Kiselyov, Oleg E., Multiresolutional/Fractal Compression of Still and Moving Pictures. Doctor of Philosophy (Computer Science), December, 1993, 134 pp., 3 tables, 51 illustrations, bibliography, 30 titles.

The scope of the present dissertation is a deep lossy compression of still and moving grayscale pictures while maintaining their fidelity, with a specific goal of creating a working prototype of a software system for use in low bandwidth transmission of still satellite imagery and weather briefings with the best preservation of features considered important by the end user.

Among the major results is a set of pyramidal compression algorithms with a loose wavelet basis specifically designed to reduce the entropy of the image representation as much as possible. Design principles are discussed and tradeoffs exposed. A number of examples of compressed and restored sample images are provided to demonstrate the quality of particular schemes.

A modification of the wavelet transform is introduced that lets the user control the amount of distortion and compression for arbitrary specified image areas. A particular feature of this non-uniform compression scheme is a seamless and smooth incorporation of almost non-distorted information into broader context of large-scale features. Examples are provided.

A regularized discrete derivative of an image is developed which effectively removes the local background and the fine-scale noise. Therefore, it may be used for localizing image patterns regardless of the lighting conditions, etc.

A discovery of the property of self-similarity of the pyramidal image transform has opened up an entirely new approach to compression: zooming out from a (possibly shrunken) low-resolution image producing a sharp and crisp “natural looking” high-resolution view. It is demonstrated that the technique has features of preserving thinness of lines on expansion, translational invariance and providing a perfect high-resolution representation of the gradient fill.

The multiresolutional transform algorithms and “smart” image magnification developed for still images have been generalized to deal with moving pictures as a three-dimensional, spatio-temporal frame sequence, which permits rapid compression, and has potential for use in video transmission in real time.

MULTIRESOLUTIONAL/FRACTAL COMPRESSION OF STILL AND MOVING PICTURES

Oleg E. Kiselyov, Univ. Diploma, Cand.Sci.

APPROVED:

____
Major Professor

____
Committee Member

____
Committee Member

____
Committee Member

____
Chair of the Department of Computer Sciences

____
Dean of the Robert B. Toulouse School of Graduate Studies

MULTIRESOLUTIONAL/FRACTAL COMPRESSION OF STILL AND MOVING PICTURES

DISSERTATION

Presented to the Graduate Council of the
University of North Texas in Partial
Fulfillment of the Requirements

For the Degree of

DOCTOR OF PHILOSOPHY

By

Oleg E. Kiselyov, Univ. Diploma, Cand.Sci.
Denton, Texas
December, 1993

ACKNOWLEDGMENTS

I would like to thank Dr. Paul S. Fisher for his help, advice, comments, and encouragement throughout this research. I am also grateful to Garland Andrews for his assistance in preparation of the manuscript.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

CHAPTER I

INTRODUCTION

1.1 The Goal and its Significance

The goal of the present research is a deep compression of still and moving grayscale pictures. The purpose was to perform basic research, develop algorithms and implement them in a working prototype of a software system for use in low bandwidth transmission of still satellite imagery and weather briefings. The software should also be suitable for other applications that would require video transmission at such limited bandwidth. A characteristic feature of the goal that determines all aspects of the research is a deep lossy compression of still and moving imagery while maintaining fidelity, which is dictated by the limited bandwidth requirement. The other feature of the present project is an extensive use of the multiresolutional image/video analysis. Specifically it includes developing modifications and extensions of the wavelet transform and exploring a discovered local self-similarity of the wavelet decomposition.

Over the past few years, a vast variety of communities, both military and non-military, are coming to realize the demand and urgent necessity for data compression, especially of images and other two-dimensional still and continuous data. First of all, it is called for by the need to transmit ever increasing amounts of information over low capacity channels. This is clearly stated in a request for research for the department of Navy: “However, due to the limited data transmission bandwidth (2 to 25 kHz) available from secured satellite channels, the need exists to develop a means of efficient imagery transmission without sacrificing resolution in areas of interest within the imagery” [SBIR93]. Data compression is even considered a stipulation in future progress of space exploration: “Space Based observations require the transmission of a variety of different types of data through the spacecraft communications on the ground. The data must be sent through a communications system that is limited in bandwidth and is being shared among sensors. Greater utilization of the limited resources of an observation system can be accommodated by the use of data compression. Data compression is an *enabling technology* that interfaces to many components of a Space Based Observation System” [DCP92]. Data compression is becoming increasingly important in efficient archiving and retrieval of image data, for example, unprecedented volumes of space data expected from Earth Observation System instruments [NOVI93]. Another example is a compact storage of fingerprints in the computer, which is essential to the development of the forthcoming Automated Fingerprint Identification System [HOPP92].

The techniques enabling video compression are key to implementation of multimedia. One such application of technology is the ability to provide video briefings to both ships in a fleet as well as to offices scattered over the world from a central source, and using low communication speed. Another application is video-conferencing, emerging on the market by efforts of AT&T, PictureTel, Northern Telecom, IBM, and a number of other companies. Though the role of video is only supplementary to audio conferring and file exchange, it takes special hardware to acquire, transmit, and display moving pictures of talking parties in real time; and it puts an immense burden on the communication lines (which in many instances are regular telephone lines). Any effort to accelerate the video processing and to reduce the bandwidth requirements necessary for video transmission while keeping the visual quality acceptable should be taken to make the teleconferencing technology more cost effective relative to communication bandwidth. The last, but not the least, application of the efficient video compression is the emerging digital television, which is entirely contingent upon it.

In most, if not all cases, a deep compression of imagery data is required, which comes from the stipulation that information is to be transmitted over low capacity channels having a limited bandwidth. Transmitting a typical weather satellite 512x512 image with 256 shades of gray over a telephone line or similar channel with capacity around 20K bits/sec within a couple of seconds requires compression ratios of 30:1 to 100:1. As far as the video transmission is concerned, compression ratios starting from 50:1 and up to 2000:1 are necessary to play eight-128x128 frames a second up to the high end of twenty-512x512 frames/sec in real time.

Despite the extreme significance of the topic, deep image and video compression has yet to be properly investigated. Various standards developed for compression of individual images, or images arranged in a temporal sequence, do not address the high levels of compression required.

For example, a popular JPEG standard does not allow a satisfactory compression of still images with ratios of more than 10:1 to 15:1. Beyond this limit, the quality of the restored image quickly deteriorates to a point where it becomes completely unacceptable. The most promising approaches to deep compression of still pictures are wavelet and fractal compression. These techniques have emerged less than a decade ago and by no means have been fully investigated. It is also the case that the performance is still erratic, sometimes excellent, sometimes not very good. Moreover, existing implementations do not properly address the question of the visual and other consequences of the data loss due to high levels of compression. The problem of which type of image deterioration would be considered most acceptable by a particular group of end users has yet to be investigated, as well.

As far as the video compression is concerned, the somber state of the art was reflected in a talk given at the Data Compression Conference '93 by Dr. Richard Baker, a chief scientist of PictureTel Corporation [BAKE93]. He was practically pleading with the audience to apply promising still image compression techniques to moving pictures, to design new methods of motion compensation, etc. "Almost everything," he asserted, "would do better than existing H.261 and MPEG standards."

1.2 How Compression is Accomplished: Methods of Attack

This section outlines the design issues needed to be addressed in the present research and provides justification of the techniques and major decisions that have been adopted in the course of this study.

• Deep image compression is lossy

The prime objective of any compression technique is to remove redundant information. If the amount of information removed still allows one to reconstruct the original data identically, the compression scheme is called lossless. Techniques that work beyond that limit are lossy in that some information deemed to be insignificant is irreversibly lost as the result of the compression. Lossless and lossy data compression algorithms are each suitable for different kinds of applications. Lossless algorithms, which take advantage of allocating different number of bits to different characters, sequences of characters, or patterns to reduce the data to a compressed form, preserve the original data precisely. It should be noted that since lossless algorithms operate within the perfect reconstruction limit, the amount of compression should not be expected to be very high; generally it does not exceed 2-3 times for typical pictures.

Lossy compression algorithms allow some approximations to be made in the process of bit allocation (by disregarding differences between distinct but "similar" patterns or units of data and treating them as identical). When dealing with imagery, a lossy compression method is acceptable and often preferable. The primary reason being that imagery information is almost never absolutely precise in that it always contains noise due to the intrinsic inaccuracy of the equipment used to produce the imagery, e.g., sensors, amplifiers, digitizers. Therefore, introducing additional distortion due to the lossy compression while storing and/or communicating the imagery information is acceptable, as long as the signal-to-noise ratio stays within acceptable limits. Moreover, it is only with the lossy compression algorithms that one can hope to achieve the deep compression discussed above.

- Eliminating redundant information: Spacio-temporal correlations

In compressing images, the redundant information manifests itself in local correlations of pixel intensities. This means that the value of a pixel does not deviate much from the values of its neighbors, and can be fairly accurately predicted from the pixel’s surroundings. This is even more the case in video images, where temporal correlations between adjacent frames exist, in addition to the spatial correlations within a frame. Indeed, pixels of the background stay the same for a number of frames. Moving parts are usually relatively big and move slowly (compared to the frame rate). This means the movement of pixels can generally be predicted by simple extrapolation of motion detected in the previous frames. Therefore, any image compression algorithm should effectively eliminate these correlations: the more prediction can be utilized, the better compression would be. Hence, the possibility for deep compression of the video imagery lies in taking full advantage of the correlation as mentioned, and omitting the constant or unchanging information that can be inferred. Wavelet decomposition and the fractal image compression based upon the property of the local image self-similarity are the two most promising techniques for decorrelating the pixel values.

• Minimizing loss of the **relevant information:** Non-uniform compression

The very nature of environmental images, or any image for that matter, suggests that not every detail of the picture is equally important to the observer. For example, the area of the hurricane eye on a satellite image should be of high resolution, while the tight cloud cover of the hurricane body is less informative and may be rendered with a lower resolution, though it cannot be completely discarded. In disseminating the weather information to ships, a meteorologist at a particular ship needs very accurate data on cloud cover, wind direction, temperature, etc. just in the vicinity of his ship. The information about what is going on outside that small area is used for prognosis and does not have to be of very high precision. Accordingly, the amount of loss and inaccuracy that can be tolerated during the communication varies not only from one user to another but also from one region of the image to another. This raises the problem of a non-uniform, lossy compression, i.e., compression where the loss varies with the location/features/frequencies, etc., and tailoring such compression to a particular user and circumstances. Preserving the information during compression to the extent the user needs, but not more, helps to drastically reduce the amount of data that has to be transmitted.

Multiresolutional image analysis is the proper tool because it allows one to both select the information which is relevant to a particular user (in automatical mode or interactively), and to transmit the selected data with the minimal number of bits. Moreover, multiresolutional analysis based upon the coarse-to-fine image processing is naturally designed for a progressive transmission that provides fast and efficient browsing. Chapter II discusses the multiresolutional transform and its advantages in greater detail.

- Rapid processing: a necessary requirement

Finally, processing should be fast enough to carry out compression and decompression on-line. It is especially the case for video, where the ability to compress and decompress a set of frames as it plays is a natural requisite.

1.3 Structure of the Thesis

Chapter II provides basic definitions and other necessary background in the techniques that have been used extensively in the present study. The multiresolutional analysis, which is a principal tool in this research, receives most of the attention. Fractal image compression and current standards of video compression are covered, as well.

Chapter III thoroughly investigates a particular scheme of the pyramidal image decomposition with loose wavelet bases and a number of its modifications. The motivations behind the algorithms specifically designed to reduce the entropy of the image representation as much as possible is explained in detail. Though there is no clear winner, several particular wavelet compression schemes proved to be successful in providing deep compression at the required level while maintaining fair to good quality of the restored image. The results are comparable with those achieved to date by other researchers [DCC93]. The developed algorithms differ in the nature and the amount of distortion incurred and the degree of compression. Since for some applications one kind of distortion is more acceptable than the other, the final choice is left to the user. The chapter gives the compression results and provides illustrations so that the reader can judge for himself. One of the first implementations of the non-uniform compression in the multiresolutional context is presented as well.

Chapter IV introduces a combined fractal/multiresolutional image compression based on the discovered property of self-similarity of the wavelet image transform. The property of self-similarity is elucidated in a variety of examples. They also help reveal such features of the property as the ability to perform the magnification of the picture that keeps thin lines thin and gives a sharp and crisp ‘naturally looking’ high-resolution view without blockiness and jaggedness. The chapter introduces a new approach to image compression based on the ‘smart’ magnification thus defined. A number of examples are provided to illustrate the capability of the method.

A generalization of the multiresolutional pyramidal approach to deal with moving pictures is discussed in Chapter V. It introduces algorithms of wavelet pyramidal decomposition of a three-dimensional spatio-temporal frame sequence. The chapter also observes that the transform coefficients arranged in the form of a four-dimensional octpyramid possess the property of self-similarity that was noted for the pyramidal decomposition of still pictures. Implications of the discovery to the real-time video compression are discussed and illustrated.

The last section of the dissertation, Conclusions, provides a summary of the results and outlines their possible extensions and directions for further research. Applications of the present study in the public and private sectors are discussed as well.

1.4 Achieved Results

The main results accomplished in this dissertation are:

- It is demonstrated that the entropy of the pyramidal representation has little to do with the number of basis functions and decomposition coefficients. It justifies the ***quest for the minimum entropy pyramid***; some criteria for the quest have been outlined. This suggests that, as far as image compression is concerned, one needs to consider not the number of basis functions but the entropy of the representation in designing the best image transformation. The redundancy, where it exists, can be exploited to help reduce the entropy of the resultant representation.
- Using this general understanding, a generic Laplacian pyramid algorithm and a number of modifications and pyramid decomposition schemes have been investigated. Several methods were found to provide the required deep compression with fair to good quality for the restored image.
- Non-uniform wavelet compression schemes have been designed, which marks one of the first developments of the non-uniform image

compression within the multiresolutional context. The technique selectively preserves image features according to the degree of importance assigned by the user to image areas. The notion of criteria sets is introduced, which allows one to formulate arbitrary rules for specifying important image characteristics. The method thus minimizes the distortion of the important features during the lossy compression and still achieves deep reduction in the information content of the image.

- The pyramidal decomposition was shown to greatly expedite linear operations on images, such as image filtration, edge crisping, and edge detection. New operations, a discrete total derivative of the picture and multiresolutional filtration, were introduced and implemented based upon the wavelet transform.
- A mixed context predictive model for the arithmetic compression of the image transform has been designed and finely tuned to the properties of the pyramidal decomposition. The experiments showed that this specialized encoder outperforms general methods such as LZC, adaptive Huffman and the generic arithmetic encoder in compressing the pyramid.
- During the course of investigation of wavelet transforms for images and image sequences, a discovery was made that a multiresolutional pyramidal wavelet decomposition of still and moving pictures does possess a property of local self-similarity. It has opened up an entirely new approach to compression: zooming out a given sequence of frames both in space and in time; in other words, magnifying a separate frame to discern greater detail as well as increasing the time resolution to produce smoother motion.

- A Laplacian octpyramid has been introduced as a multiresolutional representation of an image sequence. The pyramid was designed by generalization of the methods developed for still images to the three-dimensional space-time sequence of frames. Simple examples of video images; as well as a clip of a real weather briefing, gave evidence that the octpyramid is well suited for compression of image sequences and their enhancement based on the local self-similarity.
- An image and video manipulation language has been designed, which is based upon the C++ class library for dealing with images, rectangular areas, and image sequences.
- A portable software implementing quadtree/octtree construction and manipulation, pyramidal decomposition of still images and octpyramidal decomposition of frame sequences, local self-similarity estimation of the transforms and "smart" image magnification, arithmetic compression, and communicating imagery between the mainframe and Macintosh computer has been developed, totaling 24,000 lines of C++ code. The designed software allows the user to go from video acquisition through compression, telecommunication, and decompression to playback, and constitutes an engineering prototype of the still or video compression system.
- A real-time video compression of 128x128 frames at the rate of 4-5 frames/sec with fairly good quality has been achieved as the outcome of the present research. A VHS tape of a weather briefing has been produced that contains the original video clip and those compressed and

re-expanded using several developed algorithms, playing side-by-side. The tape is available upon request.

1.5 Related Work

The present research has been sponsored in part by the Navy as part of the following projects, where the author has been the technical director:

1) SBIR N00039-92-C-0063 Phase I *Compression of Geophysical Data*, November 1, 1991 - April 30, 1992 (N91-016)

This project was concerned with the compression of data types found in the repertoire of geophysical data elements. Such elements included images, binary data, isoplots, and text, among others. During this six-month period, we investigated the JPEG standard compression technique applied to single image data. Additionally, a wavelet compression algorithm EPIC (Efficient Pyramid Image Coder) was acquired and tested against the JPEG results. The results of this effort convinced us that the wavelet algorithm was better at higher levels of compression than JPEG, and so our research focus narrowed for image compression to consider wavelet compression as the focal point for further studies.

2) Battelle Scientific Services Program 91-589-0058 *Compression of Naval Environmental Data* , September 24, 1991 - January 31, 1993

In concert with the previous effort, this research centered on the development of a measure to better evaluate the effects of compression upon an image. Also from the results of the previous project, when the use of wavelets was decided upon as the compression technique of preference, an algorithm for non-uniform compression over an image was developed. As a second aspect of this work, an optimal arithmetic compression algorithm was developed which would work with the new implementation of the wavelet algorithm which used a non-orthogonal basis. With this algorithm we were able to test higher levels of compression, and work on eliminating the side effects of compression, such a blockiness and loss of definition. As a last effort in this contract, the algorithms were taken to a Navy site and used for end-to-end transmission of Navy environmental data. At the conclusion of this contract, an early form of the wavelet compression algorithm existed, an optimal arithmetic compression algorithm existed, and an error measure more closely matching human visualization was created.

3) SBIR N00039-93-C-0048 Phase I *Video Compression*, December 22, 1992 - June 22, 1993

In this effort, the extension of the wavelet algorithm developed in the previous projects to include fractal compression of the wavelet coefficients was achieved. This work, using the results from both of the previous efforts has provided the ground work to allow the development of a software system for compression.

CHAPTER II

BACKGROUND EXPLANATION OF THE SUBBAND AND FRACTAL IMAGE CODING

2.1 Multiresolutional Image Analysis and Wavelet Transform

Multiresolution transforms have been thoroughly studied in computer vision. At different resolutions, the visible details of an image generally characterize different types of physical structures. One of the popular examples of that is a satellite image of a coastal region [MALL89b]. A coarse resolution image gives a description of only the overall shape of the coast. When the resolution of the image is increased, we are able to successively distinguish the local relief of the region, and if the resolution gets even finer, we can recognize the various types of local vegetation. In order to process these different structures separately, researchers have attempted the extraction of the change in information between the approximation of an image at two different resolutions. The efforts resulted in a multiband image decomposition, the representation of an image as a sequence of details corresponding to progressively increasing resolutions. The coarser the resolution, the fewer details are available, so the stack of image representations at increasing resolutions appears as a pyramid; therefore the multiresolutional transforms are often referred to as a pyramidal image representation.

Separating image features according to their ‘scale’, that is, classifying them into appropriate resolution subbands, is equivalent to passing the image through a two-dimensional (or three-dimensional in case of the video) filter bank, a collection of filters where each filter passes only one particular ‘frequency/resolution band’ of image features and cuts off everything else. Each of the bands can be subsampled. This by itself provides compression gain as coding of subbands outperforms direct coding of the full-band original, which is discussed in more detail below.

The multiband image decomposition is an alternative image representation other than just a pixel matrix. From the mathematical point of view, it can be regarded as a decomposition of the image into a set of some basic features (primitive images):

$$image_{ij} = \sum_{r} c^r \Phi^r(i,j) \tag{1}$$

Here Φ^r represents a particular ‘feature’, and the corresponding decomposition coefficient c^r signifies the extent to which the feature appears in the picture; $image_{ij}$ is the value of the image pixel at the i-th row and the j-th column and $\Phi^r(i,j)$ is the value of the (i,j)-th pixel of the r-th (basis) feature. It should be noted that the set of coefficients $\{c^r,\ r = 0..rmax\}$ represents the image as well as the pixel matrix $image_{ij}$ does. Indeed, given the set of coefficients, the pixels can be reconstructed according to eq. (1). We will refer to such change of representation from the pixel matrix $image_{ij}$ to the set of coefficients as the transformation of the image; the set { c^r } itself is often called the image transform. When functions Φ^r possess some special properties, coefficients c^r might turn out to be easier to deal with. In particular, the set of coefficients may have less entropy, and, therefore, it can be encoded with fewer bits. This is indeed the case.

There is almost an unlimited number of ways of choosing the functions Φ^r and estimating the coefficients c^r. One requirement that is often imposed is that the coefficients should be computed as a linear combination of pixel values. In other words, the transformation should be linear. Though there is no ironclad rule that it cannot be done otherwise, linear transformations are easier to use, compute and analyze, and there are several important mathematical results established about such transforms. There are some other properties of the basis functions which are considered highly desirable [SIMO91], they are:

- An explicit representation of scale. Several authors [MALL89a,b] have argued that the correct partition in terms of scale is one in which the scales are related by a fixed constant of proportionality, generally by a factor of two. In the frequency domain, this corresponds to a decomposition into localized subbands with equal widths on the logarithmic scale. There is some experimental evidence [MALL89b] that the human visual perception follows the same pattern;
- Spatial localization in addition to localization in frequency; i.e., the transform should encode position information;
- Orientation. For two dimensional signals, a localized region in the frequency plane corresponds to a particular scale and orientation. Orientation of the basis functions specifically allows the transform to extract higher order structures typically found in images, such as edges and lines;

- Orthogonality. It is usually justified in terms of decorrelation, with the reference to the Karhunen-Loeve transform which is orthogonal and decorrelates the signal, i.e., makes the covariance matrix diagonal. Although there are some other advantages to be discussed later, certain types of non-orthogonal transform appear to perform better in some situations (Chapter III).

In order for eq. (1) to qualify as a multiresolutional, or subband, image decomposition, the basis functions Φ^r should satisfy the property of space-frequency localization deemed to be fundamental for that decomposition. This means the basis function should differ from zero only within a relatively small compact area of the image and should describe features that can be characterized by a single scale. The coefficient beside a particular basis function in the decomposition (1) can then tell how the corresponding region of the image (where the function significantly differs from zero) at the specific scale is represented in the image. To emphasize the fact of the space-frequency localization, we will split the index r labelling a basis function Φ^r and the decomposition coefficient into three separate indices:

$$image_{ij} = \sum_{k,l,m} c^k_{lm} \Phi^k_{lm}(i,j) \qquad (2)$$

Superscript k indicates the 'scale' of the function, while the *(l, m)* indices specify the localization of the function $\Phi^k_{lm}(i,j)$ according to that scale. Note, since the image details can be characterized by only a few samples at a coarse resolution, the range for l and m gets smaller as k decreases (generally by a factor of 2).

One particular type of the subband decomposition exists when the separate filters in the bank are constructed according to a single rule. This representation has been discovered to be closely related to the wavelet transform [MALL89a]. We will consider this first in the one-dimensional case for simplicity. Wavelets have been introduced by Grossman and Morlet [GROS84] as functions $\psi(x)$ whose translations and dilations

$$\sqrt{s}\,\psi(sx-t)\Big|_{s\in R^{+},\, t\in R}$$

can be used for expansions of $\mathbf{L}^2(\boldsymbol{R})$ functions. Moreover, it has been shown that there exist some wavelets $\psi(x)$ such that

$$\sqrt{2^k}\ \psi(2^k(x-2^{-k}l))\Big|_{k\in Z,\, l\in Z}$$

is an orthogonal basis of $\mathbf{L}^2(\boldsymbol{R})$. See [MALL89a,b and references therein]. In the case of discrete one-dimensional signals with 2^{kmax} samples (or, in other words, for the space $l^2(0..2^{kmax}-1)$) the basis functions ψ_l^k can be represented as follows:

$$\psi_l^k(i)=\sqrt{2^{k-kmax}}\ \psi(2^{k-kmax}i-l),\quad l=0..2^k-1, k=0..kmax \quad (3)$$

Some notational conventions are in order here. Throughout the rest of the thesis, discrete (sampled) one- and multidimensional signals, images, and sequences of frames would imply that all the indices pertaining to them, sample labels, etc., are integers. Division operation performed in that context is assumed to yield an *integer* result; i.e., the result of the division is rounded in the direction of zero, unless otherwise specified.

If the basis wavelet $\psi(x)$ (called also a mother wavelet) differs from zero only within a relatively small compact area of its domain, functions (3) are scale and spatially localized. To demonstrate, consider the simplest possible basis wavelet, namely, a δ-function:

$$\psi(i)=\begin{cases}1, \text{if } i=0\\ 0 \text{ otherwise}\end{cases} \quad i\in Z \quad (4)$$

With this choice of the mother wavelet, basis (3) can be written as follows:

$$\psi_l^k(i) = \sqrt{2^{k-kmax}} \begin{cases} 1 & \text{if } i=2^{kmax-k}\times l,\dots,2^{kmax-k}\times(l+1)-1 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

$$l = 0..2^k - 1, \quad k = 0..kmax,$$

which is justified by the fact that when, say, k=0, a factor $i \cdot 2^{-kmax} \equiv i/2^{kmax}$ vanishes for all integer i from 0 up to 2^{kmax} -1. The following picture demonstrates graphs of some basis functions for the interval [0..7] (zero through 7); for the sake of clearness, the function values are represented as bars, yet one has to keep in mind that the functions are discrete. Chapter III gives similar pictures for two-dimensional wavelet bases.

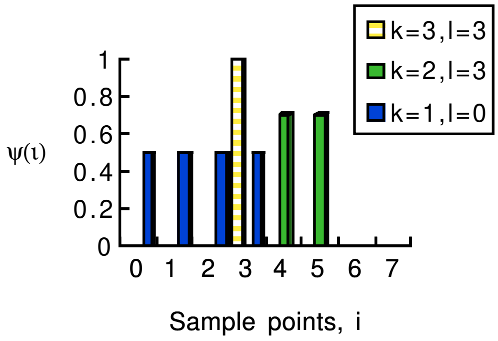


Fig. 1. Graph of sample basis functions.

The graphs clearly demonstrate that the functions are indeed localized in space. The distinction in scale between the functions is also quite obvious. Moreover, the graphs show that the finer the scale (i.e., the bigger the scale index k), the more precisely the function is localized. This is the universal property of the wavelet bases and follows from the fact that the wavelet filter bank (to be discussed later) has constant *relative* bandwidth and provides so-called constant-Q filtering with octave-band filters. Incidentally, this behavior is often regarded as an asset because the coarse-scale context of the image (or any signal for that matter) does not have to tell a precise location of features; since the features are coarse-grained themselves, approximate localization will suffice. It also implies that the coarse-scale band can be represented by only a few samples, which contributes to the compression. One has to emphasize that different bases for different scales look similar, in the sense they are built by the same rule. As far as the orthonormality is concerned, the set of functions (5) for each single value of k constitutes an orthonormal basis (yet not complete unless k=kmax).

The question remains how to perform the decomposition of a one-dimensional signal f_i into the wavelet basis (3), or, in other words, how to estimate coefficients c_l^k of the decomposition

$$f_i = \sum_{k,l} c_l^k \psi_l^k (i) \tag{6}$$

Note, that this equation is a one-dimensional version of eq. (2); therefore, we need only one index (subscript) to specify the localization of the basis function. It is easy to see that the set of functions $\{\psi_l^k\}$ for all l ***and*** k is overcomplete. It is especially evident from the basis (5), which, at k=kmax, turns to just a δ-function

$$\psi_l^{kmax}(i) = \begin{cases} 1, \text{ if } i=l \\ 0 \text{ otherwise} \end{cases} \qquad i \in Z, \quad l = 0..2^{kmax} - 1 \tag{7}$$

Obviously, any discrete one-dimensional signal with 2^{kmax} samples (or, in other words, belonging to the space $l^2(0..2^{kmax}-1)$) can be uniquely and easily decomposed in basis (7); the decomposition coefficient c_i^{kmax} is just the signal sample value f_i . The question is why we need ψ_l^k with k other than kmax. The answer is that it is the redundancy of the overall basis (3) which allows one to perform the multiresolutional decomposition. Indeed, the coefficients of the decomposition (6) can be evaluated in a great variety of ways due to the overcompleteness of the basis; some of the schemes would be consistent with the subband analysis in that the coefficients will indeed describe the contributions of the features according to their scale and position. In the following paragraphs, we will demonstrate one way how it can be done.

