UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA ELÉTRICA
PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

# COMPARAÇÃO ENTRE OS MÉTODOS DE COMPRESSÃO FRACTAL E JPEG2000 EM UM SISTEMA DE RECONHECIMENTO DE ÍRIS

**Sandreane Poliana Silva**

**Agosto**

**2008**

Dados Internacionais de Catalogação na Publicação (CIP)

S586c Silva, Sandreane Poliana, 1983-
Comparação entre os métodos de compressão fractal e JPEG 2000 em um sistema de reconhecimento de íris / Sandreane Poliana Silva. - 2008.
103 f. : il.

Orientador: Antônio Cláudio Paschoarelli Veiga.
Dissertação (mestrado) – Universidade Federal de Uberlândia, Programa de Pós-Graduação em Engenharia Elétrica.
Inclui bibliografia.

1. Processamento de imagens - Técnicas digitais -Teses. 2. Compressão de imagens - Teses. I. Veiga, Antônio Cláudio Paschoarelli. II. Universidade Federal de Uberlândia. Programa de Pós-Graduação em Engenharia Elétrica. III. Título.

CDU: 621.311

Elaborado pelo Sistema de Bibliotecas da UFU / Setor de Catalogação e Classificação

**UNIVERSIDADE FEDERAL DE UBERLÂNDIA**
**FACULDADE DE ENGENHARIA ELÉTRICA**
**PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA**

# COMPARAÇÃO ENTRE OS MÉTODOS DE COMPRESSÃO FRACTAL E JPEG2000 EM UM SISTEMA DE RECONHECIMENTO DE ÍRIS

**Sandreane Poliana Silva**

Texto da dissertação apresentada à Universidade Federal de Uberlândia, perante a banca de examinadores abaixo, como parte dos requisitos necessários à obtenção do título de Mestre em Ciências. Aprovada em 14/08/2008.

## Banca examinadora:

Prof. Dr. Antônio Cláudio Paschoarelli Veiga, Orientador(UFU)

Prof[a]. Dr[a]. Edna Lúcia Flôres (UFU)

Prof. Dr. Giberto Arantes Carrijo (UFU)

Prof. Dr. João Paulo Inácio Ferreira Ribas (UFMT)

# COMPARAÇÃO ENTRE OS MÉTODOS DE COMPRESSÃO FRACTAL E JPEG2000 EM UM SISTEMA DE RECONHECIMENTO DE ÍRIS

**Sandreane Poliana Silva**

Texto da dissertação apresentada à Universidade Federal de Uberlândia como parte dos requisitos para obtenção do título de Mestre em Ciências.

| | |
|---|---|
| Prof. Dr. A. C. Paschoarelli V. | Prof. Ph.D.Darizon Alves de Andrade |
| Orientador | Coordenador do curso de Pós-Graduação |

# Agradecimentos

Ao finalizar mais uma etapa da minha vida, gostaria de agradecer às pessoas que de uma forma ou de outra me acompanharam no desenvolvimento deste trabalho.

Antes de tudo agradeço a Deus, sem Ele, eu nada conseguiria. Muitas vezes, em meio às dificuldades, quedas e frustrações só a fé conseguiu me amparar e erguer.

À minha família, que mesmo à distância, sempre foi meu maior alicerce nesta caminhada, me dando força, me ouvindo e me aconselhando. Principalmente à minha mãe Sandra, pelas inúmeras horas ao telefone ouvindo minhas reclamações, meus desabafos e sempre tentando me animar. Nunca deixaram de acreditar em mim e isso, com certeza, foi o que mais me incentivou a seguir em frente.

Ao meu namorado Rodrigo, pessoa maravilhosa que Deus colocou na minha vida, minha eterna gratidão, não só pela luta diária como professor paciente de programação, mas principalmente pelo carinho, cuidado, apoio e companheirismo que sempre dedicou a mim.

Aos meus amigos de forma geral, os de casa, os da farra, aqueles que são para toda hora, os do trabalho, os da faculdade e mesmo os que estão distante, pelo apoio, incentivo e por compreender as muitas vezes que estive ausente em momentos necessários para dedicar-me aos meus estudos. Pelo

ombro amigo oferecido por muitos e até por aturarem meu mau humor em decorrência dos problemas surgidos no desenvolver deste trabalho. Dentre estes, em especial agradeço à minha amiga Fernanda pela contribuição direta com correções gramaticais e traduções.

Ao meu orientador Paschoarelli pela paciência, pelas palavras sábias quando surgiram as muitas dúvidas e principalmente pelo voto de confiança, por ter acreditado na minha capacidade e por me incentivar sempre a seguir em frente.

Aos professores do departamento que me acompanharam durante o curso. À professora Edna pela disposição sempre em ajudar e pelas palavras de incentivo. À Milena pela parceria nos trabalhos, foi gratificante trabalhar em conjunto com uma pessoa tão dedicada e responsável.

Aos professores participantes desta banca por disponibilizarem seu tempo, contribuindo na avaliação deste trabalho.

A todos aqui citados, com nomes ou não, e àqueles que podem ter sido esquecidos, obrigada mais uma vez, todos são, cada um à sua maneira importantes para mim.

*" Tenho posto o Senhor continuamente diante de mim, por isso que Ele está à minha mão direita, nunca vacilarei."* ***Salmos,*** 16 : 8**.**

# Resumo

***Silva***, S.P. *Investigando os Efeitos da Compressão Fractal num Sistema de Reconhecimento de Íris*, FEELT-UFU, Uberlândia-MG, 2008.

Atualmente vive-se na era digital, por isso a manipulação de dados e imagens é frequente todos os dias. Devido ao problema de espaço para armazenamento dessas imagens e tempo de transmissão, foram desenvolvidas várias técnicas de compressão, e um grande desafio é fazer com que essas técnicas tragam bons resultados em termos de taxa de compressão, qualidade da imagem e tempo de processamento. A técnica de compressão Fractal desenvolvida por *Fisher*, foi descrita, implementada e testada neste trabalho e trouxe ótimos resultados, e melhoria considerável em termos de tempo de execução, que foi bastante reduzido.

Outra área que vem se destacando é o uso de técnicas biométricas para reconhecimento de pessoas. Uma técnica muito usada é o reconhecimento de íris que tem mostrado bastante confiabilidade. Assim, aliar as duas tecnologias traz grandes benefícios. No presente trabalho, imagens de íris foram comprimidas pelo método aqui implementado e foram realizadas simulações da técnica de reconhecimento de íris desenvolvida por *Maseck*. Os resultados mostram que é possível comprimir fractalmente as imagens sem prejudicar o sistema de reconhecimento. Comparações foram realizadas e foi possível

perceber que mesmo havendo mudanças nos pixels das imagens, o sistema permanece bastante confiável, trazendo vantagens em espaço de armazenamento.

**Palavras Chave**

Compressão Fractal, quadtree, sistema biométrico, reconhecimento de íris.

# Abstract

***Silva***, S.P. *Investigating the Effects of the Fractal Compression in a Íris Recognition System*, FEELT-UFU, Uberlândia-MG, 2008.

Currently living in the digital age, so the manipulation of data and images is often all day. Due to the problem of space for storage of pictures and time of transmission, many compression techniques had been developed, and a great challenge is to make these techniques to bring good results in terms of compression rate, picture quality and processing time. The Fractal Compression technique developed by *Fisher*, was described, implemented and tested in this work and it brought great results, and considerable improvement in terms of execution time, which was rather low.

Another area that has been emphasizing is the use of biometric techniques to the people recognition. A very used technique is the iris recognition that has shown enough reliability. Thus, connecting the two technologies brings great benefits. In this work, images of iris were compressed by the method implemented here and were made simulations of the technique iris recognition developed by *Libor Maseck*. The results show that it is possible to compress fractally the images without damage the recognition system. Comparisons were made and was possible realize that even with changes in pixels of images, the system remains very reliable, bringing benefits to storage space.

**Keywords**

Fractal Compression, quadtree, biometric system, íris recognition.

# Sumário

# Lista de Figuras

# Lista de Tabelas

# Simbologia

IFS – Iterated Function Systems (Sistema de Funções Iteradas)

PIFS – Partitioned Iterated Function Systems (Sistema de Funções Iteradas Particionadas)

JPEG – Joint Photographic Experts Group

PNG – Portable Network Graphics

JFIF – Joint File Interchange Format

DCT – Transformada Cosseno Discreta

UIT – União Internacional das Telecomunicações

ISO – International Organization for Standardization (Organização Internacional de Normalização)

GIF – Graphics Interchange Format

RMS – Raíz Média Quadrática

SNR – Relação Sinal Ruído

PSNR – Relação Sinal Ruído de Pico

$D_i$ – Domain-block (bloco domínio)

$R_i$ – Range-block (bloco molde)

$s - i$ – brilho

$o_i$ – contraste

$M_i$ – Média

$V_i$ – Variância

$D_x$ – Posição $x$ do bloco domínio escolhido

$D_y$ – Posição $y$ do bloco domínio escolhido

RAM – Random Access Memory

API – Application Programming Interface

bpp – bits por pixel

dB – decibéis

ms – milisegundos

TH – Transformada Hough

CASIA – Chinese Academy of Sciences- Institute od Automation (banco de imagens da Academia Chinesa de Ciências - Instituto de Automação.

UBIRIS – A noisy iris image database (banco de imagens usado nas simulações)

DH – Distância de Hamming

FAR – False Accept Rate (taxa de erro de falsa aceitação)

FARm – Taxa média de erro de falsa aceitação

FRR – False Reject Rate (taxa de erro de falsa rejeição)

FRRm – Taxa média de erro de falsa rejeição

DSPG – Digital Signal Processing Group

HV – Horizontal and Vertical Partitioning

# Capítulo 1

## 1.1 INTRODUÇÃO

Nos últimos anos, o armazenamento e a manipulação de imagens tornaram-se tarefas altamente computadorizadas. Fotografias, vídeos e outros tipos de gráficos são usados em grande parte das aplicações.

Contudo, um fator que limita a utilização das imagens ou vídeos é o seu tamanho: é necessário muito espaço para armazenar imagens grandes, e para transmití-las em uma rede é necessário uma elevada largura de banda. Esta restrição levou ao desenvolvimento de técnicas de compressão para armazenar imagens em arquivos menores.

Com as aplicações práticas dos trabalhos teóricos iniciados na década de 40 por *C.E. Shannon*, que desenvolveu a chamada *Teoria da Informação* foi possível um grande crescimento da área até os dias atuais. Hoje, a compressão de imagens é usada por exemplo na manipulação das resoluções espaciais, sensoriamento remoto, evolução das padronizações de transmissão de TV, videoconferências, imageamento médico, transmissão de FAX, operações militares e espaciais, entre outras.

Dentre algumas dessas técnicas, a *Compressão Fractal* baseia-se no fato de que qualquer imagem natural contém redundância, que por sua vez pode

ser eliminada, resultando em uma compressão da imagem, podendo portanto, facilitar transmissão e diminuir gasto de memória em armazenamento.

Até pouco tempo atrás o termo *fractal* era visto somente como uma ligação entre a *geometria*, a *arte* e a *matemática*. No entanto a observação de grande auto-similaridade nas imagens de fractais levou ao desenvolvimento de uma técnica de compressão bastante eficiente, uma vez que a *Compressão Fractal* 'transforma' a imagem em um conjunto de transformações afim. Sendo assim, imaginou-se aplicar tal compressão em imagens que necessitam de menos espaço de armazenamento sem, no entanto, prejudicar a funcionalidade das mesmas.

Atualmente nota-se uma preocupação muito grande com segurança, e portanto, as técnicas biométricas de reconhecimento de padrões são amplamente estudadas e utilizadas. Dentre as várias, uma que se destaca, por ser de grande confiabilidade sem no entanto gerar grandes gastos é o reconhecimento de íris, isso porque a íris não se modifica após os primeiros meses de vida da pessoa, a íris esquerda é diferente da direita em uma mesma pessoa e inclusive gêmeos idênticos possuem íris diferentes.

No entanto, bancos de dados muito grandes, podem atrapalhar ou dificultar o sistema de reconhecimento ou mesmo tomar muito espaço de armazenamento. Assim, aliando as duas técnicas, pode-se obter bons resultados, uma vez que, se tais imagens forem comprimidas sem prejudicar o sistema de reconhecimento de íris, o espaço para armazenamento seria diminuido, tornando-se uma vantagem.

## 1.2 Objetivos Deste Trabalho

Os principais objetivos deste trabalho são:

- Estudar e descrever detalhadamente sistemas de compressão de imagens baseados na teoria de fractais;

- Implementação de um algoritmo de compressão fractal em uma linguagem eficiente com a finalidade de diminuir o tempo de simulação, como por exemplo, em relação ao Matlab;

- Apresentação de resultados de diversas simulações que comprovam a eficiênca do método de compressão fractal;

- Aplicar o método de compressão fractal a imagens de íris considerando diversas taxas de compressão e analisar os efeitos em termos de taxa de falsa aceitação e falsa rejeição;

- Comparar esses efeitos da compressão quando se utiliza também a técnica JPEG2000.

## 1.3 Estrutura Da Dissertação

Cada capítulo contém parte relevante das informações estudadas para o desenvolvimento desde trabalho. Os capítulos estão estruturados da seguinte forma:

- **Capítulo1**: Breve introdução, objetivos e estrutura do trabalho. Finalmente são realizadas as considerações finais sobre este capítulo.

- **Capítulo 2**: Este capítulo apresenta uma idéia geral do que vem a ser os fractais, os exemplos mais comuns e suas diversas aplicações. Apresenta também as principais definições matemáticas necessárias ao embasamento teórico da compressão fractal, conceitos tais como espaços

métricos, sejam eles abertos, fechados, conexos, desconexos, completos, compactos, tipos de métricas e suas principais propriedades. Esses temas são essencias para a construção do espaço métrico dos fractais. Finalmente são realizadas as considerações finais sobre este capítulo.

- **Capítulo 3**: Mostra uma idéia geral de diversos tipos de compressão de imagens e a evolução de algumas técnicas de compressão bem como suas vantagens e/ou desvantagens. Mostra dois tipos de particionamento para compressão fractal, analisa e compara os mesmos. São mostrados também a implementação do método utilizando particionamento quadtree e resultados de testes realizados com imagens quadradas padrão. Finalmente são realizadas as considerações finais sobre este capítulo.

- **Capítulo 4**: Estudo de um sistema de reconhecimento de íris, aplicação do método de compressão fractal implementado neste trabalho e da técnica JPEG2000 a essas imagens e a análise dos efeitos da compressão em termos de taxa de falsa aceitação e falsa rejeição. São mostrados os resultados obtidos. Finalmente são realizadas as considerações finais sobre este capítulo.

- **Capitulo 5**: Conclusões gerais a partir dos resultados obtidos nos testes realizados neste trabalho, as contribuições desta dissertação e as propostas para trabalhos futuros.

## 1.4 Considerações Finais Deste Capítulo

Este capítulo apresentou uma breve introdução sobre compressão, os objetivos e a estrutura deste trabalho.

O próximo capítulo mostra uma idéia geral do vem a ser os fractais, os exemplos mais comuns e diversas aplicações. Apresenta também as principais definições matemáticas necessárias ao embasamento teórico da compressão fractal, conceitos tais como espaços métricos, sejam eles abertos, fechados, conexos, desconexos, completos, compactos, tipos de métricas e suas principais propriedades. Esses temas são essencias para a construção do espaço métrico dos fractais.

# Capítulo 2

# FRACTAIS

## 2.1 Introdução Aos Fractais

A *geometria* é um ramo da matemática que estuda as diversas formas do universo sejam elas planas ou espaciais. A palavra geometria vem do grego e significa *medida da terra*. Dentre os diversos tipos pode-se destacar as geometrias algébrica, descritiva, analítica, diferencial, euclidiana e fractal. Durante séculos, os objetos e os conceitos da filosofia e da geometria euclidiana foram considerados como os que melhor descreviam o mundo em que o ser humano vive. A descoberta de geometrias não-euclidianas introduziu novos objetos que representam certos fenômenos do Universo, tal como se passou com os fractais. Assim, considera-se hoje que tais objetos retratam formas e fenômenos da Natureza.

A *geometria fractal* é o ramo da matemática que estuda as propriedades e comportamento dos fractais. Descreve muitas situações que não podem ser explicadas facilmente pela geometria clássica, e foi aplicada em ciência, tecnologia e arte gerada por computador.

Um fractal é um objeto geométrico que pode ser dividido em partes, cada

uma das quais semelhante ao objeto original [1]. Diz-se que os fractais têm infinitos detalhes, são geralmente auto-similares e independem de escala. Em muitos casos um fractal pode ser gerado por um padrão repetido, tipicamente um processo recorrente ou iterativo.

O termo *fractal* foi cunhado em 1975 por *Benoit Mandelbrot*, matemático francês nascido na Polônia, que descobriu a geometria fractal na década de 70 do século XX, a partir do adjetivo latino *fractus*, do verbo *frangere*, que significa quebrar [1].

Este capítulo mostra uma idéia geral do que vem a ser os fractais, os exemplos mais comuns e diversas aplicações. Apresenta também as principais definições matemáticas necessárias ao embasamento teórico da compressão fractal, conceitos tais como espaços métricos, sejam eles abertos, fechados, conexos, desconexos, completos, compactos, tipos de métricas e suas principais propriedades. Esses temas são essencias para a construção do espaço métrico dos fractais.

## 2.2 Exemplos de Fractais

Em 1872, *Karl Weierstrass* encontrou o exemplo de uma função com a propriedade de ser contínua em todo seu domínio, mas em nenhuma parte diferenciável. O gráfico desta função é chamado atualmente de fractal. Em 1883, *George Cantor* publicou um trabalho em que ele descreveu um conjunto intrigante do ponto de vista matemático, conhecido como conjunto de Cantor [1], mostrado na Figura 2.1. Esse conjunto pode ser construído executando os seguintes passos: (1) construa um segmento de reta; (2) divida esse segmento em três partes iguais e elimine a parte central; e (3) repita o passo (2) em cada segmento obtido e, assim, sucessivamente.

