) 64 and the minimum block size is 8 \( \times \) 8. The initial level \( k_0 \) is set to 2 for 64 \( \times \) 64 and 32 \( \times \) 32 blocks, and 1 for 16 \( \times \) 16 and 8 \( \times \) 8 blocks. The quality of the encoded image depends on the encoding error bound. Obviously, smaller error bounds lead to better image quality at higher bit rates. Figure 3.5 shows the promising domain locations in each level for a 32 \( \times \) 32 range block (see also Figure 2.3). The best matched domain block is located above the range block at position "*". Table 3.1 shows experimental results for the Lenna image using both full search and
pyramidal search methods. In the extreme case of \( h = 1 \), full search took 396487 CPU seconds (serial running time), while pyramidal search took 2119 CPU seconds, which leads to \( Q = 187.1 \). When \( h = 4 \), full search took 25660 seconds. Figure 3.6 shows the reconstructed image at bit rate 0.2317 bpp (CR = 34.5307) and PSNR = 30.91 dB. Our pyramidal search took 1103 seconds. Figure 3.7 is the result at the bit rate of 0.2316 bpp (CR = 34.5409) and PSNR = 30.62 dB. The actual \( Q \) is 23.3 in this case. Figure 3.8 shows the full search and pyramidal search time as a function of the search step size, respectively. Table 3.2 lists the experimental results for another Peppers image. Figures 3.9 and 3.10 show the encoded images using the full search and the pyramidal search techniques, respectively. In each case, full search method dramatically increases the compression time, but improves the image quality only very marginally. During the refinement of the fractal codes from the coarse to finer levels, thresholds are estimated in order to determine the promising locations. In many cases, the refinement process is terminated at an early stage because there are no promising locations for the range block, and further quadtree partitions are needed.

Table 3.1 Experimental results for the Lenna image

<table>
<thead>
<tr>
<th>Step ( h )</th>
<th>Full Search</th>
<th>Pyramidal Search</th>
<th>Speed-up ( Q )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time (s)</td>
<td>PSNR</td>
<td>CR</td>
<td>Time (s)</td>
</tr>
<tr>
<td>1</td>
<td>396487</td>
<td>31.57</td>
<td>32.65</td>
</tr>
<tr>
<td>2</td>
<td>100366</td>
<td>31.14</td>
<td>34.74</td>
</tr>
<tr>
<td>4</td>
<td>25660</td>
<td>30.91</td>
<td>34.53</td>
</tr>
<tr>
<td>8</td>
<td>6787</td>
<td>30.49</td>
<td>33.26</td>
</tr>
</tbody>
</table>

Table 3.2 Experimental results for the Peppers image

<table>
<thead>
<tr>
<th>Step ( h )</th>
<th>Full Search</th>
<th>Pyramidal Search</th>
<th>Speed-up ( Q )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time (s)</td>
<td>PSNR</td>
<td>CR</td>
<td>Time (s)</td>
</tr>
<tr>
<td>1</td>
<td>410324</td>
<td>31.33</td>
<td>32.03</td>
</tr>
<tr>
<td>2</td>
<td>104630</td>
<td>31.72</td>
<td>33.28</td>
</tr>
<tr>
<td>4</td>
<td>26885</td>
<td>31.12</td>
<td>33.64</td>
</tr>
<tr>
<td>8</td>
<td>7024</td>
<td>30.89</td>
<td>32.47</td>
</tr>
</tbody>
</table>
Figure 3.5 Promising domain locations in each pyramidal level for a $32 \times 32$ range block, where $k$ is the pyramidal level index for the range block. "*" is the closest matched domain block location.
Figure 3.6 Full search encoded image, 0.2317 bpp, 30.91 dB

Figure 3.7 Pyramidal search encoded image, 0.2316 bpp, 30.62 dB
Figure 3.8 Full search and pyramidal search time as a function of the search step size

Considering that other fast search techniques, such as the conjugate direction search and the 2-D logarithmic search, lead to relatively larger matching errors, we conclude that the pyramidal search algorithm is quasi-optimal in terms of minimizing the mean square error. The main advantage of the pyramidal algorithm is the greatly reduced computational complexity, when compared to full search. Simulation results show that our algorithm can reduce the encoding complexity by up to two orders of the magnitude compared with the full search method, and at the same time, gives a quasi-optimal reconstruction quality.

The pyramidal data structure can also be used to increase the decompression speed by taking advantage of the resolution-independence of the fractal method [106]: the image is decoded at a lower resolution in the first iterations, and decoded at the full image size only for the last iterations. While working at a lower resolution, the decoder handles only a fraction of all the pixels of the original image, thus the total number of instructions required for decoding is much reduced and hence a faster decoding is achieved.
Figure 3.9 Full search encoded image, 0.2378 bpp, 31.12 dB

Figure 3.10 Pyramidal search encoded image, 0.2375 bpp, 30.57 dB
Chapter 4

Perceptually Based Fractal Image Compression

4.1 Introduction

As with any lossy compression schemes, the challenge is to either maximize the image quality at a fixed bit rate, or minimize the bit rate required for a given image quality. This chapter describes how the coding fidelity can be improved at a given bit rate using fractal coding. A conventional fractal encoding algorithm uses a mean square error metric as an optimization criterion, which correlates poorly with the human visual response to distortions. As digital images and video products are designed for human viewing, human visual system models can be exploited to faithfully preserve perceptually important information and eliminate the information that the visual system cannot perceive. Specifically, when a compressed image cannot be distinguished visually from the original under certain viewing conditions, the compression is said to be perceptually lossless. In this research, we introduce a perceptually meaningful distortion measure based on the human visual system’s nonlinear response to luminance and visual masking effects. Blackwell’s psychophysical raw data [78] on contrast thresholds is first interpolated as a function of background luminance and visual angle, and is then used as an upper error bound for perceptually based image compression.
4.2 The Human Visual System (HVS) Model

The human visual system model [107] mainly addresses three visual sensitivity variations: i.e., as a function of the luminance level, the spatial frequency, and the signal content.

First, the perceived luminance is a nonlinear function of the incident luminance. According to Weber's Law [108], if the luminance \( (L_b + \Delta L) \) of an object is just noticeably different from its background luminance \( L_b \), then, \( \Delta L/L_b = \text{constant} \). Therefore, the just-noticeable-difference (JND) \( \Delta L \) increases with the increasing \( L_b \).

Second, there is a spatial filtering mechanism in the human visual system, which can be described by the modulation transfer function (MTF). The bandpass characteristic of the MTF suggests that the human visual system is most sensitive to mid-frequencies and least sensitive to high frequencies. Mach band is an example of this mechanism.

Finally, natural images contain a complex rather than a uniform luminance background. In this case, there is a reduction of visibility (i.e., an increased visual threshold by the spatial or temporal nonuniformity of the background). The effect is referred to as visual masking.

In recent years, attempts have been made to incorporate human perception models into image compression [109, 110]. Most of the early work of applying human perception models to image compression utilized the human visual system's frequency sensitivity, as described by the MTF which defines the eye's spatial frequency response to sine wave gratings [111, 112, 113]. Since only the coefficients of the DFT correspond directly to the measured spatial frequency response, the use of MTF in DCT based coders needs a transformation to account for the difference between the two bases [114]. During the encoding process, the low frequency coefficients are preferentially weighed for fine quantization, while high frequency coefficients are partially suppressed by coarse quantization to reflect the lowered response of the HVS. Since frequency sensitivity is a global property, dependent only on the image size and viewing conditions, codec systems using the MTF models do not exploit the HVS luminance nonlinearity and spatial masking. For example, the JPEG algorithm focuses on the visual spatial frequency response without taking into account the nonlinear response of the HVS to luminance. The JPEG algorithm suppresses high-frequency components within each \( 8 \times 8 \) block, and leads to less contrast to create a masking. Unfortunately, the HVS can easily detect the
change in local contrast. In the next section, we introduce an image metric based on the HVS nonlinearity and masking effects.

4.3 A Perceptually Meaningful Image Metric

Unlike transform coding, which is performed in the frequency domain, fractal coding is performed in the spatial domain. By the collage theorem (Theorem 2.4), the reconstruction error is directly related to the metric used in the encoding process. According to Weber's law, it is the image contrast, and not the linear difference, that determines the visibility of luminance. For a range block of size $N$, we define a new metric as:

$$
d = \frac{1}{N} \sum_{i=1}^{N} \frac{|L_i - \hat{L}_i|}{\bar{L}}
$$

(4.1)

where $L_i$ is the luminance of the original block, $\hat{L}_i$ is the luminance of the fractal approximated (i.e., encoded) block, and $\bar{L}$ is the average block luminance of the original block. Since each block is very small relative to the full size of the image, $\bar{L}$ may be considered as a local background luminance, and so the term in the sum is the local contrast at pixel location $i$. Thus, $d$ is an average local contrast of the block to encode.

Digital images are represented in terms of the grey levels, while the new metric uses real physical measures, i.e., in terms of optical flux in cd/m$^2$ (candela/meter$^2$). Thus, a conversion from the digital representation to physical measures is needed. Usually the dynamic range of the luminance depends on the display monitor [115]. In our work, the luminance is assumed to vary linearly in the luminance range of 0.05 cd/m$^2$ (all pixels black) to 100 cd/m$^2$ (all pixels white) [116]. This dynamic range was divided into 255 equal greyscale intervals. Therefore, the luminance $L_i$ is related to grey level $I_i$ as:

$$
L_i = 0.05 + 0.392I_i
$$

(4.2)

Let $\hat{I}$ denote the fractal encoded image grey level, and $\bar{I}$ the block mean grey level. The respective luminance levels are then
Inserting equations (4.2) and (4.3) into (4.1), we have

\[
\begin{align*}
\hat{L}_i &= 0.05 + 0.392 \hat{I}_i \\
\bar{L}_i &= 0.05 + 0.392 \bar{I}_i
\end{align*}
\]  \hspace{1cm} (4.3)

Assuming \(0.392 \bar{I} > 0.05\):

\[
d = \frac{1}{NI} \sum_{i=1}^{N} |I_i - \hat{I}_i|
\]  \hspace{1cm} (4.5)

For perceptually lossless compression, \(d\) should not exceed the visual contrast threshold \(T_B\). In other words, the average encoding error \(E\) needs to meet the constraint:

\[
E = \frac{1}{N} \sum_{i=1}^{N} |I_i - \hat{I}_i| \leq \bar{I} T_B
\]  \hspace{1cm} (4.6)

Notice that \(\hat{I}\) represents the fractal encoded image, and can be written as an affine transformation of the spatially contracted domain block, that is, \(\hat{I} = sI_{M - 1} + t\), where \(s\) is the contractivity factor and \(t\) is the grey-level shift. Inserting this expression into equation (4.6), we obtain

\[
E = \frac{1}{N} \sum_{i=1}^{N} |I_i - sI_{M - 1, i} - t| \leq \bar{I} T_B
\]  \hspace{1cm} (4.7)

The new metric is in the form of a weighed least absolute deviation (LAD). It is less sensitive to extreme errors than the least mean square metric. Since the encoding error has an approximated Laplacian distribution, as shown in Figure 3.4 of Chapter 3, the fractal code parameter estimation based on the new metric is a maximum likelihood estimation. Furthermore, since the metric is derived from the HVS, the fractal codes are expected to give an efficient representation of the visual information of the image. Parameters \(s\) and \(t\) are the solution of \(L_1\)-norm curve fitting, and can be solved by a
simple iterative procedure [117]. Other components of the fractal code include the location of the domain block and the index of the 8 rotations/reflections.

