Decomposição Triangular de Imagens: Uma Aplicação em Compressão

Vânia Cordeiro da Silva

Tese de Doutorado submetida à Coordenação dos Cursos de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para obtenção do grau de Doutor em Ciências no Domínio da Engenharia Elétrica .

Área de Concentração: Processamento da Informação

João Marques de Carvalho, PhD.
Orientador

Campina Grande, Paraíba, Brasil

S586d Silva, Vania Cordeiro da
Decomposicao triangular de imagens : uma aplicacao em compressao / Vania Corderiro da Silva. - Campina Grande, 2003.
133 f. : il.

Tese (Doutorado em Engenharia Eletrica) - Universidade Federal de Campina Grande, Centro de Ciencias e Tecnologia.

1. Decomposicao de Imagens 2. Decomposicao Triangular de Imagens 3. Tritree 4. Quadtree 5. Processamento Digital de Imagens 6. Tese I. Carvalho, Joao Marques de II. Universidade Federal de Campina Grande - Campina Grande (PB) III. Título

CDU 004.932:621.3(043)

DECOMPOSIÇÃO TRIANGULAR DE IMAGENS: UMA APLICAÇÃO EM COMPRESSÃO

VÂNIA CORDEIRO DA SILVA

Tese Aprovada em 31.10.2003

PROF. JOÃO MARQUES DE CARVALHO, Ph.D., UFCG
Orientador

PROF. NEUCIMAR JERÔNIMO LEITE, Dr., UNICAMP
Componente da Banca

PROF. ARNALDO DE ALBUQUERQUE ARAÚJO, D.Sc., UFMG
Componente da Banca (Ausência Justificada)

PROF. RONEI MARCOS DE MORAES, Dr., UFPB
Componente da Banca

PROF. BENEDITO GUIMARÃES AGUIAR NETO, Dr., Ing., UFCG
Componente da Banca

PROF. FRANCISCO MARCOS DE ASSIS, Dr. UFCG
Componente da Banaca

PROF. FRANCISCO DE ASSIS FERREIRA TEJO, D.Sc., UFCG
Componente da Banaca

CAMPINA GRANDE – PB
OUTUBRO - 2003

Dedicatória

À minha fonte de inspiração e apoio de todos os momentos, meu esposo Álvaro Vinícius.

Agradecimentos

A Deus por poder concluir mais uma etapa da minha vida.

Ao professor João Marques, por sua valiosa orientação, incentivo e paciência durante todo o trabalho.

Ao meu esposo Álvaro Vinícius por suas contribuições ao trabalho, apoio pessoal, pelas ajudas neste texto, por entender e suportar minha ausência, impedindo-me até, de desistir desta empreitada. Sem ele não teria conseguido.

Aos meus pais, Dinah e Pacele, e minhas irmãs, Vanessa e Valéria, pela ajuda inestimável e abrigo.

Ao meu sobrinho Arthur pelos momentos de descontração que me permitiram lazeres necessários para suportar e concluir tão difícil jornada.

Aos colegas do grupo de Análise e Processamento de Imagens coordenado pelo professor João Marques, em particular Josemar, Suzete e Luciana pela amizada e colaboração ao trabalho.

Um agradecimento especial ao meu amigo Josemar, sem o qual seria impossível o término desde trabalho, sendo pois, meu representante minimizando a distância.

À COPELE e seus funcionários, Ângela, Pedro, Marcos, Eleonora e Rosilda, pela disponibilidade constante.

Ao Departamento de Ciências Exatas e Tecnológicas (DCET) e ao Colegiado do Curso de Computação (COLCIC) da Universidade Estadual de Santa Cruz - UESC, nas figuras dos professores, Herlon, Evandro, Décio e às funcionárias Angélica, Aliomária e Tatiana, pelo apoio profissional, logístico e amizade.

Aos membros da banca da defesa do exame de qualificação pela presença, pelas correções e sugestões ao trabalho.

À Consuelo pelo apoio e incentivo na fase final deste trabalho.

A Doroteia, Jurubeba, Rosesheila, Doque, Junior e Riquinho, por me devolverem o sorriso nos momentos em que este parecia impossível.

Agradeço a todos aqueles que contribuíram direta ou indiretamente para a elaboração deste trabalho, cujos nomes não constam desta minúscula lista.

Por último, a todos os colegas do LAPS (laboratório de processamento de sinais) dos quais sempre recebi demostrações de carinho, simpatia e incentivo.

Resumo

O objetivo deste trabalho é dar uma contribuição à área de compressão de imagens, isto é, ao problema de reduzir a quantidade de bits necessários para representar uma dada imagem, sendo ainda capaz de restaurá-la para sua forma original (com um certo grau de fidelidade). A solução deste problema é importante uma vez que um algoritmo de compressão de imagens eficiênte, pode representar uma economia considerável em termos de espaço de armazenamento de imagens e/ou ocupação do canal para sua transmissão.

Este trabalho apresenta um novo método de decomposição aplicado a imagens, utilizando decomposição *Tritree* (TT). A decomposição TT é similar à decomposição *Quadtree* (QT), a qual tem sido largamente utilizada em algoritmos de compressão de imagens. Contudo, enquanto a QT funciona subdividindo a imagem em regiões quadrangulares, progressivamente menores, a decomposição TT o faz para regiões triangulares. A meta é segmentar a imagem em um conjunto de regiões triangulares homogêneas, onde a diferença dos valores dos *pixels* não exceda um determinado limiar.

Uma árvore é construída para representar a decomposição TT. Cada triângulo será um nó na árvore. O triângulo inicial, chamado de raíz, contém a imagem toda. Os triângulos finais, representando a imagem comprimida propriamente dita, são as folhas.

Os resultados experimentais mostram que o método proposto tem melhor desempenho quando comparado diretamente com a decomposição QT, em termos de qualidade da imagem reconstruída e taxa de compressão. Com isso, é de esperar que os métodos híbridos de compressão de imagens digitais, que fazem uso da QT, tenham seus resultados melhorados, se estes fizessem uso da TT.

Abstract

The objective of this work is to bring a contribution to the field of image compression, i. e., the problem of reducing the amount of bits needed to represent a given imge, while still being able to restore it to its shape (to within a certain degree of fidelity), whenever needed. A good solution to it is important because it can represent a considerable economy in terms of image storage space and/or transmission bandwidth.

This work presents a new method for image compression, using *Tritree* (TT) decompositon. TT decomposition is similar to *Quadtree* (QT) decomposition, which has been largely used for image compression algorithms. However, while QT works by subdividing the image into progressively smaller square regions, TT decomposiston does it for triangular regions. The goal is to segment the image into a set of triangular homogeneous regions, where the pixel values difference does not exceed a given threshold.

A tree is built to represent the decomposition. Each triangle will be a node of the TT. The initial triangle, called root, containing the whole image. The final triangles, representing the compressed image, are the leaves.

Experimental results show the proposed method to perform better than the QT decomposition, in terms of reconstructed images and compression rate.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Caracterização do Problema

1.1 Introdução

"Uma imagem vale mais do que mil palavras". Este axioma expressa a diferença essencial entre nossa habilidade para perceber a informação lingüística e perceber a informação visual. Para uma mesma mensagem, a representação visual sempre tende a ser percebida de uma maneira mais eficiente do que a palavra escrita e a falada. Isto pode ser mais facilmente entendido quando visto de uma perspectiva histórica e evolucionária. Linguagem é mais um desenvolvimento instantâneo no curso da evolução histórica e é manifestada em somente uma espécie, a nossa. Por um outro lado, a visão, de uma forma ou de outra, existe há centenas de milhões de anos, podendo ser encontrada em um sem número de organismos [1].

Ver, para nós, é, sem dúvidas, uma necessidade, pois "ouvir" o mundo, "tocar" o mundo, "cheirar" o mundo não nos transmite tanta informação e de forma tão rápida, quanto "ver" o mundo [2].

1.2 Representação Digital da Informação Visual

Nas últimas três décadas o mundo testemunhou uma grande evolução tecnológica e científica no campo das telecomunicações, computação, e entretenimento (TV/cinema). Além disso, elementos que historicamente pertenciam a cada uma dessas áreas isoladamente, estão agora sendo introduzidos em outras áreas. Por exemplo: computadores estão utilizando vídeo, áudio e capacidade de telecomunicação; vídeo e interatividade estão sendo adicionados ao mundo das telecomunicações; interatividade está chegando à TV e ao cinema. Todas estas áreas, quase sempre fazendo ampla utilização de informação visual, pois, sem dúvidas, esta é a nossa principal fonte de obtenção de informação, por natureza mais rápida e completa [3].

De fato, a troca, armazenamento e manipulação de todos os tipos de informação têm tido uma importância crescente na sociedade moderna [4]. Partindo do ponto em que o sinal analógico tem uma largura de banda muito elevada (obviamente dependendo da qualidade pretendida e da resolução espaço-temporal utilizada), sua utilização é limitada e já saturada. Para pensarmos em desenvolver novos serviços que utilizem imagens, temos que pensar, primeiro, no desenvolvimento de metodologias eficientes de representação da informação visual.

Esta integração de funcionalidades entre áreas distintas, foi possível devido ao desenvolvimento conjugado de tecnologias e metodologias digitais, em termos de processamento, transmissão e armazenamento de dados, com a metodologia de representação de sinais áudio-visuais. A partir da década de 60, novas técnicas e metodologias digitais foram estudadas e desenvolvidas. Começa a conquistar espaço a representação digital de sinais de imagem e vídeo, ou seja, uma imagem ou seqüência de imagens analógicas (vídeo) pode ter uma representação digital. Apesar do nosso mundo ser bem analógico, pois a maior parte das grandezas variam de modo contínuo, é de longa data que conhecemos a possibilidade de representar qualquer sinal a partir de um conjunto de suas amostras, devidamente espaçadas

[5], o que é conhecido como discretização do sinal.

Um sinal analógico pode ser convertido para a forma digital através da combinação de três operações básicas e seqüenciais: amostragem, quantização e codificação [6]. Na operação de amostragem são tomadas apenas amostras do sinal analógico, situadas em instantes de tempo uniformemente espaçados, caracterizando a sua discretização. No processo de quantização, cada valor de amostra tomado é aproximado pelo nível de cinza mais próximo, dentre um conjunto finito de níveis discretos. Na etapa de codificação cada nível de quantização é representado por uma palavra-código, que consiste de uma seqüência de dígitos binários (0's e 1's), qualquer que seja a origem do sinal amostrado.

A simples transformação do sinal analógico para a forma digital, no mínimo, dobra as exigências em termos de largura de banda. Mesmo assim, a digitalização do sinal de vídeo trouxe melhorias de qualidade e possibilitou novas facilidades aos usuários, principalmente em termos de interatividade. A digitalização permite o uso de métodos de análise e processamento digitais, que eliminam a informação redundante, associada à memória ou à correlação presente no sinal de imagem (conforme será explicado na Seção 2.1). Permite, também, a eliminação de informação irrelevante, ou seja, informação à qual a visão humana é não ou pouco sensível, como, por exemplo, baixas freqüências e pequenas variações de tonalidades, dando preferência à luminância sobre a crominância [7]. A idéia é ter-se uma representação digital da informação visual, que permita a recuperação da original, ou de uma aproximação desta, com uma redução significativa de bits, ainda que às custas de uma degradação objetiva, que pode ser ou não detectável subjetivamente [7].

A partir dos anos 70, a digitalização já estava sendo utilizada em dados, voz, música, fac-símile (FAX), fotografia e finalmente televisão. À medida que evoluiu a tecnologia digital - processadores, memórias, transmissão, comutação, compressão – maior foi sendo a pressão para substituir os velhos sistemas analógicos, cuja capacidade de resistência foi diminuindo à medida que as vantagens da digitalização (integração de serviços, tolerância a alguns erros, compressão, processamento) se

tornaram mais evidentes e o seu custo foi diminuindo com a evolução da física do estado sólido [5].

1.3 Princípios da Compressão de Imagens

Por compressão de dados entendemos que é a forma de codificar um certo conjunto de informação, de maneira que o código gerado seja menor que o código de origem, ou fonte. Ou ainda, comprimir uma imagem digital é o processo que permite reduzir a quantidade total de bits necessária para representar a informação nela contida. O processo inverso chama-se descompressão, e, para tanto, é necessário utilizar um descompressor. Como existem vários e diferentes algoritmos de compressão, há que existir para cada compressor, um descompressor compatível, estando muitas vezes os dois presentes no mesmo programa. Por isso mesmo, ao longo deste trabalho, o termo compressor se referirá ao par compressor/descompressor. Caso contrário, uma observação será feita.

Cada algoritmo possui um conjunto de características diferentes, apresentando também ganhos diferentes em determinados tipos de dados. Desta forma, um algoritmo que tem um bom ganho em texto, pode não ter um bom ganho para compressão de imagens, por exemplo.

Técnicas de compressão de imagens tiveram uma grande importância na revolução da representação da informação visual, pois tornaram satisfatoriamente viável a manipulação, armazenamento e transmissão desta informação, principalmente em grande quantidade.

Existem duas formas de compressão dependendo da informação retirada do código fonte da imagem: com perda ou irreversível e sem perda ou reversível. Compressão sem perdas é possível por que, de uma maneira geral, existe uma quantidade significativa de informação redundante presente no código da imagem, conforme pode ser visto na Seção 2.1.2. Esta redundância é proporcional ao montante

de correlação espacial entre os pixels vizinhos de uma imagem [8].

Métodos de compressão com perdas removem informação redundante e informação visual irrelevante, de carater irreversível, mas idealmente não perceptível a visão humana. estes métodos se caracterizam por produzirem imagens semelhantes às originais apenas visualmente, não numericamente [8].

1.4 O Problema

De uma maneira geral, pode-se afirmar que existem duas aplicações básicas para métodos de compressão de imagens: a transmissão e o armazenamento de imagens [9; 10; 11]. Nos dois casos a motivação para se comprimir a imagem é a necessidade de redução de custos, seja diminuição dos requisitos de armazenamento, ou uma economia na ocupação dos meios (canais) de transmissão, ou simplesmente para agilizar a transmissão de dados. Isto é necessário, pois este tipo de informação tem se tornado cada vez mais utilizado. Entretanto, o seu volume de uso não é acompanhado na mesma proporção pelo crescimento da capacidade dos dispositivos de armazenamento e tecnologias de telecomunicação, sendo necessário, cada vez mais, o uso de técnicas de compressão.

Para exemplificar a necessidade da compressão para transmissão, Wang [11] afirma que para se transmitir uma imagem digital, de tamanho 256x256 com 8 bits por pixel num canal a uma taxa de 1200 bit/seg, demora-se cerca de 7,28 minutos.

Como exemplo da necessidade de compressão para armazenamento, podemos citar o que acontece com a Enciclopédia Britânica[1]: para representá-la na forma digital são necessários mais do que 25 gigabytes ($25x10^9$ bytes) de dados [6].

Em aplicações para transmissão de imagens, as técnicas para compressão operam em tempo-real e variam de acordo com a capacidade e a complexidade do *hard-*

[1] A Enciclopédia Britânica contêm cerca de 25 mil páginas. Uma única página digitalizada, por exemplo, a 300 pontos por polegada (dpi - dots per inch) e quantizada com dois níveis de cinza, ou seja, preto e branco, gera aproximadamente 8.000.000 bits (1.000.000 bytes) de dados.

ware em que vão operar. Devido ao seu próprio objetivo, estes algoritmos são mais simples. As aplicações voltadas para armazenamento são mais complexas pois têm a possibilidade de executar um pré-processamento e/ou um pós-processamento em *off-line* nas imagens. Contudo, os algoritmos de compressão devem ser rápidos e eficientes, para minimizar o tempo de execução do processo de compressão.

Técnicas de compressão de dados são ainda utilizadas para o desenvolvimento de algoritmos mais rápidos, nos quais o número de operações requeridas para implementação de um determinado método é reduzido pois este passa a trabalhar diretamente com as informações comprimidas [9].

Reduzir a quantidade final de bits para representar uma dada informação, tornou-se a questão central de um campo de investigação que adquiriu importância crescente desde o início dos anos 70, largamente conhecido por compressão ou codificação de imagens e vídeo [12]. A compressão de imagens desempenha um importante papel em diversas aplicações, dentre as quais podem ser citadas: videoconferência, sensoriamento remoto (por exemplo, imagens de satélite para previsão do tempo), armazenamento de imagens médicas e documentos em geral, transmissão de fac-simile (FAX), multimídia, redes digitais de serviços integrados (ISDN, Integrated Services Digital Networks), televisão de alta definição (HDTV, High Definition TV), armazenamento de impressões digitais e transmissão de TV e imagens obtidas por satélite. No campo militar podemos citar os veículos remotamente pilotados.

1.5 Objetivo do Trabalho e Organização do Texto

Um número incontável e crescente de aplicações depende da manipulação, armazenamento e transmissão de imagens (por exemplo: internet ou albuns digitais), sejam elas em preto-e-branco, em tons de cinza ou coloridas. Para que isso seja viável, é desejável que estas imagens estejam eficientemente comprimidas. Eis a

origem dos constantes esforços despendidos em pesquisas nesta área.

Este trabalho aborda o problema de compressão de imagem, ou seja, o problema de reduzir a quantidade de bits necessários para representar uma determinada imagem, garantindo que será possível restabelecê-la em sua forma original (dentro de um certo grau de fidelidade) sempre que preciso. Este problema é importante, porque uma solução razoável para isto (ou seja, um algoritmo de compressão de imagem eficiente) pode representar uma economia considerável em termos de espaço de armazenamento da imagem e/ou banda de passagem de transmissão.

Um tipo bastante utilizado de técnicas de compressão de imagens são aquelas baseadas em segmentação ou decomposição da imagem original em regiões homogêneas, que podem ser representadas por um número reduzido de bits. Dentre as técnicas de decomposição já desenvlvidas, destaca-se aquela conhecida como *"Quadtree"*, que tem sido utilizada em vários métodos híbridos de ompressão de imagens descritos na literatura (Seção 3.1.4).

O objetivo deste trabalho é o desenvolvimento e implementação de uma nova metodologia de decomposição de imagens reais bidimensionais, em tons de cinza, através da utilização de estruturas do tipo Tritree. Como base, foi utilizado o trabalho de Wille [13] para a divisão sucessiva da imagem em regiões triangulares. Chega-se, assim, a um conjunto de regiões que possuem os seus valores de pixels aproximadamente iguais, ditas regiões homogêneas. O estudo e desenvolvimento de métodos para a reconstrução das imagens aqui comprimidas, e de formas de armazenamento das informações necessárias para a descompressão, também foram pesquisados. Foram investigados também quais os possíveis critérios para decidir se um triângulo deve ser novamente subdividido ou não.

O método de Wille [13] foi originalmente desenvolvido para a geração de malhas de elementos finitos em duas ou três dimensões, não estruturadas, para serem aplicadas na solução computacional de problemas aero e hidrodinâmicos, na forma de sistemas de equações diferenciais. Seu método, de uma maneira resumida, enquadra a imagem original em um triângulo equilátero, e divide este elemento inicial

em 4 novos triângulos equiláteros menores.

Neste trabalho é utilizado um critério para pesquisar se cada novo elemento deve ser novamente subdividido ou não (não foram utilizados critérios de homogeneidade no trabalho original de Wille). Para o caso de imagens tridimensionais o procedimento é análogo, sendo a imagem dividida em tetraedros equiláteros sucessivamente menores. Na estrutura gerada, os triângulos ou os tetraedros formam uma árvore de pesquisa chamada de Tritree (cujo nome foi mantido). Cada elemento, ou triângulo, forma um nodo da árvore. O elemento inicial, que contém a imagem completa, é a raiz da árvore e suas folhas (nós terminais) são os elementos que guardam a informação relevante da imagem comprimida, ou seja, a informação necessária para a descompressão.

Resumindo, a meta essencial desta decomposição é descobrir e classificar regiões triangulares da imagem, de acordo com seu grau de homogeneidade (valores de pixels aproximadamente iguais). Estruturas do tipo Tritree são parecidas com estruturas Quadtree, que são amplamente utilizadas por muitos algoritmos de compressão e processamento de imagens. Existem diferenças entre as duas estruturas e em como gerenciá-las, mas a principal diferença deve-se ao fato de que na primeira estrutura, a Tritree, as regiões da imagem a classificar são triangulares, enquanto que na segunda estrutura, a Quadtree, as regiões são quadrangulares. Ambas as estruturas serão explicadas com detalhes, a posteriori.

Este trabalho é descrito da seguinte forma: os Capítulos 2 e 3 fazem uma revisão bibliográfica sobre o tema compressão de imagens, sendo que no primeiro se discute a classificação dos algoritmos de compressão, enquanto que no segundo se consideram as diferentes abordagens sobre os mesmos algoritmos. No Capítulo 4 o método proposto é apresentado. Os resultados obtidos podem ser vistos no Capítulo 5. Este texto se encerra com o Capítulo 6, que apresenta as contribuições esperadas e as conclusões do trabalho.

Capítulo 2

Revisão Bibliográfica 1: Classificação e Medidas Avaliativas dos Algoritmos de Compressão de Imagens Digitais

Este capítulo apresenta a primeira parte da revisão bibliográfica feita como parte desta tese, para estabelecer uma base conceitual apropriada aos entendimentos da compressão de imagens digitais.

2.1 Classificação dos Algoritmos de Compressão

Os métodos de compressão de imagens desenvolvidos até o presente, podem ser analisados segundo dois enfoques, que correspondem a duas formas de classificação dos mesmos. O primeiro enfoque está relacionado ao fato do método produzir, ou não perda de informação contida na imagem, ou seja, o método pode ser classificado como com ou sem perdas, sendo estas classes mutuamente excludentes.

A segunda categoria leva em consideração quais os tipos de informação, dentre três tipos identificados na literatura que o método elimina na imagem, para que

seja feita a compressão, os quais podem ocorrer isoladamente ou combinados. São eles: redundância na codificação, redundância interpixel e informação psicovisual irrelevante. Historicamente, o problema da compressão sem perdas foi abordado primeiro [14]. Todas as classificações serão explicadas nas próximas seções.

2.1.1 Classificação Segundo a Perda de Informação

Segundo este enfoque, os métodos de compressão de imagens dividem-se em dois grandes grupos: os sem perdas de informação, ou reversíveis, e os métodos com perdas, ou irreversíveis [8]. Algoritmos sem perdas são caracterizados por eliminarem somente as informações redundantes, possibilitando, assim, a reconstrução total da imagem original após o processo de descompressão. Os algoritmos de compressão com perdas são caracterizados por eliminarem, além das informações redundantes, as informações que julgarem irrelevantes, possibilitando somente uma reconstrução aproximada da imagem original após a descompressão.

Quanto à taxa de compressão alcançada por estas duas categorias de algoritmos, de uma maneira geral, pode-se afirmar que os algoritmos de compressão sem perdas só atingem modestas taxas de compressão, quando comparados com os algoritmos com perdas. Os algoritmos de compressão com perdas, conhecidos como algoritmos "estado-da-arte", normalmente fornecem taxas de compressão de cerca de 30:1, mantendo uma qualidade razoável da reconstrução, enquanto os algoritmos sem perdas atingem cerca de 3:1 ou menos. Consequentemente, pode-se afirmar que alta compressão é obtida à custa de degradação na imagem reconstruída.

Em geral, a qualidade da imagem comprimida declina à medida que a taxa de compressão aumenta [15]. O grau de fidelidade da imagem descomprimida, em relação à imagem original, depende fortemente da aplicação/serviço a que se destina a imagem comprimida e o grau de relevância da informação descartada, como também dos requisitos de *software* e *hardware* disponíveis.

Os algoritmos de compressão sem perdas garantem que a imagem reconstruída

e a imagem original serão absolutamente idênticas. Isto é um importante requisito para alguns domínios de aplicações, como é o caso da medicina. Imagens médicas de altíssima qualidade mostram informações funcionais e estruturais detalhadas dos órgãos do corpo humano, possibilitando um diagnóstico preciso e confiável. Nos Estados Unidos, as clínicas e hospitais que insistem em utilizar algoritmos de compressão com perdas para armazenar suas imagens médicas, têm enfrentado processos na justiça, seja por erros de diagnósticos ou pelo FDA[1], questionando-os sobre a falta de testes e regulamentação sobre a utilização de algoritmos de compressão com perdas para esta finalidade [15]. A falta de nitidez nestas imagens pode gerar diagnósticos errôneos.

Outro uso dos algoritmos de compressão de imagens sem perdas é nas imagens de satélite, onde eventuais perdas de informação podem acarretar uma identificação errada de um foco de incêndio, por exemplo. Técnicas de compressão com perdas não devem também ser utilizadas para compressão de qualquer outro tipo de dados quando um único bit alterado não é aceitável, por exemplo, arquivos documentos e programas executáveis.

A maioria dos métodos de compressão sem perdas pode ser facilmente modificada para introduzir perdas, por exemplo, aumentando-se a quantização. A recíproca, neste caso, não é verdadeira, ou seja, os métodos de compressão com perdas, em geral, não podem ser modificados para que seja eliminada a perda de informação.

2.1.2 Classificação Segundo o Tipo de Redundância Eliminada

São identificados, na literatura, três tipos de redundâncias que podem ser eliminadas para que seja feita a compressão de imagens, os quais podem ocorrer isolados ou combinados. São eles: redundância na codificação, redundância interpixel e informação visual irelevante. Estes três tipos encontram-se resumidos nas seções que

[1]United States Food and Drug Administration (FDA).

se seguem.

Redundância na Codificação

Normalmente, os valores de nível de cinza de uma imagem são representados por um número fixo de bits. Por exemplo, se para uma determinada imagem só for possível se reservar 8 bits para cada valor de nível de cinza, isto implica que a imagem poderá no máximo assumir 256 níveis de cinza.

Entretanto, analisando-se histogramas de imagens quaisquer, não será difícil notar que existem valores de nível de cinza que aparecem mais na imagem do que outros. Aliás, existem valores que, apesar de pertencerem à gama de valores de nível de cinza que a imagem pode assumir, não aparecem na imagem.

Tabela 2.1: Exemplo de uma codificação de tamanho variável

Valores	Probabilidade	Código 1	Código 2	Tamanho
0	0.19	000	11	2
1	0.25	001	01	2
2	0.21	010	10	2
3	0.16	011	001	3
4	0.08	100	0001	4
5	0.06	101	00001	5
6	0.03	110	000001	6
7	0.02	111	000000	6

Com base nesta informação, foram desenvolvidas técnicas que usam tamanho de código variável para representar cada valor de nível de cinza das imagens comprimidas. Obviamente, aos valores que com maior freqüência aparecem na imagem é atribuído um código de menor tamanho, e vice-versa.

Tomemos o exemplo de uma imagem qualquer com apenas 8 níveis de cinza, cuja distribuição é mostrada na coluna 2 da Tabela 2.1. A distribuição é calculada pela freqüência relativa: o número de vezes que um determinado valor de nível de cinza aparece na imagem dividido pelo número total de *pixels* na imagem. A primeira maneira que se poderia pensar de codificar esta imagem seria com um código fixo de 3 bits, como aparece na coluna 3. Na coluna 4 aparece uma forma alternativa de codificação, desta vez com tamanho variável, baseado na estatística da fonte. Utilizando-se a forma alternativa para comprimir a imagem do exemplo, o número médio B de bits requeridos para codificar a imagem é reduzido de 3 para 2.7, segundo o cálculo abaixo empregando a freqüência relativa:

$$B = 2 \times 0.19 + 2 \times 0.25 + 2 \times 0.21 + 3 \times 0.16 + 4 \times 0.08$$
$$+5 \times 0.06 + 6 \times 0.03 + 6 \times 0.02 = 2.7$$

Como este tipo de eliminação de redundância só trabalha na forma como a imagem é codificada, não há perdas de informação, sendo totalmente reversível (seção 2.2). Exemplos de algoritmos que eliminam este tipo de redundância são os já consagrados algoritmos de codificação de Huffman e o algoritmo LZW.

Redundância Interpixel

Redundância interpixel também é conhecida como redundância espacial, redundância geométrica ou redundância interframe. Este tipo de redundância se caracteriza pela correlação existente entre pixels próximos em uma imagem, ou seja, o valor de um determinado pixel pode ser razoavelmente previsto, considerando-se os valores de nível de cinza de um conjunto de seus vizinhos. Freqüentemente, a diferença entre eles é relativamente pequena.

Para reduzir a redundância interpixel em uma imagem, a matriz bidimensional de *pixels* normalmente utilizada para representar a imagem, deve ser transformada para um formato mais eficiente (mas normalmente não visual). Por exemplo, as diferenças entre pixels adjacentes podem ser utilizadas para representar a imagem.

Uma operação como esta é totalmente reversível, pois os elementos da imagem original podem ser reconstruídos através da soma do valor do pixel presente com o do seu vizinho. Considerando-se os valores abaixo como uma linha qualquer de uma imagem fictícia,

$$149\ 148\ 149\ 147\ 140\ 136\ 128\ 128$$

a informação necessária para a reconstrução desta linha seria:

$$149\ -1\ +1\ -2\ -7\ -4\ -8$$

O armazenamento desta segunda linha de informação, obviamente, pode ser feito com muito menos bits. Para se reduzir ainda mais o número de bits necessários para a reconstrução da imagem, pode ser aplicado um algoritmo que elimina a redundância de codificação (seção anterior).

A eliminação da redundância interpixel normalmente produz elevadas taxas de compressão em imagens binárias, ou seja, imagens em preto e branco, pois ao invés de armazenarmos as diferenças entre os *pixels*, pode-se armazenar a quantidade de 0's e 1's que formam pequenas seqüências. Por exemplo, sendo a seqüência de números abaixo uma linha de uma imagem binária:

$$0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0$$

a informação necessária para sua reconstrução seria:

$$(0,6)\ (1,5)\ (0,5)\ (1,3)\ (0,7)$$

O que também, para ser armazenado, necessita de muito menos espaço, principalmente considerando-se uma imagem inteira, não só uma linha. Podemos citar como um representante dos algoritmos que fazem uso desta abordagem o algoritmo DPCM (Seção 3.1.1).

Informação Visual Irrelevante

Muito da contribuição visual de um único pixel de uma imagem é irrelevante, pois ele pode ser previsto com base nos seus vizinhos. Além disso, o olho humano não responde com a mesma sensibilidade a toda informação visual. Certas informações simplesmente têm menos importância relativa do que outras, em um sistema de processamento visual normal. Este tipo de informação pode ser considerado irrelevante e pode ser eliminada sem empobrecimento significativo da qualidade da percepção da imagem. Suponhamos a linha abaixo, como sendo um fragmento de linha de uma imagem fictícia de 256 níveis de cinza:

203 204 203 195 190 187 178 172 172 171 172

Qual observador humano perceberia as mudanças sutis de valores de nível de cinza de 203 para 204, por exemplo? Se essa informação (a diferença) fosse perdida, muito provavelmente a imagem ainda seria passível de reconhecimento, ou seja, não seria afetada significativamente a qualidade da imagem reconstruída.

Eliminar a informação visual irrelevante, é fundamentalmente diferente das outras informações eliminadas anteriormente, pois resulta em perda de informação de caráter irreversível. Desde que a eliminação desta redundância corresponde a perdas quantitativas de informação, este tipo de redundância também costuma ser referenciado como quantização [6]. Esta terminologia se refere à ação de mapear a grade de valores dos pixels de entrada em um limitado número de valores de saída. Esta é uma operação irreversível, ou seja, alguma informação visual é perdida.

Que a eliminação deste tipo de informação acontece não chega a ser uma surpresa, pois a percepção humana da informação contida em uma imagem, normalmente não envolve análise quantitativa de cada pixel da imagem. Em geral, um observador procura por características regionais e globais, como bordas e textura e mentalmente as combina para reconhecer o objeto [6; 16].

2.1.3 Conclusão

Nesta seção foram apresentadas as duas grandes classificações dos algoritmo de compressão. A primeira considera a perda ou não de informação resultante da compressão. A segunda classificação se baseia no tipo de informação que os métodos de compressão podem eliminar, que são três: redundância na codificação, redundância interpixel e informação psicovisual irrelevante.

As melhores taxas de compressão são atingidas pelos algoritmos com perdas, embora com alguma perda de qualidade da imagem reconstruída, que pode ser perceptível ou não. Os algoritmos que eliminam informação psicovisual irrelevante, ou seja, os que fazem quantização no momento da compressão, também alcançam melhores taxas de compressão do que os métodos que eliminam só os outros tipos de redundância. Por isso, a grande maioria dos métodos de compressão desenvolvidos que podem ser encontrados na literatura, se preocupam em eliminar a informação visual (fazem quantização) e, consequentemente, introduzem perdas.