As was mentioned before, decomposition of the signal in the basis set $\{\psi_l^{kmax}\}$

$$f_i = \sum_l a_l^{kmax} \psi_l^{kmax}(i), \quad i,l = 0..2^{kmax} - 1 \tag{8}$$

represents the signal in its entirety. Decomposition coefficients, which we will denote a_l^{kmax} to distinguish from the coefficient of the true subband decomposition, describe contribution of all the features, starting from the finest up to the largest-scale ones. In the simplest case of basis (5), the coefficients are nothing but the samples f_i of the signal themselves. The set of functions $\{\psi_l^{kmax-1}\}$, corresponding to a twice as coarse resolution, is also an orthonormal basis and can be used as well to decompose the signal. Due to the property of orthonormality, the decomposition coefficient a_l^{kmax-1} is unambiguously and easily obtained by just evaluating the inner product of the signal with the basis functions:

$$a_l^{kmax-1} = \sum_i f_i \psi_l^{kmax-1}(i), \quad l = 0..2^{kmax-1} - 1 \tag{9}$$

Recall now that $\{\psi_l^{kmax}\}$ does constitute a basis of our space $l^2(0..2^{kmax}-1)$, and any function of this space, including ψ_l^{kmax-1}, can be decomposed in that basis:

$$\psi_l^{kmax-1}(i) = \sum_m g_{lm} \psi_m^{kmax}(i), \quad l = 0..2^{kmax-1} - 1 \tag{10}$$

where g_{lm} is just the decomposition coefficient. Inserting the above equation into eq. (9) results in

$$a_l^{kmax-1} = \sum_{i,m} f_i g_{lm} \psi_m^{kmax}(i) = \sum_m g_{lm} \sum_i f_i \psi_m^{kmax}(i)$$
$$= \sum_m g_{lm} a_m^{kmax}, \quad l = 0..2^{kmax-1} - 1, m = 0..2^{kmax} - 1 \qquad (11)$$

or, in the case of simple functions (5)

$$= \sqrt{2^{-1}}(a_{2l}^{kmax} + a_{2l+1}^{kmax}) \qquad (12)$$

This gives a signal representation a_l^{kmax-1} at the twice as coarse resolution. Indeed, the coefficient describes the contribution of features of characteristic scale 2 and larger to the signal. Note that the features can now be located up to ±1 sample point (see the graph of the simple functions on Fig. 1), and small scale features are not represented. Indeed, if the signal changes significantly from one sample point to the next (i.e., at the scale of 1), that change would be smoothed out by the 'averaging' in eq. (12). Therefore, if one is to reconstruct the signal from a_l^{kmax-1} using the composition rule

$$\hat{f}_i = \sum_l a_l^{kmax-1} \psi_l^{kmax-1}(i) \qquad (13)$$

he will not obtain the original signal precisely. The finest, smallest-scale features will be missing. A restored fine-resolution representation

$$\hat{a}_l^{kmax} = \sum_i \hat{f}_i \psi_l^{kmax}(i) = \sum_{i,m} a_m^{kmax-1} \psi_m^{kmax-1}(i) \psi_l^{kmax}(i)$$
$$= \sum_m \tilde{g}_{lm} a_m^{kmax-1} \qquad (14)$$

with

$$\tilde{g}_{lm} = \sum_i \psi_m^{kmax-1}(i) \psi_l^{kmax}(i) \qquad (15)$$

is therefore different from a_l^{kmax}. Hence, taking the difference between the representation of the signal with all the features, including the finest ones, and the representation which describes all the details but the finest-scale ones,

$$c_l^{kmax} = a_l^{kmax} - \hat{a}_l^{kmax} \qquad (16)$$

would give us the contribution of only the finest features, which is exactly what we expected to obtain.

The same procedure can be applied recursively to a_l^{kmax-1} to obtain the contribution of features of scale 2 only. It follows exactly the same steps, eqs. (9-16), with kmax changed for kmax-1. We emphasize that because all functions ψ_l^k are scaled and translated versions of the same function, the mother wavelet, eq. (3), decomposition

$$\psi_l^{k-1}(i) = \sum_m g_{lm} \psi_m^k(i), \quad l = 0..2^{k-1} - 1 \qquad (17)$$

holds for any k with exactly the same set of g_{lm}. Indeed, inserting eq. (3) results in

$$\psi(i / 2^{kmax-k+1} - l) = \sqrt{2} \sum_m g_{lm} \psi(i / 2^{kmax-k} - m)$$

or

$$\psi(i / 2 - l) = \sqrt{2} \sum_m g_{lm} \psi(i - m), \quad l = 0..2^{k-1} - 1$$

Keeping in mind eq. (2), one can infer that

$$g_{lm} = g(2l - m) \qquad (18)$$

and eq. (17) then reads

$$\psi(i / 2) = \sqrt{2} \sum_m g(-m) \psi(i - m) \qquad (19)$$

which may be considered a definition of the function g. Note that neither eqs. (18) nor (19) contains index k. It means that the function g and coefficients g_{lm} are scale-independent, and, hence, the procedure of obtaining the coarse resolution view a_l^{k-1} from the finer resolution one, a_l^k, eq. (11), remains literally the same regardless of the scale k. By the same token,

$$\begin{aligned}\tilde{g}_{lm} &= \sum_i \psi_m^{k-1}(i)\,\psi_l^k(i)\\ &= 2^{k-kmax}\sqrt{2^{-1}}\sum_i \psi(i/2^{kmax-k+1}-m)\,\psi(i/2^{kmax-k}-l)\\ &= \sqrt{2^{-1}}\sum_i \psi(i/2-m)\,\psi(i-l) \qquad (20)\end{aligned}$$

does not depend on k either and stays the same for all the scales. Moreover, it is easy to see that

$$\begin{aligned}\tilde{g}_{lm} &= \tilde{g}(2m-l),\\ \tilde{g}(m) &= \sqrt{2^{-1}}\sum_i \psi(i+m)\,\psi(i/2) \qquad (21)\end{aligned}$$

One has also to emphasize that a_l^{kmax-1} contains half as many points as a_l^{kmax} does, which is obvious from eqs. (8,11). Therefore, the whole decomposition can be represented in the form

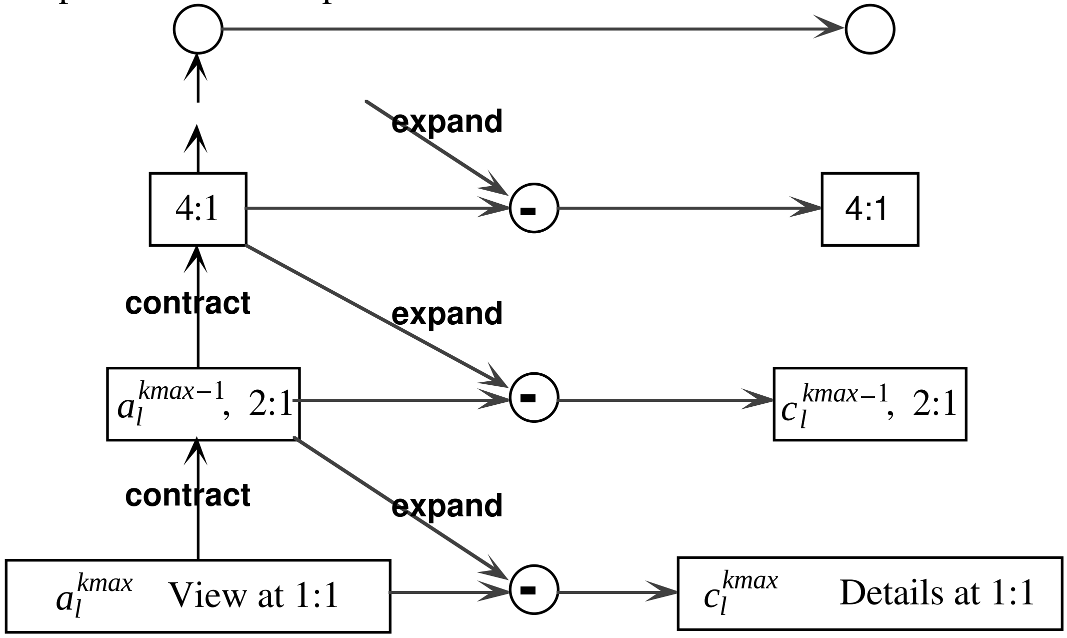


Fig. 2. Construction of the Gaussian and Laplacian pyramids.

The pyramid on the left-hand side of the figure is a Gaussian pyramid [BURT83], which represents a set of views of the signal at different resolutions. The bottom level of the pyramid is the exact representation of the original signal. Subsequent levels are constructed by successively applying a **contract** operation [UNSE92], eqs. (11)-(18),

$$a_l^{k-1} = \sum_m g(2l-m)\,a_m^k, \quad l = 0..2^{k-1}-1,\ m = 0..2^k-1 \tag{22}$$

which ‘averages out’ fine image details and performs the subsampling, thus reducing the resolution twice. The Laplacian pyramid [BURT83] is on the right-hand side of the picture. It is obtained by using an **expand** operation, eq. (14), which, keeping in mind eq. (21), can be written as

$$\hat{a}_l^k = \sum_m \tilde{g}(2m-l)\,a_m^{k-1}, \quad l = 0..2^k-1,\ m = 0..2^{k-1}-1 \tag{23}$$

The operation produces $\hat{a}_l^k$ with 2^k samples from a_l^{k-1} with 2^{k-1} samples, and, therefore, is in a sense the inverse of **contract**. Although **expand** increases the number of sample points twice, it does not increase the resolution nor restore the fine image details lost at **contract**ion. Therefore, the difference between the signal representation and the expanded contracted version of it contains the details c_l^k that are noticeable at the particular resolution, but not at the coarser resolution. The Laplacian pyramid then is a collection of signal features separated among the set of all possible resolutions, and can be considered as the most general implementation of the multiresolutional signal analysis. It should be stressed that the Laplacian pyramid is the *exact* representation of the original signal. Indeed, following the arrows on Fig. 2 in the reverse direction starting from the top converts the Laplacian pyramid back to the Gaussian one, whose bottom level represents the original signal in full detail.

It should be emphasized that the **contract** and **expand** operators, eqs. (22) and (23), are scale-independent and apply to any level of the Gaussian pyramid, because g and $\tilde{g}$ do not depend on the scale k. This, as we saw earlier, is the consequence of the fact that all wavelet bases are generated from a single function, the mother wavelet (3). It is usually considered desirable [UNSE92] for those operators to satisfy the consistency rule

$$\mathbf{contract}(a^k) \equiv \mathbf{contract}\Big[\mathbf{expand}\Big[\mathbf{contract}(a^k)\Big]\Big] \qquad (24)$$

which guarantees that the **contract** and **expand** operators are strictly complementary. It is easy to see that this is indeed the case with the orthonormal wavelet basis (3). Incidentally, if the set of $\{\psi_l^k\}$ for each k is an orthonormal basis, it can be shown that g and $\tilde{g}$ are the same.

Multiresolutional signal decomposition can be viewed from a quite different perspective. When performing recursive construction of the pyramid from the bottom to the top, Fig. 2, each step involves splitting a signal representation at hand at the current resolution a_l^k into the lower resolution signal a_l^{k-1} and the detail signal c_l^k. The **contract** operation that produces a_l^{k-1}, eq. (22), has exactly the form of the low-pass filtering of a_l^k followed by the subsampling by two (keeping every other sample). In the simple case of basis (5) this operation is just the averaging, eq. (12). It clearly illustrates the fact that the filter g is indeed a lowpass filter. Since the frequency (resolution) band of the signal is reduced during the filtration, the smoothed signal can be subsampled without any loss of information, according to the sampling theory. Operation **expand** is designed to undo the subsampling; indeed, formula (23) means inserting a zero between every two samples of a_l^{k-1} followed by a low-pass filtering with a companion kernel $\tilde{g}$ which performs interpolation. The consistency condition, eq. (24), guarantees that the result $\hat{a}_l^k$ is indeed the low-frequency component of a_l^k. Obviously, the difference c_l^k, eq. (16), between the original and the low-resolution component is the high-frequency component. Since c_l^k is obtained eventually from the samples of a_l^k after several linear operations (convolutions), eqs. (22), (23), and (16), this fact can be written in a general form as

$$c_l^k = \sum_m h(l-m)\, a_m^k, \quad l = 0..2^k - 1,\ m = 0..2^k - 1 \tag{25}$$

where $h(i)$ is obviously a high-pass filter, which is related to g and $\tilde{g}$ via eqs. (22), (23), and (16). The lowpassed band a_l^{k-1} is treated in the similar way on the next step of the procedure, and split into the even lower resolution, and the detail components. Thus, the multiresolutional decomposition is equivalent to the separation of the signal into (octave) resolution bands by applying a bank of iterated filters g and h. This justifies the name of the subband analysis. The original signal can be reconstructed from the decomposition using the expand operation; in other words, by applying the iterated filter bank $\tilde{g}$ to undo the subsampling, and merging low- and highpass bands.

In a particular case of so-called tight wavelets (which includes all orthonormal wavelets, eq. (3)), the highpassed component produced by the filtration, eq. (25), can be subsampled, too! The detail signal is then obtained as

$$c_l^k = \sum_m h(2l-m)\,a_m^k, \quad l = 0..2^{k-1}-1,\ m = 0..2^k-1 \tag{26}$$

(compare with eq. (25)). The samples discarded on decimation can be restored by performing the interpolation with a companion filter $\tilde{h}$

$$\hat{c}_l^k = \sum_m \tilde{h}(2m-l)\,c_m^k, \quad l = 0..2^k-1,\ m = 0..2^{k-1}-1 \tag{27}$$

much in the same way it was done for the lowpass band, eq. (23). In the case of the simplest basis (5),

$$g(m) = \tilde{g}(m) = \frac{1}{\sqrt{2}}\begin{cases} 1, & m=0 \\ 1, & m=-1 \\ 0, & otherwise \end{cases}, \quad h(m) = \tilde{h}(m) = \frac{1}{2}\begin{cases} -1, & m=0 \\ 1, & m=-1 \\ 0, & otherwise \end{cases} \tag{28}$$

that is, g and $\tilde{g}$ are the ideal low-pass filters and h and $\tilde{h}$ are the ideal high-pass filters. It is obvious then that the sum of two upsampled bands $\hat{a}_l^k$ and $\hat{c}_l^k$ is indeed identical to the signal a_l^k from which they have been extracted. When the filters are non-ideal, a part of the high-resolution component leaks into the lowpass band and becomes distorted upon subsampling, which, therefore, cannot be perfectly undone. It poses no danger if the detailed signal is computed according to eq. (16), because any inaccuracy in $\hat{a}_l^k$ is taken over by c_l^k. However, subsampling the detail signal leaves those errors generally uncompensated. It is quite remarkable that with the tight wavelet basis, the reconstruction errors (aliasing) in the high- and lowpass bands, $\hat{a}_l^k$ and $\hat{c}_l^k$, exactly compensate for each other! Filters with the perfect reconstruction property have been studied in the signal processing theory for some time — the survey [SIMO91] gives general treatment and general rules of design (as well as coefficients for some filters).

Thus, the tight wavelet bases of which ideal, eq. (5), orthonormal wavelets (3), biorthogonal wavelets (where functions g and $\tilde{g}$ are no longer the same), quadrature mirror filters are the particular cases — all allow us to perform a subband decomposition and perfect reconstruction of a one-dimensional set of data with *subsampled highpassed bands*. From this we can see that the total number of transform coefficients $\{c_l^k\}$ in that case is exactly 2^{kmax}, the total number of the original signal samples. In a more general case of so-called loose wavelets, cancellation of alias errors on merging reconstructed low- and highpass bands is impossible, and the detail signal has to be left unsubsampled, as on Fig. 2. The original signal is still perfectly reconstructable, as was mentioned before while discussing the Laplacian pyramid. However, the total number of transform coefficients exceeds the number of samples, almost twice in the one-dimensional case and by 33% for a two-dimensional signal. Although it appears as an expansion at first glance, in reality the redundancy of the transform pays off. The amplitude of the coefficients may turn out to be smaller, and this allows for more efficient compression. The question will be considered in detail in Chapter III.

Thus, there is a great variety of different wavelet bases and particular schemes for performing the decomposition from which to choose. Indeed, due to the overall redundancy of the basis $\{\psi_l^k\}$, there are a number of ways of carrying out subband analysis and selecting the filter coefficients (providing the consistency rules such as eq. (24) are satisfied). This makes the wavelet transform flexible enough to meet the requirements of a very broad range of applications. It is rather fast, too: due to the successive subsampling, the total complexity is $\Theta(3NM)$, where N is the total number of samples and M is the size of the filter window. One has to mention that the latter quantity, which is nothing but the number of nonzero coefficients of filters g, $\tilde{g}$, h, etc., is usually significantly less than $2 \cdot 2^{kmax}$ implied by eqs. (22-23). Indeed, in the simplest ideal basis, eq. (5), where the mother wavelet is a δ-function, the filter kernels differ from zero at only two points, eq. (28). It was shown that only the ideal δ-function wavelet basis has such short filters. Note a well-known Haar basis with the mother wavelet,

$$\psi(i) = \tfrac{1}{2}\begin{cases} 1, & i=0 \\ -1, & i=1 \\ 0, & otherwise \end{cases} \tag{29}$$

is a variation of the δ-function basis. One has to point out that in both bases, the impulse response of the filters is too clear-cut with abrupt transitions from zero to non-zero levels. If smoother filters are desired, one can use a biorthonormal Daubechies basis with 2M coefficients, which collects the signal energy into the low band to the extent that the original signal is represented by polynomials of degree M-1 or less. Daubechies six coefficient wavelets are quite efficient for compressing natural scene images. The review article by Rioul and Vetterli [RIOU91] and the lecture notes by Daubechies [DAUB92] provide good introductions to these and other wavelet types that have been developed for signal analysis and data compression.

The multiresolutional analysis is easily extended to higher dimensions by forming a Cartesian product of one-dimensional transforms. One way of doing this is first applying a one-dimensional filter bank to separate image rows. The result is a ‘lowpassed image’ and a ‘highpassed’ image, each with half the number of columns of the original image (assuming the wavelet is tight and the highpass band can be subsampled). The same filter bank is applied again, this time to the columns of these images. Four images, each one-fourth the size of the original image result: one lowpassed image, and three highpassed images that can be characterized by the column and row filter as high-lowpassed, low-highpassed and high-highpassed images. One can continue the decomposition of the lowpassed image to form a two-dimensional multiresolution transform of the original image. Chapter III expounds upon a slightly different approach that uses a two-dimensional filtering of the original image to get the lowpass band and the detail image.

Multiresolutional decomposition of the signal by itself provides for efficient compression. It was shown [PEAR91] that the subband coding is asymptotically optimal from the information/ communication theory point of view (in a rate-distortion sense) for the Gaussian source and variable width subbands: “Decimated subband processes on the average have less ‘memory’, or, for Gaussian processes, have flatter spectra, than the full-band source. The coding gain has been related to that entropy reduction” [PEAR91]. In other words, since a decomposition coefficient c_l^k describes a feature which is located within a predefined limited range of signal samples and can be discerned only within a restricted range of resolutions, and since most of the images are far from the white noise (which has no correlations in local intensities nor ‘features’), the coefficient c_l^k has a small magnitude, and requires fewer bits to encode. Moreover, the transform as a whole contains a number of repetitive patterns, which contributes to efficient compression. An entropy encoding of the coefficients as they are, gives a lossless compression, as the original signal can be restored identically. Additional (and far more significant) compression gains can be achieved with the appropriate quantizing of the coefficients. A paper [DEVO92] shows that if images can be characterized by their membership in some quite broad smoothness classes, then the wavelet-based methods are near optimal within a larger class of stable (in a particular mathematical sense) transform-based methods of image compression. Moreover, with the multiresolutional speech and image decomposition, it is possible to take advantage of the fact that the human visual and aural systems have different sensitivities for different frequency/resolution bands of features. Therefore, it is possible to adapt the quantization noise to the human sensitivity along each resolution band. This enables one to introduce a minimum amount of *perceivable* distortion in the reconstructed signal.

Thus, in summary, the wavelet decomposition offers a unique opportunity to look at the image at different scales. Data compression using wavelets is a form of the transform-based coding, where a reversible linear transformation is applied to the input data signal. The transformation decomposes the signal into a weighted sum of basis functions. These weights, or coefficients of the basis functions, are an alternate description of the original signal. If the data in this coefficient space are less correlated that the data in the original signal space, lossless or near lossless compression of the signal can result after entropy coding the weights. In addition, the transform domain may be a better space in which to quantize data for lossy compression. The transform space may have some physical interpretation or other properties that help us tailor the distortions to match some model or to otherwise optimize the reconstructed data for subsequent use. For example, we quantize natural scene images to match the human visual system as closely as possible.

2.2 Fractal Image Compression

A classical (and so far the only viable) scheme for fractal image compression is conceptually based upon simple fractal generating algorithms exploring a property of self-similarity. Thus, the basis for the scheme is completely different from traditional compression methods — an image is partitioned into parts that can be approximated by other parts of the same image after some scaling and/or rotation operations. The result of an encoding process is a set of transformations, which, when iterated upon any initial image, converge to a fixed point that is an approximation of the original image. The idea of looking for fractal algorithms to approximate a given natural image was first proposed by M.F. Barnsley. Since simple fractal algorithms can typically generate very complex images, he suggested that only relevant parameters of the algorithm need be stored, resulting in significantly reduced memory requirements for the storage of images.

Iterated function systems fractals and fractal transform fractals that constitute the basis of the fractal image compression scheme are explained in great detail in a recently published book by Barnsley and Hurd [BARN93]. One of the best introductions into the subject is a review [FISH92], which we will briefly follow. As was already mentioned, the fractal compression is based on a self-similarity; i.e., a similarity between a whole and a part of it. In Chapters IV and V we will study a number of examples of that property. To begin with, one has to define the exact meaning of being similar. In the context of fractal image compression, one small part of an image is said to be similar to another if there is an affine transformation that makes the parts coincide (or almost coincide). The affine transformation on the plane is just a transformation of the following general form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathrm{A}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{30}$$

which includes rotation, scaling, and translation. If one finds a transformation (or set of transformations) that maps an entire image into some of its corresponding blocks, and those transformations are eventually contractive [FISH92], then, according to a generalized collage theorem proved by M.F. Barnsley, applying the transforms recursively starting from an arbitrary image (i) does converge to a single image, and (ii) that picture is the original image, or a very close approximation of it. Thus, a transformation (or, precisely, a set of parameters in eq. (30) that define it) can be regarded as an encoding of an image. The following popular example of a Sierpinski gasket (triangle), borrowed from [FISH92], gives an idea how a picture can be represented by transformations.

The following figure depicts the original gasket within the coordinate frame,

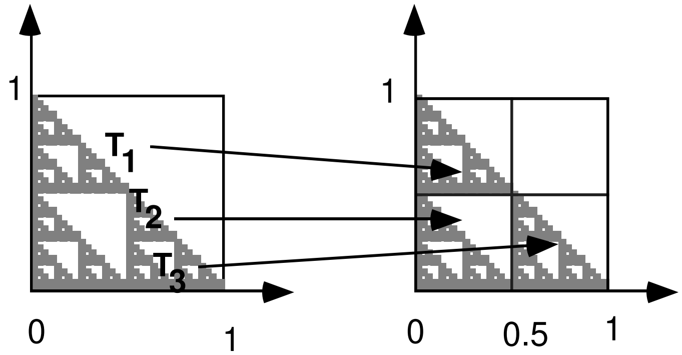


Fig. 3. Self-similarity of the Sierpinski gasket.

which demonstrates that the entire gasket is similar to the three parts (quadrants) of itself, with the transformations

$$\begin{aligned}
\boldsymbol{T}_1:\quad \begin{bmatrix} x' \\ y' \end{bmatrix} &= \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{2} \end{bmatrix} \\
\boldsymbol{T}_2:\quad \begin{bmatrix} x' \\ y' \end{bmatrix} &= \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad (31) \\
\boldsymbol{T}_3:\quad \begin{bmatrix} x' \\ y' \end{bmatrix} &= \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \frac{1}{2} \\ 0 \end{bmatrix}
\end{aligned}$$

which means squeezing the image twice and translating along coordinate axes. The following pictures show how the triangle is reconstructed iterating the transforms upon a starting image:

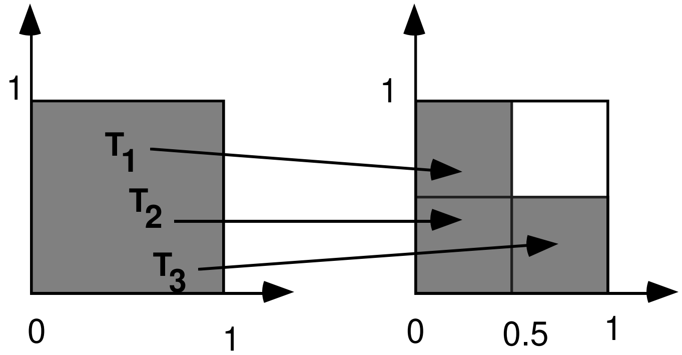


Fig. 4. Application of transforms (31) to a starting image.

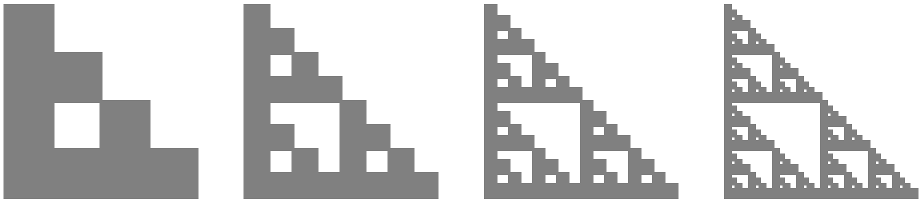


Fig. 5. Subsequent steps of iterating transforms (31).

Since the Sierpinski triangle is reconstructed precisely from the transform as Figs. 4 and 5 show, the coefficients of eq. (31) are an exact representation of the gasket, which can be encoded with only a few bits. Note that even if the starting image is different (say, with half gray and half white), the result will be the same; indeed, after a certain number of steps the starting image is contracted to a single pixel, of which it turns out the gasket is made. Since the pixels have no smaller parts, the details of the starting image are not important, either. Note that to get a gasket twice as big, one need only to apply the same algorithm to a bigger starting picture. This fact is indeed remarkable: since the transform possesses no information about the absolute size or dimension of the picture, the same set of transforms can be used to generate images of arbitrary size without any loss of resolution.

Obviously one cannot expect that every image is precisely similar to its quadrants, as the Sierpinski triangle is. However, a suitable partitioning of the image may be used to split it into relatively small patches (domains), which have much simpler structure than the whole. Therefore, a small domain can be more precisely 'covered', or found similar to, another (generally smaller) patch of the same picture (so-called range). The following figure shows a regular splitting of a sample picture into domain blocks, which is often used in practice [FISH92]:

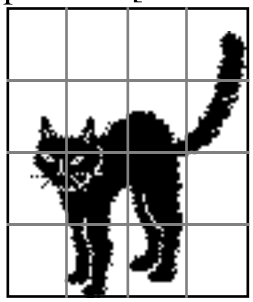


Fig. 6. Splitting a sample picture into domain blocks.

The common choice for domain blocks is a 16x16 square, and the minimum size of the range blocks is 8x8. The following pseudocode [FISH92] sketches the fractal image compression algorithm:

```
• Set $R_1$ = $entire\_image$ and mark it uncovered
• While there are uncovered ranges $R_i$
    — out of the possible domains, find the domain $D_i$ and the
      corresponding transformation $T_i$ that best covers $R_i$
    — if the cover is good enough or $R_i$ is small enough
          mark $R_i$ as covered and write out the transformation $T_i$
    — else
          partition $R_i$ into smaller ranges, mark them as uncovered, and
          remove $R_i$ from the list of uncovered ranges
```

The pseudocode gives only a general framework of the algorithm. A particular implementation must choose a specific regular or irregular splitting of the image into domain blocks, a way of obtaining the best possible match, and a criterion for a ‘good enough’ match, etc. One has to note that the algorithmic complexity of the algorithm is $\mathrm{O}(N^4)$, where N is the dimension of the image. Indeed, if the image size increases twice, the number of both range and domain blocks increases 4 times, which would make the search for a match between a domain and a range block take 16 times longer.

Since its discovery in 1988, the fractal compression of images has been in vogue. Its merits are well known, partly because of the products produced by Iterated Systems, Inc., founded and chaired by M.F. Barnsley and A.D. Sloan. Compression ratio varies with the image content, but typical images of people and nature can be compressed up to 50-100 times without significant deterioration of visual quality. The algorithm is lossy. Since the fractal compression describes the relationship between parts of the image rather than millions of individual points in the image, it opens up all sorts of possibilities for computer vision, image understanding, soft-copy photographic keys, and aids in image interpretation. Some flaws in the algorithm are also well known: it takes an enormous amount of time to find an appropriate fractal transformation (though the decompression is fast). Software compression takes up to three hours for a $512\times512\times8$ image on IBM PC 486 30 MHz. Hardware compression requires a proprietary integrated circuit board that is currently available only for the IBM PC. The algorithm itself is proprietary. At high compression ratios, the decompressed image exhibits a tile effect as if built from blocks not perfectly adjusted to one another. It is especially evident when not enough time is allocated for the compression algorithm to work. Spurious features and stray marks might also appear.

It is rather straightforward to see that the fractal image compression can be complemented (and benefit from) the multiresolutional analysis. There are several directions for investigation. One may look for similarities between different parts of the image at some reduced resolution, or one may be concerned with the similarity between representation of the whole image or parts of it at two (or several) resolutions. A combined approach is also possible. The merit of the former is that the image at a reduced resolution might still contain all the information which is ‘interesting’ or important, but as the size of the image significantly decreases, the fractal transformation is much easier to find. A single level of the multiresolutional pyramid contains only small corrections and details that are discernible just at that particular resolution, making the level easier to handle than the original image. The combined multiresolutional/fractal compression makes it straightforward to select features/areas that are important and control the fractal transform to approximate them to a better extent. Finally, different levels of the multiresolutional pyramid may exhibit some degree of similarity. This fact was mentioned previously by Mallat [MALL89a]. He pointed out that this topic is promising because multiscale decompositions, such as wavelet transform, are well adapted to evaluate the self-similarity of a signal and its fractal properties.

2.3 Video Compression Environment

In the 1980's, several standards committees worked to establish uniform approaches to video compression. These groups were the Joint Photographic Experts Group (JPEG), the Motion Picture Experts Group (MPEG), the International Standards Organization (ISO), and the Consultative Committee for Telephone and Telegraph (CCITT). So far, three popular video compression schemes have emerged. A paper [LEE92] and talk [BAKE93] provide an excellent review of the subject.