4.4 The Visual Contrast Threshold

Images can be mathematically described in terms of luminance (grey level) variations across space and time. Since the visual system tends to respond to differences in luminance, the intensity of many stimuli is best described in terms of contrast, that is, the luminance difference between an object and its background or between parts of an object or scene. For a simple luminance increment or decrement relative to the background luminance, the contrast can be defined as

\[ C = \frac{\Delta L}{L} \]  

(4.8)

where \( L \) is the background luminance and \( \Delta L \) is the increment or decrement in luminance.

**Definition 4.1** The contrast threshold is the smallest amount of luminance contrast between two adjacent spatial regions that can be detected on some specified percentage of trials.

![Figure 4.1 Parameters involved in the calculation of visual angle](image)

Another concept is needed to describe the size of objects. Instead of using the number of pixels to describe the size of objects, as is done in image processing, in visual studies, visual extent is conventionally designated in terms of the visual angle. The parameters involved in the calculation of the visual angle are depicted in Figure 4.1. In the figure, \( S \) is the size of the object, \( D \) is the distance from the object to the nodal point (\( N \)) of the
eye. For a circular object, $S$ equals the radius. In the case of a square block, $S$ is set as half of the side length. The visual angle $\alpha$ is defined as

$$\alpha = \arctan \left( \frac{S}{D} \right)$$

(4.9)

where $\alpha$ is in degrees. For small angles, equation (4.9) has an approximation of the form:

$$\alpha \approx 3438 \frac{S}{D}$$

(4.10)

where $\alpha$ is in arcmin (minutes of arc). In our experiments, the image had a resolution of $512 \times 512$. The display resolution was 38.8 pixels/cm. We used a viewing distance of 46 cm, so that the highest image frequency of 16 cycles/deg was still within the visual angle. Under this viewing condition, a block of size $B \times B$ (in pixels) had a visual angle:

$$\alpha = 0.9631B$$

(4.11)

where $\alpha$ is in arcmin.

During the Second World War, Blackwell conducted an experiment to determine the contrast threshold as a function of background luminance and visual angle [78]. Blackwell's nine observers viewed a uniformly illuminated wall that served as a screen for the presentation of targets. The wall subtended a visual angle of approximately $10^\circ$. The observers' task was to detect the presence of a circular test light projected upon the wall. Blackwell presented the average contrast threshold in Table 8 in his paper [78] for 50% correct detections. Since the dynamic luminance range of the display monitor (0.05 - 100 cd/m$^2$) was much smaller than what the HVS can detect, only a portion of Blackwell's data will be used in our work. After the luminance unit conversion from fL (foot Lambert) to cd/m$^2$, the contrast threshold data is plotted, as shown in Figure 4.2. Based on Blackwell's data, the least-squares surface fitting of Blackwell's visual contrast threshold is given by

$$h(x, y) = 0.0807x^2 + 0.5775y^2 + 0.0840xy - 0.4597x - 2.2745y - 0.1096$$

(4.12)

where $h(x, y)$ is the contrast threshold in log unit, $x$ is the average background luminance (equation (4.3)) in log cd/m$^2$, and $y$ is the visual angle (equation (4.11)) in log arcmin.
4.3 The least-squares surface fitting of Blackwell's visual contrast threshold
Figure 4.3 shows the threshold surface. For a fixed visual angle (block size), the contrast threshold decreases as the background luminance increases, while at fixed luminance, the threshold decreases as the visual angle increases. It should be pointed out that Blackwell’s test data was obtained using uniform luminance on uniform backgrounds. However, real-world images are usually not uniform in luminance over their surface and seldom appear on uniform backgrounds. Because of the spatial masking effect, the practical contrast threshold is higher than \( h(x, y) \). We define the gain factor \( G \) as the ratio between the practical threshold and Blackwell’s threshold. Thus, the contrast threshold for our application will be

\[
T_B = G \cdot 10^{h(x, y)}
\]  

(4.13)

For a nonuniform background, \( G \) can be as large as 3 [118], while for a high contrast TV display, \( G \) can be as high as 4.5 in paper [119]. In our experiments, a range of \( G = 2.5 - 3.5 \) produced a perceptually lossless compression for a wide range of images.

An immediate generalization of perceptually lossless compression is the suprathreshold compression, where encoding error is above the contrast threshold, as the desired bit rate is too low to provide transparent compression. The suprathreshold compression is perceptually lossy, but still visually optimum in terms of its reaching minimally noticeable distortion. Since the psychophysical data on suprathreshold is very limited, one simple way of determining the suprathreshold is to upshift the contrast threshold until the desired bit rate is reached (i.e., \( G > 3.5 \)). Generally speaking, the higher the value of \( G \) is, the larger the encoding error bound will be, and hence, the higher the compression ratio will be. Thus, the gain factor \( G \) can be used to control the quality of the compressed image and bit rate.

### 4.5 Results and Comparisons

In the above, we have derived a new metric for perceptually based fractal image compression based on the visual nonlinearity of luminance and the visual masking effect. We have implemented the compression scheme according to equation (4.7). The luminance dynamic range of the display and the viewing conditions are given in sections 4.3 and 4.4, respectively. Quadtree partitioning is used for the range blocks. At the initial
node, the encoding error based on equation (4.7) is determined for each range block. Blocks which have an error exceeding the encoding error bound are split into four subblocks to which the same encoding procedure is applied. Figure 4.4 shows the threshold surface of the Lenna image. The thresholds are adaptively determined by block size and local average luminance. For a given block size, brighter blocks allow a bigger reconstruction error, which is consistent with the fact that JND is proportional to the background luminance. At the given luminance level, smaller blocks permit a bigger reconstruction error, because that HVS is less sensitive to small objects. Generally speaking, a perceptually optimized coding method keeps errors just below the visual thresholds everywhere in the image.

![Figure 4.4 Threshold surface of the Lenna image](image)

In the experiments, the fractal codes are uniformly quantized, and encoded with Huffman codes. We have presented the comparison of results in Table 4.1. Figures 4.5, 4.6, 4.7, 4.8, 4.9 and 4.10 show the encoded image Lenna by the perceptually based fractal method, the conventional $L_2$ norm fractal method and the JPEG method, respectively. At a high bit rate, the results are shown at about the same perceptually lossless quality. At the same visual quality, the perceptually based fractal method produces a compression ratio $CR = 11$ (Figure 4.5). The conventional fractal method ($L_2$
norm) gives CR = 9 (Figure 4.6), and the JPEG standard gives CR = 8.2 (Figure 4.7).

Table 4.1 Comparison of results for 512 x 512 Lenna image

<table>
<thead>
<tr>
<th>Methods</th>
<th>CR</th>
<th>PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>HVS</td>
<td>11.01</td>
<td>35.5</td>
</tr>
<tr>
<td></td>
<td>33.64</td>
<td>30.9</td>
</tr>
<tr>
<td>$L_2$ Norm</td>
<td>9.00</td>
<td>36.4</td>
</tr>
<tr>
<td></td>
<td>33.61</td>
<td>31.3</td>
</tr>
<tr>
<td>JPEG</td>
<td>8.17</td>
<td>37.8</td>
</tr>
<tr>
<td></td>
<td>33.64</td>
<td>30.4</td>
</tr>
</tbody>
</table>

We noticed that at about same visual quality, the perceptually based method has a lower PSNR. This phenomenon is due to the fact that the perceptually based method minimizes the visual error by hiding more errors in a less visible region of the image. The method is different from minimizing the mean square error (i.e., maximizing PSNR). The mean-square-error (MSE) criterion is often used for mathematical simplicity in image compression. Such a distortion measure does not reflect the subjective image quality. In the design of quantizers for DPCM coding, Sharma and Netravali showed that [120] MSE is not monotonically related to the subjective image quality. The same MSE may produce different subjective image quality scores, and the same subjective image quality may possess a different MSE. To evaluate the psycho-visual effects of various compression artifacts such as blockiness, ringing and blurring, Chadda and Meng [121] used eight different compression schemes. Compared with the human viewers, the MSE distortion measure produces only 50% - 60% correct ranking of the compressed image. Table 4.1 also contains a set of results at the lower bit rate. Again, the perceptually based method (see Figure 4.8) produces a better visual image than the $L_2$ norm method (see Figure 4.9, the result of which exhibits a blocky effect). We also show the result of the JPEG coded image at the same bit rate (Figure 4.10). The severe blocky effect is obvious. From the comparison, the perceptually based fractal method produces a better reconstructed image than the conventional method and the JPEG, especially at the low bit rate. This fact is
confirmed by visual inspection of the images. Figures 4.11 and 4.12 show the original 512 \( \times \) 512 Chart image and the perceptually based, fractal method encoded image, respectively. Therefore, the method presented in this chapter aims to minimize the perceptual distortion. This goal does not correspond to the maximization of the peak signal-to-noise ratio (the minimization of mean squared error). For a wide range of monochrome images, our algorithm produces a compression ratio between 8:1 to 12:1 without introducing visual artifacts. Under the perceptually lossless criterion, the reported JPEG results produced a compression ratio between 4:1 to 6:1 in [7, 122].

We notice that the equation (4.7) can also be used to evaluate the quality of other block coding image compression methods, such as JPEG, where \( \bar{T} \) is simply the DC coefficient from the DCT. Currently, the drawback of our algorithm is that the iterative procedure for solving parameters \( s \) and \( t \) is time-consuming. Fortunately, the fast pyramidal algorithm of Chapter 3 can be extended to this case. We will present the topic in the next chapter.
Figure 4.5 Perceptually based fractal image compression, 
CR = 11.01, PSNR = 35.5 dB

Figure 4.6 Conventional fractal image compression, 
CR = 9.00, PSNR = 36.4 dB
Figure 4.7 JPEG image compression, $CR = 8.17$, $PSNR = 37.8$ dB

Figure 4.8 Perceptually based fractal compression, $CR = 33.64$, $PSNR = 30.9$ dB
Figure 4.9 Conventional fractal image compression, CR = 33.61, PSNR = 31.3 dB

4.10 JPEG image compression, CR = 33.64, PSNR = 30.4 dB
Figure 4.11 Original 512 x 512 Chart image (Courtesy of IMAX Corp.)