2.2 Medidas de Avaliação dos Algoritmos de Compressão de Imagens

Para que o desempenho dos algoritmos de compressão de imagem possa ser calculado e analisado e, principalmente, para que os algoritmos desenvolvidos possam ser comparados entre si, existe uma série de medidas avaliativas padronizadas, utilizadas pela literatura que trata do assunto. De uma maneira geral, essas medidas objetivam avaliar a eficiência da compressão, através da sua taxa de compressão e/ou taxa de bits por it pixel, bem como a qualidade da imagem reconstruída após a compressão da imagem digital, complexidade do algoritmo e a sua medida de entropia.

Os algoritmos de compressão de imagens com perdas, produzem apenas aproximações da imagem original, ou seja, existem diferenças (também chamadas de distorções ou artefatos) entre a imagem original e a imagem reconstruída. Para que

os desempenhos destes algoritmos possam ser avaliados e até mesmo comparados, estas diferenças devem ser medidas. Uma outra medida avaliativa é a qualidade da imagem reconstruída em relação à original, sendo que esta é utilizada apenas para os métodos de compressão com perdas.

As medidas de eficiência de compressão, de entropia e de complexidade são utilizadas para ambos os tipos de métodos, com e sem perdas, sendo que o desempenho dos métodos de compressão sem perdas é medido apenas por estes parâmetros.

2.2.1 Medidas Avaliativas de Distorção e de Qualidade da Reconstrução

A quantidade de distorção presente na imagem reconstruída, influencia na qualidade de sua percepção. Esta qualidade pode ser avaliada através de medidas objetivas e subjetivas. As medidas objetivas podem ser executadas computacionalmente, através de comparações matemáticas diretas entre a imagem original e a reconstruída. As medidas subjetivas também baseiam-se em comparações diretas entre a imagem original e a reconstruída, só que realizadas por um grupo de pessoas que subjetivamente classificam a qualidade da imagem comprimida segundo uma escala pré-determinada [17].

Todas as medidas objetivas são discretas, bidimensionais, que procuram explorar as diferenças entre os pixels da imagem original e seus correspondentes na imagem reconstruída. Em outras palavras, as medidas de Distorção e qualidade da reconstrução procuram mensurar a similaridade entre duas imagens digitais, explorando as diferenças na distribuição estatística dos valores dos pixels [17].

Medidas Subjetivas

Avaliações subjetivas são realizadas através de testes de visualização e são baseadas nas opiniões individuais de cada pessoa participante do teste. As medidas subjetivas são de difícil realização, pois demandam muito tempo, um grande número de avaliadores (entre eles leigos, especialistas e possíveis usuários do sistema de compressão), deslocamento dos avaliadores até o laboratório para a visualização das imagens, disponibilidade destes laboratórios e um razoável número de imagens já processadas antes da realização do teste.

A medida de qualidade subjetiva mais comumente utilizada para avaliação dos algoritmos de compressão de imagens com perdas é o teste MOS (Mean Opinion Score[2])[12]. Nesta medida, os avaliadores classificam a imagem analisada em uma das cinco categorias (ou notas), como apresentado na primeira coluna da Tabela 2.2. Cada nota está associada a um adjetivo qualificativo da imagem e a uma breve descrição da imagem reconstruída (segunda e terceira colunas da Tabela 2.2, respectivamente), este teste é conhecido como escala de Likert.

Tabela 2.2: Notas do teste MOS

Notas	Qualidade Subjetiva	Nível de Distorção
5	Excelente	Imperceptível
4	Boa	Perceptível mas não incômoda
3	Razoável	Perceptível e um pouco incomodante
2	Pobre	Incomoda mas ainda percebe-se a imagem
1	Insatisfatória	Incomoda e não percebe-se a imagem

Sendo N o número total de avaliadores e R_n a resposta do avaliador n, o teste MOS é computado pela média aritmética simples das respostas assinaladas pelos avaliadores, como se segue:

$$MOS = \frac{1}{N}\sum_{n=1}^{N} R_n$$

[2]Escore médio de opinião

Uma outra medida subjetiva de qualidade para avaliar imagens reconstruídas é o teste de preferência. Neste, a comparação é feita com relação a uma imagem de referência. Este teste utiliza uma escala de graduação com valores que variam de um a cinco, onde cada valor corresponde a um adjetivo do processo de comparação [12]:

- 5 – A imagem sob teste tem qualidade muito superior à de referência;
- 4 – A imagem sob teste tem qualidade um pouco superior à de referência;
- 3 – A imagem sob teste tem a mesma qualidade da de referência;
- 2 – A imagem sob teste tem qualidade um pouco inferior à de referência;
- 1 – A imagem sob teste tem qualidade muito inferior à de referência.

Medidas Objetivas

As medidas de qualidade objetivas são mais utilizadas que as subjetivas para avaliação dos algoritmos de compressão de imagens digitais, devido a vários fatores como os apresentados por Madeiro et. alli. [18], que são aqui reproduzidos:

- Permitem avaliação rápida;
- São mais facilmente implementadas e muito menos dispendiosas em tempo, relativamente às medidas subjetivas;
- Podem ser repetidas inúmeras vezes durante a fase de desenvolvimento de um algoritmo;
- Podem auxiliar mais facilmente o ajuste de parâmetros de algoritmos e até mesmo servir como um meio de detecção de diferenças sutis no desempenho dos algoritmos, possivelmente não detectáveis por meio de avaliação subjetiva.

Na Tabela 2.3 estão listadas as 11 principais medidas objetivas para avaliação de desempenho de algoritmos de compressão com perdas, segundo Eskicioglu e Fischer [19]. Os valores dos pixels da imagem original são representados na tabela por $I(i,j)$ e os da imagem reconstruída são representados por $R(i,j)$. Sendo L o número de linhas da imagem e C o número de colunas, o valor de i representa a i-ésima linha ($1 \leq i \leq L$) e j representa a j-ésima coluna ($1 \leq j \leq C$). No cálculo da medida PSNR, a variável A representa o valor máximo de nível de cinza que os pixels da imagem original podem alcançar.

As medidas objetivas e as subjetivas devem apresentar uma correlação forte entre si, ou seja, as medidas objetivas de qualidade devem apresentar um conteúdo subjetivo e vice-versa: pequenas e grandes variações de uma medida devem implicar em pequenas e grandes variações na outra. Mas isso nem sempre é verdadeiro, variando em função da imagem utilizada nos testes e dos algoritmos testados.

Madeiro et. alli. faz em [18] uma análise da correlação entre as medidas objetivas que estão listadas na Tabela 2.3 e as medidas subjetivas, representadas pelo teste MOS (seção 2.2.1). Para tanto são utilizados os resultados obtidos para imagens em nível de cinza por dois algoritmos de compressão de imagens por quantização vetorial, o LBG e SSN-TV.

Os resultados obtidos naquele trabalho indicam que algumas medidas realmente têm problemas quanto à fidelidade. Por exemplo, as medidas AD, SC, NK, CQ e MD não demonstraram ser muito confiáveis para avaliação de técnicas, pois ou apresentam baixa correlação com as respostas dos avaliadores, ou o sinal dos coeficientes de correlação muda. As outras medidas são mais confiáveis pois elas alcançam altos níveis de correlação com os resultados da medida subjetiva obtida pelo teste MOS.

O trabalho de Madeiro et. alli. [18] conclui ainda que deve-se ter muito cuidado com algumas medidas, por exemplo a NK, pois elas podem levar a conclusões incorretas se não forem propriamente interpretadas, quando é feita a comparação da

Tabela 2.3: Medidas de Qualidade Objetiva

Medidas	Fórmula
Diferença Média (Average Difference)	$AD = \frac{\sum_{i=1}^{L}\sum_{j=1}^{C}[I(i,j)-R(i,j)]}{LC}$
Conteúdo Estrutural (Structural Content)	$SC = \frac{\sum_{i=1}^{L}\sum_{j=1}^{C}[I(i,j)]^2}{\sum_{i=1}^{L}\sum_{j=1}^{C}[R(i,j)]^2}$
Correlação Cruzada Normalizada (Normalized Cross–Correlation)	$NK = \frac{\sum_{i=1}^{L}\sum_{j=1}^{C}I(i,j)R(i,j)}{\sum_{i=1}^{L}\sum_{j=1}^{C}[I(i,j)]^2}$
Qualidade de Correlação (Correlation Quality)	$CQ = \frac{\sum_{i=1}^{L}\sum_{j=1}^{C}I(i,j)R(i,j)}{\sum_{i=1}^{L}\sum_{j=1}^{C}I(i,j)}$
Diferença Máxima (Maximum Difference)	$MD = Max\|I(i,j) - R(i,j)\|$
Fidelidade da Imagem (Image Fidelity)	$IF = 1 - \frac{[\sum_{i=1}^{L}\sum_{j=1}^{C}[I(i,j)-R(i,j)]^2}{\sum_{i=1}^{L}\sum_{j=1}^{C}[I(i,j)]^2]}$
Erro Médio Quadrático (Mean Square Error)	$MSE = \frac{\sum_{i=1}^{L}\sum_{j=1}^{C}[I(i,j)-R(i,j)]^2}{LC}$
Relação Sinal-Ruido de pico (Peak Signal-to-Noise Ratio)	$PSNR = 10\log_{10}[\frac{A^2}{MSE}]$
Norma L_p	$l_p = \frac{\sum_{i=1}^{L}\sum_{j=1}^{C}\|I(i,j)-R(i,j)\|^p}{LC}, p = 1,2,3$
Erro Absoluto Normalizdo (Normalized Absolute Error)	$NAE = \frac{\sum_{i=1}^{L}\sum_{j=1}^{C}\|I(i,j)-R(i,j)\|}{\sum_{i=1}^{L}\sum_{j=1}^{C}\|I(i,j)\|}$
Erro Médio Quadrático Normalizado (Normalized Mean Square Error)	$NMSE = \frac{\sum_{i=1}^{L}\sum_{j=1}^{C}[OI(i,j)-OR(i,j)^2]^2}{\sum_{i=1}^{L}\sum_{j=1}^{C}[OI(i,j)]}$

performance de diferentes algoritmos de compressão de imagens.

2.2.2 Medidas de Eficiência da Compressão

A eficiência da compressão dos algoritmos com ou sem perdas pode ser medida em termos de taxa de compressão ou em taxa de bits por pixel (bpp) [1; 20; 21]. A taxa de compressão pode ser entendida como a taxa que expressa a diferença de "tamanho" entre a imagem original e a imagem comprimida. Este tamanho da imagem pode ter algumas interpretações dependendo do propósito e da implementação do algoritmo, mas em geral é medido em termos de número de pixels ou de bytes.

A taxa de bits por pixel TB é calculada como a média do número de bits requeridos por pixel para codificar a imagem comprimida [1], ou ainda [21]:

$$\text{TB} = \frac{\text{número de bits necessários para codificar a imagem}}{\text{número de pixels na imagem comprimida}}$$

A taxa de compressão *TC* é a relação entre o tamanho da imagem original e o tamanho do arquivo compactado, determinado por:

$$\text{TC} = \frac{\text{tamanho do arquivo original}}{\text{tamanho do arquivo compactado}}$$

Tomemos como exemplo uma imagem de dimensões 256×256 (linha $\times$ coluna), codificada originalmente com 8 bits por pixel, ou um byte por pixel. Esta imagem requer $256 \times 256 = 65536$ bytes ou pixels, ou $65536 \times 8 = 524288$ bits, quando armazenada da forma original (não comprimida). Supondo que após a execução de algum algoritmo de compressão seja necessário o armazenamento de apenas 32768 pixels, então a taxa de compressão é calculada como $65536/32768 = 2.0$, dizemos que a taxa de compressão foi de $2:1$ (dois para um). Desde que a imagem reconstruída ainda tenha as mesmas dimensões da imagem original, $256 \times 256 = 65536$ pixels, o

arquivo da imagem comprimida necessita de $32768 \times 8/65536 = 4$ bits por pixel, na média. Então a taxa de bits por pixel é de 4.

A taxa de compressão e a taxa de bits por pixel são inversamente proporcionais. Sendo b o número de bits por pixel necessário para codificar a imagem original, TC a taxa de compressão e TB a taxa de bits por pixel, a equação abaixo pode ser calculada:

$$b = TC \times TB \tag{2.1}$$

2.2.3 Medida de Complexidade do Algoritmo

A complexidade de um algoritmo para qualquer finalidade é o esforço computacional requerido para se realizar todas as etapas do algoritmo, no nosso caso, a compressão e descompressão das imagens [1]. Os processos de compressão e descompressão podem ser medidos pelo número de operações aritméticas *(flops)* por *pixel* necessárias para que seja totalmente executado.

A complexidade está intimamente associada a velocidade de execução do algoritmo. A velocidade de um algoritmo de qualquer propósito é função da sua complexidade e de sua implementação. Esta medida é um fator muito importante, e até mesmo decisivo, quando estão envolvidas aplicações de compressão e descompressão de imagens em tempo-real, onde a velocidade é uma característica crucial.

2.2.4 Entropia

Shannon introduziu o conceito de entropia como sendo a medida da média da informação contida em uma fonte [1; 9]. Supondo uma imagem I cuja escala de

valores de nível de cinza varia de 0 até 255, ou seja, uma imagem que tem até A = 256 valores de nível de cinza. Sendo p_i a probabilidade de um pixel de I tenha o valor i. O conteúdo da informação da imagem I – entropia – é dado por:

$$H(I) = -\sum_{i=1}^{A} p_i log_2 p_i \tag{2.2}$$

A unidade de entropia é bits por *pixel*.

2.2.5 Conclusão

Apesar de termos presente na literatura várias medidas avaliativas de algoritmos de compressão, poucas são efetivamente utilizadas. As medidas mais comumente encontradas nos artigos que tratam do assunto são as medidas objetivas PSNR e MSE. As medidas subjetivas são mais empregadas em métodos de objetivo específico, ou quando os futuros usuários do método tem fácil acesso àqueles que o desenvolveram.

Capítulo 3

Revisão Bibliográfica 2: Abordagens mais Comuns e Padronização dos Métodos de Compressão

Neste Capítulo encontra-se a segunda parte da revisão bibliográfica feita como parte desta tese, para que a orientanda tivesse um conhecimento mais aprofundado da área de pesquisa: Compressão de Imagens.

3.1 As Abordagens mais Comuns dos Métodos de Compressão

O estudo de métodos de compressão de imagens tem sido uma área ativa de pesquisa desde o surgimento das técnicas de processamento digital de sinais. Desde que imagens podem ser entendidas como um sinal 2D (bidimensional) com as variáveis independentes sendo as coordenadas espaciais 2D (linha x coluna), muitas técnicas de compressão digital para sinais unidimensionais podem ser estendidas para imagens com relativa facilidade [1].

Muitas das abordagens encontradas na literatura que tratam de compressão de imagens [6; 15; 9; 1; 22; 23; 24; 16], podem ser classificadas em uma das categorias maiores apresentadas abaixo. Alternativamente, uma combinação de algumas dessas técnicas podem ser utilizadas por um mesmo método, o que é chamado de abordagem híbrida. Entre os autores da área não existe um consenso quanto à classificação de alguns métodos, por apresentarem características de mais de uma abordagem. É o caso do método das transformadas *wavelet* que pode ser encontrado na literatura tanto classificado como um método de codificação por transformada, quanto por codificação multirresolucional.

Portanto, as classes mais gerais são:

- Codificação Preditiva;
- Codificação por Transformada;
- Quantização Vetorial;
- Codificação Multirresolucional/Multiescala;
- Abordagem Híbrida.

As duas primeiras abordagens (codificação preditiva e por transformada) são as mais tradicionais, pois foram as primeiras desenvolvidas, sendo também chamadas de "Técnicas de Compressão de Primeira Geração [6; 9; 22; 16]". Uma característica destas técnicas é o fato de não analizarem a informação contida na própria imagem como um todo para realizar a compressão, elas são baseadas na informação contida nos *pixels*, ou seja, informação local.

As demais técnicas, também chamadas de "Técnicas de Compressão de Imagens de Segunda Geração", caracterizam-se por identificar as características da imagem como um todo e utilizar estas características para alcançar a compressão. Todas as técnicas da segunda geração incorporam propriedades do sistema visual humano

(HVS[1]) na sua estratégia de codificação para atingir altas taxas de compressão, enquanto ainda mantêm uma qualidade aceitável na imagem reconstruída [16].

Muito esforço foi feito para identificar o que o observador humano considera visualmente mais importante para a caracterização de uma imagem. Uma conclusão geral foi que o nosso sistema visual normalmente descreve a imagem em termos de seus contornos e suas texturas [22]. Com base nesta informação, a maioria dos métodos desta geração procura preservar a informação de borda, separando-a da informação de textura, podendo ainda codificá-las separadamente. Em outras palavras, esses métodos identificam na imagem as regiões de significância e insignificância visual, podendo aplicar técnicas de codificação apropriada para cada área.

Constantemente são desenvolvidos novos métodos de compressão de imagens com ou sem perdas, a grande maioria se enquadra em uma das grandes abordagens listadas anteriormente. Pode ser que venha a ser desenvolvido um novo método de compressão que não se enquadre em nenhuma dessas e, provavelmente num futuro não muito distante, com a propagação desta "nova" abordagem utilizada, e consequente desenvolvimento de novos métodos que façam uso desta abordagem, pode até ser criada uma nova classificação.

Contudo, estes são conceitos muito novos e por isso mesmo ainda não consolidados. Com isso é comum encontrarmos interpretações diferentes para os mesmos conceitos. Por exemplo, existem autores que classificam as transformadas *wavelet* como um método de primeira geração, ou seja, codificação por transformada [25; 26], outros já as classificam como um método de segunda geração, como codificação multirresolucional [15; 16]. Existem também discrepâncias quanto ao nome dos métodos. Há métodos com conceitos muito parecidos e nomes diferentes, como exemplo podemos citar a codificação multirresolucional [16; 20; 21; 27] que também é conhecida como multiescala [16], codificação por sub-bandas ou piramidal [16; 11; 21; 27; 28].

Um resumo sobre cada uma das abordagens citadas será dado nas seções que se

[1]Também conhecido como propriedades HVS do inglês humam visual system

seguem, bem como exemplos dos métodos desenvolvidos mais utilizados que usam essas abordagens.

3.1.1 Codificação Preditiva

Esta abordagem tem como principal característica a simplicidade. Normalmente, imagens de cenas reais exibem um alto grau de correlação entre seus pixels vizinhos, o que é conhecido na literatura como redundância interpixel. Este alto grau de correlação implica em um alto grau de redundância em cada linha de informação [1]. Se esta redundância for removida pelo descorrelacionamento dos dados, isto por si só já representaria uma compressão de dados. Os métodos que fazem uso desta abordagem exploram justamente a correlação dos dados da imagem a ser comprimida, ou seja, quanto mais os vizinhos da imagem forem correlacionados, mais eficiente será a compressão.

A idéia básica das técnicas de codificação preditiva é gerar um vetor de variáveis aleatórias não correlacionadas, de uma imagem, pela utilização de uma transformação [9]. A estrutura da transformação é definida pela utilização de um modelo apropriado da imagem, como a diferença entre o nível de cinza da imagem e o nível de cinza predito pelo modelo. Estas diferenças não são correlacionadas com a imagem original e podem ser representadas por muito menos bits. Com as diferenças e os parâmetros do modelo, a imagem pode ser reconstruída.

Uma das maiores representantes deste tipo de abordagem é a técnica DPCM descrita a seguir.

DPCM

Um dos mais antigos métodos de codificação preditiva, o DPCM (Differential Pulse-Code Modulation) tem sido extensivamente estudado desde seu desenvolvimento. Foi muito utilizado para transmissão de sinais de imagem e vídeo digitais

[23]. Este método é baseado na eliminação da redundância interpixel, pois extrai e codifica somente a nova informação de cada *pixel*. Esta nova informação de cada *pixel* é definida como a diferença entre o *pixel* atual e o valor predito de cada *pixel* [6].

O principal componente de um método DPCM é o preditor [9], que é escolhido no início do funcionamento do método, podendo ser o primeiro *pixel* da imagem, o de posição (0,0). Então a diferença (chamado de erro de predição do *pixel*) entre o preditor e o próximo *pixel* é calculada e será este valor que será transmitido ou armazenado. Para a reconstrução da imagem, o valor de nível de cinza de um *pixel* será o seu erro de predição somado ao preditor. Um preditor pode ser atualizado ao longo da execução do método para uma mesma imagem, por exemplo, sempre que o erro de predição ultrapassar um certo limite conhecido a priori, o que evita o acúmulo do erro.

O problema principal deste método é a escolha do preditor. Uma possível abordagem para tal é utilizar um modelo estatístico dos dados para derivar uma função que relacione os *pixels* vizinhos dentro de uma janela como o explicado por Donny e Haykin em [1]. Estes mesmos autores ainda sugerem uma outra maneira de escolher o preditor, utilizando redes neurais.

Um preditor baseado numa soma linear de pesos de seus vizinhos é relativamente fácil de se projetar utilizando características estatísticas da imagem. Contudo, se um modelo não-linear for mais apropriado para uma imagem, o uso de preditor linear irá claramente resultar numa solução sub-ótima. Infelizmente, o projeto de preditores não-lineares é geralmente não tratável matematicamente, como são os preditores lineares. A utilização, então, de redes neurais para projetar preditores não-lineares resulta em excelentes resultados, como pode ser visto em Donny e Haykin [1].

A técnica DPCM tem sido muito utilizado para compressão de imagens médicas sem perdas ou em combinação com outros métodos de compressão que eliminam as informações psicovisuais irrelevantes [15].

Em aplicações que admitem alguma perda de informação na reconstrução das imagens, o erro de predição pode ser quantizado [6; 15], ou ainda, quando o erro de predição assumir valores menores que um determinado mínimo pré-estabelecido ele pode ser transmitido como sendo de valor zero.

Métodos para melhorar a escolha do preditor e maneiras de minimizar os bits para sua codificação e de atualizá-lo estão sendo investigados visando uma melhor eficiência na compressão.

Várias extensões do método DPCM foram desenvolvidas, como os apresentados por Jain [9], ou ainda utilizadas em combinação com outros métodos de codificação. Jiang em [29] afirma ter encontrado o método DPCM em sistemas de telecomunicações para processar sinais analógicos. Ele afirma ainda que encontrou codificação preditiva com redes neurais e e quantização vetorial. O próprio Jiang em [29] usa uma extensão de DPCM para 3D associado com JPEG.

3.1.2 Codificação por Transformada

A outra abordagem dos métodos de compressão de imagens da primeira geração é o uso de transformações T discretas lineares bidimensionais, que operam em uma dada imagem, produzindo como saída um conjunto de coeficientes.

Um subconjunto desses coeficientes é escolhido e então quantizado para transmissão através de um canal ou para armazenamento [1]. Quando necessário, para se reconstruir a imagem original, os coeficientes quantizados da transformada são utilizados pela transformada inversa. Em um esquema de codificação por transformada, dois problemas básicos devem ser resolvidos: a escolha da transformada discreta e a escolha dos coeficientes apropriados para representar a imagem.

A grande vantagem da utilização das transformadas para compressão de sinais é que o sinal é projetado em uma base de funções ortogonais da tal maneira que a energia do mesmo fica distribuída em um conjunto de componentes descorrelacionados [24], chamados de coeficientes.

Um método de codificação por transformada divide a imagem original em blocos, e executa a transformação nestes blocos independentemente uns dos outros [9]. Essa transformação deixa o sinal original praticamente descorrelacionado ou mais independente [22], e geralmente resulta na redistribuição da energia espectral da imagem original em pequeno conjunto de coeficientes da transformada pertencentes ao bloco [23], sempre preservando a energia [9] que normalmente fica concentrada sobre alguns pixels, os de baixa freqüência. Em muitos casos, alguns coeficientes podem ser descartados (rejeitados), o que pode acarretar em perda visual da informação.

A meta dessa abordagem é escolher uma transformação para que o subconjunto de coeficientes seja adequado para a reconstrução da imagem com um mínimo de distorção perceptível aceitável para uma determinada aplicação.

Os algoritmos que lançam mão desta abordagem exploram a informação contida na freqüência espacial da imagem para alcançar a compressão. Existem muitos tipos de técnicas de compressão por transformada, sendo os mais encontrados na literatura a transformada DCT e recentemente, a transformada *wavelet*, que serão explicadas nas seções que se seguem.

Transformada *Wavelet*

A transformada *wavelet* é uma ferramenta matemática desenvolvida recentemente. Já é um dos métodos de compressão mais utilizados, pois representa fielmente sinais com descontinuidades, tais como sinais de voz e imagens. A transformada *wavelet* é utilizada na decomposição de sinais em uma combinação linear de membros de uma família de funções, que são obtidos a partir da translação e da dilatação de uma única função base $\Psi(x)$, denominada *wavelet* mãe [30].

Esta transformada representa o sinal em diferentes escalas e com diferentes resoluções (transladadas e escalonadas) de uma mesma função modelo conhecida como *wavelet* mãe. Por exemplo, quando um sinal é analisado através de uma

grande janela é possível identificar suas características gerais, pelas baixas freqüências. Similarmente, se o sinal for analisado por uma janela pequena, pequenas características, ou detalhes, podem ser observados, através das altas freqüências [31].

As *wavelet* são ondas com comprimento e duração limitadas, por esta razão são indicadas para análise de sinais que apresentam descontinuidades, tais como imagens, uma vez que permitem associar características no domínio da freqüência com sua localização no tempo [31]. Ou seja, nas *wavelet* a noção de freqüência é trocada pela noção de escala, tendo-se uma representação tempo-escala ao invés de uma representação tempo-freqüência [24].

A função *wavelet* bidimensional discreta é dada por [32]:

$$\Psi_{j,k} = a_0^{-j/2}\Psi(a_0^{-j}t - kb_0), a_0 > 1, b_0 > 0 \ e \ j,k \in Z \tag{3.1}$$

Nesta equação a_0 é o fator de escala e b_0 é o fator de translação.

Enquanto a DCT é a transformada escolhida por muitos padrões de compressão comerciais já em uso como o JPEG e o MPEG, a transformada *wavelet* discreta está aos poucos emergindo como uma melhor alternativa. O padrão de compressão JPEG-2000, por exemplo, faz uso dessa transformada [33; 34]. A transformada *wavelet* tem se mostrado uma alternativa melhor do que a DCT, pois tem apresentado melhores resultados em termos de eficiência da compressão (Seção 2.2).

A relação taxa de compressão versus distorção da imagem reconstruída, é sempre favorável às *wavelet*, conforme estudo feito por Corrêa [24]. Neste trabalho, pode ser observado que eliminando-se o mesmo número de coeficientes para ambas, com isso alcançando-se a mesma taxa de compressão, numa abordagem pura, os melhores resultados obtidos, sejam pela PSNR ou por testes subjetivos, foram sempre por transformadas *wavelet*.

Shapiro [25] apresenta um algoritmo de compressão de imagens, entitulado EZW (embedded zerotree *wavelet*), que é baseado na transformada *wavelet* discreta.

A transformada *wavelet* também é muito utilizada em abordagens híbridas [28; 35; 36; 37]. Nestas, uma transformada *wavelet* é aplicada na imagem original,

decompondo-a em subbandas. Algumas destas bandas são codificadas com uma técnica específica e o restante é codificado com uma técnica mais simples, como quantização escalar. Tanto no trabalho de Mayer [28], quanto no trabalho de Bernardino [35], a subbanda resultante da decomposição escolhida para ser codificada por suas técnicas desenvolvidas foi a subbanda de aproximação. Resumindo, a transformada *wavelet* é muito utilizada em abordagens híbridas com o objetivo de melhorar as taxas de compressão de técnicas desenvolvidas e em desenvolvimento. Freqüentemente, a transformada *wavelet* é executada primeiro e uma outra técnica é executada em algumas de suas sub-bandas.

Transformada DCT

A transformada discreta do coseno ou DCT[2] como é mais conhecida, desde que foi introduzida nos anos 70, vem sendo utilizada por muitos algoritmos de compressão, principalmente os com perda de informação [15]. Nestes, podem ser incluídos os dos grupos de desenvolvimento das normas de compressão **JPEG** (Joint Photographic Expert Group), **MPEG-1** e **MPEG-2** (Moving Picture Experts Group), e ainda as normas **H.261**, e **ITU-R 723**, por exemplo, que serão explicadas mais adiante no texto.

A DCT é uma transformada determinística e linear, sendo a mais popular técnica de transformada utilizada tanto para compressão ou apenas codificação de imagens como para vídeo (seqüência de imagens).

A DCT bidimensional para blocos NxN da imagem é dada por [38; 39]:

$$F(u,v) = \frac{2}{N} c(u)c(v) \left(\sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m,n) \cos((\pi(2m+1)u)/2N) \cos((\pi(2n+1)v)/2N) \right) \tag{3.2}$$

onde $c(w) = \{2^{-1/2}\}$ para $w = 0$; $c(w) = 1$, para $w = 1, 2, ..., N-1$.

[2]Do inglês *Discrete Cosine Transform.*

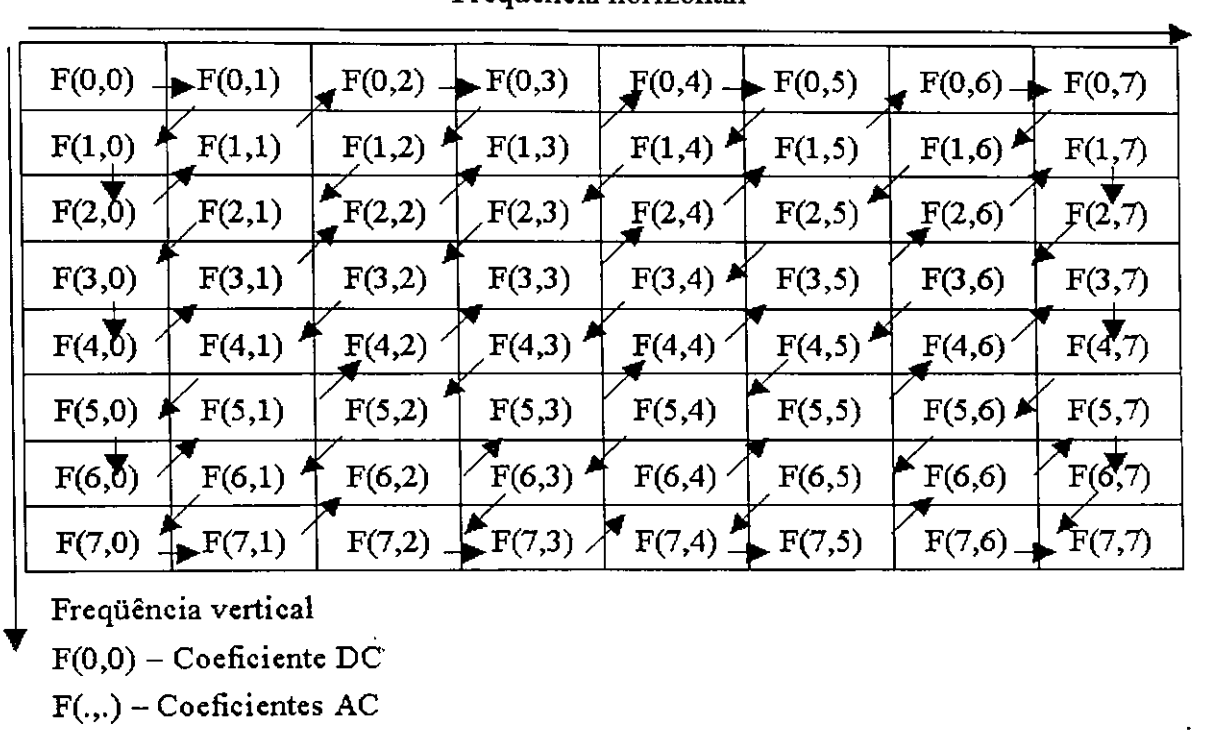


Figura 3.1: Ordenamento dos coeficientes da DCT, exemplificando a varredura bidimensional em zig-zag, das baixas para as altas freqüências.

Similarmente, a transformada discreta do coseno inversa bidimensional (IDCT) é definida como se segue [38; 39]:

$$f(m,n) = (2/N) \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} c(u)c(v)F(u,v) \cos((\pi(2m+1)u)/2N) \cos((\pi(2n+1)v)/2N) \tag{3.3}$$

O tamanho de bloco padrão da DCT é de 8x8 pixels adjacentes, como as submatrizes de tamanho 8x8 utilizadas no padrão de compressão JPEG. Mas, hoje já estão sendo utilizados outros tamanhos de blocos, como 16x16 *pixel*, utilizado, por exemplo, no trabalho de Modestino [38].

