Certificate of Approval

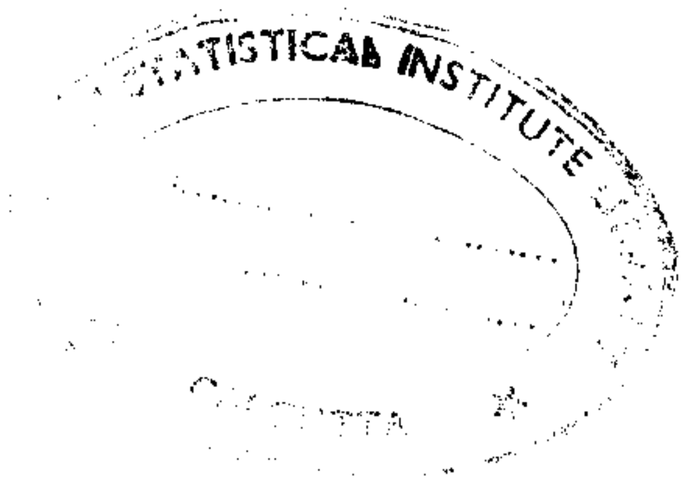
We, the undersigned, hereby certify that Mr. Halbe Devdatta R., Indian Statistical Institute, Calcutta, has completed this dissertation on "DCT and Fractal Image Compression" as a part of the M.Tech. (Computer Science) course, to our satisfaction.

External Examiner

Supervisors

Prof.M.K. Kundu

Dr. C.A. Murthy



Acknowledgment

I am taking this opportunity to express my sincere thanks to **Dr. C. A. Murthy** and **Dr. Malay Kundu**, who introduced me to this subject and gave me the freedom to explore it on my own, guiding me in correct direction whenever I seemed to lose my way. Many thanks to **Suman Maitra** for pointing out possible mistakes in my program. Printing of this report would not have been possible without the help of **Uma Shankar**. Many thanks to him too.

Contents

1 Image Compression: an introduction **1**

1.1 Why Image Compression 1

1.2 Image Compression Fundamentals 2

1.3 Karhunen-Loeve Transform 4

2 Discrete Cosine Transform **7**

2.1 Fourier Cosine Transform and DCT 7

2.2 Properties of DCT 9

2.3 Asymptotic Optimality of DCT 13

2.3.1 Asymptotic convergence of DCT-I to KLT 13

2.3.2 Asymptotic convergence of DCT-II to KLT 16

2.4 Use of DCT in JPEG 17

3 Fractal Image Compression **20**

3.1 Introduction to Fractals 20

3.2 Theoretical Foundations of FIC 22

3.3 Fractal Image Compression 24

3.4 Results 30

List of Figures

1.1 Source encoder 2

1.2 Source decoder 3

1.3 Transform coder 4

3.1 Decoded images : THR = 100, THR = 50 31

3.2 Decoded images : THR = 25, THR = 20 32

3.3 Decoded images : THR = 10 33

3.4 Decoded images : $\alpha = 0.1$ and $\beta = 0.4$ 35

3.5 Decoded images : $\alpha = 0.15$ and $\beta = 0.2, 0.4$ 36

3.6 Decoded images : $\alpha = 0.2$ and $\beta = 0.2, 0.4$ 37

3.7 THR = 25 and range block size 4×4 and the original image 40

Chapter 1

Image Compression: an introduction

1.1 Why Image Compression

A *simple gray level image* can be defined in a very simplistic way as a function $f : \mathcal{R}^2 \longrightarrow \mathcal{R}$ with the function value $f(x, y)$ giving the brightness value (or gray level may be with change in scale). When this image is to be represented on a computer screen (which has finite resolution) image is discretized both spatially and in amplitude. Then image can be considered as a matrix (of points) whose rows and column indices identify a point in the image and the corresponding matrix element value identifies the gray level at that point (the point is called *pixel or pel*).

Now suppose the size of the above mentioned matrix is $N \times N$ and the number of gray levels available is M then the number of bits required to store this image is $N \times N \times \log(M)$. Even for moderate resolution (N) and number of gray levels (M), the size of the image is too high for its practical use. Today's computing requires a constant image processing, storage and retrieval of images and that too with the least possible delay. This fast processing of images requires the image size be as small as possible. So the image compression is the need of today's computing world. With this we start our discussion of *image compression.*

1.2 Image Compression Fundamentals

In a digital image, there can be three basic types of data redundancies, **coding redundancy**, **inter-pixel redundancy** and **psycho-visual redundancy**. An image compression algorithm must identify these and exploit them to achieve good image compression. For more details one may refer to [1].

An image is said to have *coding redundancy* if the gray levels of the image are coded in a way that uses more code symbols than absolutely necessary to represent each gray level. This can be avoided if the coding of symbols is done depending on their probability of occurrence. Huffman coding reduces coding redundancy.

Various relationships among the objects in an image leads to *inter-pixel redundancy* in the image. Because of these relationships value of some of the pixels can be estimated from the pixel value(s) of its neighbor(s). If such estimation is possible then coding of each pixel is not necessary. This redundancy can be removed by certain applying certain *transformations* on the image. These transformations can be very complicated like *Karhunen-Loeve transforms* or very simple like taking difference of successive pixels on each scan line.

In normal visual processing, certain information in an image simply has less relative importance than some other information. This information is said to be *psycho-visually redundant* and can be eliminated without introduction of noticeable visual distortion in the image. Elimination of psycho-visually redundant data results in a loss of quantitative information, it is commonly referred to as *quantization*; the process is irreversible and results in a lossy data compression.

So a general model for source coder is as in figure (1.1) and that of decoder is as in figure (1.2).

$$f(x,y) \longrightarrow \underset{\text{(inter-pixel)}}{\text{Mapper}} \longrightarrow \underset{\text{(psycho-visual)}}{\text{Quantizer}} \longrightarrow \underset{\text{(coding)}}{\text{Symbol encoder}} \longrightarrow \text{channel}$$

Figure 1.1: Source encoder

After looking at the encoders in general, we classify them into two categories, *lossless coders* (whose decoder can reconstruct the original image with no loss of information of any kind) and *lossy coders* (reconstructed image contains less information

channel $\longrightarrow$ Symbol decoder $\longrightarrow$ Inverse mapper $\longrightarrow$ $\hat{f}(x, y)$

Figure 1.2: Source decoder

than the original). Lossless compression is useful when loss of information can not be afforded (e.g. medical images, satellite images), but compression ratio that can be achieved by these coders is limited. The lossy coders try to discard the psycho-visual redundancies and need not have to reproduce the image exactly; this makes them more efficient in compression of data. For more information about various image or in general data compression refer to [2] or [3].

All lossy coders, at some stage apply *quantization* process. Quantizations are of two types:

1) **Scalar Quantization**: In this, the range of the pixel values is divided into fixed number of intervals and all pixels having pixel values lying in same interval are assigned the same value. While quantizing, we would like to minimize the error due to quantization. A quantizer is specified by its input partition and output reproduction points. Optimal quantizers satisfy the following two conditions:

1. Given the encoder (input partition) the best decoder is one that puts the reproduction points on the centers of mass of the partitions. This is known as **centroid condition**.

2. Given decoder (reproduction points) the best encoder is the one that puts the partition boundaries exactly in the middle of the reproduction points. This is known as **nearest neighbor condition**.