Figure 4.12 Encoded image, CR = 8.00, PSNR = 36.0 dB
Chapter 5

A Pyramidal Algorithm for Perceptually Based Fractal Image Compression

5.1 Introduction

In Chapter 4, we introduced a perceptually appropriate criterion for perceptually based fractal image compression. The method significantly improved the encoding fidelity by using the HVS model. The initial definition of the new metric is given in terms of the average local contrast of the block. After the conversion between the physical and digital representation of the image intensity under the given display conditions, for perceptually based image compression, we have the following inequality:

\[ E = \frac{1}{N} \sum_{n=1}^{N} |I_n - \hat{I}_n| \leq \bar{T} \triangleq \bar{T}_B \]

(5.1)

where \( N \) is the block size, \( I_n \) is the original image, \( \hat{I}_n \) is the fractal encoded image, \( \bar{I} \) is the block mean, and \( \bar{T}_B \) is the visual contrast threshold (4.13). \( \hat{I}_n \) can be represented as an affine transform of the contracted domain block \( l_{m+1, n} \), i.e., \( \hat{I}_n = s l_{m+1, n} + t \), where \( s \) is the contrast scaling factor and \( t \) is the brightness offset. Obviously, the encoding error is measured in terms of a weighted \( L_1 \) norm. Thus, the encoding process needs to make the block matching under the least absolute deviation (LAD) criterion. Like other fractal
based methods, its major drawback is the high computational complexity. This disadvantage is mainly due to the fact that a full search of the domain blocks is needed in order to find the fractal code. To speed up the encoding process, in this chapter, we extend the pyramidal algorithm for $L_2$ norm of Chapter 3 to the one for $L_1$ norm. However, the encoding error threshold sequence for $L_2$ norm is invalid for $L_1$ norm. Based on the Markov random process theory, we rederive the encoding error threshold for each pyramidal level. For the perceptually lossless compression, the encoding error threshold at the finest pyramidal level will be the same as the visual threshold $\bar{T}$. Furthermore, the least square line fitting is different from the LAD line fitting, whose parameters' determination needs an iterative procedure. Therefore, for the parameters of the LAD line, we need to reconsider the propagation rule from coarse to fine pyramidal levels. The pyramidal search is first carried out on an initial coarse level of the pyramid. This initial search increases the encoding speed significantly, because not only the number of the domain blocks to be searched is reduced, but also the data within each domain block is only a fraction of those of the finest level. Then, only a few of the fractal codes from the promising domain blocks in the coarse level are refined through the pyramid to the finest level, and with little computational effort.

5.2 Fast Pyramidal Domain Block Search

We have introduced the pyramidal image model and its application for fast fractal image encoding in Chapter 3. In the following, we will re-formulate the pyramidal framework for perceptual fractal image compression. Let $I(x, y)$ be the original image of size $2^M \times 2^M$. The range block has size $2^m \times 2^m$. From $I(x, y)$, a mean pyramid is created using (3.13), the depth of which depends on the range block size. Because the range block is defined in the image, the range block pyramid will be contained in the image pyramid with the $k$th level of the range block pyramid corresponding to the $(M - m + k)$th level of the image pyramid. The previous optimization objective function (5.1) for the best matched domain block search can be rewritten as:

$$E = \frac{1}{4^m} \sum_{s=0}^{4^m-1} |D_{n}(s, t) - R_{n}| \quad (5.2)$$
where \( D_n(s, t) = s I_{M - 1, n} + t \) is an affine of the scaled domain block, and \( R_n = I_{M, n} = I(x, y) \) is the range block to encode. The contracted domain block image \( I_{M - 1, n} \) is the corresponding block in \((M - 1)\)th level of the pyramid \( I_{M - 1}(x, y) \). Note that for consistency with \((5.1)\), we use a single subscript \( n \) as the index of the pixel at location \((x, y)\). Clearly, \( n = 2^m y + x \).

Instead of a direct search of the minimum of the objective function at the finest level \( m \), we propose a fast algorithm by introducing a smaller, approximate version of the problem at a coarser level \( k \) of the range block pyramid:

\[
E^k = \frac{1}{4^k} \sum_{n=0}^{2^k - 1} |D^k_n(s^k, t^k) - R^k_n| \quad (5.3)
\]

for \( k_0 \leq k \leq m \). Therefore, at range block pyramidal level \( k \), the encoding amounts to finding the best matching domain block of size \( 2^k \times 2^k \) in the image of the size \( 2^{M - m + k - 1} \times 2^{M - m + k - 1} \). For example, for an original image of size \( 512 \times 512 \) \((M = 9)\) and a range block size \( 32 \times 32 \) \((m = 5)\), the search complexity at \( k_0 = 2 \) is that of image size \( 32 \times 32 \) and a range block of size \( 4 \times 4 \). The \( k = k_0 \) level of the range block pyramid is said to be initial and every domain location of the image from the \((M - m + k_0 - 1)\)th level of the image pyramid needs a test. Similar to \((3.16)\), a \( 2^{M \cdot m + k} \times 2^{M \cdot m + k} \) matrix \( G \) is defined to represent the promising domain block locations at level \( k \):

\[
(G)_{2p, 2q} = \begin{cases} 
1, & \text{if } E^k(p, q) < T^k \\
0, & \text{otherwise} 
\end{cases} \quad (5.4)
\]

where \((p, q)\) is the upper left corner coordinate of the domain block, and \( T^k \) is the threshold at level \( k \). Matrix \( G \) is used as a guide in the search of the domain location at the next level \( k + 1 \). Tests are to be performed only at the locations \((i, j)\) for \((G)_{i, j} = 1\) and its neighbour locations. Other parameters \( P^k \) of the promising locations are also propagated to \( P^{k+1} \) for further refining at level \( k + 1 \). For the rotation/reflection index and the domain block location, we have \( \theta^k = \theta, D^k_s = 2D^k_s, D^k_t = 2D^k_t \). Parameters \( s^{k+1} \) and \( t^{k+1} \) can be obtained from the refinement of \( s^k \) and \( t^k \). For the LAD line fitting, it is known that the desired absolute minimum line must pass through at least two points of the given data [123]. Since the refinement of the line parameters only slightly change the
position of the absolute minimum line, the initial reference point in the iterative process of the fine level \( k + 1 \) is best chosen as the last reference point through which the absolute minimum line of coarse level \( k \) passes. Such a choice of the initial reference point will lead to a lesser number of iterations to locate the line of the minimum deviations of level \( k + 1 \). The algorithm provides a gradual refinement of the fractal code. The process is repeated recursively until the finest level \( m \) is reached. At the finest level, if there exists more than one location \((p, q)\), such that \((G)_{p,q} = 1\), select the parameters with the smallest match error as the fractal code. Different thresholds are used to determine promising locations in the corresponding pyramidal levels. The next section shows how to estimate these thresholds using the Markov random process model.

### 5.3 Determining the Thresholds

The two-dimensional encoding error image can be converted into a one-dimensional time series \( X_n \) after row-by-row scanning. According to (5.1), \( X_n = I_n - s I_{M-1,n} - t \). We assume that the time series is modeled as a stationary first-order Markov process. Since the series has marginal Laplacian distribution, it can be represented as a first-order Laplacian autoregressive (LAR(1)) process:

\[
X_n = \rho X_{n-1} + \varepsilon_n
\]  

(5.5)

where \(|\rho| < 1\), and \(\{\varepsilon_n\}\) is a sequence of independent, identically distributed (i.i.d.), zero mean random variables (RVs). The process \(X_n\) is determined in terms of \(X_{n-1}\) and \(\varepsilon_n\). Hence, it is independent of \(X_r\) for \(r < n - 1\). Thus, \(X_n\) is a first-order Markov process. The \(l\)-step correlation coefficient is related to the one-step correlation coefficient \(\rho\) in the exponential form:

\[
\rho_l = \text{Corr}(X_n, X_{n-l}) = \rho^l
\]  

(5.6)

where \(\rho\) is a one-step correlation coefficient of the Laplacian variables \(X_n\) and \(X_{n+1}\).

Assume that the marginal Laplacian distribution density function of \(X_n\) is given by

\[
f_X(x) = \frac{\alpha}{2} e^{-\alpha |x|}
\]  

(5.7)
It can be shown [Appendix B] that the bivariate Laplacian distribution of \( X_n \) and \( X_{n+1} \) can be written as

\[
f_{X_n, X_{n+1}}(x_n, x_{n+1}) = \frac{\alpha \rho^2}{2} e^{-\alpha |x_n|} \delta(x_n - \rho x_{n+1}) + \frac{\alpha^2(1 - \rho^2)}{4} e^{-\alpha |x_n| - |x_{n+1} - \rho x_n|}
\]

where \( \delta(x) \) is the Dirac's delta function. When the parameter \( \alpha = 0.2 \), and the correlation coefficient \( \rho = 0.3 \), figures 5.1a and 5.1b show the first part and the second part of (5.8), respectively. We notice that the density function is not symmetric in \( x_n \) and \( x_{n+1} \), and so the LAR(1) process is not time reversible.

In the following, we derive an approximated distribution function of the encoding error for each pyramidal level, and give an estimation of the thresholds which are needed in the pyramidal search algorithm. At the finest pyramidal level (original image) \( k = m \), the encoding error can be expressed as

\[
E^{(m)} = \frac{1}{4^m} \sum_{n=0}^{4^m-1} |X_n|
\]

In Appendix C, we show that \( E^{(m)} \) has an approximated gamma probability density function:

\[
f_{E^{(m)}}(y) = \frac{\gamma^\beta y^{\beta - 1} e^{-\gamma y}}{\Gamma(\beta)}, \quad (y > 0, \beta > 0, \gamma \geq 0)
\]

where \( \beta = 4^m(1 - \rho_E)/(1 - \rho_E) \), \( \gamma = \beta \alpha \), and \( \rho_E \) is the correlation coefficient between the exponential variables \( |X_n| \) and \( |X_{n+1}| \). \( \rho_E \) is related to \( \rho \) as follows [Appendix D]:

\[
\rho_E = \frac{2\rho^2}{1 + |\rho|}
\]

Let \( P_0 \) be the probability of finding the best match under the threshold \( T^{(m)} \), i.e.,

\[
P(E^{(m)} < T^{(m)}) = P_0, \text{ then } P_0 = I(\mu, \beta - 1), \text{ where}
\]

\[
I(\mu, \beta - 1) = \frac{1}{\Gamma(\beta)} \int_0^\mu t^{\beta - 1} e^{-t} dt, \quad (\beta > 0)
\]

76
Figure 5.1a Bivariate Laplacian PDF part I:
The amplitude of the impulse function

Figure 5.1b Bivariate Laplacian PDF part II:
Bivariate double-sided exponential function
is an incomplete gamma function and has the series representation [124]:

\[ I(\mu, \beta - 1) = \frac{1}{\Gamma(\beta)}(\mu \sqrt{\beta})^{\beta} \sum_{n=0}^{\infty} \frac{(-\mu\sqrt{\beta})^{n}}{n! (n + \beta)} \]  \hspace{1cm} (5.13)

In our application:

\[ \mu = \frac{\gamma T^{(m)}}{\sqrt{\beta}} \]  \hspace{1cm} (5.14)

which is listed in the incomplete gamma function table [125]. Therefore

\[ T^{(m)} = \frac{\sqrt{\beta}}{\gamma} \mu \]  \hspace{1cm} (5.15)

where \( \beta \) and \( \gamma \) are the parameters of the gamma distribution. For perceptually lossless compression, \( T^{(m)} \) is set to the visual threshold:

\[ T^{(m)} = \frac{1}{T_{\beta}} \]  \hspace{1cm} (5.16)

Inserting (5.15) into (5.16), and replacing \( \gamma \) with \( \beta \alpha \), we have

\[ \alpha = \frac{\mu}{\sqrt{\beta} \frac{1}{T_{\beta}}} \]  \hspace{1cm} (5.17)

where \( \alpha \) is the parameter of Laplacian distribution in (5.7).