A DCT é computada para cada bloco da imagem e um quantizador é aplicado para os coeficientes da transformada gerados. Esta computação equivale a uma análise em freqüência, ou seja, cada coeficiente da transformada carrega informações do conteúdo da imagem em alguma porção do espectro de freqüências, ou

ainda, cada coeficiente carrega a informação da soma da ocorrência de determinada freqüência na janela NxN. Este fato permite a priorização dos coeficientes no sentido de um melhor casamento com a resposta do sistema visual humano.

Os coeficientes transformados são arranjados em uma seqüência unidimensional através de uma varredura bidimensional dos blocos da imagem. Comumente esta varredura se faz em zig-zag como exemplificado na Figura 3.1 , para o caso de um bloco de tamanho 8x8. Este tipo de varredura se dá das baixas para as altas freqüências [15].

Os coeficientes de um mesmo bloco são codificados de maneira diferente. O primeiro coeficiente varrido dos 64 que aparecem na Figura 4.1, o F(0,0), é chamado de coeficiente DC, o qual representa a quantidade de valores da freqüência zero que foram encontrados naquele bloco. Este coeficiente especial é codificado preditivamente em relação ao coeficiente DC do bloco anterior para economizar bits em seu armazenamento ou transmissão, o que comumente ocorre através do método DPCM. Deste modo, somente a diferença entre eles passará a representá-lo, o que normalmente é um valor muito menor que o valor absoluto do coeficiente. Isto é feito para explorar a forte correlação existente entre os blocos vizinho.

Os 63 coeficientes remanescentes do bloco analisado, são chamados de coeficientes AC, e também são codificados por codificação preditiva só que desta vez em relação ao coeficiente DC do mesmo bloco, para tanto normalmente é utilizado também o método DPCM [33]. Isto é feito para descorrelacionar os elementos de um mesmo bloco.

O método de compressão desenvolvido por Bárbara e Alcaim [40] utiliza DCT com máscaras zonais passa-baixas de 3 coeficientes apenas, com o objetivo de melhorar a qualidade subjetiva da imagem, diminuindo o efeito de blocagem originado pela *Quadtree* (seção 3.1.4).

3.1.3 Quantização Vetorial

A quantização vetorial (QV) tem sido objeto de estudo para aplicações envolvendo compressão de sinais de voz e imagem, apresentando-se como uma poderosa ferramenta, permitindo elevadas taxas de compressão [12].

A QV é embasada na teoria da distorção versus taxa, formulada por Shannon, que diz que é sempre possível obter um melhor desempenho codificando um conjunto de amostras (isto é, um vetor de amostras), ao invés de codificar cada amostra individualmente. Em outras palavras, essa teoria ressalta a superioridade da QV sobre a quantização escalar [35; 15].

A QV pode ser vista como um mapeamento Q de um espaço vetorial k-dimensional R^k em um subconjunto finito W de R^k [12; 17; 15; 11; 41; 42]:

$Q : R^k \longrightarrow W$

O conjunto W é dado por $W = \{w_i | i = 1, 2, ..., N\}$, sendo o conjunto de vetores de reprodução, e N é o número de vetores de W. Cada w_i em W é chamado de vetor-código e W é chamado de dicionário do quantizador vetorial.

A imagem é particionada em blocos preferencialmente quadráticos, de dimensão k, em que k representa o número de *pixel* de cada bloco. Cada bloco é, então, casado com algum dos vetores–código do dicionário, em geral, o mais "próximo"do bloco. Um algoritmo de QV calcula a distância d entre os blocos de entrada e cada vetor-código w_i do dicionário. O índice i do vetor-código que apresentar a menor distância será então utilizado para representar este bloco de entrada, seja para transmissão, ou para armazenamento [43].

Os dados comprimidos da imagem nada mais são do que uma lista dos índices dos vetores–códigos escolhidos para representar os blocos em que a imagem fora divida. O decodificador utiliza estes índices para reconstruir a imagem através da concatenação dos correspondentes vetores–códigos.

O cálculo da distância d que é comumente utilizada é o erro médio quadrático dado pela média dos ds, onde x' é o vetor–código associado a x [41; 42]. O tamanho dos elementos do dicionário é tipicamente uma potência de 2, isto é, $N = 2b$, em que os índices dos vetores–códigos são representados por uma quantidade de b bits

[15].

Obviamente, a escolha do conjunto de vetores–código é de grande importância. É este exatamente o problema da quantização vetorial: o projeto de um dicionário que minimize a distorção média da imagem reconstruída em relação à imagem original, introduzida pela aproximação de cada bloco de entrada por um dos vetores-código.

Dentre as técnicas utilizadas para projeto de dicionários para QV, destaca-se o algoritmo LBG, que é a técnica mais popular e amplamente utilizada para projeto de QV [12]. A outra parte dos projetos utilizam redes neurais, como o algoritmo de rede neural de Kohonen. Ambos algoritmos serão descritos nas duas seções que se seguem [44].

Algoritmo LBG

Este algoritmo e suas variações são, atualmente, os mais utilizados para projeto de dicionários para QV. O algoritmo LBG (Linde – Buzo – Gray) utiliza uma seqüência representativa de dados para treinamento do dicionário, que define uma densidade de probabilidade de entrada, baseada na densidade atual [43].

O algoritmo LBG consiste da seguinte seqüência de passos [12]:

- Passo 1 – condição inicial: inicialize com qualquer configuração inicial desejada. Vá para o passo 2, se a inicialização ocorreu com um conjunto de vetores–código (dicionário inicial); vá para o passo 3 se a inicialização ocorreu com a partição dos dados;
- Passo 2 – particionamento: aloque cada dado (ou vetor de entrada) na respectiva classe segundo o critério do vetor-código mais próximo;
- Passo 3 – atualização do dicionário: compute os novos vetores–códigos como os centróides das classes de dados;

- Passo 4 – teste de convergência: repita o passo 2 e passo 3 até a convergência do processo.

Inicialmente, neste algoritmo, os vetores–código formam um conjunto com valores aleatórios. Em cada iteração, cada bloco do conjunto de treinamento é associado ao vetor–código mais próximo. Então é verificado se este "próximo" satisfaz um valor de erro mínimo aceitável, se não, um novo valor para o vetor–código é calculado. As iterações continuam até que todos os blocos estejam associados a vetores–códigos aceitáveis. Com isso, este algoritmo minimiza o erro médio quadrático do conjunto de treinamento. Enquanto o algoritmo LBG converge para um mínimo local, não é garantida a pesquisa para um mínimo global. O LBG é muito sensível ao dicionário inicial podendo ser muito lento pois ele faz uma pesquisa exaustiva no dicionário inteiro a cada iteração [15].

Um novo vetor–código é calculado, como explica o passo 3, através do cálculo do centróide. O centróide é um vetor fortemente representativo do dicionário, que assume o valor da distorção mínima média e é atualizado a cada iteração [43].

Existem alguns problemas apresentados pelo algoritmo LBG [12]:

- Alguns vetores-códigos podem ser sub-utilizados e, em casos extremos, até mesmo nunca serem acessados, na fase de quantização.
- O algoritmo converge para um dicionário localmente ótimo sob determinadas condições, entretanto, converge para um dicionário diferente quando uma condição inicial diferente é aplicada.
- A velocidade de convergência e o desempenho do dicionário final dependem do dicionário inicial.

Objetivando resolver um conjunto destes problemas, muitos trabalhos foram e ainda estão sendo desenvolvidos, propondo variações deste algoritmo clássico de quantização vetorial.

Rede Neural de Kohonen

O algoritmo de Kohonen consiste em uma técnica para treinamento de uma rede neural de aspecto auto–organizante (*self–organization*). Ao contrário dos algoritmos de aprendizagem supervisionada, que durante a fase de treinamento necessitam de uma resposta externa para cada padrão de entrada ou dado de treinamento, o algoritmo de Kohonen utiliza aprendizagem não supervisionada para modificar o estado interno da rede neural e desta forma modelar as características encontradas nos dados de treinamento. Em outras palavras, nos mapas auto–organizantes de Kohonen, a rede neural apresenta, por si mesma, sem supervisão, uma organização coerente dos dados de treinamento [12].

Kohonen introduziu uma técnica de agrupamento de rede neural onde os pesos entre os neurônios são adaptados. Os valores dos pesos definem o dicionário de entrada [42].

No algoritmo de Kohonen, um mapa topográfico é organizado, de maneira autônoma, por um processo cíclico de comparação dos padrões de entrada com vetores armazenados em cada neurônio. Nenhuma resposta de treinamento é especificada para qualquer entrada de treinamento. Onde as entradas se equiparam aos vetores dos neurônios, aquela área do mapa é seletivamente otimizada para representar uma espécie de média dos dados de treinamento para aquelas classe. De um conjunto de neurônios organizados aleatoriamente, a grade atinge um mapa de características que apresenta representação local e é auto–organizado.

O algoritmo é muito simples [12]:

1. Apresente o vetor de treinamento x;

2. Encontre o neurônio vencedor $w_{i.}$ (w é o peso de i), de acordo com o critério de distância mínima:

$d(x, w_i*) < d(x, w_i), \forall i \neq i_*$;

3. Atualize $w_{i.}$ e a vizinhança $N_{wi.} = \{w_i | d(w_i, w_{i.}) \leq r(n)\}$, na direção de x, ou seja, $\Delta w_{i,j} = h(n) O_i x_j w_{i,j}$.

No algoritmo descrito acima, d é a medida de distorção, n é o passo ($1 \leq n \leq n_{max}$), $h(n)$ é a taxa de aprendizagem na n-ésima iteração, $r(n)$ é o raio de vizinhança na n-ésima iteração, O_i é a função que define a vizinhança ao redor do neurônio w_{i*}, (O_i é não-nulo para $w_i \in N_i$, e nulo caso contrário), x_j é a j-ésima componente de x, w_{ij} é a j-ésima componente de $w_i (1 \leq i \leq N, 1 \leq j \leq k)$. Os passos de 1 a 3 são repetidos até que todos os vetores da seqüência de treino sejam apresentados. Tanto a taxa de aprendizagem como a função que define a vizinhança decrescem com a iteração e os passos de 1 a 3 são repetidos. O procedimento completo é repetido iterativamente n_{max} vezes (n_{max} passagens da seqüência de treino). A taxa de aprendizagem $h(n)$ e o raio de vizinhança $r(n)$ decrescem a cada iteração n.

Resumindo:

- Encontre a unidade mais semelhante à unidade de treinamento;
- Aumente a similaridade dessa unidade e das unidades pertencentes à vizinhança da entrada.

3.1.4 Codificação Multirresolucional/Multiescala

Este tipo de abordagem se caracteriza por operar na imagem original para produzir vários níveis de detalhes da imagem, progressivamente menores [16]. Estes níveis de detalhes, também chamados de pirâmides ou multiresoluções, são classes de representação hierárquica de imagens, baseado no princípio de decomposição regular [22].

Codificação multirresolucional também pode ser chamada de codificação Subbanda desde que os níveis de resolução em que a imagem original é dividida sejam os componentes espectrais, ou seja, as faixas de freqüência, obtidas através da execução de um conjunto de operações de filtragem. Por exemplo, de uma imagem pode ser obtida uma subimagem menor que representa os componentes de baixa freqüência. A sua respectiva subimagem com os componentes de alta freqüência

deve conter necessariamente as informações de borda. Neste caso, a compressão pode ser feita através de codificações diferentes para cada subimagem, dando ênfase para a subimagem que representa os componentes de alta freqüência. Maiores explicações sobre codificação sub-banda, utilizando–se filtros, podem ser encontrados em [45; 46].

A transformada *wavelet* foi neste trabalho, classificada como sendo um método baseado em transformadas [25; 26]. Entretanto, não está errado classificar as transformadas *wavelet* como sendo um método de codificação sub-banda [15; 16], pois esta transformada também atende a definição deste tipo de abordagem.

Como outros exemplos de abordagem multirresolucional, podemos citar as quadtrees e as pirâmides Laplacianas. Ambas se encontram resumidas a seguir.

Quadtrees

As quadtrees são representações bidimensionais construídas através de divisões recursivas da imagem original (retangular) em quatro retângulos, blocos ou quadrantes, os quais podem ser subdivididos em outros quatro sub–quadrantes, e assim por diante. Esta estrutura, constrói uma árvore de pesquisa com grau 4. A raiz da árvore está associada com a imagem completa. Uma árvore quadtree, com n níveis de profundidade, permite a representação de até 4^{n-1} quadrados, lembrando que a raiz é o nível 0. A Figura 3.2 apresenta uma imagem qualquer dividida em quadrantes e sub–quadrantes, e a sua respectiva árvore quadtree [10].

Estas estruturas são amplamente utilizadas por algoritmos de processamento e compressão de imagens. A maior parte das aplicações de Quadtree nestas áreas visa tirar proveito da grande capacidade de descrição de regiões homogêneas e de identificar bordas dos objetos presentes na cena [20; 21; 40; 27; 47; 48; 49].

No trabalho de Shusterman e Feder [20], são citadas 3 razões para a codificação de imagens em níveis de cinza por decomposição quadtree:

- Relativa simplicidade comparada com outros métodos (por exemplo, codifi-

cação baseada em DCT), o que a torna um método atrativo para aplicações como compressão de vídeo e HDTV;

- Adaptabilidade das decomposições. A decomposição divide a imagem em regiões com tamanho variável, dependendo dos valores de nível de cinza na região. A performance da técnica de compressão é assim adaptada às várias regiões da imagem.

- Usabilidade da saída da decomposição. A decomposição pode resultar em um tipo de segmentação de imagens. Esta segmentação pode ser utilizada por uma variedade de diferentes aplicações de processamento de imagens, por exemplo, reconhecimento de padrões.

Segundo Sethuraman et alli [27], os métodos para segmentação baseados em regiões, como a *Quadtree*, tem sido escolhidos para processamento de imagens, ao invés dos métodos baseados em características, como a segmentação baseada em contornos, porque nestes, a extração de características é computacionalmente custosa, e o estabelecimento de correspondências entre as características extraídas é bem mais complexo. Conseqüentemente a propagação do movimento / disparidade correspondente a regiões descritas por contorno não é trivial. Em outras palavras, segmentação baseada em regiões é realizada somente em função dos valores de intensidade dos *pixels* que estão a priori disponíveis: desde que a segmentação por si só faz a identificação de uma região de textura, não é requerido a propagação de correspondências. Contudo, o *overhead* para codificação de decomposição em regiões com formas arbitrárias (como o trabalho de Radha et alli em [50], por exemplo) é muito maior do que o requerido para a codificação de regiões divididas em blocos retangulares de mesmo tamanho em cada divisão e sucessivamente menores em cada recursão.

Resumindo, com a decomposição por quadtree as regiões da imagem com muitos detalhes são divididas em pequenas regiões para uma análise mais apurada destas. Por isso, pode ser demostrado que realizar a segmentação com quadtree é

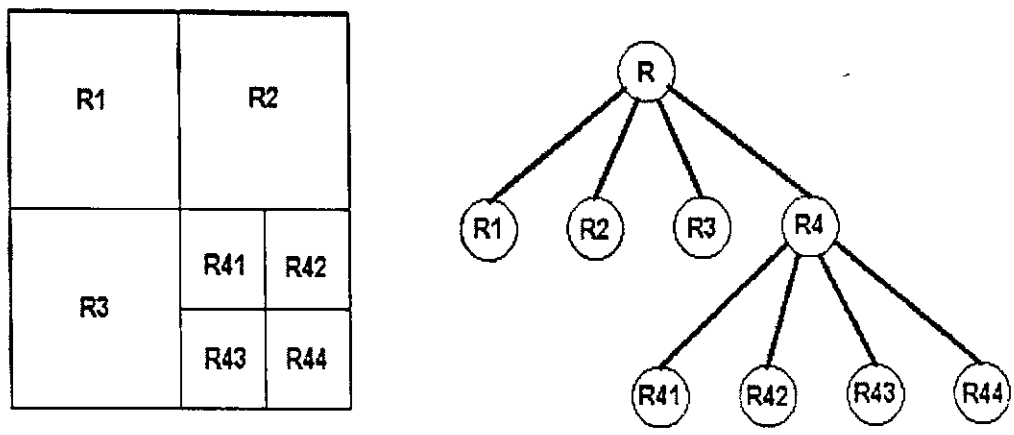


Figura 3.2: Imagem qualquer dividida em quadrantes e subquadrantes, e sua respectiva árvore quadtree.

uma maneira eficiente de fazê-la [48].

Existem duas técnicas básicas para a construção de quadtree: *top–down* (técnica de cima–para–baixo) e *bottom–up* (técnica de baixo–para–cima) [20; 40; 47]. A Figura 3.3 exemplifica estas duas técnicas.

Na construção *top–down* um teste na imagem completa é feito para pesquisar se ela deve ser representada por uma única folha na árvore (a raiz) ou se deve ser dividida em mais quatro nodos. Caso a imagem inicial seja dividida, então a mesma pesquisa é feita para cada novo nodo para determinar se eles podem ser subdivididos ou não. O processo termina quando não se puder mais subdividir os nodos, seja por que estes assumiram o menor tamanho possível, seja por que a árvore atingiu o nível máximo permitido a priori [51].

A construção *bottom–up* é baseada em critérios de agrupamento (*merge*). Inicialmente divide–se toda a imagem em blocos (no caso das quadtrees) no tamanho mínimo possível ou permitido (as folhas). A seguir analisa-se se quatro blocos adjacentes podem ser unidos em um único bloco homogêneo, dependendo do critério de homogeneidade. Se a união dos quatro blocos resultar em um bloco homogêneo, então os blocos são unidos formando um bloco de tamanho maior. O processo termina quando chega-se ao tamanho de um bloco máximo pré-fixado ou ao tamanho

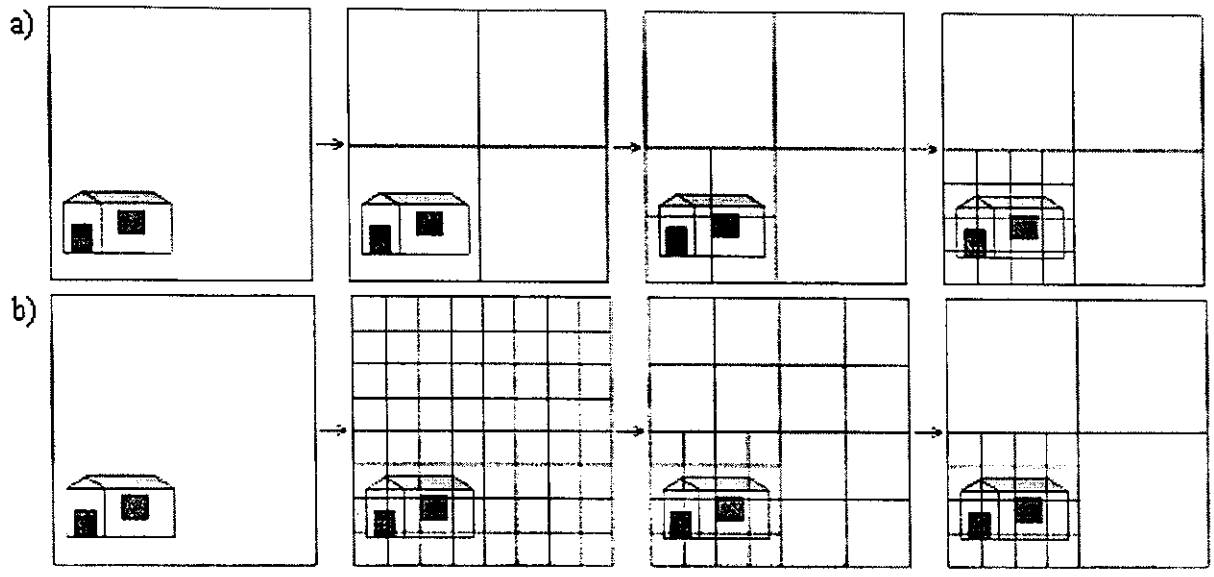


Figura 3.3: Maneiras de construção de uma Quad-tree: a) top-down e b) bottom-up.

da imagem original.

Bárbara e Alcaim [40] melhoraram o desempenho em termos de PSNR das Quadtrees introduzindo o conceito de limiares variáveis para o teste de homogeneidade, para decidir se um retângulo pode ser subdividido ou não. Quanto menor o tamanho do bloco sendo analisado maior o limiar, ou seja, à medida que o tamanho dos blocos aumenta, diminui o valor de limiar.

Shusterman e Feder [20] apresentam em seu artigo uma Quadtree melhorada em termos de taxa de compressão, através do emprego de alocação dinâmica de bits para cada um dos níveis da árvore. Em outro artigo [36] os mesmo autores utilizam a quadtree numa abordagem híbrida com *wavelets*. Primeiro é executada a transformada *wavelet* na imagem original para obter sua representação multirresolucional. Em seguida, são construídas 3 quadtrees para representar as componentes de detalhes horizontal, vertical e diagonal, que são codificadas através do algoritmo desenvolvido em [20]. O objetivo deste trabalho é estudar a correlação entre diferentes níveis de resolução.

No trabalho de Vaisey e Gersho [52] também são utilizadas Quadtrees *top-down* para segmentar imagens sempre em blocos que variam de tamanho de 32x32 a 4x4

(sempre potência de 2). Contudo, eles introduzem o conceito de codificadores de imagem com taxa variável que mudam a resolução da codificação (em bits por unidade de área) de acordo com a característica local e importância da região codificada. Ou seja, Vaisey e Gersho classificam as regiões decompostas da imagem de acordo com sua importância perceptual e as codifica de forma diferente. As regiões com muito detalhe são sempre segmentadas em blocos 4x4 e codificadas com uma maior fidelidade. Regiões de poucos detalhes são segmentadas em blocos o mais largo possível entre 32x32 e 8x8 e codificados com pouca fidelidade. As pequenas regiões de muito detalhe são codificadas com quantização vetorial. Regiões de poucos detalhes são codificadas com abordagem híbrida de quantização vetorial com transformada. Ainda é utilizado um pós-processamento para diminuir o efeito de blocagem.

Wu em [53] faz codificação de imagens baseada em segmentação em formas fixas através de uma Quadtree modificada. Neste trabalho ele permite até 4 formas geométricas para decompor a imagem, não apenas a retângular como no caso da Quadtree. Wu afirma que os bits necessários para guardar a informação de qual particionamento foi utilizado, são justificados pela melhor aproximação da cena.

Na literatura estudada, foi encontrado um outro trabalho que utiliza uma abordagem híbrida de *Quadtree* e *wavelets*, o de Munteanu em [37]. Neste trabalho, também a transformada *wavelet* é aplicada primeiro. Entretanto, uma diferença significativa deste trabalho é que o método desenvolvido é flexível podendo ser com ou sem perdas.

Para fins de comparação de resultados com o novo método implementado [3; 10], chamado de *Tritree*, foi implementada a *Quadtree*, cujos resultados e comparações podem ser encontrados no próximo capítulo.

Pirâmides Laplacianas

Pioneiramente, Burt e Adelson [54] apresentaram uma técnica baseada nas pirâmides Laplacianas como um método de compressão de imagens. A codificação

de imagens por pirâmides Laplacianas consiste de alguns estágios importantes. No primeiro estágio, um filtro passa–baixa é aplicado à imagem original. Este estágio produz subimagens progressivamente menores.

No segundo estágio, cada pirâmide laplaciana, começando pela menor, é interpolada para atingir o tamanho do seu predecessor. Esta interpolação é feita utilizando–se o mesmo filtro passa–baixa utilizado no primeiro estágio. Esta versão expandida é então subtraída do seu predecessor, uma vez que ambos tem a mesma extensão espacial. A diferença entre estas duas imagens fornece os detalhes espaciais de alta freqüência, para aquele nível. A coleção final de detalhes das imagens em diferentes níveis de resolução é chamado de pirâmides laplacianas [21].

O estágio final deste processo de codificação é quantizar cada nível da pirâmide. A compressão é atingida de duas maneiras: através do processo de quantização dos *pixels*, ou através de omissões conscientes de alguns níveis da pirâmide laplaciana. Pela própria natureza deste método ele é particularmente interessante para a transmissão progressiva de imagens. Isto deve-se ao fato de que cada nível da pirâmide fornece um conjunto de detalhes da imagem original. Se uma melhor qualidade da imagem é necessária, é suficiente que o próximo nível da pirâmide seja enviado [16].

3.2 Padronização das Normas de Compressão

O salto para o mundo digital na área de compressão de imagens ganhou mais força com o aparecimento das primeiras normas internacionais digitais. Isto permitiu que a comunicação audiovisual ultrapassasse os limites do entretenimento em larga escala, e chegasse de maneira significativa à área das comunicações pessoais, como por exemplo a video-telefônia e a video-conferência. Mesmo usando sinais digitais, as normas atuais não mudaram o essencial do modelo de representação: uma imagem é formada por um conjunto de linhas, fazendo apenas a tradução do modelo analógico.

A normalização foi necessária uma vez que cada fabricante adotava a sua solução para compressão/codificação de imagens de vídeo. Estas soluções eram normalmente esquemas de codificação híbridos, que faziam uso de várias das abordagens estudadas na Seção anterior, que tiveram e ainda têm uma importância fundamental no contexto das normas internacionais [55]. Apenas foram acrescentadas à lista da Seção anterior as duas abordagens seguintes:

- Pulse Code Modulation (PCM): uma representação discreta no tempo, não eficiente em termos de representação por conservar a redundância, mas que é utilizada como método de representação intermediária e como termo de comparação.

- Codificação Entrópica: elimina a redundância na codificação, codifica os símbolos gerados pelas técnicas do Capítulo quatro, atribuindo a cada símbolo uma palavra de código, levando em consideração a distribuição estatística da ocorrência dos símbolos, ou seja, aos mais prováveis são atribuídas palavras de menor comprimento e aos menos prováveis, palavras de maior comprimento. Todas as norma internacionais de codificação de imagem e vídeo utilizam codificação entrópica, normalmente codificação de Huffman ou aritmética. Essa codificação é totalmente reversível pois não introduz perdas em nenhuma circunstância.

Os organismos internacionais de padronização como o ISO (International Organization for Standardisation), ITU (International Telecommunication Union) e IEC (International Electrotechnical Commission) realizaram um estudo da grande variedade de soluções disponíveis entre os pesquisadores da área, reunindo o que cada um produziu de melhor na tecnologia que ficou sendo a mais utilizada: os codificadores híbridos. Com base nestes, os codificadores híbridos, foram criados a partir dos anos 80, várias normas internacionais para codificação e compressão de imagem e vídeo, garantindo maior interoperabilidade e viabilizando a explosão da tecnologia digital [7]. Estas normas são descritas a seguir.

3.2.1 ITU-R 601

Este foi o primeiro passo em termos da normalização da representação de sinais digitais de vídeo, dado pelo então Comitê Consultivo Internacional para a Radiodifusão (CCIR, hoje ITU-R), que estabeleceu em 1982 o método de codificação de sinais televisivos de estúdio. A norma estabelece basicamente que se deve usar um sinal de luminância, Y, e 2 de crominância, U e V, amostrados a uma freqüência, respectivamente, de 13.5 e 6.75MHz o que, usando 8 bit/amostra, dá origem a um taxa de 216 Mbit/s. A norma não estabelece qualquer método de compressão da informação limitando-se a usar PCM. Para os sistemas europeus, esta norma indica uma resolução temporal de 25 imagem/s, entrelaçadas, com uma resolução espacial de 720x576 amostras de luminância e 360x576 amostras de cada uma das crominâncias [7].

3.2.2 ITU-T H.120

Dois anos depois do lançamento da norma anterior, em 1984 recebeu a aprovação final a norma ITU-T H.120, o primeiro padrão internacional para codificação de vídeo [7]. Talvez tenha sido esta também a primeira norma internacional para compressão de imagem ou vídeo tone-continuo. Esta norma foi desenvolvida pela organização ITU-T [3] [55].

Esta norma destinava-se essencialmente à transmissão de video-conferência através das linhas digitais de 1° hierarquia com taxas de 1544 e 2048 kbit/s. O método de codificação usado é o "preenchimento condicional[4] " (CR) ou seja uma forma particular de DPCM (Seção 3.1.1) onde se envia o erro de predição se for superior a um dado limite. Esta norma não utilizava ainda compensação de movimento para melhorar a predição e foi rapidamente "ultrapassada" pela norma H.261 (Próxima Seção) [7].

[3]International Telecommunications Union – Telecomunications Standardization Sector, anteriormente chamado de CCITT

[4]Conditional replenishment

3.2.3 ITU-T H.261

No fim da década de 80, em 1988, aparece a norma CCITT (International Telegraph and Telephone Consultative Committee), agora chamada por ITU-T H.261, a qual só foi aprovada em 1991 (com o conteúdo técnico aprovado em 1990). Este foi o primeiro padrão de utilização prática com sucesso, para codificação do sinal de vídeo associado a serviços video-telefônicos e de video- conferência com débitos de $p \times 64kbit/s$, onde p é um inteiro que varia de 1 a 30 [7].

Esta norma consagra, pela primeira vez, o esquema de codificação híbrida, cuja estrutura básica ainda é predominante até os dias de hoje. A H.261 é baseada em codificação por transformada DCT com blocos de tamanho 8x8 (Seção 3.1.2), codificação preditiva (Seção 3.1.1) com compensação de movimento com blocos de tamanho 16 × 16 e codificação entrópica. Depois de calcular o movimento relativo entre o quadro atual e o anterior, o algoritmo decide entre codificar apenas a diferença entre os quadros, ou simplesmente codificar um novo quadro. É o que se chama de codificação inter-frame e intra-frame, respectivamente. Então os dados são transformados em coeficientes pela DCT, quantizados e codificados por Huffman [55].

A resolução temporal base é de 30 Hz e a espacial é de 360 × 288 amostras para a luminância e 180 × 144 para cada uma das crominâncias (resolução CIF). Versões sub-amostradas no espaço e no tempo são normalmente usadas para os débitos mais baixos.

Esta norma foi revisada pela última vez em 1993 para a inclusão de uma compatibilidade para trás com modos de transferência gráficos de alta resolução. Esta modificação objetiva taxas de bits entre 64 - 2048 Kbit/s [4].

3.2.4 ITU-T H.263

Quando, em 1993, duas grandes empresas mundiais puseram no mercado videotelefones destinados a serem utilizados na rede telefônica analógica, foi evidente a necessidade de normalizar também a codificação de vídeo na zona das

mais baixas taxas, já que a norma H.261 tem como limite inferior de uso os 64 kbit/s. Sendo os dois produtos apresentados incompatíveis entre si, havia o perigo de deixar as portas abertas à criação de alguma confusão no mercado, para não falar da má qualidade da imagem que os produtos ofereciam, fato que poderia ter sérios efeitos negativos na expansão futura dos serviços audiovisuais de comunicação pessoal [4].

É nesse contexto que surge a norma H.263 que, em termos de codificação, pode ser vista como uma nova melhoria do esquema de codificação híbrida, pela primeira vez consagrado com a norma anterior. A norma H.263 destina-se a comunicações video-telefônicas de muito baixas taxas, cerca de 10 - 30 Kbits/s, sendo ainda reconhecido que poderá oferecer melhores desempenhos que a norma H.261 para os débitos mais baixos de uso desta norma, nomeadamente 64 e 128 kbit/s. Esta norma oferece perspectivas de aplicação não só para a videotelefonia na rede analógica mas também para a videotelefonia em redes móveis e controle remoto audiovisual.

A norma H.263 também é um projeto do órgão ITU-T, e foi aprovada no início de 1996 (com o conteúdo técnico aprovado em 1995). Uma das grandes vantagens técnicas deste padrão é a introdução da idéia de compensação de movimento, feita através de blocos de tamanhos variáveis [55].

3.2.5 ITU-T H.263+

Tecnicamente esta é uma segunda versão da norma H.263. Entretanto, o projeto desta norma adiciona um número tão grande de novas características à norma H.263, que convencionou-se classificá-la como uma outra norma. Um avanço notável desta norma sobre a anterior, é que este é o primeiro padrão de codificação de vídeo a oferecer alto grau de resistência a erros para redes de transporte baseadas em pacotes ou redes sem-fio (*"wireless"*).

A norma H.263+ ainda adiciona um bom número de melhoramentos de outros tipos como melhor eficiência da compressão, formatos de vídeo flexíveis e personalizados, escaláveis e informação otimizada suplementar para a compatibilidade para

trás[5].

Este padrão foi aprovado em janeiro de 1998 pelo organismo ITU-T (com o conteúdo técnico aprovado em setembro de 1997) [55].