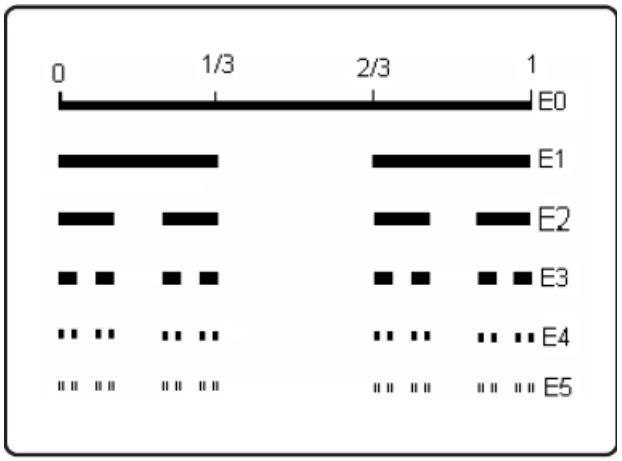


Figura 2.1: Conjunto de Cantor

### 2.2.1 Curva de Peano

Curvas de Peano são curvas descritas pelo matemático italiano *Giuseppe Peano* de forma a preencher completamente um espaço bidimensional (como um quadrado) ou tridimensional[2]. O processo de construção dessa curva é iterativo: (1) o processo iterativo inicia-se com um segmento de reta. (2) divide-se esse segmento em três partes iguais. (3) sobre o traço médio, constrói-se um retângulo bissectado pelo traço, formando dois quadrados com lado igual ao traço que lhes originou. (4) em cada segmento dos nove restantes, repetem-se os passos 2 e 3, e assim sucessivamente. Veja um exemplo de Curva de Peano bidimensional na Figura 2.2:

### 2.2.2 Triângulo de Sierpinski

O triângulo de Sierpinski foi descrito por *Waclaw Sierpinski* em 1915 e obtém-se como limite de um processo recursivo[3]. O processo se inicia de um triângulo equilátero. Em seguida unem-se os pontos médios de cada lado do triângulo formando 4 triângulos cujos lados estão ligados e finalmente

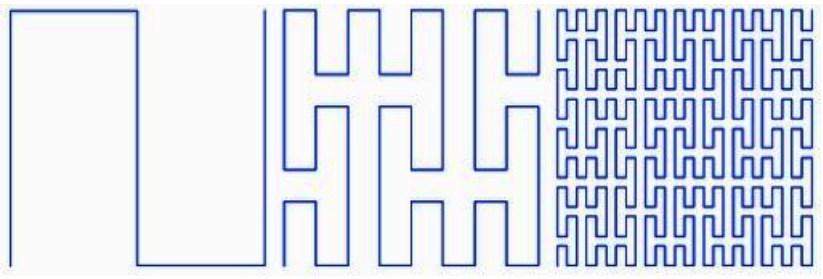

Figura 2.2: Curva de Peano

retira-se o triângulo central. A recursão consiste em repetir indefinidamente o procedimento anterior em relação a cada um dos triângulos obtidos. A Figura 2.3 mostra um exemplo do triângulo de Sierpinski:

Figura 2.3: Triângulo de Sierpinski

### 2.2.3 Conjunto de Mandelbrot

O conjunto de Mandelbrot, em sua representação gráfica, pode ser dividido em um conjunto infinito de figuras pretas, sendo a maior delas um cardióide localizado no centro do plano complexo. Existe uma infinidade de quase-círculos (o maior deles é a única figura que, de fato, é um círculo exato e localiza-se à esquerda do cardióide) que tangenciam o cardióide e variam de tamanho com raio tendendo assintoticamente a zero. Cada um desses círculos tem seu próprio conjunto infinito de pequenos círculos cujos raios também tendem assintoticamente a zero. Esse processo se repete infinitamente, gerando um fractal [4]. A Figura 2.4 mostra o conjunto de Mandelbrot.

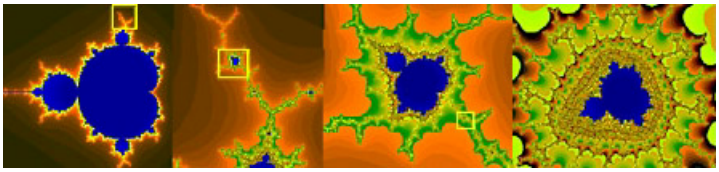

Figura 2.4: Conjunto de Mandelbrot

### 2.2.4 Conjunto de Julia

O conjunto de Júlia foi criado por *Gaston Julia*. Esse conjunto é uma família de conjuntos de fractais que surgiram com o estudo do comportamento dos números complexos [5]. Na verdade, a cada ponto do plano complexo corresponde um conjunto de Julia diferente. Existem uma infinidade de conjuntos de Julia. Um exemplo está na Figura 2.5.

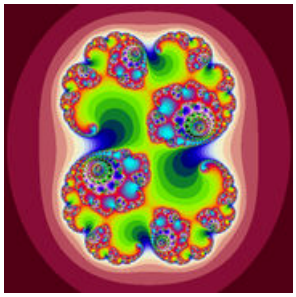

Figura 2.5: Uma das variedades do conjunto de Julia

### 2.2.5 Curva de Koch

A curva de Koch é uma curva geométrica e um dos primeiros fractais a serem descritos. Apareceu pela primeira vez em um artigo em 1906 de autoria do matemático sueco *Helge Von Koch* [4]. Sua construção também é a partir de um processo iterativo e começa com um segmento de reta. (1) constrói-se um segmento de reta. (2) divide-se o segmento de reta em três segmentos de igual comprimento. (3) desenha-se um triângulo equilátero, em que o

segmento central, referido no primeiro passo, servirá de base. (4) apaga-se o segmento que serviu de base ao triângulo do segundo passo. Depois de realizado isto, o resultado será semelhante à secção longitudinal de um 'chapéu de bruxa'. Procedendo da mesma forma para cada um dos quatro segmentos que ficam, formam-se dezesseis novos segmentos menores. A Figura 2.6 representa as seis primeiras etapas da construção da curva de Koch.

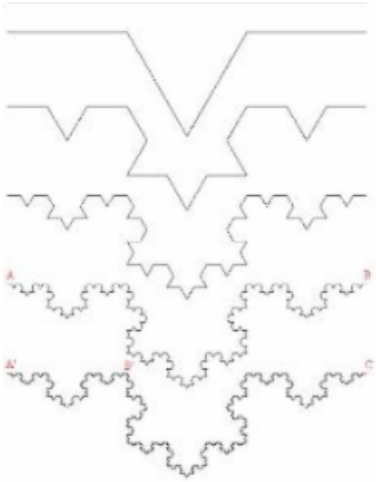

Figura 2.6: Curva de Koch

## 2.3 Aplicações dos Fractais

Nos últimos 20 anos, a geometria fractal e seus conceitos têm se tornado uma ferramenta central em muitas ciências como geologia, meteorologia e outras. Ao mesmo tempo, fractais são do interesse de designers gráficos e cineastas pela sua habilidade de criar formas novas e mundos artificiais mais realistas. Na Computação Gráfica, fractais, entre outras coisas, são utilizados para representar elementos da Natureza como crateras, planetas, costas, superfícies lunares, plantas, ondulações em águas, representação de

nuvens e também são de grande importância para a criação de efeitos especiais em filmes, como por exemplo a criação do *planeta Gênesis* no filme *Jornada nas Estrelas 2*. Algumas aplicações de fractais são comentadas a seguir[6]:

- Em sala de aula: as formas dos fractais estão presentes em várias formas não só reais mas também em vários ramos da matemática. Eles podem ser usados para ilustrar temas tais como semelhança de figuras, ampliação, redução, simetria, transformações geométricas, gráficos de funções, dentre outros. Isso desperta curiosidade e interesse facilitando o aprendizado;

- Fractais nas Ciências: a geometria fractal ajuda a aproximar as ciências naturais e a computação da pesquisa matemática, e é utilizada como instrumento principal em várias ciências como geologia e meteorologia. Na biologia contribui no estudo da influência da superfície irregular das proteínas nas iterações moleculares. Na geografia nos modelos de crescimento demográficos;

- Computação gráfica: criação de novas formas e mundos artificiais mais realistas, e na representação de elementos da natureza que a geometria tradicional não pode representar.

### 2.3.1 Compressão de Imagens

As formas similares dos fractais permitem que eles sejam representados por meio de trasformações afins, o que ocupa um espaço de armazenamento relativamente pequeno. Sendo assim, a compressão fractal é uma técnica que tem sido bastante usada, uma vez que permite taxas de compressões altas sem ocasionar grandes perdas [7].

Neste âmbito, este trabalho tem por objetivo realizar a compressão fractal em imagens de íris, analisando o desempenho de um método de reconhecimento de íris quando as imagens são comprimidas.

## 2.4 Espaços

Definir os principais conceitos matemáticos nos pelas suas principais propriedades e características, é de grande importância antes de partir para a Compresssão Fractal propriamente dita. Alguns conceitos Matemáticos básicos usados no presente trabalho, não foram detalhados por questão de espaço, mas podem ser facilmente encontrados no livro de Domingues [8].

A Geometria Fractal se concentra na estrutura dos subconjuntos dos espaços geométricos. Assim, denotando o espaço por $X$ pode-se nele contruir fractais, uma vez que, por hora considera-se um fractal como sendo um subconjunto do espaço. Como exemplos podem ser citados a reta real $X = \Re$, o conjunto das funções contínuas contido no intervalo fechado [0, 1], o plano euclidiano $X = \Re$, o plano complexo $C$ e a esfera de Riemann $X = \hat{C}$, que formalmente é $C \cup \infty$ [8].

**Definição 2.1** *Um espaço $X$ é um conjunto. Os pontos deste espaço são elementos deste conjunto [8].*

## 2.5 Espaços Métricos

**Definição 2.2** *Um espaço métrico $(X, d)$ é um espaço $X$ munido da função real $d : X_x X \rightarrow R$, que mede a distância entre um par de pontos $x, y$ em $X$. A função $d$ obedece aos seguintes axiomas:*

1. $d(x, y) = d(y, x), \forall x, y \in X$;

*2.* $0 < d(x, y) < \infty, \forall x, y \in X, x \neq y$*;*

*3.* $d(x, x) = 0, \forall x \in X$*;*

*4.* $d(x, y) \leq d(x, z) + d(z, y), \forall x, y, z \in X$*;*

*Assim, a função d é chamada de métrica.*

**Definição 2.3** *Duas métricas* $d_1$ *e* $d_2$ *no espaço* $X$ *são chamadas de métricas equivalentes se existem constantes* $0 < c_1 < c_2 < \infty$ *tal que* $c_1 d_1(x, y) \leq d_2(x, y) \leq c_2 d_1(x, y), \forall (x, y) \in X_x X$*.*

Um par de métricas equivalentes pode fornecer uma noção de quanto os pontos podem estar próximos ou distantes. Isto é como imaginar que existe um norma para limitar a deformação no espaço, contudo, as distâncias foram medidas antes e depois da deformação. O conceito de métrica equivalente é importante porque permite utilizar uma métrica alternativa mais simples de se manipular.

**Definição 2.4** *Dois espaços métricos* $(X_1, d_1)$ *e* $(X_2, d_2)$ *são equivalentes se existe uma função* $h : X_1 \to X_2$*, bijetora, tal que a métrica* $\tilde{d_1}$ *de* $X_1$ *definida por* $\tilde{d_1}(x, y) = d_2(h(x), h(y)), \forall x, y \in X_1$ *é equivalente a* $d_1$*.*

O conceito de espaços métricos equivalentes é importante porque pode facilitar o estudo das propriedades de um espaço métrico operando-se no seu equivalente mais simples.

**Definição 2.5** *A função* $f : X_1 \to X_2$ *de um espaço métrico* $(X_1, d_1)$ *para um espaço métrico* $(X_2, d_2)$ *é contínua se, para cada* $\epsilon > 0$ *e* $x \in X_1$*, existe* $\delta > 0$ *tal que* $d_1(x, y) < \delta \Rightarrow d_2(f(x), f(y)) < \epsilon$*.*

## 2.6 Algumas Propriedades dos Espaços Métricos

Existe um número grande de propriedades e subconjuntos dos espaços métricos que são usados para descrever conjuntos fractais. Aqui são introduzidas algumas características e propriedades topológicas desses espaços tais como os conceitos de *aberto, fechado, limitado, completo, compacto e perfeito.*

**Definição 2.6** *A sequência* $(x_n)_{n=1}^{\infty}$ *de pontos no espaço métrico* $(X, d)$ *é chamada sequência de Cauchy se, para dado* $\epsilon > 0$*, há um inteiro* $N > 0$*, tal que* $d(x_n, x_m) < \epsilon, \forall n, m > N$

**Definição 2.7** *A sequência* $(x_n)_{n=1}^{\infty}$ *de pontos no espaço métrico* $(X, d)$ *é chamada sequência Convergente para o ponto* $x \in X$ *se, para dado* $\epsilon > 0$ *existe um inteiro* $N > 0$*, tal que* $d(x_n, x) < \epsilon, \forall n > N$ *. Neste caso, o ponto* $x \in X$ *para o qual a sequência converge é chamado de limite da sequência e utiliza-se a notação* $x = \lim_{n\to\infty} x_n$

**Teorema 2.1** *Se uma sequência de pontos* $(x_n)_{n=1}^{\infty}$ *em um espaço métrico* $(X, d)$ *converge para um ponto* $x \in X$*, então* $(x_n)_{n=1}^{\infty}$ *é sequência de Cauchy.*

**Definição 2.8** *Um espaço métrico* $(X, d)$ *é completo se toda sequência de Cauchy* $(x_n)_{n=1}^{\infty}$ *em* $X$ *tem limite* $x \in X$*.*

Em outras palavras, existe, no espaço, o ponto $x$ para o qual a sequência de Cauchy converge.

**Definição 2.9** *Seja* $S \subset X$ *um subconjunto do espaço métrico* $(X, d)$*. O ponto* $x \in X$ *é chamado de ponto limite de* $S$ *se existe uma sequência* $(x_n)_{n=1}^{\infty}$ *de pontos* $x_n \in S$ *tal que* $x = \lim_{n\to\infty} x_n$*.*

**Definição 2.10** *Seja $S \subset X$ um subconjunto do espaço métrico $(X, d)$. O feixe de $S$, denotado por $\bar{S}$ é definido como $S = S \cup$ pontos limite de $S$. $S$ é fechado se contém todos os seus pontos limite.*

## 2.7 Conjuntos Compactos, Conjuntos Limitados, Interiores e Fronteiras

**Definição 2.11** *Seja $S \subset X$ um subconjunto do espaço métrico $(X, d)$. $S$ é compacto, se toda sequência infinita $(x_n)_{n=1}^{\infty}$ em $S$ contém uma subsequência limitada em $S$.*

**Definição 2.12** *Seja $S \subset X$ um subconjunto do espaço métrico $(X, d)$. $S$ é limitado se existe um ponto $a \subset X$ e um número $R > 0$ tal que $d(a, x) < R$, $\forall x \in S$.*

**Definição 2.13** *Seja $S \subset X$ um subconjunto do espaço métrico $(X, d)$. $S$ é totalmente limitado se, para cada $\epsilon > 0$, existe um conjunto finito de pontos $(y_1, y_2, ..., y_n) \subset S$, tal que, sempre que $x \in X$, $d(x, y_i) < \epsilon$ para cada $y_i \in (y_1, y_2, ..., y_n)$.*

**Teorema 2.2** *Seja $(X, d)$ um espaço métrico completo. Seja $S \subset X$. Então $S$ é compacto, se e somente se, é fechado e totalmente limitado.*

**Definição 2.14** *Seja $S \subset X$ um subconjunto do espaço métrico $(X, d)$. $S$ é aberto se para cada $x \in S$ existe um $\epsilon > 0$ tal que $B(x, \epsilon) = (y \in X : d(x, y) \leq \epsilon) \subset S$.*

Observação: $B(x, \epsilon) = (y \in X : d(x, y) \leq \epsilon)$ denota a bola fechada de raio $\epsilon > 0$, centrada em $x$.

**Definição 2.15** *Seja $S \subset X$ um subconjunto do espaço métrico $(X, d)$. Um ponto $x \in X$ é um ponto fronteira de $S$ se para todo $\epsilon > 0$, a $B(x, \epsilon)$ contém um ponto em $x$ menos $S$ e um ponto em $S$. O conjunto de todos os pontos fronteira em $S$ é chamado de fronteira de $S$ e é denotado por $\partial S$.*

**Definição 2.16** *Seja $S \subset X$ um subconjunto do espaço métrico $(X, d)$. Um ponto $x \in X$ é chamado de ponto interior de $S$ se existe $\epsilon > 0$ tal que $B(x, \epsilon) \subset S$. O conjunto dos pontos interiores de $S$ é chamado de interior de $S$ e é denotado por $S^o$.*

## 2.8 O Espaço Métrico $(H(X), h)$

O espaço descrito a seguir, é considerado espaço ideal onde vivem os fractais. São trabalhados espaços métricos mais simples tais como $(R_2, Euclidiana)$ ou ($\hat{C}$, esférica), denotados por $(X, d)$.

**Definição 2.17** *Seja $(X, d)$ um espaço métrico completo. Então $H(X)$ denota o espaço cujos pontos são subconjuntos compactos e não vazios de $X$.*

**Definição 2.18** *Seja $(X, d)$ um espaço métrico completo, $x \in X$ e $B \in H(X)$. Define-se $d(x, B) = Min(d(x, y) : y \in B)$. Então $d(x, B)$ é a distância do ponto $x$ ao conjunto $B$.*

**Definição 2.19** *Seja $(X, d)$ um espaço métrico completo. Seja $A, B \in H(X)$. Define-se $d(A, B) = Max(d(x, B) : x \in A)$. Denota-se a distância do conjunto $A \in H(X)$ ao conjunto $B \in H(X)$ de $d(A, B)$.*

**Definição 2.20** *Seja $(X, d)$ um espaço métrico completo. Então a distância de Hausdorff entre os pontos $A$ e $B$ em $H(X)$ é definida por $h(A, B) = d(A, B) \vee d(B, A)$.*

## 2.9 O Completamento do Espaços dos Fractais

Até o presente momento no desenvolvimento da Matemática, a idéia de fractal é mais usada com um amplo conceito. Fractais não possuem uma definição formal, mas por muitas figuras e conceitos em contextos que se referem ao tema. Aqui, considera-se que um fractal é um subconjunto de $(H(X), h)$.