The derivation of the thresholds at the coarse level \( k \) \( (k_{0} \leq k < m) \) follows a similar procedure to the above, but is more complicated. At pyramidal level \( k \), due to the lowpass filtering, the encoding error signal \( X_{n}^{(k)} \) is related to the fine level \( k + 1 \) signal \( X_{n}^{(k+1)} \) in the form of

\[ X_{n}^{(k)} = \frac{1}{4} \sum_{j=0}^{3} X_{n}^{(k+1)}_{i+j} \]  \hspace{1cm} (5.18)

A reasonable assumption is to consider \( X_{n}^{(k)} \) as an approximate Laplacian distribution given by [Appendix E]
where parameter $\alpha_k$ is related to fine level $k+1$ parameter $\alpha_{k+1}$ as

$$\alpha_k = \frac{2\alpha_{k+1}}{\sqrt{(1 + \rho_{L_{r+1}})(2 + \rho_{L_{r+1}} + \rho_{L_{r+1}}^2)}}$$

(5.20)

where $\rho_{L_{r+1}}$ is the one-step correlation coefficient between Laplacian RVs $X_n^{(k-1)}$ and $X_n^{(k-1)}$. At the finest level $\alpha_m = \alpha$, $\rho_{L_{r+1}} = \rho$.

The objective function in level $k$ is the sum of $|X_n^{(k)}|$ scaled down by the block size:

$$E^{(k)} = \frac{1}{4^k} \sum_{n=0}^{4^k-1} |X_n^{(k)}|$$

(5.21)

It can be shown [Appendix F] that $E^{(k)}$ is an approximate gamma variable and has the density function:

$$f_{E^{(k)}}(y) = \frac{y^{\beta_k-1} e^{-\gamma_k y}}{\Gamma(\beta_k)}$$

(5.22)

where the distribution parameters $\beta_k$ and $\gamma_k$ are given by

$$\beta_k = 4^k \frac{1 - \rho_{\varepsilon_i}}{1 + \rho_{\varepsilon_i}}$$

(5.23)

$$\gamma_k = \beta_k \alpha_k$$

(5.24)

where $\alpha_k$ is given in (5.20), and $\rho_{\varepsilon_i}$ is the one-step correlation coefficient between exponential variables $|X_n^{(k)}|$ and $|X_n^{(k-1)}|$. Equation (5.11) still gives a valid relation between $\rho_{\varepsilon_i}$ and $\rho_{L_{r+1}}$, which is the one-step correlation coefficient between Laplacian variable $X_n^{(k)}$ and $X_n^{(k-1)}$, that is

$$\rho_{\varepsilon_i} = \frac{2\rho_{L_{r+1}}^2}{1 + |\rho_{L_{r+1}}|}$$

(5.25)
In turn, \( \rho_{L_k} \), the correlation coefficient at coarse level \( k \), is related to \( \rho_{L_{k+1}} \), the correlation coefficient at fine level \( k+1 \), by [Appendix G]

\[
\rho_{L_k} = \frac{\rho_{L_{k+1}}(1 + \rho_{L_{k+1}})(1 + \rho_{L_{k+1}}^2)^2}{2(2 + \rho_{L_{k+1}} + \rho_{L_{k+1}}^2)}
\]  

(5.26)

Obviously, at the finest level, \( \rho_{L_n} = \rho \). With the above iterative formula, when \( \rho_{L_{k+1}} \) is given, we can find the next coarse level correlation coefficient \( \rho_{L_k} \), which leads to the solution of \( \rho_{E^k} \). It is expected that in the equation (5.22), when \( \beta_k \to \infty \), \( E^{k+1} \) tends to a Gaussian random variable [126].

The relationship (5.26) shows that \(|\rho_{L_k}| \leq |\rho_{L_{k+1}}|\). It means that when the stationary Laplacian autoregressive model is appropriate, the low resolution image is usually less correlated than its corresponding high resolution version and hence more difficult to compress. On the other hand, given the same encoding error, the fractal codes of the low resolution image can be obtained by coarse quantization of the fractal codes of the higher resolution image [127]. In this sense, multiple bit rates and multiresolution compression can be achieved by the fractal technique.

As we have found the approximate distribution of \( E^{(k)} \), under the assumption that \( P(E^{(k)} < T^{(k)}) = P_0 \), the threshold \( T^{(k)} \) for level \( k \) is given by

\[
T^{(k)} = \frac{\sqrt{\beta_k}}{\gamma_k} \mu_k
\]

(5.27)

where \( \beta_k \) and \( \gamma_k \) are given by (5.23) and (5.24), respectively, and \( \mu_k \) is a number listed in the incomplete gamma function table [125].

**Thresholds Summary**: Given the finest block size \( 4^n \), the parameter of Laplacian distribution \( \alpha \), correlation coefficient \( \rho \), a probability of finding the best match \( P_0 \), the thresholds are derived as follows.

1. At the finest level \( k = m \):

\[
T^{(m)} = \frac{\sqrt{\beta}}{\gamma} \mu
\]

(5.28)
where

\[
\beta = \frac{4^m}{1 + \rho_E} \\
\gamma = \beta \alpha \\
\rho_E = \frac{2\rho^2}{1 + |\rho|}
\]

(5.29)

2. At level \( k (k_0 \leq k < m) \):

\[
T^{(k)} = \frac{\sqrt{\beta_k}}{\gamma_k} \mu_k
\]

(5.30)

where

\[
\beta_k = \frac{4^k}{1 + \rho_{E_k}} \\
\gamma_k = \beta_k \alpha_k
\]

(5.31)

Parameters \( \rho_{E_k} \) and \( \alpha_k \) follow the iterative equations:

\[
\rho_{L_k} = \frac{\rho_{L_{k-1}}(1 + \rho_{L_{k-1}})(1 + \rho_{L_{k-1}}^2)}{2(2 + \rho_{L_{k-1}} + \rho_{L_{k-1}}^2)}
\]

(5.32)

\[
\rho_{E_k} = \frac{2\rho_{L_k}^2}{1 + |\rho_{L_k}|}
\]

\[
\alpha_k = \frac{2\sqrt{2} \alpha_{k-1}}{\sqrt{(1 + \rho_{L_{k-1}})(2 + \rho_{L_{k-1}} + \rho_{L_{k-1}}^2)}}
\]

with the initial conditions \( \alpha_m = \alpha, \rho_{L_0} = \rho \). In the above, the parameter \( \alpha \) is related to the visual contrast threshold \( T_x \) as in (5.17) for perceptually lossless compression.

Example. Consider a block of size \( 8 \times 8 \) \((m = 3)\), given correlation coefficient \( \rho = 0.2 \) and the probability \( P_0 = 0.9 \). By equations (5.28) and (5.30), we get the thresholds for
pyramidal level $k = 3, 2$ and $1$ as follows:

\[
\begin{align*}
\tau^{(3)} &= \frac{1.1739}{\alpha} \\
\tau^{(2)} &= \frac{0.7718}{\alpha} \\
\tau^{(1)} &= \frac{0.5048}{\alpha}
\end{align*}
\]  \hspace{1cm} (5.33)

where $\sqrt{2}/\alpha$ is the standard deviation value of the encoding error at the finest level, which can be obtained from (5.17) for perceptually lossless compression. From the example, we notice that $\tau^{(k)}$ is a monotonic increasing function of the pyramidal level, that is, fine levels have a larger threshold than the coarser levels. To reduce the computational complexity, the thresholds in (5.33) are used for encoding all $8 \times 8$ blocks without the repeating calculation of (5.30) to (5.32) for every block.

### 5.4 Computational Efficiency

In this research, we have used Karst's iterative procedure [123] to determine the parameters $s$ and $t$. Computer simulation results for general, least absolute deviations curve-fitting showed that the actual computation complexity grows linearly with the number of data points [128]. Thus, the computation efficiency analysis in Chapter 3 is still valid and is listed in the following:

\[
Q = \frac{C_1}{C_2}
\]

\[
C_1 = 8 \left( \frac{2^M - 1 - 2^m}{h} + 1 \right)^2 2^{2m}
\]

\[
C_2 = 8 \left( \frac{2^{(M - m \cdot k_c - 1)} - 2^{k_c}}{h(k_0)} + 1 \right)^2 2^{2k_c} + n_p n_r \sum_{i \cdot k_c \cdot i} n_i 2^{2i}
\]

\[
h(k_0) = \max \left( 1, \frac{h}{2^{(m \cdot k_c)}} \right)
\]

82
where \( C_1 \) and \( C_2 \) represent the computational cost of full search and pyramidal search, respectively, \( h(k_0) \) is the search step size at the initial pyramidal level \( k_0 \), and \( Q \) is a computational saving factor. As in Chapter 3, the pyramidal computational saving factor \( Q \) (relative to the LAD full search) depends on the depth of the pyramid and the search step size. For the extreme case, the computation efficiency could be improved up to two orders of magnitude when compared with the LAD full search of the original image.

### 5.5 Coding Results

The algorithm in this chapter is implemented serially on the KSR1 computer. Quadtree partition is used for range blocks. The initial range block size is \( 16 \times 16 \). The encoding error is determined for each range block. Blocks which had an error exceeding the visual suprathresholds (\( G = 5 \)), are split into four \( 8 \times 8 \) blocks. The initial level \( k_0 \) is set to 1. The domain location searches are restricted to one quarter of the full image size. The contractive factor \( s \) and grey level shift \( t \) are coded using 5 and 7 bit uniform quantizers, respectively. Huffman encoding is also used for further compression.

Table 5.1 shows experimental results for the Lenna image using both a full search and pyramidal search method, respectively. When \( h = 2 \), the speed-up is 167. Figures 5.1 and 5.2 show the reconstructed images by full search and pyramidal search method, respectively.

Table 5.2 lists the experimental results for another image, Peppers. Figures 5.3 and 5.4 show the samples of the encoded image.