2) **Vector Quantization**: In this, instead of quantizing pixels individually, a group of pixels (vector) is quantized. During the decoding the decoder needs a code book (a lookup table). Let $D(x, y)$ be an appropriate distortion measure and $\{P_k\}$ forms the partition of given source image. Then the optimal vector quantizer satisfies the following two conditions (modified versions of above mentioned two conditions):

1. Nearest Neighbor Condition: If the given reconstruction values are $\{\hat{x}_k\}$ then the optimal partition is given by

$$P_k = \{x: D(x, \hat{x}_k) \leq D(x, \hat{x}_j) \quad for\ j \neq k\}$$

2. Centroid Condition: Given partition P_k the optimal reconstruction values are given by

$$\hat{x}_k = arg\min \int p_x(z) D(z, \hat{x}_k) dz$$

Theoretically, a vector quantizer captures maximum possible compression, but it can achieve it only asymptotically as its dimensionality increases. Also for design of optimal vector quantizers the knowledge of underlying probability density function of the image is required, which is more and more difficult to obtain as dimensionality increases. Also computation cost and delay increases exponentially with dimensionality, limiting the practical applicability of vector quantizer. So most practical coders have chosen *transform coding* as their basis.

The basic structure of transform coder is as follows:

$$f(x,y) \xrightarrow{T} \begin{matrix}\text{transform}\\ \text{domain}\end{matrix} \xrightarrow[quantization]{} \begin{matrix}\text{quantized}\\ \text{coefficients}\end{matrix} \xrightarrow[coding]{} \begin{matrix}\text{encoded}\\ \text{image}\end{matrix}$$

Figure 1.3: Transform coder

The efficiency of transform coders depends on the way their basis function represents the image. The efficiency increases with the transform coders ability to decorrelate the image data points.[1] Thus transform coders enable us to obtain some of the benefits of vector quantization with much lower complexity. So now we will concentrate on transform coders.

1.3 Karhunen-Loeve Transform

Also known as *Hotelling transform*, the Karhunen-Loeve transform is the optimal transform in the sense that it completely decorrelates the input data values. We briefly describe this optimal transform.

Suppose the input data is in the form of vector, say $\vec{x}$. Let T be the transformation matrix and let $\vec{y}$ be the transformed vector. Our aim is to find the conditions on

[1]The nearby pixel values are not independent but are highly correlated. This correlation hinders the efficiency of representation (inter-pixel redundancy).

matrix T, such that the components of $\vec{y}$ are completely uncorrelated.[2] The problem can be reformulated as follows:

Given a random vector $\vec{x} = (x(0), x(1), \ldots, x(N-1))^T$, find a set of basis vectors $\Phi_i's, i = 0, \ldots, N-1$, such that the error of a truncated representation is minimized in the *mean squared error* sense.

i.e. we want to minimize

$$\epsilon = E[(\vec{x} - \bar{x})^2] = E\left[\left(\sum_{i=0}^{N-1} X_i\Phi_i - \sum_{i=0}^{D-1} X_i\Phi_i\right)^2\right]$$

This problem of optimization directly leads to eigen-value problem. Let

$$[\Phi] = [\Phi_0, \Phi_1, \cdots, \Phi_{N-1}]$$

where $\Phi_0, \Phi_1, \ldots, \Phi_{N-1}$ are eigenvectors of variance-covariance matrix (which is positive definite) of input signal $\vec{x}$ (given by $E[(\vec{x} - \bar{x})(\vec{x} - \bar{x})^T]$, where $\bar{x}$ is the mean) corresponding to eigenvalues $\mu_0, \mu_1, \cdots, \mu_{N-1}$.

Then it can proved that the required K-L transform matrix is given by $[\Phi]$ and the mean square error is given by

$$\epsilon = \sum_{i=D}^{N-1} \mu_i$$

and note that ϵ is minimized if we arrange eigenvalues μ_i's in a descending order.

So to summarize, K-L transform is optimal in the sense that:

1. It completely decorrelates input data values.
2. It minimizes the mean square error in data compression.
3. It contains the most of the variance (energy) in the fewest number of transform coefficients.

[2]To encode a sinusoid, the number points that are to be encoded is proportional to the accuracy of reproduction. But if we know its magnitude, phase, frequency, starting time, and the fact that it is sinusoid then we can reproduce the signal exactly. So these 5 *completely uncorrelated* values have same information as the original waveform. So complete uncorrelation of data values will give high compression.

4. It minimizes total representation entropy in the sequence.

But practical implementations of K-L transform are far from efficient, for the basis vectors of K-L transform depend on the input signal. To calculate them we have to first calculate the variance-covariance matrix of input signal, find its eigenvalues and corresponding eigenvectors (matrix eigenvalue problem). This dependence of basis vectors on input and the complexity of eigenvalue problem makes K-L transform impractical. But theoretically the transform is the best and can be used to benchmark the efficiency of other transforms.

Now our aim is to find a transform whose basis vectors can be determined apriori and which can approximate the K-L transform reasonably well. In the next chapter we will look at the *Discrete Cosine Transforms* (DCTs) which will serve our purpose. We end this discussion on K-L transform by looking at a special case, in which the analytic solution of eigenvalue problem mentioned above can be obtained. We will use this case in the next chapter while proving the efficiency of DCT.

A special case:

Let the variance-covariance matrix A of the input $\bar{x}$ has a special form as

$$[A]_{ik} = \rho^{|i-k|} \quad i,k = 0,1,\cdots,N-1$$

for $0 < \rho < 1$ and ρ is adjacent correlation coefficient. (Such a signal is said to be a *stationary Markov-1 signal*).

The solution in this case is provided by Ray and Driver (see ([4])) as:
$[\phi]_{nm} = \Phi_m(n) =$ n'th component of m'th eigenvector

$$= \left[\frac{2}{(N+\mu_m)}\right]^{1/2} sin\left\{w_m\left[(n+1) - \frac{(N+1)}{2}\right] + (m+1)\frac{\pi}{2}\right\} \tag{1.1}$$

where $m,n = 0,1,\cdots,N-1$, and $\mu_m = (1-\rho^2)/[1-2\cos(w_m)+\rho^2]$ are eigenvalues and w_m's are the real positive roots of the transcendental equation:

$$tan(Nw) = -\frac{(1-\rho^2)\sin(w)}{[\cos(w) - 2\rho + \rho^2\cos(w)]} \tag{1.2}$$

Chapter 2

Discrete Cosine Transform

The Discrete Cosine Transform (hereafter referred to as DCT) is one of the most widely used transform in signal processing because of its various useful properties. In this chapter we will derive few important properties of DCT and then prove the asymptotic optimality of DCT (convergence to K-L transform) under certain assumptions.

2.1 Fourier Cosine Transform and DCT

For an input $x(t)$ with $-\infty < t < \infty$, its Fourier transform is given by

$$X(\omega) = F[x(t)] = \left(\frac{1}{2\pi}\right)^{1/2} \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt \tag{2.1}$$

and the inverse transform is given by

$$x(t) = F^{-1}[X(\omega)] = \left(\frac{1}{2\pi}\right)^{1/2} \int_{-\infty}^{\infty} X(\omega) e^{j\omega t} d\omega \tag{2.2}$$

Now if $x(t)$ is defined only for $t \geq 0$, we can construct $y(t)$ as

$$\begin{aligned} y(t) &= x(t); \qquad t \geq 0 \\ &= x(-t); \qquad t \leq 0 \end{aligned}$$

Then

$$F[y(t)] = \left(\frac{1}{2\pi}\right)^{1/2}\left\{\int_0^\infty x(t)e^{-j\omega t}dt + \int_{-\infty}^0 x(-t)e^{-j\omega t}dt\right\}$$
$$= \left(\frac{1}{2\pi}\right)^{1/2}\int_0^\infty x(t)[e^{-jwt} + e^{jwt}]dt$$
$$= \left(\frac{2}{\pi}\right)^{1/2}\int_0^\infty x(t)\cos(\omega t)dt$$