The H.261 standard — commonly called Px64, and optimized to achieve very high compression ratios for full-color, real-time motion video transmission — was finalized in 1989 by CCITT. The JPEG standard, optimized for full-color images, was published in 1991. The MPEG standard, aimed at full-color, full-motion video, is still under development, though in the final stage.

The CCITT Px64 compression algorithm combines intraframe and interframe coding to provide fast processing for on-the-fly video compression and decompression applications. The algorithm begins by coding an intraframe block using the Discrete Cosine Transform (DCT), quantization and entropy coding, in much the same way JPEG does.

As the coded block is converted to bit stream output, however, it is also decompressed using a reverse process and then stored in an internal buffer memory. Each subsequent frame is then coded, in terms of its predecessor, using predictive interframe coding.

In the predictive interframe coding process, a pixel block from the current image frame is loaded. Then a motion estimator executes a series of search and comparison operations with respect to the previous image stored in the past reference memory. This search identifies a matched block from the previous frame and the motion vector which associates the past block with the current block. A differentiator takes the two matched blocks as its input and outputs a new pixel block representing the difference between the two blocks. Finally, the difference block is coded using the DCT, quantization and entropy coding. The block is output as coded bits, along with an encoding of the associated motion vector that is needed to reconstruct the original block upon decompression. That block is also decompressed internally and stored as the new reference for the next frame block.

The CCITT Px64 standard is optimized for applications such as video based telecommunications. Because these applications are usually not motion-intensive, the algorithm uses limited motion search and estimation strategies to achieve higher compression ratios. However, achieved compression is not high enough for a full-scale video transmission, and even for videoconferencing [BAKE93]. A videoconferencing system developed by PictureTel which uses a proprietary algorithm based on the Laplacian pyramid provides twice as high video throughput [BAKE93].

The JPEG coding utilizes DCT, quantization and entropy coding to achieve intra-frame coding. Under JPEG compression, a YUV (chrominance and luminosity) pixel block is DCT transformed into a frequency matrix value. Then a quantization operation yields a matrix of compressed frequency values, which is entropy coded to produce the resultant compressed bit-stream. The coded bits can be stored or transmitted digitally and then decompressed through a reverse process to regenerate the pixel image. Originally targeted for the full-color, still-frame applications (compression rates average 15:1), JPEG is also used for some real-time, full-motion video applications.

MPEG provides all of the basic intraframe compression functions and combines predictive interframe and interpolative interframe coding for motion compensation. The MPEG algorithm first compresses an initial intraframe block using DCT, quantization and entropy coding. The same coded block is also decompressed and stored in the internal past memory buffer. The algorithm then uses predictive interframe coding similar to that used by the Px64 to code nonadjacent future frames. Again, the future frame block is internally decompressed and stored in the internal future memory buffer.

Interpolated interframe coding is used to estimate and code frames between subsequent intraframes or interframes. The interpolated interframe coding used in MPEG is very similar to the predictive interframe coding used by Px64, except that the motion estimation involves a comparison and search with respect to both images in the past and future reference frame buffers. The results of the motion estimation are an average of the matched blocks found in the reference buffers and a motion vector associated with each matched block found. Differentiation produces a new pixel block representing the difference between the current block and the average of the past and future blocks. Then DCT, quantization and entropy coding are applied to the difference block, and the eventual coded bit-stream thus produced is stored and transmitted, along with the associated motion vectors needed to reconstruct the original pixel block. After coding all interpolated pixel frame blocks, the image data in the future memory buffer is moved to the past memory buffer, and a new future-basis frame is coded using either intraframe or predictive interframe coding. The MPEG standard supports deep-motion search, and is optimized for motion intensive video applications such as found in CD's. The MPEG approach achieves compression rates of 200:1, but requires higher transmission rates. One has to stress that the backward frame prediction used in MPEG requires one to know a future frame before the current one. It is easy to achieve while playing back movies recorded on the CD-ROM where a random access to information blocks is possible. Incidentally, CD-ROM movie playback was the primary objective of the MPEG standard. Using MPEG for the on-the-fly video compression/decompression is very cumbersome and demands immense computing and memory capacities [BAKE93]. Moreover, a required bit rate of 1.5 MBits/sec is far too high for regular telephone communication channels.

QuickTime [QUIC92] is a development environment for video image processing. It was originally released for the Macintosh, but it is now available in PC-Windows. We have chosen to adopt QuickTime as a host environment for processing video signals, primarily because it has an open architecture and provides a means to use a multiplicity of compression schemes which can be easily incorporated into the QuickTime framework.

QuickTime provides several features that make it an ideal development environment. These are:

- Open architecture for addition of new media types, compression routines, and sound processing;
- Standard support of custom compression boards or digitizer boards;
- Human interface capabilities to allow considerable user input to process and control both media input and output;
- Several standard routines for media processing and control;
- Simple interface into the AppleTalk communication system for transmission of media between computer systems;
- Storage and preview components to allow replay or review of media prior to processing;
- Media grabber interface to allow easy selection of compression, source, and sample settings for video and sound capture channels;
- Media conversion routines for all standard formats.

Using QuickTime has allowed the project to continue with little time required for developing standard routines which duplicate the capabilities of QuickTime but are necessary for the development of a system capable of video and voice communication.

CHAPTER III

MULTIRESOLUTIONAL IMAGE ANALYSIS

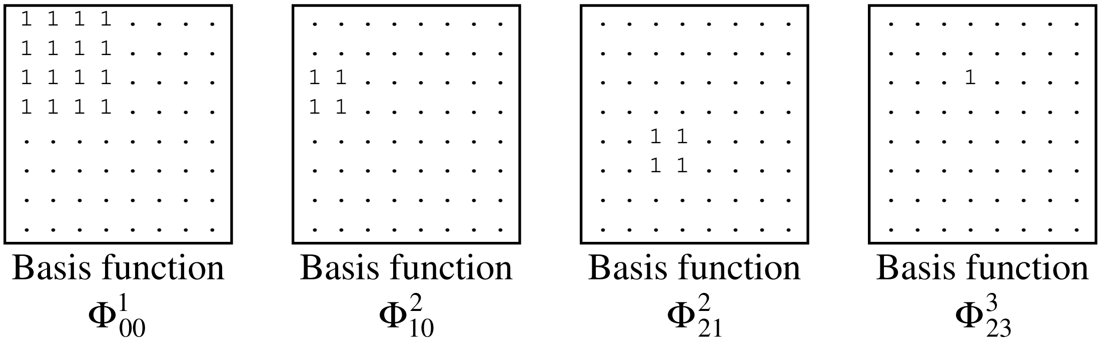
3.1 Laplacian Pyramid Image Decompositions: Overview of the Algorithm

In the present section, we will be concerned with a multiresolutional analysis of a two-dimensional signal:

$$image_{ij} = \sum_{k,l,m} c^k_{lm} \Phi^k_{lm}(i,j) \qquad (1)$$

where $image_{ij}$ is the value of a pixel at the crossing of the i-th row and the j-th column of the image, and $\Phi^k_{lm}(i,j)$ is the value of a basis function Φ^k_{lm} at this position. Recall that index k tells the 'scale' of the function, and the *l, m* indices specify the localization of the function Φ^k_{lm} according to that scale. Note, since the image details can be characterized by only a few samples at a coarse resolution, the range for l and m gets smaller as k decreases (generally by a factor of 2).

The simplest possible decomposition basis is a generalization of the δ-function basis (2.5) for two dimensions. The basis was shown to provide for the optimal compression of images in a sense of the rate-distortion function [DEVO92], and was extensively used in the present research [KIFI92 and KIFI93]. Specifically, Φ^k_{lm} are indicator functions of a square of the size N /2$^{\text{k}}$ with the upper left corner at $\left(l \times N/2^k\ ,\ m \times N/2^k\right)$, where N is the image dimension (which is assumed to be a power of two). In other words, the basis function equals one within the square and zero elsewhere. The following figure depicts several sample basis functions with different scales and various localizations for a 8x8 image. To avoid the clutter, zeros are displayed as dots.



```
1 1 1 1 . . . .
1 1 1 1 . . . .
1 1 1 1 . . . .
1 1 1 1 . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
```

Basis function Φ_{00}^{1}

```
. . . . . . . .
. . . . . . . .
1 1 . . . . . .
1 1 . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
```

Basis function Φ_{10}^{2}

```
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . 1 1 . . . .
. . 1 1 . . . .
. . . . . . . .
. . . . . . . .
```

Basis function Φ_{21}^{2}

```
. . . . . . . .
. . . . . . . .
. . . 1 . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
```

Basis function Φ_{23}^{3}

Fig. 7. Example of functions Φ_{lm}^{k} for the simplest basis.

A coefficient c_{lm}^{k} of decomposition (1) tells the contribution of the corresponding square to the image. Clearly, the basis functions with the same k are orthogonal, because the squares do not overlap. Moreover, the two-dimensional basis $\Phi_{lm}^{k}(i,j)$ is a Cartesian product of one-dimensional bases $\psi_{l}^{k}(i)$, eq. (2.5):

$$\Phi_{lm}^{k}(i,j) = \psi_{l}^{k}(i)\psi_{m}^{k}(j) \qquad (2)$$

Multiresolutional decomposition of an image follows the general scheme, Fig. 2, which is redrawn below for easy reference (and slightly adjusted for the two-dimensional case):

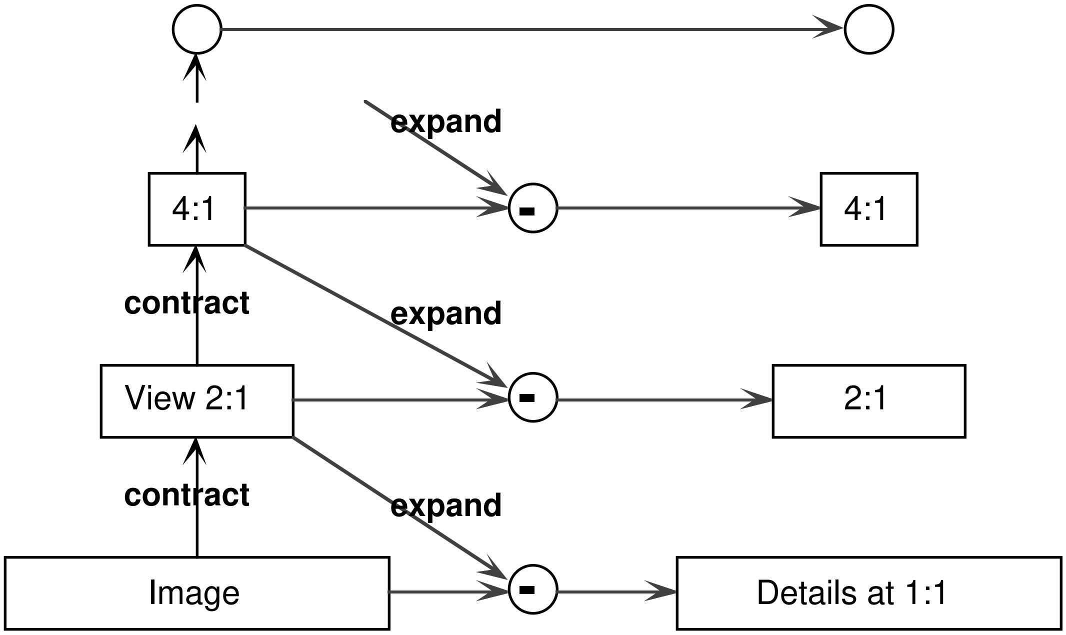


Fig. 8. Multiresolutional analysis of an image.

Recall that the Gaussian pyramid (which is on the left-hand side of the Fig. 8) represents a set of views of the image at different resolutions. It is constructed by successively applying a **contract** operation, which reduces the image resolution (and, therefore, the size) twice in each dimension. The Laplacian pyramid, on the right-hand side of the picture, is obtained by using an **expand** operation. It reverses the subsampling of the **contract** operator and increases the span of the picture twice in each dimension, but it does not increase the resolution. Image details filtered out when reducing the resolution are lost and cannot be restored from the contracted version. Therefore, the difference between the image and the expanded contracted version of it contains the details that are visible at the particular resolution, but not at the coarser resolution. The Laplacian pyramid then is a collection of image features separated among the set of all possible resolutions.

As follows from the discussion in Chapter II, **contract** operation involves filtering with an appropriate lowpass filter followed by subsampling, eq. (22). Operator **expand** reverses the subsampling by performing the intersample interpolation, which is carried out as upsampling followed by another lowpass filtration, eq. (23). It should be stressed that due to the overcompleteness of the overall basis, an exact form of the **contract** and **expand** operations can be selected in a variety of ways. One particular form of the **contract** operator that was used with the δ -function basis for the one-dimensional signal is averaging two adjacent samples, eq. (12). The same idea also works for images; namely, the **contract** operation is selected to literally take an average of the corresponding 2x2 square,

$$\begin{aligned} &a_{ij}^{kmax} = image_{ij}, \quad kmax = \log N, \quad i,j = 0..N-1 \\ &a_{ij}^{k-1} = average\left(a_{2i,2j}^{k} + a_{2i+1,2j}^{k} + a_{2i,2j+1}^{k} + a_{2i+1,2j+1}^{k}\right), \\ &\qquad k = kmax..1, \quad i,j = 0..2^{k-1}-1 \end{aligned} \tag{3}$$

where the *average* is the mean arithmetical average rounded to the closest integer. In the one-dimensional case, the **expand** operation was just duplicating every sample with appropriate normalization. By the same token, **expand**ing an image is to be performed by replacing each pixel by a 2x2-block of the same intensity. Thus, the expansion and subtraction operations (see Fig. 8) necessary to obtain nodes c_{ij}^{k} of the Laplacian pyramid can be written in the following simple form:

$$c_{ij}^{k} = a_{ij}^{k} - a_{i/2,j/2}^{k-1}, \quad k = kmax..1, \; i,j = 0..2^{k}-1 \tag{4}$$

Note that the way the pyramid is constructed satisfies the consistency condition, in that the **contract** and **expand** operators are strictly complementary; i.e.,

$$\mathbf{contract}(image) \equiv \mathbf{contract}\left[\mathbf{expand}\left[\mathbf{contract}(image)\right]\right] \qquad (5)$$

compared to eq. (2.24). This can easily be verified by considering a sample image with one quadrant blackened.

We emphasize once again that the decomposition $\{c^k_{lm}\}$ is an exact representation of the original image regardless of the way the **contract** and **expand** operations are performed. Indeed, reversing arrows on Fig. 8 restores the Gaussian pyramid from the Laplacian one, the bottom level of which is the original image itself (or its exact transform). For example, the averaging in eq. (3) can be computed as either the mean arithmetical value, or median value, etc. of the four integers. One can take advantage of this arbitrariness in designing the pyramid with the least entropy. This question will be discussed in more detail in the next section.

The generic algorithm of the multiresolutional image analysis (construction of the Laplacian pyramid) can now be outlined as follows:

```
Step 1
        assign $c^{kmax}_{ij} := image_{ij}$,  for $i,j = 0..2^{kmax}-1$
Step 2
        for k=kmax-1 downto 0 do
          $c^k_{ij}$ := "average" of  $c^{k+1}_{2i,2j}$, $c^{k+1}_{2i+1,2j}$, $c^{k+1}_{2i,2j+1}$, and $c^{k+1}_{2i+1,2j+1}$;
             $i,j = 0..2^k-1$
Step 3
        for k=1 to kmax
          $c^k_{ij}$ := $c^k_{ij} - c^{k-1}_{i/2,j/2}$
Step 4 (optional)
        for k=1 to kmax
        apply linear or nonlinear transformation to the set
          $c^k_{2i,2j}$, $c^k_{2i+1,2j}$, $c^k_{2i,2j+1}$, and $c^k_{2i+1,2j+1}$
Step 5 (optional)
        quantization
Step 6
        Arithmetic encoding of the set of resulting
          coefficients $c^k_{ij}$
```

Step 2 of the algorithm builds the Gaussian pyramid [BURT83], Step 3 turns it into the Laplacian pyramid. The coefficients $\{c^k_{lm}\}$ of the decompositions are arranged into a quadtree. This quadtree is represented in the program as a one-dimensional array of nodes in the breadth-first traversal of the tree with a certain canonical ordering of nodes within one level. Note that the image was assumed to be a square of size $N = 2^{kmax}$.

Step 4 of the algorithm performs a renormalization of the decomposition coefficients, for example:

$$
\begin{aligned}
&c^k_{2i,2j} := c^k_{2i,2j} + c^k_{2i+1,2j} + c^k_{2i,2j+1} + c^k_{2i+1,2j+1},\\
&c^k_{2i+1,2j} := c^k_{2i+1,2j} + c^k_{2i,2j}, \quad c^k_{2i,2j+1} := c^k_{2i,2j+1} + c^k_{2i,2j},\\
&c^k_{2i+1,2j+1} := c^k_{2i+1,2j+1} + c^k_{2i,2j}
\end{aligned}
\tag{6}
$$

This is equivalent to a transition from the decomposition basis illustrated on Fig. 7 to slightly different basis functions sketched on the following pictures:

```
1 1 1 1 . . . .    1 1 . . . . . .    1 1 1 1 . . . .    1 1 . . . . . .
1 1 1 1 . . . .    1 1 . . . . . .    1 1 1 1 . . . .    1 1 . . . . . .
1 1 1 1 . . . .    1 1 . . . . . .    . . . . . . . .    . . 1 1 . . . .
1 1 1 1 . . . .    1 1 . . . . . .    . . . . . . . .    . . 1 1 . . . .
. . . . . . . .    . . . . . . . .    . . . . . . . .    . . . . . . . .
. . . . . . . .    . . . . . . . .    . . . . . . . .    . . . . . . . .
. . . . . . . .    . . . . . . . .    . . . . . . . .    . . . . . . . .
. . . . . . . .    . . . . . . . .    . . . . . . . .    . . . . . . . .
```

Basis function Φ^2_{00}	Basis function Φ^2_{10}	Basis function Φ^2_{01}	Basis function Φ^2_{11}

Fig. 9. Examples of functions Φ^k_{lm} for the renormalized basis.

As one can notice, basis functions bear orientation now; i.e., they separate image features not only according to their scale and localization, but also their orientation, which is a desired property (see Chapter II). What is more important is that this basis provides a noticeably better compression. It should be mentioned that the basis is non-orthogonal, as easily seen on Fig. 9. Hence, the renormalization can be viewed as a simple way of decomposing the signal in a non-orthogonal wavelet basis. The procedure may also be regarded as an implementation of the generic rule of computing the detail signal c^k_{ij} as a high-pass filtration of the corresponding level of the Gaussian pyramid, see eq. (2.25). However, if the average in eq. (3) is meant to be a *rounded* mean arithmetical value, a non-linear operation, the convolution in eq. (2.25) with the high-pass filter kernel should be understood rather figuratively. Note that in general, the highpass band c^k_{ij} cannot be downsampled; i.e., the wavelet basis is loose. We will return to that question in the next section.

Step 5 of the algorithm performs trimming or quantization of the Laplacian pyramid. It means the c^k_{ij} are quantized (i.e., rounded-off to be an even multiple of the quantizing factor) or simply chopped off if ‘small’. The goal is to reduce the number of distinct c^k_{ij} values and/or make some of them zero. Both operations lower the entropy of the decomposition and thus contribute to efficient compression. Note that this is the only operation that introduces a non-recoverable loss. It is because of the quantization that the image restored after the decomposition differs from the original one.

Trimming of the Laplacian pyramid is based on the observation that some terms of the decomposition, eq. (1), can be dropped without significant impact to the reconstructed image. Indeed, according to Fig. 7, a decomposition coefficient c_{ij}^{k} tells how one square of the image stands out against the larger embracing square. It implies that the quantity

$$\left\| c_{lm}^{k} \Phi_{lm}^{k} \right\| \equiv \left| c_{lm}^{k} \right| \cdot \left\| \Phi_{lm}^{k} \right\| \equiv \left| c_{lm}^{k} \right| 2^{kmax-k} \tag{7}$$

represents the product of the level of "contrast" by the level of "detail". Thus, having specified a certain threshold τ, we keep only those terms in the decomposition whose norm (7) is above the threshold. This produces the image with large scale (large grain) and/or large ‘outstanding’ features preserved and with fine grain and small contrast features eliminated. Note that the last identity in eq. (7) assumes a space $\boldsymbol{L}_1$ for the evaluation of the norm of basis functions. Trimming the Laplacian pyramid also guarantees that the overall image reconstruction error (per pixel) in $\boldsymbol{L}_1$ norm is limited by $\frac{4}{3}\tau$ [DEVO92]. It is argued that using this norm in evaluating the image distortion goes along with the human visual perception [DEVO92]. Our experiments seem to confirm that.

Quantizing is a milder form of trimming. It means that the c_{ij}^{k} are rounded to be an even multiple of some quantization criterion rather than being either kept or discarded. Again, we want to make sure that the resulting inaccuracy in the coefficient multiplied by the norm of the corresponding basis function stays less than a predefined threshold τ. This requirement defines the quantization criterion θ^{k} for c_{ij}^{k}:

$$\theta^k = floor(\tau / 2^{kmax-k}) + 1 \qquad (8)$$

It should be stressed that the quantization according to (8) is consistent with the trimming, and guarantees an upper bound of $\frac{4}{3}\tau$ of the reconstructed image error. Moreover, quantization noticeably reduces the number of distinct coefficient values, which gives a significantly better compression. Our experience attests to that.

3.2 Entropy Encoding of the Decomposition Coefficients

An arithmetic coding of the stream of c_{ij}^k assures an (asymptotically) entropy encoding; i.e., in a sense as much compression as one can get without distorting the information [BELL90]. The coding procedure is carried out by a universal arithmetic codec that compresses/decompresses a given symbol using an estimated probability for its occurrence. The codec has to be complemented by a model which is to supply the probabilities to the codec. The encoding part has been implemented in the object-oriented style in C++, based on the sample code given in [BELL90]. However, a simple adaptive first order model found in the book was significantly modified and tailored to the distribution of the wavelet decomposition coefficients for gray-scale still images. The distribution of the coefficient values is found to be almost ideally modeled by the Lorentzian curve with a very strong peak at zero, as the following plot demonstrates:

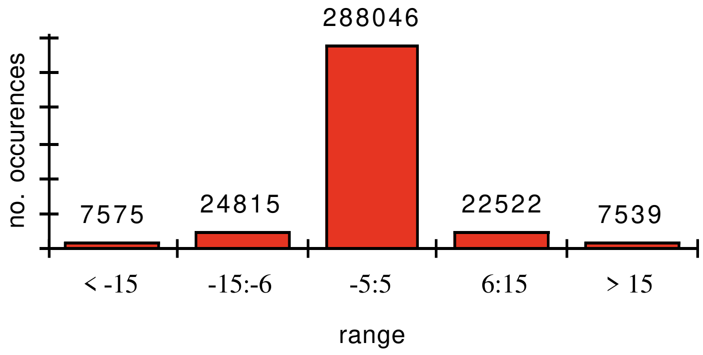


Fig. 10. Histogram of c_{ij}^{k} for the sample image ‘lenna’.

This a priori information is used for an initial setup of the symbol frequency tables in the prediction model. As symbols are processed, the frequency tables are updated in a way similar to that described in [BELL90] to finely adjust the distribution. However, rescaling of the frequencies is carried out more frequently than is necessary to prevent overflow in the coder. The reason is that the rescaling of the model makes it gradually forget past experiences, and tunes in the model to the recent data. It makes sense if the ideal source model changes with the time, as is certainly the case in the Laplacian pyramid, where the probability distribution for different layers may be different. A number of parameters controlling the adaptation of the model and the rescaling were carefully adjusted for typical image decompositions. Note that the initial setup, adaptation and rescaling of the model implicitly increases its order; that is why we will refer to the model as a mixed predictive adaptive model adjusted to the Laplacian image decomposition.

Another major innovation of the predictive model is the use of the histogram to determine which symbols are expected to occur and how often.

It effectively solves the zero-frequency problem [BELL90]. Indeed, the simple adaptive model wastes the probability space by assigning the frequency count of 1 to a number of symbols that might have appeared but never occur in reality. For example, a decomposition coefficient c_{ij}^{k} for 8-bit images can, in principle, take any value from the interval [-255,255]. The simple adaptive model assigns all possible 511 symbols a frequency count of at least 1. However, only about 130 of these symbols turn out to occur for typical gray-scale images; i.e., only a 130/511 fraction of the probability space is utilized. In the present model, the histogram of symbols (decomposition coefficient values) is built prior the encoding to find out which symbols actually occur, and to estimate very roughly their probability. The frequency tables of the model are allocated only for the symbols that actually occur rather than those potentially needed. The frequency information from the histogram is also used for the initial distribution setup. The tables are subject to modification and rescaling as symbols are encoded, much in the same way it was outlined above. The following table gives an example of compression of the Laplacian pyramid for a 512x512 test image 'clouds'. The pyramid has been trimmed with the threshold τ=50.

Compression Method	Size of the encoded file, bytes
Run-length+*UNIX* compress	16,070
Run-length+Witten's arithmetic	14,106
Mixed context adaptive model	18,680
Histogram-based adaptive model	12,435

Table 1. Comparison of different encoding schemes for the Laplacian pyramid image decomposition of 'clouds'.

Note that the last two procedures mentioned in the table (which are based on the predictive models developed in the present research) encode the stream of decomposition coefficients as it is, without the run-length coding of commonly occurring stretches of zeros. However, the overall compression does not suffer, which demonstrates the optimality of the arithmetic compression. As one can see, the histogram-based mixed context predictive model (discussed above) together with the arithmetic encoder, outperform general compression procedures such as LZC (*UNIX* compress utility), adaptive Huffman (*UNIX* compact utility) and a generic first order model arithmetic encoder [BELL90] in compressing the Laplacian pyramid image decomposition.

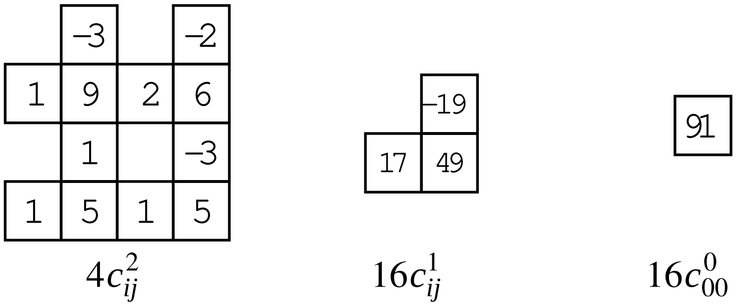
3.3 Quest for the Minimum Entropy Laplacian Pyramid

It should be stressed once again that the decomposition $\{c_{lm}^{k}\}$ is an exact representation of the original image, no matter how the coefficients are evaluated, renormalized, or encoded. However, different methods of computing the decomposition may lead to image representations with different entropies. To illustrate that, consider the following simple example of a 4x4 gradient-filled image:

1	2	3	4
3	5	5	6
5	7	8	8
7	8	9	10

Fig. 11. Sample image with the zero order entropy H=3.155.

The Laplacian pyramid decomposition, eqs. (3-4), of the sample image is shown on the next figure, where the average in eq. (3) is assumed to be an *exact* mean arithmetical value. Since this operation generally does not yield an integral result, we need to perform a special normalization to make the decomposition coefficients integral.



	–3		–2
1	9	2	6
	1		–3
1	5	1	5

$4c_{ij}^2$

	–19
17	49

$16c_{ij}^1$

91

$16c_{00}^0$

Fig. 12. Tight wavelet decomposition with the entropy H=3.250.

Note that this decomposition is tight. Indeed, if the average in eq. (3) is an exact mean arithmetical, the sum $c_{2i,2j}^k + c_{2i+1,2j}^k + c_{2i,2j+1}^k + c_{2i+1,2j+1}^k$ is precisely zero. It means that one coefficient of every four may be discarded without any loss of information; in other words, the highpass band can be downsampled. It has an advantage that the total number of coefficients, 16, in the decomposition, Fig. 12, is exactly equal to the number of pixels. However, the entropy of the coefficients is higher than the original image itself.

Taking the average in eq. (3) as a rounded mean arithmetical value yields already integer coefficients without any need for normalization. The Laplacian pyramid in this case is as follows:

$$\boxed{\begin{matrix}-2 & -1 & -2 & -1\\ 0 & 2 & 0 & 1\\ -2 & 0 & -1 & -1\\ 0 & 1 & 0 & 1\end{matrix}} \qquad \boxed{\begin{matrix}-3 & -1\\ 1 & 3\end{matrix}} \qquad \boxed{6}$$

Basis function c_{ij}^{2} Basis function c_{ij}^{1} Basis function c_{00}^{0}

Fig. 13. Loose wavelet decomposition with the entropy H=2.679.

Note that the sum of the four coefficients is no longer exactly zero, because of the rounding in computing the mean value. It indicates that the wavelet basis is loose, resulting in a total of twenty-one coefficients of the decomposition, which is five more than the number of pixels. However, the entropy turns out to be smaller, which is a consequence of the smaller number of distinct values of the coefficients and their magnitude. The example, Fig. 11, is admittedly rather crude, but it does illustrate general features of the wavelet transform. The histogram of Fig. 10 shows that even in the case of typical real images, most of the values of the loose Laplacian decomposition are small, which contributes to the small entropy of the transform. To give another, more realistic example, let us consider different exact representations of the test image ‘lenna’, Table 2.

Image Representation	Number of symbols	Size of the encoded file, bytes	Compression ratio
Original image	262,144	265,322	
gzip compressed image	262,144	212,951	1.25:1
Laplacian pyramid	349,525	177,808	1.49:1
Renormalized pyramid	349,525	169,389	1.57:1

Table 2. Comparison of different exact representations of the test image 'lenna'.