**Table 5.1 Experimental results for the Lenna image**

<table>
<thead>
<tr>
<th>Step ( h )</th>
<th>Full Search</th>
<th>Pyramidal Search</th>
<th>Speed-up ( Q )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time (m)</td>
<td>PSNR</td>
<td>CR</td>
</tr>
<tr>
<td>2</td>
<td>15365</td>
<td>30.8</td>
<td>26.3</td>
</tr>
<tr>
<td>4</td>
<td>4174</td>
<td>30.5</td>
<td>26.3</td>
</tr>
<tr>
<td>8</td>
<td>1255</td>
<td>30.0</td>
<td>26.3</td>
</tr>
<tr>
<td>16</td>
<td>428</td>
<td>29.2</td>
<td>26.3</td>
</tr>
</tbody>
</table>
Table 5.2 Experimental results for the Peppers image

<table>
<thead>
<tr>
<th>Step $h$</th>
<th>Full Search</th>
<th></th>
<th></th>
<th>Pyramidal Search</th>
<th></th>
<th></th>
<th>Speed-up $Q$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time (m)</td>
<td>PSNR</td>
<td>CR</td>
<td>Time (m)</td>
<td>PSNR</td>
<td>CR</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>19223</td>
<td>30.6</td>
<td>25.1</td>
<td>102</td>
<td>29.4</td>
<td>24.9</td>
<td>188</td>
</tr>
<tr>
<td>4</td>
<td>4626</td>
<td>30.9</td>
<td>24.7</td>
<td>51</td>
<td>30.3</td>
<td>24.9</td>
<td>91</td>
</tr>
<tr>
<td>8</td>
<td>1333</td>
<td>30.5</td>
<td>24.6</td>
<td>20</td>
<td>30.2</td>
<td>25.1</td>
<td>67</td>
</tr>
<tr>
<td>16</td>
<td>404</td>
<td>29.1</td>
<td>25.1</td>
<td>8</td>
<td>29.1</td>
<td>25.0</td>
<td>51</td>
</tr>
</tbody>
</table>

The $L_1$ norm optimization used in the software implementation is the Karst’s iterative weighted median algorithm [123]. Other faster algorithms for the $L_1$ norm optimization are available [117]. Since our software was not optimized and did not use faster $L_1$ norm methods, improvements were expected.

Considering that other fast search techniques such as the conjugate direction search, the 3-step search and the 2-D logarithmic search will lead to relatively larger matching errors as in motion estimation, thus, we conclude that the pyramidal search algorithm is quasi-optimal in terms of minimizing the least absolute error. The main advantage of the pyramidal algorithm is the greatly reduced computational complexity, when compared to the LAD full search.
Figure 5.2 Full search encoded image, CR = 26.3, PSNR = 30.5 dB

Figure 5.3 Pyramidal search encoded image, CR = 26.3, PSNR = 30.2 dB
Figure 5.4 Full search encoded image, CR = 24.6, PSNR = 30.5 dB

Figure 5.5 Pyramidal search encoded image, CR = 25.1, PSNR = 30.2 dB
Chapter 6

Post-Processing and Conclusions

6.1 Removing the Blocking Effects
Using Laplacian Pyramidal Filtering

6.1.1 Introduction
Any practical fractal image compression scheme is block-based coding. At low bit rates, the reconstructed image often produces noticeable blocking artifacts [73]. The blocking artifacts are attributed to the discontinuity of the adjacent block boundary pixels. Their positions in the reconstructed image are determined by the encoding partition scheme (i.e., quadtree partition in our case). The application of a simple lowpass filter cannot effectively remove the artifacts [31]. The use of overlapped partition is helpful in reducing blockiness [129]. Other estimation-based algorithms [130, 131, 132] usually give better performances, but implementations are very complex. We believe that post-processing should be matched to the partition scheme and the human visual response to different frequency bands, so as to provide the best overall performance. In this chapter, we present a method to remove the blocking effect by Laplacian pyramidal filtering, which incorporates a human visual system model. It starts by decomposing each of the four blocks of a quadtree into a set of bandpass channels — the Laplacian pyramid. At each pyramidal level, the adjacent boundary pixels from the four Laplacian pyramids are filtered using a weighted average to remove the blocking effect within the channel. The
four descendant components are then joined to form one level of the parent pyramid. Once this process is done for every level, the filtered image block is recovered from interpolation and summation of the parent pyramid.

6.1.2 Pyramidal Generation for Quadtree Blocks

A study of the human visual system model [133] shows the existence of multiple, spatial, frequency-tuned channel mechanisms in the human vision system. Generally, the use of four to six bandpass filters with characteristics being close to a difference of Gaussians, and with the peak frequency one octave apart, allows a close agreement between the model prediction and the measured data. The information at different spatial frequencies is processed separately. As a sub-optimal approximation, the frequency selectivity of the visual system is modeled by the Laplacian pyramid [134, 135].

Just as we may consider the Gaussian pyramid as a set of lowpass filtered images, we may view the Laplacian pyramid as a set of bandpass filtered images. In general, a spatial bandpass filter can be implemented by simply subtracting one lowpass filter from another [136]. In other words, a bandpass filtered image can be obtained from the difference between two lowpass filtered images. In order to generate multichannel images, we need to generate a set of lowpass filtered images, which is nothing more than the Gaussian

![Figure 6.1 Quadtree partition of image into range blocks](image-url)
pyramid. Let $G_0$ denote an initial level ($k = 0$) of the pyramid. $G_0$ is also one of the four descendant blocks $G_0^0$, $G_0^1$, $G_0^2$ and $G_0^3$ in a certain representation level of a quadtree, as shown in Figure 6.1. Each of the four blocks has the size of $2^n \times 2^n$. An $N$-level Gaussian pyramid is obtained by convolving the image with a set of Gaussian-like functions $w(i, j)$:

$$G_k(x, y) = \sum_{i=-2}^{2} \sum_{j=-2}^{2} w(i, j) G_{k-1}(2x + i, 2y + j)$$

(6.1)

$k = 1, 2, \ldots, N (N < m)$, where the weights $w(i, j)$ are chosen to be separable and approximate to a Gaussian-like function with $w(i) = [0.05, 0.25, 0.4, 0.25, 0.05]$. Since the lowpass filter is approximately Gaussian in shape, the pyramid is called the Gaussian pyramid. $G_k$ is a lowpass version of $G_{k-1}$, and is subsampled by a factor of 2 in each coordinate direction.

Figure 6.2 Four blocks of a quadtree at pyramidal level $k$ and inner boundary conditions

When generating the last column in the pyramidal level $k$ for the first block $G_k$, $w(i, j)$ extends beyond the level $k - 1$ two column pixels, as shown in Figure 6.2. We define the two boundary conditions as:

$$b_{1k}(2^{(m-k)}, y) = \frac{2}{3} G_{k-1}^0(2^{(m-k)} - 1, y) + \frac{1}{3} G_{k-1}^1(0, y)$$

(6.2)

$$b_{2k}(2^{(m-k)} + 1, y) = \frac{1}{3} G_{k-1}^0(2^{(m-k)} - 1, y) + \frac{2}{3} G_{k-1}^1(0, y)$$
These conditions also serve as boundary conditions for the first two columns of the second block $G^1$. The inner boundary conditions for the last two rows of $G^0$ and $G^1$ (same as the first two rows of $G^2$ and $G^3$) are defined similarly. In the filtering process described in the next section, we proceed from the root of a quadtree, so that large quadtree blocks are filtered first. Thus, in Figure 6.2, we only need to consider removing the inner blocking effects of the quadtree blocks. Consequently, the outer boundary conditions are taken as 0.

Level $k$ of the Laplacian pyramidal image is generated by interpolating $G_{k+1}$ and then subtracting it from $G_k$:

$$L_k(x, y) = G_k(x, y) - I[G_{k+1}(x, y)]$$

(6.3)

where $I[.]$ is the spatial interpolation operator, and $L_N = G_N$.

$$I[G_{k+1}(x, y)] = 4 \sum_{i=-2}^{2} \sum_{j=-2}^{2} w(i, j) G_k \left(\frac{x-i}{2}, \frac{y-j}{2}\right)$$

(6.4)

The pyramid generated by (6.3) is called the Laplacian pyramid because it is similar to the output produced by filtering the original image with a Laplacian weighting function. The original image $G_0$ can be restored by reversing the Laplacian pyramid-generating process. $L_N$ is first interpolated and added to $L_{N-1}$ to restore $G_{N-1}$. $G_{N-1}$ is then interpolated and added to $L_{N-2}$ to restore $G_{N-2}$, and so on until $G_0$ is obtained. This interpolation and summation process will be used to reconstruct the parent block from its four bandpass filtered descendant components.

### 6.1.3 The Laplacian Pyramidal Filtering

Corresponding to the four descendant blocks in a quadtree, four Laplacian pyramids $L^0$, $L^1$, $L^2$ and $L^3$ are generated, using the procedure described in the last section. We now construct the parent Laplacian pyramid $L$ by copying pixels from four descendant pyramidal pixels into the appropriate positions. Pixels along the inner block boundary are smoothed by weighting the related two blocks. For blocks 0 and 1, the filtered inner boundary pixels are given by
\[ L_k(2^{m-k} - 1, y) = w(k)L_k^0(2^{m-k} - 1, y) + (1 - w(k))L_k^1(0, y) \]
\[ L_k(2^{m-k}, y) = (1 - w(k))L_k^0(2^{m-k} - 1, y) + w(k)L_k^1(0, y) \]  

(6.5)

Other inner boundary pixels can be defined similarly.

The weights are chosen to match the different visual responses of different channels. In high-resolution, passband channels, the image contains high frequency components, which contribute more to the blocking effect, and therefore the boundary pixel should be averaged more evenly. In low-resolution channels, the images contain mainly low frequency components, which are visually smooth and therefore averaged to a lesser extent. Generally, the weights may be defined as

\[ w(k) = f(k) \]  

(6.6)

\[ k = 0, 1, \ldots, N, \] where \( k \) is the index of resolution level, with \( k = 0 \) being the highest resolution level, and \( k = N \) is the initial level. \( f(k) \) is an increasing function of \( k \), and \( 0.5 \leq f(k) \leq 1 \).

After the construction of the pyramid \( L \), the filtered image is obtained by the interpolation and summation procedure given in the last section.

In paper [137], Burt and Adelson presented a multiresolution spline technique for smoothly combining two images. Our multichannel filtering method differs from the spline technique in three aspects. First, the spline method requires overlapping two images to form a transition zone. In our case, the fractal coded image blocks do not overlap. The application of Burt and Adelson's method requires the extrapolation of the image to form the transition zone, which may not be robust. Our algorithm does not need the overlap requirement. Second, the weights in Burt's algorithm are constant (0.5) over different channels. In our method, the weights vary so as to adapt to the different blocky contributions from different channels. Third, Burt's algorithm uses the first two derivatives of the boundary pixels to define boundary conditions, which may lead to a low PSNR of the filtered image. Our interpolated boundary conditions have avoided such a side-effect.
6.1.4 Experimental Results

During the encoding of the Lenna image, the largest quadtree block is $64 \times 64$ and the smallest, $8 \times 8$. Figure 6.3 shows an enlarged part of the nonlinear contractive function (see Section 2.2) encoded image at 0.1515 bpp. Fractal compression at this bit rate creates prominent blocking artifacts, in addition to other image degradation. The weights are chosen as $w(k) = 0.7 + 0.1k$, for $k = 0$ and 1. Figure 6.4 is the resultant image after post-processing. The blocking artifacts have been almost completely removed.