Now we define the Fourier Cosine Transform as:

$$X_c[\omega] = F_c[x(t)] = \left(\frac{2}{\pi}\right)^{1/2}\int_0^\infty x(t)\cos(\omega t)dt \tag{2.3}$$

and its inverse as:

$$x(t) = F_c^{-1}[X_c(\omega)] = \left(\frac{2}{\pi}\right)^{1/2}\int_0^\infty X_c(\omega)\cos(\omega t)d\omega \quad (t \geq 0) \tag{2.4}$$

From equation (2.3) it follows that the FCT is an integral transform with the kernel $K_c(\omega, t) = \cos(\omega t)$ up-to a normalization constant. Let

$$\omega_m = 2\pi m\delta f \quad and \quad t_n = n\delta t$$

be sampled frequency and time, m, n are integers. Then we can rewrite the equation for the kernel as, assuming $\delta f \delta t = 1/(2N)$:

$$K_c(\omega_m, t_n) = K_c(2\pi m\delta f, n\delta t) = cos(2\pi mn\delta f\delta t) = K_c(m, n) \tag{2.5}$$
$$K_c(m, n) = \cos\left(\frac{\pi mn}{N}\right) \tag{2.6}$$

Now we define various forms of DCT as follows:

- **DCT-I:**

$$[C_{N+1}^I]_{mn} = \left(\frac{2}{N}\right)^{1/2}\left[k_m k_n \cos\left(\frac{mn\pi}{N}\right)\right] \qquad m, n = 0, 1, \cdots, N.$$

- **DCT-II:**

$$[C_N^{II}]_{mn} = \left(\frac{2}{N}\right)^{1/2} \left[k_m \cos\left(\frac{m(n+1/2)\pi}{N}\right)\right] \quad m,n = 0,1,\cdots,N-1.$$

- **DCT-III:**

$$[C_N^{III}]_{mn} = \left(\frac{2}{N}\right)^{1/2} \left[k_n \cos\left(\frac{(m+1/2)n\pi}{N}\right)\right] \quad m,n = 0,1,\cdots,N-1.$$

- **DCT-IV:**

$$[C_N^{IV}]_{mn} = \left(\frac{2}{N}\right)^{1/2} \left[\cos\left(\frac{(m+1/2)(n+1/2)\pi}{N}\right)\right] \quad m,n = 0,1,\cdots,N-1.$$

where

$$k_j = 1 \quad if\ j \neq 0\ or\ N \tag{2.7}$$

$$= \frac{1}{\sqrt{2}} \quad if\ j = 0\ or\ N. \tag{2.8}$$

2.2 Properties of DCT

1. **Unitarity of DCT**
   We outline the proof of unitarity of DCT-I. Let g_k denote the k'th column vector in $(N+1) \times (N+1)$ transform matrix C_{N+1}^I, then

$$\begin{aligned}
&\langle g_l, g_m \rangle \\
&= \sum_{n=0}^{N} \left(\tfrac{2}{N}\right) k_l k_m k_n^2 \cos\left(\tfrac{nl\pi}{N}\right) \cos\left(\tfrac{nm\pi}{N}\right) \\
&= \left(\tfrac{k_l k_m}{N}\right) \left[\sum_{n=1}^{N-1} \left[\cos\left(\tfrac{n\pi(l+m)}{N}\right) + \cos\left(\tfrac{n\pi(l-m)}{N}\right)\right] + 1 + (-1)^{l+m}\right] \\
&= \left(\tfrac{k_l k_m}{N}\right) \left[\sum_{n=0}^{N-1} \cos\left(\tfrac{(l-m)n\pi}{N}\right) + \sum_{n=1}^{N} \cos\left(\tfrac{(l+m)n\pi}{N}\right)\right] \\
&= \left(\tfrac{k_l k_m}{N}\right) Re\left[\sum_{n=0}^{N-1} W_{2N}^{-n(l-m)} + \sum_{n=1}^{N} W_{2N}{}^{-n(l+m)}\right]
\end{aligned}$$

   where W_{2N} is primitive 2N'th root of unity. Manipulating the last equation it can be proved

$$\langle g_l, g_m \rangle = \delta_{lm} \quad (Kroneker\ delta)$$

   which implies the unitarity of DCT-I. Similarly the unitarity of other DCT's can be proved in similar manner. For detail proofs refer to So we have following:

- $[C_{N+1}^{I}]^{-1} = [C_{N+1}^{I}]^{T} = [C_{N+1}^{I}]$
- $[C_{N}^{II}]^{-1} = [C_{N}^{II}]^{T} = [C_{N}^{III}]$
- $[C_{N}^{III}]^{-1} = [C_{N}^{III}]^{T} = [C_{N}^{II}]$
- $[C_{N}^{IV}]^{-1} = [C_{N}^{IV}]^{T} = [C_{N}^{IV}]$

This immediately gives us the formulae for forward and inverse DCT transforms:

- **DCT-I**

Forward:

$$X^{C(1)}(m) = \left(\frac{2}{N}\right)^{1/2} k_m \sum_{n=0}^{N} k_n x(n) \cos\left(\frac{mn\pi}{N}\right) m = 0, 1, \cdots, N;$$

Inverse:

$$x(n) = \left(\frac{2}{N}\right)^{1/2} k_n \sum_{m=0}^{N} k_m X^{C(1)}(m) \cos\left(\frac{mn\pi}{N}\right) n = 0, 1, \cdots, N.$$

- **DCT-II**

Forward:

$$X^{C(2)}(m) = \left(\frac{2}{N}\right)^{1/2} k_m \sum_{n=0}^{N-1} x(n) \cos\left[\frac{m(2n+1)\pi}{2N}\right] m = 0, 1, \cdots, N-1;$$

Inverse:

$$x(n) = \left(\frac{2}{N}\right)^{1/2} \sum_{m=0}^{N-1} k_m X^{C(1)}(m) \cos\left[\frac{m(2n+1)\pi}{2N}\right] n = 0, 1, \cdots, N-1.$$

Forward and backward transformations for DCT-III and DCT-IV can be obtained similarly, (compare the kernels for the transforms).

2. **Scaling in time**
Since DCT s deal with discrete sample points, a scaling in time has no effect in the transform, except changing the unit frequency interval in the transform domain. Thus as δt changes to $a\delta t$, δf changes to $\delta f/a$, provided N remains same.

3. **Shift in time**
Let $\bar{x} = [x(0), \cdots, x(N)]^T$ and $\bar{x}_+ = [x(1), \cdots, x(N+1)]^T$ be two sampled data

vectors in dimension N+1, with $\bar{x}_+$ denoting the one sample shifted data vector. Then it can proved that

- **DCT-I**

$$\begin{aligned} X_+^{C(1)}(m) &= \cos\left(\frac{m\pi}{n}\right) X^{C(1)}(m) + k_m \sin\left(\frac{m\pi}{N}\right) X^{S(1)}(m) \\ &+ \left(\frac{2}{N}\right)^{1/2} k_m \left[\left(\frac{-1}{\sqrt{2}}\right)\cos\left(\frac{m\pi}{N}\right) x(0) + \left(\frac{1}{\sqrt{2}} - 1\right) x(1)\right. \\ &+ \left.(-1)^m \left(1 - \frac{1}{\sqrt{2}}\right)\cos\left(\frac{m\pi}{N}\right) x(N) + (-1)^m \frac{x(N+1)}{\sqrt{2}}\right] \end{aligned}$$

where

$$X^{S(1)}(m) = \left(\frac{2}{N}\right)^{1/2} \sum_{n=1}^{N-1} \sin\left(\frac{mn\pi}{N}\right) x(n)$$

which is actually *Discrete Sine Transform* of type-I (DST-I).