3.2.6 ISO/JPEG

A história da norma ISO/JPEG começa em 1982, quando se formou o Photographic Experts Group (PEG) com o objetivo de desenvolver pesquisas na área de transmissão de vídeo, imagens e texto através de uma rede digital de serviços integrados (ISDN). Em 1986, um subgrupo do CCITT já fazia pesquisas na área de compressão de imagens para transmissão de fax. Porém, só em 1987, os dois grupos se juntaram e formaram o grupo JPEG (Joint PEG) para gerar padrões internacionais na área de compressão de imagens [4]. Por isso pode também ser chamada de IS 10918-1 ou ainda ITU-T T.81.

Esta norma foi aprovada em 1992, sendo a primeira das normas aqui referidas destinada a imagens fixas, muito poderosa e popular até os dias de hoje. Ela especifica um algoritmo de codificação para imagens fotográficas multi-nível (tons de cinza) ou a cores baseado na codificação por transformada DCT (Seção 3.1.2).

Do padrão JPEG ainda pode ser dito que ele é em sua essência a norma H.261 INTRA codificada com predição de valores médios e com habilidade para personalizar a escala de quantizadores [55].

Para que esta norma pudesse cobrir a grande maioria das aplicações utilizando imagens fotográficas, foram definidos quatro modos principais de operação: seqüencial, progressivo, hierárquico e sem perdas. Depois de calcular o movimento relativo entre o bloco atual e o anterior, o algoritmo decide entre codificar apenas a diferença entre os blocos, ou simplesmente codificar um novo bloco. É o que se chama de codificação INTER-FRAME e INTRA-FRAME, respectivamente. Então, os dados são transformados em coeficientes pela DCT, quantizados e comprimidos

[5]Do inglês "backward–compatible", isto pressupõe que as funcionalidades das versões anteriores serão compatíveis com a versão em questão.

por Huffman. Explora características da visão humana no descarte de informação, pois a codificação JPEG é com perda, mas uma perda seletiva e regulável [7].

A norma JPEG está hoje largamente difundida quer através de hardware quer através de software destinados a maioria das plataformas de trabalho disponíveis. Entretanto este padrão já tem seus dias contados, está apenas aguardando o lançamento e propagação da norma JPEG 2000.

3.2.7 ISO/MPEG - 1

O MPEG (Moving Pictures Experts Group) é um grupo de trabalho operando com ISO e IEC. Desde o início de suas atividades em 1988, MPEG tem produzido padrões extensamente utilizados, como o ISO/IEC 11172 e o ISO/IEC 13818, mais conhecidos por MPEG 1 e MPEG 2 (Próxima Seção), respectivamente. O padrão de compressão de vídeo MPEG - 1 foi um projeto da organização ISO/IEC JTC1 e foi aprovado em 1993 [55].

A norma MPEG 1 surgiu como resposta à crescente procura de uma norma para gravação digital de vídeo. O principal suporte de gravação considerado foi o CD-ROM tendo a norma sido otimizada para taxas totais (audio e vídeo) de aproximadamente 1.5 Mbit/s. Esta norma surge pouco depois da norma H.261 e, naturalmente, baseia-se no mesmo esquema de codificação híbrida já utilizado.

A necessidade de uma nova norma prende-se aos diferentes requisitos das aplicações de gravação face aos da videotelefônia e da videoconferência, nomeadamente em termos do atraso inicial permitido e das facilidades de gravação desejadas, por exemplo o acesso aleatório e o avanço e o recuo rápidos. Estes requisitos levam a uma gestão temporal mais rígida das ferramentas de codificação, por exemplo as facilidades de codificação exigem periodicamente imagens codificadas sem exploração de redundância temporal (âncoras), mas também o uso da compensação de movimento utilizando imagens futuras a custa de atraso inicial adicional, o que era permitido na videotelefonia e videoconferência, já que o atraso é crítico em aplicações executadas em tempo real [7].

É importante ressaltar que, enquanto a norma H.261 se destina a aplicações que requerem codificação em tempo real, a norma MPEG 1 é utilizada normalmente para gravação, por exemplo em CD-ROM. Isto permite que a codificação não seja realizada em tempo real, podendo manipular cuidadosamente a distribuição dos recursos (bits) disponíveis, aumentando com isso a qualidade subjetiva final e a taxa de compressão (Capítulo 2.2). Esta norma usa como resolução típica, imagens não entrelaçadas com resolução espacial CIF, a 25 Hz.

O objetivo de qualidade subjetiva desta norma é uma qualidade semelhante ou superior à oferecida pelas fitas cassetes VHS. Saliente-se que esta norma de sistema, definindo nomeadamente a multiplexagem e a sincronização dos sinais de vídeo e áudio. A codificação de áudio pode ser feita segundo vários modos, usando taxas entre 32 e 448 kbit/s e um ou dois canais [7].

Em termos de suas características técnicas, a norma MPEG - 1 adiciona os conceitos de quadros preditados bidirecionalmente (também chamados de B- quadros) e movimento de meio pixel[6] (do inglês *"Half pixel motion"*) [55].

A norma MPEG 1, que se destinava inicialmente à gravação em CD-ROM, é hoje aplicada noutros contextos, com particular relevância para o video *on–demand* e o acesso a bases de dados multimídia [56].

3.2.8 ISO/MPEG 2 ou ITU-T H.262

Como se viu na seção anterior, a norma MPEG 1 destina-se a aplicações onde o número de usuários simultâneos da informação é muito limitado, o que influencia no resultado da qualidade pedida. Quando terminaram os trabalhos relacionados com a norma MPEG 1, tornou-se evidente a necessidade, a possibilidade e até a inevitabilidade de dar o passo seguinte ou seja, a especificação de uma norma de codificação para a televisão digital ou seja para sinais audiovisuais digitais destina-

[6]O conceito de movimento de meio *pixel* foi originalmente proposto durante o desenvolvimento do padrão H.261, entretanto, aparentemente foi considerado como sendo muito complexo para a época.

dos a ser difundidos para um elevado número de usuários [7].

A norma MPEG 2 usa basicamente as mesmas ferramentas de codificação da norma MPEG 1 tendo objetivos de qualidade mais exigentes e logo requerendo taxa de bits por pixel mais elevados. O núcleo do algoritmo MPEG 2 é um esquema de codificação híbrido DCT/compensação de movimento.

A norma ISO/MPEG - 2 (IS 13818 -2 / ITU-T H.262) forma o coração da transmissão de dados com qualidade para televisão digital, definida tanto para a televisão de definição padrão, quanto para a televisão de alta definição (SDTV e HDTV). Esta norma foi um projeto oficial desenvolvido pelo grupo formado pela união de duas organizações o ISO/IEC JTC1 e ITU-T, entretanto, é comumente referida como sendo uma norma ISO. A norma MPEG -2 foi completada em 1994, e taxa de bits por pixel oscila entre aproximadamente 4 -30 Mbits/s [55].

Inicialmente a norma MPEG 2 tinha como objetivo especificar a codificação de sinais de vídeo para taxas de 4 até 10 Mbit/s, para resolução ITU-R 601. No entanto, a evolução dos métodos de codificação de sinais de vídeo para estender a sua aplicação a formatos de imagem e taxas de codificação mais elevados, inicialmente não foram previstos. Durante 1993, foram apresentados alguns estudos (realizados nos laboratórios da emissora de televisão BBC de Londres) que demostraram a possibilidade de codificar vídeo de alta definição, com boa qualidade, a cerca de 25 Mbits/s o que explica a inexistência de uma norma MPEG 3 para vídeo, que deveria servir a alta definição digital [7].

A norma MPEG-2 está organizada segundo perfis e níveis: cada perfil está associado a um conjunto de ferramentas de codificação, sendo este conjunto sempre um subconjunto das ferramentas de codificação do perfil seguinte (existem quatro perfis); a cada nível está associada uma dada resolução e logo uma dada taxa de bits por pixel e uma certa complexidade, por exemplo em termos de memória. Cada combinação perfil-nível é uma solução de codificação com características diferentes, devendo o usuário escolher a combinação que melhor se adapta à aplicação em causa. A norma MPEG - 2 para áudio considera, como a norma MPEG - 1, taxas entre 32 e

448 kbit/s, mas permite a utilização de um a cinco canais de áudio [4].

Na área do vídeo digital, a norma MPEG - 2 é hoje reconhecida como o novo consenso em termos de representação, existindo já inúmeros canais (sobretudo via satélite) a transmitir segundo esta norma, quer nos Estados Unidos quer em países da Europa. Espera-se assim que, a longo prazo, desapareçam os sistemas analógicos ainda utilizados até hoje - NTSC, PAL e SECAM, e os semi-analógicos de alta definição, MUSE (Japonês) e HD-MAC (Europeu). Com base nesta norma, o mundo das comunicações audiovisuais assiste a uma verdadeira corrida quer das indústrias, produzindo chips, *software* e terminais, quer das operadoras, definindo novos serviços com particular incidência na televisão digital via cabo, rádio ou satélite ou no acesso a bases de dados multimídia. Vários serviços já estão disponíveis hoje ao público [4].

3.2.9 ISO/MPEG - 4

Diante do exposto nas seções anteriores e resumido na Tabela 3.1, podemos ver que os padrões de codificação cobrem praticamente toda a faixa de taxa de bits por pixel seja para transmissão ou não, e vários tipos de aplicações, não sobrando nenhum serviço muito relevante sem padronização.

Algumas pesquisas estão sendo realizadas hoje, no que poderá ser a televisão do futuro. Destacamos aqui duas tendências muito fortes: televisão iterativa e a integração de funcionalidades em um único terminal [56].

Com a baixa taxa do MPEG - 2, na mesma largura de banda onde passava um canal analógico, pode-se passar mais de um digital, é o chamado "milagre da multiplicação dos canais". Então a questão é: o que fazer com esses canais que ficaram disponíveis? Uma opção seria por mais canais, ou seja, mais do mesmo, mais do que já se tinha! A outra seria aproveitar a oportunidade e modificar o velho modelo de serviços da televisão, impossibilitado durante tantas décadas pelo saturamento do sinal analógico. Nasce então a televisão iterativa, onde o usuário terá mais poder de escolha. Por exemplo, televisão em várias línguas, o telespectador é que escolherá a

que melhor lhe convier. Ou então, a mesma cena sob vários ângulos, um ângulo seria enviado em cada canal. Ou ainda, em transmissões esportivas teríamos um canal com um delay de por exemplo 30 segundos, para o usuário ter a oportunidade de rever um lance, e outro só com os melhores lances em câmera lenta.

Uma outra pesquisa já bem mais avançada seria a integração das funcionalidade de Informática/Computador, Telecomunicação/Telefone, Televisão/Entretenimento num único terminal a receber bits, uma vez que todos trabalham com sinais digitais, isso seria um futuro possível. Seria, pois nenhuma das normas referidas anteriormente prevê esta situação, que necessita de um alto grau de compressão das imagens. Precisamos então de uma nova forma de representação da informação visual, que não as até agora sendo utilizadas (uma imagem é formada por uma seqüência de linhas), uma representação mais complexa, mais parecida com a maneira como nós vemos uma imagem. Surge então a necessidade de se esquematizar outra norma de compressão de imagem e vídeo: MPEG-4 [4].

Os trabalhos para o desenvolvimento da norma ISO/MPEG - 4 começaram oficialmente em setembro de 1993, num encontro em Bruxelas, e estava previsto para terminar em dezembro de 2000. O organismo responsável por este desenvolvimento é o ISO/IEC SC29 WG11, e o está fazendo em duas partes, chamadas de versão 1 e versão 2 do MPEG - 4. A aprovação final da versão 1 aconteceu em janeiro de 1999 (com conteúdo técnico aprovado em outubro de 1998), e a aprovação final da versão deverá acontecer em dezembro de 2003.

Nesta norma a imagem será formada por objetos/regiões 2D ou 3D, com formas arbitrárias, as quais está associado um certo comportamento no espaço e no tempo. A representação da informação visual através da composição de vários objetos acessíveis de modo independente vai permitir uma variedade de novas funcionalidades, nomeadamente [55]:

- Cada objeto poderá ser codificado com o método de codificação mais adequado ao seu tipo de dados, por exemplo, imagens naturais, gráficos, texto, etc.

Tabela 3.1: Principais características das normas internacionais de codificação de imagem e vídeo. Nota: Os serviços e as taxas referidos são apenas os mais significativos

Norma	Serviços/Aplicações	Taxa	Técnicas de Codificação
ITU-R 601 (1982)	Televisão em estúdio	216 Mbit/s	PCM
ITU-T H.120 (1984)	Videoconferência	2 Mbit/s	DPCM Codificação entrópica
ITU-T H.261 (1990)	Videotelefonia e Videoconferência	p x 64 kbit/s p= 1, ..., 30	DPCM Transformada DCT Compensação de movimento Codificação entrópica
ISO / JPEG (1990)	Fotografia	-	Transformada DCT Codificação entrópica
ISO / MPEG-1 (1991)	Gravação em CD-ROM	1.5 Mbit/s	DPCM Transformada DCT Compensação de movimento Codificação entrópica
ISO / MPEG-2 ITU - T H.262 (1993)	Televisão de média e alta definição Gravação digital de vídeo	> 2 Mbit/s	DPCM Transformada DCT Compensação de movimento Codificação entrópica
ITU - T H.263 (1995)	Videotelefonia na rede analógica	< 64 kbit/s	DPCM Transformada DCT Compensação de movimento Codificação entrópica

- Cada objeto poderá ser ou não transmitido consoante os recursos disponíveis (por exemplo, banda ou capacidade computacional) e a sua relevância no contexto da cena composta.
- Cada objeto poderá ser codificado com mais ou menos qualidade e proteção contra erros de canal consoante a sua relevância no contexto da cena composta.
- Cada objeto poderá ser codificado com a resolução espacial e temporal mais adequada às suas características intrínsecas.
- Cada objeto poderá ser individualmente acedido, reutilizado e manipulado.
- Uma cena pode ser definida a partir de vários objetos previamente disponíveis ou a partir de objetos segmentados numa seqüência convencional.

Estas funcionalidades associadas à representação de vídeo baseada em objetos permitem não só aumentar a eficiência da codificação mas também oferecer novas formas de interatividade e o acesso universal à informação através de terminais e redes com recursos variados.

Vamos estudar a Figura 3.4: temos a imagem final composta por 2 objetos (mapa meteorológico e a apresentadora). Cada objeto pode ser comprimido de acordo com suas próprias características e enviados em canais separados, e o usuário pode decidir se quer ver só o mapa e apenas escutar a apresentadora ou ambos. Ou ainda, pensando em maior interatividade, enviar vários apresentadores em vários diferentes canais, cabendo ao telespectador escolher qual o de sua preferência [7]!

3.2.10 Uma breve Visão do Futuro dos Projetos de Padronização de Codificação

Os órgãos internacionais responsáveis pela padronização das normas de compressão / codificação, estão pesquisando novas normas constantemente. Podemos citar como exemplo três normas, que são abaixo citadas:

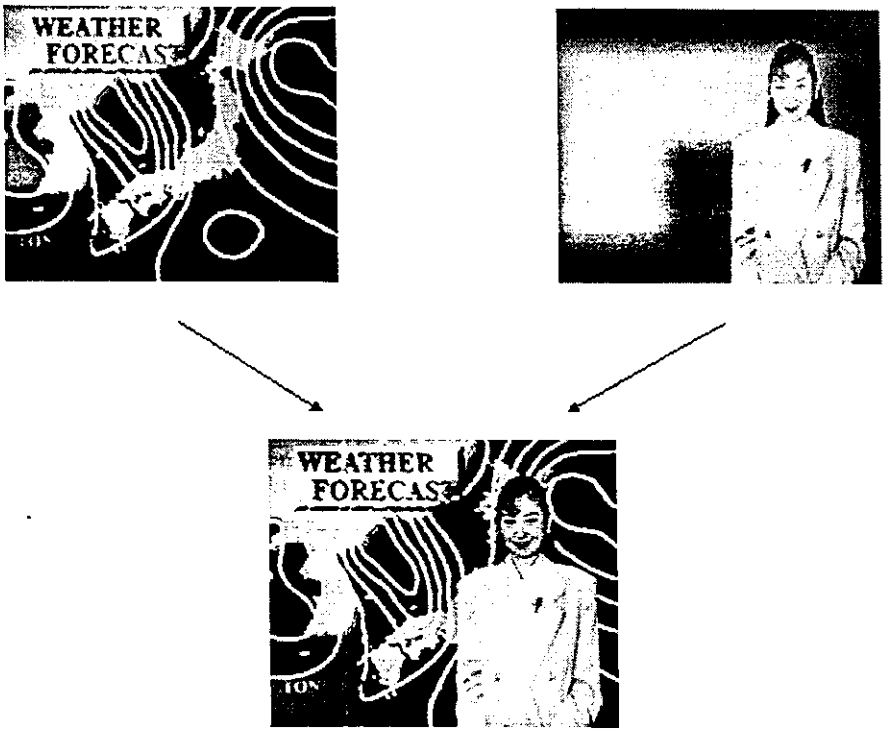


Figura 3.4: Seqüência composta a partir de objetos previamente disponíveis.

- Projeto H.263++ é considerado uma adição de mais melhorias opcionais ao H.263 e está sendo esperado para 2003. Este é um projeto do orgão ITU - T Advanced Video Coding Experts Group (SG16/Q15) [56].

- ITU - T H.26L: Uma futura geração de padrões de codificação / compressão de vídeo com melhorias na eficiência da compressão e resistência a erros. A H.26L está atualmente esquematizada para ter sua versão final aprovada no ano de 2003. Este também é um projeto do orgão ITU - T Advanced Video Coding Experts Group (SG16/Q15) [56].

- ISO/JPEG 2000: Este padrão promete taxas de compressão muito melhores que a de seu irmão JPEG. Permite a extração de diferentes níveis de resolução e regiões de interesse. Os trabalhos para este padrão começaram em 1996 pelo grupo ISO/IEC JTC1/SC29/WG1, mas comumente referenciado com simplesmente WG1. Este método realiza a compressão com ou sem perdas. Utiliza a

transformada *wavelet*, subdividindo a imagem em 3 níveis [34].

3.2.11 Conclusão

A grande aceitação da normalização não está só na necessidade de regular o mercado, mas também no modo particularmente inteligente como essas normas foram pensadas. Elas não são rígidas ao ponto de estabelecer exatamente como se tem que fazer, o que deixaria pouca ou nenhuma margem à criatividade e concorrência das empresas, o que significaria a quebra das pequenas. Estas normas apenas definem a sintaxe da codificação e o processo de decodificação. A compatibilidade é garantida através do respeito à sintaxe e pela uniformidade do processo de decodificação. Mas as empresas podem concorrer ao desenvolverem algoritmos mais ou menos eficientes, inteligentes ou complexos de detecção de movimento. Além de que essas normas não penetram nos campos de pré- processamento, filtragem e pós-processamento [56].

Capítulo 4

Tritree

O objetivo deste trabalho é o desenvolvimento de uma metodologia completa para decomposição de imagens reais bidimensionais, em tons de cinza, em regiões triangulares homogeneas, gerando a estrutura conhecida como *Tritree*. Esta decomposição é aplicada para compressão de imagens.

A meta essencial desta decomposição é descobrir e classificar regiões triangulares da imagem de acordo com seu grau de homogeneidade. Estruturas do tipo *Tritree* são parecidas com estruturas *Quadtree*, que são amplamente utilizadas por muitos algoritmos de compressão e processamento de imagens, conforme foi visto na Seção 3.1.4. Existem diferenças entre as duas estruturas e em como gerenciá-las, mas a principal diferença se deve ao fato que na primeira estrutura, a *Tritree*, as regiões da imagem a classificar são triangulares, enquanto que na segunda estrutura, a *Quadtree*, as regiões são quadrangulares. As estruturas *Tritree* serão explicadas, detalhadamente, mais adiante no texto.

O método de decomposição de imagens chamado de *Tritree* (TT) aqui desenvolvido, utilizou como base o trabalho de Wille [13] para a divisão sucessiva de uma imagem em regiões triangulares. Chega-se, assim, a um conjunto de regiões que possuem os seus valores de pixels aproximadamente iguais, ditas regiões homogêneas.

Nas próximas seções vamos estudar, com detalhes, o método desenvolvido neste trabalho para decomposição por *Tritree*. A Seção 4.1 explica como ocorre cada uma das etapas da decomposição *Tritree*. A seção 4.2 mostra como são armazenadas as informações necessárias para a reconstrução da imagem. O processo de reconstrução da imagem é apresentado na Seção 4.3. Este Capítulo encerra-se com conclusões.

4.1 Decomposição *Tritree*

4.1.1 Árvores de pesquisa

Programas de computador são constituídos, basicamente, por dois elementos, um ativo e outro passivo: algoritmos e dados, respectivamente, estando ambos correlacionados. Os algoritmos constituem a organização das tarefas que serão executadas com o objetivo de processar os dados, de acordo com o problema proposto. Os dados são os elementos passivos e para que possam ser convenientemente manipulados pelos algoritmos, são organizados em estruturas complexas, mas de fácil manipulação, chamadas "Estruturas de Dados". A estrutura de dados pode ser entendida como a maneira pela qual a informação é organizada e acessada, normalmente determinada pela própria natureza do problema. A escolha correta de uma estrutura de dados influenciará em quase todos os aspectos de um sistema, como, por exemplo, legibilidade, complexidade, desempenho e manutenção, entre outros [57].

Árvores são estruturas de dados extremamente importantes e de larga utilização para organização de informação. Quando convenientemente ordenadas, elas permitem consultas, inclusões e exclusões de elementos, de maneira muito prática, rápida e eficiente.

Em termos intuitivos, a estrutura árvore significa que os dados estão organizados

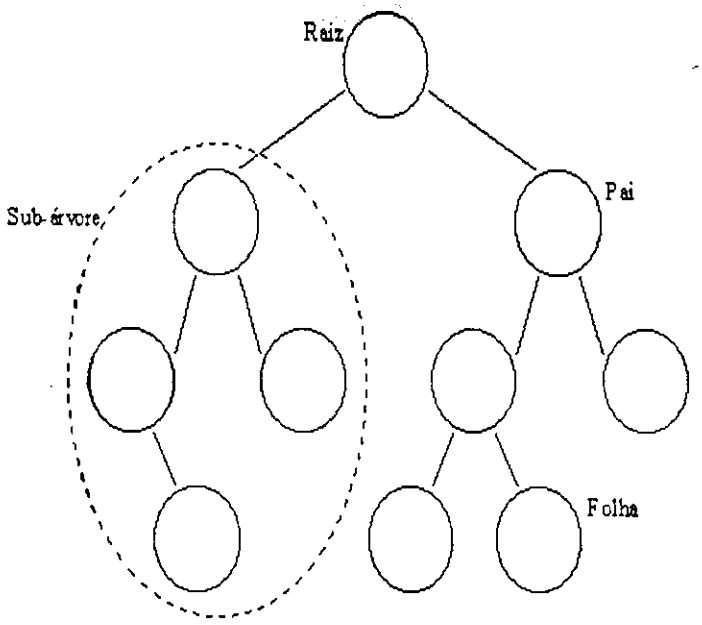


Figura 4.1: Exemplo de uma Árvore Binária

de maneira hierárquica, de forma que os diversos itens de informação estejam inter-relacionados por ramos descendentes [57]. Cada item em uma árvore consiste em um conjunto de informações determinado pela natureza da aplicação, juntamente com um elo para cada ramo imediatamente inferior hierarquicamente a ele.

Uma terminologia especial é necessária quando se trabalha com árvores. Cada item de dado é chamado de nó ou nodo da árvore e qualquer ramo é chamado de sub-árvore ou filho. O primeiro item em uma árvore ou sub-árvore é denominado raíz ou pai. Um nodo que não tem sub-árvores (filhos) ligadas a ele é chamado de nó terminal ou, simplesmente, folha. O número de filhos de um nodo é denominado de grau da árvore. O nível de um nodo é determinado pelo número de gerações percorridas da raíz até ele, sendo que a raíz está no nível 0, seus filhos no nível 1, netos no nível 2 e assim por diante. A altura ou profundidade de uma árvore é o maior nível que um nodo qualquer da árvore atingiu. Outras relações de parentesco que são admitidas em uma árvore genealógica, como avós, netos, irmão, antepassados e sucessores, por exemplo, também são admitidos em uma árvore de pesquisa [57;

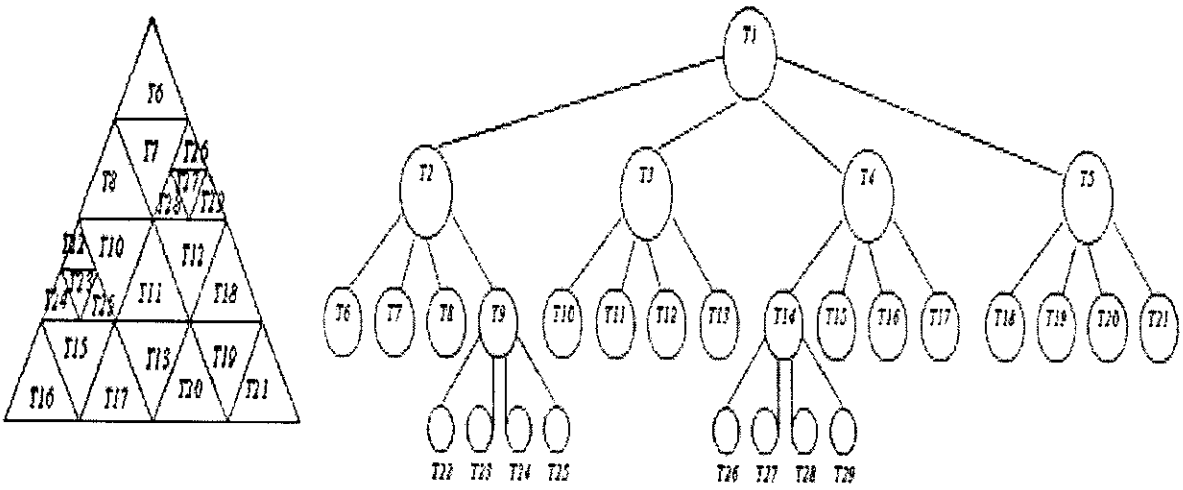


Figura 4.2: Estrutura *Tritree*

58].

A Figura 4.1 apresenta uma árvore binária, ou seja, uma árvore cujo grau máximo de cada nodo é dois.

4.1.2 Estruturas *Tritree*

A decomposição *Tritree* de uma imagem bidimensional (2D) em regiões homogêneas constrói uma árvore de pesquisa com grau 4, como a apresentada na Figura 4.2. Portanto, cada item tem (sempre, pela própria natureza do algoritmo) quatro filhos, e cada um destes filhos está associado com uma única região triangular bem definida da imagem. A raíz da árvore está associada com a imagem completa. Uma árvore *Tritree*, com n níveis de profundidade, permite a representação de até 4^{n-1} triângulos, lembrando que a raíz é o nível 0.

Para o armazenamento interno das informações de uma estrutura *Tritree*, faz-se necessário a construção de uma estrutura de dados complexa, de gerenciamento e manipulação também complexos. Esta estrutura consiste de registros com 9 campos de dados, como ilustrado na Figura 4.3. A princípio, é armazenado um registro

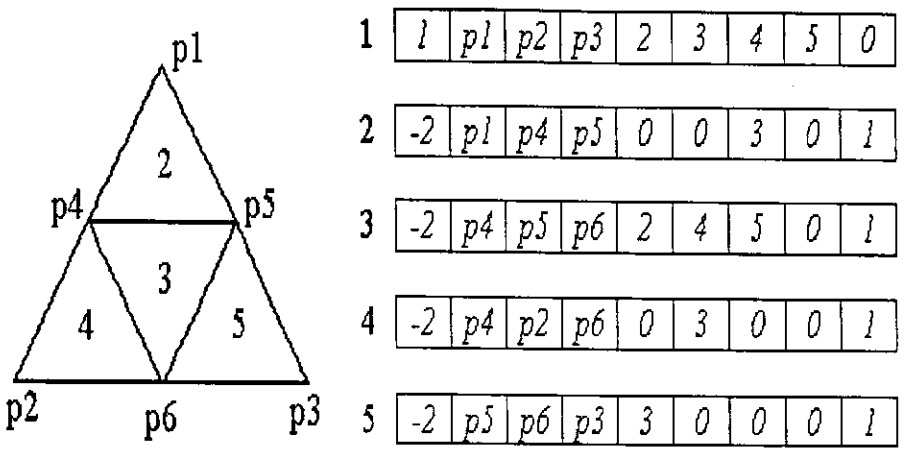


Figura 4.3: *Tritree*: Estrutura de Dados

representando cada triângulo isoladamente, independente dele ser folha ou não.

O primeiro campo de um registro da *Tritree* contém o número do nível do triângulo associado ao registro na árvore. Quando este é negativo, significa que o triângulo é terminal, ou seja, uma folha. Os três próximos campos consistem de ponteiros para os vértices do triângulo na lista de pontos (outra árvore que será explacada mais adiante no texto). Estes pontos de vértice poderiam ser facilmente calculados a qualquer momento da execução do algoritmo de compressão, mas eles são aqui armazenados para reduzir consideravelmente o tempo de execução, pois os pontos são utilizados com muita freqüência pelo algoritmo.

As 4 posições seguintes, no registro da estrutura *Tritree* são ponteiros para registros do mesmo tipo, que representam os filhos do triângulo. Quando este triângulo é uma folha, alguns destes campos podem ser utilizados para armazenar os ponteiros para seus vizinhos, o valor zero significando que o triângulo não tem vizinhos naquela direção. O último destes quatro campos sempre será nulo, pois os vizinhos considerados são sempre do mesmo nível ou menor. O último campo de registro é utilizado para armazenar o ponteiro para o registro correspondente ao triângulo pai deste na árvore. Assim, garante-se que a qualquer momento da execução sempre será possível caminhar, para cima ou para baixo da árvore, e para os lados, indiretamente.

Os pontos da imagem correspondentes aos vértices de todos os triângulos gerados também são organizados internamente em uma árvore de pesquisa. Neste caso eles formam uma árvore binária como a que foi apresentada na Figura 4.1.

Para o armazenamento interno da lista de vértices também faz-se necessário uma estrutura de dados complexa, composta por registros com 5 campos. Os dois primeiros armazenam as coordenadas espaciais do vértice na imagem enquadrada (vide próxima seção), primeiro o X que representa a coluna, e depois Y que representa a linha. O terceiro campo é reservado para o valor do nível de cinza daquele ponto da imagem. Os dois últimos campos são ponteiros para os registros dos seus dois possíveis filhos: o da direita e o da esquerda. Nesta árvore, devido a sua própria natureza, podem ocorrer itens que só tenham um filho, o que é permitido nas estruturas de árvores de pesquisa.

Quando a imagem é enquadrada no triângulo inicial, os vértices deste são calculados e inseridos na árvore binária que, até então, estava vazia. O primeiro vértice calculado passa a ocupar a raíz da árvore. Para a inserção de qualquer outro ponto, deve-se percorrer a árvore a partir da raíz até que seja encontrada a posição correta. O ponto é comparado com todos os vértices já armazenados, a partir da raíz, para saber se ele é maior ou menor que este. Se for considerado maior que o ponto testado, segundo os critérios abaixo, então pesquisa-se o nodo à direita, caso contrário pesquisa-se o nodo da esquerda. Um ponto com coordenadas x e y, é definido como sendo menor que um ponto de coordenadas u e v respectivamente, se:

$x < u$ *ou*

$x = y$ *e* $y < v$

O novo ponto é inserido quando não for mais possível descer na árvore na direção indicada pelo teste acima, ou seja, quando o ponto pesquisado não tiver filho naquela direção. O novo ponto, então, ocupa a posição de filho do último ponto pesquisado, à direita se for menor ou à esquerda se for maior que este.

Toda vez que um triângulo é dividido, o ponto médio de cada aresta é calculado e pesquisado na árvore de vértices para averiguar se ele já existe, o que pode acontecer

se os seus vizinhos tiverem o número do nível maior que o dele. Caso o ponto já exista na árvore binária nada acontece, senão, ele deve ser adicionado à mesma. A árvore binária de vértices é ordenada de forma crescente, da esquerda para a direita, com base nas coordenadas x e y dos vértices, nesta ordem.

Este tipo de representação de dados exige muito espaço na memória. A cada triângulo dividido tem-se que alocar espaço para mais quatro registros de triângulo, e talvez, espaço para mais três registros de pontos.