![Figure 6.3 An enlarged part of the Lenna image, 0.1515 bpp](image1)

![Figure 6.4 After post-processing](image2)

Figures 6.5 and 6.6 show the whole encoded Lenna image and the result after the post-processing, respectively. Although the peak signal to noise ratio (PSNR) of the post-processed image is about the same as the unprocessed one, the improvement in the visual quality is obvious. The overall low quality of the image is due to the very low bit rate encoding rather than to the filtering. Figure 6.7 shows the encoded Peppers image at the bit rate 0.1635 bpp with $\text{PSNR} = 30.0 \text{ dB}$, and Figure 6.8 is the result after post-processing. As described in Chapter 2, fractal image decompression is based on the fixed
Figure 6.5 Nonlinear contractive functions encoded image, 0.1515 bpp, 30.2 dB

Figure 6.6 After post-processing
Figure 6.7 Conventional $L_2$ norm fractal encoded image, 0.1635 bpp, 30.0 dB

Figure 6.8 After post-processing
point theorem which leads to an iterative image reconstruction process. To remove the blocking effects of different scales, the above algorithm can be used following each iteration. Except for the specific partition structure, which is available to the decoder, our algorithm does not need the knowledge of other aspects of the encoding algorithm. Therefore, it can also be used for post-processing of other block-coded images, such as VQ, and transform-coded images.

6.2 Conclusions
In this thesis, efforts have been made to reduce the encoding complexity and to improve the encoding fidelity. The main contributions are summarized below.

1. Proposed an original pyramidal framework for fractal coding. The encoding complexity is reduced by as much as two orders of magnitude.

2. Introduced and derived the threshold sequence for pyramidal searching. The results are useful, not only for fractal coding, but also in general for other multiresolution analysis schemes, such as motion compensation for video coding, and object detection in image analysis.

3. Proposed a method for perceptually based fractal image compression. Based on the human visual system's nonlinear response to luminance and visual masking effects, an appropriate perceptual metric is defined. The psychophysical raw data on visual contrast thresholds was established early during the Second World War. However, it is not clear how to use them in image compression until we interpolate them as a 2-D surface, which is then used as an encoding error bound for perceptually based compression. To represent the visual masking effort, a gain factor is also introduced and used to control the encoding quality. Encoding efficiency is about twice that of transform coding at a given image quality.

4. Extended the fast pyramidal search algorithm to the case of perceptually based
Fractal image compression. A threshold sequence for the perceptual metric is rederived, based on Markov random processes. With the pyramidal algorithm, the encoding complexity is significantly reduced, up to two orders of magnitude. In addition, mathematical results derived in this thesis, such as the joint density function for correlated Laplacian random variables, the relationship of correlation coefficients between pyramidal levels, and so on, may be useful for other compression schemes where statistical models are needed.

5. Introduced nonlinear contractive functions into the fractal coding. The method produces visually better, reconstructed images than the conventional fractal method, while its decoding speed is about twice as fast as that of conventional fractal decoding.

6. Proposed and implemented a parallel method for fractal image compression. The method is suitable for implementation in multiprocessor DSP chips.

7. Proposed a general postprocessing method to reduce the blocking effects for the low bit rate, blocking coding. The method exploits the coding error structure and perceptual relevance of the human visual system model.

6.3 Recommendations

Fractal image compression is based on the contractive mappings from domain to range blocks. As described in Chapter 2, the mapping is contractive in both spatial dimensions and the intensity dimension. For simplicity of implementation, the reduction factor in each spatial dimension is taken as 0.5. However, the research result for binary fractal image compression showed that different reduction factors lead to different compression ratios [138]. For greyscale fractal image compression, further research is needed to find the optimal reduction factor.

Textures in an image can be described by its basic properties such as coarseness, uniformity, roughness and directionality. Fractal geometry uses fractal dimensions to characterize the roughness of an image. High-order fractal dimensions, called multifractal
dimensions, are used to characterize the underlying inhomogeneity of texture in the image [139]. The corresponding inverse problem —— how to use multifractal to encode the texture image —— still needs to be investigated.

The pyramidal search algorithm implemented in this project is based on the mean pyramids. However, other pyramidal structures [140] may be used. For example, the spline pyramid [141], which has less of an aliasing effect in the coarse level, may lead to better matching, and hence, better encoding performance.

Pyramidal algorithms can also be used for fast decoding [106] by performing the iterations in the coarse levels, and then adding the details by iterations in the fine levels.

Obviously, the methods in this thesis can be extended to encode colour images. Because the human visual system is not sensitive to colour information, encoding R, G and B components, separately, is not efficient. To take the advantage of the insensitivity, RGB is converted to YUV space, where U and V channels are subsampled to one-quarter of the original size. Furthermore, the results for Y component, domain-block matching can be used for UV components to reduce the searching complexity. Generally, a colour image has a compression ratio that is larger by a factor of two than that of greyscale image, at about the same perceived quality.

In perceptually based, fractal still image compression, we have used the spatial property of the human visual system. For image sequences, the temporal property comes into play. The existing still fractal image compression algorithms are altered to accommodate image sequences by extending the 2-D methods to a 3-D volume of data [48, 51], or by using intraframe coding with motion compensation prediction techniques [52]. Further investigations are needed to incorporate the human spatial/temporal perception model into video encoding techniques.

For post-processing, it is possible to improve the performance of the multichannel filter. For example, asymmetrical weights may be designed in each channel so that the blocks with a higher encoding accuracy take more. This approach will lead to an increase of PSNR of the processed images. The following are two more ideas which may be used to reduce the blocking effects.

1. The perceived block effects are caused by the discontinuity of the grey level along the block boundary. When the range boundary pixels are approximated more
accurately by the transformed domain boundary pixels, the visual block effects will be reduced. This idea may be realized by giving relatively larger weights to the boundary pixels in the objective function.

2. A second approach is based on a property of the human visual system. When an image is partitioned horizontally and vertically into square range blocks, the block edginess is oriented horizontally and vertically. Experiments on the contrast sensitivity of the human visual system show that the contrast sensitivity is maximum for horizontal and vertical directions and decreases with the angle from either axis to about 3 dB at an angle of 45° [111]. To utilize this property, we propose to partition the image with 45° and 135° oriented lines. As a result, the perceptual block edginess will be about 3 dB lower than the horizontal and vertical block edginess.

Real-time applications of fractal image compression require hardware implementations. The second generation fractal transform 32 Mhz ASIC (FTC-4000) is available from Iterated Systems Inc. [66]. A number of VLSI pyramidal chips have been designed for image analysis applications [103, 142]. FPGA chips and DSP chips have also been used in similar computational-intensive applications [143, 144]. The issue of how to implement our fast pyramidal algorithm in VLSI chips still needs further investigation.
Appendix A

The Coarse Pyramidal Level Thresholds

Let $x_i$ denote the difference between the domain and range block in the finest level $m$ as (3.19) in the Section 3.4. At level $k$ ($k_0 \leq k \leq m - 1$), due to the lowpass filtering, this signal will be

$$x_i^{(k)} = \frac{1}{4^m - k} \sum_{r=0}^{4^m - 1 - 1} x_{i^{(m-k-r)}}$$  \hspace{1cm} (A1)$$

When $4^m - k$ is large, according central limit theorem, $x_i^{(k)}$ is approximately normal with:

$$\mu_{x_i} = 0, \quad \sigma_{x_i}^2 = \frac{1}{4^m - k} \sigma_0^2$$  \hspace{1cm} (A2)$$

The objective function in level $k$ is the sum of squared $x_i^{(k)}$ scaled by the block size:

$$E^k = \frac{1}{4^k} \sum_{i=0}^{4^k - 1} [x_i^{(k)}]^2$$  \hspace{1cm} (A3)$$

As $x_i^{(k)}$ is i.i.d. normal, it can be proved that $E^k$ has chi-square ($\chi^2$) distribution [105]:

$$f_{E^k}(x) = \frac{n (n x)^{n - 1}}{2^{\frac{n}{2}} \sigma_i^n \Gamma(n/2)} e^{-n x/2 \sigma_i^n}$$  \hspace{1cm} (A4)$$

where $n = 4^k$ is the number of the degrees of freedom. For a standard chi-square random
variable, its density function is

$$f_x(x) = \frac{x^{n-1}}{\Gamma(n/2)} \frac{e^{-x/2}}{2^{n/2}}$$

(A5)

When \( n \) is large, the random variable

$$\sqrt{2\chi^2} - \sqrt{2n - 1}$$

(A6)

has an approximate standard normal distribution [145]. Thus, the \( \alpha \) point of the standard chi-square distribution, may be computed from the equation:

$$\sqrt{2\chi^2} - \sqrt{2n - 1} = x_\alpha$$

(A7)

where \( x_\alpha \) is the \( \alpha \) point of the standard normal distribution. Solving (A7), we have

$$\chi^2 = \frac{1}{2} \left( x_\alpha + \sqrt{2n - 1} \right)^2$$

(A8)

Comparing (A4) with (A5), we establish the relationship:

$$\chi^2 = \frac{nT^k}{\sigma^2_{\chi^2}}$$

(A9)

Therefore

$$T^k = \frac{\sigma^2_{\chi^2}}{2n} \left( x_\alpha + \sqrt{2n - 1} \right)^2 = \frac{\sigma^2_0}{4^{\alpha - 4}} \left( 1 + \frac{x_\alpha}{\frac{4^{\alpha - 4} (k+0.5)}} \right)^2$$

(A10)

This is the equation (3.22) in the Section 3.4.

In the above, we have assumed that \( \chi^4 \) is approximately normal. However, at level \( k = m - 1 \), it is only an average of 4 Laplacian variables:
Instead of a normal approximation, its true distribution is given by the following convolution:

\[ f_{X^{(m-1)}}(x) = 4f(4x) \ast f(4x) \ast f(4x) \ast f(4x) \]

\[ = \frac{\alpha}{24} (64\alpha^3|x|^3 + 96\alpha^2x^2 + 60\alpha|x| + 15) e^{-\alpha|x|} \quad (A12) \]

where \( f(.) \) is the Laplacian density function (3.17) in the Section 3.4.