The first row in the table refers to the image as it is. The second row describes the representation obtained by the lossless compression of the image using one of the best universal entropy encoders, *UNIX* 'gzip -best' utility (based on the LZ77 algorithm). The last two rows deal with the loose Laplacian pyramid decomposition encoded using the prediction model discussed in the previous section. Coefficients are not quantized or trimmed, therefore the compression is lossless and the representation is exact. Since both LZ77 and arithmetic compression provide the entropy encoding, the size of the output file can be considered as a measure of the entropy of the input: a stream of pixels or decomposition coefficients. As the table clearly indicates, the redundant Laplacian pyramid (with more coefficients than the pixels) has smaller entropy. Additional evidence that the transforms with the larger number of coefficients may have actually less entropy can be found in [GOLD91].

Thus, the apparent redundancy in a representation proves to have little to do with the redundancy of the information contained in the representation. This suggests that, as far as image compression is concerned, one needs to consider not the number of basis functions and their smoothness, but the entropy of the representation in designing the best image transformation. The redundancy of the representation, where it exists, can be exploited to help reduce the entropy of the resultant representation. This understanding is one of the primary features of the present research. Specifically, it implies that one ought to strive to reduce the amplitude of the transform coefficients and the number of the unique values the coefficients can take.

According to the generic framework for the Laplacian pyramid decomposition discussed in the previous section, there are two types of arbitrariness one can take advantage of to reduce the entropy of the decomposition. First, there is a wide margin in selecting the averaging of four numbers in eq. (3). It can be chosen to be a median value, median value clipped to the quartiles, any of the four numbers, an exact mean arithmetical value, a mean arithmetical value rounded to a closest integer, or to an even integer, etc. Note that the median, quartiles, rounded mean arithmetical values and their convex combinations are proved to be near best constant approximations of a function on the interval [DEVO92]. The median value and other order parameters do not seem to be very suitable candidates because they may fail to reduce the amplitude of the decomposition coefficients. Consider a simple example of four nodes of the Gaussian pyramid,

$$a_{2i,2j}^{k}=0,\quad a_{2i+1,2j}^{k}=0,\quad a_{2i,2j+1}^{k}=0,\quad a_{2i+1,2j+1}^{k}=x \tag{9}$$

where x is an arbitrary number. Taking the average in eq. (3) as a median value gives $a_{ij}^{k-1}=0$. According to eq. (4), the amplitude of c_{ij}^{k} is then the same as that of a_{ij}^{k}, that is x, which can be quite big. On the other hand, if the average in eq. (3) is taken as a rounded mean arithmetical value, then $a_{ij}^{k-1}=[x/4]$ and the amplitude of the transform coefficients is $x-[x/4]$, which is relatively smaller.

Another way of controlling the entropy of the transform coefficients is to perform the renormalization, for example, according to eq. (6) or simply as

$$c^k_{2i,2j} := c^k_{2i,2j} + c^k_{2i+1,2j} + c^k_{2i,2j+1} + c^k_{2i+1,2j+1} \tag{10}$$

with the other c^k_{ij} unchanged. Note that the Haar transform can also be obtained in the same way using an appropriate renormalization. As our experiments show, the renormalization leading to the Haar transform does not provide any improvement. Although transformation (10) gives the least entropy of the unquantized representation (the best lossless compression), the renormalization according to eq. (6) proves to be more 'robust' against the quantization.

We have experimented with a number of different methods of computing the average and performing the renormalization. Some of the best results are summarized in the following table:

Averaging/renormalization method	No. distinct values of c^k_{ij}	Size of the encoded file, bytes
Rounded mean arithmetical	129	12,502
Median	160	13,333
Mean arithmetical rounded to an even integer	124	12,424
Rounded mean arithmetical and renormalization, eq. (6)	144	12,314
The same as above with quantizing	95	9,924

Table 3. Comparison of different methods of computing the Laplacian pyramid image decomposition for the test image 'clouds'.

All the results referred to in the table are obtained for the 512x512 eight-bit test image 'clouds', which was decomposed using several modifications of the Laplacian pyramid algorithm. The transform was then trimmed with the threshold τ=50, except for the method described in the last row of Table 3 when quantization was used instead. The resulting stream of coefficients was encoded using the arithmetic coding with the histogram-based mixed predictive model discussed in the previous section. Computing the average as a rounded mean arithmetical value followed by renormalization, eq. (6), proved to be the best modification found in the present study. Incidentally, it provides the compression of 26:1 with the root mean square error (RMSE) 5.35, and almost without any visual degradation in the image quality. The examples of several test images restored after being compressed with the best found algorithm are given below.

We should point out that printing a gray-scale image with 256 shades of gray is a formidable task. It is clear that the printed image would always be different (and much worse) than that displayed on the screen of the terminal with at least 8-bit depth. That is why taking pictures from the screen of a high quality monitor still remains the best way of including the gray-scale images in a document. However, if one does wish to print such pictures, a special processing is necessary to produce the output that retains at least a fraction of features and visual appearance of the original image. One method of such processing is a histogram equalization [PAVL82], which achieves full utilization of the dynamic range of pixels and yields the pixel values distribution with brightness levels spaced apart. Sometimes better results can be produced by a contrast enhancement technique, a normalization of the contrast by forcing the lightest pixels to white, the darkest pixels to black, and linearly rescaling the ones in between. For dark images, the enhancement procedure yields a better printed picture if one first exchanges black and white (called LUT, the color look-up table, inversion). Light, bright images may require another correction, which is essentially a recalibration of pixel values to make a printed dot appear with the same subjective darkness as the corresponding spot of the displayed image. This recalibration is called γ-correction [PAVL82]. Most image editing packages can carry out these corrections, in particular, a freeware *Image* package created and distributed by the National Institute of Health. We indicate the image correction method in captions to the figures.

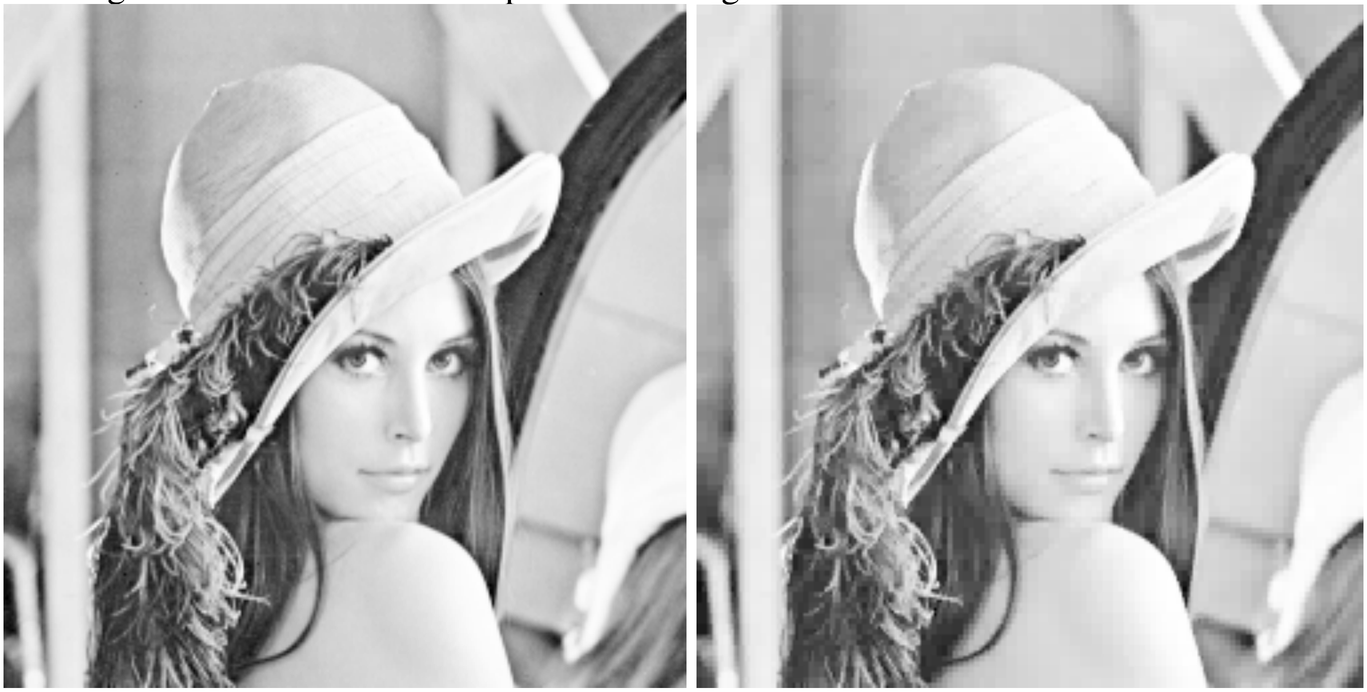


Fig. 14. Original and compressed image ‘lenna’, threshold 50, compression ratio 19:1, RMSE 6.60, $\boldsymbol{L}_1$ error 4.25. Printed at scale 2:1 after the γ-correction with the factor 1.85.

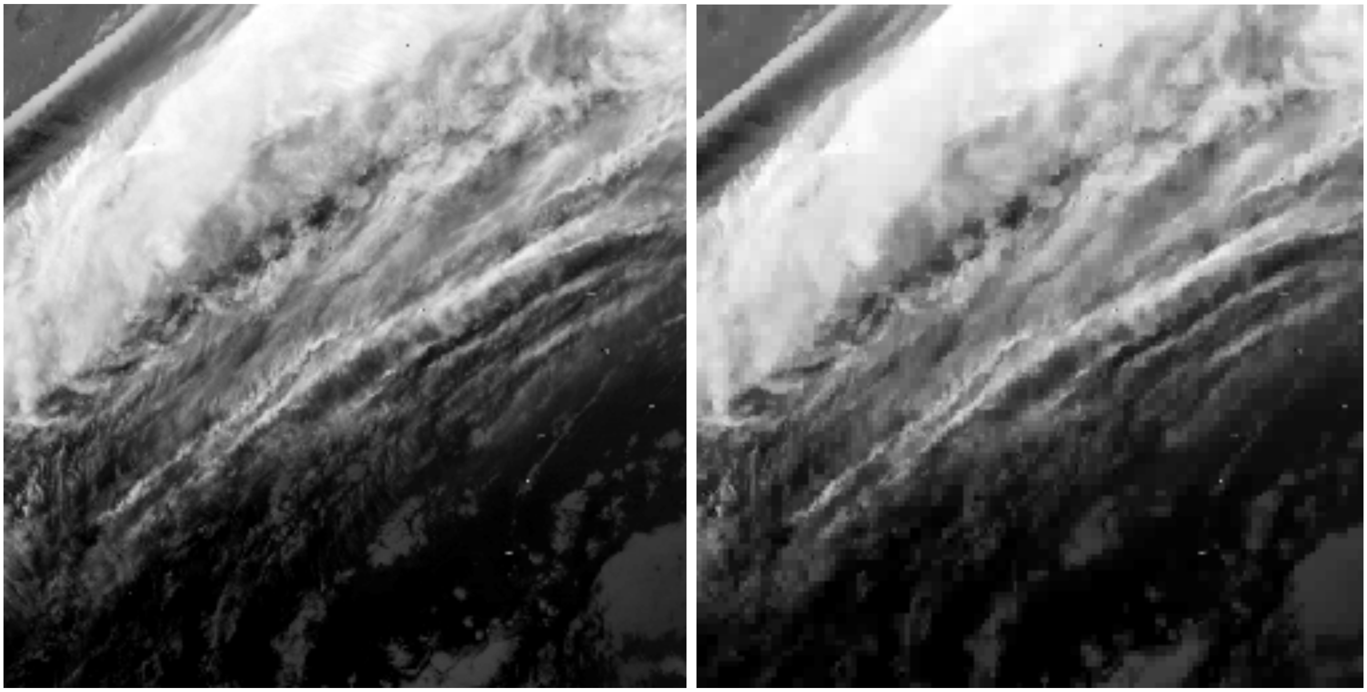


Fig. 15. Original and compressed image ‘clouds’, threshold 50, compression ratio 26:1, RMSE 5.35, $\boldsymbol{L}_1$ error 3.64. Printed at scale 2:1 after the contrast enhancement of the inverted LUT.

Figure 16 is a test image from the National Imagery Transmission Format (NITF) Technical Board, released to the public domain by Bill Puckett (Chairman).

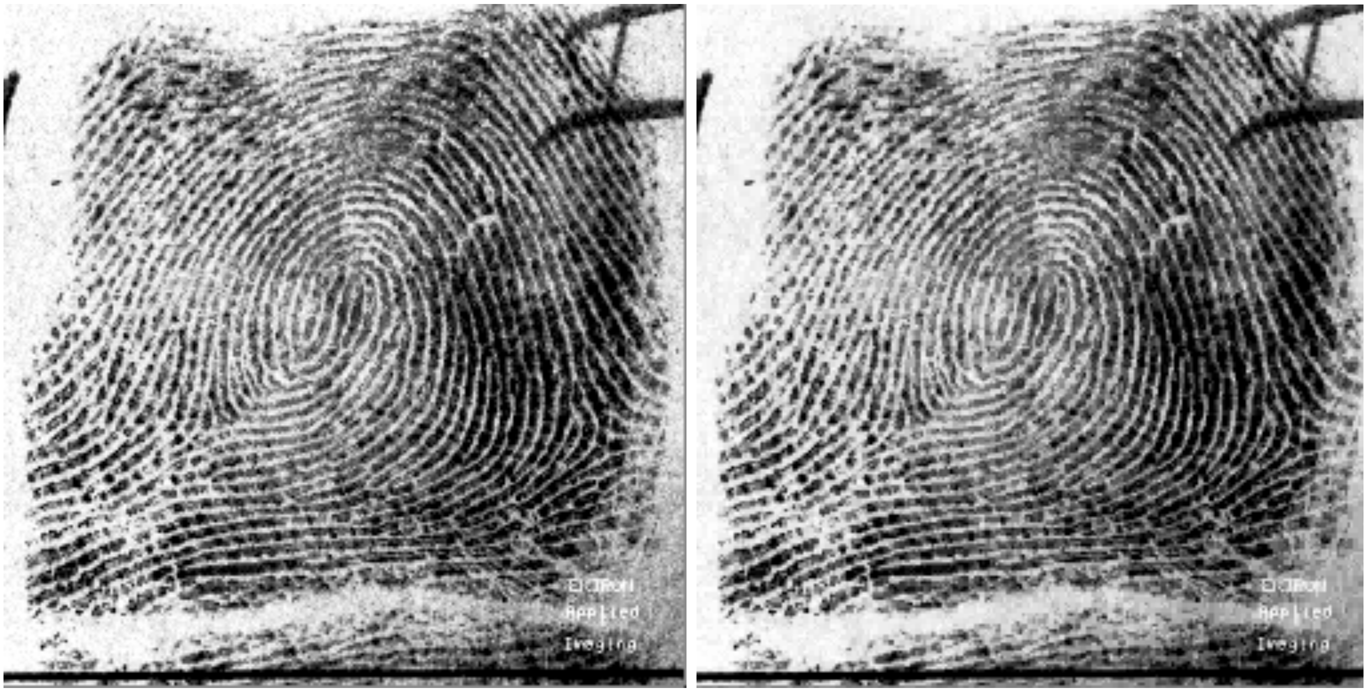


Fig. 16. Original and compressed image ‘fingerprints’, threshold 80, compression ratio 9:1, RMSE 25.50, $\boldsymbol{L}_1$ error 20.17. Printed at scale 1:1 after histogram equalization.

3.4 Non-uniform Compression

As was discussed in the Introduction, it is the omission of ‘insignificant’ information and elimination of features of little importance (which however are lost and cannot be recovered) that makes possible a deep compression of images. The generic algorithm of the pyramidal image decomposition, outlined in Section 3.1, achieves such information reduction through trimming or quantization of the transform coefficients. This is equivalent to discarding certain features of the image (which are of low contrast and/or ‘small’). The algorithm of trimming/quantization discussed in Section 3.1 (and commonly used in applications of the wavelet transform [DEVO92]) pares the image features regardless of their position within the image regardless of their significance to a particular end user. This may be adequate in many cases. However, for some applications, such a sweeping elimination of details is not appropriate simply because the interest within the image is not uniform over the entirety of the view. This brings to light a problem of non-uniform image compression, a lossy image compression with the amount of loss (and, correspondingly, compression) varying over the area of the image, or, more general, over some regions of features of interest.

In order to the select areas of interest, we have defined some preliminary concepts which allow specification of portions of an image or its characteristics that certain users may wish to preserve according to the utilization or function of the image. We call the rules used to select the areas of interest **‘criteria sets’**. For example, such a set might include an algorithm that would govern the threshold value or other criterion used in approximating transform coefficients for better compression. In the long run, controlling the threshold relates to the distortion that is introduced during the compression within a specific area of image, or at specific frequencies, etc. One has to recall that a wavelet transform coefficient bears information about the scale, absolute position, and sometimes, orientation of specific image features. This is the reason that the wavelet transform lends itself perfectly to the non-uniform compression. The actual establishment of criteria sets will depend upon the final disposition and requirements of the users and the understanding of the use or purpose of the images. The criteria sets that are listed here still seem to cover all of the applications that we can find to date:

- High or low frequency portions of a picture which are characterized by the frequency with which the intensity of the adjacent pixels changes along the rows and/or columns of the image. This criteria set will be called '**FREQUENCY**';
- Edges characterized by the sharp and abrupt change in the local contrast. This criteria set will be called '**EDGE**';
- Selected areas or regions of the picture specified by their shape, size and position. Region specifications may be either absolute (relating only to the image border and its size but not to the contents of the picture), or bound to specific image edges, features, etc. This criteria set will be called '**AREA**';
- Selected ranges of pixel intensity found within a specific region. This criteria set is called '**RANGE**';
- Selected patterns of pixel arrangements such as those described by texture, colors, or relative intensity. This criteria set is called '**PATTERN**'.

Our numerous discussions with a variety of people who use image compression showed that the idea of non-uniform compression is valid and very interesting. Moreover, several implementations of the non-uniform wavelet compression have appeared since the idea was introduced in 1992 [KIFI92]. The rest of the present section discusses an original implementation of a criteria set '**AREA**' [KIFI92] that uses a weighted norm in estimating the norm of a basis function.

The general transform quantization algorithm discussed in Section 3.1 utilizes a norm $\left\|\Phi_{lm}^{k}\right\| \equiv 2^{kmax-k}$ of the basis function; see eq. (7). Since the norm does not depend on the localization of the function (on indices l and m), it implies that the image features are pared uniformly over the entire image. One of the methods that can be used to introduce a non-uniform, position-dependent norm, is to employ a weight function that controls the level of detail to be retained in any particular region of the image. The non-negative weight function $r(i,j)$ is specified as an image of the same size as that being processed. The function is used to compute a non-uniform norm of a basis function in the functional space $\boldsymbol{l}_1([0..N-1]^2)$ as follows:

$$\left\|\Phi_{lm}^{k}\right\|_{r(i,j)} = \sum_{i,j}\left|\Phi_{lm}^{k}(i,j)\right| r(i,j)$$

or, using the definition of the basis functions, Fig. 7,

$$= \sum_{i=l2^{kmax-k}}^{(l+1)2^{kmax-k}} \sum_{j=m2^{kmax-k}}^{(m+1)2^{kmax-k}} r(i,j)$$

$$= a_{r\,l,m}^{k} \times 2^{kmax-k} \qquad (11)$$

where $a_{r\,l,m}^{k}$ is a coefficient of the Gaussian pyramid decomposition of the ‘mask’ image that specifies $r(i,j)$. Strictly speaking, if the average in eq. (3), which is used to construct the Gaussian pyramid, is a rounded mean arithmetical value, the last equality in eq. (11) is only approximate. However, the high precision is not required as the user specified distinction between an ‘important’ region(s) of an image and ‘unimportant’ ones is only semi-quantitative at best. Thus, the quantization criterion for the image transform coefficient c_{lm}^{k} becomes (compare to eq. (8)):

$$\theta_{lm}^{k} = floor(\tau / 2^{kmax-k} / a_{r\,l,m}^{\;k}) + 1 \qquad (12)$$

Obviously, the quantization of the transform coefficient depends now on the localization of the coefficient (see Fig. 7 as to how subscripts l and m relate to the position of an image detail), and is controlled by the weight function. If $r(i,j) \equiv 1$ then eq. (12) reduces to the uniform quantization rule, eq. (8). On the other hand, if $r(i,j)$ is significantly larger for some (i,j)-s, then the features within the corresponding regions of the original picture would be left almost untouched during quantization. Note that the larger the degree to which an image detail intersects with the region of the 'significant interest', the less amount of loss it will incur.

Thus, the criteria set '**AREA**', telling relative importance of image regions, is specified as a new image, the mask image, which defines the weight function $r(i,j)$. The bright regions of the mask image, or in other words, where $r(i,j)$ is large, are considered 'more important'; they get more 'weight' and the features of the original image within the corresponding regions have better chances to survive trimming or quantization against the threshold. Figuratively speaking, if one imagines the mask image lying over the original one, then white spots of the mask image shield the underlying areas of the original picture during the quantization. It should be emphasized that the pixel of the mask image can take any value from 1 to, say, 255. Therefore, there is an ample leeway in assigning the relative significance to image regions. Obviously, the regions of importance can be of any form and shape. Figures 17 and 18 below compare original test images with those compressed using the non-uniform compression with the mask (weight function) that is equal to 10 within the central 256x256 square of the image, and equals 1 elsewhere. Clearly, details in the center of the restored images look almost as good as those in the original images, while the fringes are fairly blurred. However, it is very difficult to find a sharp boundary between the regions of harsh and mild quantizations. Indeed, blurring emerges very gradually. This feature can be regarded as an (somewhat unexpected) advantage of the present non-uniform compression method.

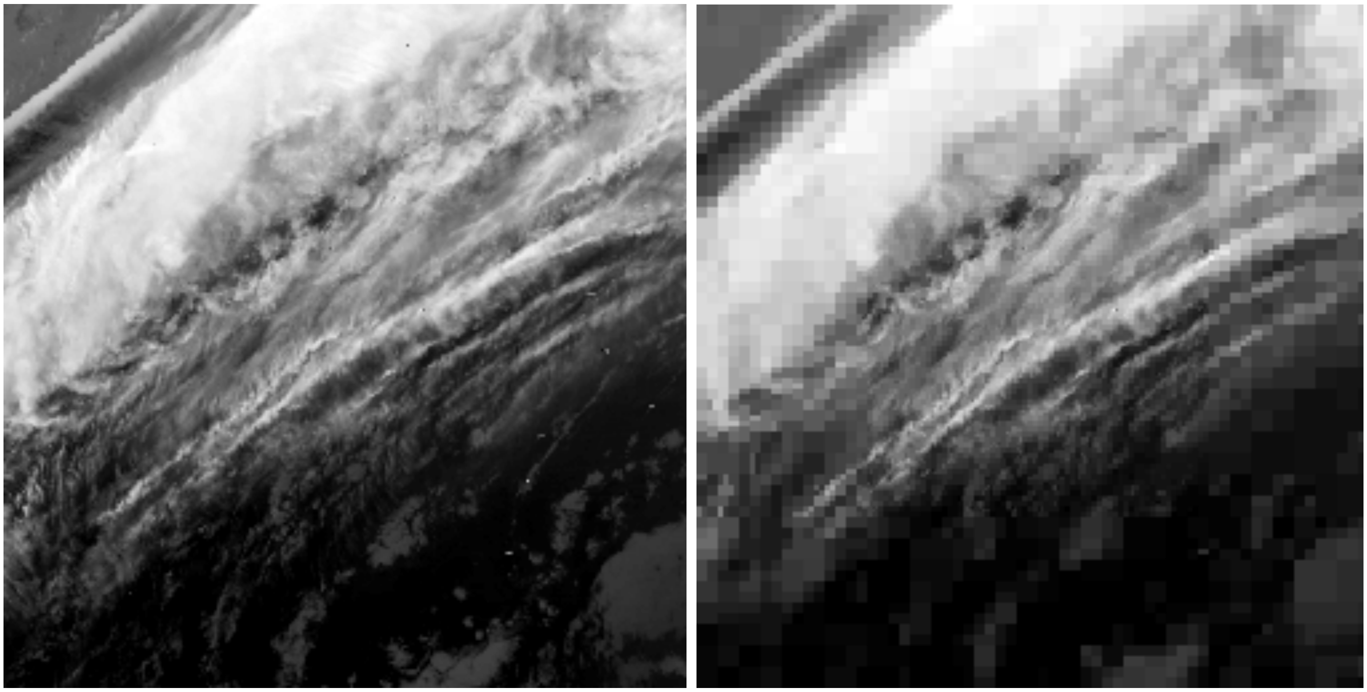


Fig. 17. Original and non-uniformly compressed image ‘clouds’, threshold 310, compression ratio 34:1, RMSE 7.12, $\boldsymbol{L}_1$ error 4.79. Printed at scale 2:1 after the contrast enhancement of the inverted LUT.

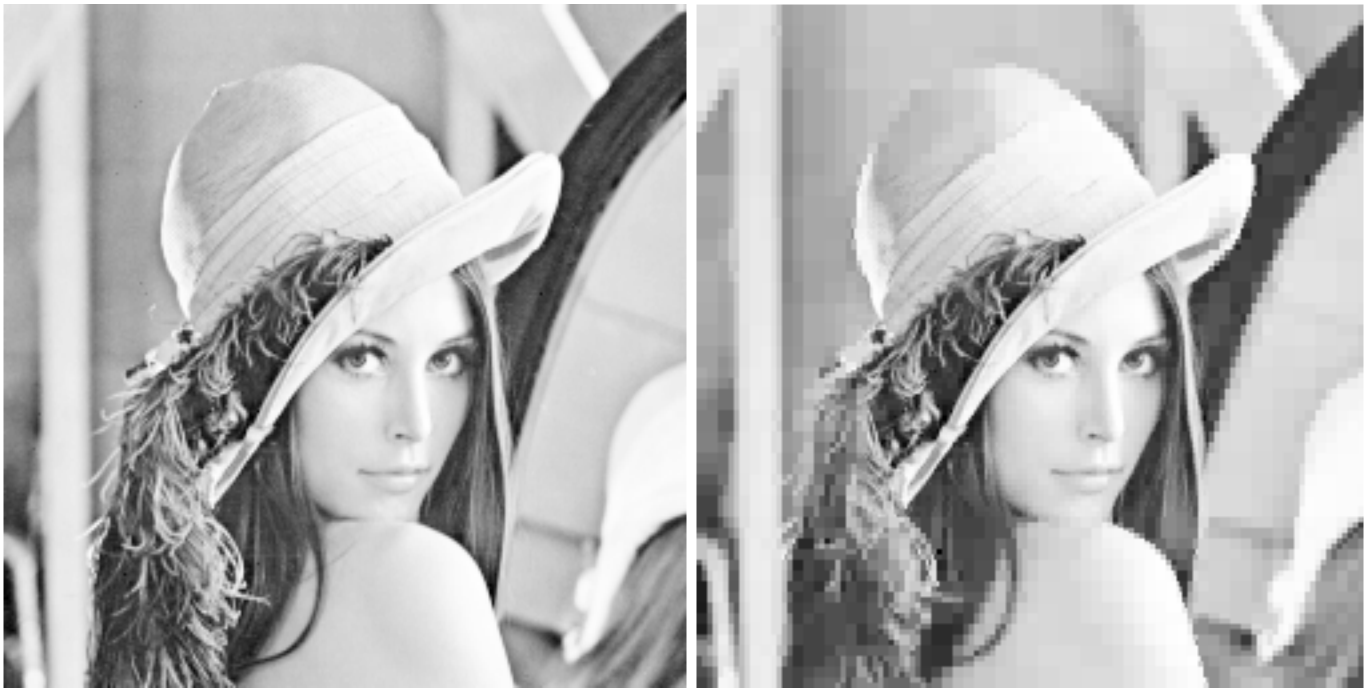


Fig. 18. Original and non-uniformly compressed image ‘lenna’, threshold 310, compression ratio 24:1, RMSE 9.17, $\boldsymbol{L}_1$ error 5.63. Printed at scale 2:1 after the γ-correction with the factor 1.85.