- **DCT-II**

$$\begin{aligned} X_+^{C(2)}(m) &= \cos\left(\frac{m\pi}{N}\right) X^{C(2)}(m) + \sin\left(\frac{m\pi}{N}\right) X^{S(2)}(m) \\ &+ \left(\frac{2}{N}\right)^{1/2} k_m[(-1)^m x(N) - x(0)] \cos\left(\frac{m\pi}{2N}\right), \end{aligned}$$

where

$$X^{C(2)} = [C_N^{II}]x \;\; and \;\; X_+^{C(2)}] = [C_N^{II}]x_+$$

and $X^{S(2)}$ (DST-II) is defined similarly as $X^{S(1)}$ (with appropriate in the kernel). For time-shift formulae for other DCTs, refer to

4. **Convolution Property** Instead of taking conventional linear or circular convolution we take convolution of even extension of the given sequence as follows: If the given sequence is $f(n)$ for $n = 0, 1, \cdots, N-1$ then we construct $\hat{f}(n)$ as

$$\begin{aligned} \hat{f}(n) &= f(n) \qquad for\ 0 < n < N-1 \\ &= f(2N-n-1) \quad for\ N < n < 2N \end{aligned}$$

Now the $2N$ point DFT of this extended sequence is given by

$$\hat{F}_k(k) = \left(\frac{1}{\sqrt{2N}}\right) \sum_{n=0}^{2N-1} \hat{f}(n) \exp\left[-\frac{j2\pi kn}{2N}\right] \tag{2.9}$$

Using symmetry in above equation we can write it as

$$\hat{F}_k(k) = \left(\frac{1}{\sqrt{2N}}\right) \sum_{n=0}^{N-1} f(n) \left\{ \exp\left[\frac{-j2\pi kn}{2N}\right] + \exp\left[\frac{j2\pi k(n+1)}{2N}\right] \right\} \quad (2.10)$$

Introducing shift and a weight factor in the above equation we get

$$k_k \hat{F}_F(k) \exp\left[\frac{-j\pi k}{2N}\right] = \sqrt{\left(\frac{2}{N}\right)} \sum_{n=0}^{N-1} k_k f(n) \cos\left\{\frac{k(n+1/2)\pi}{N}\right\} \quad (2.11)$$

where

$$\begin{aligned} k_m &= \frac{1}{\sqrt{2}} \quad m = 0 \\ &= 1 \quad 1 < m < N-1 \\ &= 0 \quad otherwise \end{aligned}$$

Note that RHS of above equation is the DCT-II for sequence for $f(n)$. Therefore, using $F^{C(2)}(k)$ to denote the DCT-II for the sequence we get

$$F^{C(2)}(k) = k_k \exp\left[\frac{-j\pi k}{2N}\right] \hat{F}_F(k) \quad k = 0, 1, \cdots, N-1. \quad (2.12)$$

From DCT-II transform the original sequence can be obtained using following equation

$$f(n) = \left(\frac{2}{N}\right)^{1/2} Re\left\{ \sum_{k=0}^{2N-1} k_k F^{C(2)}(k) \exp\left[\frac{j\pi k}{2N}\right] \exp\left[\frac{2j\pi kn}{2N}\right] \right\} \quad (2.13)$$

where $n = 0, 1, \cdots, N-1$.

Now we give convolution theorem for DCT-II (similar to convolution theorem for Fourier transform). Let $F^{C(2)}$ *and* $G^{C(2)}$ be DCT-II transforms corresponding to $f(n)$ *and* $g(n)$ respectively. Then by equation (2.12) we get

$$W^{C(2)}(k) = k_k^2 \exp\left[\frac{-j\pi k}{N}\right] \hat{F}_F(k) \hat{G}_F(k), \quad (2.14)$$

Now applying equation (2.13) and using convolution theorem for Fourier transforms we get

$$w(n) = \hat{h}(n) \star \hat{f}(n) \star \hat{g}(n) \quad (2.15)$$

where

$$\frac{\hat{h}(n)}{2} = \left\{\left(\frac{1}{2\sqrt{2}} - 1\right) + \exp\left[\frac{j(2n-1)(N-1)\pi}{4N}\right] \times \frac{\sin\left[\frac{(2n-1)\pi}{4}\right]}{\sin\left[\frac{(2n-1)\pi}{4N}\right]}\right\} / \sqrt{2N}$$

So the *convolution theorem* is

$$F^{C(2)}[\hat{h}(n) \star \hat{f}(n) \star \hat{g}(n)] = F^{C(2)}(k)G^{C(2)}(k), \quad k = 0, 1, \cdots, N-1. \tag{2.16}$$

This completes our discussion of some important mathematical properties of DCT. In the next section we look into the asymptotic optimality of DCTs.

2.3 Asymptotic Optimality of DCT

In this section we will derive the asymptotic optimality of DCT-I and DCT-II, in the sense that both of them converge to KLT under certain conditions. And then we will see that by a generalization of the framework used for above proofs, we can prove asymptotic convergence of DCT-II to KLT for the Markov signals of all orders. So we start with optimality of DCT-I.

2.3.1 Asymptotic convergence of DCT-I to KLT

In the previous section on Fourier Cosine transforms we derived formula DCT-I using discretized Fourier DCT. But the original derivation by *Kitajima* was motivated by an attempt to approximate KLT basis by some simpler functions. The derivation is based on recursive relation of the Tchebyshev polynomials of first kind, the diagonalization of tridiagonal matrix, and its asymptotic equivalence (as N tends to ∞) to the inverse of the Markov-I auto-covariance matrix. The derivation is as follows:

Symmetric Toeplitz matrix $[A]$ for the auto-covariance of the Markov-I signal has

an inverse in tridiagonal form given by,

$$[A]^{-1} = (1-\rho)^{-1}\begin{bmatrix} 1 & -\rho & 0 & 0 & \cdots & \cdots & \cdots \\ -\rho & (1+\rho^2) & -\rho & 0 & \cdots & \cdots & \cdots \\ 0 & -\rho & (1+\rho^2) & -\rho & 0 & \cdots & \cdots \\ \cdots & 0 & -\rho & (1+\rho^2) & -\rho & 0 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & (1+\rho^2) & -\rho \\ \cdots & \cdots & \cdots & \cdots & \cdots & -\rho & 1 \end{bmatrix} \quad (2.17)$$

Then we decompose $[A]^{-1}$ as $[A]^{-1} = [B] + [R]$ where

$$[B] = (1-\rho^2)^{-1}\begin{bmatrix} (1+\rho^2) & -\sqrt{2\rho} & 0 & \cdots & \cdots \\ -\sqrt{2\rho} & (1+\rho^2) & -\rho & \cdots & \cdots \\ 0 & -\rho & (1+\rho^2) & -\rho & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & -\sqrt{2\rho} & (1+\rho^2) \end{bmatrix} \quad (2.18)$$

and

$$[R] = (1+\rho^2)^{-1}\begin{bmatrix} -\rho^2 & (\sqrt{2}-1)\rho & 0 & \cdots & \cdots \\ (\sqrt{2}-1)\rho & 0 & 0 & \cdots & \cdots \\ \cdots & \cdots & \cdots & 0 & (\sqrt{2}-1)\rho \\ \cdots & \cdots & \cdots & (\sqrt{2}-1)\rho & -\rho^2 \end{bmatrix} \quad (2.19)$$

Note that $[R]$ is very nearly equal to zero. The matrix B can be further decomposed as

$$[B] = \left\{\frac{-2\rho}{(1-\rho^2)}\right\}[B_1] + \left\{\frac{(1+\rho^2)}{(1-\rho^2)}\right\}[I_n] \quad (2.20)$$

where

$$[B_1] = \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & 0 & 0 & \cdots & \cdots & \cdots \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{2} & 0 & \cdots & \cdots & \cdots \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \frac{1}{2} & 0 & \frac{1}{\sqrt{2}} \\ \cdots & \cdots & \cdots & \cdots & 0 & \frac{1}{\sqrt{2}} & 0 \end{bmatrix} \quad (2.21)$$