We have

\[ T_{(m-1)} = \frac{\sigma_0^2}{4} (1 + \frac{11}{2^m}) \quad (A15) \]

If we set \( k = m - 1 \) in the threshold equation (A10), we get

\[ T_{c(m-1)} = \frac{\sigma_0^2}{4} (1 + \frac{2\sqrt{2}}{2^m}x_\alpha + \frac{2}{4^m}x_\alpha^2) \quad (A16) \]

The numerical difference between the (A15) and (A16) for practical applications is small. After all, what we need is only an estimation of the threshold. As a result, the range of \( k \) in the threshold equation (3.22) in Section 3.4 is \( k_0 \leq k \leq m - 1 \), where \( k_0 \) is the initial level of the pyramid.
Appendix B

Bivariate Probability Density Function of $X_n$ and $X_{n+1}$

From equation (5.5), the Laplace transform $\phi_{X_n}(s)$ of the distribution $X_n$ is

$$\phi_{X_n}(s) = E[\exp(-sX_n)] = E[\exp[-s(\rho X_{n-1} + \varepsilon_n)]] = \phi_{X_{n-1}}(\rho s) \phi_{\varepsilon_n}(s)$$  \hspace{1cm} (B1)

Assuming that $X_n$ is stationary and solving for $\phi_{\varepsilon}(s)$, we have

$$\phi_{\varepsilon}(s) = \frac{\phi_X(s)}{\phi_X(\rho s)}$$  \hspace{1cm} (B2)

As $X_n$ has a Laplacian distribution in (5.7), thus

$$\phi_X(s) = \frac{\alpha^2}{\alpha^2 - s^2}$$  \hspace{1cm} (B3)

and

$$\phi_{\varepsilon}(s) = \frac{\alpha^2 - (\rho s)^2}{\alpha^2 - s^2}$$  \hspace{1cm} (B4)

The two-dimensional, double-side Laplace transform of the joint probability density
function \( f_{x_i, x_{-i}}(x_n, x_{-n}, \gamma) \) of \( X_n \) and \( X_{n+1} \) is

\[
\phi_{x_i, x_{-i}}(s_1, s_2) = E\{\exp(-s_1 X_n - s_2 X_{-1})\} = E\{\exp[-(s_1 + \rho s_2)X_n]\} \phi_{x_{-i}}(s_2)
\]

\[
= \frac{\alpha^2[\alpha^2 - (\rho s_2)^2]}{[\alpha^2 - (s_1 + \rho s_2)^2](\alpha^2 - s_2^2)}
\]

Expanding \( \phi_{x_i, x_{-i}}(s_1, s_2) \) into the partial fraction form:

\[
\phi_{x_i, x_{-i}}(s_1, s_2) = \frac{\alpha p^2}{2} \left( \frac{1}{\alpha + s_1 + \rho s_2} + \frac{1}{\alpha - (s_1 + \rho s_2)} \right)
\]

\[
+ \frac{\alpha(1 - \rho^2)}{4} \left( \frac{1}{(\alpha + s_2)(\alpha + s_1 + \rho s_2)} + \frac{1}{(\alpha + s_2)[\alpha - (s_1 + \rho s_2)]} \right)
\]

\[
+ \frac{1}{(\alpha - s_2)(\alpha + s_1 + \rho s_2)} + \frac{1}{(\alpha - s_2)[\alpha - (s_1 + \rho s_2)]} \right)
\]

For two-dimensional, double-side Laplace transform, we have the following pairs:

<table>
<thead>
<tr>
<th>( f(x, y) )</th>
<th>( F(s_1, s_2) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. ( e^{-\alpha s}(y - \rho x)U(x) )</td>
<td>( \frac{1}{\alpha + s_1 + \rho s_2} )</td>
</tr>
<tr>
<td>2. ( e^{-\alpha s}(y - \rho x)U(-x) )</td>
<td>( \frac{1}{\alpha - (s_1 + \rho s_2)} )</td>
</tr>
<tr>
<td>3. ( e^{-\alpha s}e^{\alpha y - \rho x}U(x)U(y - \rho x) )</td>
<td>( \frac{1}{(\alpha + s_2)(\alpha + s_1 + \rho s_2)} )</td>
</tr>
<tr>
<td>4. ( e^{\alpha s}e^{\alpha y - \rho x}U(-x)U(y - \rho x) )</td>
<td>( \frac{1}{(\alpha - s_2)[\alpha - (s_1 + \rho s_2)]} )</td>
</tr>
<tr>
<td>5. ( e^{\alpha s}e^{-\alpha y + \rho x}U(-x)U(y - \rho x) )</td>
<td>( \frac{1}{(\alpha + s_2)[\alpha - (s_1 + \rho s_2)]} )</td>
</tr>
<tr>
<td>6. ( e^{-\alpha s}e^{-\alpha y + \rho x}U(x)U(y - \rho x) )</td>
<td>( \frac{1}{(\alpha - s_2)(\alpha + s_1 + \rho s_2)} )</td>
</tr>
</tbody>
</table>

where \( U(x) \) is the unit step function.
Inversion of (B6), after simplification, leads to

\[ f_{x_1 x_2 \ldots}(x_1, x_2, \ldots) = \frac{\alpha \rho^2}{2} e^{-\alpha |x_1|} \delta(x_{n-1} - \rho x_n) + \frac{\alpha^2 (1 - \rho^2)}{4} e^{-\alpha |x_1|} \delta(x_{1} \cdot x_{2} \cdots - \rho x_n) \]  

(B7)

where \( \delta(x) \) is the Dirac's delta function. This is equation (5.8).
Appendix C

Probability Density Function of $E^{(m)}$

At the finest pyramidal level, the encoding error is given as

$$E^{(m)} = \frac{1}{4^m} \sum_{j=0}^{2^m-1} |X_j|$$

(C1)

where $X_j$ is Laplacian distributed random variables with a density function given in (5.7). From $X_j$, let us construct a new random variable $Y_{n+j} = |X_j|$. It can be shown that $Y_{n+j}$ is exponentially distributed with density function:

$$f_Y(y) = \alpha e^{-\alpha y}, \quad (y \geq 0)$$

(C2)

Furthermore, denote $p = n + j$, then $Y_p$ can be modeled as a first-order exponential autoregressive sequence (EAR(1)) [126]:

$$Y_p = \rho_E Y_{p-1} + W_p$$

(C3)

where $\{W_p\}$ is a sequence of independent and identically distributed (i.i.d.) random variables, and $0 \leq \rho_E < 1$. $\rho_E$ is a one-step serial correlation coefficient of $Y_p$. $\rho_E = \text{Corr}(Y_p, Y_{p+1})$. Assuming $Y_p$ is stationary, taking Laplace transform of the distribution of $Y_p$ gives
\[ \phi_Y(s) = E\{\exp(-sY)\} = \phi_Y(\rho_E s) \phi_w(s) \quad (C4) \]

Hence

\[ \phi_w(s) = \frac{\phi_Y(s)}{\phi_Y(\rho_E s)} \quad (C5) \]

From (C2)

\[ \phi_Y(s) = \frac{\alpha}{\alpha + s} \quad (C6) \]

Thus

\[ \phi_w(s) = \frac{\alpha + \rho_E s}{\alpha + s} \quad (C7) \]

Now, constructing \( T_N \) as a sum of the random variables:

\[ T_N = Y_n + Y_{n-1} + \ldots + Y_{n-N-1} \quad (C8) \]

where \( N = 4^m \).

According to (C3), \( Y_{n+j} \) may be written as

\[ Y_{n+j} = \rho_E^j Y_n + \rho_E^{j-1} W_{n+1} + \rho_E^{j-2} W_{n+2} + \ldots + W_{n+j}, \quad (j = 0, \ldots, N-1) \quad (C9) \]

Thus

\[ T_N = \sum_{j=0}^{N-1} Y_{n+j} = \left( \frac{1 - \rho_E^N}{1 - \rho_E} \right) Y_n + \sum_{j=1}^{N-1} \left( \frac{1 - \rho_E^{N-j}}{1 - \rho_E} \right) W_{n+j} \quad (C10) \]

It follows that

\[ \phi_{T_n}(s) = E\{\exp(-sT_n)\} = \phi_Y \left[ s \left( \frac{1 - \rho_E^N}{1 - \rho_E} \right) \prod_{j=1}^{N-1} \phi_w \left[ s \left( \frac{1 - \rho_E^{N-j}}{1 - \rho_E} \right) \right] \right] \quad (C11) \]

Substituting (C6) and (C7) in to (C11), then
Clearly, the exact distribution of $T_N$ can be found by taking the inverse of (C12). The result is analytically awkward, and will not be given here.

Generally speaking, if the random variables $Y_p$ are i.i.d. gamma or Gaussian variables, their sum will also be of the same type. If RVs $Y_p$ are dependent, however, there is no certainty that their sum variable is of the same type. In our application, the coefficient $\rho_E$ of the encoding error is very small ($< 0.3$), the density function of the sum variable $T_N$ may still be approximated by a new gamma density function whose first two moments are identical with the exact density function of the sum variable. In their meteorological application [146], Kotz et al. made a numerical comparison between the approximated and the exact distribution. The discrepancies are relatively small and may be considered insignificant for most practical proposes in the cases of the small values of $\rho_E$. Therefore, $T_N$ is approximately gamma distributed:

$$f_T(t) = \frac{\gamma_T^{\beta_T} t^{\beta_T - 1} e^{-\gamma_T t}}{\Gamma(\beta_T)} \quad (t > 0)$$  \hspace{1cm} (C13)

The first two moments of $T_N$ can be found by the moment theorem:

$$\mu_T = \left. \frac{d\Phi_{T_N}}{ds} \right|_{s = 0} = \frac{N}{\alpha}$$  \hspace{1cm} (C14)

$$\sigma_T^2 = \left. \frac{d^2\Phi_{T_N}}{ds^2} \right|_{s = 0} - \mu_T^2 = \frac{1}{\alpha^2} \left[ N + \frac{2\rho_E}{1 - \rho_E} \left( N - 1 - \frac{\rho_E - \rho_N}{1 - \rho_E} \right) \right]$$  \hspace{1cm} (C15)

Thus, the parameters of the gamma distribution for $T_N$ are
\[ \beta_T = \frac{\mu_T^2}{\sigma_T^2} = \frac{N(1 - \rho_E)}{1 + \rho_E} \]  
\[ \gamma_T = \frac{\mu_T}{\sigma_T} = \frac{1 - \rho_E}{1 + \rho_E} \alpha \]  

The above approximation is made under the condition:

\[ N > 1 + \frac{\rho_E - \rho^N_E}{1 - \rho_E} \]  

Finally, notice that \( T_N \) is related to \( E^{(m)} \) as follows:

\[ E^{(m)} = \frac{1}{N} \sum_{j=0}^{N-1} Y_{n*j} = \frac{1}{N} T_N \]  

Hence, the density function of \( E^{(m)} \) is

\[ f_{E^{(m)}} = Nf_T(t)|_{t=N} \]  

Since \( N = 4^n \), after simplification of (C19), we have

\[ f_{E^{(m)}}(y) = \frac{\gamma^\beta y^{\beta - 1} e^{-\gamma y}}{\Gamma(\beta)} \quad (y > 0, \beta > 0, \gamma > 0) \]  

This is the equation (5.10), where

\[ \beta = 4^m \frac{1 - \rho_E}{1 + \rho_E}, \quad \gamma = \beta \alpha = 4^m \frac{1 - \rho_E}{1 + \rho_E} \alpha \]
Appendix D

Relationship Between Correlation Coefficients $\rho_E$ and $\rho$

Set $Y_n = |X_n|$, $Y_{n+1} = |X_{n+1}|$. As $X_n$ and $X_{n+1}$ are Laplacian RVs, $Y_n$ and $Y_{n+1}$ will be exponential RVs. By definition, $\rho_E$ is the correlation coefficient between $Y_n$ and $Y_{n+1}$:

$$\rho_E = \text{Corr}(Y_n, Y_{n+1}) = \frac{C}{\sigma_{Y_n} \sigma_{Y_{n+1}}}$$  \hspace{1cm} (D1)

where $\sigma_{Y_n}$ and $\sigma_{Y_{n+1}}$ are standard deviations of $Y_n$ and $Y_{n+1}$, respectively, and $C$ is the covariance of RVs $Y_n$ and $Y_{n+1}$:

$$C = E\{Y_n Y_{n+1}\} - E\{Y_n\} E\{Y_{n+1}\}$$  \hspace{1cm} (D2)