3.5 Filtering the Decomposed Image and Estimation of a Discrete Derivative

The image decomposition, eq. (1), may turn out to be of some help in performing various image transformations. Such transformations are often necessary or desirable in order to improve the visual appearance of the image, or to obtain some unusual visual effect, or highlight details or areas of particular interest. Another application is edge detection or, in general, finding a derivative of the image [KIFI92] where such a derivative marks changes in the local contrast in the image. Most of the transformations of this kind, for example, smoothing, low- or highpass filtration, edge sharpening, edge detection, are linear and can be represented by the following general formula:

$$image'_{ij} = \sum_{i',j'} image_{i'j'} H_{ij;i'j'} \tag{13}$$

or, if the impulse response function $H_{ij;i'j'}$ is translationally invariant, in the form of the convolution:

$$image'_{ij} = \sum_{i',j'} image_{i'j'} H(i-i', j-j') \tag{14}$$

Although the formulas look rather simple, a straightforward computation is very time consuming and requires $\Theta(N^2M^2)$ operations, with M being a window size of the transform kernel H. Utilizing the Fast Fourier Transform reduces the computational (theoretical) complexity, but at the expense of using floating point (and, in general, complex) arithmetic and evaluating transcendental functions. Multi-resolutional image decomposition, eq. (1), provides a better alternative. Inserting eq. (1) into eq. (13) gives

$$image'_{ij} = \sum_{k,l,m} c^k_{lm} \Psi^k_{lm}(i,j) \tag{15}$$

where

$$\Psi^k_{lm}(i,j) = \sum_{i',j'} \Phi^k_{lm}(i,j) H_{ij;i'j'} \tag{16}$$

Calculations in eq. (16) can generally be performed in advance. The computations become much simpler in case of a translational invariant $H(i,j)$ of eq. (14). Indeed, basis functions $\Phi^k_{lm}(i,j)$of the multi-resolutional decomposition within the same scale are all translated versions of a single function $\Phi_{00}^{k}(i,j)$. And so is $\Psi_{lm}^{k}(i,j)$, due to the linearity of eq. (16). This is represented by the following formula:

$$\Psi_{lm}^{k}(i,j) = Translation\left(\Psi_{00}^{k}(i,j)\right) \quad by\,(l \cdot 2^{kmax-k}, m \cdot 2^{kmax-k}) \qquad (17)$$

Equation (17) implies that only $\Psi_{00}^{k}(i,j)$ needs to be evaluated according to eq. (16), the other functions $\Psi_{lm}^{k}(i,j)$ can be obtained solely by translation (as a matter of fact, in the actual algorithm no translations are required at all). All this greatly reduces the complexity of the transformation to $O(\log N \cdot N^2)$. This implies that the entire process is approximately as fast as the restoration of the image after the compression, which is carried out by reversing steps 1 through 3 of the generic algorithm in Section 3.1. Obviously, this requires no floating point arithmetic. It should be pointed out that the procedures represented by eq. (17) need only be applied to the $\Psi_{lm}^{k}(i,j)$ that contribute to the image; i.e., whose coefficients c_{lm}^{k} in the decomposition are non-zeros. Trimming and quantization of the decomposition, which is necessary to achieve the deep compression, reduces the number of these non-zero coefficients to a fraction of the total number of coefficients. This fact further contracts the computational time. Our experiments show that the filtering procedures become very fast indeed.

Multi-resolutional decomposition also provides for multi-resolutional filtration, i.e., a linear transformation of the image with the kernel depending on the resolution. One possible application is smoothing out large-scale jaggedness that appears in the images which are restored after compression.

A difficult aspect of the criteria sets is an idea of finding patterns or ranges. The derivative (in the calculus sense) of an image can provide information concerning edges and frequencies. Furthermore, the derivative is immediately available from the wavelet transform. Specifically, we define the image derivative as a $\boldsymbol{L}_1$ norm of the discrete gradient vector:

$$\mathbf{D}\mathit{image} \equiv \left|\mathbf{D}_\mathrm{r}\mathit{image}\right| + \left|\mathbf{D}_\mathrm{c}\mathit{image}\right| \tag{18}$$

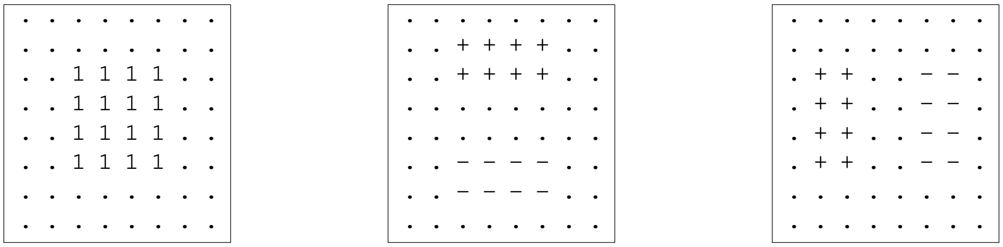
where the row and column partial derivatives are defined as

$$\begin{aligned} &\mathbf{D}_\mathrm{r}\mathit{image}(i,j) \equiv \mathit{image}(i+1,j) - \mathit{image}(i-1,j) \\ &\mathbf{D}_\mathrm{c}\mathit{image}(i,j) \equiv \mathit{image}(i,j+1) - \mathit{image}(i,j-1). \end{aligned} \tag{19}$$

This is a particular case of the general linear transformation upon an image, eq. (14) and can be treated in the same way. Specifically, a derivative operator can be transferred from the image to a basis function by inserting eq. (1) into eq. (19):

$$\mathbf{D}_\mathrm{r}\mathit{image}(i,j) = \sum_{k,l,m} c^k_{lm}\mathbf{D}_\mathrm{r}\Phi^k_{lm}(i,j), \tag{20}$$

and similarly for the column partial derivative. It is straightforward to compute partial derivatives of the basis function: Figure 19 schematically outlines a basis function $\Phi^k_{lm}(i,j)$ and its row and column partial derivatives.



```
. . . . . . . .     . . . . . . . .     . . . . . . . .
. . . . . . . .     . . + + + + . .     . . . . . . . .
. . 1 1 1 1 . .     . . + + + + . .     . + + . . − − .
. . 1 1 1 1 . .     . . . . . . . .     . + + . . − − .
. . 1 1 1 1 . .     . . . . . . . .     . + + . . − − .
. . 1 1 1 1 . .     . . − − − − . .     . + + . . − − .
. . . . . . . .     . . − − − − . .     . . . . . . . .
. . . . . . . .     . . . . . . . .     . . . . . . . .
```

A basis function $\Phi^k_{lm}(i,j)$

A row derivative $\mathbf{D}_\mathrm{r}\Phi^k_{lm}(i,j)$

A column derivative $\mathbf{D}_\mathrm{c}\Phi^k_{lm}(i,j)$

Fig. 19. Partial derivatives of a basis function

Taking a numerical derivative of a signal, it should be noted, is an ill-posed problem because it tends to intensify the noise always present in the digital signal. Fortunately, the wavelet transform offers a simple way to perform a regularization; i.e., to obtain a stable numerical approximation of a derivative in the presence of noise. Since the noise is usually related to high frequencies, which are represented by c_{lm}^{kmax}, performing trimming/quantizing of the transform coefficients with an appropriate threshold can neutralize the impact of local noise. This is an example of the lossy compression acting as a (non-linear) filter — an idea that is becoming increasingly popular [DEVO92a]. Note that this approach allows one to obtain a ‘large scale’ derivative image. Figure 20 provides an example of a ‘regularized’ numerical derivative of a test image.

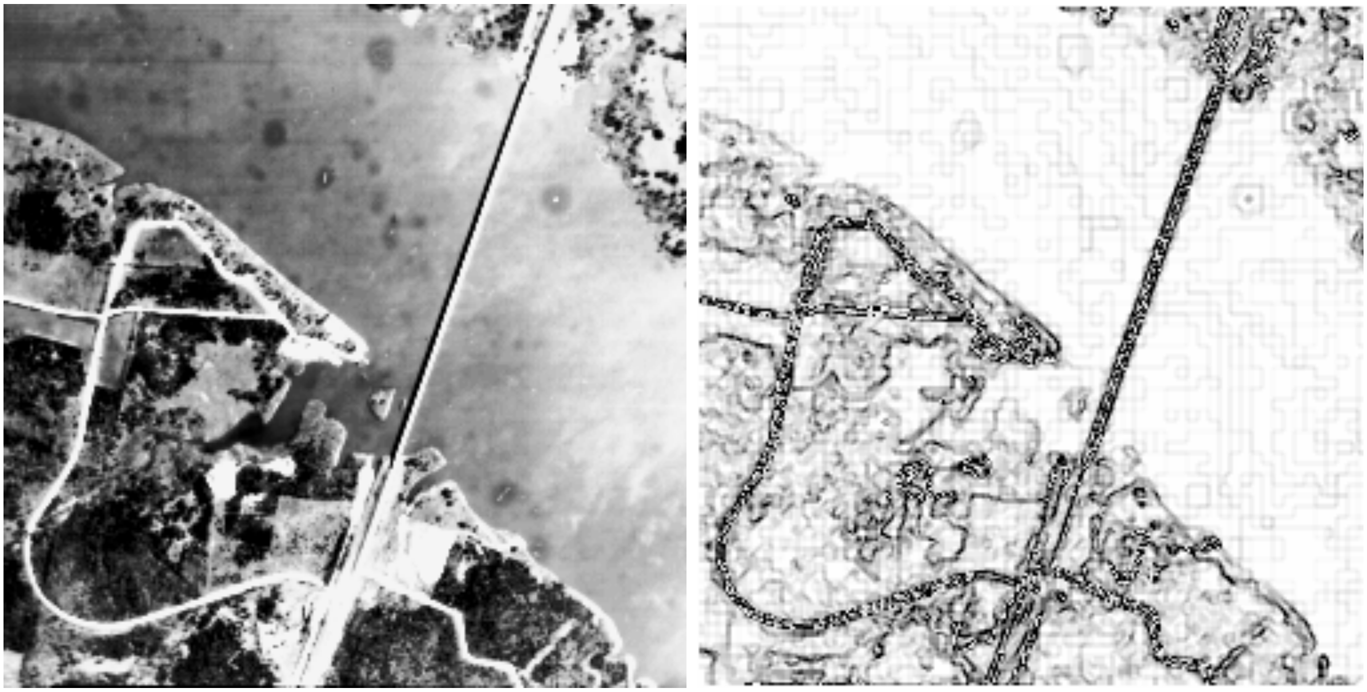


Fig. 20. Original high-altitude image and its regularized derivative, threshold 19. Printed at scale 2:1 after histogram equalization.

3.6 Smooth Laplacian Pyramid

A Smooth Laplacian pyramid is yet another pyramid representation of the image. Like the regular Laplacian pyramid, it implements the multiresolutional image analysis, eq. (1). As has been discussed earlier, decomposition (1) is an *exact* representation of an image; given the set of coefficients c_{lm}^{k}, the image can be reconstructed in its entirety without any distortion. However, if the coefficients of the decomposition are discarded, quantized or otherwise approximated to achieve better compression, composing the image back according to eq. (1) gives only an approximation of the original image. It has been shown [DEVO92] that the approximation is optimal in that it minimizes the mean square or mean absolute error at a given bit rate (provided the quantization is appropriate). However, it is well known that objective image quality criteria like the mean square error do not always agree with the subjective visual perception of the picture. In the case of the Laplacian pyramid, the distortion of the image due to the approximation of the decomposition coefficients manifests itself in block image artifacts and false edges. The tiling effect [FARR90] is especially noticeable when one approximates the coefficients corresponding to the medium scale basis functions. The following picture illustrates the effect of a harsh quantization on the test image ‘lenna’.

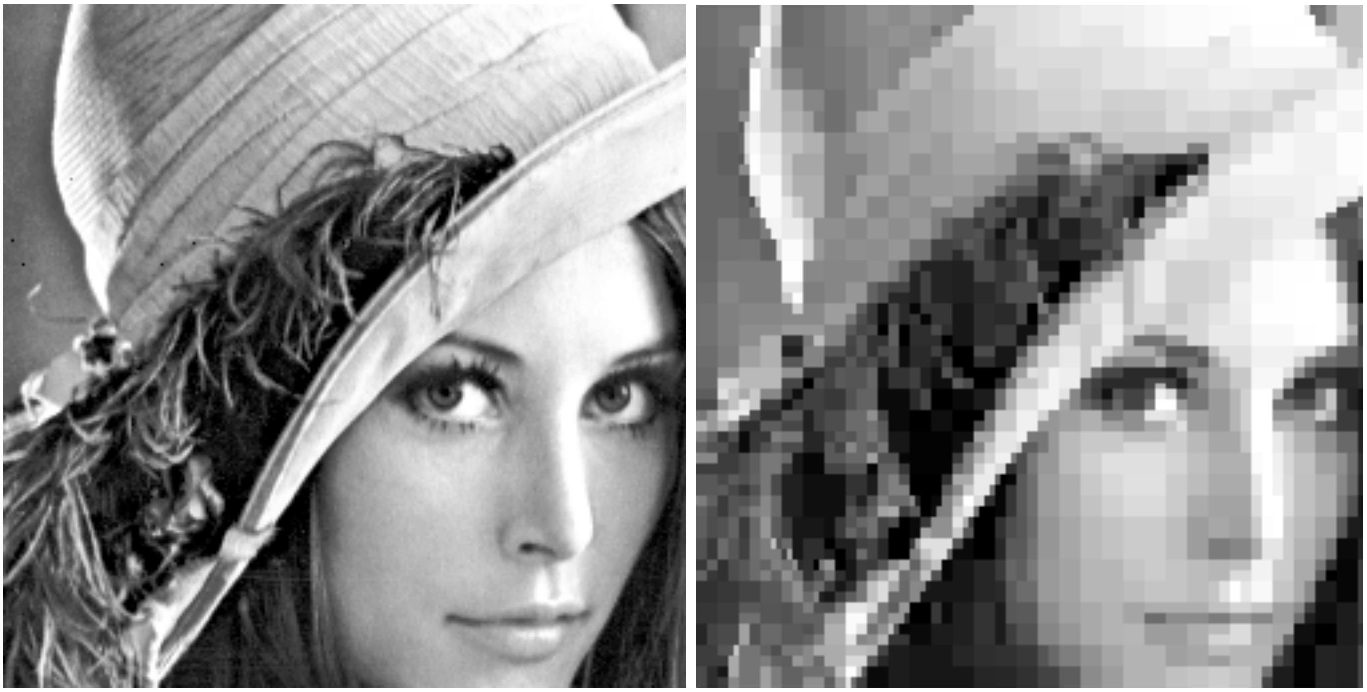


Fig. 21. A central part of the image ‘lenna’, original picture and restoration from the decomposition quantized with threshold 250.

Indeed, the quantization is harsh, but so is the compression; the compression ratio is 39:1.

Blocking artifacts and false edges can be suppressed should the basis functions be not so clear-cut. Indeed, with the basis functions of the regular Laplacian pyramid, Fig. 7, should some coefficient c_{lm}^{k} get discarded, the corresponding square (associated with the basis function) ‘vanishes’ from the image, leaving the black hole with sharply defined edges. If the basis function were a little bit blurred, the hole would not be so clear cut and noticeable; at least, it would not look like a missing square tile. Moreover, if the support regions for the basis functions partially overlap, then the vanishing of one basis function would be partly made up for by the neighbors, thus reducing the effect.

The simplest possible choice is to use a blurred version of the Laplacian pyramid basis,

$$\Phi_{ij}^{k} = \frac{1}{16} \begin{bmatrix} 1 & 3 & 3 & 1 \\ 3 & 9 & 9 & 3 \\ 3 & 9 & 9 & 3 \\ 1 & 3 & 3 & 1 \end{bmatrix} \qquad (21)$$

where each element of the matrix represents a square of the size N/2^{k+1} and the upper left square of the matrix is located at ((i-1)•N/2^{k+1}, (j-1)•N/2^{k+1}). The following figure, Figure 22, displays some of the functions (compare with Fig. 7):

```
9 9 9 9 3 3 . .
9 9 9 9 3 3 . .
9 9 9 9 3 3 . .
9 9 9 9 3 3 . .
3 3 3 3 1 1 . .
3 3 3 3 1 1 . .
. . . . . . . .
. . . . . . . .
```

Basis function
$16\bullet\Phi_{00}^{1}$

```
. . . . . . . .
3 3 1 . . . . .
9 9 3 . . . . .
9 9 3 . . . . .
3 3 1 . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
```

Basis function
$16\bullet\Phi_{10}^{2}$

```
. . . . . . . .
. . . . . . . .
. . . . . . . .
. 1 3 3 1 . . .
. 3 9 9 3 . . .
. 3 9 9 3 . . .
. 1 3 3 1 . . .
. . . . . . . .
```

Basis function
$16\bullet\Phi_{21}^{2}$

```
. . . . . . . .
. . . . . . . .
. . .16 . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
```

Basis function
$16\bullet\Phi_{23}^{3}$

Fig. 22. Example of the basis functions $16\bullet\Phi_{lm}^{k}$ for the smooth Laplacian pyramid, eq. (21).

Note that the function is clipped if it sticks out of the boundary of the image. As one can see, the functions decay not so abruptly, are overlapping and not orthogonal. However, dealing with these functions is not as easy. From general mathematical considerations, decomposition into a non-orthogonal basis is quite cumbersome and involves the solution of a set of simultaneous equations. However, redundancy of the pyramidal decomposition makes the process much easier. Indeed, we need to perform decomposition for a whole set of levels, *k*.. At each single level we are content only with an *approximate* decomposition; the error can be propagated up to the next level and resolved there. The separation of image features into different scales is distorted slightly, but the resulting decomposition is exact (in the same sense as was discussed for the regular Laplacian pyramid). Moreover, since the basis functions are blurred and overlapping, quantizing the decomposition coefficients would not be so noticeable.

The construction of the smooth Laplacian pyramid follows the same general scheme, Fig. 8. However, the implementations of the **contract** and **expand** operators have to be changed. The first version of the smooth Laplacian decomposition program provided a different implementation for only one operator, **expand**; that is, the Gaussian pyramid is constructed in exactly the same way it was done for the regular Laplacian pyramid analysis, eq. (3). As previously mentioned, we have a large choice in selecting a particular implementation for the two basic operators. An algorithm, eq. (3), for the **contract** operator works well for the regular Laplacian pyramid decomposition, and, moreover, possesses some optimal properties mentioned earlier. Since the smooth basis, Fig. 22, does not differ much from the regular one, Fig. 7, carrying out the **contract** operator according to eq. (3) appears to be quite appropriate for the smooth Laplacian pyramid decomposition. The **expand** operator must change to accommodate the new basis function, eq. (21). One of the possibilities is to use eq. (21) literally:

$$\hat{a}_{ij}^{k+1} = \sum_{lm} a_{lm}^{k} \Phi_{lm}^{k}(i,j) \tag{22}$$

Note that since the basis function $\Phi_{lm}^{k}(i,j)$ differs from zero only for a small range of i's and j's, Fig. 22, the sum in eq. (22) actually includes only a few terms. Thus, given a_{ij}^{k} evaluated according to eq. (3), the value of the node c_{ij}^{k} of the Smooth Laplacian pyramid is computed then as follows:

$$c_{ij}^{k} = a_{ij}^{k} - \tfrac{9}{16}a_{i/2,j/2}^{k-1} - \tfrac{3}{16}a_{i'/2,j/2}^{k-1} - \tfrac{3}{16}a_{i/2,j'/2}^{k-1} - \tfrac{1}{16}a_{i'/2,j'/2}^{k-1} \qquad (23)$$

Here i' is the 'closest' to i row in the next quadrant, i.e.,

$$\begin{aligned} &i' = (2m-2)\,N/2^{k+1} \text{ if } i \text{ is "even"} \\ &\qquad\qquad (\text{i.e. } i = 2m \times N/2^{k+1} \text{ for some } m\,) \qquad (24) \\ &i' = (2m+2)\,N/2^{k+1} \text{ if } i \text{ is "odd" (i.e. } i = (2m+1) \times N/2^{k+1}\,) \end{aligned}$$

and so is j'. The formula should be clear from the observation that the basis function Φ_{lm}^{k}, say, for *k=kmax-1*, covers not only its characteristic square 2x2, but spans one more pixel in each direction (see Fig. 22). It means that each node of a pyramid level is affected not only by its parent (corresponding node of the higher level), but its three closest 'uncles' as well.

Level 1 of the smooth Laplacian pyramid is treated exactly in the same way as that of the regular one,

$$c_{ij}^{1} = a_{ij}^{1} - a_{00}^{0}, \quad i,j = 0,1 \qquad (25)$$

This is because at such a coarse resolution it makes no sense to use smooth functions, as their 'fringes' lie completely outside the image and have no effect on smoothness of the image within its boundaries.

To reconstruct the image from its decomposition, one need only apply eq. (1) from right to left; in other words, compute corresponding sums. Even simpler, one can reverse the algorithm, Fig. 8, that computes the Laplacian pyramid from the Gaussian one, and take advantage of the fact that the bottom level of the Gaussian pyramid is the image itself.

The problem with the first version of the smooth Laplacian pyramid decomposition outlined above is that the **contract** and **expand** operators are no longer complementary; i.e., the consistency condition, eq. (5), no longer holds. It should be clear from the fact that the **contract** operator does not take into account that the basis, eq. (21), has a broader support than the regular Laplacian pyramid basis, Fig. 7, while the **expand** operator does. Although the consistency condition is by no means an ironclad rule, it does guarantee to a significant extent the separation of the image features according to their scale. Therefore, in the second version of the Smooth Laplacian Pyramid, decomposition program a_{ij}^{k-1} is obtained not only from $a_{2i,\,2j}^{k}$, $a_{2i+1,\,2j}^{k}$, $a_{2i,\,2j+1}^{k}$, *and* $a_{2i+1,\,2j+1}^{k}$ but also from their neighbors. The **contract** operation now is not the simple averaging, but rather a filtering with the kernel,

$$G_{ij} = \frac{1}{16}\begin{bmatrix} 1 & -3 & -3 & 1 \\ -3 & 9 & 9 & -3 \\ -3 & 9 & 9 & -3 \\ 1 & -3 & -3 & 1 \end{bmatrix} \tag{26}$$

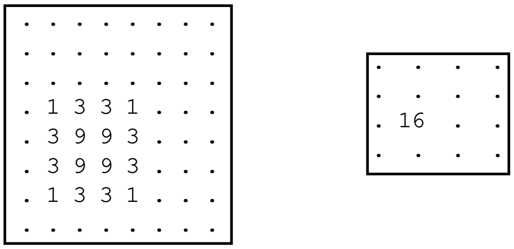
followed by decimation by two (dropping every other sample), just as in the generic case of the wavelet decomposition discussed in Chapter II. This implementation for the **contract** operator is consistent with the **expand** operator discussed above, and condition (5) now holds. However, performing a convolution as implied by the filtration may seem rather expensive. Fortunately, there is a simpler way of achieving the same results, namely,

$$a_{i/2,j/2}^{k-1} \mathrel{+}= \tfrac{9}{16} a_{ij}^{k},\quad a_{i'/2,j/2}^{k-1} \mathrel{-}= \tfrac{3}{16} a_{ij}^{k},\quad a_{i/2,j'/2}^{k-1} \mathrel{-}= \tfrac{3}{16} a_{ij}^{k}$$

$$a^{k-1}_{i'/2,j'/2} \mathrel{+}= \tfrac{1}{16} a^{k}_{ij} \qquad (27)$$

for all $i, j = 0..2^k - 1$, providing that initially all a^{k-1}_{ij} are zeros.

As was already mentioned, the clear advantage of the current version over the previous one is the consistency in the sense of eq. (5). To illustrate that, consider an image that is comprised of a single basis function located somewhere, as that on the left-hand side of the figure below:



```
. . . . . . . .
. . . . . . . .
. . . . . . . .
. 1 3 3 1 . . .
. 3 9 9 3 . . .
. 3 9 9 3 . . .
. 1 3 3 1 . . .
. . . . . . . .
```

```
.  .  .  .
.  .  .  .
. 16  .  .
.  .  .  .
```

Original image **contract**ed image

Fig. 23. Original and **contract**ed sample images.

The contracted image; i.e., the layer above the bottom one in the Gaussian pyramid, would contain only a single non-zero node which corresponds to the basis function. Applying the **expand** operator, eq. (22), yields the original image, on the left-hand side of Fig. 23. Obviously this is even more than the consistency condition, eq. (5), asks for. Incidentally, it makes the bottom level of the Laplacian pyramid completely zero, according to the generic algorithm, Fig. 8, as well as eqs. (23-24). Unfortunately, the improvement stops here. The next level of the Gaussian pyramid, obtained by **contract**ing the contracted image, on the right-hand side of Fig. 23, would contain a number of non-zero nodes because the image that consists of only a single non-zero pixel requires many basis functions with broad support to represent. The situation is similar to that in the Fourier analysis, where a single impulse requires infinitely many sine waves to represent. From the informational point of view, the multitude of non-zero decomposition coefficients means that the entropy of the decomposition is high and the compression is poor. Note that using an approximate **contract**ion, eq. (3), in the previous version of the smooth Laplacian pyramid decomposition prevented a single impulse from spreading onto the next level of the Gaussian pyramid. We have tried to combat that spreading by utilizing the redundancy in constructing the pyramid and replacing negative a_{ij}^{k-1} (which may well result from formula (27)) by zeros. The rationale here is that each level of the Gaussian pyramid should represent a view of the original image at some particular resolution; an image view at the lower resolution ought not to contain ‘negative’ pixels. It helps, but only minimally.

Another way of reducing the entropy of the smooth Laplacian pyramid decomposition is to play with the normalization of the basis function and the filters utilized to perform two basis operations, **contract** and **expand**. For example, the following basis function:

$$\Phi_{ij}^{k} = \frac{1}{4} \begin{bmatrix} 1 & 3 & 3 & 1 \\ 3 & 9 & 9 & 3 \\ 3 & 9 & 9 & 3 \\ 1 & 3 & 3 & 1 \end{bmatrix} \qquad (28)$$

differs from that specified in eq. (21) only by a normalization factor, which is now 1/4 rather than 1/16. The filter, eq. (26), used in the **contract**ion operation, has also to change its factor for 1/64 . Since the basis function has now a bigger effect (or, in other words, coefficients after filtration with the new filter come out four times smaller), the dynamic range of the node values of the pyramid is smaller. That decreases the entropy of the pyramidal representation and makes the transform coefficients easier to compress. On the downside, because of the bigger ‘weight’ of the basis functions, a slight change in coefficients has a large impact on the reconstructed image. Accordingly, upon the quantization, the compression is bigger, but so is the distortion.

Therefore, one definitely has to balance the entropy of the decomposition with the impact an approximation of the decomposition coefficients has. The following choice of the basis functions appears to be near optimal:

$$\Phi_{ij}^{k} = \frac{1}{8} \begin{bmatrix} 1 & 3 & 3 & 1 \\ 3 & 9 & 9 & 3 \\ 3 & 9 & 9 & 3 \\ 1 & 3 & 3 & 1 \end{bmatrix} \qquad (29)$$

The function has now a scaling factor of 1/8 (unlike 1/4 in the previous version), and the Gaussian pyramid construction filter has the scale factor of 1/32 (compared to 1/64 in the previous version). The smaller factor at the basis function has now made the cost of one unit of the coefficient value slightly lower, and the value itself slightly larger. This results in a bigger amplitude of the transform coefficients, and, therefore, smaller (relatively to the previous version) compression. However, the quantization of the coefficients will not lead to large distortion, as it did in the previous version. Actually, the results turned out to be better than those for the original version of the smooth Laplacian pyramid, eqs. (3), (23-24); the present version provides a better compression at a smaller distortion (error). The following picture in Fig. 24 shows the result of a test image compression using the new version of the smooth Laplacian pyramid.

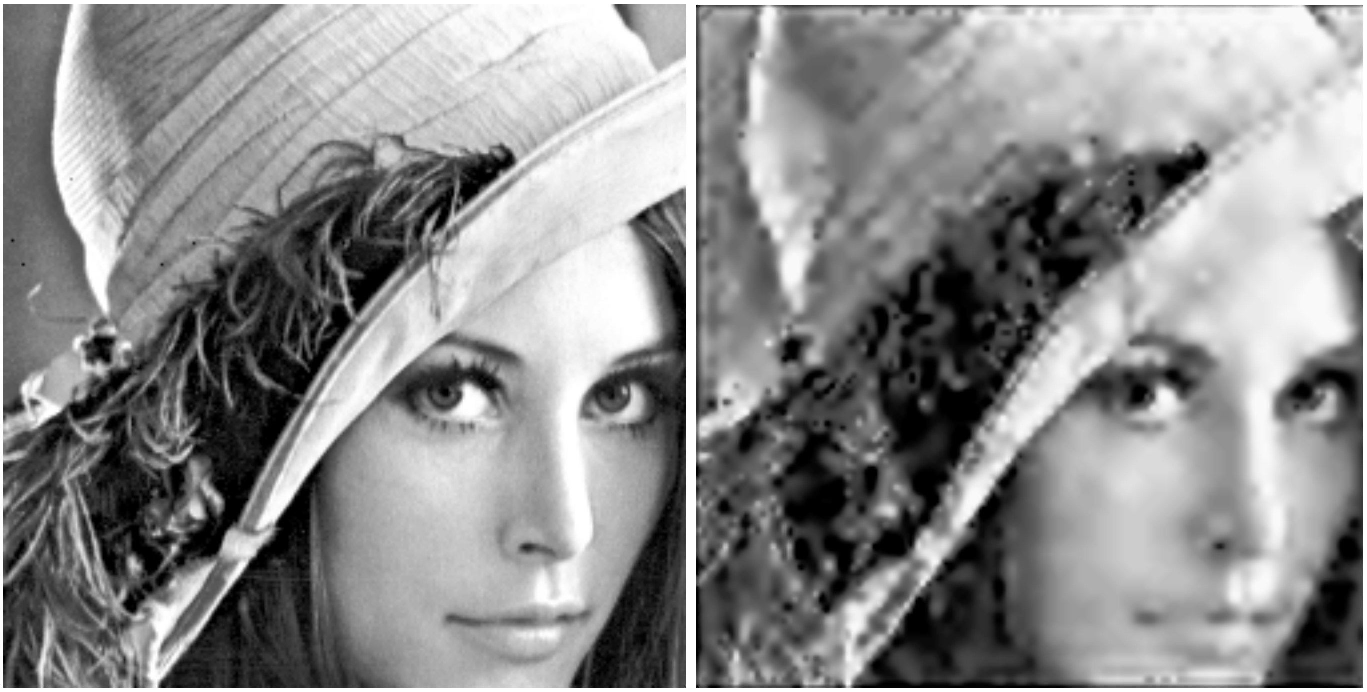


Fig. 24. A central part of the image ‘lenna’, original picture and restoration after the compression, threshold 100, compression ratio 25:1. Printed at scale 1:1 after histogram equalization.