From equation (2.20) it follows that the transformation that diagonalizes $[B_1]$ will diagonalize $[B]$. The diagonalization of $[B_1]$ depends on recurrence relation of Tchebyshev polynomials of first kind, which are given by

$$T_n(x) = \cos(n\theta), \quad \cos(\theta) = x, \quad n = 0, 1, \cdots \quad (2.22)$$

and the relevant recurrence relation is

$$\frac{1}{2}T_{n-1}(x) + \frac{1}{2}T_{n+1}(x) = xT_n(x) \tag{2.23}$$

If first N polynomials are considered at the discrete values of x given by

$$x_k = \cos\left\{\frac{k\pi}{(N-1)}\right\}, \quad k = 0, 1, \cdots, N-1. \tag{2.24}$$

then

$$\begin{aligned} \frac{1}{2}T_{n-1}(x_k) + \frac{1}{2}T_{n+1}(x_k) &= x_k T_n(x_k), \quad n = 1, 2, \cdots, N-2, \\ T_1(x_k) &= x_k T_0(x_k) \end{aligned}$$

and

$$T_{N-2}(x_k) = x_k T_{N-1}(x_k) \tag{2.25}$$

Above equation can be represented in matrix form if we define an $N-$dimensional vector given by

$$\mathbf{v}_k = \left\{\frac{T_0(x_k)}{\sqrt{2}}, T_1(x_k), \cdots, \frac{T_{N-1}(x_k)}{\sqrt{2}}\right\}^T \tag{2.26}$$

Then

$$[B_1]\mathbf{v}_k = x_k\mathbf{v}_k, \quad k = 0, 1, \cdots, N-1. \tag{2.27}$$

Hence

$$[B]\mathbf{v}_k = w_k\mathbf{v}_k, \quad k = 0, 1, \cdots, N-1. \tag{2.28}$$

where the eigenvalues w_k are given by

$$w_k = (1-\rho^2)^{-1}\left\{1 + \rho^2 - 2\rho\cos\left[\frac{k\pi}{(N-1)}\right]\right\} \tag{2.29}$$

Note that in equation (2.18) as $N \to \infty$, $[B]$ should asymptotically approaches $[A]^{-1}$. This is valid in transform domain at least as proved below: Let

$$[V] = [\hat{v}_0, \hat{v}_1, \cdots, \hat{v}_{N-1}]^T$$

where $\hat{v}_k$ is normalized eigenvectors in (2.28). Since $\hat{v}_k$ are real, the matrix $[V]$ are real and unitary. Thus $[V][V]^T = [I_N]$. Applying similarity transform to (2.18) we get

$$\begin{aligned} [V]^T[A]^{-1}[V] &= [V]^T[B][V] + [V]^T[R][V] \\ &= diag(w_0, w_1, \cdots, w_{N-1}) + [V]^T[R][V] \end{aligned} \tag{2.30}$$

By direct multiplication it can be shown that

$$\begin{aligned}([V]^T[R][V])_{nm} &= 0, \quad for\ n+m = odd \\ &= k_n k_m 2\rho \frac{\left(-\rho + (2-\sqrt{2})\left[\cos\left(\frac{m\pi}{N-1}\right) + \cos\left(\frac{n\pi}{N-1}\right)\right]\right)}{[(N-1)(1-\rho^2)]} \\ &\quad for\ n+m = even \end{aligned} \tag{2.31}$$

where

$$\begin{aligned} k_m &= \frac{1}{\sqrt{2}} \ \ if\ m = 0\ or\ N-1 \\ &= 1 \quad otherwise. \end{aligned} \tag{2.32}$$

For $\rho \neq 1$ (2.32) vanishes as $N \to \infty$. Therefore we get as $N \to \infty$,

$$[V]^T[A][V] = diag[\mu_0, \mu_1, \cdots, \mu_{N-1}] \ \ where\ \mu_k = \frac{1}{w_k}$$

and

$$[V]_{nm} = k_n k_m \left[\frac{2}{(N-1)}\right]^{1/2} \cos\left[\frac{nm\pi}{(N-1)}\right] \quad n, m = 0, 1, \cdots, N-1. \tag{2.33}$$

Note that this above equation gives us asymptotic equivalence of DCT-I to KLT as $N \to \infty$. But note that as $\rho \to 1$, (2.31) becomes indeterminate, even as $N \to \infty$. This predicts that the decorrelation power of DCT-I decreases as $\rho \to 1$, even when N is large.

2.3.2 Asymptotic convergence of DCT-II to KLT

Now let us look at the asymptotic convergence of DCT-II to KLT. For that look at the equations (1.1) and (1.2). As $\rho \to 1$, we have for the trascedental equation (1.2)

$$\tan(Nw) = 0$$

thus giving

$$w_k = \frac{k\pi}{N} \ \ for\ k = 0, 1, \ldots, N-1. \tag{2.34}$$

The eigenvalues μ_m vanishes when w_m are nonzero. For μ_0, it appears that it should approach infinity. But, the invariance of the trace of a matrix under similarity transformation, we have

$$\sum_{m=0}^{N-1}[A]_{mm} = \sum_{m=0}^{N-1}\mu_m,$$

and since diagonal elements of $[A]$ in Markov-1 signal are all 1's we have $\mu_0 = N$. Combining these and substituting into (1.1), we obtain

$$[\Phi]_{n0} = \frac{1}{\sqrt{N}}$$

and

$$\begin{aligned}[\Phi]_{nm} &= \left(\frac{2}{N}\right)^{1/2}\sin\left[m(n+1/2)\frac{\pi}{N}+\frac{\pi}{2}\right]\\ &= \left(\frac{2}{N}\right)^{1/2}\cos\left[m(n+1/2)\frac{\pi}{N}\right] \quad m\neq 0\end{aligned}$$

Thus upon simplification we get,

$$[\Phi]_{nm} = \left(\frac{2}{N}\right)^{1/2}k_m\cos\left[m(n+1/2)\frac{\pi}{N}\right],\ m,n=0,1,\ldots,N-1 \tag{2.35}$$

and

$$\begin{aligned}k_m &= \frac{1}{\sqrt{2}}\ \ for\ m=0\\ &= 1 \quad otherwise.\end{aligned}$$

This gives the asymptotic convergence of DCT-II to KLT for Markov-1 signals as $\rho \to 1$. Note that this is independent of N. Also note that when $\rho = 1$ all eigenvalues except μ_0 are zero, giving least possible mean square error, which increases as value of ρ deviates from 1. This equations also tells us why DCT-II performs well when ρ is near to 1.

2.4 Use of DCT in JPEG

The most widely used standard for image compression JPEG (Joint Photographic Expert Group) uses DCT-II as a part of its encoder. These standards are for compressing still images. Standards are for both lossy as well as lossless image compression. Lossless image compression standards are further divided into *baseline* JPEG

and *progressive* JPEG standards. In baseline standards, image is compressed in a one go while in progressive JPEG image is first compressed too coarsely (by taking very less number of transformed coefficients) and displayed and then progressively its quality is improved. Both methods give nearly the same compression. For detailed discussion of JPEG standards refer to Wallace's paper ([5]). Here we give brief overview of image compression method described by JPEG standards.

The encoding process consists of following steps:

1. Conversion of RGB(Red/Blue/Green colors format) image to YUV (luminance/ crominance) image, this is because human eye is more sensitive to changes in the intensities of the light as compared to changes in colors.

2. Image is then divided into blocks of size 8×8. The size of these blocks is so chosen so that the computational complexity is reasonable (The computational complexity increases rapidly with the reduction of block size, because then the number of blocks to be processed increases) and to avoid the rapid changes in the image pixel values within a block (edges are good examples of rapid changes in pixel values).