**Lema 2.1 (Lema da Extensão)** *Seja $(X, d)$ um espaço métrico. Seja $A_n : n = 1, 2, ..., \infty$ uma sequência de Cauchy de pontos em $(H(X), h)$. Seja $(n_j)_{n=1}^{\infty}$ uma sequência infinita de inteiros $0 < n_1 < n_2 < n_3 < ....$ Suponha que tem-se uma sequência de Cauchy $x_{n_j} : j = 1, 2, 3, ...$ em $(X, d)$. Então existe uma sequência de Cauchy $\tilde{x}_n \in A_n : n = 1, 2, ...$ tal que $\tilde{x}_{n_j} = x_{n_j}$ $\forall j = 1, 2, 3, ...$*

**Teorema 2.3 (O completamento do espaço dos Fractais)** *Seja $(X, d)$ um espaço métrico completo. Então $(H(X), h)$ é um espaço métrico completo. Além disso, se $(A_n)_{n=1}^{\infty}$ onde $A_n \in H(X)$ é uma sequência de Cauchy, então $A = \lim_{n\to\infty} A_n \in H(X)$ pode ser caracterizado da seguinte maneira: $A = x \in X : \exists$ sequência de Cauchy $x_n \in A_n$ que converge para $x$.*

**Lema 2.2** *1. $A \neq \emptyset$*

*2. $A$ é fechado e por isso completo desde que $X$ seja completo;*

*3. Para $\epsilon > 0$, existe $N$ tal que para $n \geq N, A \subset A_n + \epsilon$;*

*4. $A$ é totalmente limitado e então por (2) é completo;*

*5. $A = \lim A_n$;*

## 2.10 O Espaço dos Fractais

### 2.10.1 Transformações na reta real

**Definição 2.21** *Seja $(X, d)$ um espaço métrico. Uma transformação em $X$ é um função $f : X \to X$ que atribui exatamente um ponto $f(x) \in X$ para cada ponto $x \in X$. Se $S \subset X$, então $f(S) = f(x) : x \in S$.*

**Definição 2.22** *A função $f$ é chamada injetora, se $f(x) = f(y)$ implica $x = y$ onde $x, y \in X$. A função $f$ é sobrejetora se $f(X) = X$. Neste caso é possível definir a transformação $f^{-1} : X \to X$, chamada inversa de $f$, por $f^{-1}(y) = x$, onde $x \in X$ é o único ponto tal que $y = f(x)$.*

**Definição 2.23** *Seja $f : X \to X$ uma transformação em um espaço métrico. As composições de $f$ são as transformações $f^{on} : X \to X$ definidas por $f^{o0}(x) = x, f^{o1}(x) = f(x), ..., f^{o(n+1)}(x) = f^o f^{on}(x) = f(f^{on}(x))$ para $n = 0, 1, 2, ....$ E se $f$ é invertível, então a decomposição de $f$ é a transformação $f^{o-m}(x) : X \to X$ definida por $f^{o-1}(x) = f^{-1}(x), f^{o-m}(x) = (f^{om})^{-1}$ para $m = 1, 2, 3, ....$*

**Definição 2.24** *A transformação $f : \Re \to \Re$ da forma $f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + ... + a_N x^N$ com coeficientes $a_i, (i = 1, 2, ..., N)$ reais, $a_N \neq 0$ e $N$ um inteiro não negativo é chamada transformação polinomial de grau $N$.*

### 2.10.2 Transformações afins no plano Euclidiano

**Definição 2.25** *A transformação $W : \Re^2 \to \Re^2$ da forma $W(x_1, x_2) = (ax_1 + bx_2 + e, cx_1 + dx_2 + f)$, onde $a, b, c, d, e, f \in \Re$, é chamada transformação afim.*

Freqüentemente pode-se usar as notações equivalentes $W(x) = W\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} = Ax + t$. Aqui, $A$ é uma matriz real $2 \times 2$ e $t$ é o vetor coluna que não se distingue da coordenada $(e, f) \in \Re^2$.

**Definição 2.26** *A transformação $W : \Re^2 \to \Re^2$ é chamada similitude se for uma transformação afim tendo uma das formas especiais:*

$$W\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} r\cos\theta & -r\sin\theta \\ r\sin\theta & r\cos\theta \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}.$$

$$W\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} r\cos\theta & r\sin\theta \\ r\sin\theta & -r\cos\theta \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}.$$

*para alguma translação $(e, f) \in \Re^2$, para algum número real $r \neq 0$ e para algum ângulo $(\theta, 0 \leq \theta \leq 2\pi)$. $\theta$ é chamado ângulo de rotação e $r$ é denominado fator de escala.*

**Definição 2.27** *A transformação afim $R\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ é uma reflexão.*

**Definição 2.28** *Seja $f : X \to X$ uma transformação sobre o espaço métrico. Um ponto $x_f \in X$ tal que $f(x_f) = x_f$ é chamado de ponto fixo da transformação.*

Os pontos fixos das transformações são muito importantes uma vez que eles dizem qual parte do espaço não será movida pela transformação.

## 2.11 O Teorema da Contração

**Definição 2.29** *A transformação $f : X \to X$ no espaço métrico $(X, d)$ é chamada de contratividade se existe uma constante $0 \leq s \leq 1$, tal que $d(f(x), f(y)) \leq sd(x, y), \forall x, y \in X$. E assim, $s$ é chamado fator de contração para $f$.*

**Teorema 2.4 (O teorema da contração)** *Seja $f : X \to X$ uma contração em um espaço métrico completo $(X, d)$. Então $f$ possui exatamente um ponto fixo $x_f \in X$ e além disso, para algum ponto $x \in X$, a seqüencia $f^{\circ n}(x) : n = 0, 1, 2, ...$ converge para $x_f$. Isto é, $\lim_{n\to\infty} f^{\circ n}(x) = x_f$ para cada $x \in X$.*

## 2.12 Contratividade no Espaço dos Fractais

**Lema 2.3** *Seja $w : X \to X$ uma contratividade em um espaço métrico $(X, d)$. Então $w$ é contínua e além disso, ela mapeia $H(x)$ nele mesmo. Aqui, $(H(x), h(d))$ denota o espaço de conjuntos compactos munidos da Métrica de Hausdorff.*

**Lema 2.4** *Seja $w : X \to X$ uma contratividade em um espaço métrico $(X, d)$. Então $W : H(X) \to H(X)$ definida por $w(B) = w(x) : x \in B, \forall B \in H(X)$ é uma contratividade em $(H(X), h(d))$ com fator de contração $s$.*

**Lema 2.5** *Seja $(X, d)$ um espaço métrico. Seja $W_n : n = 1, 2, ..., N$ uma contratividade em $(H(x), h(d))$. Seja o fator de contração $w_n$ denotado por $s_n$ para cada $n$. Defina $W : H(X) \to H(X)$ por $W(B) = w_1(B) \cup w_2(B) \cup ... \cup w_n(B) = \cup_{n=1}^{N} W_N(B)$ para cada $B \in H(X)$. Então $W$ é uma contratividade com fator de contração $s = Max(s_n : n = 1, 2, ..., N)$.*

**Definição 2.30** *Um sistema de funções iteradas (IFS) consiste em um espaço métrico juntamente com um conjunto finito de contratividades* $W_n : X \to X$ *com respectivos fatores de contração* $s_n$*, para* $n = 1, 2, ..., N$*. A notação usada é* $(X, w_n, n = 1, 2, ..., N)$*.*

**Teorema 2.5 (Teorema da unicidade do ponto fixo)** *Seja* $(X, w_n, n = 1, 2, ..., N)$ *um IFS com fator de contração* $s$*. Então a transformação* $W : H(X) \to H(X)$ *definida por* $W(B) = \cup_{n=1}^{N} W_N(B), \forall B \in H(X)$ *é uma contratividade no espaço métrico completo* $(H(X), h(d))$ *com fator de contração* $s$*. Disto,* $h(W(B), W(C)) \leq s.h(B, C), \forall B, C \in H(X)$*. Existe um único ponto fixo,* $A \in H(X)$ *obedecendo* $A = w(A) = \cup_{n=1}^{N} W_N(A)$ *e é dado por* $A = \lim_{n\to\infty} W^{\circ n}(B)$ *para algum* $B \in H(X)$*.*

**Definição 2.31** *O ponto fixo* $A \in H(X)$ *descrito no teorema anterior é chamado de 'atrator' do IFS.*

## 2.13 O Teorema da Colagem

Obter o atrator de um IFS é uma operação relativamente simples, entretanto o problema contrário, ou seja, tendo-se o atrator, encontrar um IFS que o gere é uma tarefa difícil. Sendo assim, o próximo teorema nos diz que para encontrar um IFS cujo atrator se aproxima de um determinado conjunto deve-se encontrar um conjunto de transformações tal que a união ou a colagem das imagens consideradas para um conjunto de transformações estejam próximas a este determinado conjunto.

**Teorema 2.6 (Teorema da colagem)** *Seja* $(X, b)$ *um espaço métrico completo. Seja* $L \in H(X)$ *dado, e seja* $\epsilon \geq 0$ *dado. Escolha um IFS,* $(X, w_n, n = 1, 2, ..., N)$ *com fator de contratividade* $0 \leq s \leq 1$*, tal que* $h(L, \cup_{n=0ou1}^{N} W_N(L)) \leq$

$\epsilon$, *onde* $h(d)$ *é a métrica de Hausdorff. Então* $h(L, A) \leq \epsilon/(1-s)$, *onde* $A$ *é o atrator do IFS. Equivalentemente,* $h(L, A) \leq (1-s)^{-1}.h(L, \cup_{n=0ou1}^{N} W_N(L)), \forall L \in H(X)$.

**Lema 2.6** *Seja* $(X, d)$ *um espaço métrico completo. Seja* $f : X \to X$ *um mapeamento contrativo com fator de contratividade* $0 \leq s \leq 1$, *onde o ponto fixo de* $f$ *é* $x_f$. *Então* $d(x, x_f) \leq (1-s)^{-1}.d(x, f(x)), \forall x \in X$.

## 2.14 Considerações Finais Deste Capítulo

Este capítulo mostrou uma idéia geral do que vem a ser os fractais, os exemplos mais comuns e diversas aplicações. Apresentou também as principais definições matemáticas necessárias ao embasamento teórico da compressão fractal, conceitos tais como espaços métricos, sejam eles abertos, fechados, conexos, desconexos, completos, compactos, tipos de métricas e suas principais propriedades. Esses temas foram essencias para a construção do espaço métrico dos fractais.

O próximo capítulo mostra uma idéia geral de diversos tipos de compressão de imagens e a evolução de algumas técnicas de compressão bem como suas vantagens e/ou desvantagens. Mostra dois tipos de particionamento para compressão fractal, analisa e compara os mesmos. São mostrados também a implementação do método utilizando particionamento quadtree e resultados de testes realizados com imagens quadradas padrão.

# Capítulo 3

# Compressão Fractal de Imagens

## 3.1 Noções de Compressão de Imagens

Este capítulo mostra idéia geral de diversos tipos de compressão de imagens e a evolução de algumas técnicas de compressão bem como suas vantagens e/ou desvantagens. Mostra dois tipos de particionamento para compressão fractal, analisa e compara os mesmos. São mostrados também a implementação do método utilizando particionamento quadtree e resultados de testes realizados com imagens quadradas padrão.

A compressão de imagens trata o problema de reduzir a quantidade de dados necessária para representar uma imagem digital fazendo a remoção de dados redundantes. Uma consequência desse processamento é que a quantidade de memória necessária para representar e armazenar uma imagem é diminuída. Matematicamente, isto corresponde a transformar uma matriz de pixels de duas dimensões em um conjunto de dados estatisticamente descorrelacionado[9]. Comprimir uma imagem é, na prática, reduzir a redundância dos dados dessa imagem, de forma a armazenar ou transmitir esses mesmos dados de forma eficiente. Essas redundâncias podem ser de

três tipos[9]:

- A redundância de codificação ocorre quando os níveis de cinza de uma imagem são codificados usando mais símbolos do que o estritamente necessário;

- A redundância interpixel ocorre porque boa parte da contribuição visual de um pixel é redundante e pode ser deduzida de valores de pixels vizinhos;

- A redundância psicovisual está associada a informações quantificáveis ou reais e sua eliminação só é possível quando a informação não é essencial ao processo visual humano.

A compressão de uma imagem digital tem por objetivo reconstruir uma imagem que esteja o mais próximo possível da imagem original. Sendo assim, ao reconstruir a imagem comprimida deve se avaliar essa tal "aproximação", para isso são usados alguns critérios de fidelidade que podem ser de dois tipos:

**Critério de fidelidade objetivo:** quando o nível de perda da informação for expresso através de uma função que mede a diferença entre a imagem original e a imagem comprimida.

**Critério de fidelidade subjetivo:** é expresso por meio de avaliações subjetivas de um avaliador, observando a qualidade da imagem a olho nu.

### 3.1.1 Tipos de compressão

Os processos de compressão de imagens dividem-se em dois tipos: *compressão com perda* e *compressão sem perda*.

**Compressão com perda:** é utilizada nos casos em que a redução da imagem é mais importante que a qualidade, sem no entanto menosprezar esta. Nesse caso, geralmente existe uma alta taxa de compressão. É o caso das máquinas fotográficas digitais em geral, que gravam mais informação do que o olho humano detecta, que por sua vez podem ser eliminadas por técnicas de compressão de imagens. Ocorrem também em transmissões de TV, videoconferências e internet.

**Compressão sem perda:** é utilizada quando se deseja reconstruir a imagem exatamente idêntica a imagem original. Esse método prioriza a qualidade da imagem por isso perde um pouco em taxa de compressão. É usada em alguns tipos de imagens médicas e códigos de programas por exemplo.

## 3.2 Técnicas de Compressão de Imagens

Os fundamentos teóricos da compressão de dados surgiram na década de 40 com a publicação da *Teoria da Informação* de C.E. Shannon. Pelas aplicações práticas desses trabalhos teóricos de Shannon e outros, foi possível um grande crescimento da área até os dias atuais. Hoje, reconhecida como *tecnologia de capacitação*, a compressão de imagens é usada por exemplo na manipulação das resoluções espaciais, sensoriamento remoto, evolução das padronizações de transmissão de TV, videoconferências, imageamento médico, transmissão de FAX, operações militares e espaciais, entre outros [9]. Abaixo, são relacionadas algumas técnicas conhecidas, sejam elas compressões propriamente ditas, como é o caso do JPEG e do PNG, ou mesmo, codificações que cumprem papel fundamental na compressão , como é o caso da codificação de Huffman ou da codificação wavelet.

**Codificação de Huffman:** é um método que usa as probabilidades de ocorrência dos símbolos no conjunto de dados a ser comprimido para determinar códigos de tamanho variáveis para cada símbolo, atribuindo para aqueles de maior probabilidade de ocorrência menor comprimento. Ele foi desenvolvido em 1952 por *David A. Huffman*.

Uma árvore binária completa, chamada de árvore de Huffman é construída recursivamente a partir da junção dos dois símbolos de menor probabilidade, que são então somados em símbolos auxiliares e estes símbolos são recolocados no conjunto de símbolos. O processo termina quando todos os símbolos foram unidos em símbolos auxiliares, formando uma árvore binária. A árvore é então percorrida, atribuindo-se valores binários de 1 ou 0 para cada aresta, e os códigos são gerados a partir desse percurso.

**Codificação Aritmética:** a partir de um modelo estatístico, constrói-se uma tabela onde são listadas as probabilidades de o próximo símbolo lido ser cada um dos possíveis símbolos. Em geral esta probabilidade é simplesmente a contagem de todas as ocorrências do símbolo no arquivo dividida pelo tamanho do arquivo.

Representa-se a probabilidade de ocorrência de cada caractere de acordo com intervalos. Parte-se do intervalo [0,1) e nele identifica-se o subintervalo ao qual corresponde o primeiro símbolo lido do arquivo. Para cada símbolo subseqüente, subdivide-se o intervalo atual em sub-intervalos proporcionais às probabilidades da tabela de intervalos, e encontra-se novamente o intervalo que corresponde ao próximo símbolo. Ao final do processo, tem-se um intervalo que corresponde a probabilidade de ocorrência de todos os símbolos na ordem correta.

**Codificação JPEG**: este formato de arquivo, foi desenvolvido inicialmente por *Eric Hamilton* da C-Cube Microsystems que decidiu disponibilizá-lo em domínio público sob o nome de JPEG File Interchange Format (JFIF). Mas por razões alheias ao autor, generalizou-se chamar a este formato JPEG. Trata-se de um formato de compressão, geralmente com perda de dados, aplicado em imagens fotográficas que faz uso da transformada discreta do cosseno (DCT). A perda de dados é proporcional ao fator de compressão desejado. O arquivo que usa esse método de compressão é chamado normalmente por JPEG; as extensões de arquivos para esse formato são .jpeg , .jfif , .jpe e .jpg, este último, é o mais comum.

O processo de compactação JPEG é composto das seguintes fases: (1) a imagem é divida em blocos de $8 \times 8$ pixels e em cada um destes blocos é calculada a DCT. (2) os coeficientes gerados pela DCT são quantizados e alguns desses coeficientes são eliminados. O processo de quantização é que irá definir o grau de compactação da imagem. (3) na última etapa a codificação de Huffman é aplicada aos coeficientes quantizados. O JPEG é uma técnica de compressão com perdas, mas como a taxa de compressão depende de vários fatores, inclusive da imagem em questão, uma taxa de 10:1 ou 20:1 não causará uma perda visível siginificativa na imagem original.