Another option of the basis for the smooth Laplacian decomposition that has been tried is

$$\Phi_{ij}^{k} = \frac{1}{64} \begin{bmatrix} 1 & 7 & 7 & 1 \\ 7 & 49 & 49 & 7 \\ 7 & 49 & 49 & 7 \\ 1 & 7 & 7 & 1 \end{bmatrix} \tag{30}$$

Note that the function is more contained within the characteristic square and has a smaller margin. In a sense, it is a middle point between the sharp basis of the regular Laplacian pyramid, Fig. 7, and a significantly blurred basis on Fig. 22. The **contract**ion filter is given by the following formula:

$$G_{ij} = \frac{1}{256} \begin{bmatrix} 1 & -9 & -9 & 1 \\ -9 & 81 & 81 & -9 \\ -9 & 81 & 81 & -9 \\ 1 & -9 & -9 & 1 \end{bmatrix} \tag{31}$$

The pictures below show some sample results of the compression with that type of the smooth Laplacian pyramid. Note that the pyramid gives very little distortion (as seen from the RMSE error, and visually).

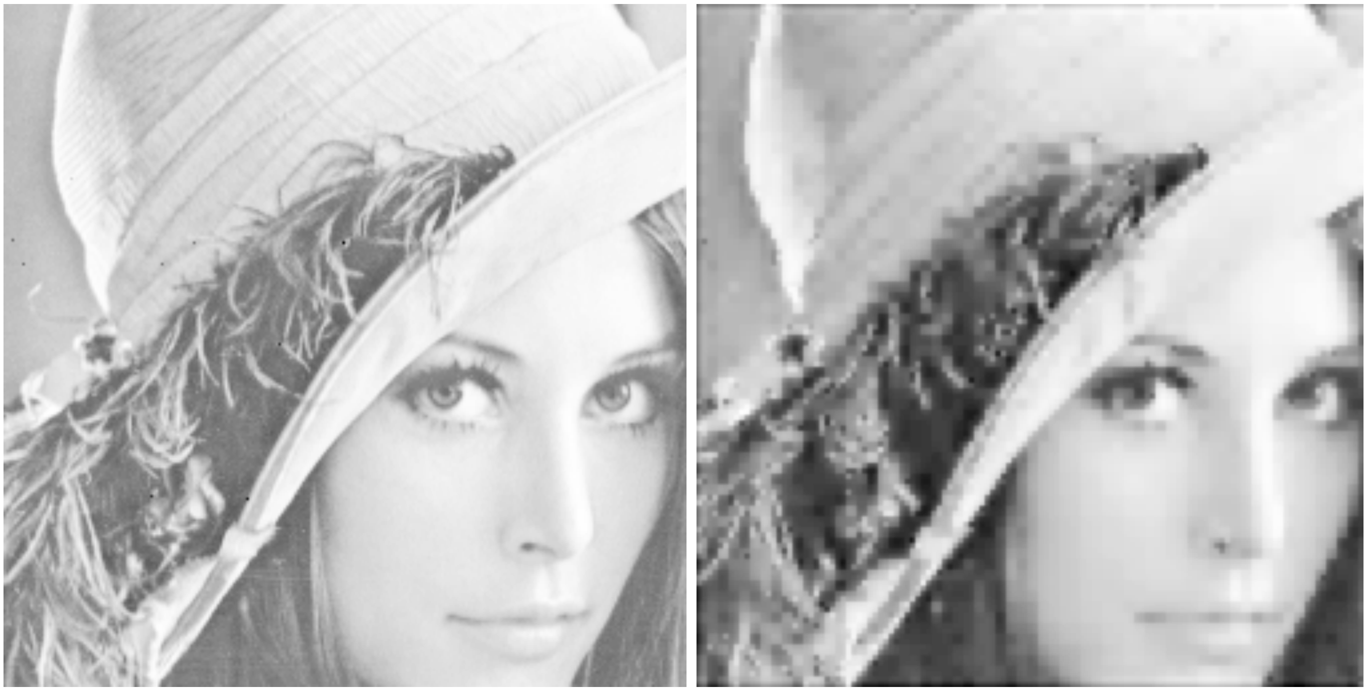


Fig. 25. A central part of the image ‘lenna’, original picture and restoration after the compression, threshold 100, compression ratio 19:1, RMSE 10.70, $\boldsymbol{L}_1$ error 7.50. Printed at scale 1:1 after the γ-correction with the factor 1.85.

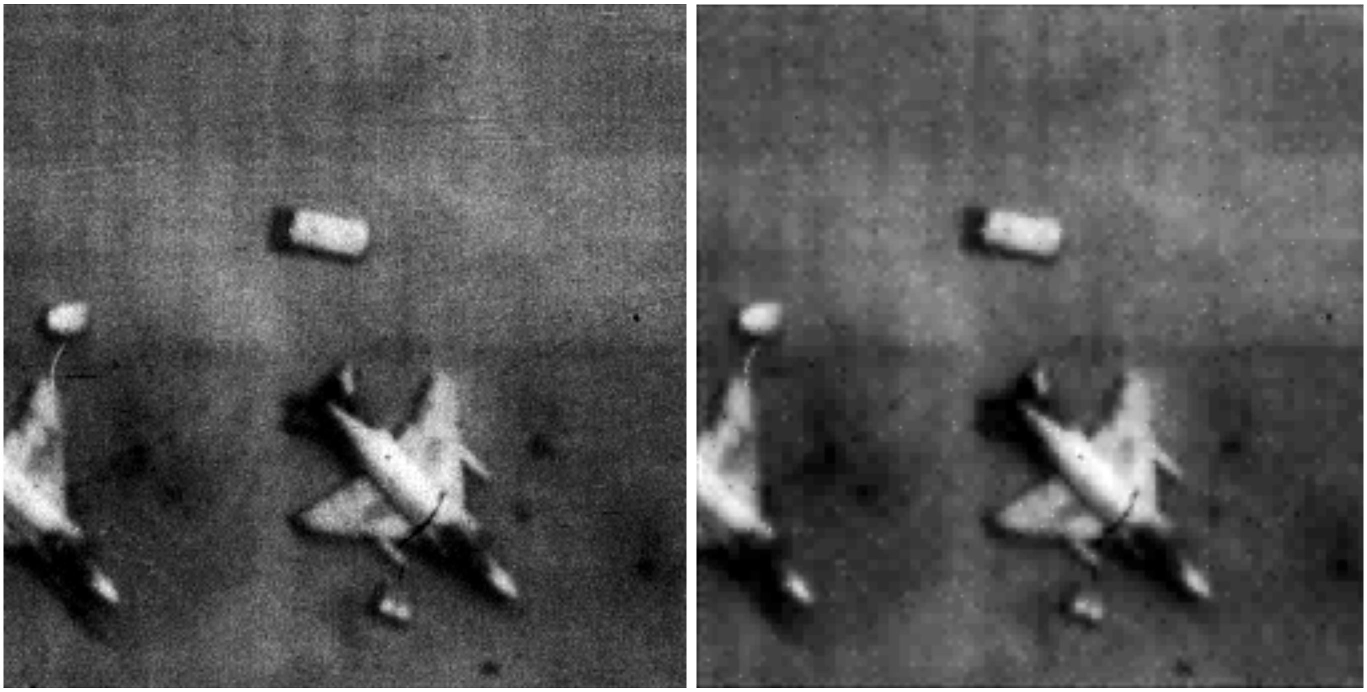


Fig. 26. Original and compressed image NITF11, threshold 50, compression ratio 52:1, RMSE 4.82, $\boldsymbol{L}_1$ error 3.71. Printed at scale 2:1 after the contrast enhancement of the inverted LUT.

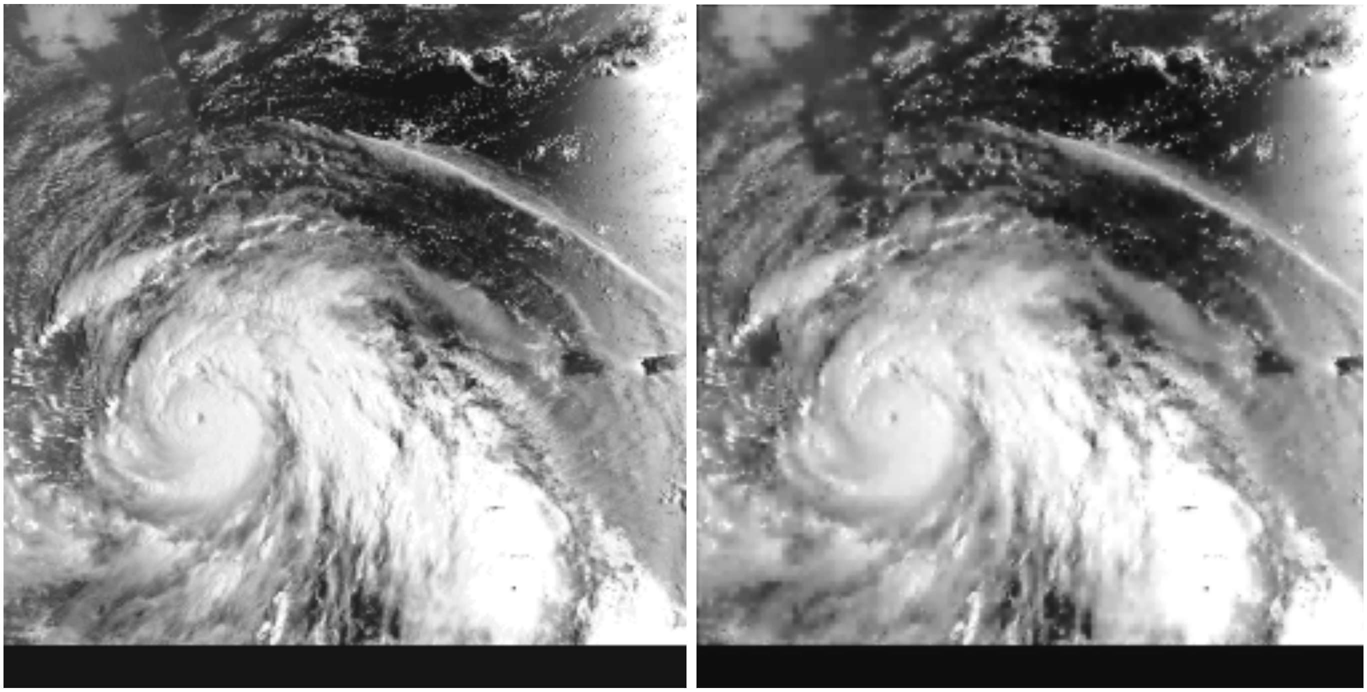


Fig. 27. Original and compressed image of the hurricane Gilbert, threshold 100, compression ratio 17:1, RMSE 15.46, $\boldsymbol{L}_1$ error 10.25. Printed at scale 2:1 after histogram equalization.

CHAPTER IV

SMART PICTURE MAGNIFICATION BASED ON THE LOCAL SELF-SIMILARITY OF THE MULTIRESOLUTIONAL DECOMPOSITION

4.1 The Problem and the Objectives

The present chapter deals with an alternative scheme to eliminate the redundant information from the image on compression and to fill it in on decompression. The technique is based on, and makes extensive use of the property of self-similarity. However, unlike the Fractal Image Compression algorithm discussed in Chapter II, we are estimating and exploring the self-similarity within the multiresolutional pyramidal decomposition of the image rather than within the image as it is. Specifically, the redundant information in an image, which comes mainly from local correlations in pixel intensities, can be effectively reduced by simply squeezing (or decimating) the picture. This by itself can be regarded as a very fast compression. Indeed, squeezing an image 4 times in each dimension reduces the amount of required storage 16 times. Compressing the shrunken picture at least 10:1 (which as we saw in Chapter III introduces little visual loss) would result in a hefty overall compression of 160:1! Unfortunately, a dumb expansion of the squeezed image back to its original size gives a very jagged and blocky picture (see examples below). Therefore, the problem is to explore the remaining (mostly high-level) correlations in the squeezed image to perform a smart image magnification, which ought to result in a smooth and ‘natural looking’ picture.

Thus, our goal is to find a way to estimate the self-similarity of the multiresolutional image decomposition and perform an image expansion based upon that. We have developed an idea of a formula of self-similarity to estimate the self-similarity in the pyramidal decomposition. The next section illustrates the idea on a very simple example of a self-similar image. We will next introduce a concept of *local* self-similarity in a complex image and consider its application to a ‘smart’ image expansion. Several particular implementations will be discussed and the results of the smart image expansion will be given for a variety of images. Simple images of straight lines, gradient fills and other basic elements that constitute a complex realistic picture will help to clarify the potential of this approach and its implementations. In conclusion, we will show how the method works in the case of realistic complex images, including frames of a weather video broadcast. It is significant that the algorithm is very fast, much faster than that based on the Iterative Function System Fractals [BARN93].

4.2 A Simple Example of the Self-similarity in the Multiresolutional Image Decomposition

We start the discussion of self-similarity and how it emerges in the multiresolution decomposition by examining the following example of a simple self-similar image. Each small square on the image in Fig. 28 stands for a pixel of an 8x8 image.

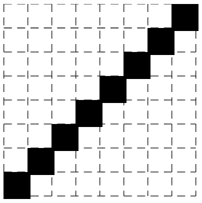


Fig. 28. A simple test image.

The image appears too trivial to be called fractal, but in fact, it is one of the simplest fractal images. Consider, for example, a set of views of the image at different resolutions, which are stacked one upon another to build a pyramid-like structure, a Gaussian pyramid [BURT83] (Fig. 29).

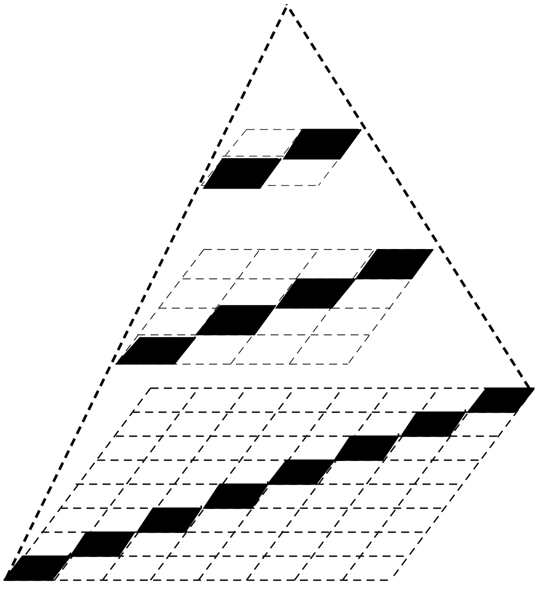


Fig. 29. Gaussian pyramid for the simple fractal image.

The bottom of the pyramid is the original image itself. The layer just above it is a view of the image at a coarser resolution, obtained by replacing a 2x2 square of 4 pixels by one pixel. This pixel takes an average brightness of the original 2x2 square. In our case, the new pixel is blackened if the 2x2 square contained at least two black pixels. The middle layer is, in turn, reduced to give the top layer of the pyramid. From this example we can easily see that all image views at different resolutions ‘look alike’, indicating that our test image is indeed fractal.

More complicated examples of the pictures that possess the same property are clouds and coastline maps. “Until now we stressed that coastlines' geometry is complicated, but there is also a great degree of order in their structure. Although maps drawn at different scales differ in their specific details, they have the same generic features. In a rough approximation, the small and large details of coastlines are geometrically identical except for scale. When each piece of a shape is geometrically similar to the whole, both the shape and the mechanism that generates it are called *self-similar*” [MAND83]. This notion can be given a more precise meaning. Consider the first two top layers of the pyramid. The top 2x2 image represents the entire image at a very coarse resolution; but it is identical to, say, the left lower quadrant of the layer below it, which represents only a part of the original image. Thus, we are able to map an entire image into a part of itself at a finer resolution. So, in precise terms, “A bounded set S is *self-similar*, with respect to the ratio r and an integer N, when S is the union of N nonoverlapping subsets, each of which is congruent to r(S), a scaled version of S. *Congruent* means identical except for displacement and/or rotation” [MAND83].

As far as the pyramid in Fig. 29 is concerned, the property of self-similarity can be expressed in the form of a precise algorithm that tells how an image view at some resolution can be arranged such that it produces an image view at a lower resolution. In the case of a Gaussian pyramid, the difference in resolution between two adjacent layers is always 4, two in each dimension.

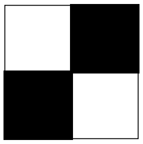


Fig. 30a. Image A, a view of the original image at the very coarse resolution.

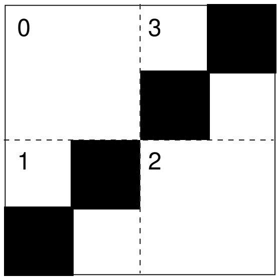


Fig. 30b. Image B, a finer view of image A, divided in quadrants.

Figure 30 displays two views of the original test image, Fig. 28, at resolutions 4:1 and 2:1. The images are two adjacent layers of the Gaussian pyramid, Fig. 29. Obviously, image B is 4 times bigger than A. Image B is split into 4 quadrants, labelled by the numbers in the upper left corner of a quadrant. A quadrant of image B has exactly the same size (dimensions) as the entire image A, so that it is possible to consider a mapping between A and quadrants of B. These mappings define a formula of the self-similarity of the image, which in our case looks like—

quadrant #0 of B is equal to 0 * A
quadrant #1 of B is equal to 1 * A
quadrant #2 of B is equal to 0 * A
quadrant #3 of B is equal to 1 * A

Exactly the same relationship holds between the middle layer of the pyramid, Fig. 29, and the bottom one. Consequently, the formula above is a unique description of the image, and captures all its essence. It opens up a possibility of expanding the image to get an image at even finer resolution. Indeed, applying the formula to the original image in Fig. 28 gives us Fig 31a. Fig. 31b depicts the result of a 'dumb' expansion; i.e., inflating each pixel into a 2x2 square. Obviously the expansion based on the property of self-similarity looks less jagged and blocky, and goes along with our intrinsic feeling as to how the expansion of Fig. 28 should look.

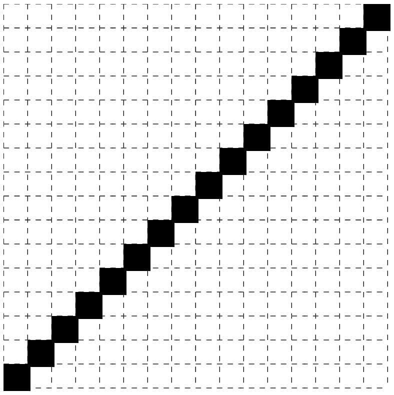


Fig. 31a. 'Smart' expansion of the original image, Fig. 28, based on the formula of self-similarity.

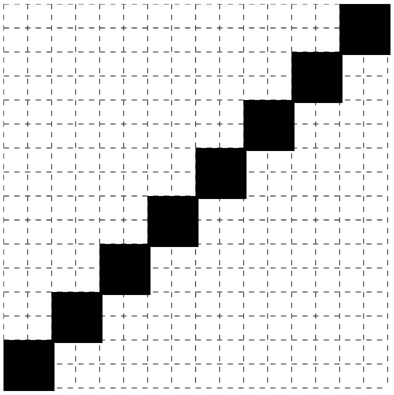


Fig. 31b. 'Dumb' expansion of the original image, Fig. 28.

In general, the relation between the image and a quadrant of itself at a finer resolution may be more complicated than in the simplest case shown above. First, a quadrant of the finer resolution image may be darker or brighter than the entire coarse resolution image by some specific factor. Moreover, we might need to rotate image A and/or perform the reflection before trying to match it with quadrants of B. Thus the formula of the self-similarity between two images is defined to be a set of the four pairs. Each pair {x,y} in the set tells the relation between the entire smaller picture and the corresponding quadrant of the larger image. The first number in the pair, x, specifies that the image A is to be rotated by $|x| \times 90^\circ$ clockwise and, if x < 0, reflected in y-axis. The second number, y, tells that the intensity of all the pixels of A is to be increased y times. After rotation, flipping, and intensity scaling, the image A then matches a corresponding quadrant of B.

4.3 Local Self-similarity and 'Smart' Expansion

Realistic images with which we are concerned are a bit more complicated than the test example, Fig. 28. One can hardly expect that, say, a frame of the weather broadcast video can be described entirely by a single simple formula as we have discussed earlier. However, we believe that a small separate part of the complicated picture does possess that property. Indeed, one small piece of an image may well be a segment of the straight line, which as we saw is very well described by the formula of self-similarity. Moreover, clouds, flowers, rugged views of mountains, and other well-known examples of self-similar images can make up a (significant) part of a realistic picture. In other words, although the property of self-similarity normally holds only for fractal images, one can hope that it applies *locally* for usual images as well.

At present, we have explored the application of self-similarity to the 'smart' expansion of the image as sketched above. In more detail, the procedure is as follows: Given an image, simple or realistic, it is reduced 4 times in each dimension, 16 times total. We construct different kinds of multiresolutional representations for the shrunken image, and estimate the local self-similarity between the bottom level of the pyramid and the level just above it. The formulas of self-similarity are then used to predict a new layer of the multiresolutional pyramid, one level deeper than the bottom level. The procedure is then repeated and the pyramid is extended one more level. Thus, the multiresolutional pyramid of the 4 times shrunken image is extended by two levels in depth. From that extended pyramid, we can reconstruct an image, which in each dimension is 4-times larger than the shrunken image. Thus we shrank the original image 16 times and then expanded it 'smartly' to the original size. Comparison of the two images, original and shrunken/expanded, then gives an idea how good the expansion procedures is.

Local self-similarity can be estimated by comparing quadrants of the 4x4 square of the bottom level of the pyramid with the 2x2 pixel square of the higher level that is a view of the original 4x4 square at a coarser resolution. Fig. 32 displays the squares we are talking about.

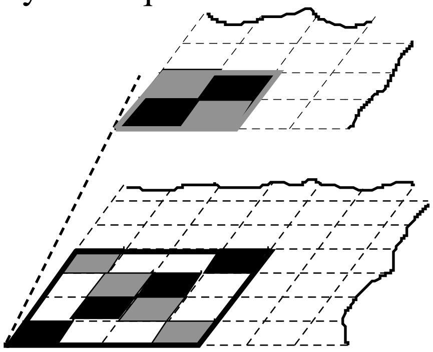


Fig. 32. Local self-similarity in the multiresolutional pyramid.

As demonstrated above, the formula of the self-similarity tells how one image (a 2x2 square in our case) can be rotated and brightened/darkened to match a part of itself at a finer resolution (a quadrant of the 4x4 pixel image). Apparently, it is not always possible to get a perfect match; i.e., no rotation and scaling of the intensity of the 2x2 square can give an image that looks *exactly* like some quadrant of the 4x4 square. However, we are content with a match which is rather good, though it might not be perfect. To choose among non-perfect matches we need some quantitative criterion. At present, we are using a scaled projection of one image considered as a multi-dimensional vector onto another. Thus if a 2x2 block of pixels is considered a multidimensional vector, then the similarity between two vectors may be related to an angle between them. Indeed, if the angle is zero, we have a perfect match; on the other hand, if the vectors are orthogonal, they have nothing in common. The cosine of the angle between vectors *v1* and *v2* can easily be estimated from their scalar product,

$$\cos(v1, v2) = \frac{(v1, v2)}{\|v1\|\|v2\|} \qquad (1)$$

If the cosine is zero or negative (meaning the vectors are orthogonal or pointing to different directions), the vectors are apparently dissimilar, so the coefficient of the formula of self-similarity is put to zero. If the match is perfect, the angle between the vectors is zero (the cosine equals exactly one), and we can estimate the coefficient of the formula, the amount one vector is bigger than the other, just by taking the ratio of their sizes:

$$coeff = \frac{\|v1\|}{\|v2\|} \qquad (2)$$

The determination of the coefficient becomes complicated in the situation where the match between two vectors is good (i.e., the cosine, Eq. (1), is positive) but not perfect (the cosine is smaller than one). One solution, which is actually implemented in the program, is to scale the 'ideal' coefficient, Eq. (2), down by the degree of the dissimilarity, i.e.:

$$coeff = \frac{\|v1\|}{\|v2\|}\cos(v1, v2) \tag{3}$$

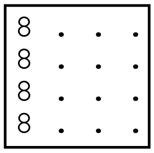
Note that one may want to scale down the coefficient even more if the match is not perfect; for example, one may multiply the ideal coefficient by the squared cosine (or use even higher degrees). Thus, given one 2x2 block of pixels and another block which is a quadrant of the larger 4x4 block (see Fig. 32), a scalar product of two image-vectors is computed for different rotations and/or reflections of the coarse resolution block, and the maximum is noted. The transformation that provides the maximum of the similarity is then encoded in x, the first component of the formula of self-similarity. The coefficient y of the formula is then estimated according to eqs. (3) and (1). Incidentally, since the multiresolutional pyramid is represented internally as a quadtree, the pixels of the bottom level are child nodes of the quadtree, and pixels of the higher levels are their parent and (great)grandparent nodes. One can hope that while the magnitude of father and son nodes may vary among different levels of the quadtree, the coefficient, eq. (3), a scaled quantity, would remain almost constant (at least, among the descendants of the patriarch that was used to evaluate the coefficient in the first place). This justifies 'prediction' procedures, which try to predict grandchildren and great-grandchildren based on the relation established between parents and children, somewhat like tracing down the lineage.

There are many ways for the image to be represented in the multiresolutional form. The first choice is a Gaussian pyramid, Fig. 29, a set of image views at progressively decreasing resolutions. The Laplacian pyramid and its various smooth modifications discussed in Chapter III should also be considered. Given below are results of the ‘smart’ expansion obtained with several different multiresolutional representations.

4.4 Smart Expansion of Simple Pictures

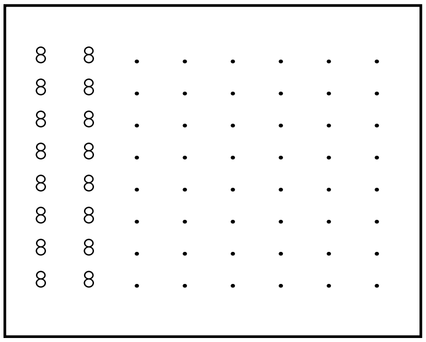
In the present section, we will show how the method works for simple images. Note that the figures throughout the section show the pixel value as the number rather than the shade of gray. For example, a pixel with value 8 is displayed as the number eight. A zero pixel is shown as a dot for clarity. The selected test images are indeed very simple, but one can consider them as a tiny part of a complex picture. Therefore, if expansion of the simple images seems indeed ‘smart’, one can expect good results for realistic images, too.

Figures 33 through 35 are segments of the straight line, which is the atomic part of any picture. In these examples, we are interested to see if the expansion of a *thin* line looks as thin rather than bloated. Thinness of the line is very important here, because if this property is preserved upon expansion, the magnified image would look as crisp and sharp as does the original. In Fig. 34, which is a shifted version of Fig. 33, we are also looking at whether the property of keeping thin lines thin upon magnification is preserved through translation, the bane of the pyramidal transforms [MALL89]. As one can see, surprisingly it does. This result is indeed astonishing since the reduced resolution representations can be severely distorted when the input is shifted. The problem is known in the quadtree literature as the shift-dependence of the description. In the worst case, a one-pixel shift of the input image can lead to a significantly modified pyramid structure [see refs. in MONT91]. As it turns out, even though the multiresolution decomposition is altered by the shift, it still preserves the property of self-similarity.



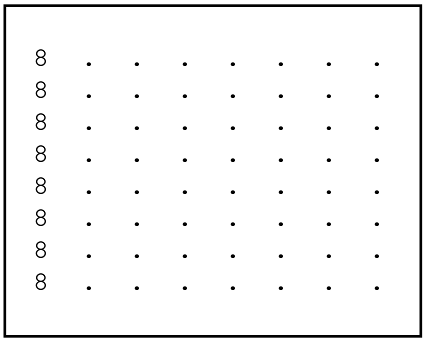
```
8 . . .
8 . . .
8 . . .
8 . . .
```

Fig. 33a. Original image.