From Appendix C, equation (C2), $Y_n$ has the density function:

$$f_Y(y) = \alpha e^{-\alpha y}, \quad (y \geq 0)$$  \hspace{1cm} (D3)

Thus

$$E\{Y_n\} = E\{Y_{n+1}\} = \frac{1}{\alpha}$$  \hspace{1cm} (D4)

As the joint density function $f_{X_n, X_{n+1}}(x_n, x_{n+1})$ is known (C7), the moment $E\{Y_n Y_{n+1}\}$ will be
\[ \sigma^2_{\gamma} = \sigma^2_{\gamma_{\cdots}} = \frac{1}{\alpha^2} \]  \hspace{1cm} \text{(D5)}

\[ E\{Y_n Y_{n-1}\} = E\{ |X_n| |X_{n-1}| \} \]

\[ = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |x_n| |x_{n-1}| f_{X_n, X_{n-1}}(x_n, x_{n-1}) \, dx_n \, dx_{n-1} \]  \hspace{1cm} \text{(D6)}

Inserting (B7) into (D6) and carrying out the integral, we obtain

\[ E\{Y_n Y_{n-1}\} = \frac{1}{\alpha^2} \left( 1 + \frac{2\alpha^2}{1 + |\rho|} \right) \]  \hspace{1cm} \text{(D7)}

Inserting (D7) and (D4) into (D2), we have

\[ C = \frac{2\rho^2}{\alpha^2 (1 + |\rho|)} \]  \hspace{1cm} \text{(D8)}

Inserting (D8) and (D5) into (D1), we conclude that

\[ \rho_E = \frac{2\rho^2}{1 + |\rho|} \]  \hspace{1cm} \text{(D9)}

This is equation (5.11).
Appendix E

Probability Density Function of $X_n^{(k)}$

At level $k$, the encoding error signal $X_n^{(k)}$ is related to the finest level $k + 1$ signal $X_n^{(k+1)}$ by

$$
X_n^{(k)} = \frac{1}{4} \sum_{j=0}^{3} X_n^{(k+1)}
$$

(E1)

Set $p = 4n$, and $Y_{p} = X_n^{(k+1)}$. Then

$$
X_n^{(k)} = \frac{1}{4} \sum_{j=0}^{3} Y_{p}.
$$

(E2)

Assume $Y_p$ is modeled as a first-order stationary Laplacian autoregressive (LAR(1)) process:

$$
Y_p = \rho L_{\circ} Y_{p-1} + \epsilon_p
$$

(E3)

Hence

$$
Y_{p-j} = \rho^j L_{\circ} Y_p + \rho^{j-1} L_{\circ} \epsilon_{p-1} + ... + \epsilon_{p-j}, \quad (j = 0, 1, 2, 3)
$$

(E4)

Construct the sum RV $T_4$ as
\begin{equation}
T_4 = \sum_{j=0}^{3} Y_{p,j}
= Y_p \left( \frac{1 - \rho_{T_4}}{1 - \rho_{T_4}} \right) + \sum_{j=1}^{3} \varepsilon_{p,j} \left( \frac{1 - \rho_{T_4}}{1 - \rho_{T_4}} \right)
\end{equation}

It follows that

\begin{equation}
\phi_{T_4}(s) = \phi_p \left[ \frac{1 - \rho_{T_4}}{1 - \rho_{T_4}} \right] \prod_{j=1}^{3} \phi_{\varepsilon_{p,j}} \left[ \frac{1 - \rho_{T_4}}{1 - \rho_{T_4}} \right]
\end{equation}

where \( \phi(.) \) and \( \phi(.) \) have the same form as in Appendix B equations (B3) and (B4), respectively, except that notations \( \alpha_{k+1} \) and \( \rho_{\alpha_\cdot} \) are used here. \( T_4 \) is the sum of the correlated Laplacian RVs, for which the exact distribution can be found by inverting the \( \phi_{T_4}(s) \). Notice that \( X_n^{(k)} = \frac{1}{4} T_4 \) is the encoding error in level \( k \), it would not be too incorrect to approximate its exact distribution with the same type distribution as the finest level, i.e., Laplacian, with its first two moments the same as the exact distribution of the coarse level. The first two moments of \( X_n^{(k)} \) are

\begin{equation}
\mu_{X_n^{(k)}} = \frac{1}{4} \mu_{T_4} = \frac{1}{4} \left. \frac{d\phi_{T_4}(s)}{ds} \right|_{s=0} = 0
\end{equation}

\begin{equation}
\sigma_{X_n^{(k)}}^2 = \frac{1}{16} \sigma_{T_4}^2 = \frac{1}{16} \left. \frac{d^2\phi_{T_4}(s)}{ds^2} \right|_{s=0}
\end{equation}

\begin{equation}
= \frac{1}{4\alpha_{k+1}^2} (1 + \rho_{\alpha_{k+1}})(2 + \rho_{\alpha_{k+1}} + \rho_{\alpha_{k+1}}^2)
\end{equation}

Thus, the density function of \( X_n^{(k)} \) can be written as:

\begin{equation}
f_{X_n^{(k)}}(x) = \frac{\alpha_k}{2} e^{-\alpha_k|x|}, \quad (\alpha_k > 0)
\end{equation}

where \( \alpha_k \) is the parameter of the distribution and is related to the variance \( \sigma_{X_n^{(k)}}^2 \) by
\[ \alpha_k^2 = \frac{2}{\sigma_{x..}^2} \]  

(E10)

Inserting (E8) into (E10), then

\[ \alpha_k = \frac{2 \sqrt{2} \alpha_{k..}^2}{\sqrt{(1 + \rho_{L..}) (2 + \rho_{L..} + \rho_{L..}^2)}} \]  

(E11)

This is equation (5.20). At the finest level \( m \), \( \alpha_m = \alpha \), \( \rho_{L..} = \rho \).
Appendix F

Probability Density Function of $E^{(k)}$

The proof closely follows the procedure of Appendix C, with the following new notations for level $k$.

<table>
<thead>
<tr>
<th>Notations for Level $k$</th>
<th>Notations for Level $m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E^{(k)}$</td>
<td>$E^{(m)}$</td>
</tr>
<tr>
<td>$4^k$</td>
<td>$4^m$</td>
</tr>
<tr>
<td>$\alpha_k$</td>
<td>$\alpha$</td>
</tr>
<tr>
<td>$\rho_{E_k}$</td>
<td>$\rho_E$</td>
</tr>
<tr>
<td>$X^{(k)}_n$</td>
<td>$X_n$</td>
</tr>
<tr>
<td>$Y^{(k)}_p$</td>
<td>$Y_p$</td>
</tr>
<tr>
<td>$W^{(k)}_p$</td>
<td>$W_p$</td>
</tr>
<tr>
<td>$\beta_k$</td>
<td>$\beta$</td>
</tr>
<tr>
<td>$\gamma_k$</td>
<td>$\gamma$</td>
</tr>
</tbody>
</table>
From Appendix C, equation (C20), we can obtain the density function of $E^{(k)}$ as

$$f_{E^{(k)}}(y) = \frac{\gamma_k^{\beta_k} y^{\beta_k - 1} e^{-\gamma_k y}}{\Gamma(\beta_k)}, \quad (y > 0, \beta_k > 0, \gamma_k \geq 0) \tag{F1}$$

where $\beta_k$ and $\gamma_k$ are parameters of the gamma distribution. Because $\rho_{E_i}$ is very small in the coarse level, and since $N = 4^k$, the condition (C17) in Appendix C is still valid. Hence

$$\beta_k = \frac{4^k (1 - \rho_{E_i})}{1 + \rho_{E_i}} \tag{F2}$$

$$\gamma_k = \beta_k \alpha_k$$

where $\alpha_k$ is given in Appendix E, equation (E11).
Appendix G

Relationship Between Correlation Coefficients $\rho_{L_k}$ and $\rho_{L_{k-1}}$

At coarse level $k$, the encoding error signal $X_n^{(k)}$ is related to the fine level $k+1$ signal $X_p^{(k-1)}$ as

$$X_n^{(k)} = \frac{1}{4} \sum_{j=0}^{3} X_j^{(k-1)}$$

where $4n + j = p$. Assume $X_p^{(k-1)}$ is modeled as a first-order, stationary, Laplacian autoregressive (LAR(1)) process, then [111]

$$E[X_p^{(k-1)}X_p^{(k-1)}] = \sigma_{X^{(k-1)}}^2 \rho_{L_{n_{(k-1)}}}$$

where $\rho_{L_{n_{(k-1)}}}$ is the one-step correlation coefficient of RV $X_p^{(k-1)}$, and $\sigma_{X^{(k-1)}}^2$ is the variance of $X_p^{(k-1)}$. From (G1), $X_n^{(k)}$ is a weighted sum of correlated RVs $X_p^{(k-1)}$. As was done in Appendix E, it can be shown that $X_n^{(k)}$ has the following mean and variance, respectively:

$$\mu_{X_n^{(k)}} = 0$$

$$\sigma_{X_n^{(k)}}^2 = \frac{1}{8} \sigma_{X^{(k-1)}}^2 (1 + \rho_{L_{n_{(k-1)}}}) (2 + \rho_{L_{n_{(k-1)}}} + \rho_{L_{n_{(k-1)}}}^2)$$

116
Furthermore, $X_n^{(k)}$ and $X_n^{(k-1)}$ have the covariance:

$$C = E[(X_n^{(k)} - \mu_{X_n})(X_n^{(k-1)} - \mu_{X_n})] = E(X_n^{(k)}X_n^{(k-1)})$$

$$= \frac{1}{16}E\{(\sum_{j=0}^{3} X_n^{(k-1)j})(\sum_{j=0}^{3} X_n^{(k-1)j})\}$$

Inserting (G2) into (G4) and simplifying, we have

$$C = \frac{1}{16} \sigma_{X_n}^2 \rho_{L_{n...},}(1 + \rho_{L_{n...},})^2 (1 + \rho_{L_{n...},}^2)$$

By definition, the one-step correlation coefficient of $X_n^{(k)}$ is

$$\rho_{L_{n...},} = \frac{C}{\sigma_{X_n}^2}$$

$$= \frac{\rho_{L_{n...},}(1 + \rho_{L_{n...},})(1 + \rho_{L_{n...},}^2)}{2(2 + \rho_{L_{n...},} + \rho_{L_{n...},}^2)}$$

In general, $|\rho_{L_{n...},}| \leq 1$, it can be verified that

$$|\rho_{L_{n...},}| \leq |\rho_{L_{n...},}|$$

and the same holds only for $\rho_{L_{n...},} = 0$ or 1. Relationship (G7) means that when the first-order, stationary, Laplacian autoregressive (LAR(1)) model is appropriate, a low resolution version of an image is usually less correlated than its high resolution version.
References


[40] N. T. Thao, "A hybrid fractal-DCT coding scheme for image compression,"


European Signal Processing Conference, p308-311, Trieste, Italy, Sept. 1996


122


[80] H. Lin and A. N. Venetsanopoulos, "Post-processing block coded images using the human visual system (HVS) model," *Proc. IASTED International Conf. on Signal*