**Codificação JPEG2000**: é um tipo de codificação um pouco mais recente, resultante de um trabalho colaborativo entre a União Internacional das Telecomunicações (UIT) e a Organização Internacional de Normalização (ISO). Exige um pouco mais de tempo de processamento do que o JPEG, no entanto, as taxas de compressão geralmente são maiores e como vantagem mantêm uma melhor qualidade da imagem.

**Codificação Wavelet**: é uma função capaz de decompor e descrever outras funções no domínio da freqüência, tornando possível a análise destas funções em diferentes escalas de freqüência e de tempo. A decomposição de uma função com o uso de wavelets é conhecida como "transformada wavelet" e tem suas variantes contínua e discreta. Graças a capacidade de decompor as funções tanto no domínio da freqüência quanto no domínio do tempo, as funções wavelet são ferramentas poderosas para a análise de sinais e compressão de dados. Apresenta algoritmos de implementação rápida, boa capacidade de concentrar a energia do sinal em poucos coeficientes, além de não introduzir os efeitos de blocagem.

**Codificação GIF:** abreviação de Graphics Interchange Format, é considerado um algoritmo de compressão sem perda, criado pela CompuServe em 1987. Geralmente é usado em animações e figuras geométricas.

**Codificação PNG:** criada em 1996 e motivado pela possível cobrança de royalties por parte da Unisys detentora dos direitos do formato GIF. O PNG tinha como objetivo ser um GIF melhorado e de fato é. Permite comprimir as imagens sem perda de qualidade, por isso é um formato válido para imagens que precisam manter boa qualidade. Pode ser usado na maioria dos programas de edição de imagens.

## 3.3 Compressão Fractal de Imagens

Em 1988, *Michael Barnsley* e *Alan Sloan* propuseram a compressão fractal de imagens como uma técnica de armazenamento de imagens usando pouco

espaço. Isto ocorre porque a técnica de compressão fractal se baseia no fato de que se armazenam coeficientes de funções afins, o que ocupa bem menos espaço. O processo de descompressão é realizado por uma série de iterações de funções afins, o que o torna mais rápido. A desvantagem está na busca dos coeficientes que melhor se ajustam a cada parte da imagem, mas nesse âmbito já foram e estão sendo desenvolvidos vários métodos para melhorar e até solucionar tais problemas.

A técnica mais simples de compressão fractal, começa por dividir uma imagem em inúmeras partições denominadas de blocos molde ou *range-blocks*. Quanto maiores esses blocos, mais reduzida será a qualidade da respectiva imagem. Tal algoritmo foi implementado pela primeira vez por *Barnsley* na década de 80 [10]. Para cada um desses blocos molde, é feita uma procura na imagem, aplicando um conjunto de 8 transformações afins e fazendo ajuste de brilho e contraste, com o objetivo de encontrar os blocos domínio ou *domain-blocks* similares a esse bloco. Cada vez que é encontrada uma igualdade, guarda-se a localização do bloco de dominio e a transformação associada ao bloco onde ocorreu a igualdade. Da execução iterativa desse processo resulta um conjunto de dados cujo armazenamento é menor do que o necessário para a representação da imagem original.

Com esse método, apesar de já se terem conseguido compressões na ordem dos 100:1 [7], o grau de compressão médio que se consegue obter situa-se na ordem dos 50:1, visto que, como já foi referido, esse é um método cujo resultado depende muito da imagem que ele for aplicado.

## 3.4 Princípios da Compressão Fractal

Fischer [7] fez uma analogia do sistema de compressão fractal agindo com uma máquina que reduz a imagem à metade de seu tamanho, a triplica e translada. Se esse processo se repetir por várias vezes, ou seja, após várias iterações, observa-se que a seqüência de cópias geradas converge para a mesma imagem final, independente da imagem inicial, isto é, a imagem converge para um *atrator*, que seria nada mais do que a imagem final gerada após o término das iterações. A Figura 3.1 mostra esse processo.

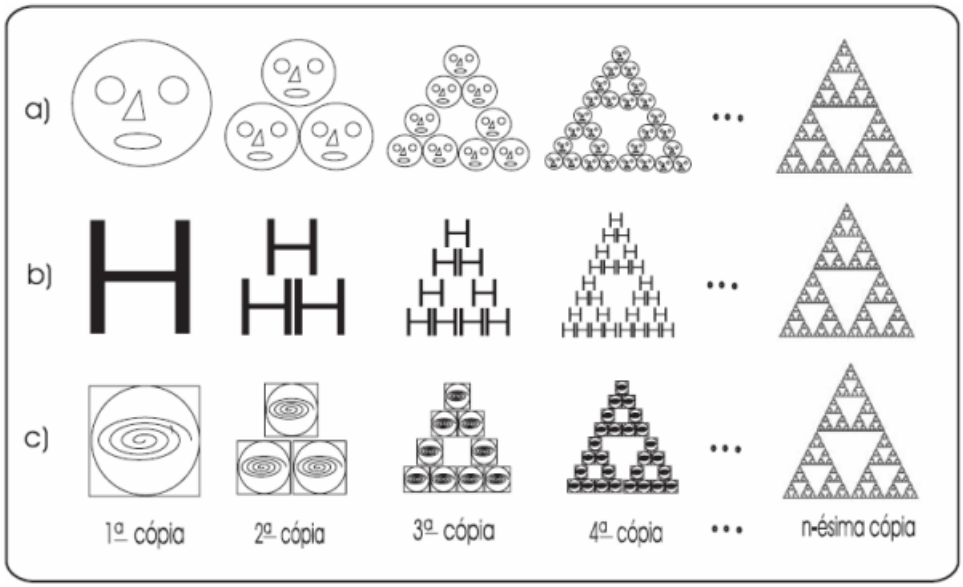


Figura 3.1: Figuras geradas pelo sistema de compressão fractal

A idéia básica do sistema de compressão fractal parte do princípio de que uma imagem fractal é rica em detalhes tanto no todo quanto em cada parte de si e por isso é chamada de auto-similar [1]. Pode-se perceber claramente na Figura 3.1 que a imagem final independe da imagem inicial. Isso só varia se forem mudadas as transformações afins que foram aplicadas, ou seja, transformações diferentes geram imagens diferentes.

Quando são realizadas repetidas iterações, independente da imagem original, as cópias de saída vão convergir sempre para a mesma imagem final que é chamada de *atrator*. É importante lembrar que tais transformações têm a propriedade de serem contrativas, ou seja, a distância entre quaisquer dois pontos de uma imagem de saída é sempre menor do que a distância entre quaisquer dois pontos de uma imagem de entrada.

Na prática, as transformações responsáveis por gerarem tais atratores são obtidas pela equação 3.1 [1]:

$$W_i \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \end{bmatrix}. \tag{3.1}$$

No caso em questão, pode-se tomar como exemplo as transformações que geram uma *Fractal Tree*(árvore de fractais), ilustrada na Figura 3.2.

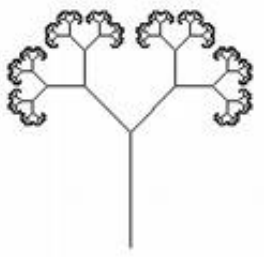

Figura 3.2: Fractal tree

Para formar a árvore de fractais são utilizadas as transformações $w_1, w_2, w_3, w_4$ exibidas a seguir e os 24 coeficientes obtidos com essas 4 equações formam o *código fractal* da *fractal tree* [1].

$$w_1 \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0.5 \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{3.2}$$

$$w_2 \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0.42 & -0.42 \\ 0.42 & 0.42 \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0.2 \end{bmatrix}. \tag{3.3}$$

$$w_3 \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0.42 & 0.42 \\ -0.42 & 0.42 \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0.2 \end{bmatrix}. \tag{3.4}$$

$$w_4 \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0.2 \end{bmatrix}. \tag{3.5}$$

Para a geração da *fractal tree* aplica-se as transformações $w_1, w_2, w_3, w_4$ sobre uma imagem inicial qualquer $I$, então é realizada uma composição com os resultados $w_1(I) \cup w_2(I) \cup w_3(I) \cup w_4(I)$, obtendo assim uma nova imagem que vai ser utilizada como entrada na próxima iteração. Este processo é representado pela Equação 3.6.

$$W(I) = \cup_{i=1}^{n} W_i(I) \tag{3.6}$$

onde:

$n$ é o número de transformações usadas.

No caso da *Fractal Tree*, $n = 4$.

Assim, ao invés de armazenar a imagem, armazena-se somente estes coeficientes de $w_i$, o que faz com que a compressão fractal seja eficiente e ocupe

bem menos espaço. A imagem da *fractal tree* de $256 \times 256$ *pixels* por exemplo, com 8 *bits por pixel*, gastaria ($256 * 256 * 8 = 524288$ *bits*). Com a codificação fractal ela pode ser armazenada com um número bastante inferior de bits.

Uma vez armazenados esses coeficientes para reconstruir a imagem basta aplicar iterativamente as transformações com seus respectivos coeficientes em uma imagem qualquer.

## 3.5 Métricas e Medidas de Fidelidade

Com o intuito de verificar se as transformações uutilizadas em questão são mesmo contrativas e usar o sistema de busca necessário para a codificação fractal, é preciso definir e utilizar uma função distância entre duas imagens, ou seja, uma métrica, cuja definição e propriedades foram apresentadas no Capítulo 2. Dentre as métricas mais usadas foi escolhida uma que simplifica o processamento computacional[11]. Essa métrica é a *distância euclidiana* também conhecida com *distância RMS (Raíz média quadrática)*, obtida pela equãção 3.7.

$$d_{rms}(x, y) = \sqrt{\frac{1}{MN} \sum_{x=1}^{M} \sum_{y=1}^{N} (x - y)^2} \tag{3.7}$$

Quando uma técnica de compressão é aplicada a uma imagem, é necessário ver até que ponto a imagem obtida se aproxima da imagem original. Podem ser utilizados critérios de fidelidade objetivos, tais como[4]:

$$e_t = \sum_{x=1}^{M} \sum_{y=1}^{N} [f'(x, y) - f(x, y)] \tag{3.8}$$

$$e_{rms} = \sqrt{\frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}[f'(x,y) - f(x,y)]^2} \tag{3.9}$$

$$SNR_{rms} = \frac{\sum_{x=1}^{M}\sum_{y=1}^{N} f'(x,y)^2}{\sum_{x=1}^{M}\sum_{y=1}^{N}[f'(x,y) - f(x,y)]^2} \tag{3.10}$$

onde:

$M$ é a largura da imagem é $M$;

$N$ é a altura da imagem;

$f'$ representa a imagem descomprimida;

$f$ é a imagem original.

## 3.6 Codificação e Decodificação utilizando Sistema de Funções Iteradas Particionado

Como já citado no Capítulo 2 desta dissertação, imagens fractais têm a característica de serem auto-similares, ou seja, são similares a uma parte de si mesma. No entanto, muitas vezes, as imagens reais não possuem, ou possuem pouca auto-similaridade. Sendo assim, uma imagem desse tipo pode ser codificada por um conjunto de transformações que retorna apenas uma aproximação da imagem original e não uma cópia idêntica como seria o ideal.

Isso é basicamente o que a compressão fractal faz, ela elimina a redundância da auto-similaridade de uma imagem. No caso da codificação utilizando os (Partitioned Iterated Function Systems) PIFS, a imagem é formada pela codificação de partes de si mesma, por isso, na maioria das vezes o sistema não obtém uma imagem idêntica à original.

Aqui, uma parte da imagem original que é chamada de *(domain-block)* ($D_i$), é copiada, com ajuste de brilho e contraste e estabelecimento de localização, escala e rotação, para uma outra parte da imagem, chamada de *(range-block)* ($R_i$). A transformação que faz essa cópia é indicada por $w_i$, ou seja, $w_i : D_i \rightarrow R_i$. A aplicação dessa transformação iterativamente em partes de uma imagem semente $f_0$ é que reproduz a imagem comprimida.

### 3.6.1 Sistema de Funções Iteradas Particionado - (PIFS)

**Definição 3.1** *Um PIFS é um sistema de funções de iteração que considera certas regiões de uma imagem, as particiona e mapeia em regiões menores da mesma imagem.*

Agora, a Equação 3.1 pode ser definida conforme a Equação 3.11:

$$w_i \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \\ o_i \end{bmatrix}. \tag{3.11}$$

onde:

$s_i$ - contraste

$o_i$ - brilho

$a_i, b_i, c_i, d_i, e_i, f_i$ - coeficientes das transformações geométricas

Para codificar uma imagem $f$ usando $PIFS$ é preciso encontrar o mapeamento $w_i, w_2, ..., w_N$, onde $W = \cup_{i=1}^{n} w_i$, tal que $W(f) = f$, ou seja, a imagem $f$ que seja o ponto fixo(ou atrator) do mapeamento $W$. Para decodificar a imagem basta escolher uma imagem semente $f_o$ qualquer e aplicar a Equação 3.12.

$$W(f_0), W(W(f_0)), W(W(W(f_0))), ... \tag{3.12}$$

onde $W$ é uma transformação linear. O teorema da Unicidade do Ponto Fixo apresentado no Capítulo 2 desta dissertação, garante que, independentemente da imagem $f_0$ escolhida, a imagem gerada será a mesma.

Estudos realizados até o presente momento mostram que, do ponto de vista computacional, a compressão fractal é um processo assimétrico, ou seja, a codificação é mais trabalhosa e lenta, exigindo maior esforço computacional que a decodificação.

### 3.6.2 Codificação PIFS

De forma mais simples particiona-se uma imagem a ser codificada em blocos quadrados de tamanho fixo, por exemplo, $4 \times 4$ *pixels* que são os chamados *range-blocks* ou *blocos molde* e a seguir particiona-se a mesma imagem em blocos também quadrados de tamanho fixo $8 \times 8$ *pixels* que são os chamados *domain-blocks* ou *blocos dominio*. Os *range-blocks* não se sobrepõem e devem ocupar todos os pixels da imagem, os *domain-blocks* podem sobrepor-se e não precisam necessariamente ocupar todos os pixels da imagem. A cada *domain-block* são aplicadas filtragem média e subamostragem com o objetivo de reduzí-lo ao mesmo tamanho do *range-block*, logo a seguir aplica-se as transformações $w_i$. Os resultados são comparados utilizando-se uma métrica apropriada. Uma vez encontrada a melhor aproximação, serão armazenados

somente os coeficientes da transformação equivalente a cada *range-block* e a posição do *domain-block* escolhido.

Nesse caso encontra-se uma imagem $f'$, usando uma métrica apropriada $d$, onde $d(f, f')$ seja pequena o suficiente, ou seja, procura-se $W$ tal que o ponto fixo ou atrator $f'$ seja o mais próximo possível da imagem original $f$.

### 3.6.3 Decodificação PIFS

Na decodificação PIFS, geralmente, uma máscara seleciona uma parte ($D_i$) da imagem semente, na qual aplica-se filtragem média, subamostragem e uma transformação $w_i$ (já armazenada e transmitida). Copia-se a imagem transformada para uma região específica ($R_i$) da imagem de saída. Esse processo é repetido para todos os *domain-blocks* da imagem semente. O conjunto final de todos os *range-blocks* é o *bloco imagem* e formará a imagem já decodificada. Assim, considerando-se uma imagem $f$ em escala de cinza, as iterações do processo de decodificação utilizando uma partição com $N$ *range-blocks*, são realizadas utilizando-se o mapeamento da Equação 3.13.

$$W(f) = w_1(D_1) \cup w_2(D_2) \cup ... \cup w_N(d_N) \tag{3.13}$$

onde $w_i$ é aplicada somente sobre a região $D_i$, uma vez que no processo de codificação obtém-se para cada $R_i$ um $D_i$ específico.

## 3.7 Compressão Fractal Utilizando o Método da Força Bruta

### 3.7.1 Codificação

O processo de compressão fractal conhecido como *método da força bruta*, exige esforço computacional intenso. Esse método foi apresentado e implementado por *Barnsley* e *Hurd* [10] na década de 90.

No método da força bruta, particiona-se a imagem em *range-blocks* quadrados de tamanho fixo $4 \times 4$ *pixels* e em *domain-blocks* também quadrados de tamanho fixo $8 \times 8$ *pixels*. Os *domain-blocks* são reduzidos a 1/4 de seu tamanho ao aplicar a filtragem média e a subamostragem. Utilizando-se então a métrica $RMS$, para cada *range-block* $R_i$, todos os blocos $D_i$ são percorridos ajustando-se o brilho ($o_i$), o contraste ($s_i$), e aplicando transformações geométricas , de modo a encontrar a menor distância entre o bloco $R_i$ e seu respectivo bloco $D_i$.

Na prática, isto significa que, para $R_i$ e $D_i$ contendo cada um $n$ *pixels* com intensidades $(d_i, d_2, ..., d_n)$ de $D_i$ e $(r_1, r_2, ..., r_n)$ de $R_i$, deve-se procurar $s_i$ e $o_i$ que minimize e Equação 3.14:

$$R = \sum_{i=1}^{n} (s.d_i + o - r_i)^2 \tag{3.14}$$

O valor mínimo da Equação 3.14 ocorre quando as derivadas parciais em relação a $s_i$ e $o_i$ são iguais a *zero*. Isto ocorre quando:

$$s_i = \frac{[n. \sum_{i=1}^{n} d_i r_i - \sum_{i=1}^{n} d_i \sum_{i=n}^{n} r_i]}{\left[n. \sum_{i=1}^{n} {d_i}^2 - (\sum_{i=1}^{n} d_i)^2\right]} \tag{3.15}$$

$$o_i = \frac{1}{n}. \left[\sum_{i=1}^{n} r_i - \sum_{i=1}^{n} d_i\right] \tag{3.16}$$

Neste caso obtém-se a equação 3.17.

$$R = \frac{1}{n}.\left[\sum_{i=1}^{n} r_i^2 + s.\left(\sum_{i=1}^{n} d_i^2 - 2.\sum_{i=1}^{n} d_i r_i + 2.o.\sum_{i=1}^{n} d_i\right) + o.\left(n.o - 2.\sum_{i=1}^{n} r_i\right)\right] \tag{3.17}$$

Assim, a $d_{rms}$ é obtida calculando-se $\sqrt{R}$. Na prática, neste ponto é computado $d_{rms}(f \cap (R_i \times I), w_i(f))$, onde $f$ é a imagem a ser processada, $I = [0, 1] \in \Re$ e $w_i(f)$ é a transformação a ser aplicada sobre $f$.