3. After the division of image into blocks, each block is transformed using DCT. Still the process is lossless (ignoring the loss due to finite precision arithmetic). This standards does not mention how the DCT transformation should be implemented and in what should be the precision of the transform. It is left to the implementor.

4. After the transformation, the resulting coefficients are quantized using quantization tables (these tables are predefined in the implementation according to the psycho-visual response of humans to images or the user can provide the quantization tables of his own). JPEG puts no restrictions on that.

5. Depending on the user's choice of desired quality, the number coefficients to be retained is decided (100% corresponds to preserve all coefficients etc.)

6. After the quantization process, coefficients are arranged into a zigzag sequence so that the sequence of zeros (corresponding to high frequency components of image), can be replaced by a single block symbol EOB.

7. Finally thiis sequence is coded in a lossless manner (this lossless coding method is not specified in the standards, generally it is arithmetic coding). This stan-

dard requires that the DC and AC coefficients be coded separately. DC coefficients are coded by differential encoding.

Decoder operates by following all relevant steps in opposite direction. The lossy coders differ from lossless coders only in the quantization step (theoretically only the quantization step introduces the image energy loss).

But these standards give good results for the images which do not have abrupt changes in their pixel values (no edges or corners). Because, the presence of an edge gives rise to high frequency components, which are ignored because they represent relatively small portion of the image. This leads to *blurring of the edges*. So these JPEG coders perform well on natural images but fail completely on line drawings.

Another situation in which these coders fail is when the number of pixels representing higher gray level values is very less (but they represent important information in the image) as compared to other pixel values. An example of this is a photograph of a wild cat taken in the night. The beast's shiny eyes are represented by very less number of pixels with higher gray level values (but they represent very important information, namely the eyes of the beast) . As a result of the quantization strategy adopted by JPEG, these pixels will not be present in the encoded images, leading to complete alteration of the image.

Because of these inherent problems in JPEG coders, we need to study other image compression method that can overcome these problems, and one of them is Fractal Image compression, that will be the topic of our discussion in the next chapter.

Chapter 3

Fractal Image Compression

In the previous chapter we described JPEG image compression scheme which uses Discrete Cosine Transform (DCT). One major disadvantage of that method is *edge blurring* . This defect arises because the edges give rise to high frequency components in transformed domain, which when discarded, lead to edge blurring. Any compression scheme, based on spatial to frequency domain transformation will give rise to edge blurring defects. Instead if we use compression scheme based on geometric transformations, we can avoid the problem of edge blurring. One such method is *Fractal Image Compression* (FIC).

Transform coders, as described in previous chapter, take advantage of inter-pixel correlation. Fractal coders are motivated by the observation that straight edges, and constant regions are invariant under rescaling. These coders take advantage of this scale invariance by using coarse scale image features to quantize fine scale features. They perform vector quantization of image blocks but unlike vector quantizers, the fractal coders do not require separate storage of fixed vector codebook. In this chapter we will learn more about fractal coders.

3.1 Introduction to Fractals

Almost all natural objects show fractal structure. But, precise definition of the term fractal is still elusive. Roughly we can call a set to be a fractal object, if typically

shows one or more of the following properties:

1. F has details at every scale.
2. F is exactly or approximately self-similar.
3. Box dimension of F is grater than its topological dimension.

For more detailed discussion on this refer to [6] or [7]. The third property is the most rigorous and sq we define the terms it uses:

1. **Topological Dimension:** Topological dimension of a totally disconnected set is always zero. The topological dimension of set F is n if arbitrarily small neighborhood of every point of F has boundary of dimension n-1.
2. **Box Dimension:** For $F \subseteq \mathcal{R}^n$, let $N_\epsilon(F)$ denote the the smallest number of sets with diameter no longer than ϵ that can cover F. Then *Box dimension* of F is

$$\lim_{\epsilon \to 0} \frac{N_\epsilon(F)}{-\log \epsilon}$$

when limit exists.

A simple example of fractal object is *Cantor's (middle third) set* . It can be constructed as follows:

1. Let $S_0 = [0, 1]$ and let f_1, f_2: $[0, 1] \longrightarrow [0, 1]$ such that

$$f_1(x) = \frac{x}{3} \; and \; f_2(x) = \frac{x}{3} + \frac{2}{3}$$

2. Let $S_1 = f_1(S_0) \bigcup f_2(S_0)$. In general $S_{n+1} = f_1(S_n) \bigcup f_2(S_n)$.

Then the cantor's set is

$$C = \bigcap_{n=0}^{\infty} S_n$$

Then it can be proved that the set is totally disconnected, so its topological dimension is zero. But its box dimension is $\frac{\log 2}{\log 3}$. Now look at the functions f_1 and f_2, defined above.

If we define $F: \mathcal{P}[0,1] \longrightarrow \mathcal{P}[0,1]$ such that $F(A) = f_1(A) \bigcup f_2(A)$ where $\mathcal{P}[I,\infty]$ denote power set of $[0,1]$ and $f_i(A), i = 1,2$ is image of set A under f_i. Then it can be seen that $\mathcal{C} = \lim_{n\to\infty} F^n([0,1])$ where $F^n[0,1] = F(F^{n-1}[0,1])$.

In general system of functions $\{f_1, f_2, \ldots, f_n\}$ is called **iterated function system**, which or their generalizations form the basis of Fractal Image Compression.

3.2 Theoretical Foundations of FIC

The theorem which lies at the heart of FIC is called *collage theorem*. We will state and prove this theorem in this section. Let $\langle X, d\rangle$ be a complete metric space, then we will concentrate on the space of non-empty compact subsets of X, denoted by $\mathcal{H}(X)$, where

$$\mathcal{H}(X) = \{S \subset X : S \; is \; compact, S \neq \phi\}$$

If $A \in \mathcal{H}(X)$, let $\epsilon-$neighborhood of set A is

$$A_d(\epsilon) = \{x : d(x,y) < \epsilon \; for \; some \; y \in A\}$$

Then we define the **Hausdorff metric**, denoted by h_d, as

$$if \; A, B \in \mathcal{H}(X) \; then \; h_d(A,B) = \max\{\inf\{\epsilon : B \subset A_d(\epsilon)\}, \inf\{\epsilon : A \subset B_d(\epsilon)\}\}$$

It can be checked that $\langle \mathcal{H}(X), h_d\rangle$ is a complete metric a metric space.

Note: When the metric d under consideration is clear, then we will denote ϵ-neighborhood simply by $A(\epsilon)$ and Hausdorff metric by h.

Now let's look at the function f defined on reals, such that $f(x) = \frac{x}{2} \quad \forall x$. Let y be any real number then $\lim_{n\to\infty} f^n(y) = 0$, also note that $F(0) = 0$. These type of functions are *Lipschitz functions* and if the space on which they are defined is complete then they have a unique fixed point. In above example f is Lipschitz on $\mathcal{R}$ (with contractivity factor $\frac{1}{2}$), and 0 is its fixed point.

Definition: Let $\langle X, d\rangle$ be a metric space. A map $\omega : X \longrightarrow X$ is **Lipschitz** with Lipschitz factor s, if there exist a real positive value s such that

$$d(w(x), w(y)) < sd(x,y) \quad \forall x, y \in X$$

If the Lipschitz $s < 1$; ω is said be contractive with contractivity s.

An immediate consequence of the above definition is that if $f: X \longrightarrow X$ is Lipschitz then f is continuous. Another useful consequence is the following theorem

Theorem: Let $w_i: \mathcal{R}^2 \longrightarrow \mathcal{R}^2$ is contractive with contractivity factor $s_i, i = 1$ *to* n then

$$W: \mathcal{H}(\mathcal{R}^2) \longrightarrow \mathcal{H}(\mathcal{R}^2) \; where \; W(A) = \bigcup_{i=1}^{n} w_i(A) \;\; \forall A \in \mathcal{H}(\mathcal{R}^2)$$

is contractive in Hausdorff metric with contractivity $s = \max_{i=1,\ldots,n} s_i$.