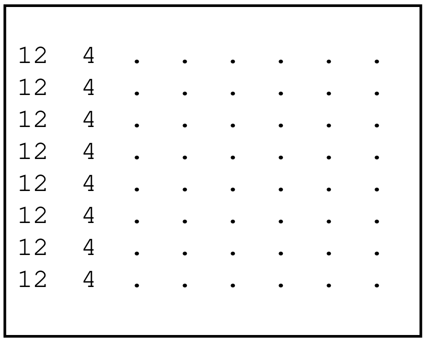
```
8  8  .  .  .  .  .  .
8  8  .  .  .  .  .  .
8  8  .  .  .  .  .  .
8  8  .  .  .  .  .  .
8  8  .  .  .  .  .  .
8  8  .  .  .  .  .  .
8  8  .  .  .  .  .  .
8  8  .  .  .  .  .  .
```

Fig. 33b. Dumb expansion.



```
8  .  .  .  .  .  .  .
8  .  .  .  .  .  .  .
8  .  .  .  .  .  .  .
8  .  .  .  .  .  .  .
8  .  .  .  .  .  .  .
8  .  .  .  .  .  .  .
8  .  .  .  .  .  .  .
8  .  .  .  .  .  .  .
```

Fig. 33c. Smart expansion based on Gaussian pyramid.



```
12  4  .  .  .  .  .  .
12  4  .  .  .  .  .  .
12  4  .  .  .  .  .  .
12  4  .  .  .  .  .  .
12  4  .  .  .  .  .  .
12  4  .  .  .  .  .  .
12  4  .  .  .  .  .  .
12  4  .  .  .  .  .  .
```

Fig. 33d. Smart expansion based on Laplacian pyramid.

```
13  3  .  .  .  .  .  .
13  4  .  .  .  .  .  .
12  3  .  .  .  .  .  .
13  3  .  .  .  .  .  .
13  3  .  .  .  .  .  .
12  3  .  .  .  .  .  .
13  4  .  .  .  .  .  .
13  3  .  .  .  .  .  .
```

Fig. 33e. Smart expansion based on smooth Laplacian pyramid.

```
. 8 . .
. 8 . .
. 8 . .
. 8 . .
```

Fig. 34a. Original image.

```
.  .  8  8  .  .  .  .
.  .  8  8  .  .  .  .
.  .  8  8  .  .  .  .
.  .  8  8  .  .  .  .
.  .  8  8  .  .  .  .
.  .  8  8  .  .  .  .
.  .  8  8  .  .  .  .
.  .  8  8  .  .  .  .
```

Fig. 34b. Dumb expansion.

```
.  .  8  .  .  .  .  .
.  .  8  .  .  .  .  .
.  .  8  .  .  .  .  .
.  .  8  .  .  .  .  .
.  .  8  .  .  .  .  .
.  .  8  .  .  .  .  .
.  .  8  .  .  .  .  .
.  .  8  .  .  .  .  .
```

Fig. 34c. Smart expansion based on Gaussian pyramid.

```
.  . 12  4  .  .  .  .
.  . 12  4  .  .  .  .
.  . 12  4  .  .  .  .
.  . 12  4  .  .  .  .
.  . 12  4  .  .  .  .
.  . 12  4  .  .  .  .
.  . 12  4  .  .  .  .
.  . 12  4  .  .  .  .
```

Fig. 34d. Smart expansion based on Laplacian pyramid.

```
 .   . 13   3   1   .   .   .
 .   . 13   4   1   .   .   .
 .   . 12   3   1   .   .   .
 .   . 13   3   1   .   .   .
 .   . 13   3   1   .   .   .
 .   . 12   3   1   .   .   .
 .   . 13   4   1   .   .   .
 .   . 13   3   1   .   .   .
```

Fig. 34e. Smart expansion based on smooth Laplacian pyramid.

```
8 . . .
. 8 . .
. . 8 .
. . . 8
```

Fig. 35a. Original image.

```
 8   8   .   .   .   .   .   .
 8   8   .   .   .   .   .   .
 .   .   8   8   .   .   .   .
 .   .   8   8   .   .   .   .
 .   .   .   .   8   8   .   .
 .   .   .   .   8   8   .   .
 .   .   .   .   .   .   8   8
 .   .   .   .   .   .   8   8
```

Fig. 35b. Dumb expansion.

```
 8   .   .   .   .   .   .   .
 .   8   .   .   .   .   .   .
 .   .   8   .   .   .   .   .
 .   .   .   8   .   .   .   .
 .   .   .   .   8   .   .   .
 .   .   .   .   .   8   .   .
 .   .   .   .   .   .   8   .
 .   .   .   .   .   .   .   8
```

Fig. 35c. Smart expansion based on Gaussian pyramid.

```
12   4   .   .   .   .   .   .
 4  12   .   .   .   .   .   .
 .   .  12   4   .   .   .   .
 .   .   4  12   .   .   .   .
 .   .   .   .  12   4   .   .
 .   .   .   .   4  12   .   .
 .   .   .   .   .   .  12   4
 .   .   .   .   .   .   4  12
```

Fig. 35d. Smart expansion based on Laplacian pyramid.

```
13  3  .  .  .  .  .  .
 3 13  1  .  .  .  .  .
 .  1 13  3  1  .  .  .
 .  .  3 13  2  1  .  .
 .  .  1  2 13  3  .  .
 .  .  .  1  3 13  1  .
 .  .  .  .  .  1 13  3
 .  .  .  .  .  .  3 13
```

Fig. 35e. Smart expansion based on smooth Laplacian pyramid.

```
8 10 12 14
8 10 12 14
8 10 12 14
8 10 12 14
```

Fig. 36a. Original image.

```
8  8 10 10 12 12 14 14
8  8 10 10 12 12 14 14
8  8 10 10 12 12 14 14
8  8 10 10 12 12 14 14
8  8 10 10 12 12 14 14
8  8 10 10 12 12 14 14
8  8 10 10 12 12 14 14
8  8 10 10 12 12 14 14
```

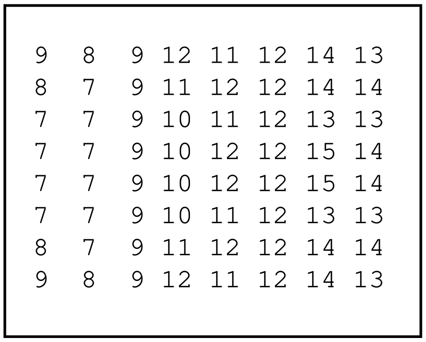
Fig. 36b. Dumb expansion.

```
6  7  9 10  8 10 12 14
6  7  9 10  8 10 12 14
6  7  9 10  8 10 12 14
6  7  9 10  8 10 12 14
6  7  9 10  8 10 12 14
6  7  9 10  8 10 12 14
6  7  9 10  8 10 12 14
6  7  9 10  8 10 12 14
```

Fig. 36c. Smart expansion based on Gaussian pyramid.

```
7  8  9 10 11 12 13 14
7  8  9 10 11 12 13 14
7  8  9 10 11 12 13 14
7  8  9 10 11 12 13 14
7  8  9 10 11 12 13 14
7  8  9 10 11 12 13 14
7  8  9 10 11 12 13 14
7  8  9 10 11 12 13 14
```

Fig. 36d. Smart expansion based on Laplacian pyramid.



```
9  8  9 12 11 12 14 13
8  7  9 11 12 12 14 14
7  7  9 10 11 12 13 13
7  7  9 10 12 12 15 14
7  7  9 10 12 12 15 14
7  7  9 10 11 12 13 13
8  7  9 11 12 12 14 14
9  8  9 12 11 12 14 13
```

Fig. 36e. Smart expansion based on
smooth Laplacian pyramid.

Figure 36 presents the results for a stripy image with a horizontal gradient fill. Note that the smart expansion based on the Gaussian pyramid does not perform well. This illustrates the situation when two layers of the pyramid look very similar yet have different background brightness. This may happen because a layer of the Gaussian pyramid represents the entire image rather than local contrast at a particular resolution. Obviously offsetting a vector (a block of pixels) changes its norm, which interferes with the determination of the coefficient of the self-similarity formula; see eq. (3). This problem does not arise for the Laplacian pyramid, where all levels represent only the local contrast, and therefore, the background brightness (or average intensity over a 2x2 block of pixels) for any layer stays very close to zero. One astounding feature of the smart image expansion based on the Laplacian pyramid is its ability to reproduce an utterly perfect, smooth gradient fill upon magnification. Indeed, Fig. 36d shows gradual smooth transitions from one stripe to another, which is precisely how a high-resolution picture of the gradient fill is supposed to look. The results for the smooth Laplacian pyramid might look less ‘perfect’; but one should keep in mind that the decomposition performs smoothing by definition, which is apparent in the realistic images below.

Since our method of expansion is based on the self-similarity, it ought to work perfectly for ‘classical’ self-similar images. To demonstrate that this is really the case, we chose one such picture, a Sierpinski gasket [MAND83, FISH92]. All the various implementations of the smart image magnification we have discussed earlier have expanded Fig. 37a into Fig. 37c, which is a correct view of the gasket at the finer resolution.

```
1 . . .
1 1 . .
1 . 1 .
1 1 1 1
```

Fig. 37a. Sierpinski gasket.

```
1  1  .  .  .  .  .  .
1  1  .  .  .  .  .  .
1  1  1  1  .  .  .  .
1  1  1  1  .  .  .  .
1  1  .  .  1  1  .  .
1  1  .  .  1  1  .  .
1  1  1  1  1  1  1  1
1  1  1  1  1  1  1  1
```

Fig. 37b. Dumb expansion.

```
1  .  .  .  .  .  .  .
1  1  .  .  .  .  .  .
1  .  1  .  .  .  .  .
1  1  1  1  .  .  .  .
1  .  .  .  1  .  .  .
1  1  .  .  1  1  .  .
1  .  1  .  1  .  1  .
1  1  1  1  1  1  1  1
```

Fig. 37c. Smart expansion.

4.5 Smart Expansion of Realistic Images

The first example to which we apply our method of expansion is ‘lenna’, a famous image of a lady before the mirror. It is a very complicated image including both smooth regions and areas with many tiny high-frequency details. The picture has become an unofficial standard in image processing. Virtually every paper that proposes a new technique in image compression demonstrates it first for that image. The picture even appeared on the cover of several books on image analysis and compression. Fig. 38a is an original image, Fig. 38b displays a shrunken/dumb expanded one. It is very easy to see the staircase view which is so typical of dumb expanded images. Fig. 38c gives the result of the smart expansion using the multiresolution decomposition based on a regular Laplacian pyramid. Finally, Fig. 38d is obtained using the smooth Laplacian pyramid. Figures 39a-d demonstrate the same set of expansion methods for an IR satellite image of a part of an ocean covered with clouds, and Figs. 40a-d do the same for a frame of a weather video broadcast.

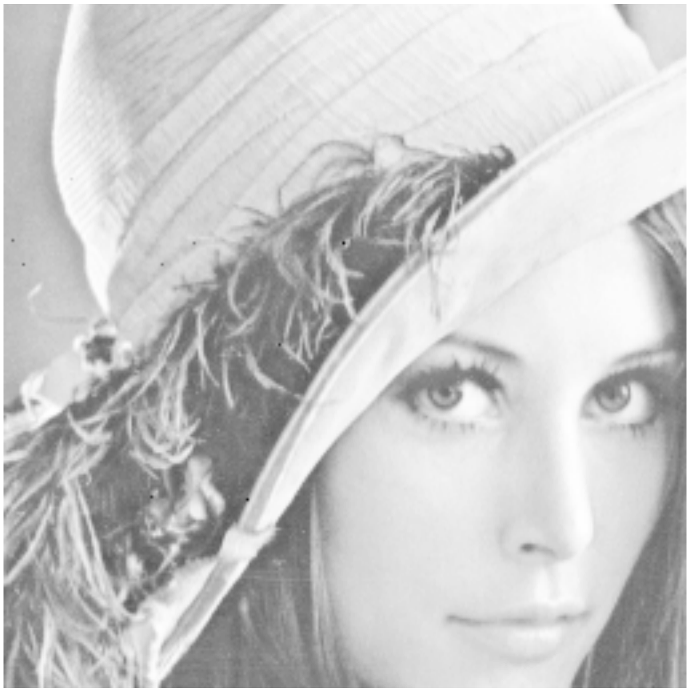


Fig. 38a. A central part of the original lenna. Printed at scale 1:1 after the γ-correction with the factor 1.85.

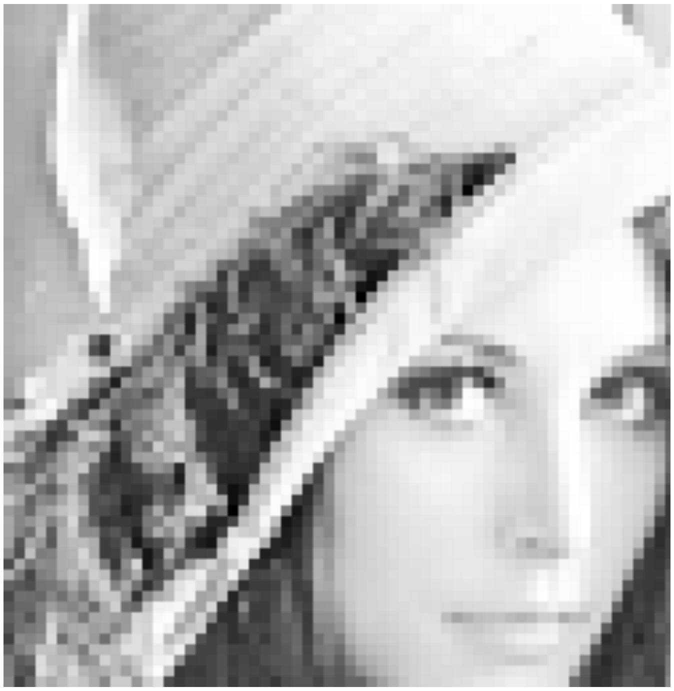


Fig. 38b. Shrunken/dumb expanded.

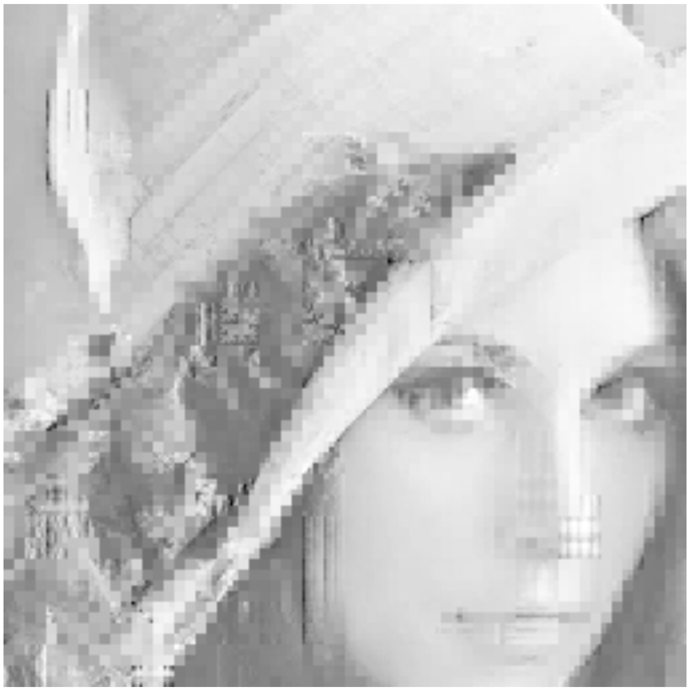


Fig. 38c. Smart expansion using the Laplacian pyramid.

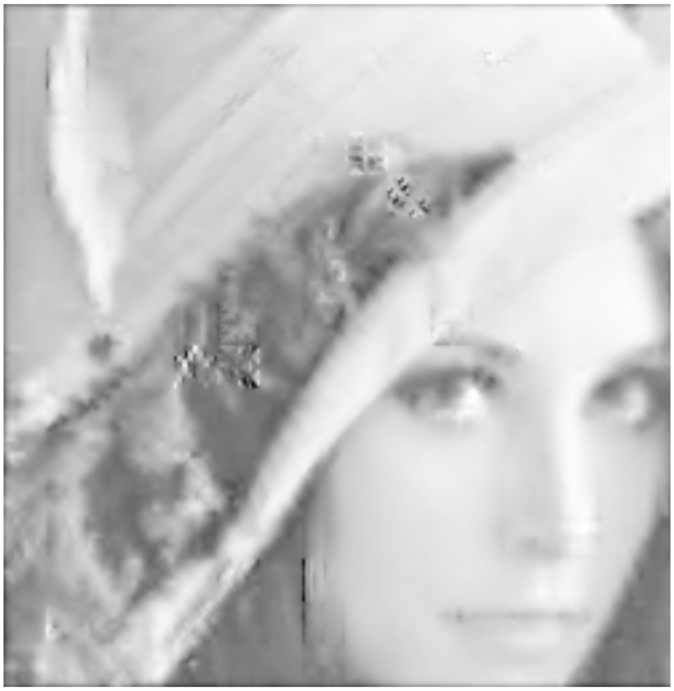


Fig. 38d. Smart expansion using the smooth Laplacian pyramid.

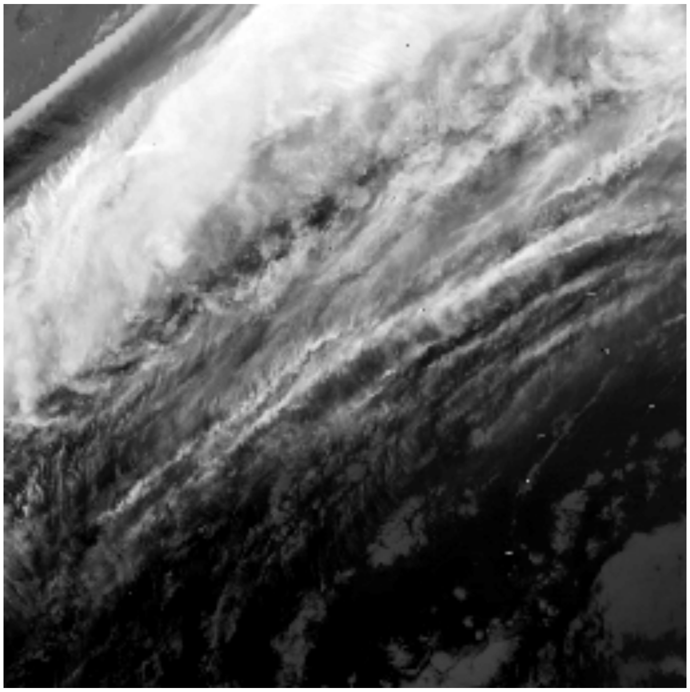


Fig. 39a. Original Clouds. Printed at scale 2:1 after the contrast enhancement of the inverted LUT.

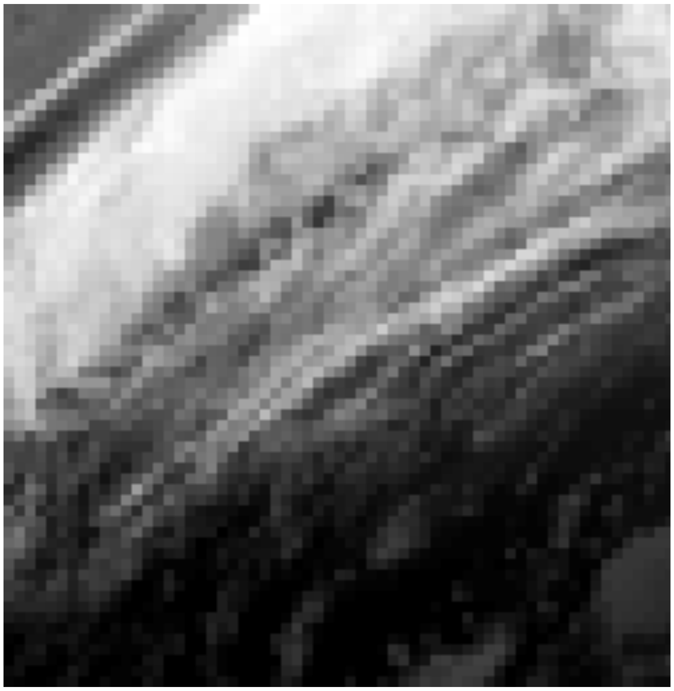
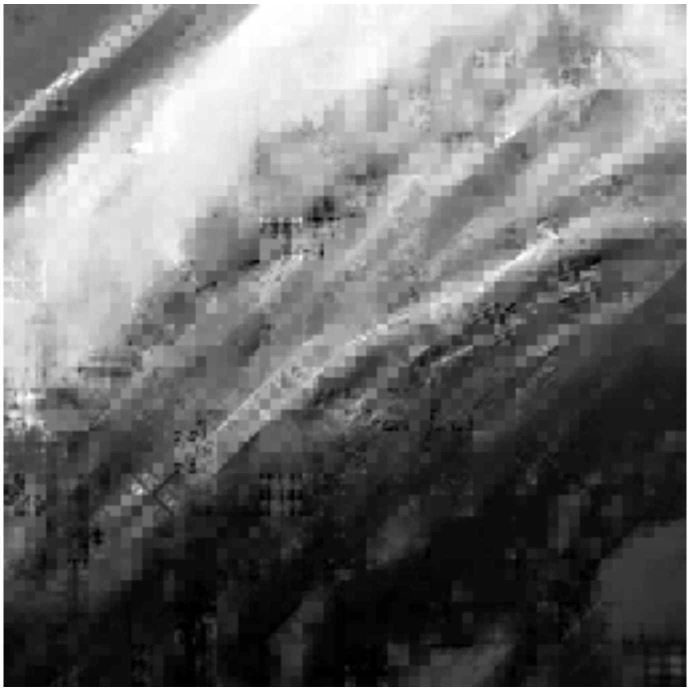


Fig. 39b. Shrunken/dumb expanded.



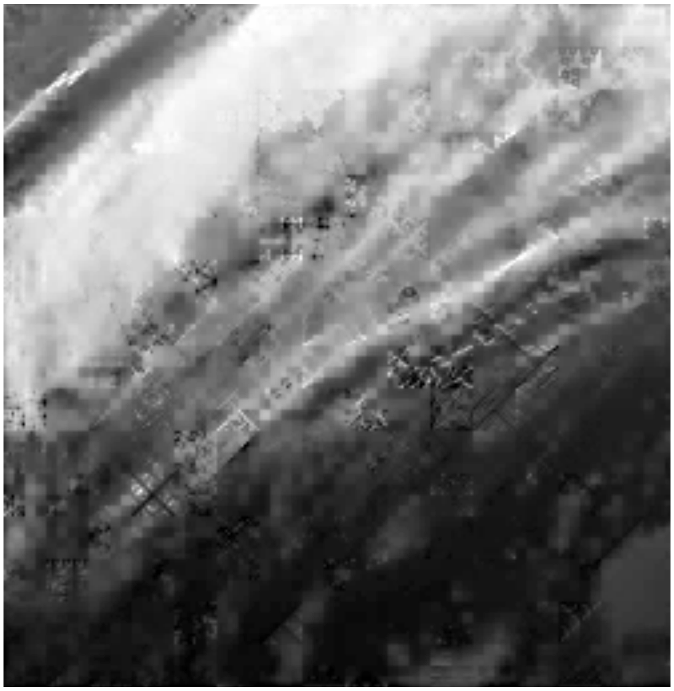


Fig. 39c. Smart expansion using the Laplacian pyramid.

Fig. 39d. Smart expansion using the smooth Laplacian pyramid.

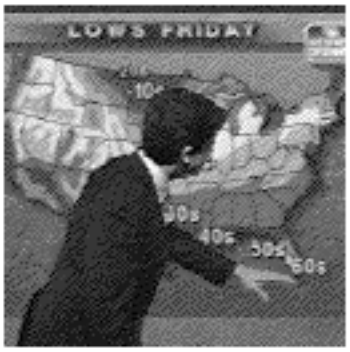


Fig. 40a. Original video frame of a weather briefing.

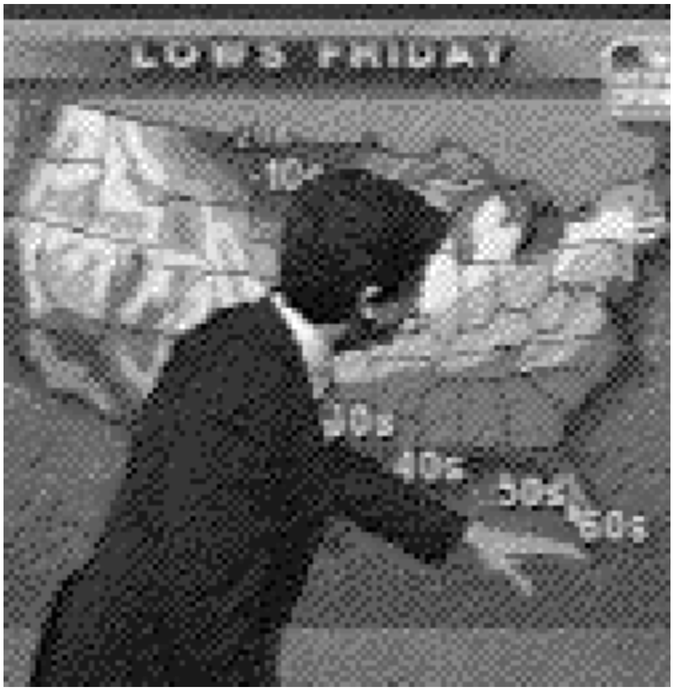


Fig. 40b. Dumb expanded x2.

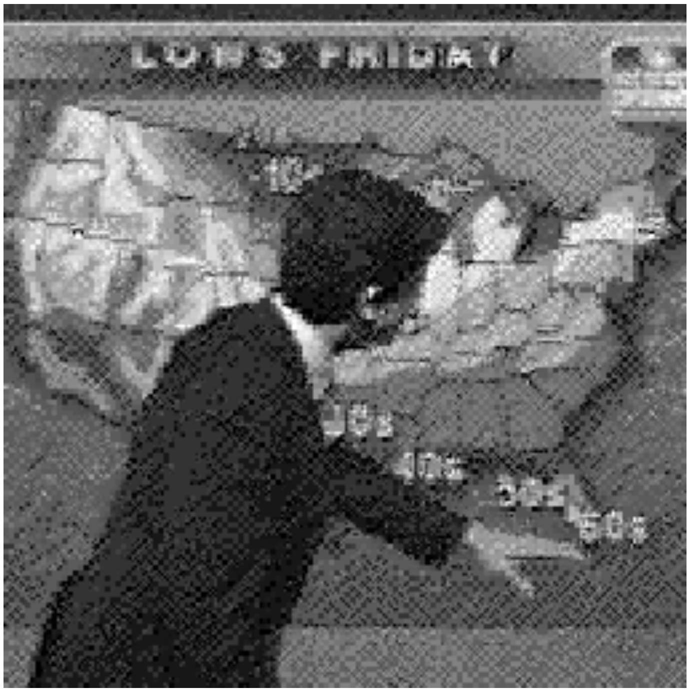


Fig. 40c. Smart expansion using the Laplacian pyramid.

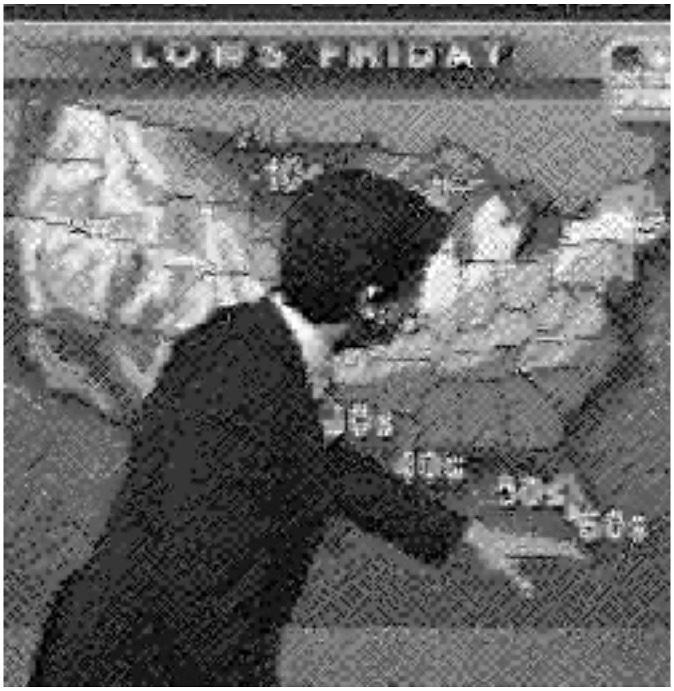


Fig. 40d. Smart expansion using the smooth Laplacian pyramid.

CHAPTER V

SMART MAGNIFICATION OF MOVING PICTURES

5.1 Objectives of the Search for Multiresolutional Self-similarity in Moving Pictures

In the previous chapter, we have discussed a fractal image magnification based on self-similarity of the multiresolutional image decomposition. The central idea of the approach is the similarity between a part and a whole; in our case, between an image at some resolution and parts of it at a finer resolution. That similarity allowed us to extend the multiresolutional pyramid in depth, thus producing image views at finer and finer resolutions. On the other hand, zooming out of the picture implies that one has to fill in gaps between the magnified features, i.e., to perform interpolation. It has been demonstrated that the method of ‘smart’ magnification produces a more ‘natural looking’ magnified image.

In the present chapter, the approach is focused on video imagery. Now, we are given not just a single image but an ordered sequence of frames, which are snapshots of some time-varying scenery. The challenge now is to take into account the third dimension, time. The smart magnification problem extends now to zooming out a given sequence of frames both in space and in time. In other words, the issue here is how to magnify a separate frame to discern greater detail as well as increase the time resolution to produce smoother motion. Note that an alternative approach might be to perform time extrapolation rather than interpolation. The extrapolation may appear more promising in that it would then allow one to predict the position of moving details in subsequent frames based on few initial frames. On the other hand, the time interpolation also has its merits. In fact, the minimum frame rate of around 6-8 frames a second [HAIN93] is justified mainly not from the informational, but from that psychophysiological point of view. In fact, chopped motion, large area flicker (which is especially noticeable on large and bright picture areas) and other visible artifacts of the moving picture are very disturbing to the viewer and impair his/her ability to judge the scene [HAIN93]. Bandwidth restrictions generally do not allow for a high field rate used in the transmission, leaving the elimination of the display flicker to the receiver. So, if it is possible to fill in between two adjacent frames by performing the interpolation of motion, one can accommodate the viewer without sacrificing the bandwidth (the bit rate). Thus, the goal is not only to eliminate blockiness within the frame, but the chopped motion and resulting flicker between frames.

A multiresolutional decomposition of the frame sequence provides a foundation for the smart magnification of moving pictures, as it does for still images. In our explanation, we will closely follow the path of Chapter IV, but perform all the work in three dimensions. The subsequent sections will show that it is possible to expand the image sequence in all three dimensions, producing crisp, sharp looking individual frames as well as smooth motion between the adjacent frames. A program has been written that constructs the multiresolutional frame sequence decomposition, determines the formula of local self-similarity and uses it to perform the frame sequence expansion. Sample results of the program will be demonstrated and discussed below. The examples used in the discussion are rather simple (though capturing the essential characteristics of typical moving pictures), as it is unfeasible to include pictures of all the frames of a real movie in the document. However, a real-life example of compressing and decompressing a clip of a weather briefing is presented on a demonstrational video tape (available on request).