4.1.3 Enquadrando a Imagem

Para o início do funcionamento do método de compressão, é necessário que seja feito o enquadramento da imagem toda em um triângulo equilátero inicial que representará a raíz da árvore *Tritree*. Para facilitar o processamento, o desejável é que a base do triângulo e a base da imagem sejam coincidentes e que a base da imagem esteja centrada em relação à base do triângulo. É desejável também que os dois cantos superiores da imagem tendam a tangenciar as arestas do triângulo, ou seja, as dimensões do triângulo - a base e a altura - são calculadas em função das dimensões da imagem - número de linhas e número de colunas. Assim, é garantido que o triângulo será suficientemente grande para comportar todo o perímetro da imagem. Às vezes ele é maior que o necessário, dependendo das dimensões da imagem, o que será explicado mais adiante. A Figura 4.4 exemplifica a disposição de uma imagem hipotética dentro do triângulo inicial.

Para se descobrir quais devem ser as dimensões do triângulo inicial serão necessários alguns cálculos a fim de realizar o enquadramento da imagem. Este cálculo é feito para cada imagem a ser comprimida.

Sendo c e l o número de colunas e linhas da imagem, respectivamente, observando a Figura 4.4 pode-se afirmar que, por semelhança de triângulos, o triângulo menor superior formado pelas três arestas de dimensão c , também é equilátero, e a sua altura a' pode ser obtida por:

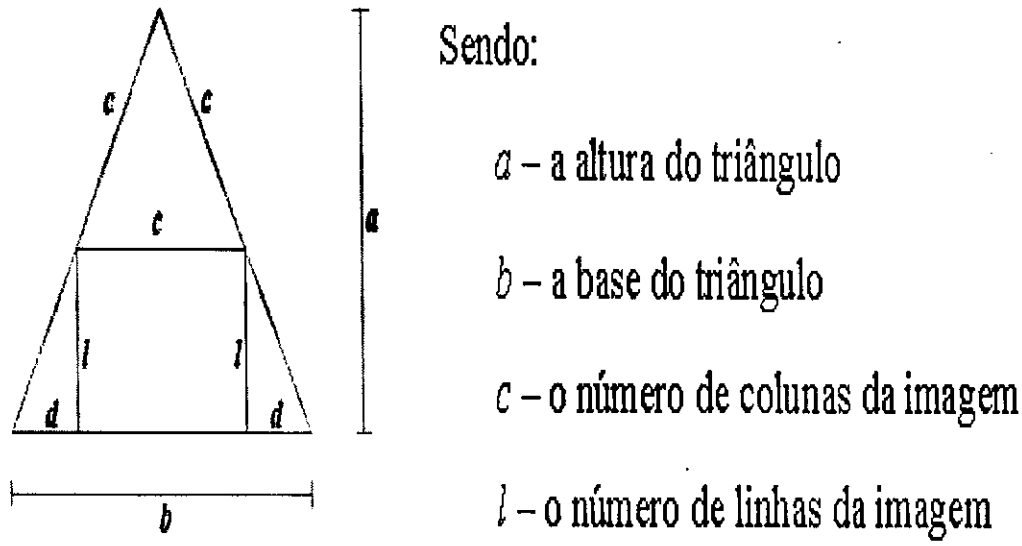


Figura 4.4: *Tritree*: Disposição Inicial

$a' = c \times sen60^o$; então, a altura a do triângulo maior é:

$a = l + c \times sen60^o$

Obviamente, a base b do triângulo pode ser expressa por:

$b = c + 2 \times d$, sabendo-se que:

$tan60^o = \frac{l}{d}$, temos:

$d = l \times cotan60^o = l \times \frac{cos60^o}{sen60^o} = l \times \frac{\sqrt{3}}{3}$

Consequentemente temos:

$b = c + 2 \times l \times \frac{\sqrt{3}}{3} = c + \frac{2\sqrt{3}}{3} l$

Os valores de a e b provavelmente serão fracionários. Como estamos trabalhando com imagens e não existem pixels com coordenadas fracionárias, adotamos os tetos da altura e da base. Contudo, isso não é suficiente para garantir que todos os valores serão inteiros, pois mesmo a base b tendo um valor inteiro, d pode assumir um valor fracionário, tornando-se necessário incrementar de 1 a base b até que o valor da expressão $(b - c)/2$ seja um inteiro. Como conseqüência, o triângulo aumenta progressivamente suas dimensões, fazendo com que os cantos superiores da imagem original muito provavelmente não tangenciem as arestas do triângulo

inicial.

Internamente, quando da implementação, o que temos é uma matriz cujas dimensões são os valores finais de a e b. Cada pixel, $I(x, y)$, da imagem a ser comprimida na matriz, passará a ter coordenadas $I(x + d, y + a - l)$ na matriz. Àqueles elementos da matriz que não pertencem à imagem (isto inclui os que não pertencem sequer ao triângulo) é atribuído o valor negativo do nível de cinza máximo da imagem, para não influenciar na decisão de dividir ou não um triângulo, quando comparado com o limiar que será discutido na próxima seção.

4.1.4 Triangulação

No início da execução do método desenvolvido é perguntado ao usuário qual o limiar de decisão que deverá ser utilizado no teste de homogeneidade (vide seção 4.1.5).

O método é flexível quanto ao número de linhas e colunas das imagens que serão comprimidas, ou seja, ele funciona corretamente para quaisquer números de linha e coluna. O método começa contando automaticamente o número de linhas e colunas da imagem de entrada, não sendo necessário que o usuário do método saiba, a priori, as dimensões desta. Em outras palavras, não é necessária a supervisão humana.

De posse dos valores dos números de linhas e colunas, o enquadramento da imagem original no triângulo inicial pode ser realizado, calculando-se as dimensões deste triângulo, isto é, a sua base e a sua altura. A inicialização das árvores utilizadas pelo processo é feita, o triângulo inicial ocupa a raíz da *Tritree* e seus três vértices são inseridos na árvore binária de pontos. Na seqüência, a matriz que contém tanto o triângulo quanto a imagem é criada na memória, com as mesmas dimensões, já calculadas, do triângulo inicial.

A primeira divisão, a do triângulo inicial, acontece sem a execução do teste de homogeneidade. Como a imagem corresponde a um retângulo dentro do triângulo inicial, existem regiões deste triângulo que não contêm pixels da imagem. Estas

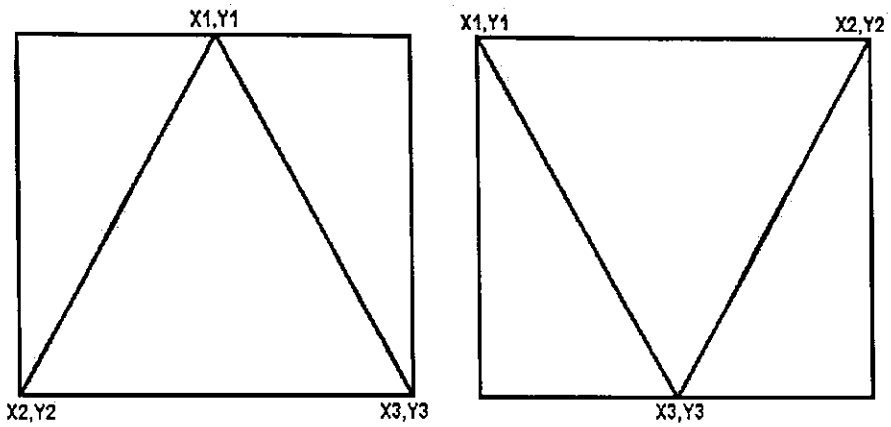


Figura 4.5: *Tritree*: Retângulo Contendo um Triângulo de Dimensões Exatas

regiões devem ser identificadas para serem descartadas ao final do processamento, pois são irrelevantes para a reconstrução da imagem comprimida.

Após este início, o restante do processo é recursivo e chamado de triangulação, sendo executado da mesma forma para cada um dos quatro filhos do triângulo pai, o qual é recebido como parâmetro. Começa-se testando se o triângulo atual pode ser subdividido. Para tanto, é preciso descobrir um retângulo dentro da matriz que contenha exatamente o triângulo testado, ou seja, este retângulo tem que ter necessariamente as mesmas base e altura do triângulo, como aparece na Figura 4.5.

Analisando-se a Figura 4.5, percebe-se que o retângulo na matriz é delimitado exatamente pelo menor e maior valores das coordenadas X do triângulo na direção horizontal. Verticalmente, este retângulo está delimitado pelo menor e maior valores de Y do triângulo, independente de seu sentido.

Delimitado o retângulo que contém o triângulo, o próximo passo é descobrir os *pixels* do retângulo que também estão contidos no triângulo. O sistema de equações utilizado para descobrir se um ponto pertence ao triângulo é dado a seguir [13]:

$$\begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \\ L_3 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Resolvendo o sistema, obtém-se:

$L_1 = a_1x + b_1y + c_1$,

$L_2 = a_2x + b_2y + c_2$,

$L_3 = a_3x + b_3y + c_3$,

Em que

$a_1 = \frac{y_2 - y_3}{2A}$; $a_2 = \frac{y_3 - y_1}{2A}$; $a_3 = \frac{y_1 - y_2}{2A}$;

$b_1 = \frac{x_3 - x_2}{2A}$; $b_2 = \frac{x_1 - x_3}{2A}$; $b_3 = \frac{x_2 - x_1}{2A}$;

$c_1 = \frac{x_2y_3 - x_3y_2}{2A}$; $c_2 = \frac{x_3y_1 - x_1y_3}{2A}$; $c_3 = \frac{x_1y_2 - x_2y_1}{2A}$;

Além do que se verifica a propriedade fundamental das coordenadas de área L_1, L_2 *e* L_3 , isto é

$L_1 + L_2 + L_3 = 1$

A área do triângulo de vértice (x_1, y_1), (x_2, y_2) *e* (x_3, y_3) é dada por:

$$A = \frac{1}{2} \begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{bmatrix}$$

Os valores x_k e y_k são as coordenadas do vértice V_k para $k = 1, 2\ e\ 3$, x e y são as coordenadas do ponto que se quer descobrir se pertence ou não ao triângulo. Resolvendo este sistema, chega-se aos valores de L_1, L_2 *e* L_3. O ponto estará contido no triângulo se $0 < L_k < 1$, para $k = 1, 2\ e\ 3$.

Uma vez que os triângulos e seus pontos são conhecidos, aplica-se o teste de homogeneidade em cada um destes afim de decidir sobre sua divisão ou não. Detalhes sobre o teste de homogeneidade serão dados mais adiante no texto.

Caso o teste de homogeneidade falhe, isto é, o triângulo não seja suficientemente homogêneo, ele será dividido. A etapa da divisão começa com o cálculo das coordenadas dos pontos médios de cada aresta do triângulo, para a geração dos vértices dos quatro triângulos filhos. Sendo estes valores fracionários considera-se, respectivamente, apenas a parte inteira (piso). Como os pontos médios são as médias aritméticas simples das coordenadas dos vértices, é necessário verificar se seus valores e os das coordenadas são iguais. Caso isso aconteça, significa que, devido às

suas dimensões o triângulo não pode mais ser dividido, o que provoca o término da etapa de divisão. Quando o triângulo passa no teste acima, tanto os pontos médios quanto os quatro triângulos já gerados são inseridos na árvore.

O processo de triangulação pode terminar de duas maneiras diferentes: ou quando nenhum triângulo puder mais ser dividido, devido às suas dimensões, ou quando o teste de homogeneidade atribuir status de folha para todos os triângulos filhos testados.

4.1.5 Teste de Homogeneidade

Foram utilizados três testes de homogeneidade diferentes. No primeiro, descobre-se para cada triângulo os *pixels* de maior e menor valores de nível de cinza. O teste consiste em comparar a diferença entre estes *pixels* com um determinado limiar de decisão. Se a diferença for menor ou igual ao limiar, o triângulo é considerado homogêneo, não sendo dividido, e este passa a representar uma folha na árvore *Tritree*. Caso contrário, o triângulo deve ser dividido e o mesmo procedimento é repetido para cada filho seu, e assim por diante.

Contudo, uma forma alternativa de se realizar o teste de homogeneidade é pelo desvio da média, conforme proposto por Bárbara e Alcaim [40], em cujo trabalho foram utilizadas *Quadtrees* para o algoritmo de compressão. Entretanto, o método é facilmente aplicável para o caso das *Tritrees*. A decisão de divisão ou não do triângulo é feita através da comparação dos valores de nível de cinza de cada *pixel* do triângulo com o valor médio destes níveis de cinza. Após calculada a média, analisa-se o módulo da diferença entre essa média e o valor de cada *pixel*. O triângulo é classificado a partir desta análise, comparando-as com um determinado limiar dado. Se as diferenças forem todas menores do que um dado limiar, o triângulo é caracterizado como homogêneo, e marcado como folha na árvore da *Tritree*. Caso contrário ele será dividido.

O último tipo de teste de homogeneidade testado experimentalmente foi uti-

lizando o desvio padrão da média (DP). A estatística descritiva descreve o desvio padrão como sendo uma medida do quanto os dados individuais de uma população (no nosso caso, os *pixels* pertencentes a um determinado triângulo) desviam-se da média [59]. Ou seja, quanto maior/menor o valor calculado para o DP, podemos afirmar que maior/menor é a variabilidade entre os dados pesquisados. Sendo o valor do desvio padrão igual a zero, não existe variabilidade entre os dados. Após o cálculo da média (M) dos valores de nível de cinza e do DP de um triângulo, são calculados os limites superior (*Lim_Sup*)e inferior (*Lim_Inf*) do mesmo, como se segue:

$Lim\_Sup = M + DP$

$Lim\_Inf = M - DP$

É considerado homogêneo o triângulo cujos valores de nível de cinza dos *pixels*, pertençam todos ao intervalo gerado pelos limites inferior e superior. Satisfazendo esta condição, o triângulo é marcado como folha na árvore da *Tritree*, caso contrário ele será dividido.

4.1.6 Limiar de Decisão

O limiar de decisão é escolhido de forma empírica: no inicio da execução do método é perguntado ao usuário qual o valor do limiar de decisão que será usado. Duas estratégias de emprego do limiar foram testadas: fixo e variável. Ambos podem ser utilizado nos dois tipos de teste de homogeneidade, tanto com o desvio da média, como com o que utiliza a diferença entre o menor e maior valores de nível de cinza. Na primeira forma de utilização, limiar fixo, seu valor não é alterado durante a execução da decomposição, sendo o mesmo para todos os triângulos.

Para melhorar o desempenho do seu algoritmo de compressão com estruturas *Quadtree*, Shusterman e Feder [20] sugere a utilização de diferentes valores de limiar para cada nível de resolução. Sua sugestão também foi utilizada no trabalho de Bárbara e Alcaim [40]. O emprego de limiares variáveis para o teste de homogeneidade

para o algoritmo da *Tritree* foi utilizado com poucas adaptações, como uma forma alternativa de execução do método.

A estratégia proposta de determinação de limiares variáveis, definida por Shusterman e Feder em [20], é:

$$L_k = L_i/2^{(n-k)}, 1 \leq k < n \tag{4.1}$$

Nesta equação, L_k é o valor do limiar ao nível k da árvore. O valor do limiar inicial, fornecido pelo usuário, é L_i. O nível zero é aquele que corresponde à raíz da árvore da *Tritree*, e é sempre dividido sem a realização do teste de homogeneidade. O número n representa a profundidade que uma árvore *Tritree* pode alcançar, dependendo das dimensões da própria imagem.

Pode ser observado da equação 4.1 que, subindo na árvore *Tritree*, a cada nível o limiar é menor que o limiar do nível anterior, por um fator de 2 . Como conclusão pode-se dizer que, na estratégia de limiar variável adotada, há um maior rigor na decisão de divisão, para triângulos de maior tamanho.

4.1.7 Equalização dos Vértices

Originalmente, os triângulos foram divididos até o limite imposto por suas próprias dimensões, ou seja, até que os vértices sejam pixels vizinhos ou ainda coincidentes. Neste ponto da divisão, deixamos de ter triângulos equiláteros.

Para testar a vantagem de mantermos a essência do método, ou seja, termos, até o final, apenas triângulos equiláteros, foi implementada uma modificação no método original, pela qual os triângulos só sejam divididos se seus filhos continuarem a ser triângulos equiláteros.

A adoção desta modificação resultou em melhorias significativas em termos de percentual de compressão com uma diminuição não significativa nos valores de PSNR (Seção 2.2), e nas diferenças na qualidade subjetiva, como será observado no Capítulo 5. Com isso, esta estratégia de divisão foi adotada como sendo definitiva.

4.2 Arquivo Comprimido

As informações resultantes da decomposição *Tritree*, necessárias para a reconstrução da imagem, são armazenadas em um arquivo, que é o arquivo da imagem comprimida, propriamente dita. Este pode ser armazenado e/ou transmitido. O arquivo é composto de duas partes distintas: cabeçalho e corpo. No cabeçalho se encontram os valores das dimensões da imagem original: o número de linhas e colunas.

A segunda parte do arquivo comprimido é subdividida em dois tipos de informações. A primeira é o código da *Tritree* que traz a informação sobre sua estrutura, informação esta que é necessária para sua reconstrução. O outro tipo de informação, os dados, contido na segunda parte do arquivo comprimido, é uma sequência de níveis de cinza. Dependendo da forma para a reconstrução dos triângulos, seja por interpolação linear ou replicação da média (como será explicado mais adiante no texto), estes níveis de cinza podem ser de vértices dos triângulos folhas ou o valor médio do nível de cinza dos triângulos folhas.

O procedimento de codificação da informação que será repassada ao decodificador, inclui codificar a informação da estrutura da árvore e codificar níveis de cinza, conforme será explicado nas próximas seções.

4.2.1 Codificação da Estrutura da Árvore

Para a informação da estrutura da árvore o valor '1' é utilizado para rotular os nodos pais e o '0' é reservado para rotular um nodo folha. Obviamente, todos os nodos do último nível de decomposição são folhas e, portanto, o processo de codificação da estrutura da árvore pode parar no penúltimo nível. O único nodo do primeiro nível, a raíz, será sempre um nodo pai, podendo ser suprimida, também, essa informação. O código resultante para a árvore *Tritree*, apresentada na Figura

4.2, é 1111 - 0001 - 0000 - 1000 - 0000.

É facil perceber que é possível encontrarmos longas seqüências de 0's e 1's dentro do código resultante. Com base nessa certeza, podemos lançar mão de uma forma de codificação sem perdas, conhecida por *run-lenght coding*, que tem como característica, justamente explorar longas seqüências de mesmos valores [51]. Por exemplo, sendo a seqüência de números abaixo um fragmento do código resultante de uma *Tritree* qualquer:

$$000000111110000011100000000$$

A informação necessária para sua reconstrução seria,

$$(0,6)(1,5)(0,5)(1,3)(0,7)$$

que, para ser armazenada, necessita de muito menos espaço.

4.2.2 Codificação Preditiva dos Vértices

Será visto que a imagem é reconstruída pela reconstrução dos triângulos, e estes podem ser reconstruídos de duas formas diferentes: por interpolação linear ou por replicação da média. Para a primeira forma, a informação dos níveis de cinza dos vértices pode ser armazenada, utilizando-se codificação preditiva sem perdas (Seção 2.1.1). Um ponto de vértice, o primeiro a ser calculado, é escolhido para ser o preditor e o único a ser codificado com 8 bits (para o caso de imagens com 256 níveis de cinza em sua grade). Para os outros pontos de vértice, somente a diferença entre estes e o preditor é codificada, com o mesmo número de bits do limiar.

4.3 Reconstrução das Imagens

A reconstrução das imagens acontece realizando-se a reconstrução dos triângulos folhas da *Tritree* que contêm *pixels* pertencentes à imagem original. Foram

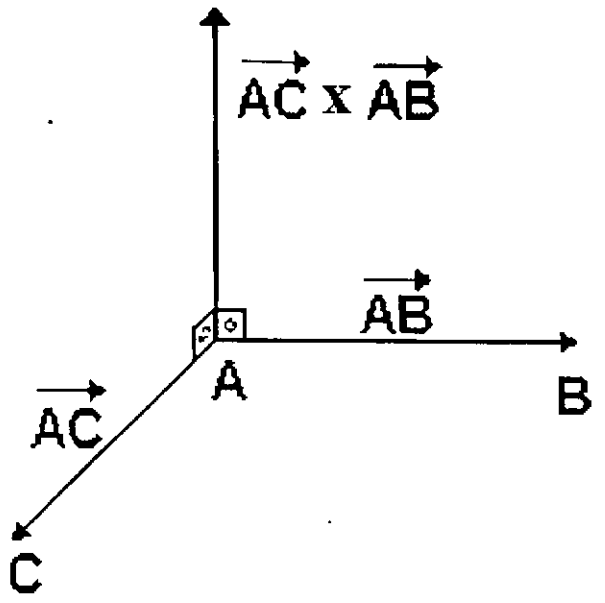


Figura 4.6: Os três pontos de vértice de um triângulo A, B e C, definindo um plano e o vetor normal a este plano.

pesquisadas duas formas de reconstrução: replicação da média e interpolação linear. Na primeira, ocorre replicação do valor médio de níveis de cinza dos triângulos. Ou seja, é armazenado apenas um valor, representando a média dos valores de nível de cinza de cada triângulo.

A outra forma de se realizar a reconstrução da imagem comprimida é por interpolação linear entre os vértices de cada triângulo relevante da *Tritree*. Esta interpolação é feita de acordo com a equação abaixo, que é a equação cartesiana do plano [60]:

$$ax + by + cz = d \tag{4.2}$$

Os parâmetros a, b, c e d da equação são determinados em função das coordenadas dos vértices do triângulo e seus valores de nível de cinza. O valor de z é o nível de cinza interpolado para os pixels com coordenadas (x, y).

Tendo-se os três vértices de um triângulo, $\vec{A}$, $\vec{B}$ e $\vec{C}$, para descobrir a equação do plano no espaço que passa pelos três pontos, precisa-se de um plano e de um

vetor perpendicular ao plano como mostra a Figura 4.6 [60]. O plano é determinado pelos vetores $\vec{AB}$ e $\vec{AC}$ e o vetor perpendicular ao plano é dado pelo vetor normal entre $\vec{AB}$ e $\vec{AC}$. É importante ressaltar que não é relevante a escolha do ponto que será a origem para a formação da equação, pois esta, no final, será sempre a mesma, independente do ponto escolhido.

Sendo x_k, y_k e z_k as coordenadas cartesianas do vértice k e o seu valor de nível de cinza, respectivamente, para $k = 1,\ 2\ e\ 3$, temos que:

$$\begin{aligned} a &= (y2 - y1) \times (z3 - z1) - (y3 - y1) \times (z2 - z1) \\ b &= (x3 - x1) \times (z2 - z1) - (x2 - x1) \times (z3 - z1) \\ c &= (x2 - x1) \times (y3 - y1) - (x3 - x1) \times (y2 - y1) \\ d &= a \times x1 + b \times y1 + c \times z1 \end{aligned}$$

Com os valores dos parâmetros a, b, c e d conhecidos, a equação do plano também é conhecida e para cada ponto testado, basta substituir os valores de suas coordenadas x e y e calcular o valor de z que é o valor de nível de cinza interpolado para este ponto.

4.3.1 Abordagem Híbrida com *wavelets*

Como foi relatado por Silva [51], a maioria dos trabalhos encontrados na literatura sobre compressão faz uso de uma abordagem híbrida entre diferentes métodos, cada um explorando características próprias. A Seção 4.2 descreveu maneiras de melhorar a taxa de compressão. Agora, veremos uma tentativa de melhorar a qualidade visual da imagem reconstruída, bem como sua taxa PSNR (Seção 2.2).

Transformadas *wavelet* foi visto na Seção qlfwavelets, nesta pode ser observado que sua decomposição fornece imagens decimadas da imagem original, uma resultante de filtragem passa-baixa com as características globais, e três imagens de detalhes da original, resultantes de filtragem passa-alta, sendo uma com características verticais, outra com características horizontais e a última com características

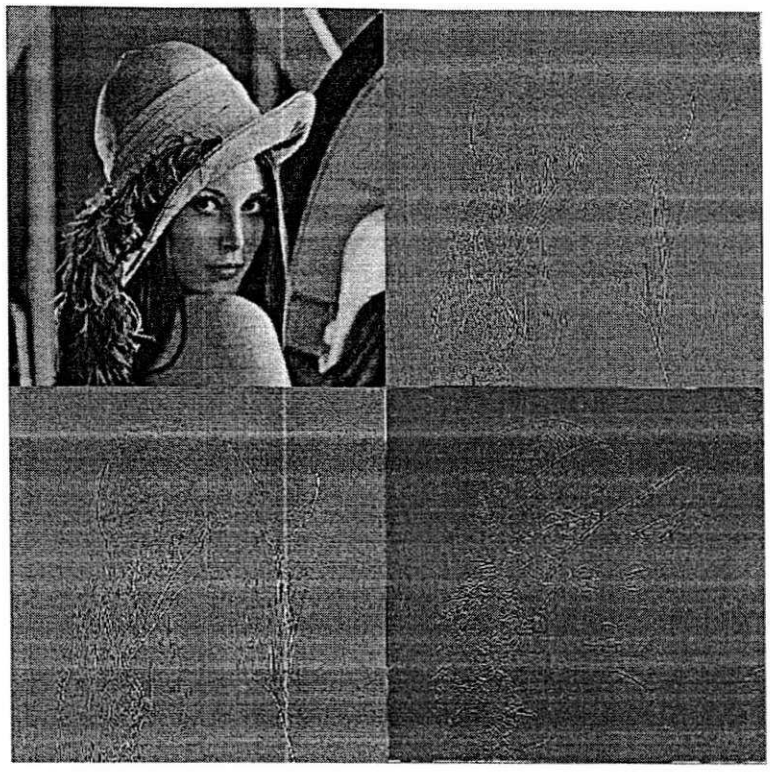


Figura 4.7: Imagem Lena de 512x512 com 1 nível de decomposição *wavelet*

diagonais. A banda com as características globais pode ser recursivamente decomposta em mais níveis de decomposição. A Figura 4.7 apresenta um exemplo de decomposição *wavelet*, aplicada à imagem "Lena" com um nível de decomposição.

Uma vantagem da decomposição *wavelet* é que ela produz um número de coeficientes igual ao número de *pixels* da imagem decomposta, o que não ocorre, por exemplo, com a decomposição Laplaciana, que pode aumentar em até 33% o número de coeficientes [28].

As *wavelets* são empregadas neste trabalho para suavizar as bordas dos triângulos das imagens reconstruídas. Estas bordas são visíveis nas imagens comprimidas com valores de limiar mais altos, como pode ser visto no próximo capítulo. A intenção é diminuir a visibilidade destes triângulos, através da redução ou eliminação das componentes de alta freqüência. Estas componentes se encontram nas três imagens de detalhes. Os testes foram realizados em cada uma destas três sub-bandas separadamente, depois realizando-se combinações entre elas. Também foram re-

alizados testes nos demais níveis de decomposição. Os coeficientes destas bandas foram alterados de duas formas diferentes, ora zerando-os ora dividindo-os pela metade.

4.4 Conclusão

Neste capítulo foi presentada todas as etapas componentes da decomposição *Tritree*. Como esta é aplicada para fins de compressão de imagens, foi apresentado também formas de armazenamento das informações passadas para o descompressor, relevantes para a reconstrução da imagem. Um pós-processamento com transformadas *wavelet*, visando a suavisação das bordas dos triângulos reconstruídos, foi introduzido. Isso faz-se necessário pois a reconstrução da imagem é feita através da reconstrução dos triângulos em que esta fora decomposta.

Capítulo 5

Resultados Experimentais

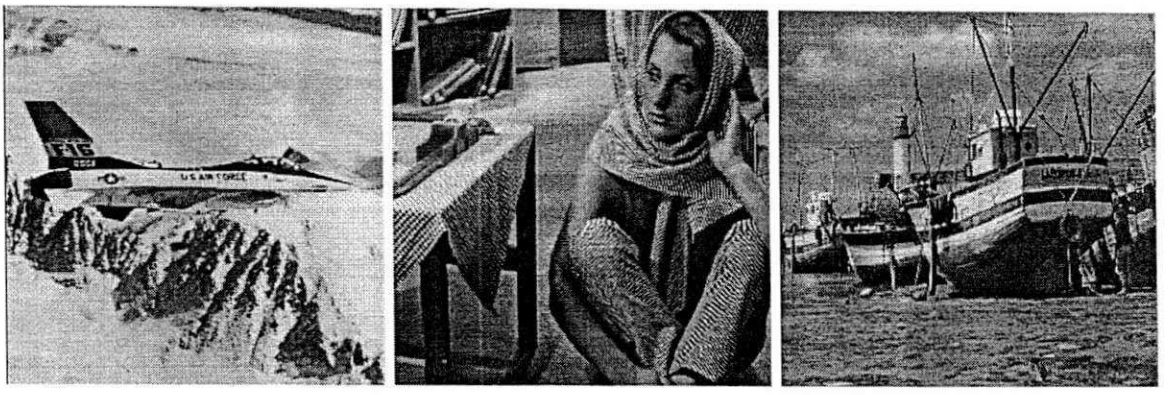
Este capítulo apresenta uma amostra dos resultados conseguidos neste trabalho, para o algoritmo da *Tritree*[1]. O algoritmo da *Quadtree* (Seção 3.1.4) foi também implementado para que seus resultados possam ser comparados diretamente com os da *Tritree*, sendo, também, aqui apresentados.

Foram utilizadas nos testes, 10 imagens reais bidimensionais, a saber: Airplane (Figura 5.1(a)), Barb (Figura 5.1(b)), Boat (Figura 5.1(c)), Frog (Figura 5.1(d)), Goldhill (Figura 5.1(e)), Gull (Figura 5.1(f)), Lena (Figura 5.1(g)), Mandrill (Figura 5.1(h)), Peppers (Figura 5.1(i)), com dimensões 256x256. A décima figura é a mesma Lena sendo que com dimensões 512x512, que são as que podem ser encontradas na literatura.

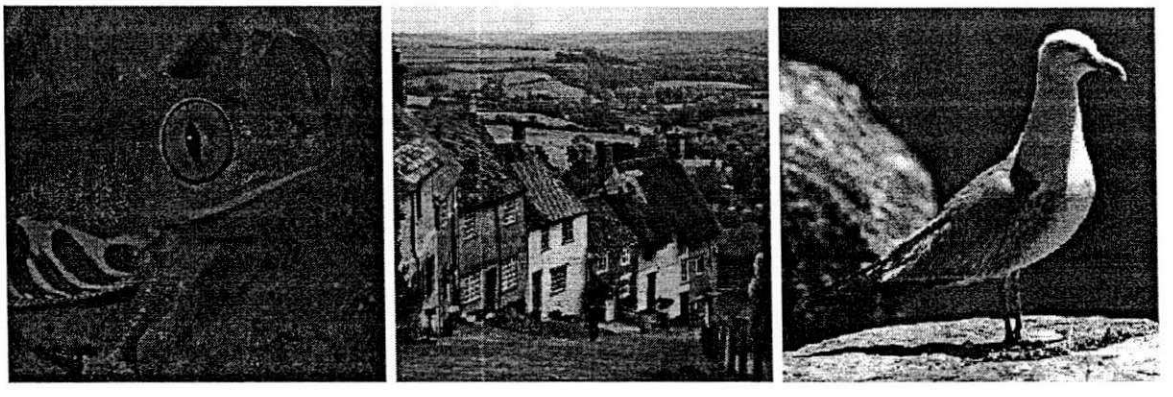
Todas estas imagens possuem 256 níveis de cinza codificadas com 8 bits por *pixel*. Estas imagens são as mesmas comumente utilizadas nos testes dos algoritmos de compressão de imagens, encontrados na literatura, garantindo, assim, a legitimi-

[1]Devido a um erro na fórmula utilizada para reconstrução por interpolação linear (Seção 4.3), identificado pelos professores componentes da banca de defesa do exame de qualificação, toda a gama de resultados gerados com esse tipo de reconstrução, tiveram que ser refeitos e reinterpretados. Assim procedendo, verificou-se que a medida PSNR foi sempre aumentada, consequentemente, todas as conclusões anteriores à correção, tiveram que ser refeitas. Mais uma vez agradeço aos mesmos pela valorosa contribuição.