Os passos do processo de compressão fractal utilizando-se o método da força bruta são:

**Passo 1:** Divide-se a imagem a ser comprimida de tamanho $N \times N$ pixels em $(N/n)^2$ blocos quadrados de tamanho $n \times n$ que são os chamados *range-blocks*. Esses *range-blocks* não se sobrepoem e ocupam todos os pixels da imagem.

**Passo 2:** Divide-se a imagem a ser comprimida em blocos quadrados de tamanho $2n \times 2n$ pixels que são os *domain-blocks*. Esses *domain-blocks* devem sobrepor-se e não precisam ocupar todos os pixels da imagem. Por exemplo, utilizando-se sobreposição de $2n - 1$ *pixels* tanto na horizontal quanto na vertical, ocupa-se todos os pixels da imagem obtendo-se o maior número possível de *domain-blocks*, o que traz uma fidelidade maior para a imagem, em contrapartida, uma taxa de compressão menor.

**Passo 3:** Reduz-se os *domain-blocks* a 1/4 de seu tamanho, ou seja, $(2n \times 2n)/4$, o que permite a comparação com os *range-blocks*, já que agora

possuem a mesma dimensão. Tal redução é realizada pela filtragem média e subamostragem. Por exemplo, no caso dos *domain-blocks* $D_i$, $8 \times 8$ pixels, toma-se bloquinhos $2 \times 2$ pixels e calcula-se a média entre os pixels, assim, a média desses 4 *pixels* será representada por apenas 1 *pixel* que fará parte do $D_i$ reduzido.

**Passo 4:** Busca-se o $D_i$ reduzido que melhor se aproxima do $R_i$. Isso é realizado utilizando-se uma métrica apropriada, ajuste de brilho $o_i$, contraste $s_i$ (obtidos pelo ajuste das médias das intensidades dos pixels de $D_i$ e dos pixels de $R_i$) e aplicação das 8 simetrias utilizadas. Na prática, são transformações geométricas que são identificadas por 3 *bits*. São elas:

$(0, 0, 0)$ - identidade

$(0, 0, 1)$ - reflexão horizontal

$(0, 1, 0)$ - reflexão vertical

$(1, 0, 0)$ - reflexão em relação à diagonal principal

$(1, 1, 0)$ - rotação de $90^o$

$(0, 1, 1)$ - rotação de $180^o$

$(1, 0, 1)$ - rotação de $270^o$

$(1, 1, 1)$ - reflexão horizontal composta com reflexão vertical e reflexão sobre a diagonal.

Para $s_i$ e $o_i$ geralmente utiliza-se respectivamente 7 e 5 *bits*.

Seleciona-se e armazena-se a posição $(D_x, D_y)$ (pixels do topo mais à esquerda), a simetria $m_i$, onde $i \in (0, 1, ..., 7)$, $o_i$ e $s_i$ do bloco que possuir a menor distância, ou seja, que possuir o menor valor de $\sqrt{R}$.

**Passo 5:** Armazena-se para cada bloco $R_i$, a tupla $(D_x, D_y, m_i, o_i, s_i)$. O conjunto de todas as tuplas é o código fractal.

### 3.7.2 Decodificação

No processo de descompressão primeiro a imagem é dividida em *domain-blocks*, então deve-se aplicar uma filtragem mediana para que tenham o mesmo tamanho dos *range-blocks*, como no processo de compressão. Para cada *domain-block*, aplica-se $w_i$ iteradas vezes (cerca de 10), e copia-se a imagem transformada para o *range-block* de mesmo endereçamento. Repete-se o mesmo processo para cada *domain-block*. A união de todos os *range-blocks* forma a imagem já decodificada.

## 3.8 Compressão Fractal Acelerada

O método descrito anteriormente e implementado por *Barnsley* foi muito utilizado e forneceu bons resultados. Apesar disso, o esforço computacional exigido é intenso o que causa grande gasto de tempo. Sendo assim, no presente trabalho, optou-se por implementar um método que acelera o tempo de processamento ao mesmo tempo que equilibra taxa de compressão com fidelidade. Isto porque trás uma abordagem diferente no que diz respeito ao particionamento da imagem, possibilitando codificar regiões cheias de pequenos detalhes com blocos menores (maior fidelidade) e regiões uniformes com blocos maiores (maior taxa de compressão), utilizando-se também, um tipo de classificação para os blocos que agiliza drasticamente o processo de busca.

### 3.8.1 Particionamento Quadtree

No particionamento *quadtree* representa-se a imagem como uma árvore onde cada parte quadrada da imagem (nó), possui quatro quadrantes (são as ramificações). A raiz da árvore é a imagem original. No método proposto por

*Fischer* [7] primeiro classifica-se os *domain-blocks* para depois fazer a busca. A Figura 3.3 mostra o modelo de partição quadtree.

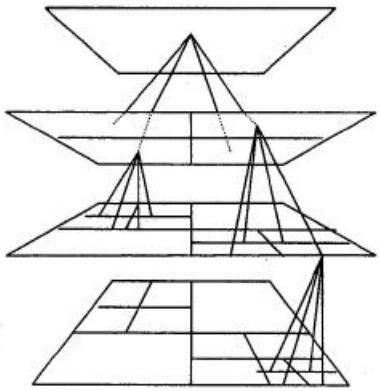

Figura 3.3: Modelo de partição quadtree

### 3.8.2 Processamento

Determina-se os níveis máximo e mínimo da *quadtree*. No primeiro nível divide-se a imagem original em quatro *range-blocks*. Cada *range-block* será comparado com os *domain-blocks* que tenham quatro vezes o tamanho do *range-block*. Cada *domain-block* é subamostrado com filtragem média ficando assim do mesmo tamanho dos *range-block*. Para a classificação dos *domain-block*, é utilizado o "acelerador de Fischer". Primeiro são classificados os *domain-blocks* e durante a codificação os *range-blocks* são também classificados.

*Fischer*[7] utilizou a classificação por *superclasses* e *subclasses*. O bloco quadrado é divido em quatro quadrantes: esquerdo alto, direito alto, esquerdo baixo e direito baixo, numerados sequencialmente. Em cada quadrante calcula-se a média e a variância utilizando-se a intensidade dos pixels da seguinte forma: sejam $(r_1^i, ..., r_n^i)$, $i = 1, 2, 3, 4$ os $n$ pixels do quadrante $i$.

$$M_i = \frac{1}{n}\sum_{j=1}^{n} r_j^i \tag{3.18}$$

$$V_i = \frac{1}{n}\left(\sum_{j=1}^{n} (r_j^i)^2 - M_i^2\right) \tag{3.19}$$

Assim, pelas de rotações e inversões, ordena-se as $M_i$, obtendo-se três superclasses de média:

1. Superclasse 1: $M_1 \geq M_2 \geq M_3 \geq M_4$
2. Superclasse 2: $M_1 \geq M_2 \geq M_4 \geq M_3$
3. Superclasse 3: $M_1 \geq M_4 \geq M_2 \geq M_3$

A Figura 3.4 mostra as superclasses 1, 2 e 3.

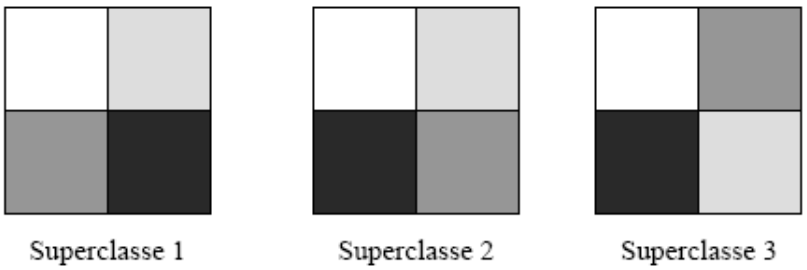


Figura 3.4: Superclasses 1, 2 e 3

Para cada superclasse aplica-se oito transformações geométricas já descritas no item 3.5, obtendo-se assim 24 subclasses de variância que são as representações das 24 possíveis ordenações dos valores de $V_i$, totalizando 72 classes. Tendo classificado os *domain-blocks*, classifica-se, recursivamente, segundo a $M_i$ e/ou $V_i$ também os *range-blocks* e começa-se a busca, considerando-se cada *range-block* e procurando-se no *domain-block* de mesma classe [11].

Considera-se como limiar um $e_{RMS}$ mínimo como utilizado no método descrito na Seção 3.5 deste capítulo. Se tal limiar é atingido pára-se a busca e armazena-se a tupla $(D_x, D_y, m_i, s_i, o_i, n)$, onde $n$ indica o nível da *quadtree*. Caso contrário, particiona-se a imagem adotando-se uma dimensão do *range-block* que seja metade das dimensões do passo imediatamente anterior, e recomeça-se a busca. O processo se repete para cada *range-block* até que se encontre o limiar desejado ou até atingir o nível máximo de particionamento *quadtree*.

Quando a finalidade é decodificar deve-se levar em conta que para cada *range-block*, deve-se transmitir a transformação geométrica utilizada para mapear os pixels do *domain-block* sobre o *range-block*.

Fischer[7] utilizou 1 *bit* para representar cada nó da quadtree(0 ou 1 para indicar se ocorreu quebra ou não), 5 $bits$ para $s_i$ e 7 $bits$ para $o_i$ assim como no *método da força bruta*.

As Figuras 3.5 e 3.6 fornecem uma idéia básica de como se desenvolve o algoritmo de compressão fractal utilizando-se o particionamento quadtree implementado neste trabalho e na sequência é mostrada uma breve explicação de seu funcionamento.

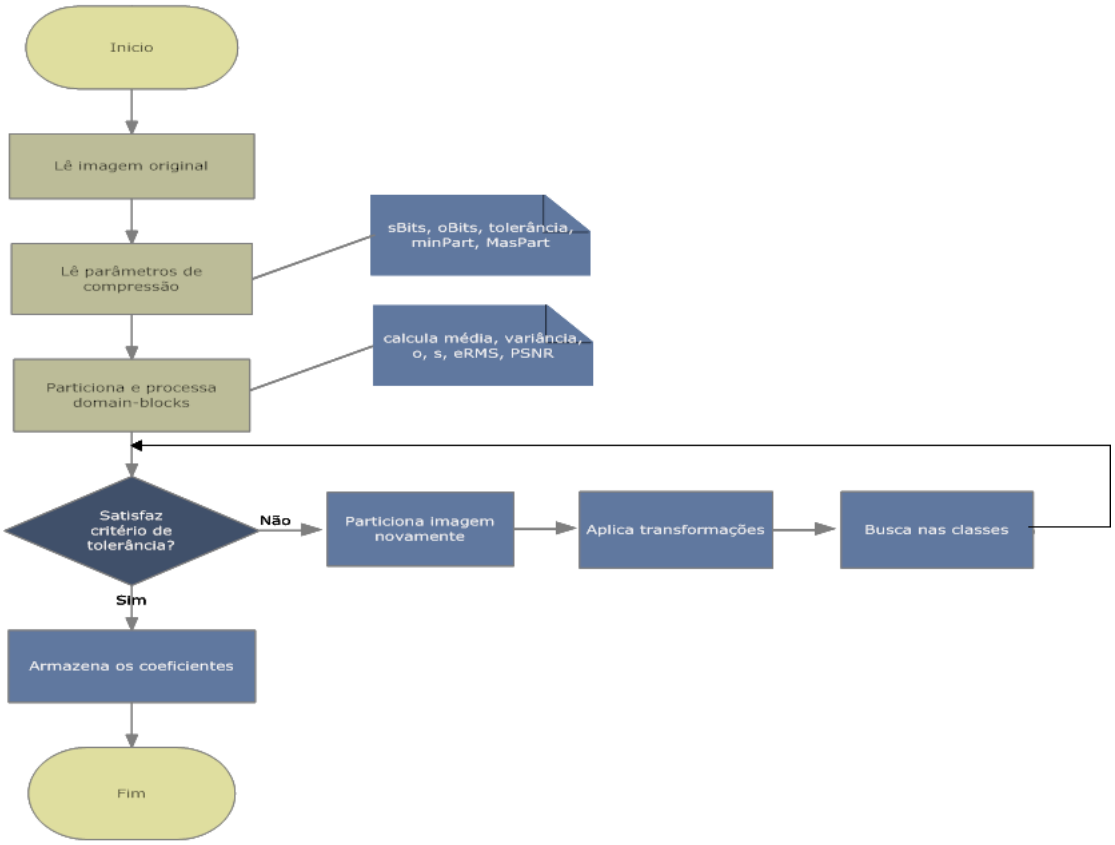


Figura 3.5: Fluxograma do codificador fractal usando particionamento quadtree

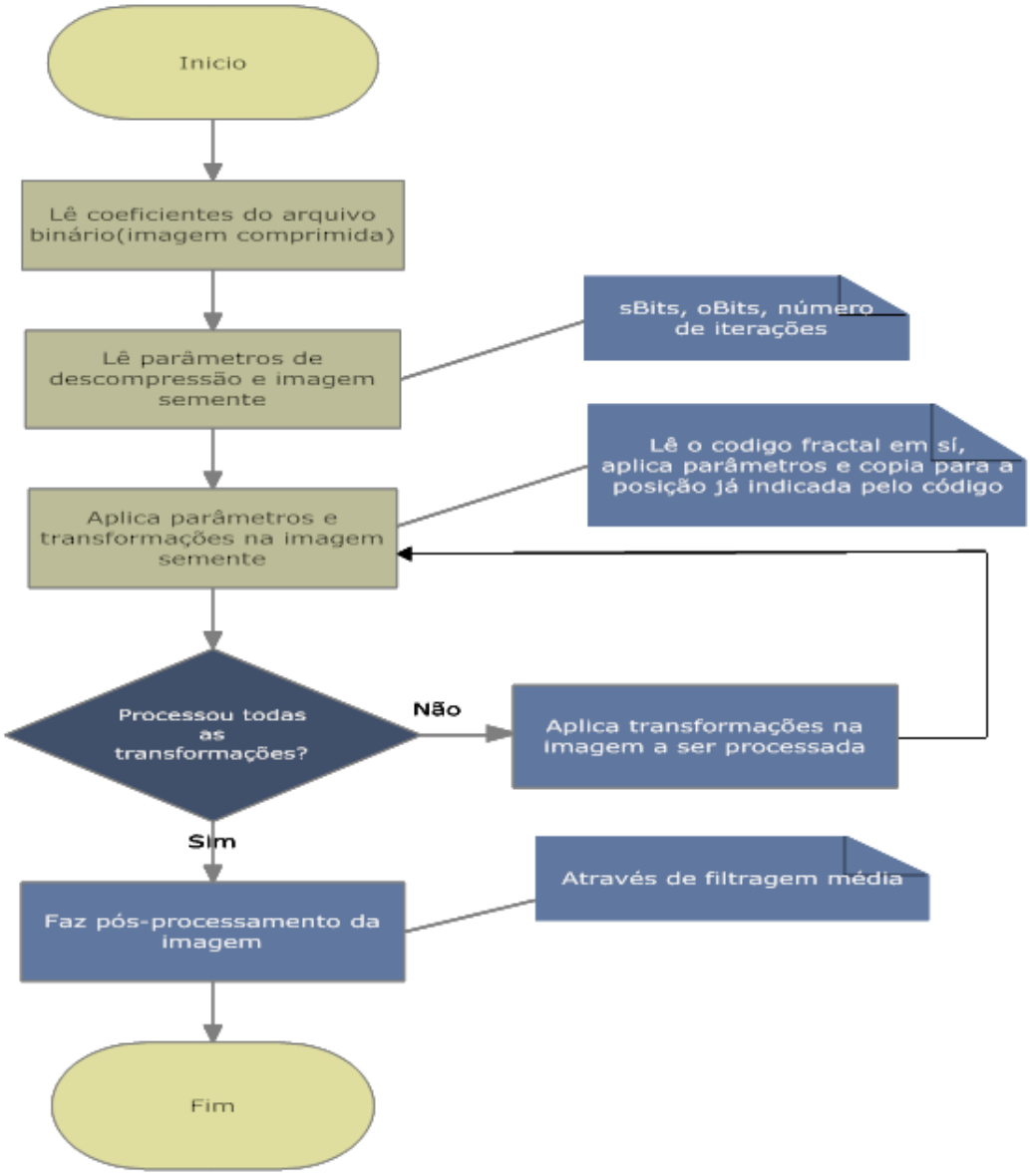


Figura 3.6: Fluxograma do decodificador fractal usando particionamento quadtree

Os passos do algoritmo (código) para fazer a codificação das imagens são:

**1°passo:** particionamento da imagem inteira em quatro quadrantes (*domain-blocks*), é o inicio da montagem do particionamento recursivo e cálculo das médias e variâncias.

**2° passo:** cálculo do tamanho do maior e do menor blocos da imagem através da informação dos níveis mínimo e máximo de particionamento quadtree.

**3° passo:** pelo tamanho do maior e do menor blocos da imagem, calcula-se a quantidade de *domain-blocks*, são realizadas as classificações e a montagem da 'árvore' quadtree.

**4° passo:** criação do arquivo de saída (parâmetros necessários ao código).

**5° passo:** particiona-se recursivamente a imagem em *range-blocks*, calculando-se a média e a variância, faz-se a classificação e inicia-se a busca.

**6° passo:** começando a busca, para cada transformação geométrica (simetria) aplicada, faz-se o ajuste de brilho, contraste e cálculo do erro mínimo quadrático.

**7° passo:** compara-se os erros e armazena-se o código com menor erro ou aquele que já atingiu a tolerância esperada, quando isso não ocorre, passa-se para o próximo nível quadtree. O código armazenado contém as informações da posição do *domain-blocks* da melhor escolha, nível quadtree, brilho, contraste e simetria aplicada.