5.2 A Simple Example of the Self-similarity in the Multiresolutional Frame Sequence Decomposition

Similarly to the two-dimensional case, we start the discussion of self-similarity in the spatial-temporal multiresolutional decomposition by examining the following test frame sequence, which is composed of snapshots of a single moving vertical line. Each small square in Fig. 41 stands for a pixel of a 4x4 image.

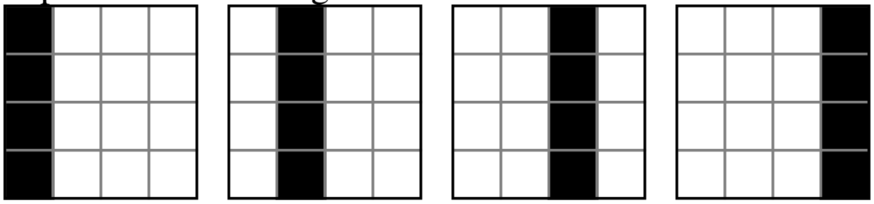


Fig. 41. A simple self-similar test frame sequence.

By decreasing the resolution twice in each dimension (both in space and in time), the following sequence is obtained:

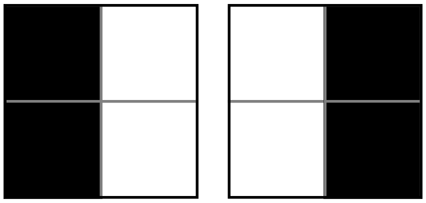


Fig. 42. A low resolution frame sequence.

Reducing the resolution even more results in a single black pixel. Stacking the frame sequences one upon another gives us a Gaussian pyramid, a set of views of the original frame sequence at different resolutions. Note that a pixel of a frame sequence is represented by a small cube rather than square, which emphasizes the idea of the third, temporal dimension of the moving picture.

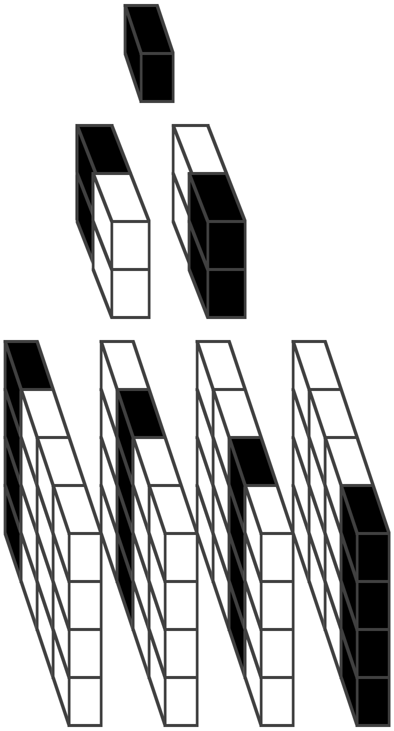


Fig. 43. Gaussian pyramid for the test frame sequence.

The bottom of the pyramid is the original sequence itself, Fig. 41. Located above it is the same sequence but at the resolution 2:1, Fig. 42, which has half the number of frames at half the size. As was already mentioned, reducing the resolution twice in all dimensions turns the middle layer to a single pixel, the tip of the pyramid. Incidentally, the layout of the pyramid illustrates clearly the way the resolution is reduced. The middle layer consists of 8 pixels arranged in a 2x2x2 cube. Four pixels of the one face of the cube belong to one frame and the other 4 pixels comprise the corresponding 2x2 square of pixels of the next frame. The process of resolution reduction replaces all these 8 pixels by a single one, with the average intensity. This new pixel can be considered a ‘parent’ of the 8 original pixels, since it describes the same part of the moving picture as do they, but it does so at twice as coarse resolution. Indeed, averaging over 2x2 squares of a frame leaves only large-scale features intact, while averaging over adjacent frames eliminates fine changes between the frames; in other words, averages out a small-scale motion. The nodes of the pyramid along with their ancestral links constitute a tree. Each node (except the top one) of a tree has one parent, and each node (except the leaf) has exactly 8 offsprings. Incidentally, this is the reason the tree is called octtree.

Another multiresolutional image representation would be to take the difference between two adjacent levels of the Gaussian pyramid, which leads to a Laplacian pyramid. In precise terms, the algorithm for constructing the Laplacian pyramid is to subtract the value of a parent node from each of its 8 children. Therefore, a node of the pyramid tells how this particular piece of the moving picture stands out from its local background at a given resolution. In other words, each level of the Laplacian pyramid represents the features that are visible just at this particular resolution. Coincidentally, the top level of the Laplacian pyramid is the same as that of the Gaussian one. The following figure depicts the Laplacian pyramid for the test sequence, Fig. 41. Cross-hatched cubes on the picture stand for the pyramidal nodes with the value of minus one.

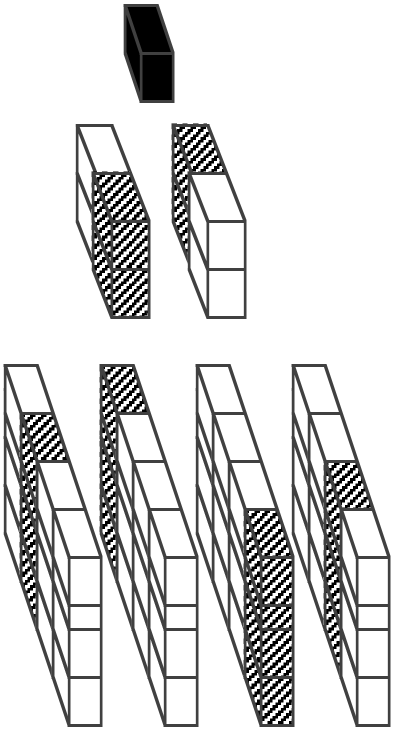


Fig. 44. Laplacian pyramid for the test frame sequence.

From the point of view of estimating the formula of self-similarity, the advantage of the Laplacian pyramid is that subtracting the value of the parent node from a pyramid node removes the local background. It really makes the difference as far as the estimation of self-similarity is concerned. Consider, for example, two sets of nodes of the Gaussian pyramid that look almost the same except one set is a bit brighter than the other. The present algorithm for estimating the formula of self-similarity, outlined in Chapter IV, would fail to recognize the resemblance of the two sets of pixels. However, after the background is removed, the sets become identical, and their similarity is easy to detect.

As was mentioned before, the multiresolutional pyramid separates features of the original frame sequence, Fig. 41, according to their scale. Each level of the pyramid contains the characteristics of the entire frame sequence at a particular resolution. Though the scale and features of two pyramid layers are all different, there is a striking similarity between them. Moreover, *parts* of the finer resolution layer appear similar or identical to the *entire* coarse-resolution layer. We have met the similarity of that type before while discussing fractal images. Now we see that certain image sequences also possess the self-similarity of the same kind. As in the two-dimensional case, the property of self-similarity can be expressed in the form of a precise algorithm that tells how a frame sequence viewed at some resolution can be arranged such that it produces a frame sequence view at a higher resolution in space and in time.

The formula of the self-similarity in the three-dimensional case can be derived then as follows: The bottom level of the pyramid on Fig. 44 describes the frame details and the motion that are discernible at the resolution 1:1; the level just above it contains larger scale details and refers to a coarser motion, corresponding to the resolution 2:1. Obviously, the former sequence is 8 times bigger than the latter. The high resolution sequence can be split into eight octants, each of them having exactly the same size (dimensions) as the entire low resolution sequence, so that it is possible to consider a mapping between the latter and the octants of the former. These mappings define a formula of the self-similarity of the moving picture, which in our case looks like—

upper left back octant of the high-res seq. = 1 · low resolution seq.
lower left back octant of the high-res seq. = 1 · low resolution seq.
upper left front octant of the high-res seq. = 0 · low resolution seq.
lower left front octant of the high-res seq. = 0 · low resolution seq.
upper right back octant of the high-res seq. = 0 · low resolution seq.
lower right back octant of the high-res seq. = 0 · low resolution seq.
upper right front octant of the high-res seq. = 1 · low resolution seq.
lower right front octant of the high-res seq. = 1 · low resolution seq.

Consequently, the formula above is a unique description of the sequence, and captures all its essence. It opens up a possibility of expanding the sequence to create a moving picture at even finer resolution with even more incremental motion. Indeed, applying the formula to the bottom level of the pyramid on Fig. 44 produces—

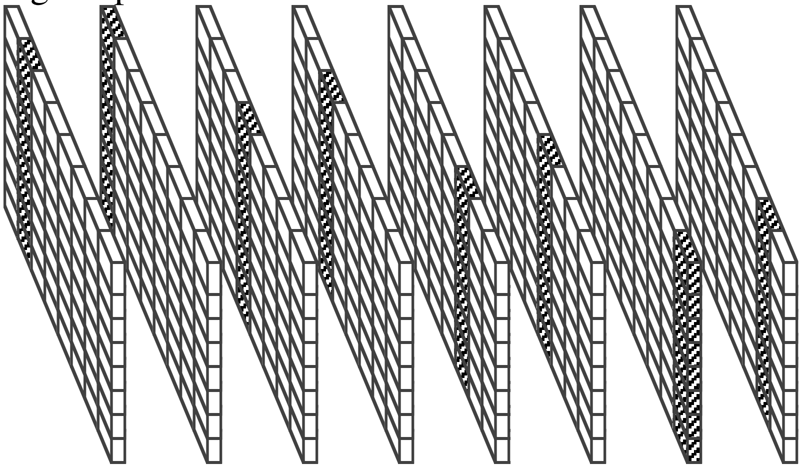


Fig. 45. New level of the Laplacian pyramid obtained by the application of the formula of the self-similarity.

Converting the extended Laplacian pyramid (appended with the layer in Fig. 45) back to the Gaussian one and taking the bottom level of it gives us a magnified original sequence:

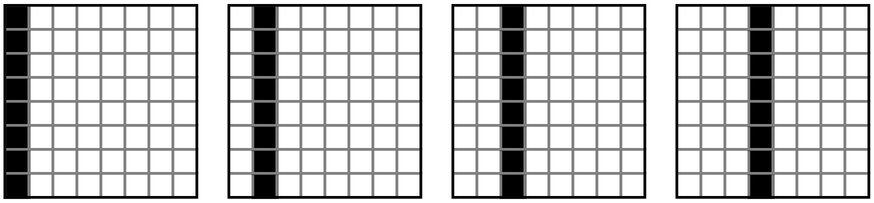


Fig. 46. Magnified image sequence obtained using the formula of self-similarity.

Figure 47 displays for comparison the frame sequence obtained by the ‘dumb’ magnification:

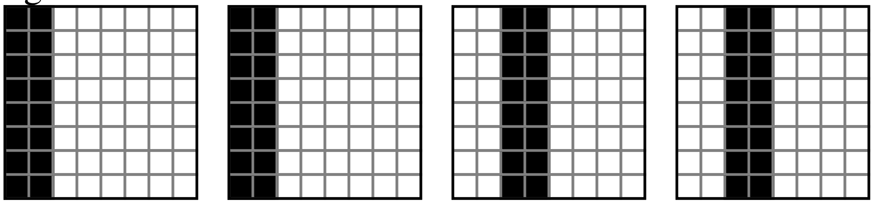


Fig. 47. "Dumb" magnified image sequence.

The moving line remains thin upon the ‘smart’ magnification, a critical feature of the multiresolutional fractal magnification that we have already seen in the case of still pictures. However, it should be noted that the movement of the line between the snapshots is smooth and even flowing. It indicates that the algorithm is powerful enough to permit the motion prediction, which is a very important result here.

In general, the relation between the frame sequence and an octant of itself at a finer resolution may be more complicated than in the simplest case shown above. Rotation and/or scaling may be needed to match the cube and the octant, just as in the two- dimensional case. In addition, rotations relative to the third axis (time) are now possible, though we have not examined this possibility. As in the two-dimensional case, the formula of the self-similarity is specified by a pair of numbers. The first number is a code of the rotation/reflection operation upon one sequence necessary to match it with the octant of the other. The other number of the pair, a coefficient, tells the intensity ratio between the frame sequences under consideration. In the three-dimensional case, we are also content with a ‘good’ match, not necessarily the perfect one. The method for obtaining the match is the same as that discussed in detail in Chapter IV.

5.3 Local Self-similarity and ‘Smart’ expansion

Realistic moving pictures, which are our concern here, are a bit more complicated than the test example, Fig. 41. Hardly can one expect that, say, all frames of the weather broadcast video can be described entirely by a single simple formula as we have discussed earlier. However, we believe that a small separate part of the moving picture does lend itself to the self-similarity analysis; i.e., the self-similarity property does hold *locally*, locally in space and in time.

At present, we have looked into the application of the self-similarity to the ‘smart’ expansion of frame sequences as sketched above. The algorithm looks almost the same as that for still images. It is repeated here for completeness. Given a moving picture, simple or realistic, it is reduced 2 times in each dimension, 8 times total. We construct different kinds of multiresolutional representations for the shrunken sequence, and estimate the local self-similarity between the bottom level of the pyramid and the level just above it. The formulas of self-similarity are then used to predict a new layer of the multiresolutional pyramid, one level deeper than the bottom level. Thus, the multiresolutional pyramid of the twice-shrunken frame sequence is extended by one level in depth. From that extended pyramid, we can reconstruct a sequence, which is twice as big in each dimension as compared to the shrunken sequence. Thus, we shrank the original moving picture 8 times and then expanded it 'smartly' to the original size. Comparison of the two frame sequences, original and shrunken/expanded, then gives an idea of how good the 'smart' expansion is. In the following section we present results of the program for some relatively simple sample moving pictures.

5.4 Smart Expansion of Simple Moving Pictures

The example of the movie we will consider is a translation of a simple object against the static background. Four consecutive frames of the sample movie are presented below:

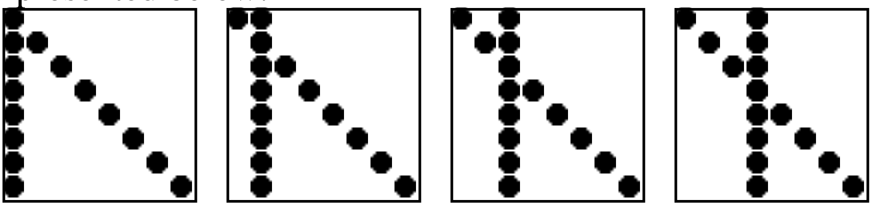


Fig. 48. Frames of the sample movie.

The following figure shows the results of the smart magnification of the sequence; only first 5 frames are shown, but it is easy to imagine the rest.

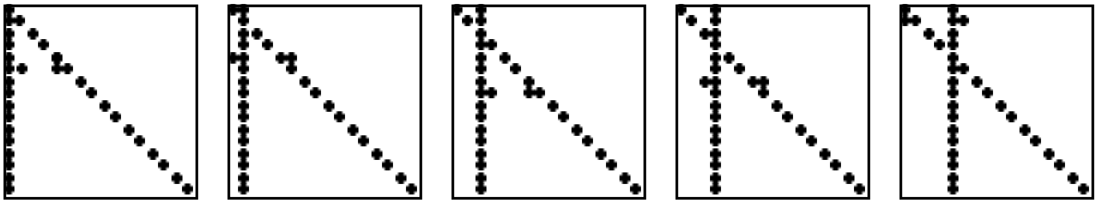


Fig. 49. Several consecutive frames of the predicted magnified sample movie.

Another example of smart magnification is presented in Figs. 50 and 51. Unlike the previous example, the translated object is relatively small and compact and does not span across the frame. Moreover, it is translated diagonally rather then horizontally.

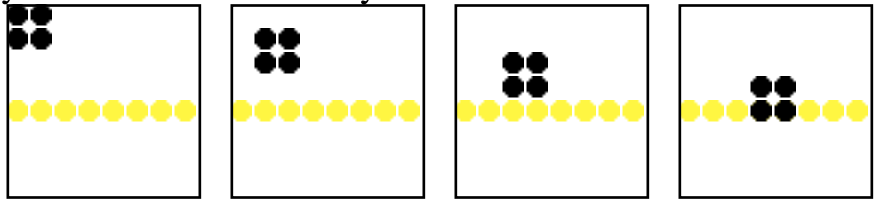


Fig. 50. Frames of the sample movie.

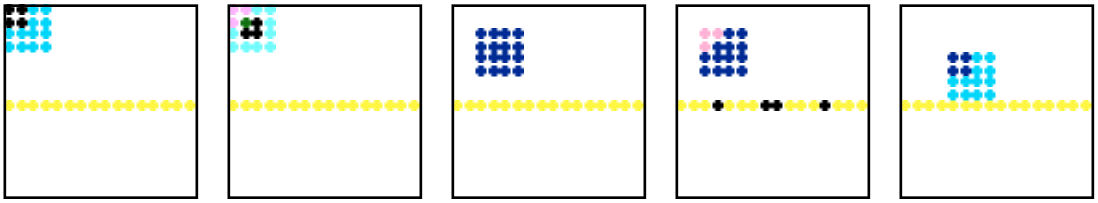


Fig. 51. Several consecutive frames of the predicted magnified sample movie, Fig. 50.

Though the predictions for the high resolution sequence are not exact, they are quite close to what one would expect. It should be noted here that a prediction does not have to be perfect, as one can always encode the error separately. If the prediction is quite accurate, the error will be small and can be transmitted at a low bit rate. As shown in Fig. 51, this is the case here. It must be stressed that the motion in the predicted frame sequence is indeed small and smooth as it should be.

CHAPTER VI

CONCLUSIONS

New data compression technologies which yield high-ratio data compression while maintaining the fidelity of raw data are sought everywhere by researchers, by agencies of the Federal Government and by industry. Potential applications range from on-board data reduction in the Space Exploration Program and disseminating weather and other types of imagery between a variety of different computer systems spread across thousands of miles, to compressing fingerprints and pictures of ID holders for efficient storage in databases. The present work has outlined a prototype technology, a multiresolutional image analysis and pyramidal decomposition, that lends itself to all these applications, and, in fact, has proved to be feasible.

The quest for the minimal entropy pyramid is by no means complete. In the present work, we have summarized several basic principles to follow in search for the less (and least) redundant pyramidal image representation. We have given and thoroughly discussed several designs of the Laplacian pyramid, which clearly demonstrate how the principles can be applied and what tradeoffs are involved. The discussion would help in developing new pyramidal representations of images, e.g., based on the filters with broader margins, etc. From the practical point of view, the methods of the wavelet image compression designed in the present work offer a variety of lossy image compression schemes with different properties; each particular algorithm preserves certain features better than the others at the same compression ratio. The performance of the schemes is illustrated in Chapter III on a large number of examples of compressed and restored sample images of various kinds. The examples clearly demonstrate what is preserved and what is lost during the lossy compression and should help an informed user choose the scheme that best suits his requirements.

A non-uniform compression has emerged as a powerful technique that is crucial to a variety of applications like disseminating weather information over the broad area of users or space imagery communications over low bandwidth communication channels. The non-uniform compression allows for a significant compression with a high fidelity of useful information, as features of little interest or known to be distorted by noise during the data acquisition would not be transmitted at all. Note that in this guise, the non-uniform compression is related to a (generally nonlinear) filtration, which unlike traditional schemes is more flexible and allows for greater control on the part of the end user. Chapter III has presented one of the first implementations of the non-uniform wavelet image compression. It has a particular feature of seamless and smooth incorporation of almost non-distorted information within a broader context of large-scale features. Formulation of the criteria sets allows for the further development of the non-uniform compression of areas of a picture into a non-uniform compression of arbitrary image features that are of interest to a particular user.

One formulation of the non-uniform compression relates to a progressive transmission of information. Spatial-frequency localization of the wavelet transform coefficients and their layout in a quadtree are vital parts of the technique. An example would be an interactive transmission of a satellite image of a coastal region. The image itself can be stored in the data processing center in a decomposed (transform) form, which is just another exact representation of the image and allows perfect reconstruction. A user at a remote site may want to request first only a coarse view of the picture, which involves communicating only a few coefficients of the decomposition. In fact, our experiments demonstrate that it takes merely 0.5% of the total number of the decomposition coefficients to get a fairly good coarse resolution representation. Having received the coarse view, the user can select those areas of the coast which interest him, and more coefficients will be delivered describing those selected areas in more detail. Since the user now is able to see medium-size features (e.g., the relief) within the selected areas, he may want to narrow his choice. More coefficients will be sent in from the data processing center. Ultimately, the user will be able to see small selected areas with as good a resolution as the original image, while having the context(s) rendered at coarser resolutions at his wish. Note that the total amount of information communicated is only a fraction of that of the entire original image. All the transmitted transform coefficients are used to reconstruct the image to the extent and detail the user desires, and no communicated information is wasted.

In automatically selecting features and areas of an image for the non-uniform compression, a discrete derivative of the image can be useful. The regularized discrete derivative discussed in Chapter III effectively removes the local background and the fine-scale noise; therefore, it may be used for localizing image patterns regardless of the brightness of the picture, lighting conditions, etc. Finally, the derivative of an image can have value by itself, as in interpreting the IR satellite imagery.

The present work has embarked on the study of self-similarity of the wavelet image transform and combined fractal/ multiresolutional image compression. We have outlined only a bare idea of the approach and demonstrated some startling properties as the preservation of the self-similarity upon translation. It has opened up an entirely new approach to compression: zooming out a (possibly shrunken) low-resolution picture producing a sharp and crisp ‘natural looking’ high-resolution view without blockiness and jaggedness. The smart image magnification based on the self-similarity of the multiresolutional pyramidal transform has been shown to preserve thinness of lines on expansion and provide a perfect high-resolution representation of the gradient fill. The method is moderately successful in magnifying real images. Although the study of the multiresolutional self-similarity has just begun, the achieved results look encouraging and clearly warrant further investigation.

Another feature of the present research is the generalization of the multiresolutional pyramidal approach to deal with moving pictures. We have developed and experimented with algorithms of wavelet pyramidal decomposition of a three-dimensional spatio-temporal frame sequence. Moreover, we have discovered that the transform coefficients arranged in the form of a four-dimensional octpyramid possess the property of the self-similarity that we observed for the pyramidal decomposition of still images. This leads to the idea of the ‘smart’ magnification both in space and in time, which allows one to restore fairly accurately a high resolution video sequence from the low resolution one. This concept permits rapid compression, and has potential for use in video compression in real time. Note that the algorithm for estimating the formula of local self-similarity implicitly involves a non-uniform compression. However, we believe that the algorithms and techniques we have developed for the non-uniform compression of still images will be even more successful when applied to moving pictures.

A tangible result of the research is the software that has been developed to handle images and image sequences, quadtrees and octtrees, carry out a variety of image decompositions and experiment with them, communicate imagery information between different computers (Macintosh and UNIX server) over the network using the TCP protocol. Although some code is specific to a pyramidal image decomposition, the bulk of it has a broad applicability and can be used in any application dealing with image and image sequences. The object-oriented structure of the software makes it easy to experiment with and develop new versions of the image processing and compression techniques.

Another accomplished result is an engineering prototype of the image and video communication over lower capacity channels, such as regular telephone lines. The prototype can be developed to produce a system which is as simple to use as the telephone, and would provide the shared working environment for individuals involved in any joint projects. The commercial market for this technology is unlimited since every firm that deals with remote offices or locations is a potential user of such facilities. With the availability of cellular phones, the portable or notebook computer will provide enough computational power to allow video communication outside of hard wire connections. Now the office at home concept can be fully realized even in today's limited interconnection scheme. Truly, a capability like that emerging as the prospect of the present research will have major impact upon the development of working life within the next decade.

REFERENCES

[BAKE93] Baker R., “Video Compression Standards,” Mid-day Data Compression Standards Talks, Wed., March 31, Data Compression Conference, Snowbird, Utah, March 30-April 2, 1993.

[BARN93] Barnsley, M.F., Hurd L.P., Fractal Image Compression, AK Peters Ltd., 1993, 244 p.

[BELL90] Bell, T., Cleary, J. and I. Whitten, Text Compression, Prentice Hall Advanced Reference Series, 1990.

[BURT83] Burt, P., Adelson, E., "The Laplacian Pyramid as a Compact Image Code," *IEEE Trans. Comm.*, Vol. 31, No. 4, pp. 532-540, April 1983.

[DAUB92] I. Daubechies, *Ten Lectures on Wavelets*, CBMS-61, SIAM publ., 1992.

[DCC93] Proceedings of DCC'93, 1993 Data Compression Conference, Snowbird, Utah, March 30-April 2, 1993, IEEE Computer Society Press, 1993, 505 p.

[DCP92] Proposal to the AIAA Development of a standard for space data base compression, presented at Data Compression Panel Discussion at Communications and Data Systems Workshop sponsored by Data Based Observation Systems Committee on Standards, Colorado Springs, Colorado, May 20, 1992.

[DEVO92] DeVore, R. A., B. Jawerth, B. J. Lucier, "Image Compression Through Wavelet Transform Coding," *IEEE Trans. Inf. Theory*, Vol. 38, No. 2, pp. 719-746, 1992.

[DEVO92a] DeVore R.A., Lucier B.J., “Fast wavelet techniques for near-optimal image processing,” presented at MICOM ‘92.

[FARR90] Farrelle P.M., Recursive Block Coding for Image Data Compression, Springer-Verlag, 1990, 297 p.

[FISH92] Fisher Y., Jacobs E.W., Boss, R.D., “Fractal Image Compression Using Iterated Transforms,” in Image and Text Compression, Ed. J.A. Storer, Chapter 2, pp. 35-61, Kluwer Academic Publishers, 1992.

[GOLD91] Goldberg, M. and L. Wang, "Comparative Performance of Pyramid Data Structures for Progressive Image Transmission," *IEEE Trans. Comm*., Vol. 39, No. 4, pp. 540-548, 1991.

[GROS84] Grossmann A., Morlet J., “Decomposition of Hardy Functions into Square Integrable Wavelets of Constant Shape,” *SIAM J. Math. Anal*., Vol. 15, No. 4, pp.723-736, July 1984.

[HAIN93] Haines R.F., and S.L.Chuang, "A study of video frame rate on the perception of moving imagery detail," in *Proc. 1993 Space and Earth Science Data Compression Workshop*, Snowbird, Utah, April 2, pp.75-84, 1993.

[HOPP92] Hopper T., Preston F., “Compression of Grey-scale Fingerprint Images,” in *Proc. DCC'92, 1992 Data Compression Conference*, Snowbird, Utah, pp.309-318, March 24-27, 1992.

[KIFI92] Kiselyov, O. and P. Fisher, "Wavelet Compression with Feature Preservation and Derivative Definition," in *Proc. DCC'92, 1992 Data Compression Conference*, Snowbird, Utah, p.442, March 24-27, 1992.

[KIFI93] Kiselyov, O. and P. Fisher, "Pyramidal Image Decompositions: A New Look," in *Proc. DCC'93, 1993 Data Compression Conference*, Snowbird, Utah, p.479, March 30-April 2, 1993.

[LEE92] Lee V., "Video Compression Standards - Making Sense of the Alphabet Soup," *Computer Design*, pp.84-85, January 1992.

[MALL89a] Mallat S., "A Theory for Multiresolutional Signal Decomposition: the Wavelet Representation," *IEEE Trans. on Pattern Analysis and Machine Intell.*, Vol. 11, No. 7, pp.674-693, July 1989.

[MALL89b] Mallat, S., "Multifrequency Channel Decompositions of Images and Wavelet Models," *IEEE Trans. on Acoust., Speech, Signal Proc*., Vol. 37, No. 12, pp.2091-2110, December 1989.

[MAND83] Mandelbrot, B.B., The Fractal Geometry of Nature, W.H. Freeman and Co., 1983.

[MONT91] Montanvert, A., Meer P., and A. Rosenfeld, "Hierarchical Image Analysis Using Irregular Tessellations," *IEEE Trans. on Pattern Analysis and Machine Intell*, Vol. 13, No. 4, pp. 307-316, April 1991.

[NOVI93] Novik D.A., Tilton J.C., Manohar M., "Compression through Decomposition into Browse and Residual images," in *Proc. 1993 Space and Earth Science Data Compression Workshop*, Snowbird, Utah, April 2, pp.7-12, 1993.

[PAVL82] Pavlidis T., Algorithms for Graphics and Image Processing, Rockville, MD, Computer Science Press, 1982, 416 p.

[PEAR91] Pearlman W.A., “Performance Bounds for Subband Coding,” in Subband Image Coding, Ed. J.W.Woods, Chapter 1, Kluwer Academic Publishers, 1991.

[QUIC92] QuickTime 1.5 QuickHit List, Apple Corp., 1992

[RIOU91] Rioul O., and Vetterli M., “Wavelets and Signal Processing,” *IEEE Signal Processing Magazine*, Vol. 8, No. 4, pp.14-38, October 1991.

[SBIR93] Requests N93-035 and N93-115, Program solicitation 93.1, FY 1993 Small Business Innovation Research (SBIR) Program, U.S. Department of Defense, SBIR Program Office, Washington, DC 20301, 1993.

[SIMO91] Simoncelli E.P., Adelson E.H., “Subband Transforms,” in Subband Image Coding, Ed. J.W.Woods, Chapter 4, Kluwer Academic Publishers, 1991.

[UNSE92] Unser M., “An Improved Least Squares Laplacian Pyramid for Image Compression,” *Signal Processing*, Vol. 27, pp.187-203, 1992.