Proof: We prove it for $n = 2$. By induction the general case can be proved. Let $A, B \in \mathcal{H}(\mathcal{R}^2)$ and $h(A, B) = \epsilon$. Then $A \subset B(\epsilon)$ *and* $B \subset A(\epsilon)$. Let $s = \max(s_1, s_2)$ and

$$A_i = w_i(A), B_i = w_i(B) \; and \; A' = W(A) = \bigcup_{i=1}^{2} A_i \; and \; B' = \bigcup_{i=2}^{2} B_i$$

Then note that

$$B_i \subset w_i(A(\epsilon)) \subset A_i(s_i\epsilon) \subset A_i(s\epsilon) \; and \; A_i \subset w_i(B(\epsilon)) \subset B_i(s_i\epsilon) \subset B_i(s\epsilon) \; \forall i$$

$\Rightarrow$ taking union over i's we get

$$B' \subset A'(s\epsilon) \; and \; A' \subset B'(s\epsilon)$$

i.e. $h(W(A), W(B)) \leq sh(A, B)$.

Note: As a consequence of this theorem, an IFS is contractive if all its component functions are contractive.

Now we state an important tool in FIC.

Theorem: (Fixed Point Theorem) Let f be contractive on a complete metric space $\langle X, d \rangle$ then it has a unique fixed point $x_f = f(x_f) = \lim_{n\to\infty} f^n(x) \; \forall x \in X$.

Proof: Let $x \in X$. Then for $n > m$ we have

$$d(f^n(x), f^m(x)) < sd(f^{n-1}(x), f^{m-1}(x)) < s^m d(f^{n-m}(x), x)$$

Apply triangle inequality repeatedly we get,

$$\begin{aligned} d(x, f^k(x)) &\leq d(x, f(x)) + d(f(x), f^2(x)) + \cdots + d(f^{k-1}(x), f^k(x)) \\ &\leq (1 + s + s^2 + \cdots + s^{k-1}) d(x, f(x)) \\ &\leq \frac{1}{1-s} d(x, f(x)) \end{aligned}$$

$\Rightarrow \{f^n(x)\}$ is Cauchy sequence $\forall x \in X$. Then by completeness of X, it has a limit point say x_f. Then by continuity of f,

$$x_f = \lim(f^n(x)) = f(\lim f^{n-1}(x)) = f(x_f)$$

Thus x_f is a fixed point of f.

Uniqueness: Let y be another fixed point. Then $d(f(y), f(x_f)) = d(y, x_f) < s\, d(y, x_f)$, a contradiction. This completes proof of the theorem.

Take the limit as $k \longrightarrow \infty$ in above derivation then we get

Theorem:(Collage Theorem) Under the hypothesis of Fixed point theorem,

$$d(x, x_f) \leq \frac{1}{1-s} d(x, f(x))$$

3.3 Fractal Image Compression

After the theoretical foundations, we move onto actual fractal image compression. Central problem of FIC is **image encoding problem** also called the *inverse problem*: Given a set S find an IFS whose fixed point is S. No completely general solution to this problem is available but the fixed point theorem and collage theorem provide some insight. By fixed point theorem we have

$$x_w = W(x_w) = \bigcup_{i=1}^{n} W_i(x_w)$$

i.e. the fixed point is constructed out of transformed copies of itself. By collage theorem we have

$$d(S, x_w) \leq \frac{1}{1-s} d(S, W(S))$$

i.e. we can construct S by pasting its images under certain contractive transformations and by uniqueness of the fixed point, it follows $S = W(S)$ then $S = x_w$. Also note that better is the fit between original set S and the collage, $W(S)$, closer will be the attractor to S. Even though this method can be applied to many naturally occurring objects, the class of images those can be encoded satisfactorily is restricted. So we need to generalize this method. It is more reasonable to assume that a part of image is similar to another part of the same image, and image compression is

achieved if we are able to remove this kind of redundancy in representation of parts of the image. This generalization uses *partitioned iterated function system*, defined formally as:

Definition:(PIFS) Let $\langle X, d \rangle$ be a complete metric space, and let $D_i \subset X$ for i = 1 to n. PIFS is collection of contractive maps w_i: $D_i \longrightarrow X$, for i = 1 to n.

Now we very briefly describe FIC method proposed by Arnaud Jacquin. Rough outline is as follows:[1]

- **Design of the fractal coding system**

  The three main issues involved in the design and implementation of fractal block coding system are as follows:

  1. (Image dependent) partitioning of the image.
  2. Choice of measure of distortion between two images.
  3. Specifications of the class of discrete contractive image transforms.

Jacquin has proposed following method of dealing with these issues.

1. **Image Partitioning**

   Image is partitioned into non-overlapping square cells of two types and of size $B \times B$ (parent range cells) and size $B/2 \times B/2$ (child range cells), thus forming two level partitioning. Decision of whether to split the parent cell or not is made by the coder so as to adjust itself to varying complexity of image structure.

2. **Distortion Measure**

   To measure the distortion between two image blocks, one from the original image and the other from approximating image, *mean square distortion* mean is used. So

$$d(I_{orig_B}, I_{appr_B}) = \sum_{0<i,j<B} (I_{orig}(x+i, y+j) - I_{appr}(x+i, y+j))^2$$

   where I_{orig_B} is the block B from the original image. Similarly I_{appr_B} is the block taken from the approximation to the original.

[1] For further details refer to [8]

Distortion between two images is measured by *Peak to peak Signal to Noise Ratio (PSNR)*, define as:

$$PSNR = 10 \log_{10} \left(\frac{dr(I_{orig})^2}{d(I_{orig}, I_{appr})/r^2} \right)$$

where $dr(I)$ denotes dynamic range of the image I.

3. **Class of Discrete Image Transforms**

The transformations are of 2 types: geometric and massic.

- *Geometric transform:* Let $D > B$. Then averaging and sub-sampling of $D \times D$ block results into a $B \times B$ sized block. For a simple case of $D = 2 \times B$, this transform can be defined as (for simplicity assume both the blocks have their left hand bottom corner at origin. This can be easily generalized):

$$Q_{x,y} = \frac{P_{2x,2y} + P_{2x+1,2y} + P_{2x,2y+1} + P_{2x+1,2y+1}}{4}$$

where P and Q are blocks of size $D \times D$ and $B \times B$ resp.

- *Massic Transformations:*
  (a) Absorption at gray level g: All the pixels in the block are assigned same gray level value g..
  (b) Gray level shift by δg : For all pixel in the block the gray level values are shifted by gray level δg..
  (c) Contrast scaling by $\alpha \in [0, 1]$: For all pixels in the block the gray level values are multiplied by α.
  (d) Isometries : Eight isometries of a square. They are identity, rotations through 90, 180, 270 degrees, flip about the two diagonals and mid-horizontal and mid-vertical axes.

4. **Overview of Encoding Procedure**

Following is a brief description of encoding procedure.

(a) Partition the image into range blocks as described as mentioned above. Classify each range block as either a shade block, mid-range block or edge block.
(b) Choose a suitable domain pool and classify each domain block into one of the three classes mentioned above.

(c) Do the following for all range blocks

if (range block is a shade block)
Store mean pixel value for the block.

if (range block is mid-range block)
Search only the class of the mid-range domain blocks. Find contrast scaling factor $\alpha \in \{0.7, 0.8, 0.9, 1.0\}$ and gray level shift with appropriate domain block so as to minimize ms error described above.

if (range block is an edge block)
Search only the class of the edge domain blocks. Find contrast scaling factor $\alpha \in \{0.2, 0.3, \ldots, 0.9\}$ and gray level shift with appropriate domain block with appropriate isometry so as to minimize ms error described above.