**8° passo:** repete-se o mesmo processo para os demais *range-blocks*. O conjunto de todos os códigos (tuplas) armazenados é que forma o código fractal que seria o arquivo da imagem comprimida.

Os passos para decodificar a imagem são.

**1°passo:** o código lê o arquivo da imagem comprimida, os parâmetros de compressão, e a resolução da imagem original.

**2°passo:** cálculo dos valores máximo e mínimo do tamanho dos *domain-blocks*.

**3°passo:** o código lê a imagem semente que vai ser usada na descompressão, e faz o particionamento quadtree recursivo da mesma.

**4°passo:** busca dos valores das transformações(código armazenado) e início do processamento das mesmas sobre a imagem semente já particionada. O processamento é realizado para cada *range-block* e logo após é feita a composição (união) dessas imagens resultantes.

**5°passo:** tendo lido a quantidade de iterações que são setadas, o processamento é realizado repetidas vezes para cada *range-block* até se atingir a quantidade informada de iterações. Assim a imagem resultante é a imagem recuperada pelo processo de compresssão fractal implementado neste trabalho.

Para se processar uma imagem (comprimir e descomprimir), deve-se realizar o ajuste dos seguinte parâmetros:

**Tolerância ou $e_{RMS}$ mínimo:** é um valor real que serve de limiar para que o algoritmo pare a busca ou faça um novo particionamento. Se tal valor é atingido, pára-se a busca e armazena-se a posição do *domain-block* em questão, caso contrário, realiza-se um novo particionamento.

**Níveis máximo e mínimo quadtree:** é a quantidade máxima e mínina de descidas em particionamentos. No processo podem ocorrer duas situações, ou o limiar de tolerância é atingido ou o nível máximo de quadtree. Os dois

são fatores que influenciam diretamente na quantidade de buscas realizadas.

**Número de bits para armazenar $s_i$ e $o_i$:** normalmente são utilizados os valores sugeridos por [7] de 5 bits para $s_i$ e 7 bits para $o_i$.

**Tipo de busca:** escolhe-se entre realizar as buscas nas 3 superclasses, nas 24 subclasses ou em todas.

**Número de iterações:** como já foi citado neste capítulo, geralmente após a oitava iteração praticamente não existe diferença na imagem, sendo assim, normalmente escolhe-se dez iterações, uma vez que uma quantidade maior só aumentaria o tempo de processamento.

**Pós-processamento da imagem:** é realizado com o intuito de suavisar os efeitos de blocagem, mas a opção por não realizá-lo em nada muda o processo de compressão.

### 3.8.3 Estrutura e ambiente de implementação

O método de compressão fractal utilizando particionamento quadtree e acelerador de Fischer [7], foi implementado no seguinte ambiente computacional: Celeron 2,40 GHz, 512 MB de RAM, sistema operacional Windows XP e linguagem de programação Java, que é uma linguagem livre, de código aberto, multi plataforma, alto desempenho, orientada a objetos e dispõe de um vasto conjunto de bibliotecas(APIs) e documentação, mostrando-se bastante eficiente no processamento de imagens.

## 3.9 Testes realizados

Para os testes realizados nesta seção foram utilizadas imagens extremamente conhecidas na literatura científica, tais como *Lena*, *Peppers* e *Cameraman*. Estas imagens e várias outras foram testadas em formato quadrado,

variando tamanho, imagem semente para a reconstrução e demais parâmetros de compressão e descompressão.

As Figuras 3.7 a 3.13 e a Tabela 3.1 mostram os resultados de alguns testes com imagens $256 \times 256$ pixels.

Figura 3.7: Imagem original da *Lena* e imagem recuperada a $0,19$ bpp com PSNR $= 30,57$ dB.

Figura 3.8: Imagem original *Goldhill* e imagem recuperada a $0,19$ bpp com PSNR $= 28,73$ dB.

Os resultados obtidos nas Figuras 3.7 a 3.13 foram bons em termos de compromisso entre PSNR e taxa de compressão, esses valores estão dentro da

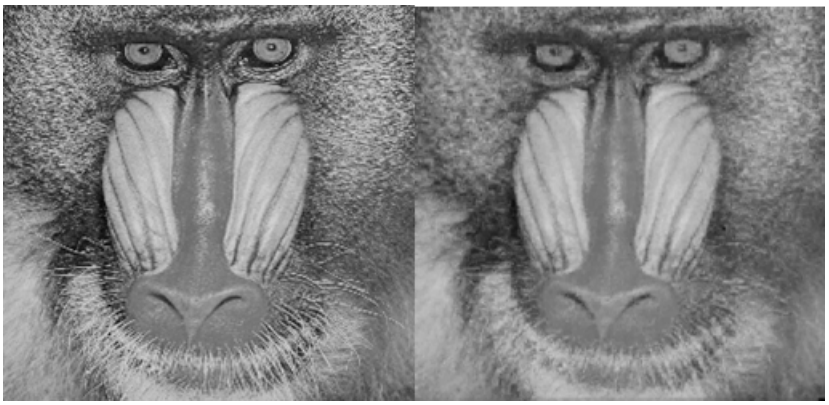

Figura 3.9: Imagen original *Mandrill* e imagem recuperada a $0,19$ bpp com PSNR $= 21,93$ dB.

Figura 3.10: Imagem original da *Pepeers* e imagem recuperada a $0,19$ bpp com PSNR $= 29,49$ dB.

Figura 3.11: Imagen original *Bird* e imagem recuperada a $0,19$ bpp com PSNR $= 36,09$ dB.

Figura 3.12: Imagem original *Bridge* e imagem recuperada a $0,19$ bpp com PSNR $= 26,49$ dB.

Figura 3.13: Imagem original *Cameraman* e imagem recuperada a $0,19$ bpp com PSNR $= 26,83$ dB.

média encontrada na literatura científica [7, 11], e valores bastante inferiores em relação ao tempo de execução. As imagens foram testadas variando-se também o tamanho, a imagem semente usada na reconstrução, a tolerância, os valores máximo e mínimo de nível quadtree.

O objetivo dos testes era ajustar os valores que garatissem uma melhor fidelidade, sem uma grande discrepância nas taxas de compressão e de PSNR. Nesses testes pode-se observar que quando a tolerância era zero tinha-se uma taxa de erro menor, isto ocorreu porque neste caso, mais divisões eram necessárias, resultando em blocos menores, priorizando assim os detalhes e portanto, uma maior fidelidade à imagem. Variando-se as imagens semente, as diferenças encontradas foram quase insignificantes, e pequenas diferenças no tempo de execução. Na maioria delas, o menor tempo de execução foi obtido quando utilizou-se como imagem semente um quadrado cinza. Pode-se verificar também que a partir da oitava iteração os valores resultantes se estabilizavam, o que tornava desnecessário a execução de várias iterações que aumentaria o tempo de processamento.

Os resultados obtidos na Tabela 3.1 mostram a eficiência do método im-

plementado neste trabalho em termos de taxa de compressão, de PSNR e principalmente em relação ao tempo de execução que teve uma melhora bastante significativa, o que pode ser justificado escolha da linguagem de implementação. Nas Figuras 3.7 a 3.13 foi mantida a mesma taxa de compressão para imagens do mesmo tamanho com a finalidade de comparar a eficiência do método em se tratando de diferentes imagens.

Tabela 3.1: Resultados obtidos nos testes realizados

| Imagem (256 × 256) | Taxa(bpp) | PSNR(dB) | Tempo execução(ms) |
|---|---|---|---|
| *Lena* | 0, 19 | 30, 57 | 766 |
| *Peppers* | 0, 19 | 29, 49 | 781 |
| *Cameraman* | 0, 19 | 26, 83 | 750 |
| *Mandrill* | 0, 19 | 21, 93 | 719 |
| *Bird* | 0, 19 | 36, 09 | 719 |
| *Bridge* | 0, 19 | 26, 49 | 734 |
| *Goldhill* | 0, 19 | 28, 73 | 781 |

## 3.10 Considerações Finais deste Capítulo

Este capítulo apresentou uma idéia geral de diversos tipos de compressão de imagens e a evolução de algumas técnicas de compressão bem como suas vantagens e/ou desvantagens. Mostrou dois tipos de particionamento para compressão fractal, analisa e compara os mesmos. Foram mostrados também a implementação do método utilizando particionamento quadtree e resultados de testes realizados com imagens quadradas padrão.

A partir da análise dos resultados obtidos com os testes é possível con-

cluir que o algorítmo de compressão fractal aqui implementado é bastante satisfatório em termos de equilíbrio entre taxa de compressão e qualidade da imagem e que houve melhora substancial em termos de processamento das imagens, fato que possibilitou a realização de vários testes em tempo bastante reduzido.

O próximo capítulo mostra o estudo de um sistema de reconhecimento de íris, a aplicação do método de compressão fractal implementado neste trabalho e da técnica de compressão JPEG2000 a essas imagens, bem como a análise dos resultados em termos de taxa de falsa aceitação e falsa rejeição. São mostrados os resultados obtidos nos testes realizados e finalmente são realizadas conclusões sobre esses resultados.

# Capítulo 4

# Efeitos da Compressão Fractal no Reconhecimento de Íris

## 4.1 Biometria

A palavra *biometria (bio = vida e metria = medida)* é um conjunto de técnicas utilizadas para se medir e analisar características humanas, com o objetivo de identificar as pessoas com certa margem de segurança. Atualmente é muito usada na identificação criminal, controle de ponto, controle a acesso, dentre outros.

Na prática, primeiro o usuário se registra em um sistema, coletando assim, imagens de suas características, que serão convertidas em um código padrão que é armazenado e posteriormente utilizado para comparação. Tal comparação pode ser realizada de duas formas diferentes: como *verificação* ou como *identificação*. No caso da verificação o sistema apenas confere se o usuário que se apresenta é realmente determinada pessoa, liberando ou não seu acesso. Esse sistema é chamado de $1-1$(um-para-um). Para o caso de identificação considera-se o dado biométrico do usuário e se realiza uma

busca em um banco de dados comparando até que se encontre ou não um registro idêntico ao seu, esse sistema é chamado $1 - n$(um-para-muitos) e na prática, pode-se ter uma certa margem de erro.

Dentre as técnicas biométricas utilizadas as mais conhecidas são [15]:

- Veias: realizado pelo escaneamento dos vasos sanguíneos dos dedos ou da mão inteira. Tem média confiabilidade, difícil de fraudar, alto custo.

- Impressão digital: utiliza a disposição de estrias bem finas que se localizam nas pontas dos dedos. O método é rápido, com alta confiabilidade e baixo custo. Mas existem obstáculos como o desgaste manual da pessoa ou problemas com o dispositivo de captura como oleosidade ou encardimento.

- Reconhecimento da face: uma câmera captura a imagem da face e um código é gerado para comparação. A confiabilidade não é grande mas é um método rápido e de baixo custo. Porém não identifica gêmeos idênticos.

- Reconhecimento pela retina: a retina é um nervo bem fino que fica localizada atrás do olho. Para reconhecimento utiliza-se a disposição de vasos sanguíneos localizados nela. O método é confiável, porém de leitura difícil e incômoda na medida em que exige que a pessoa olhe fixamente para um ponto de luz, alto custo.

- Reconhecimento de voz: captura-se amostras por um microfone ou telefone. Pouco confiável, podem ocorrer problemas com ruídos no ambiente, por mudança na voz do usuário devido a gripes ou estresse, demora no processo de cadastramento e leitura, porém tem baixo custo.

- Geometria da mão: não é muito confiável, problemas com anéis, o usuário precisa encaixar a mão na posição correta, médio custo.

- Reconhecimento da assinatura: pouco confiável, algumas assinaturas mudam com o passar do tempo, também existem problemas na velocidade e pressão na hora da escrita, médio custo.

Este capítulo mostra o estudo de um sistema de reconhecimento de íris, a aplicação do método de compressão fractal implementado neste trabalho e da técnica de compressão JPEG2000 a essas imagens, bem como a análise dos resultados em termos de taxa de falsa aceitação e falsa rejeição. São mostrados os resultados obtidos nos testes realizados e finalmente são realizadas conclusões sobre esses resultados.

## 4.2 Reconhecimento de Íris

### 4.2.1 Estrutura da íris

A íris é a região colorida do olho. É uma membrana contrátil, diferentemente pigmentada nos indivíduos. Quando existe pouca pigmentação ela se torna azul (a luz de maior comprimento de onda, parte vermelha do espectro, penetra com maior facilidade nos tecidos e é absorvida, por outro lado os comprimentos de onda menores são dispersos) e com muita pigmentação ela é mais escura até se tornar marrom. É um órgão interno, porém externamente visível, centralizada pela pupila e sua extremidade é formada por um conjunto de fibras musculares lisas. A íris possui diâmetro de aproximadamente $12mm$ e a pupila ocupa de 10% a 80% do diâmetro da íris [16]. Suas características são geradas de forma aleatória, a única que sofre influência de fator genético é a cor. A íris esquerda é diferente da direita em uma

mesma pessoa e gêmeos, mesmo que univitelinos possuem íris diferentes. A partir dos primeiros anos de vida suas várias características permanecem estáveis e fixas, o que torna interessante e confiável seu uso em sistemas de reconhecimento.

A Figura 4.1 mostra uma íris humana.

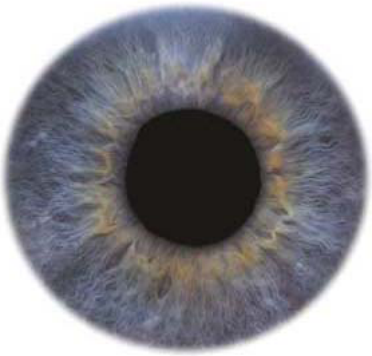

Figura 4.1: Íris humana

### 4.2.2 Sistema de reconhecimento de íris

A íris começou a ser utilizada como um sistema biométrico na década de 80 pelo oftalmologista francês *Alphonse Bertillon*. Esse sistema foi patenteado e desde então se tornou bastante usado [16].

Para a aquisição das imagens do olho deve-se utilizar alta resolução(no mínimo 70 pixels entre a borda da pupila e a borda externa da íris). Usa-se luz infra-vermelha que é melhor para a visualização de detalhes de íris escuras. A distância que a pessoa deve estar do equipamento de captura depende da resolução e varia de centímetros a metros. No processamento são utilizadas imagens em tons de cinza, pois a cor não é importante no reconhecimento.

## 4.3 Etapas de Processamento

Após a aquisição da imagem do olho devem ser executadas as seguintes etapas: localização da região da íris na imagem do olho, normalização, codificação e comparação.

### 4.3.1 Localização

Quando a imagem do olho é captada, é na verdade a imagem completa de um olho e não somente da íris. Assim, para a realização do processo de reconhecimento antes é necessário localizar e isolar a íris na imagem. Para isso são utilizados algorítmos de detecção de bordas e detecção de círculos [17, 18, 19]. Quando as pálpebras ou os cílios cobrem parte da íris, somente a porção da imagem entre as pálpebras superir e inferior é utilizada. Pereira [16], implementou-se o algoritmo de localização de íris desenvolvido por Libor Masek [19], que utilizou a Transformada de Hough Circular [9, 21] para localizar os círculos da borda da íris e da pupila.

A Transformada de *Hough* (TH) é aplicada após a imagem passar por um processo de detecção de bordas, dessa maneira só se utiliza no processamento os pixels da borda. A TH define um mapeamento entre o espaço de parâmetros de um círculo, gerando uma matriz de acumulação de votos que no caso em estudo tem dimensão 3. Primeiro faz-se a detecção da íris e depois da pupila, separadamente. Tal processamento exige esforço computacional intenso, não sendo adequado para aplicações em tempo real. Seguindo a sugestão de Wildes [22], utilizou-se o gradiente vertical para detectar a borda externa da íris e gradiente horizontal para detectar as pálpebras, o que reduz a interferência das pálpebras e torna a localização do círculo mais precisa, rápida e eficiente. Para a pupila utilizou-se gradientes verticais e horizontais

. Para isolar a pálpebra admitiu-se que sua borda pode ser aproximada por um segmento de reta, utilizando-se a Transformada de Hough Linear. Para isolar os cílios, estabeleceu-se um limiar e como eles são mais escuros que o restante da imagem, os pixels em tons de cinza mais escuros que o limiar são excluídos.

A etapa de localização da íris é muito importante para a eficiência no processo de reconhecimento, pois se ela não for localizada corretamente o código para comparação será corrompido provocando erros de reconhecimento.

### 4.3.2 Normalização

O processo de normalização tem o objetivo de gerar imagens com dimensões constantes uma vez que as regiões da íris e da pupila não são concêntricas. Podem também existirem problemas com o eixo ótico da câmera ou inclinação da cabeça no momento da aquisição das imagens.

A técnica proposta por *Daugmam* gera uma representação retangular da região anular da íris tendo o centro da pupila como ponto de referência. Nessa técnica foram escolhidos uniformemente 10 pontos (pixels) ao longo de cada linha radial o que define a dimensão vertical da região retangular. Foram escolhidas, também uniformemente, 40 linhas radiais que definem a dimensão vertical da região retangular. Esses números devem ser constantes independente da largura entre as bordas da pupila e da íris. Também é necessário gerar uma máscara de ruídos que marca os pixels pertencentes às pálpebras e aos cílios e os substitui por pixels com valores médios de nível de cinza dos demais pixels.

### 4.3.3 Codificação

Na fase de codificação é criado o código binário que informa as características mais significativas da íris e a máscara de ruídos que mostra as regiões de interferência das pálpebras e cílios. Utiliza-se uma variação do filtro de *Gabor* conhecida como Log-Gabor proposta por *Field* [25]. Com esse filtro obtém-se simultaneamente uma localização espacial e de frequência de um determinado sinal. A frequência central ($f_c$) é determinada pela frequência de uma senóide/cossenóide e a largura de faixa ($\sigma$) pela largura de uma Gaussiana representada em uma escala logaritmica[19]. A resposta em frequência do filtro Log-Gabor é dada pela Equação 4.1:

$$G(f) = exp\left[-\frac{(log(f/f_c))^2}{2.log(\sigma/f_c)^2}\right] \tag{4.1}$$

A codificação das características é realizada pela convolução da representação normalizada da íris com o filtro de Log-Gabor $1D$. Neste trabalho, utilizou-se os parâmetros determinados por Libor Maseck ($f_c = 18$ $pixels, \sigma/f = 0,5$), e apenas 1 filtro.