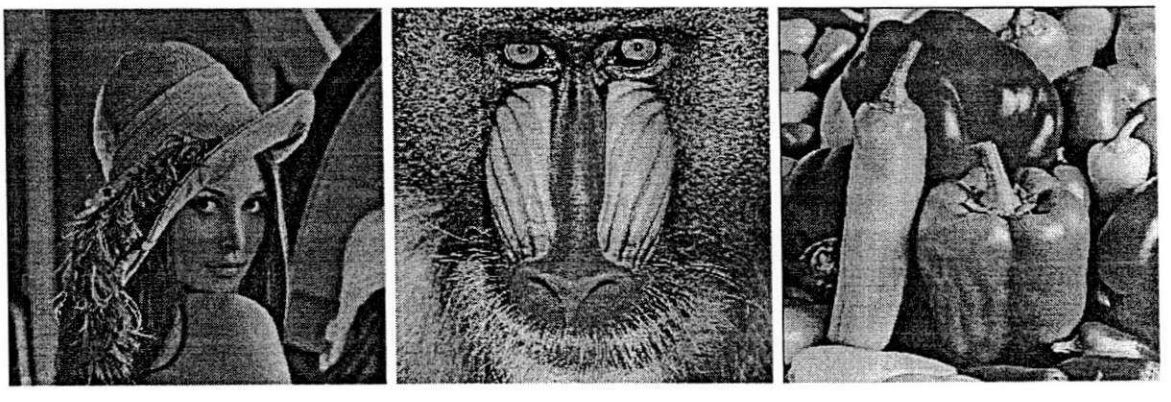
dade das eventuais comparações entre o método desenvolvido e outros. Entretanto, só serão apresentados, aqui, os resultados para a imagem "Lena" com dimensão 512x512 *pixels*, por se tratar da imagem mais utilizada.



(a) Airplane (b) Barb (c) Boat



(d) Frog (e) Goldhill (f) Gull



(g) Lena (h) Mandrill (i) Peppers

Figura 5.1: Imagens Utilizadas nos Testes.

Observando-se as imagens da Figura 5.1, pode-se perceber que trata-se de um conjunto de imagens de teste com atividade espacial bastante variada. Temos imagens escuras (por exemplo, Figura 5.1(d)), claras (por exemplo, Figura 5.1(a)), mas na sua maioria, as imagens possuem histogramas bem espalhados.

Os resultados serão apresentados na forma de tabelas e gráficos, comparativos de desempenho. Os gráficos sintetizam o conteúdo de algumas tabelas, permitindo uma melhor visualização e comparação dos resultados. Cada ponto do gráfico corresponde a uma entrada da tabela de origem correspondente, ou seja, um ponto para um dado limiar.

Em todas as tabelas, a coluna "Limiar" apresenta os valores utilizados no teste de homogeneidade; a coluna "BITS" apresenta o número de bits necessários para representar a árvore (TT ou QT); a coluna "Vértices" apresenta o número total de vértices gerados pela decomposição da imagem, seja em quadrados, ou em triângulos; a coluna "Partições", nas Tabelas referentes a QT, indica o número total de quadrados em que a imagem foi dividida, e nas Tabelas referentes à TT, o número total de triângulos; os valores da coluna "Taxa" foram obtidos utilizando-se 8 bits por nível de cinza, sendo dados em bits por *pixel* (bpp), salvo os casos em que alguma modificação seja mencionada; a última coluna, "PSNR", apresenta uma medida objetiva da qualidade da reconstrução das imagens, dada em termos de PSNR[2].

Algumas tabelas apresentam os resultados obtidos com limiar maior que 50, o que acontece quando a qualidade subjetiva visual da imagem ainda é boa neste valor de limiar. Entretanto, alguns outros experimentos só conseguem obter resultados aceitáveis para valores de limiar menores que 50. No caso dos métodos que trabalham com limiar variável, os limiares que aparecem nas tabelas de resultados são os valores iniciais fornecidos para os métodos.

Os primeiros resultados obtidos referem-se à equalização ou não dos vértices, com a seguinte configuração: teste de homogeneidade original (diferença entre o

[2]Apesar da autora não acreditar que a PSNR seja a medida mais correta para essa comparação, concordando sobre isso com muitos autores [51; 18; 50; 52] que só a utilizam por ser a medida constantemente utilizada na literatura.

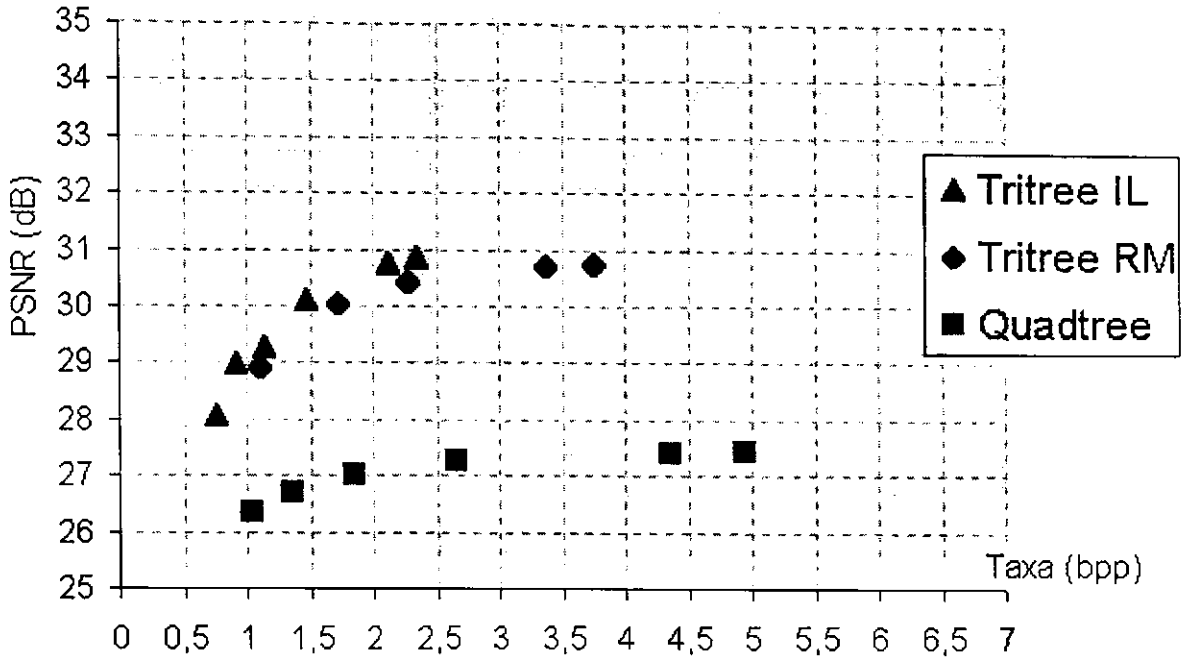


Figura 5.2: *Tritree* x *Quadtree* - *Tritree*: Vértices não Equalizados, Reconstrução por Interpolação Linear (IL) e Replicação da Média (RM). *Quadtree*: Original.

maior e menor valores de nível de cinza dentro do triângulo), limiar fixo e as duas formas de reconstrução: interpolação linear e replicação da média. Esses resultados podem ser encontrados na Seção 5.1.

Os testes seguintes resultaram dos diferentes testes de homogeneidade (Seção 4.1.5) pesquisados que aparecem na Seção 5.2. Na Seção 5.3 serão apresentados os resultados obtidos para algumas das configurações anteriores com limiar fixo, comparando o desempenho com limiar variável. Os resultados para *Tritree*, utilizando-se codificação preditiva dos vértices, podem ser encontrados na seção 5.4. A Seção 5.5 mostra os resultados conseguidos, com as duas formas de codificação da informação da estrutura da árvore. Na seqüência, a Seção 5.6 apresenta os resultados que foram conseguidos através da abordagem híbrida com transformadas *wavelet*. Finalmente, a última seção apresenta uma análise dos resultados obtidos no domínio da freqüência, ou seja, no domínio da Transformada de Fourier. Este Capítulo encerra-se com as conclusões obtidas a partir dos resultados apresentados.

5.1 Equalização dos Vértices

Os resultados experimentais obtidos antes da equalização dos vértices para a *Tritree* estão nas Tabelas 5.1 e 5.2, para reconstrução por interpolação linear e replicação da média, respectivamente. Os resultados obtidos para o algoritmo da *Quadtree* estão na Tabela 5.3, para os quais não foi necessária a utilização da estratégia de equalização. O gráfico da Figura 5.2 sumariza as três tabelas anteriores, para uma melhor comparação dos resultados, em termos de PSNR x Taxa. Analisando-se este gráfico percebe-se a grande superioridade dos algoritmos da *Tritree* em relação ao da *Quadtree*, pois as duas formas de reconstrução da *Tritree* exibem um melhor desempenho, alcançando maiores PSNR's para valores de taxa muito próximos.

Tabela 5.1: *Tritree*: Vértices não Equalizados e Reconstrução por Interpolação Linear.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	123953	61355	107656	2,345	30,842
10	112221	55529	96742	2,123	30,757
20	77285	38829	65371	1,480	30,104
30	58741	30016	49168	1,140	29,269
40	46941	24359	38917	0,922	28,981
50	38477	20257	31609	0,765	28,049

Tabela 5.2: *Tritree*: Vértices não Equalizados e Reconstrução por Replicação da Média.

Limiar	BITS	Vértices	Partições	Taxa	PSNR
8	123953	61355	107656	3,758	30,743
10	112221	55529	96742	3,380	30,705
20	77285	38829	65371	2,290	30,427
30	58741	30016	49168	1,725	30,019
50	38477	20257	31609	1,111	28,926

Tabela 5.3: *Quadtree* Original.

Limiar	BITS	Vértices	Partições	Taxa	PSNR
8	68133	165739	153925	4,957	27,420
10	61677	146802	135115	4,359	27,4001
20	43053	92329	81808	2,661	27,245
30	32877	65822	56539	1,851	27,008
40	26201	50009	41713	1,373	26,686
50	21557	39112	31702	1,050	26,339

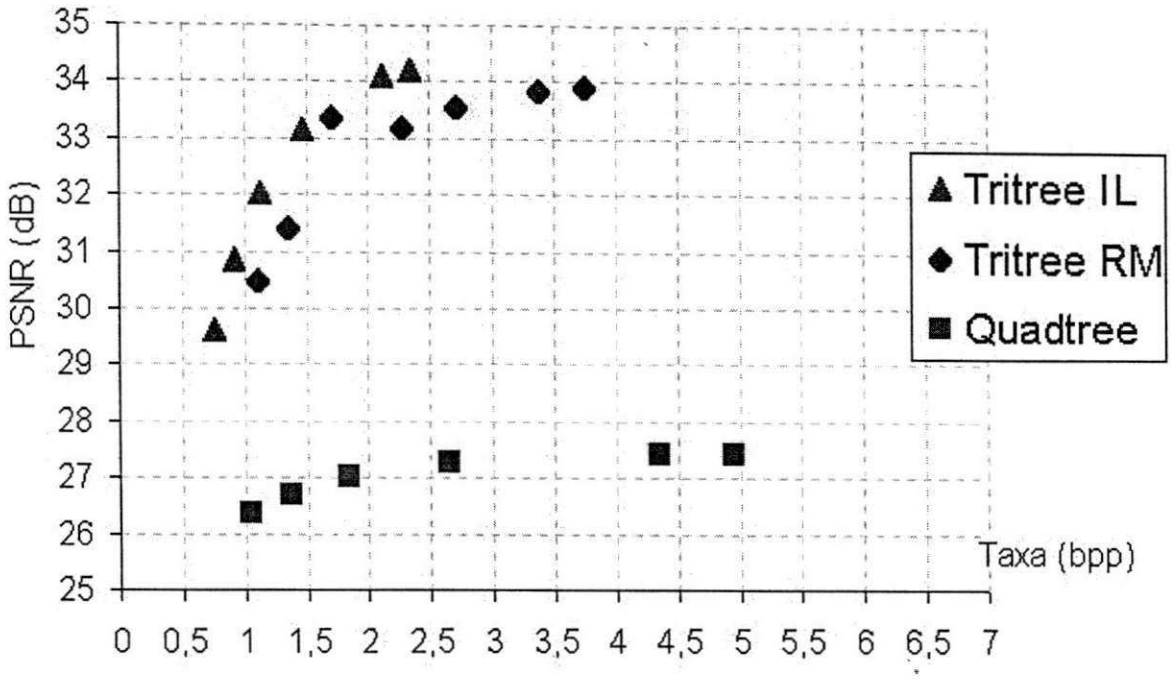


Figura 5.3: *Tritree* x *Quadtree* - *Tritree*: Vértices Equalizados, Reconstrução por Interpolação Linear (IL) e Replicação da Média (RM). *Quadtree*: Original.

Tabela 5.4: *Tritree*: Reconstrução por Replicação da Média.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (db)
8	123953	61355	107656	3,758	33,908
10	112221	55529	96742	3,380	33,837
15	91161	45349	77665	2,718	33,550
20	77285	38829	65371	2,290	33,170
30	58741	30016	49168	1,725	33,362
40	46941	24359	38917	1,367	31,395
50	38477	20257	31609	1,111	30,491

Resultados considerando os vértices equalizados para a *Tritree*, aparecem nas Tabelas 5.4 e 5.5, para reconstrução por interpolação linear e replicação da média, respectivamante. Estes resultados também estão em forma de gráficos na

Figura 5.3, juntamente com os valores da *Quadtree* da Tabela 5.3, para comparação. Comparando-se estes resultados percebe-se a superioridade do método *Tritree* em relação ao *Quadtree*.

Comparando os resultados obtidos, com e sem equalização dos vértices, fica evidente a superioridade do primeiro caso. Conclui-se, então, que esta equalização trouxe melhoras significativas para os valores de PSNR, com insignificante prejuízo na taxa. Por isso mesmo, a equalização dos vértices foi adotada como definitiva e todos os resultados apresentados, a partir deste ponto do trabalho, foram gerados utilizando esta equalização.

Tabela 5.5: *Tritree*: Reconstrução por Interpolação Linear.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	123953	61355	107656	2,345	34,186
10	112221	55529	96742	2,123	34,079
20	77285	38829	65371	1,480	33,123
30	58741	30016	49168	1,140	32,015
40	46941	24359	38917	0,922	30,830
50	38477	20257	31609	0,765	29,600

Observando-se as colunas "Vértices" e "Partições" das tabelas, pode-se observar que o número de vértices gerados pela *Quadtree* é sempre muito maior do que o número de vértices gerados pela *Tritree*. Tomemos, como exemplo, a linha de limiar 20 da Tabela 5.3, referente à *Quadtree*, e de limiar 10 da Tabela 5.5. Na primeira temos um número de partições de 81.808 que gerou um total de 92.329 vértices, enquanto que na tabelas da TT temos um número de partições bem maior de 96.742, enquanto que o número de vértices é bem menor, ou seja, de 55.529.

Isto poderá ser observado em todas as tabelas, até o final do texto. Este fato é demonstrado matematicamente, por Mayer em [28], que conclui no final de sua pesquisa: "Dada uma imagem retangular I, particionada em um dado número de

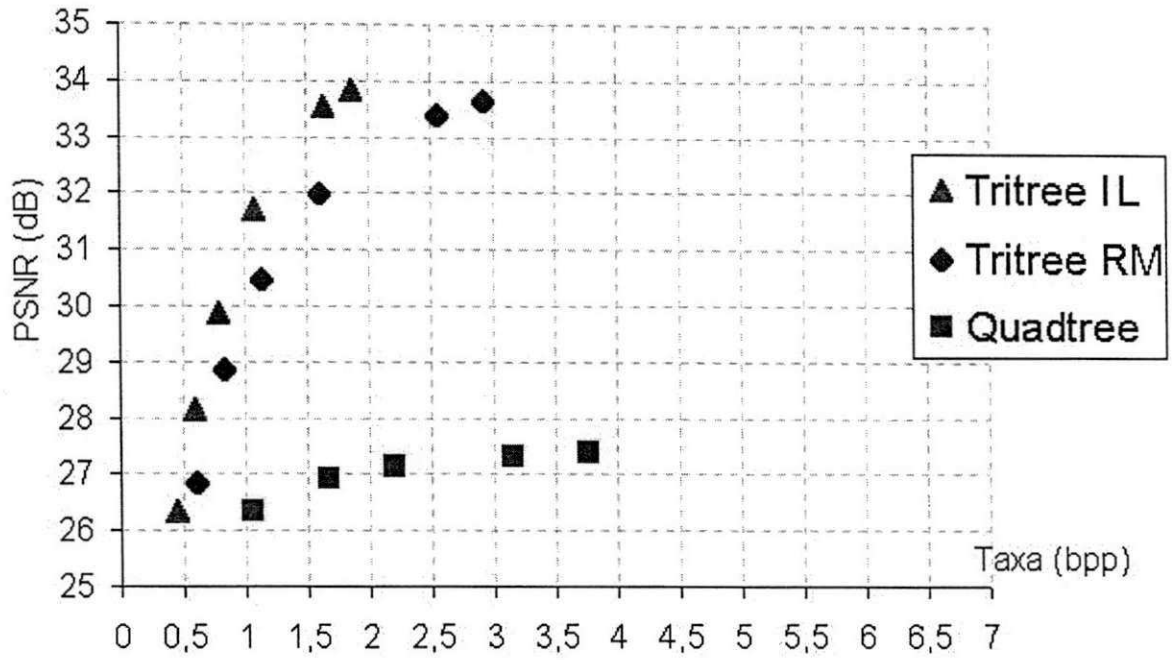


Figura 5.4: *Tritree* x *Quadtree* - *Tritree*: Teste de Homogeneidade pelo Desvio da Média, Reconstrução por Interpolação Linear (IL) e Replicação da Média (RM). *Quadtree*: Teste de Homogeneidade pelo Desvio da Média.

regiões de mesma área A, o particionamento triangular requer cerca de metade dos vértices requeridos para o particionamento em quadrados." Para o particionamento em regiões de diferentes áreas, esta afirmação pode ser verificada empiricamente, através da análise dos experimentos, tanto neste trabalho quanto no de Mayer [28].

Tabela 5.6: *Tritree*: Interpolação Linear e Teste de Homogeneidade pelo Desvio da Média.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	98041	48803	83788	1,863	33,822
10	86397	43292	73384	1,651	33,523
20	54925	28295	45808	1,073	31,708
30	39393	20803	32362	0,785	29,855
40	29453	15853	23896	0,596	28,138
50	21829	11906	17557	0,447	26,316

5.2 Teste de Homogeneidade

As Tabelas 5.6, 5.7 e 5.8 apresentam resultados obtidos com limiar fixo e teste de homogeneidade pelo desvio da média, sendo a primeira para o algoritmo da *Tritree* com reconstrução por interpolação linear, a segunda com reconstrução por replicação da média, enquanto que a última se refere aos resultados para a *Quadtree*. A Figura 5.4 apresenta os resultados das três tabelas anteriores, em um mesmo gráfico comparativo.

Tabela 5.7: *Tritree*: Replicação da Média e Teste de Homogeneidade pelo Desvio da Média.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	98041	48803	83788	2,931	33,640
10	86397	43292	73384	2,569	33,398
15	67025	34138	56377	1,976	32,731
20	54925	28295	45808	1,607	31,986
30	39393	20803	32362	1,138	30,478
40	29453	15853	23896	0,842	28,858
50	21829	11906	17557	0,619	26,817

Tabela 5.8: *Quadtree*: Teste de Homogeneidade pelo Desvio da Média.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	55693	128220	116632	3,772	27,368
10	48849	108472	97270	3,155	27,311
15	37653	77847	67948	2,217	27,130
20	30709	60307	51364	1,685	26,907
30	21957	39305	32017	1,061	26,315

Nestes resultados, mais uma vez, verifica-se a superioridade do método *Tritree* em relação ao *Quadtree*. Tomemos, como exemplo, o valor de taxa na Tabela 5.6 obtido com o limiar 20 e na Tabela 5.8 com o limiar 30. No primeiro caso, temos uma taxa de 1,073 bpp para uma PSNR de 31,708 dB, enquanto que na Tabela da *Quadtree* para se obter uma taxa aproximadamente igual, de 1,061 bpp, é necessário tolerar uma PSNR significantemente menor, com um valor de 26,315 dB. O mesmo se verifica comparando-se as Tabelas 5.7 e 5.8.

Já as Tabelas 5.9, 5.10 e 5.11 ilustram os resultados obtidos com limiar fixo e teste de homogeneidade pelo desvio padrão, sendo, também, a primeira para o algoritmo da *Tritree* com reconstrução por interpolação linear, a segunda para a *Tritree* com reconstrução por replicação da média, enquanto que a última se refere aos resultados para a *Quadtree*. A Figura 5.5 sumariza, em um gráfico comparativo, estas três tabelas. Nesta figura, percebe-se ainda os melhores resultados obtidos pelos dois algoritmos da *Tritree*, quando comparados com os resultados da *Quadtree*.

Tabela 5.9: *Tritree*: Interpolação Linear e Teste de Homogeneidade pelo Desvio Padrão.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	142834	78953	119458	4,105	33,313
10	123256	66412	106015	3,653	32,964
20	101971	47344	84529	2,985	32,650
30	84562	39601	70415	2,122	31,549
40	59654	31101	52639	1,378	30,486
50	43456	25589	48607	0,957	29,230

Tabela 5.10: *Tritree*: Replicação da Média e Teste de Homogeneidade pelo Desvio Padrão.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	142834	78953	119458	4,020	33,152
10	123256	66412	106015	3,422	32,519
20	101971	47344	84529	2,555	32,006
30	84562	39601	70415	1,989	30,885
40	59654	31101	52639	1,256	30,169
50	43456	25589	48607	0,854	28,981

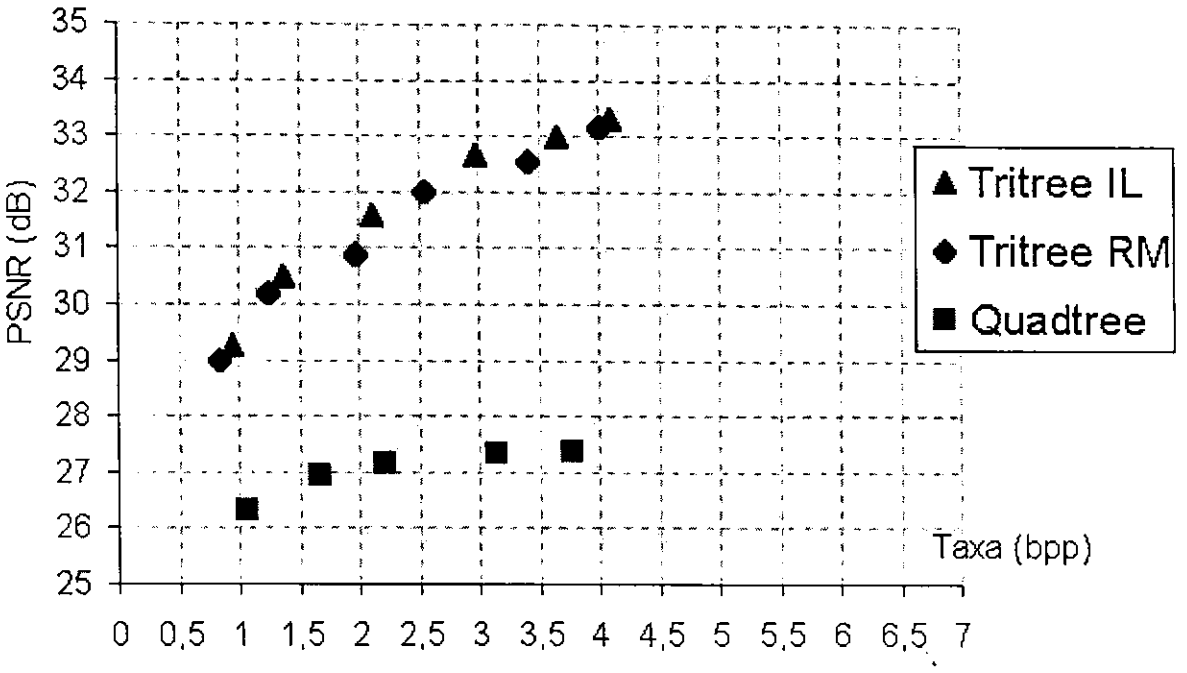


Figura 5.5: *Tritree* x *Quadtree* - *Tritree*: Teste de Homogeneidade pelo Desvio Padrão, Reconstrução por Interpolação Linear (IL) e Replicação da Média (RM). *Quadtree*: Teste de Homogeneidade pelo Desvio Padrão.

Tabela 5.11: *Quadtree*: Teste de Homogeneidade pelo Desvio Padrão.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	55693	128220	116632	3,772	27,368
10	48849	108472	97270	3,155	27,311
15	37653	77847	67948	2,217	27,130
20	30709	60307	51364	1,685	26,907
30	21957	39305	32017	1,061	26,315

Fazendo-se agora a análise entre os três testes de homogeneidade, plotados juntos na Figura 5.6, para as duas formas de reconstrução, percebe-se que as duas curvas referentes ao teste pelo desvio padrão encontram-se abaixo das demais. Esta observação foi feita também para as demais imagens testadas, não incluídas neste

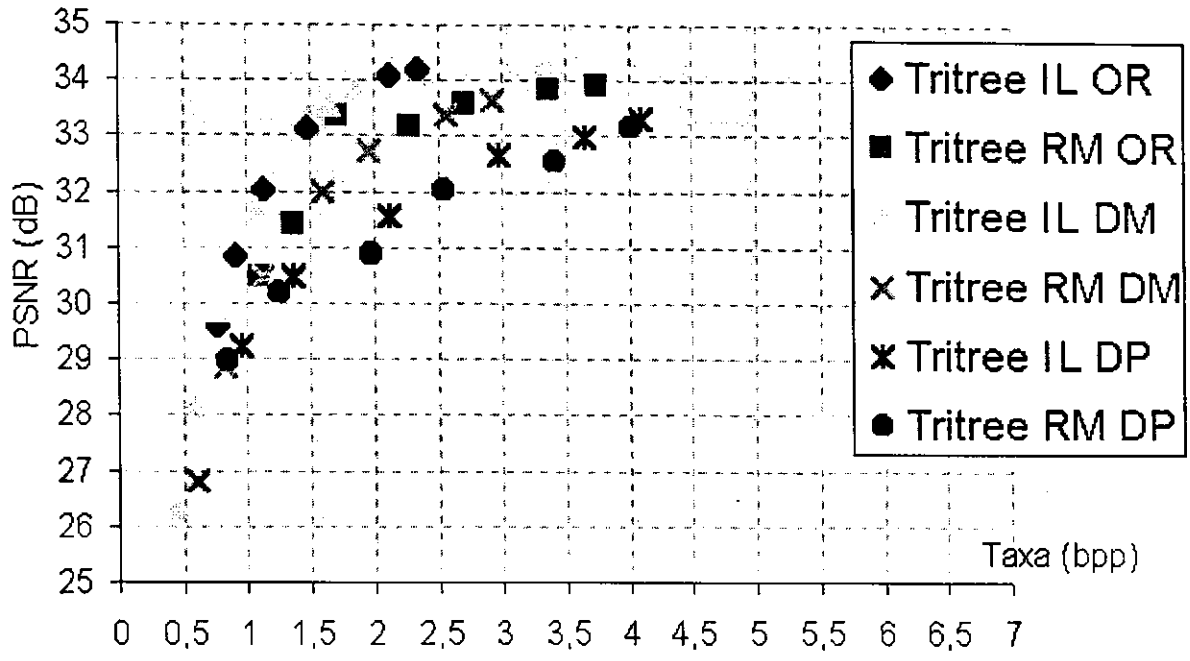


Figura 5.6: *Tritree*: Comparação entre os Diferentes Testes de Homogeneidade (Teste Original - OR, Teste pelo Desvio da Média - DM e Teste pelo Desvio Padrão - DP) e as Diferentes Formas de Reconstrução (Interpolação Linear - IL e Replicação da Média - RM)

texto. Por este motivo, descartou-se o teste de homogeneidade pelo desvio padrão, no restante do trabalho.

Comparando-se os outros dois testes de homogeneidade ainda na Figura 5.6, pode-se perceber que as duas curvas referentes a uma mesma forma de reconstrução praticamente se sobrepõem, independente do tipo de teste utilizado. Disto pode ser concluído, então, que os dois testes geram resultados equivalentes, para uma mesma forma de reconstrução.

Uma vantagem realmente significativa do teste de homogeneidade original, é o número menor de operações requeridas, pois, para o teste pelo desvio da média, antes de se fazer as comparações *pixel* a *pixel*, tem-se que calcular média aritmética simples dos níveis de cinza para cada triângulo.

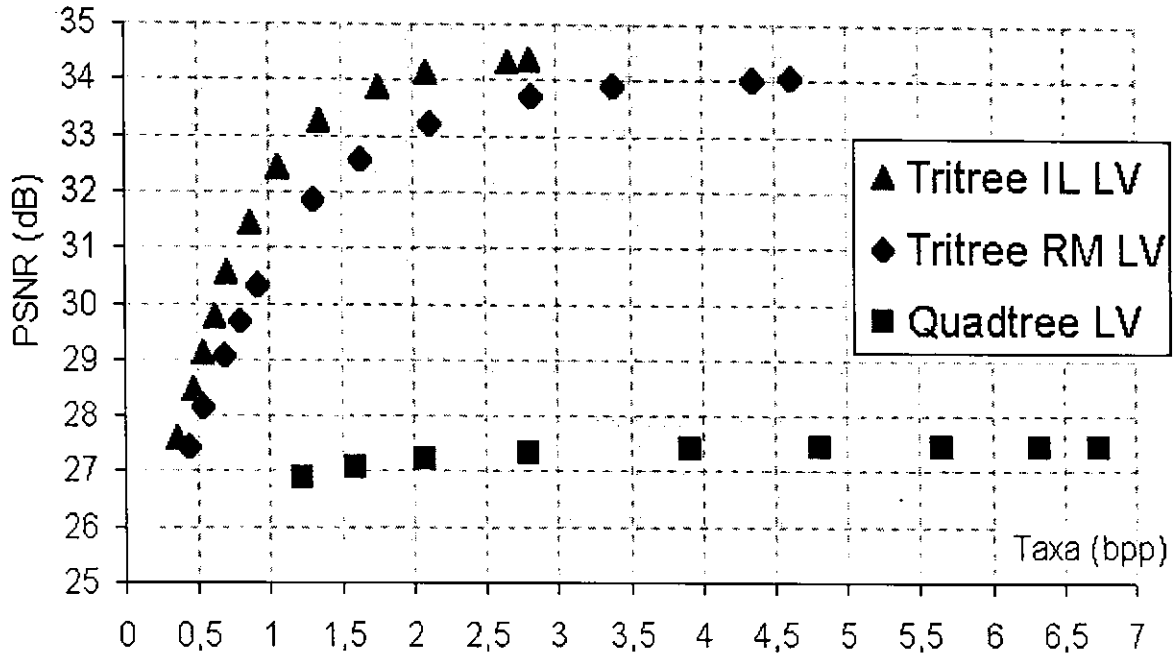


Figura 5.7: *Tritree* x *Quadtree* - *Tritree*: Limiar Variável (LV), Teste de Homogeneidade Original, Reconstrução por Interpolação Linear (IL) e Replicação da Média (RM). *Quadtree*: Limiar Variável (LV) e Teste de Homogeneidade Original.

5.3 Limiar Fixo x Limiar Variável

As tabelas 5.12, 5.13 e 5.14 apresentam os resultados obtidos com o limiar variável e teste de homogeneidade original, as duas primeiras para a *Tritree*, com reconstrução por interpolação linear e replicação da média, respectivamente, e a última para a *Quadtree*. Os altos valores de limiar são justificados pela baixa velocidade de degradação das imagens: os testes só paravam quando um baixo valor de PSNR e/ou uma baixa qualidade subjetiva da imagem aconteciam. Estes resultados plotados juntos encontram-se na Figura 5.7.

A superioridade da TT sobre a QT continua em evidência, sendo muito mais acentuada neste caso do limiar variável. Tomando-se valores aproximados de PSNR temos 27,452 dB na Tabela 5.13, conseguido com limiar 300 e uma taxa de 0,445 bpp, enquanto que na Tabela 5.14, para um valor um pouco mais baixo, 27,443 dB, temos

Tabela 5.12: *Tritree*: Interpolação Linear, Teste de Homogeneidade Original e Limiar Variável.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	156465	72613	132148	2,813	34,359
10	148713	68504	124219	2,658	34,332
20	118837	53938	96586	2,099	34,141
30	100193	45455	80305	1,769	33,876
50	76613	34734	60229	1,352	33,233
75	60121	27341	46519	1,064	32,402
100	48269	22251	36982	0,863	31,428
120	39085	18144	29671	0,703	30,538
150	34253	15963	25936	0,618	29,744
175	29921	13872	22501	0,537	29,099
200	25889	12070	19459	0,467	28,463
250	17229	9585	15469	0,358	27,595

uma taxa inaceitável de 6,746 bpp para o limiar 8.

Comparando-se os resultados da *Tritree* com limiar variável para os dois tipos de reconstrução, a interpolação linear continua produzindo resultados sutilmente superiores aos da replicação da média. Na tabela 5.13 temos valores de PSNR de 27,452 dB com uma taxa de 0,445 bpp, enquanto que na Tabela 5.12 temos uma PSNR de 27,595 dB para uma taxa de 0,358 bpp.