We have slightly modified the above mentioned FIC method. Here are some of the differences:

1. There is no 2 level classification of range and domain blocks. Also the image is not subdivided into any sub-blocks. There are two reasons for this modifications: 1) This will give us more compression as we will not have to keep track of the size of the range block. 2) This will simplify the encoding algorithm.

2. Instead of classifying blocks into 3 classes: shade, mid-range and edge blocks, we divide the blocks only into 2 classes: smooth and non-smooth blocks. Two classification criteria are used for this classification. Again, the above stated reasons motivate us to do this modification.

   (a) **Variance of the block:** For each which is to be classified, we calculate its variance and it is compared with certain predetermined threshold value (THR).
   If (variance of the block < THR)
   then the block smooth
   else the block is not-smooth.
   When there is an edge in the block, then there is sharp variation in gray level values which will get reflected in the high value of variance. So variance can be used to detect edge block.

(b) **Sobel operator method:** First 3×3 Sobel's gradient operator is applied to the image (to find gradient value at boundary points image is padded with one layer of pixels). Its output image is, say, gradimage. Then 3×3 averaging operator is applied to the image. Its output image is, say, avgimage. Two parameters α and β are used in the classification.
α measures relative deviation of a pixel value from the average pixel value at that point.
β quantifies the degree of roughness of the block.
So criteria for smoothness of the block can be described as
count = 0
for all pixels in the block
do
if($\frac{|gradimage[i][j]|}{avgimage[i][j]} > \alpha$)
then count = count + 1
done
if (count < β)
then block is smooth
else block is not smooth.
By this method we are trying to detect edges comparing the gradient value against the background (average gray value at that point). The use of Sobel's operator is preferred for its simplicity.

3. **Computation of bit rate:**

- Since blocks are divided into 2 classes, to indicate the class we require 1 bit.
- If block is smooth then we need to store the average pixel value of that block, which can be in the range 0 to 255. So we require 8 bits for that.
- If block is not smooth then we need to store following:
  (a) Contrast scaling coefficient: requires 2 or 3 bits. We have used 2 sets from which the contrast scaling coefficient can take values. Observations are obtained when α takes values from the set having 8 elements, in that case we require 3 bits, and when α takes values from the set having 4 elements, we require 2 bits.
  (b) Gray level shift: can be in the range -255 to 255, so requires 9 bits to store that value.
  (c) Isometry: Requires 3 bits to store 8 isometries of a square.

(d) Domain block corner points : abscissa 8-bits and ordinate 8-bits.

So the bit rate in bits/pixel is given by

$$\frac{(N_s \times 9) + (DNUM - N_s) \times 31}{r^2 \times 8} \tag{3.1}$$

where N_s = number of smooth blocks, DNUM = Total number of range blocks
r = image width = image height

Original Image	Lenna
Resolution	256
Gray Levels	256 ($gl_{\min}$ = min gray level value = 0, $gl_{\max}$ = max gray level value = 255)
Range Block	size 8×8
Domain Pool	Domain block size 16×16 Vertical shift = Horizontal shift = 4
Block Classification	Smooth and rough
Transformations	
for smooth block	Absorption at mean gray level of range block
for rough block	Contrast scaling by : (Group A) $\alpha \in \{0.125.0.25, \ldots, 0.875\}$ or (Group B) $\alpha \in \{0.7, 0.8, 0.9, 1.0\}$ or (Group C) $\alpha \in \{0.4, 0.6, 0.75, 0.875\}$ Gray level shift by : $\delta g \in \{gl_{\min}, \ldots, gl_{\max}\}$ isometries: {0,1,...,7}

Table 3.1: Design specifications

3.4 Results

- **Specifications of FIC method used**

  Specifications of FIC method are as given in the table 3.1.

- **Observations when variance is used to classify the blocks**

  For observations on the resultsof the method that uses variance as classification criteria. and corresponding remarks refer to table 3.2 and figures from 3.1 to 3.3.

  Remarks:

  1. Blockyness in the decoded image decreases with the decreasing THR value. (Refer to figures 3.1 to 3.3). Note that the decoded images for THR

Figure 3.1: Decoded images : THR = 100, THR = 50

Figure 3.2: Decoded images : THR = 25, THR = 20

Figure 3.3: Decoded images : THR = 10

THR value	Group A (db)	Group B (db)	Group C (db)	% smooth blocks	comp. ratio
100	25.95	26.68	26.27	47.26	24.85
50	26.20	26.97	26.51	36.43	22.27
25	26.25	27.08	26.61	27.14	20.33
20	26.38	27.15	26.64	23.82	19.88
10	26.35	27.19	26.19	15.72	18.59

Table 3.2: PSNR values for variance method

value less than 25 are considerably less blocky. This indicates that in an image a deviation of 5 gray levels from the original gray level value may not be noticeable but beyond that it becomes noticeable and gives rise to blockyness in the decoded image.

2. Even though there is not much increase in PSNR values from THR = 100 to THR = 10, the quality of images is quite different. This indicates PSNR is not a good measure of image quality.
3. It has been observed that using more number of bits to represent contrast scaling factor need not necessarily improve the image quality.

	β							
α	0.2		0.3		0.4		0.5	
	Group A	Group B	Group A	Group B	Group A	Group B	Group A	Group B
0.1	26.14	26.89	26.13	26.98	26.12	26.87	26.0	26.73
0.15	26.12	26.84	26.07	26.75	25.90	26.53	25.40	26.06
0.2	26.00	26.66	25.87	26.59	25.39	26.15	25.01	25.72

Table 3.3: PSNR values for Sobel's gradient operator method

- **Observations for method using Sobel's Gradient operator**

  For observations refer to the table (3.3).

  Remarks:

Figure 3.4: Decoded images : $\alpha = 0.1$ and $\beta = 0.4$

Figure 3.5: Decoded images : $\alpha = 0.15$ and $\beta = 0.2, 0.4$

Figure 3.6: Decoded images : $\alpha = 0.2$ and $\beta = 0.2, 0.4$

1. Of the two parameters α and β, α is more dominant since change in alpha value changes image quality perceivably. (Refer to figures3.4 to 3.6)
2. Decreasing the α value too much does not improve the quality of decoded image much. Also comparing the best outputs given by above two methods we can say that variance method is better than the Sobel operator method. Further investigations are needed to justify this observation.

- **Results when range block size is 4×4:**

  To obtain this result we have used THR = 25. The PSNR value for the decoded image is 33.7 db and compression ratio achieved is 5.23. The output is as in figure (3.7).

Figure 3.7: THR = 25 and range block size 4×4 and the original image

Bibliography

[1] Raphael C. Gonzalez and Richard E. Woods, "Digital Image Processing",[1993], Addison-Wesle Publishing Company.

[2] Khalid sayood, "Introduction to Data Compression", [1996], Morgan Kaufmann Publishing Inc.

[3] Geoffrey M. Davis, Aria Nostratinia, "Wavelets Based Image Compression: an overview ", Texas Instruments Tech. Reports.

[4] W. D. Ray and R. M. Driver,"Further Decomposition of Karhunen- Loeve series representation of stationary random process", IEEE Transactions on Inform. Theory, Vol. IT-16, Nov.1970, pp. 663-668.

[5] Gregory Wallace,"The JPEG still picture compression standard", Communications of ACM, Vol. 34, Number 4, pp. 30-44.

[6] Barnsley Michael, "Fractals Everywhere",[1986], Academic Press,Inc..

[7] Yuval Fisher, "Fractal Image Compression",[1995], SpringerVerlag NewYork,Inc.

[8] Jacquin Arnaud, "Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformation", IEEE Transactions on Image Processing, Vol. 1, No. 1, January 1992, pp. 18-30.