### 4.3.4 Comparação

Com o código pronto, é necessário utilizar uma métrica para comparar 2 códigos, sejam eles da mesma íris (intra-classe), ou de íris diferentes (inter-classe). A *métrica de Hamming* foi adotada por já trabalhar com padrões binários, fazendo comparação *bit* a *bit* dos códigos e calculando a razão entre a quantidade de *bits* que não se correlacionam e a quantidade total de *bits* comparados. Foi utilizado o algorítmo da *distância de Hamming*(DH)

proposto por Daugman [17, 18, 23, 25] que usa no cálculo somente os *bits* pertencentes à região da íris.

Na teoria a DH pode variar de $0,0$ a $1,0$, mas na prática se aproxima de 0 quando compara dois códigos de mesma íris e vale $0,5$ quando compara dois códigos de íris diferentes, pois a independência implica que a correspondência entre 2 *bits* é totalmente aleatória. Para que o método se torne mais preciso, efeitos de desalinhamento como rotação da câmera ou desalinhamento da cabeça foram minimizados da seguinte forma: depois que a DH entre os dois códigos foi calculada fez-se deslocamentos horizontais de 2 *bits* nos mesmos e calculou-se novamente a DH. No final escolhe-se o menor valor de DH que correponde à maior semelhança entre os códigos, o que gera o reconhecimento da íris.

No processo de reconhecimento da íris, são tratados dois tipos de taxas de erro. Uma delas é a taxa de erro de falsa aceitação (False Accept Rate - FAR), que representa a probabilidade de um impostor ser aceito pelo sistema. A outra é a taxa de errro de falsa rejeição (False Reject Rate - FRR), que mede a probabilidade de um indivíduo apto ser considerado impostor. Essas taxas são calculadas à partir da sobreposição entre as distribuições intra e inter-classe[16]. Um limiar (ponto de separação) entre as duas distribuuições deve ser escolhido de maneira a minimizar a soma das duas taxas, com a finalidade de se obter um certo equilíbrio entre elas.

## 4.4 Aplicação da Compressão Fractal

Com o algoritmo de compressão fractal implementado e testado em imagens padrão da literatura científica, o objetivo agora é o de comprimir imagens da íris humana com tal método, com a finalidade de minimizar o espaço

gasto para armazenamento das imagens no banco de dados, sem no entanto, prejudicar o sistema de reconhecimento de íris implementado por Masek [19]. As imagens de íris utilizadas neste trabalho são todas de tamanho $200 \times 150$ *pixels*. Para as simulações foram testadas aproximadamente 1202 imagens do banco de dados UBIRIS - Seção 1[27].

As Figuras 4.2 a 4.10 mostram os testes realizados com imagens de íris de pessoas diferentes. Nesses testes e nos demais variou-se os parâmetros de modo a conseguir um balanceamento entre taxa de compressão, PSNR e tempo de execução. Com esses testes foi possível perceber que um resultado satisfatório segundo Fisher [7] é obtido, principalmente, se for utilizada uma tolerância de 2.0, o que equilibra a taxa de compressão, nesse caso de aproximadamente $0,7$ *bpp*, mantendo uma boa fidelidade com o erro dentro do esperado e inclusive, com tempo de execução bastante satisfatório. Isso é importante uma vez que as imagens precisam ser comprimidas, porém, com perda mínima, com a finalidade de que o sistema de reconhecimento continue eficaz. Fazendo-se uso de uma rotina que busca as imagens, comprime, descomprime e as salva, foi possível processar todas as imagens do banco de dados UBIRIS - Seção 1 [27] em um tempo bastante reduzido, cerca de 50 minutos, o que é uma grande vantagem, uma vez que foram processadas muitas imagens em taxas de compressão diferentes.

A Tabela 4.1 mostra os testes realizados com imagens de íris de 3 pessoas, com variadas taxas de compressão. Observa-se variações na PSNR e até mesmo no tempo de execução, isto pode ser explicado pelo fato de que cada imagem possui uma tonalidade diferente e como o método trabalha com os níveis de cinza da imagem, os valores mudam quando se passa de uma imagem mais clara para uma imagem mais escura por exemplo, mas, apesar das mudanças, permanecem bons resultados.

Com isso, pode-se concluir que o algoritmo implementado neste trabalho e testado possui bastante eficácia na compressão de variados tipos de imagens, inclusive, na compressão das imagens de íris do banco de dados UBIRIS - Seção 1[27]. Embora aqui sejam mostradas poucas imagens, vários testes foram realizados com o intuito de variar os parâmetros para se observar os efeitos da compressão no reconhecimento de íris, onde para tal foram utilizadas várias taxas de compressão distintas.

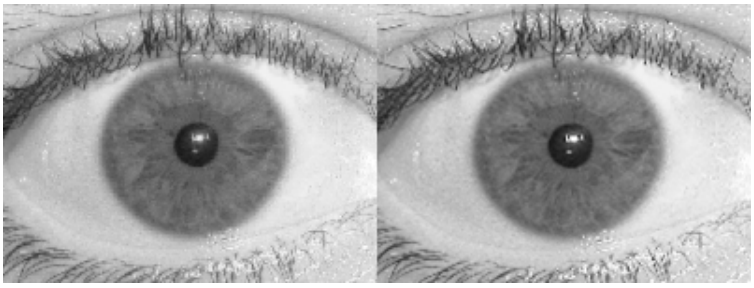

Figura 4.2: Imagem original *Img_1_1_1* do banco de dados UBIRIS-Seção1 [27] e imagem recuperada a $0,7$ bpp com PSNR $= 29,16$ dB.

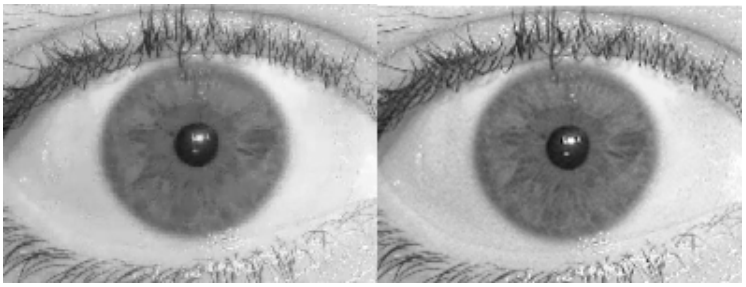

Figura 4.3: Imagem original *Img_1_1_1* do banco de dados UBIRIS-Seção1 [27] e imagem recuperada a $0,5$ bpp com PSNR $= 28,52$ dB.

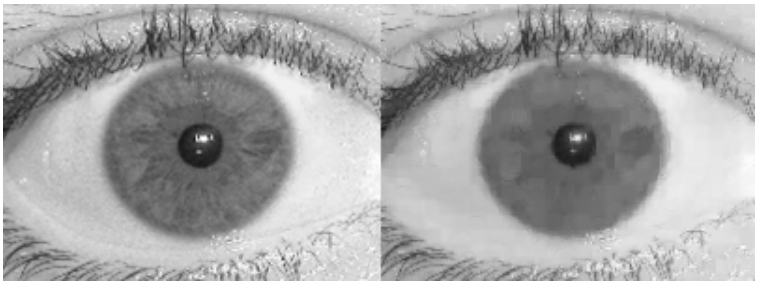

Figura 4.4: Imagem original *Img_1_1_1* do banco de dados UBIRIS-Seção1 [27] e imagem recuperada a $0,3$ bpp com PSNR $= 27,9$ dB.

Figura 4.5: Imagem original *Img_76_1_1* do banco de dados UBIRIS-Seção1 [27] e imagem recuperada a $0,7$ bpp com PSNR $= 32,91$ dB.

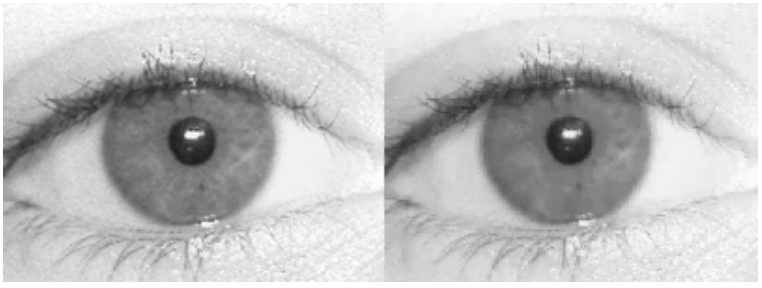

Figura 4.6: Imagem original *Img_76_1_1* do banco de dados UBIRIS-Seção1 [27] e imagem recuperada a $0,5$ bpp com PSNR $= 31,63$ dB.

Figura 4.7: Imagem original *Img_76_1_1* do banco de dados UBIRIS-Seção1 [27] e imagem recuperada a $0,3$ bpp com PSNR $= 30,76$ dB.

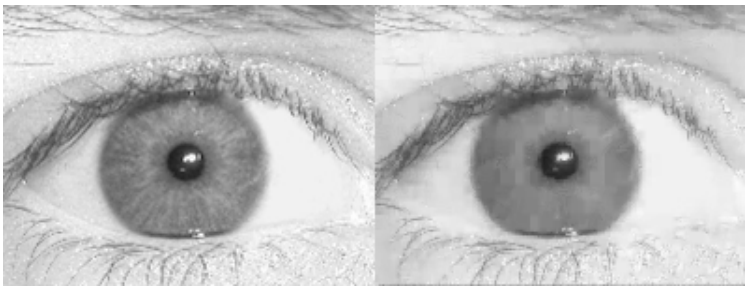

Figura 4.8: Imagem original *Img_161_1_1* do banco de dados UBIRIS-Seção1 [27] e imagem recuperada a $0,7$ bpp com PSNR $= 30,06$ dB.

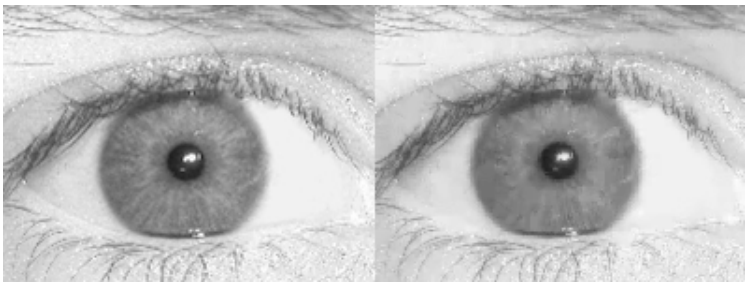

Figura 4.9: Imagem original *Img_161_1_1* do banco de dados UBIRIS-Seção1 [27] e imagem recuperada a $0,5$ bpp com PSNR $= 29,34$ dB.

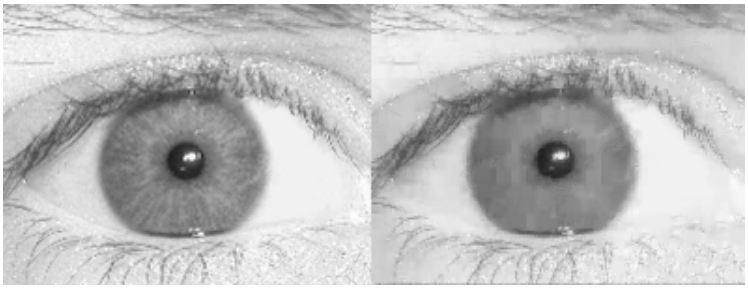

Figura 4.10: Imagem original *Img_161_1_1* do banco de dados UBIRIS-Seção1 [27] e imagem recuperada a $0,3$ bpp com PSNR $= 27,86$ dB.

## 4.5 Efeitos da Compressão no Reconhecimento

A utilização do sistema biométrico de reconhecimento de íris vem sendo muito ampliada atualmente, por se tratar de um método bastante seguro, que não exige grandes gastos. Em decorrência da grande utilização desse sistema, um problema que surge é como armazenar tais imagens e seus templates ocupando o mínimo de espaço possível. Neste ponto a compressão das imagens, inclusive a compressão fractal pode ser bastante vantajosa, diminuindo espaço de armazenamento sem prejudicar o processo de reconhecimento.

Assim, o objetivo deste trabalho agora é considerar as imagens geradas a partir do processo de compressão fractal implementado nesta dissertação e fazer uma comparação binária com a finalidade de que a imagem continue a ser reconhecida pelo método apresentado por Masek [19], porém com a vantagem de diminuir espaço de armazenamento no banco de dados, mantendo um limiar de erro aceitável.

O sistema utilizado nos testes foi o mesmo descrito nas seções anteriores deste capítulo, implementado por *Libor Maseck* [19]. Foram realizadas várias comparações intra-classe (de mesmas íris) e inter-classe (de íris diferentes)

Tabela 4.1: Testes realizados com imagens de íris [27]

| Imagem (200 × 150) | Taxa(bpp) | PSNR(dB) | Tempo execução(ms) |
|---|---|---|---|
| *Img_1_1_1* | 0,3 | 27,9 | 1203 |
| | 0,5 | 28,52 | 1657 |
| | 0,7 | 29,16 | 2804 |
| *Img_76_1_1* | 0,3 | 30,76 | 1046 |
| | 0,5 | 31,63 | 1469 |
| | 0,7 | 32,91 | 2406 |
| *Img_161_1_1* | 0,3 | 27,86 | 1187 |
| | 0,5 | 29,34 | 1750 |
| | 0,7 | 30,26 | 2532 |

com a finalidade de computar taxas de falsa-aceitação( FAR), de falsa rejeição (FRR), erro mínimo e ainda com o auxilio da *métrica de Hamming* medir a diferença dos *bits* dos templates gerados pelas imagens com compressão fractal em relação aos templates obtidos das imagens originais.

As Tabelas 4.2 a 4.4 mostram os resultados obtidos com algumas imagens de íris também do banco de dados UBIRIS - Seção 1[27], bem como dados de simulações usando o método de *Libor Maseck* [19].

Para se chegar aos resultados apresentados nas tabelas 4.2 a 4.4 foram testadas imagens comprimidas a taxas de aproximadamente 0,7 bpp (1200 imagens) 0,5 bpp (665 imagens) e 0,3 bpp (535 imagens). É importante salientar que na hora de comprimir todas essas imagens em variadas taxas, foi de grande importância ter em mãos um algoritmo rápido, que possibilitou o processamento em tempo reduzido, como foi citado na seção 4.4 deste capítulo.

Nas simulações usando o método de *Libor Maseck*[19], foram realizadas 1709 comparações intra-classe e 465352 comparações inter-classe para as imagens comprimidas a uma taxa de $0,7$ bpp, sendo que após tabulados os valores de FAR e FRR, obteve-se um limiar para erro mínimo de $0,32$. Para os resultados com imagens comprimidas a taxa de aproximadamente $0,5$ bpp foram realizadas 937 comparações intra-classe e 138191 comparações inter-classe, obtendo-se limiar para erro mínimo de $0,334$ e para as imagens comprimidas uma taxa de aproximadamente $0,3$ bpp foram realizadas 737 comparações intra-classe e 90214 comparações inter-classe chegando-se a um limiar para erro mínimo de $0,321$. Os erros mínimos obtidos nas comparações de FAR e FRR, bem como os limiares obtidos para os templates das imagens originais são mostrados nas Tabelas 4.2 a 4.4 e percebe-se que para as imagens comprimidas a taxa de erro foi maior, e que proporcionalmente, essa taxa aumenta a medida que se comprime mais a imagem.

Depois de realizadas as simulações, comparou-se os templates obtidos das imagens originais com os templates obtidos das imagens comprimidas e, pelo cálculo da distância de Hamming (DH) pode-se verificar que: para as 1200 imagens comprimidas a uma taxa de $0,7$ bpp aproximadamente $8,63\%$ dos bits mudaram com a compressão. Para as 665 imagens comprimidas com taxa de $0,5$ bpp aproximadamente $15,73\%$ dos bits mudaram e para as 535 imagens comprimidas a uma taxa de $0,3$ bpp, aproximadamente $15,46\%$. A Tabela 4.5 mostra esses valores.

Com os resultados obtidos na Tabela 4.5 pode-se observar, que mesmo ocorrendo mudanças nos bits das imagens, o que, obviamente foi ocasionado pela compressão da imagem, o sistema ainda sim é bastante confiável no reconhecimento de íris, isso mostra a vantagem da compressão fractal.

Apesar de vários trabalhos, inclusive este, mostrarem o quanto a com-

Tabela 4.2: Resultados obtidos com taxa de compressão Fractal de $0,7$ bpp.

| Taxa $= 0,7$ | Originais | Recuperadas |
|---|---|---|
| FARm | $0,97\%$ | $0,68\%$ |
| FRRm | $1,81\%$ | $2,17\%$ |
| Erro mínimo | $2,78\%$ | $2,84\%$ |
| Limiar para erro mínimo | $0,333$ | $0,32$ |

Tabela 4.3: Resultados obtidos com taxa de compressão Fractal de $0,5$ bpp.

| Taxa $= 0,5$ | Originais | Recuperadas |
|---|---|---|
| FARm | $0,71\%$ | $1,73\%$ |
| FRRm | $1,81\%$ | $2,35\%$ |
| Erro mínimo | $2,53\%$ | $4,08\%$ |
| Limiar para erro mínimo | $0,337$ | $0,334$ |

Tabela 4.4: Resultados obtidos com taxa de compressão Fractal de $0,3$ bpp.

| Taxa $= 0,3$ | Originais | Recuperadas |
|---|---|---|
| FARm | $1,32\%$ | $2,28\%$ |
| FRRm | $1,09\%$ | $2,85\%$ |
| Erro mínimo | $2,41\%$ | $5,13\%$ |
| Limiar para erro mínimo | $0,328$ | $0,321$ |

Tabela 4.5: Distância de Hamming x taxas de compressão Fractal

| Taxa(bpp) | 0,7 | 0,5 | 0,3 |
|---|---|---|---|
| Mudança nos bits | 8,63% | 15,73% | 15,46% |

pressão Fractal é vantajosa atualmente, esse tipo de compressão ainda tem sido pouco utilizada em comparação com algumas outras técnicas, tais como JPEG ou JPEG2000 já comentadas no Capítulo 2 desta dissertação. Assim, com o intuito de comparar o desempenho da compressão Fractal com essas técnicas mais populares, foram realizados os mesmos testes, sob as mesmas condições com imagens obtidas pela técnica de compressão JPEG2000. As mesmas imagens do banco de dados UBIRIS - Seção 1 [27] foram comprimidas com JPEG2000, utilizando as mesmas taxas obtidas para a Compressão Fractal. Para tal, foi utilizado o software de compressão *Jasper*[28], desenvolvido por *Michael Adams* afiliado ao Digital Signal Processing Group (DSPG) no Departamento de Engenharia Elétrica e de Computação da University of Victoria. O software, desenvolvido para ser operado em Linux, foi executado num AMD ATHLON 64 3600 de 400 MHz, 1 GB de RAM, sistema operacional Arch Linux 64 versão 0.8.