Tabela 5.13: *Tritree*: Replicação da Média, Teste de Homogeneidade Original e Limiar Variável.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	156465	72613	132148	4,630	34,037
10	148713	68504	124219	4,358	34,020
20	118837	53938	96586	3,401	33,885
30	100193	45455	80305	2,833	33,700
50	76613	34734	60229	2,130	33,222
75	60121	27341	46519	1,649	32,582
100	48269	22251	36982	1,313	31,839
150	34253	15963	25936	0,922	30,311
175	29921	13872	22501	0,801	29,675
200	25889	12070	19459	0,693	29,081
250	17229	9585	15469	0,538	28,156
300	15113	7872	1269	0,445	27,452

Tabela 5.14: *Quadtree*: Limiar Variável.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	86145	220284	210271	6,746	27,443
10	86145	207243	197017	6,341	27,440
15	84677	185282	174847	5,659	27,434
20	78973	159024	148087	4,821	27,420
30	72773	129834	119365	3,920	27,391
50	59957	93466	84199	2,798	27,313
75	49969	70501	62113	2,086	27,202
100	42237	54740	47212	1,602	27,054
130	35945	41973	35857	1,231	26,860

Tabela 5.15: *Tritree*: Interpolação Linear, Limiar Variável e Desvio da Média.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	140137	64024	115564	2,488	34,291
10	129601	58992	105943	2,295	34,231
20	95513	43620	76306	1,696	33,777
30	76529	35055	60298	1,362	33,256
50	55813	25771	43135	0,999	32,715
75	41489	19536	31720	0,754	32,132
100	32285	15232	24445	0,588	30,667
125	26277	12448	19819	0,480	29,496
150	22181	10489	16672	0,405	28,576
200	14401	7962	12418	0,298	27,887

Comparando-se as mesmas combinações de execuções do método e variando apenas o limiar (fixo ou variável), percebe-se uma vantagem para o limiar variável.

Fazendo o mesmo tipo de análise, procurando PSNR's aproximadas e comparando as taxas bpp, nota-se sempre uma superioridade dos valores obtidos para limiar variável. Essa superioridade se acentua, comparando-se subjetivamente as imagens recontruídas. Na imagem com o limiar variável, as bordas dos triângulos são mais suaves, o que deixa a imagem visivelmente mais agradável, em relação às imagens com limiar fixo.

Tabela 5.16: *Tritree*: Replicação da Média, Teste de Homogeneidade pelo desvio da Média e Limiar Variável.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	140137	64024	115564	4,061	33,986
10	129601	58992	105943	3,728	33,941
20	95513	43620	76306	2,693	33,615
30	76529	35055	60298	2,132	33,188
50	55813	25771	43135	1,529	32,277
75	41489	19536	31720	1,126	31,045
100	32285	15232	24445	0,869	29,863
130	25329	11997	19075	0,679	28,749
160	20937	9948	15733	0,560	27,997

Tabela 5.17: *Quadtree*: Limiar Variável e Desvio da Média.

Limiar	BITS	Vértices	Partições	Taxa (bpp)	PSNR (dB)
8	86449	203932	193456	6,234	27,439
10	86449	185213	174940	5,669	27,433
15	83509	157432	146962	4,803	27,418
20	72573	125937	115237	3,794	27,383
30	63193	96579	87280	2,905	27,318
50	48601	65730	57601	1,943	27,153
75	38169	45351	38914	1,333	26,900
100	30721	32559	27895	0,969	26,618
130	25613	24977	21331	0,749	26,340

Nas Tabelas 5.15, 5.16 e 5.17 estão os resultados obtidos para limiar variável e teste pelo desvio da média. Estes resultados podem ser vistos também na Figura 5.8. Na Tabela 5.15 a reconstrução foi feita pela interpolação linear, e na Tabela 5.16 pela replicação da média. Também, neste caso, manifesta-se a superioridade da *Tritree* sobre a *Quadtree*: na Tabela 5.15 temos uma PSNR de 27,887 dB com taxa de 0,298 bpp, enquanto que na Tabela da *Quadtree* (5.17), para uma PSNR aproximada, mas inferior, de 27,318 dB, a taxa é bem maior, ou seja 2,905 bpp.

Comparando-se apenas as Tabelas referentes à *Tritree* (5.15 e 5.16), pode ser concluído, a respeito dos métodos de reconstrução, que interpolação linear apresenta melhores resultados. Isto pode ser confirmado analisando-se suas respectivas curvas na Figura 5.8.

5.4 Codificação Preditiva dos Vértices

Triângulos vizinhos possuem vértices em comum. Assim, um único vértice pode

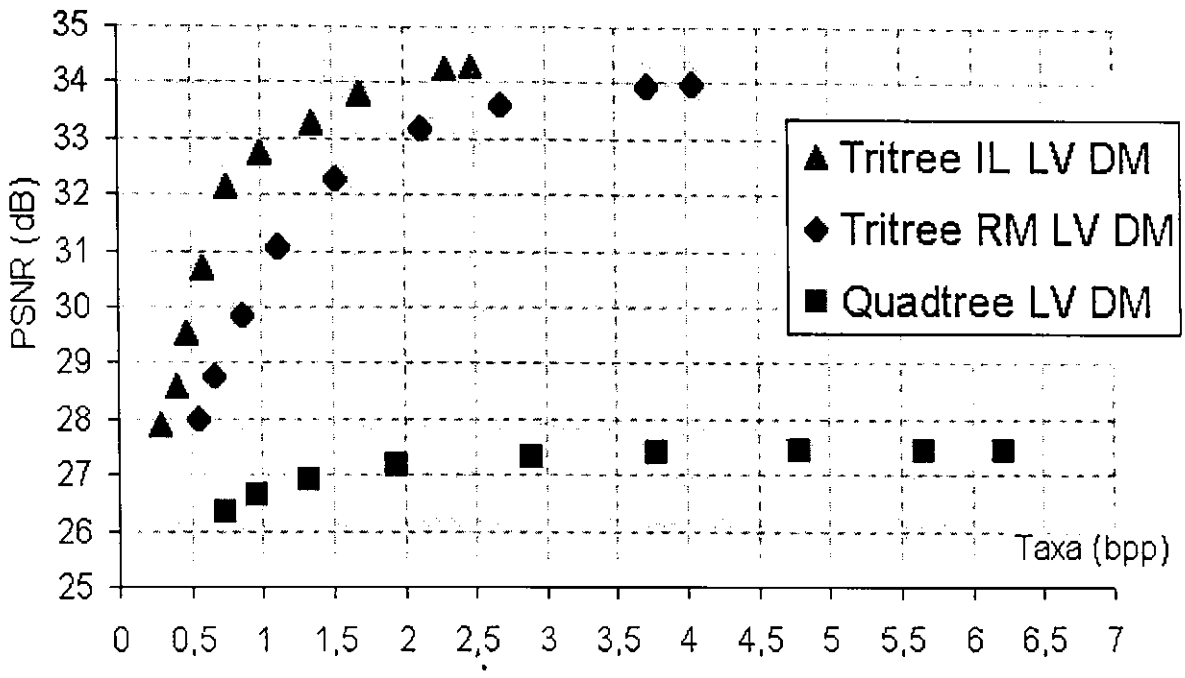


Figura 5.8: *Tritree* x *Quadtree* - *Tritree*: Limiar Variável (LV), Teste de Homogeneidade pelo Desvio da Média (DM), Interpolação Linear (IL) e Replicação da Média (RM). *Quadtree*: Limiar Variável (LV), Teste de Homogeneidade pelo Desvio da Média (DM).

pertencer a até seis triângulos distintos. No caso de limiares fixos, pode-se garantir que, dentro de um mesmo triângulo, não existem valores de nível de cinza cuja diferença exceda o valor do limiar, incluindo-se os vértices. Com isso, pode-se garantir também que, entre triângulos vizinhos, não existem valores de nível de cinza cuja diferença exceda o valor do limiar. Disto resultou o desenvolvimento de uma técnica de codificação preditiva para armazenar os valores de nível de cinza dos vértices, descrita a seguir.

Tabela 5.18: *Tritree*: Codificação Preditiva e Interpolação Linear.

Limiar	Taxa Anterior (bpp)	Taxa (bpp)	Ganho (%)	PSNR (dB)
8	2,345	1,383	41,02	34,186
10	2,123	1,249	41,17	34,079
20	1,480	1,003	32,23	33,123
30	1,140	0,763	33,07	32,015
40	0,922	0,698	24,30	30,830
50	0,765	0,573	25,10	29,600

Tabela 5.19: *Tritree*: Codificação Preditiva, Interpolação Linear e Limiar Variável.

Limiar Inicial	Taxa Anterior (bpp)	Taxa (bpp)	Ganho (%)	PSNR (dB)
8	2,813	1,465	47,92	34,291
10	2,658	1,378	48,16	34,231
20	2,099	1,272	39,40	33,777
30	1,769	1,057	40,25	33,256
50	1,352	0,924	31,66	32,715
75	1,064	0,817	23,21	32,132
100	0,863	0,653	24,33	30,667
130	0,703	0,592	15,79	29,496
150	0,618	0,515	16,67	28,576

Um vértice é escolhido arbitrariamente (o primeiro a ser calculado ou qualquer outro) e codificado com 8 bits (dito preditor). Para os demais vértices do mesmo triângulo apenas a diferença entre seus valores de nível de cinza e o nível de cinza do preditor é armazenada. O processo é repetido sucessivamente, com os triângulos

vizinhos e com os vizinhos dos vizinhos. Para o caso de limares variáveis, este método é aplicado por nível. Com isso, o armazenamento também é feito por nível da árvore.

O método de codificação preditiva desenvolvido só pode ser aplicado conjuntamente com o teste de homogeneidade original, pois só com este sabe-se a priori que a diferença entre o maior e menor valor de nível de cinza é o mesmo para todos os triângulos, o próprio limiar. Para o teste pelo desvio da média, um outro método deve ser desenvolvido. Como as *Quadtrees* não armazenam os valores de nível de cinza dos vértices, e sim a média de cada quadrado, a codificação preditiva também não pode ser aplicado neste caso. Pelo mesmo motivo o método não pode ser aplicado para a *Tritree* com reconstrução pela replicação da média.

Na Tabela 5.18 observam-se os resultados para a *Tritree* com limiar fixo e teste de homogeneidade original, com codificação preditiva. A diferença entre esta Tabela e a Tabela 5.5 são os valores da coluna "Taxa" que, neste caso, são significativamente menores. Com esta modificação, os resultados para a *Tritree* com interpolação linear tornaram-se melhores do que os obtidos com replicação da média. Da mesma forma, tornaram-se melhores os resultados para o teste de homogeneidade original em termos de taxa.

Os resultados obtidos para a *Tritree* com limiar variável e codificação preditiva, aparecem na Tabela 5.19. Como no caso anterior, a diferença entre estes resultados e os anteriores, da Tabela 5.12, estão somente no valor da coluna "Taxa", sendo os da Tabela 5.19 bem inferiores. Os resultados apresentados nas Tabelas 5.18 e 5.19 mostram que o uso de um método de codificação preditiva, para armazenar os níveis de cinza dos vértices, foi bem sucedido, incentivando mais pesquisas nesta área.

5.5 Codificação da Estrutura da Árvore

Na tentativa de reduzir a taxa de bits por *pixels*, sem aumentar a degradação da imagem, foram pesquisadas duas técnicas de codificações sem perdas, para codificar a informação da estrutura da árvore *Tritree*. Em ambas, parte-se da idéia de representar as seqüências de 0s e 1s por valores numéricos que representam o tamanho de cada sequência. As duas técnicas são explicadas a seguir.

Nesta etapa do trabalho foram realizados testes exaustivos com vários limiares diferentes, para os melhores casos detectados através dos resultados apresentados anteriormente.

Todos estes testes foram realizados procurando-se determinar os parâmetros que produzem o melhor resultado. Na busca deste objetivo, foi gerada uma quantidade muito grande de dados e informação. Para não tornar esta leitura cansativa, será apresentado aqui um pequeno conjunto de resultados que, contudo, representa uma amostra significativa dos resultados obtidos, tanto para a imagem "Lena", quanto para as demais imagens.

5.5.1 Codificação da estrutura da Árvore sem Utilização de *Flag*

Para que não fosse necessário o desenvolvimento de alguma *flag* para informar o número de bits utilizado em cada tamanho de sequência, foi empregado o mesmo tamanho para todas.

Foram realizados testes com número de bits diferentes para codificar o tamanho das sequências, variando de dois a oito bits. Como foi observado que estas sequências de 0s e 1s crescem com o nível de profundidade da árvore, também foi testada a utilização desta codificação a partir de níveis diferentes, variando entre o nível um e o nove (lembrando que a raiz é o nível zero). Isto foi testado para cada limiar de decisão, sendo fixo ou variável, para as execuções apresentadas nas seções anteriores, que conseguiram melhorar os resultados.

Tabela 5.20: *Tritree*: Codificação da Estrutura da Árvore TT sem Flag, para Limiar Fixo 8, Teste de Homogeneidade Original, Interpolação Linear, Taxa Anterior de 1,3829 bpp e PSNR de 34,186 dB.

Bits	Nível Inicial	Taxa Nova (bpp)
2	3	1,2702
3	1	1,2133
4	4	1,1942
5	4	1,2027
6	9	1,2083
7	9	1,2215
8	9	1,2399

Tabela 5.21: *Tritree*: Codificação da Estrutura da Árvore TT sem Flag, para Limiar Fixo 20, Teste de Homogeneidade Original, Interpolação Linear, Taxa Anterior de 1,0027 bpp e PSNR de 33,123 dB.

Bits	Nível Inicial	Taxa Nova (bpp)
2	1	0,9416
3	1	0,9196
4	9	0,9023
5	9	0,8960
6	9	0,8961
7	9	0,9019
8	9	0,9112

Nas Tabelas 5.20 à 5.24 estão os resultados para a *Tritree* com limiar fixo, teste de homogeneidade original e interpolação linear. São apresentados os melhores re-

sultados em função do número de bits empregado para codificar as sequências. Na coluna "Nível Inicial" pode-se observar a partir de qual nível da árvore se utilizou codificação. Na primeira Tabela estão os resultados para o limiar 8, cuja taxa anterior era de 1,3829 bpp e PSNR de 34,186 dB. Na Tabela 5.21 estão os resultados para o limiar 20, que teve taxa anterior de 1,0027 bpp e PSNR de 33,123 dB. A Tabela 5.22 mostra os resultados para o limiar 30, com taxa anterior de 0,7639 e PSNR de 32,015 dB. Os resultados para o limiar 40 podem ser vistos na Tabela 5.23, com taxa anterior de 0,6981 bpp e PSNR de 30,830 dB. A última Tabela desta seção, a 5.24, apresenta os resultados para o limiar 50, com taxa anterior de 0,5731 bpp e PSNR de 29,600 dB.

Tabela 5.22: *Tritree*: Codificação da Estrutura da Árvore TT sem Flag, para Limiar Fixo 30, Teste de Homogeneidade Original, Interpolação Linear, Taxa Anterior de 0,7639 bpp e PSNR de 32,015 dB.

Bits	Nível Inicial	Taxa Nova (bpp)
2	1	0,7227
3	9	0,7044
4	9	0,6915
5	9	0,6865
6	9	0,6866
7	9	0,6908
8	9	0,6974

Todas as cinco tabelas apresentam seus melhores resultados com cinco bits para codificar o tamanho das sequências, iniciando a codificação no nono nível da árvore. Isto também se verificou nos demais resultados conseguidos e que não foram apresentados aqui. Saliente-se ainda que, sempre se obteve uma taxa melhorada, utilizando-se esta técnica, independente da configuração utilizada. Por este motivo esta técnica foi adotada como definitiva.

Tabela 5.23: *Tritree*: Codificação da Estrutura da Árvore TT sem Flag, para Limiar 40, Teste de Homogeneidade Original, Interpolação Linear, Taxa Anterior de 0,6981 bpp e PSNR de 30,830 dB.

Bits	Nível Inicial	Taxa Nova (bpp)
2	1	0,7227
3	9	0,7044
4	9	0,6915
5	9	0,6865
6	9	0,6866
7	9	0,6908
8	9	0,6974

5.5.2 Codificação da estrutura da Árvore com Utilização de *Flag*

Foi pesquisada também, uma forma de codificar a estrutura da árvore *Tritree* com seqüências de 0s e 1s de tamanhos variados. Para tanto, foi necessário a utilização de uma flag para informar o final de cada seqüência.

Optou-se por implementar uma forma de codificação intitulada: "Codificação Binária Desdobrada" [61]. Para realizar esta codificação, começa-se convertendo para o sistema de numeração binária (base dois) os números que quantificam o tamanho de cada seqüência de 0s e 1s. Em seguida, faz-se as substituições dos 0s e 1s resultantes da seguinte maneira:

$0 \rightarrow 0$

$1 \rightarrow 10$

$FLAG \rightarrow 110$

Por exemplo, considere-se uma seqüência original de seis (6) 1s, seguidos por oito (8) 0s. A codificação em binário resultante seria: 110 e 1000, respectivamente.

Tabela 5.24: *Tritree*: Codificação da Estrutura da Árvore TT sem Flag, para Limiar 50, Teste de Homogeneidade Original, Interpolação linear, Taxa Anterior de 0,5731 bpp e PSNR de 29,600 dB

Bits	Nível Inicial	Taxa Nova (bpp)
2	9	0,5511
3	9	0,5380
4	9	0,5302
5	9	0,5270
6	9	0,5271
7	9	0,5294
8	8	0,5332

Utilizando-se a forma binária desdobrada para armazenar esta informação no arquivo comprimido, teríamos:

1010011010000

Os resultados obtidos, utilizando-se esse tipo de codificação, com limiar fixo e teste de homogeneidade original são apresentados na Tabela 5.25. Como pode ser visto nesta Tabela, tem-se duas coluna com valores de taxas bpp, sendo que, na primeira temos as taxas obtidas sem nenhuma forma de codificação da estrutura da árvore *Tritree*, enquanto que na segunda os valores de taxas referem-se aos obtidos com a codificação binária desdobrada. Comparando-se estas duas colunas, percebe-se um aumento na taxa bpp, normalmente, na ordem dos centésimos, na segunda coluna, ou seja, nos resultados obtidos com esta forma de codificação. Isso pôde ser verificado em todos os teste feitos, seja com limiar variável, com outros testes de homogeneidade e outras imagens. Assim sendo, esta forma de codificação mostrou-se ineficiente, sendo então descartada.

Tabela 5.25: *Tritree*: Codificação da Estrutura da Árvore TT com Flag, Teste de Homogeneidade Original e Interpolação Linear.

Limiar	Taxa Anterior (bpp)	Taxa Nova (bpp)	PSNR (dB)
8	1,3829	1,4090	34,186
10	1,2492	1,2754	34,079
20	1,0027	1,0354	33,564
30	0,7639	0,7966	32,015
40	0,6981	0,7366	30,830
50	0,5731	0,6104	29,600

Tabela 5.26: *Tritree*: Processamento com Wavelets com 1 nível de Decomposição e PSNR Anterior de 30,830 dB.

Bandas	Fator de Atenuação dos Coeficientes	PSNR Nova
D	0	31,113
D	$\frac{1}{2}$	31,042
H	0	30,934
H	$\frac{1}{2}$	30,947
V	0	30,860
V	$\frac{1}{2}$	31,061
D e H	0	31,223
D e H	$\frac{1}{2}$	31,168
D e V	0	31,155
D e V	$\frac{1}{2}$	31,288
H e V	0	30,976
H e V	$\frac{1}{2}$	31,186
D, H e V	0	31,270
D, H e V	$\frac{1}{2}$	31,419
D, H e V	D - 0; V e H - $\frac{1}{2}$	31,493

5.6 Abordagem Híbrida com *Wavelets*

Esta implementação é uma tentativa de melhorar a qualidade subjetiva da imagem e sua PSNR. Para tanto, a imagem reconstruída é processada pelo algoritmo de decomposição de Mallat para a Transformada *Wavelets*. Os coeficientes de algumas das bandas resultantes são modificados e a imagem é finalmente reconstruída pela aplicação do algoritmo inverso da decomposição. As modificações aplicadas aos coeficientes de uma mesma banda são duas: eles são zerados ou divididos por dois. Os testes foram feitos com até três níveis de decomposição para todas as possibilidades de execução da decomposição *Tritree* vistas neste capítulo. Por isto mesmo, a gama de resultados gerados para posterior analise é imensa. Conseqüentemente, aqui só é apresentada uma pequena parte dos resultados, salientando-se que as conclusões tiradas a partir destes são suportadas pelo restante dos resultados.

Na Tabela 5.26, temos os resultados obtidos para a *Tritree* com limiar fixo de 40, teste de homogeneidade original, interpolação linear e um (1) nível de decomposição. Para este limiar, a melhor taxa alcançada foi de 0,639 bpp com cinco bits para codificar a informação do tamanho da seqüência, apenas no nível nove (Seção 5.5.1), e a PSNR anterior ao processamento foi de 28,555 dB. Na coluna "Bandas" estão as bandas que foram alteradas, sendo D - diagonal, V - vertical e H - horizontal. Na coluna "Fator de Atenuação dos Coeficientes", está o valor pelo qual foram multiplicados todos os coeficientes de cada banda.

Os resultados obtidos para três níveis de decomposição são omitidos pois sob qualquer configuração o resultado da PSNR é piorado para este caso, sendo então, esta possibilidade de execução descartada. Já para o segundo nível de decomposição, até foi conseguido, às vezes, melhorar a PSNR, entretando sem muita significância se comparado com os valores obtidos para PSNR com apenas um nível de decomposição.

Analisando o montante de resultados obtidos, pode-se afirmar que sempre se conseguiu melhorar a PSNR e a qualidade visual das imagens reconstruídas. Geral-

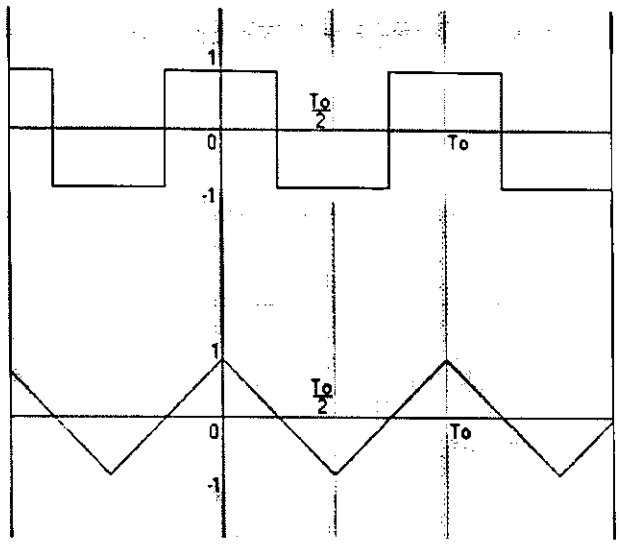


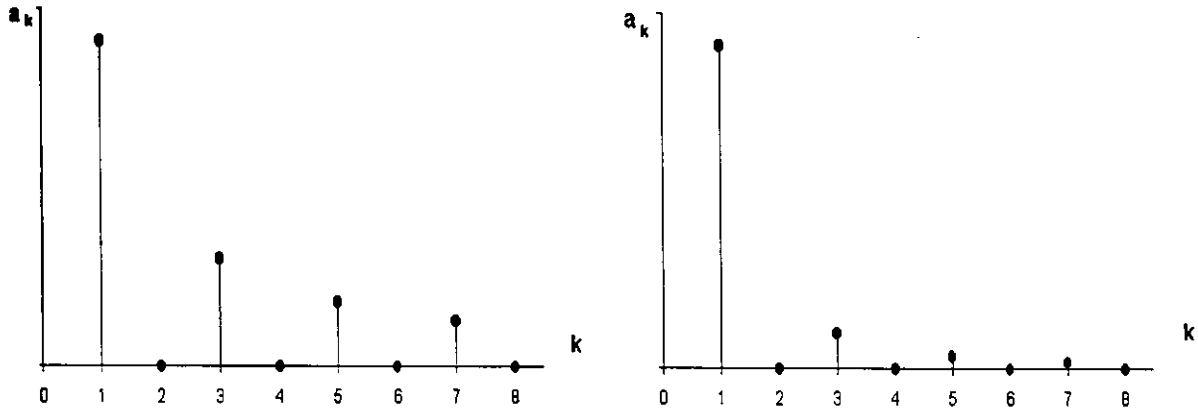
Figura 5.9: Sinais Unidimensionas: Onda Quadrada e Onda Triangular.

mente, os maiores valores de PSNR foram obtidos zerando-se as bandas D e H, ou D, H e V simultaneamente, ou ainda, zerando-se D e dividindo-se pela metade os coeficientes das outras bandas.

5.7 Análise de Fourier das Imagens Reconstruídas

Sendo imagens sinais digitais de duas variáveis, pretende-se utilizar a análise de Fourier para sinais unidimensionais e bidimensionais para mostrar que aproximar uma imagem por triângulos, como faz a *Tritree*, introduz menos distorção do que se a aproximarmos por quadrados, como faz a *Quadtree*. Essa discussão objetiva explicar os melhores resultados atingidos pela *Tritree*, comparando-os com os atingidos pela *Quadtree*. Isso se justificaria pela introdução de menos harmônicas por parte das *Tritrees*, conforme explanação abaixo.

Analisando inicialmente o caso unidimensional, consideremos os sinais que aparecem na Figura 5.9temos que os coeficientes da Série de Fourier para uma onda quadrada e para uma onda triangular são dados por [62; 6]:



(a) Onda Quadrada. (b) Onda Triangular.

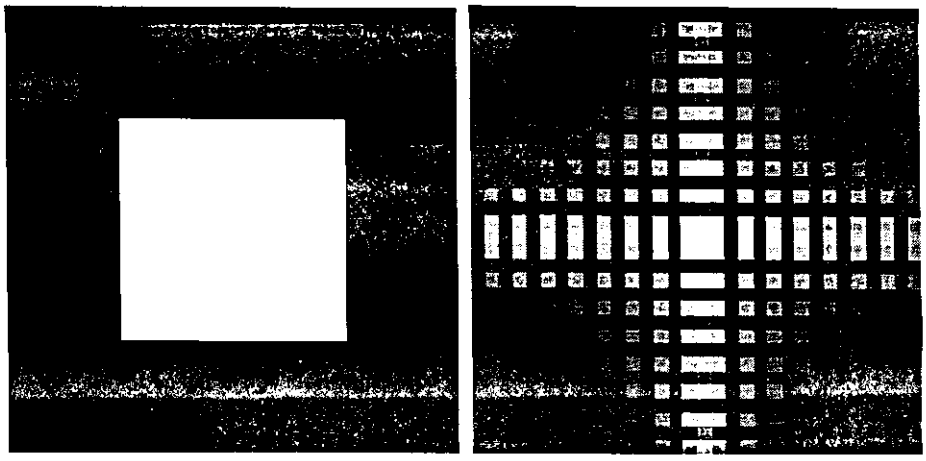
Figura 5.10: Coeficientes da Série de Fourier.

Onda Quadrada: $a_k = \frac{sen(k\pi/2)}{k\pi}$

Onda Triangular: $a_k = \frac{sen(k\pi/2)}{(k\pi)^2}$

Pode-se observar que os coeficientes a_k para a onda quadrada são inversamente proporcionais a k, enquanto que os a_k para a onda triangular são inversamente proporcionais a k^2. Portanto, as harmônicas decaem muito mais rapidamente no caso da onda triangular, podendo ser consideradas desprezíveis para valores de k relativamente baixos, como mostra a Tabela 5.27 e a Figura 5.10, onde seus coeficientes a_ks da série de Fourier podem ser vistos juntos.

Como pode ser observado tanto na Tabela, quanto na Figura, as componentes fundamentais (a_1s, primeiras harmônicas) possuem a mesma amplitude, enquanto que as outras componentes, da segunda harmônica em diante, decaem de forma diferente. Disto pode-se concluir que, para uma onda triangular, a componente fundamental já é suficiente para fazer uma boa aproximação para o sinal senoidal, enquanto que a onda quadrada precisa de mais algumas harmônicas para fazer uma melhor aproximação.



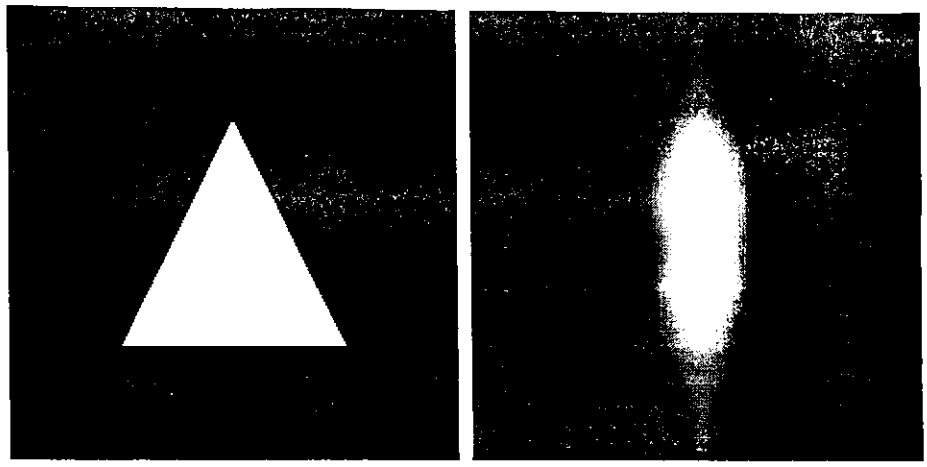
(a) Função 2D (b) Magnitude da FFT

Figura 5.11: Transformada de Fourier para a Imagem de um Quadrado.

Tabela 5.27: Coeficientes a_ks para Ondas Quadrada e Triangular.

Coeficientes	Janela Quadrada	Janela Triangular
a_1	$\frac{2}{\pi}$	$\frac{2}{\pi}$
a_2	0	0
a_3	$\frac{2}{3\pi}$	$\frac{2}{9\pi}$
a_4	0	0
a_5	$\frac{2}{5\pi}$	$\frac{2}{25\pi}$
a_6	0	0
a_7	$\frac{2}{7\pi}$	$\frac{2}{49\pi}$
a_8	0	0

A conclusão é que ondas triangulares possuem bem menos componentes harmônicas de alta frequência significativas do que ondas quadradas. Por extensão, pode-se concluir que a representação de um sinal por uma combinação linear de triângulos deslocados no tempo deverá possuir bem menos harmônicas significati-



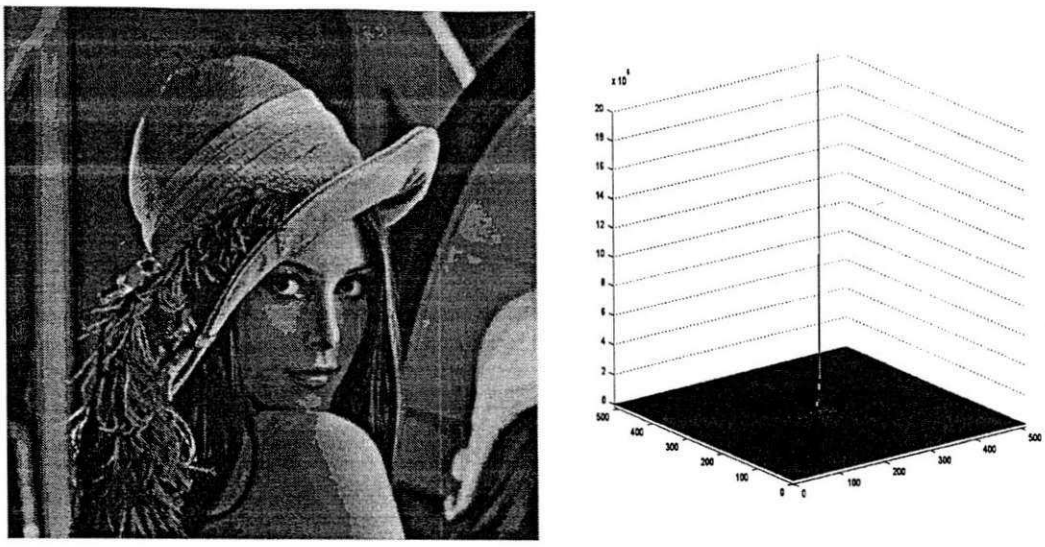
(a) Função 2D (b) Magnitude da FFT

Figura 5.12: Transformada de Fourier para a Imagem de um Triângulo

vas do que a representação do mesmo sinal por uma combinação linear de funções tipo "porta", ou seja, a representação obtida com triângulos (ou ondas triângulares) seria bem mais fiel ao sinal original do que a representação obtida para o mesmo sinal por meio de "quadrados" (ondas quadradas).