Nas simulações foram realizadas 1697 comparações intra-classe e 458623 comparações inter-classe para as imagens comprimidas a uma taxa de 0,7 bpp, sendo que após tabulados os valores de FAR e FRR, obteve-se um limiar para erro mínimo de 0,321. Para os resultados com imagens comprimidas a taxa de aproximadamente 0,5 bpp foram feitas 931 comparações intra-classe e 134529 comparações inter-classe, obtendo-se limiar para erro mínimo de 0,332 e para as imagens comprimidas uma taxa de aproximadamente 0,3 bpp foram feitas 694 comparações intra-classe e 81116 comparações inter-

classe chegando-se a um limiar para erro mínimo de $0,328$. Os erros mínimos obtidos nas comparações de FAR e FRR, bem como os limiares obtidos para os templates das imagens originais são mostrados nas Tabelas 4.6 a 4.8 e pode-se verificar que as diferenças são muito pequenas entre as imagens comprimidas e as originais.

Tabela 4.6: Resultados obtidos para a taxa de compressão JPEG2000 de $0,7$ bpp.

| Taxa $= 0,7$ | Originais | Recuperadas |
|---|---|---|
| FARm | $0,97\%$ | $0,55\%$ |
| FRRm | $1,81\%$ | $2,24\%$ |
| Erro mínimo | $2,78\%$ | $2,79\%$ |
| Limiar para erro mínimo | $0,333$ | $0,321$ |

Tabela 4.7: Resultados obtidos para a taxa de compressão JPEG2000 de $0,5$ bpp.

| Taxa $= 0,5$ | Originais | Recuperadas |
|---|---|---|
| FARm | $0,71\%$ | $0,47\%$ |
| FRRm | $1,81\%$ | $1,72\%$ |
| Erro mínimo | $2,53\%$ | $2,18\%$ |
| Limiar para erro mínimo | $0,337$ | $0,332$ |

Também foram comparados os templates obtidos pelas imagens originais com os templates obtidos com as imagens comprimidas utilizando JPEG2000. Pode-se verificar nessa comparação que para imagens comprimidas a uma

Tabela 4.8: Resultados obtidos para a taxa de compressão JPEG2000 de $0,3$ bpp.

| Taxa $= 0,3$ | Originais | Recuperadas |
|---|---|---|
| FARm | $1,32\%$ | $1,13\%$ |
| FRRm | $1,09\%$ | $1,30\%$ |
| Erro mínimo | $2,41\%$ | $2,43\%$ |
| Limiar para erro mínimo | $0,328$ | $0,33$ |

Tabela 4.9: Distância de Hamming x taxas de compressão JPEG 2000

| Taxa(bpp) | $0,7$ | $0,5$ | $0,3$ |
|---|---|---|---|
| Mudança nos bits | $9,16\%$ | $10,32\%$ | $11,2\%$ |

taxa de $0,7$ bpp aproximadamente $9,16\%$ dos bits mudaram com a compressão. Para imagens comprimidas com taxa de $0,5$ bpp aproximadamente $10,32\%$ dos bits mudaram e para as imagens comprimidas a uma taxa de $0,3$ bpp, aproximadamente $11,06\%$. Tais valores são mostrados na tabela 4.9.

Pode-se observar nas Tabelas 4.6 a 4.8 que nas imagens obtidas pela técnica de compresão JPEG2000 a discrepância de erro em relação aos templates obtidos das imagens originais foi muito pequena. O que indica que seria mais vantajoso ainda utilizar essa técnica junto com o reconhecimento de íris, uma vez que ela diminui o espaço de armazenamento sem prejudicar a eficácia do sistema.

## 4.6 Considerações Finais Deste Capítulo

Este capítulo mostrou o estudo de um sistema de reconhecimento de íris, a aplicação do método de compressão Fractal implementado neste trabalho e da técnica JPEG2000 a essas imagens e a análise dos efeitos da compressão em termos de taxa de falsa aceitação e falsa rejeição. Foram mostrados os resultados obtidos nos testes realizados e finalmente foram realizadas conclusões sobre esses resultados.

Se forem observados os dois conjuntos de resultados das simulações utilizando compressão Fractal e JPEG2000, pode-se verificar a discrepância entre os efeitos dessas duas técnicas de compressão junto com o reconhecimento de íris, uma vez que existem diferenças entre as taxas de FAR, de FRR e de erro entre elas. Os valores obtidos quando se utiliza compressão JPEG2000 são melhores, apesar disso, é vantagem utilizar tanto uma técnica quanto a outra junto com o sistema de reconhecimento de íris, pois para essas duas técnicas a taxa de erro ainda é aceitável. A diferença é que a técnica JPEG2000 é bastante popular e largamente utilizada, ao passo que a compressão Fractal vem emergindo aos poucos, mas mostra ótimos resultados em termos de equilíbrio entre taxa de compressão, qualidade da imagem e tempo de processamento, o que a torna uma técnica bastante interessante de ser explorada.

O próximo capítulo mostra as conclusões dos resultados obtidos nos testes realizados neste trabalho, as contribuições desta dissertação e as propostas para trabalhos futuros.

# Capítulo 5

# Conclusões Gerais

## 5.1 Análise no contexto

Em uma ampla gama de aplicações a digitalização de imagens vem tomando proporções cada vez maiores atualmente. Portanto, a necessidade da utilização de técnicas de compressão de dados eficientes é imperativa em termos de espaço e tempo, espaço para armazenamento e rapidez nas transmissões.

Neste contexto, a compressão fractal vem se despontando aos poucos como técnica poderosa, unindo eficácia e qualidade.

Os dois conjuntos de resultados obtidos das simulações utilizando as compressões fractal e JPEG2000 mostram a discrepância entre os efeitos dessas duas técnicas de compressão junto com o reconhecimento de íris, uma vez que existem diferenças entre as taxas de FAR, de FRR e de erro entre elas. Os valores obtidos quando se utiliza compressão JPEG2000 são melhores, apesar disso, é vantagem utilizar tanto uma técnica quanto a outra junto com o sistema de reconhecimento de íris, pois para essas duas técnicas a taxa de erro ainda é aceitável. A diferença é que a técnica JPEG2000 é bastante popular e largamente utilizada, ao passo que a compressão Fractal vem emergindo aos

poucos, mas mostra ótimos resultados em termos de equilíbrio entre taxa de compressão, qualidade da imagem e tempo de processamento, o que a torna uma técnica bastante interessante de ser explorada.

Dos resultados obtidos nos testes realizados neste trabalho, pode-se concluir que o método de compressão Fractal pode ser utilizado em simulações de sistemas de reconhecimento de íris sem prejudicar o mesmo, pois mesmo ocorrendo mudanças nos pixels das imagens e aumento na taxa de erro, ainda sim o sistema continua eficaz.

O mesmo pode ser observado quando se utiliza compressão JPEG2000, que também traz vantagens ao ser utilizada junto com o reconhecimento de íris, onde seu desempenho é melhor do que quando se utiliza compressão Fractal, isso pode ser observado pela diferença entre as taxas de erro nesses métodos.

De modo geral a aplicação das técnicas de compressão Fractal e JPEG2000 no banco de dados UBIRIS - Seção 1 [27] trouxe grandes vantagens, pois a utilização dos templates de imagens obtidas com essas técnicas não ocasionou nenhum problema no processo de reconhecimento desenvolvido por Libor Masek [19], que se mostrou ainda bastante eficaz.

Outro fato que merece destaque é que, neste trabalho, tanto o algoritmo de compressão Fractal implementado quanto o software de compressão JPEG2000 utilizados não foram desenvolvidos somente para compressão de imagens de íris, portanto adaptações nesses algorítmos poderiam trazer significativas melhoras na compressão, influenciando na interferência da mesma no sistema de reconhecimento.

## 5.2 Contribuições deste trabalho

Este trabalho mostrou como a técnica proposta por Fischer [7] é bastante eficiente em termos de taxa de compressão e qualidade das imagens, sendo que o algoritmo implementado nesta dissertação destacou-se principalmente em relação ao tempo de execução, causando uma queda bastante acentuada no mesmo, fato que possibilitou o processamento de muitas imagens (cerca de 1200) do banco de dados UBIRIS - Seção 1[27] em um tempo bastante reduzido. Esse fato merece destaque, uma vez que uma das muitas preocupações das técnicas de compressão de imagens é exatamente esta, ou seja, conseguir um algoritmo que equilibre taxa de compressão, qualidade e rapidez no processamento.

## 5.3 Prospostas para Futuros Trabalhos

O método implementado neste trabalho mostrou-se bastante satisfatório. No entanto, durante seu desenvolvimento surgiram idéias que futuramente podem ser exploradas para aumentar as contribuições no campo de processamento de imagens. Essas idéias são:

- OA utilização de técnicas híbridas de compressão como fractal-wavelet [11] pode trazer melhorias consideráveis na qualidade das imagens de íris uma vez que pode diminuir os efeitos de blocagem, e isso provavelmente influenciará menos no sistema de reconhecimento de íris;

- A combinação da compressão fractal com quantização vetorial poderia também melhorar a compressão da imagem da íris, pois Ribas [20] comprovou melhorias na qualidade das imagens codificadas com a combinação dessas duas técnicas em relação à cada uma delas separadamente;

- Os efeitos da compressão fractal no sistema de reconhecimento de íris foi testado apenas para um banco de dados restrito [27]. Assim, simulações considerando também outros bancos de dados, mostrariam uma consistência maior em termos dos efeitos dessa compressão ou mesmo na Compressão JPEG2000;

- O algoritmo implementado neste trabalho destacou-se pela eficácia em termos de tempo de execução, comparado-se por exemplo, com algoritmos implementados em Matlab. A implementação desse mesmo algoritmo utilizando outras linguagens de programação, bem como testes usando a mesma máquina com as mesmas imagens, permitiriam uma boa comparação em termos de escolha de software.

# Referências Bibliográficas


[1] BARNSLEY, M., *Fractals Everywhere*, Academic Press, San Diego, USA, 1993. 394 p.

[2] PEANO. Página web: *http://www.educ.fc.ul.pt/icm/icm99/icm14/peano.htm*.

[3] SIERPINSKI. Página web: *http://www.educ.fc.ul.pt/icm/icm99/icm48/sierpinski.htm*.

[4] SILVA, J. M., *Compressão de Imagens Médicas utilizando Fractais*, Universidade Federal de Uberlândia, Uberlândia, MG, Brasil, 2006. Relatório de Qualificação.

[5] FRACTAL . Página web: *http://pt.wikipedia.org/wiki/Fractal*.

[6] SILVA, J. M., *Introdução à Geometria Fractal e Aplicações*, Goiânia, GO, Brasil, 2006.

[7] FISHER, Y.,*Fractal Image Compression - Theory and Application*, New York: Springer- Verlag, 1995. 341 p. ISBN: 0-387-94211-4 (New York) - 3-540-94211-4 (Berlin).

[8] DOMINGUES, H. H., *Espaços Métricos e Introdução à Topologia*, São Paulo - SP - Brasil: Atual Ltda/EDUSP, 1992. 184 p.

[9] GONZALEZ, R. C.; WOODS, R. E., *Processamento de Imagens Digitais*, São Paulo - SP - Brasil: Editora Edgard Blücher Ltda, 2000. 509 p.

[10] BARNSLEY, M. F.; HURD, L. P., *Fractal Image Compression*, Welesley, MA: AK Peters, Ltd., 1993. 244 p. ISBN: 1-56881-000-8.

[11] SILVA, A. L. C. S., *Procedimentos para Método Híbrido de Compressão de Imagens Digitais utilizando Transformadas Wavelet e Codificação Fractal*, Universidade Estadual de Campinas, Campinas, SP, Brasil, 2005. Tese de doutorado.

[12] FISHER, Y., *Fractal Image Compression*, San Diego - California, 1992. SIGGRAPH 92, Curse Notes.

[13] JACQUIN, A., *A Fractal Theory of Iterated Markov Operators with Applications to Digital Image Compression*, PhD. Thesis, Georgia Institute of Technology, USA, 1989.

[14] CHULING, C., SHAOD, W., BINGZHE S., *A Fractal Image Coding Based on the Quadtree*, Department of Computer Science and Technology, Nanjing Institute of Posts and Telecommunications, 2 10003, Nanjing, PRC, 1998.

[15] BIOMETRIA. Página web: *http://pt.wikipedia.org/wiki/Biometria*.

[16] PEREIRA, M. B., *Uma Proposta para o Aumento da Confiabilidade de um Sistema de Reconhecimento de Íris e sua Implementação através de Algoritmo Genético*, Universidade Federal de Uberlândia, Uberlândia, MG, 2005. Dissertação de mestrado.

[17] DAUGMAN, J., *Iris Recognition*, American Scientist, vol.89, 2001.

[18] DAUGMAN, J., *How Iris Recognition Works*, Proceedings of 2002 Internation Conference on Image Processing, vol.1, 2002.

[19] MASEK, L., *Recognition of Human Iris Patterns for Biometric Identification*, Dissertação de mestrado, The University os Western Autralia, 2003.

[20] RIBAS, J. P. I. F., *Um Modelo de Compressão de Imagens Digitais Baseado em Codificação Fractal e Quantização Vetorial*, Universidade Federal de Uberlândia, Uberlândia, MG, 2008. Tese de Doutorado.

[21] JAMUNDÁ, T., *Reconhecimento de Formas: a Transformada de Hough*, Seminário Visão Computacional - CPGCC/UFSC, 2000.

[22] WILDES, R. P., ASMUTH, J. C., GREEN, G. L.,HSU, S. C., KOLCZYNSKI, R.,MATEY e MACBRIDE, S., *A System for Automated Iris Recognition*, Proceedings IEEE Workshop on Applications os Computer Vision, 1994.

[23] DAUGMAN, J., *High Confidence Visual Recognition of Person by a Test os Statistical Independence*, IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 15, 1993.

[24] DAUGMAN, J., *High Cofidence Personal Identification by Rapid Video Analisys os Iris Texture*, Proceedings of the IEEE.

[25] FIELD, D., *Relations Between the Statistics of Natural Images and the Response Properties of Cortical Cells*, Journal of the Optica Society of America, 1987.

[26] KOMINEK, J., *Fractal Image Compression*, Department of Computer Science, University of Waterloo, Ontario, Canadá, 1995.

[27] H. Proença, L. A. Alexandre, "UBIRIS: a noisy iris image database", ICIAP 2005, *13th Int. Conf. on Image Analysis and Processing*, Cagliari, Italy, 6-8 September 2005, Lect. Notes Comput. Sci., 3617, pp. 970-977, ISBN 3-540-28869-4. http//iris.di.ubi.pt

[28] ADAMS, M., Página web: *http://www.ece.uvic.ca/ mdadams/jasper/*.

[29] KOMINEK, J., *Advances in Fractal Compression for Multimedia Applications*, Department of Computer Science, University of Waterloo, Ontario, Canadá, 1995.

[30] JACKSON, D. J., MAHMOUD, W., STAPLETON, W. A., GAUGHAN, P. T., *Faster Fractal Image Compression Using Quadtree Recomposition*, Department of Eletrical Engineering, The University of Alobama, Tuscaloosa, AL 35487 USA, 1995.

[31] DEITEL, H. M., DEITEL, P. J., *Java Como Programar*, Tradução e revisão técnica: Carlos Arthur Lang Lisbôa, Ed. Bookman, Universidade Federal do Rio Grande do Sul, Porto Alegre, RS, Brasil, 2003.

[32] SANTOS, R., *Processamento de Imagens em Java*, Instituto Nacional de Pesquisas Espaciais, Brasil.

[33] RICARTE, I. L. M., *Programação Orientada a Objetos: Uma Abordagem com Java*, Universidade Estadual de Campinas, Campinas, SP, Brasil, 2001.

[34] SECCO, F. R., ROCHA T. T., *Fractais*, Universidade Federal de Santa Catarina, SC, Brasil, 2004. Trabalho da disciplina Teoria da Computação.

[35] SANTOS, C., *Fractais e Sistemas de Funções Iteradas*, Departamento de Matemática, Universidade de Lisboa, Lisboa, Portugal. Seminário de Matemática para o Ensino.

[36] GEOMETRIA. Página web: *http://www.geometrias.com.br/12.html*.

[37] GALABOV. M., *Fractal Image Compression*, International Conference on Computer Systems and Technologies - CompSysTech 2003.

[38] DAUGMAN, J., DOWNING, C., *Effect of Severe Image Compression on Iris Recognition Performance*, IEEE Transactions on Information Forensics and Security, vol. 3, nº - 1, March 2008.

[39] DAUGMAN, J., *New Methods in Iris Recognition*, IEEE Transactions on Systems, Man, and CyberneticsPart B: Cybernetics, vol. 37, nº - 5, October 2007.