Consideremos, agora, o caso bidimensional (2D). Para tanto, foi utilizado o software MATLAB R12 para o cálculo e posterior visualização da magnitude da transformada de Fourier (FFT) da imagem de um quadrado e da imagem de um triângulo, ambos brancos em um fundo preto. Os resultados obtidos podem ser vistos nas Figuras 5.11 e 5.12, para o quadrado e triângulo, respectivamente. Nas Figuras 5.11(a) e 5.12(a) podem ser visualizadas as imagens testadas e nas Figuras 5.11(b) e 5.12(b), as magnitudes de suas FFT. As regiões mais claras correspondem a coeficientes da FFT de maior magnitude.

Como pode ser visto, os coeficientes mais significativos da FFT do triângulo são mais concentrados perto do centro, significando que a maior parte da informação está nas baixas freqüências. Já os coeficientes mais significativos da FFT do quadrado se encontram mais espalhados, significando informação relevante tanto



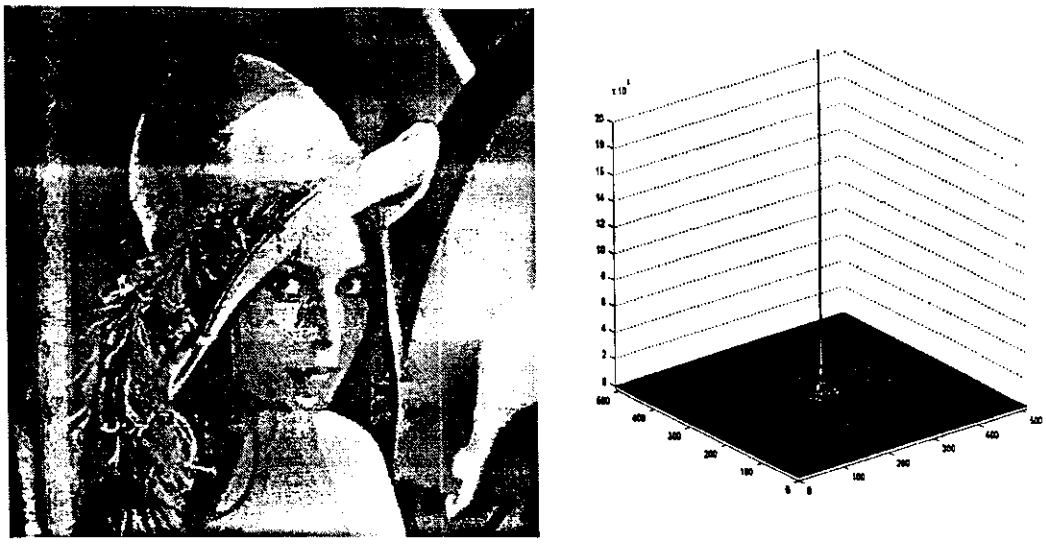
(a) Imagem Original. (b) Imagem 3D da Magnitude da FFT.

Figura 5.13: Transformada de Fourier para a Imagem da Lena de Dimensões 512x512

na baixas freqüências, quanto nas altas. Pode-se concluir que, aproximar uma imagem por uma soma ponderada de quadrados deslocados introduz mais harmônicas de altas freqüências do que uma aproximação produzida por triângulos. Esta maior presença de harmônicas indesejáveis na aproximação implica em maior distorção da imagem resultante.

Consideremos agora a FFT da imagem Lena original, de dimensões 512x512, que pode ser vista na Figura 5.13, como função da intensidade da magnitude em 3D. A Figura 5.14(a) apresenta a mesma Lena comprimida e reconstruída pelo algoritmo da *Tritree*, ou seja, aproximada por triângulos, com interpolação linear, teste de homogeneidade original e limiar 50. Sua FFT também como função da intensidade da magnitude em 3D, aparece na Figura 5.14(b). A Figura 5.15(a) exemplifica a Lena 512x512, comprimida pelo algoritmo da *Quadtree*, ou seja, reconstruída ou aproximada por quadrados, com teste de homogeneidade original e limiar 50. A sua FFT em 3D pode ser observada na Figura 5.15(b).

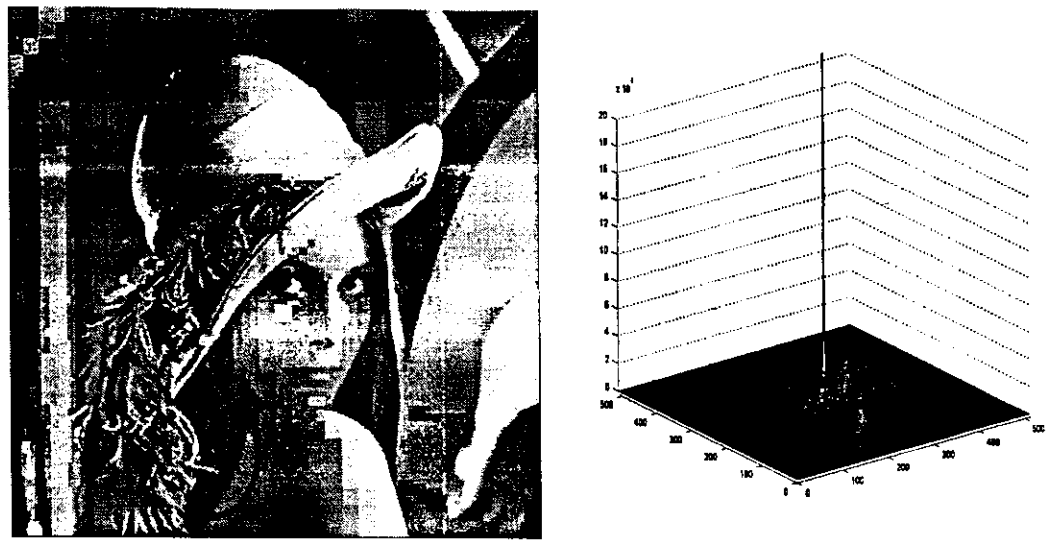
Comparando-se as imagens das FFTs, percebe-se que a FFT da imagem com-



(a) Lena Reconstruída por Triângulos (b) Imagem 3D da Magnitude da FFT

Figura 5.14: *Tritree*: Transformada de Fourier da Lena de Dimensões 512x512, com Limiar 50, Teste de homogeneidade Original e Interpolação Linear.

primida por *Tritree* (Figura 5.14(b)), assemelha-se mais à FFT da Lena original (Figura 5.13(b)), do que a FFT da Lena comprimida por *Quadtree* (Figura 5.15(b)).



(a) Lena Reconstruída por Quadrados. (b) Imagem 3D da Magnitude da FFT.

Figura 5.15: *Quadtree*: Transformada de Fourier da Lena de Dimensões 512x512, com Limiar 50 e Teste de homogeneidade original.

A afirmação acima é melhor compreendida analisando-se a Figura 5.16. Nesta figura, o erro entre as FFTs da Lena original e da Lena comprimida por *Tritree*, é calculado e apresentado em três dimensões na Figura 5.16(a), e em duas dimensões (plano Y-Z), para uma melhor visualização, na Figura 5.16(b). O mesmo é feito para o erro entre as FFTs da Lena original e da Lena comprimida por *Quadtree*, sendo os resultados apresentados nas Figuras 5.16(c) e 5.16(d).

Fazendo-se uma comparação das Figuras 5.16(a) e 5.16(b) com as Figuras 5.16(c) e 5.16(d), respectivamente, percebe-se que no segundo caso existe muito mais informação do que no primeiro, ou seja, a imagem da Lena reconstruída por quadrados, tem um erro maior do que a reconstruída por triângulos.

Esta análise fornece uma justificativa empírica para a diferença de desempenho observada nos experimentos relatados, favorável à *Tritree*. Basicamente, os melhores valores de PSNR obtidos com a *Tritree*, para taxas de bits equivalentes, deve-se ao fato de triângulos introduzirem menos harmônicas indesejáveis do que quadrados.

5.8 Conclusão

Este capítulo apresentou os resultados experimentais para o método apresentado no capítulo anterior. Como foi mostrado, a utilização de formas de codificação para a informação da árvore *Tritree* e a abordagem com *wavelets* melhoraram significativamente os resultados, justificando sua utilização permanente.

Tabela 5.28: Resultados Encontrados na Literatura.

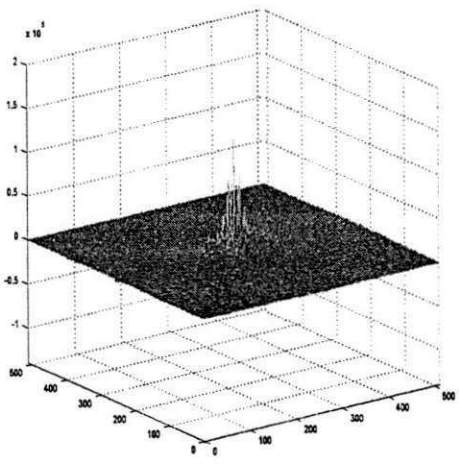
Método	Taxa (bpp)	PSNR (dB)
BSP Tree [50]	0,1	25,6
Wavelet [32]	0,21	29,11
Kim et alli [52]	0,25	30,0
JPEG [50]	0,25	30,81
Varsey e Gersho [52]	0,28	30,20
Barbara e Alcaim [40]	0,53	31,48
EZW [25]	0,39	29,39

Apenas a título de ilustração, a Tabela 5.28 apresenta alguns resultados encontrados na literatura, para técnicas de compressão de imagens. Todos estes trabalhos já foram citados ao longo do texto, sendo que alguns de maneira resumida. Estes resultados não podem ser comparados diretamente com os obtidos pela *Tritree*, pois estes trabalhos estão na forma final, isto é, são sempre compostos de pré e/ou pós-processamento, além de, sua maior parte, consistirem de abordagens híbridas, ao contrário da *Tritree*, conforme apresentado neste trabalho. Entretanto, podemos observar que, mesmo em sua forma mais básica, o algoritmo *Tritree* produz resultados competitivos com outras técnicas.

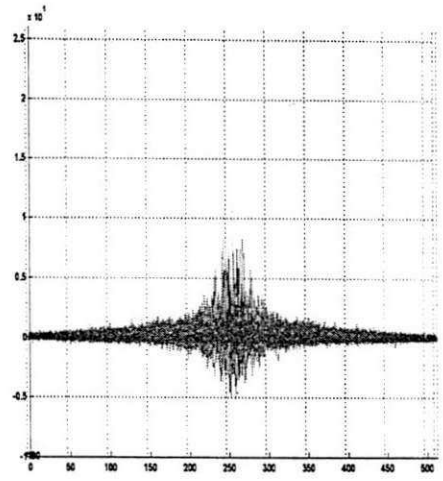
Na Figura 5.17 podemos ver exemplos de imagens comprimidas/reconstruídas com os algoritmos *Tritree* e *Quadtree*. A imagem "Lena" original (Figura 5.17(a)) tem dimensões de 512x512 *pixels* com 256 níveis de cinza. As Figuras 5.17(b) e 5.17(c)

foram reconstruídas com interpolação linear (IP). Replicação da média (RM) foi utilizada nas Figuras 5.17(d) e 5.17(e). Nas Figuras 5.17(f) e 5.17(g) foi utilizado o algoritmo da *Quadtree*. Para as Figuras 5.17(b), 5.17(d) e 5.17(f) o valor de limiar fixo empregado foi igual a 15 e para as Figuras 5.17(c), 5.17(e) e 5.17(g) o valor de limiar fixo empregado para o teste de homogeneidade durante a decomposição foi igual a 30. Pode-se perceber que o resultado final é fortemente dependente do valor de limiar, ou seja, a medida que o valor do limiar aumenta, aumenta também a degradação da imagem reconstruída, e diminiu os valores de PSNR e taxa de compressão bits por *pixels*.

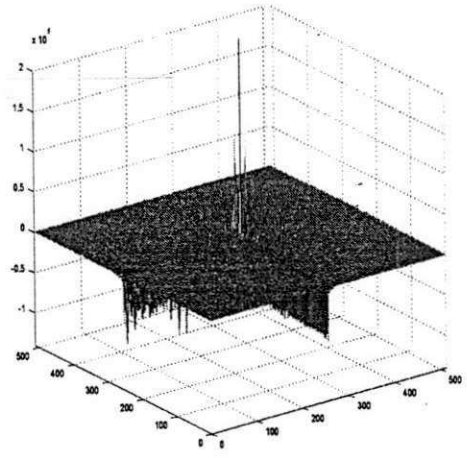
A seção 5.7 teve como objetivo justificar os melhores resultados obtidos com o algoritmo da *Tritree* em relação aos obtidos com *Quadtree*, mostrando que aproximar uma imagem por triângulos, como faz a *Tritree*, introduz menos distorção do que faze-lo por quadrados, como no caso da *Quadtree*.



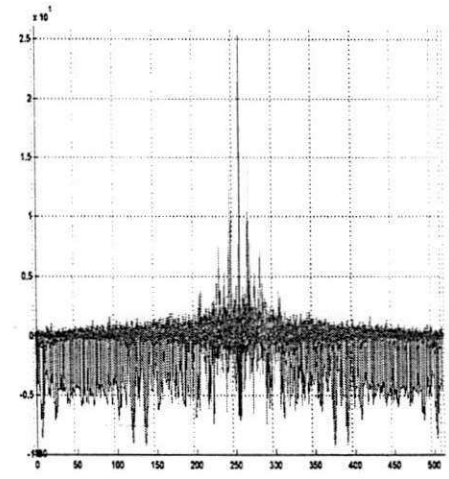
(a) Entre Lena Original e Aproximada por Triângulos (Visualização em 3D).



(b) Entre Lena Original e Aproximada por Triângulos (Visualização em 2D).



(c) Entre Lena Original e Aproximada por Quadrados (Visualização em 3D).

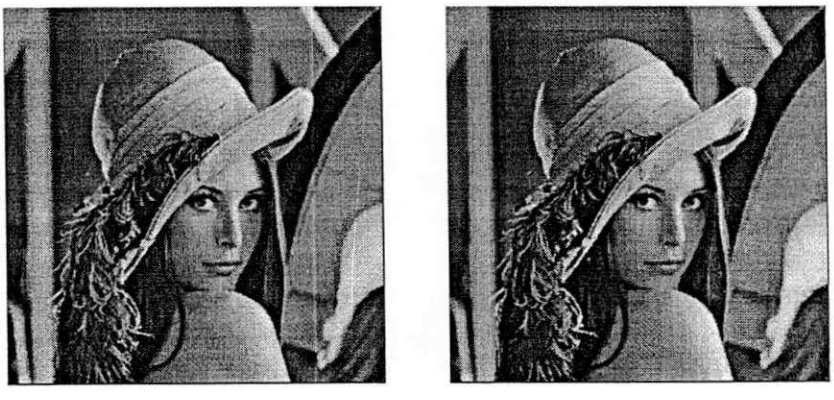


(d) Entre Lena Original e Aproximada por Quadrados (Visualização em 2D).

Figura 5.16: Erros no Domínio da Freqüência.

(a) Imagem Original (b) TT: IP com Limiar 15 (c) TT: IP com Limiar 30

(d) TT: RM com Limiar 15 (e) TT: RM com Limiar 30



(f) QT com Limiar 15 (g) QT com Limiar 30

Figura 5.17: Exemplos de Imagens Reconstruídas.

Capítulo 6

Conclusões, Contribuições e Sugestões de Trabalhos Futuros

6.1 Conclusões

Com o crescente uso de imagens digitais, envolvendo armazenamento e transmissão, tornou-se imperativa a adoção de métodos de compressão que permitam a operacionalidade da informação visual de uma forma eficiente.

Esta tese apresenta um novo método para decomposição de imagens bidimensionais (2D), em tons de cinza, baseado na decomposição *Tritree*. A meta essencial desta decomposição é descobrir e classificar regiões triangulares da imagem, de acordo com seu grau de homogeneidade.

O método foi desenvolvido e implementado no Laboratório de Automação e Processamento de Sinais - LAPS, do Departamento de Engenharia Elétrica da Universidade Federal de Campina Grande. Para este fim, foi utilizado o sistema operacional Linux com distribuição *Slackware* . Os algoritmos foram implementados na linguagem de programação C - ANSI, e compilados pelo GCC da GNU.

Foram utilizadas nos testes 10 imagens reais bidimensionais, todas com 256 níveis de cinza, sendo nove de tamanho 256x256 (linhas x colunas) e uma com dimensões 512x512. Estas imagens são as mesmas comumente utilizadas nos testes dos algoritmos de compressão de imagens, encontrados na literatura a respeito, garantindo, assim, a legitimidade das eventuais comparações entre o método ora desenvolvido e outros.

Dentre as principais características do método, merecem destaque:

- *Decomposição em formas regulares:* segmentar uma imagem em regiões de formas regulares em relação a fazê-lo em regiões arbitrárias, provê algumas vantagens. Formas fixas fornecem uma segmentação mais rápida, já que são necessários menos cálculos para cada subdivisão por se saber *a priori* como será dividida a imagem, sendo, portanto, desnecessários quaisquer testes em busca da melhor forma de fazê-lo. O custo associado a cada informação de partição também diminui, pois o montante de informação passada ao descompressor é menor, prescindindo de informações sobre a forma de cada particionamento.

- *Decomposição em regiões triangulares:* a imagem original é dividida, recursivamente, em regiões triangulares, de acordo com seu grau de homogeneidade. Divisões recursivas em regiões retangulares sempre geram mais vértices, quando comparadas às divisões triangulares, considerando um mesmo número de partições [28]. Isso é particularmente importante, quando da utilização destes vértices para a reconstrução da imagem. Outra vantagem apresentada nesta tese é que aproximar uma imagem por triângulos introduz menos harmônicas quando comparado ao montante introduzido por uma aproximação por quadrados, considerando o mesmo sinal original (Seção 5.7), implicando em menos distorção.

- *Utilidade da saída da decomposição:* a decomposição pode resultar em um tipo de segmentação de imagens. Esta segmentação pode ser utilizada para uma

variedade de diferentes aplicações, como, por exemplo, em reconhecimento de padrões. De uma maneira geral, estas estruturas podem ser amplamente utilizadas por algoritmos de processamento de imagens, que tenham por escopo tirar proveito da sua grande capacidade descritiva de regiões homogêneas e de identificar bordas dos objetos presentes na cena.

Melhores resultados em compressão de imagens digitais são obtidos com métodos híbridos. Dentre estes, aqueles que combinam a decomposição em *Quadtree* com técnicas de de pré e pós-processamento e que codificam o resultado da decomposição, geram baixas taxas de bits com valores aceitáveis de PSNR.

Como os resultaos obtidos com a decomposição *Tritree* são bastante superiores aos da *Quadtree*, pode-se afirmar que a utilização da decomposição *Tritree* nos mesmos métodos híbridos de compressão de imagens que fazem uso da *Quadtree*, deve gerar resultados superiores. Isto posto, a gama de possíveis aplicações da *Tritree* fica consideravelmente ampliada.

Portanto, a principal contribuição desta tese é apresentar um método de decomposição que representa uma alternativa vantajosa à *Quadtree*. A aplicação mais imediata deste método seria em algoritmos híbridos de compressão de imagens. Entretanto, várias outras aplicações que fazem uso de decomposição, também seriam viáveis.

6.2 Sugestões de Trabalhos Futuros

Os resultados alcançados mostram que o método, além de original, é eficiente e pode ter uma utilização mais ampla. Baseado nas investigações relatadas, é possível pesquisar outras formas de melhorar o método proposto. Portanto, as seguintes etapas de trabalhos futuros são sugeridas:

1. *Investigar outros testes de homogeneidade*: creditar ou não ao triângulo o status de homogêneo é fundamental para a definição da própria estrutura da árvore,

influenciando diretamente tanto na taxa de compressão quanto na PSNR. Por isso, o conceito de homogeneidade de triângulos carece de mais atenção, merecendo mais pesquisas a respeito.

2. *Diferentes famílias de wavelets*: investigar outras famílias de wavelets para a suavização das bordas dos triângulos. Investigar também outras formar de atenuar as componentes da decomposição para obtenção de uma melhor suavização sem perda da informação de contorno, como a DCT e outras transformadas.

3. *Reconstrução dos triângulos*: investigar outras formas de reconstrução dos triângulos tentando melhorar a qualidade da imagem reconstruída. Estas formas altenativas de reconstrução devem fazer uso da informação dos vértices para garantir que seja passado ao descompressor menos informação.

4. *Avaliação subjetiva*: relizar testes de avaliação subjetivos com as imagens originadas da diferentes configurações possíveis de execução do método para confirmar qual configuração produz melhores resultados e em quais circunstâncias, do ponto de vista de um usuário humano.

5. *Desenvolvimento 3D*: estender a decomposição *Tritree* para 3D, onde a imagem passaria a ser subdividida recursivamente em tetraédros. Possíveis aplicações seriam em holografias, compressão de vídeo (següência de imagens) e imagens médicas, como, por exemplo, tomografias computadorizadas.

Bibliografia

[1] Robert D. Dony e Simom Haykin . Neural Network Approaches to Image Compression. *Proceedings of the IEEE*, 83(2):288–303, Fevereiro 1995.

[2] Vânia Cordeiro da Silva. Identificação De Vértices Em Imagens Multi–Vista De Formas 3D. , Julho 1999. Dissertação de Mestrado - COPIN/CCT/UFPB.

[3] Vânia Cordeiro da Silva. *Compressão de Imagens Utilizando Estruturas Tritree.* Relatório Técnico COPELE/CCT/UFPB., 2000.

[4] Fernando M. B. Pereira. Video Analysis and Coding for the Emerging MPEG4 Standard. *I Conferência Nacional de Telecomunicações, Aveiro – Portugal,* Abril 1997.

[5] Vânia Cordeiro da Silva. *Normas de Compressão de Imagens.* Relatório Técnico Final da Disciplina Técnicas de Compressão de Imagens COPIN/UFPb, 1997.

[6] Rafael C. Gonzales e Richard E. Woods . *Digital Image Processing.* Addison–Wesley Publishing Company,, 1993.

[7] Fernando M. B. Pereira. A Digitalização e a Compressão na Era da Convergência. *Site: www.img.lx.it.pt/fp/artigos.*

[8] Lina J. Karam . *Lossless Coding.* Artigo em Handbook of Image and Video Processing, editor Al BoviK, Academic Press, 2000.

[9] Anil K. Jain. Image Data Compression: A Review. *Proceedings of IEEE,* 69(3):349–389, Março 1981.

[10] Vânia Cordeiro da Silva. *Reconstrução de Imagens comprimidas por Decomposição Tritree.* Relatório Técnico COPELE/CCT/UFPB., 2000.

[11] Limin Wang e Goldberg Morris . Progressive Image Transmission Using Vector Quantization on Images in Pyramid Form. *IEEE Transaction Communications,* 37(12):1339–1349, December 1989.

[12] Francisco Madeiro Bernardino Junior. Quantização Vetorial Aplicada à Compressão de Sinais de Voz e Imagem, Maio 1998. Dissertação de Mestrado COPELE/CCT/UFPB.

[13] Sven O. Wille. Structured Tri-Tree Search Method for Generation of Optimal Unstructured Finite Element Grids in Two and Three Dimensions . *International Journal for Numerical Methods in Fluids,* 14:861–881, 1992.

[14] Zixiang Xiong e Kannan Ramchandran . *Wavelet Image Compression.* Artigo em Handbook of Image and Video Processing, editor Al BoviK, Academic Press, 2000.

[15] Stephen Wong, Loren Zarenba, David Gooden e K. H. Huang. Radiolagic Image Compression – A Review. *Proceedings of the IEEE,* 83(2):194–219, February 1995.

[16] M. M. Reid, R. J. Millar e D. N. Black. Second–Generation Image Coding: a Overview. *ACM Computing Surveys,* 29(1):3–9, Março 1992.

[17] Francisco Madeiro. *Avaliação de Desempenho de Algoritmos para Projeto de Quantizadores Vetoriais Aplicados à Compressão de Imagens.* Relatório Técnico 177/98 COPELE/CCT/UFPB., 1998.

[18] F. Madeiro, J. M. Fechine, B. G. Aguiar Neto e M. S. Alencar . On the Performance of Objetive Quality Measures for Evaluating Vector–quantized Images . *Simpósio Brasileiro de Telecomunicações.*

[19] A. M. Eskicioglu e S. P. Fischer . Image Quality Measures and Their Performance. *IEEE Transaction on Communications*, 43(12):2959–2965, Dezembro 1995.

[20] Eli Shusterman e Meir Feder. Image Compression via Improved Quadtree Decomposition Algorithms. *IEEE Transaction on Image Processing*, 3(2), Março 1994.

[21] Sriram Sethuraman. Stereoscopic Image Sequence Compression Using Multiresolution and Quadtree Decomposition Based Disparity and Motion–Adaptive Segmentation.

[22] Murat Kunt e Michel Kocher. Second-Generation Image-Coding Techniques . *Proceedings of IEEE*, 73(4):549–574, April 1985.

[23] Hans G. Musmann, Peter Pirsch and Hans-Joachim Grallert . Advances in Picture Coding. *Proceedings of IEEE*, 73(4):523–548, April 1985.

[24] Gilmarcos R. Corrêa. Compressão de Imagens Usando a Transformada Wavelet. *Universidade Federal de Uberlândia*, Agosto 1996. Dissertação de Mestrado.

[25] Jerome M. Shapiro. Embedded Image Coding Using Zerotrees of Wavelet Coefficients. *IEEE Transaction on Signal Processing*, 41(12):3445–3462, December 1993.

[26] S. Sahni e B. C. Vemuri e F. Chen e C. Kapoor e C. Leonard e J. Fitzsimmons . State of Art Lossless Image Compression Algorithms. *Document Submited to the IEEE Transaction on Image Processing for Review*, October 1997.

[27] Sriram Sethuraman, M. W. Siegel e Angel G. Jordan . A multiresolutional region based segmentation scheme for stereoscopic image compression. *SPIE Digital Video Compression Algorithms and Tecnologies*, 2419, 1995.

[28] Joceli Mayer. *Blending Models for Image Enhancement and Coding* . PhD thesis, University of California, Santa Cruz, December 1999.

[29] J. Jiang, E. A. Edirisinghe e H. Schroder . A 3–D Image Compression System Using JPEG. *IPA97 – Conference Publication*, 15–17(443):81–85, July 1997.

[30] Stephane G. Mallat. Multifrequency Channel Decompositions of Images and Wavelet Models. *IEEE Transactions on Acoustics, Speech, and Signal processing*, 37(12):2091–2110, December 1989.

[31] Suzete Élida Nóbrega Correia. Reconhecimento de Caracteres Numéricos Manuscritos Usando a Transformada Wavelet, Junho 2000. Dissertação de Mestrado COPELE/DEE/UFPB.

[32] P. Barlau, P. Mathieu, M. Antonini e I. Daubechies . Image Coding Using Using Wavelet Transform . *IEEE Transaction on Image Processing*, 1(2):205–220, Abril 1992.

[33] Antonio Ortega e Kannan Ramchandran . Rate-Distortion Methods for Image and Video Compression. *IEEE Signal Processing Magazine*, 15(6):23–50, November 1998.

[34] Michael W. Marcellin, Michael J. Gormish, Ali Bilgin e Martin P. Boliek . An Overview of JPEG-2000.

[35] Francisco Madeiro Bernadino Junior. Quantizacao Vetorial de Voz e Imagem . *Universidade Federal da Paraíba, Departamento de Engenharia Eletrica*, Dezembro 2001. Campina Grande – PB. Tese de Doutorado.

[36] E. Shusterman e M. Feder . Image compression via Quadtree decomposition of Wavelets . *Proceedings of ASILOMAR–29*, pages 1424–1428, 1996.

[37] A. Munteanu, Jan Cornelis, G. Van der Auwera e Paul Cristea . Wavelet Image Compression – The Quadtree Coding Approach. *IEEE Transactions on Information Technology in Biomedicine*, 3(3):176–185, September 1999.

[38] James W. Modestino, David G. Daut, Acie L. Vickers . Combined Source-Channel Coding of Images Using the Block Cosine Transform . *IEEE Transactions on Commuications*, COM–29(9):1261–1274, Setember 1981.

[39] Vinay A. Vaishampayan e Nariman Farvardin . Optimal Block Cosine Transform Image Coding for Noise Channels. *IEEE Transactions on Communications*, 38(3):327–336, March 1990.

[40] Rodolfo M. B. Bárbara e Abraham Alcaim . Compressão de Imagens Baseada em Decomposição Quadtree com Limiares Variáveis, Alocação de bits Adaptativa e DCT. volume 38, Março 1999.

[41] Robert M. Gray. Vector Quantization. *IEEE ASSP Magazine*, pages 4–29, Abril 1984.

[42] Nasser M. Nasrabadi e Robert A. King . Image Coding Using Vector Quantization: A Review. *IEEE Transactions on Communications*, 36(8):957–971, Agosto 1988.

[43] Bhaskar Ramamurthi e Allen Gersho . Classified Vector Quantization of Images . *IEEE Transaction on Communications*, COM – 34(11):1105–1115, Novembro 1986.

[44] M. E. Blain e T. R. Fischer . A Comparison of VQ Techniques . *Eurasip - image Commun*, 3(1), Fevereiro 1991.

[45] O. Egger, Wei Li e Murat Kunt . High Compression Image Coding Using an Adaptive Morphological Subband Decomposition. *Proceedings of the IEEE*, 83(2):272–287, Fevereiro 1995.

[46] John W. Woods e Sean D. O'Neil . Subband Coding of Images. *IEEE Transaction on Acoustics, Speech e Signal Processing*, ASSP – 34(5):1278–1288, October 1986.

[47] Gary J. Sullivan e Richard L. Baker . Efficiente Quadtree Coding of Images and Vídeo . *IEEE Transaction on Image Processing*, 3(3), May 1994.

[48] Chia-Yuan Teng e David L. Neuhoff . A New Quadtree Predictive Image Coder, Anais do ICIP . 1995.

[49] Richard Szeliski e Heung-Yeung Shum . Motion Estimation with Quadtre Splines. *IEEE Transaction on Pattern Analisis and Machine Intelligence*, 18(12):1199–1210, December 1996.

[50] Hayder Radha, Martin Vetterli e Riccardo Leonardi . Image Compression Using Binary Space Partitioning Trees . *IEEE Transaction on Image Processing*, 5(12):1610–1624, Dezembro 1996.

[51] Vânia Cordeiro da Silva. *Compressão de Imagens: um Estudo Sobre Classificação, Abordagens, Medidas Avaliativas e Técnicas.* Relatório Técnico COPELE/CCT/UFPB., 2002.

[52] Jacques Vaisey e Allen Gersho . Image Compression with Variable Block Size Segmentation . *IEEE Transaction on Signal Processing*, 40(8):2040–2060, Agosto 1992.

[53] Xiaolin Wu. Image Coding by Adaptive Tree-Structured Segmentation. *IEEE Transaction on Information Theory*, 38(6), November 1992.

[54] P. Burt e E. Adelson . The Laplacian Pyramid as a Compact Image Code . *IEEE Transaction on Communication*, COM – 31(4):532–540, 1983.

[55] Fernando M. B. Pereira. A Revolução Multimídia: O papel da Normalização. *Site: www.img.lx.it.pt/fp/artigos.*

[56] R. Koenen, F. Pereira e L. Chiariglione . MPEG-4: Context and Objetives. *Signal Processing: Image Communication - Special edition about MPEG-4*, 9, Maio 1997.

[57] Ellis Horowitz e Sartaj Sahni . *Fundamentos de Estrutura de Dados.* Editora Campus Ltda,, 1984.

[58] Herbert Schildt. *C Completo e Total.* McGraw-Hill e Makron Books do Brasil Ltda, 1991. Tradução de Marcos Ricardo Morais.

[59] Y. A. Rozanov. *Propability Theory - A Concise Course.* Dover Publications, Inc, New York, 1977. Revised English Edition Translated and Edited by Richard A. Silverman.

[60] G. Lima Reis e V. Vilmar Silva . *Geometria Analítica – Segunda Edição.* Livros Técnicos e Científicos Editora, 1997.

[61] Roger Penrose. *A Mente Nova do Rei - Computadores, Mentes e as Leis da Física.* Editora Campus, Rio de Janeiro, 1997. Tradução de Waltensir Dutra.

[62] Simon Haykin e Barry Van Veen . *Sinais e Sistemas.* Editora Bookman, Porto Alegre, 2001. Tradução de José Carlos Barbosa dos Santos.