**Universidade Estadual de Campinas**
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO - FEEC
DEPARTAMENTO DE COMUNICAÇÕES - DECOM

# Aplicação da Transformada Wavelet na Compressão de Imagens

Mylène Christine Queiroz de Farias

Orientador:

Amauri Lopes



Dissertação submetida à Faculdade de Engenharia Elétrica e de Computação - FEEC da Universidade Estadual de Campinas - Unicamp, como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica.

26 de Junho de 1998.

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

F225a

Farias, Mylène Christine Queiroz de
Aplicação da transformada wavelet na compressão de imagens. / Mylène Christine Queiroz de Farias.-- Campinas, SP: [s.n.], 1998.

Orientador: Amauri Lopes.
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Wavelets (Matemática). 2. Processamento de imagens. 3. Compressão de dados (Telecomunicações). I. Lopes, Amauri. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

*Não sou nada,*
*Não serei nada,*
*Não posso querer nada,*
*À parte disso,*
*Tenho em mim,*
*Todos os sonhos do mundo.*

Fernando Pessoa.

Aos meus pais, **Gutemberg** e **Célia**,
que me ensinaram a sonhar.
A meu marido, **Marcelo**, que está me
ajudando a realizar estes sonhos.

# Agradecimentos

Gostaria de agradecer a todos que, de forma direta ou indireta, contribuíram para a realização deste trabalho.

Agradeço ao meu orientador, Professor Doutor Amauri Lopes, cuja competência e dedicação foram essenciais para o êxito deste trabalho. A sua simplicidade, clareza e objetividade influenciaram definitivamente o meu modo de trabalhar. Agradeço especialmente o apoio que me deu durante estes anos. Sem sua ajuda a realização de um grande sonho estaria comprometida.

Meus agradecimentos ao Prof. Dr. Dalton Soares Arantes pelas sugestões e pelo apoio dado a mim e ao meu marido durantes os anos de mestrado. Agradeço também ao Prof. Max Costa a sua ajuda em questões na área de compressão de imagens.

Agradeço ao Engenheiro Luiz Rômulo Mendes, que me cedeu as rotinas de implementação do código *run-lenght,* e ao Engenheiro Fabbryccio Akkazzha C. M. Cardoso, que me auxiliou com a linguagem C. Agradeço também às amigas, Maria Luiza, Jhuli Meire, Anna Catharina e Alexandra, que por muito tempo foram a minha família aqui em Campinas.

Aos funcionários Juracir, Mário, Lúcia e Noêmia meus agradecimentos pela sua ajuda.

Agradeço à Fundação Coordenação de Aperfeiçoamento do Pessoal de Nível Superior - CAPES pelo suporte financeiro concedido a este trabalho desde seu início. Meus agradecimentos também ao CPqD - Telebrás pelo apoio concedido através do convênio Unicamp/ Telebrás.

Agradeço ao meu pai Gutemberg, meu maior incentivador, que sempre me questionou, me apoiou e sofreu comigo nos momentos difíceis. À minha mãe Célia, que esteve sempre presente em todos os momentos da minha vida, obrigada pelo carinho, pela torcida e, principalmente, pelas palavras de ânimo nos momentos difíceis. À minha irmã, Michelle, obrigada pelo carinho e atenção que sempre demonstrou.

Agradeço muitíssimo ao meu marido Marcelo, que me auxiliou na redação deste trabalho com críticas e sugestões. Agradeço, principalmente, o apoio e carinho que me dedicou durante todos estes anos, sem os quais eu teria sucumbido nos momentos de insegurança. A tarefa de terminar este trabalho seria infinitamente mais árdua sem a sua presença. Ao meu companheiro meu eterno amor.

Mylène Christine Queiroz de Farias
Campinas, Junho de 1998.

# Resumo

O objetivo deste trabalho foi realizar uma análise sobre as potencialidades de aplicação da transformada wavelet na compressão de imagens estáticas monocromáticas. Realizamos um estudo da teoria wavelet, com ênfase na aplicação em compressão de imagens. Realizamos também um estudo sobre as principais técnicas de compressão de imagens estáticas, detendo-nos um pouco mais no estudo dos decodificadores com-perdas. Utilizando esta bagagem, construímos alguns esquemas de compressão de imagens baseados na transformada wavelet. Através de simulações, pudemos fazer uma análise do tipo de codificador baseado na transformada wavelet mais adequado à compressão de imagens. Vislumbrando novas aplicações da transformada wavelet na área de processameno de imagens, estudamos também alguns esquemas de compressão de sinais de vídeo baseados na transformada wavelet, apresentando inclusive alguns conceitos sobre a escalonabilidade.

# Abstract

The purpose of this work is to analyze the applications of the wavelet transform in the compression of static monochromatic images. We have made a study of the wavelet theory with emphasis on its applications in image compression. We have also studied some of the most important techniques used in image compression. Based on this background a few image compression schemes have been constructed and simulated in order to evaluate their performance. Taking into account the modern telecommunications scenario, we have also made a brief study of the application of the wavelet transform in video compression, including an introduction to scalability and its relation with wavelet transform.

# Sumário

# Lista de Figuras

# Lista de Tabelas

# Capítulo 1

# Introdução

O crescimento vertiginoso da tecnologia digital nos últimos anos, tem provocado uma ampla revolução no panorama mundial das telecomunicações, criando uma série de novos serviços e permitindo a melhoria dos já existentes. Uma das áreas que mais tem evoluído é a de processamento digital de imagens, pois a utilização de imagens digitais em serviços de telecomunicações aumentou consideravelmente, dadas as possibilidades, cada vez maiores, de armazenamento e velocidade de processamento. Além disso, com o advento da Rede Digital de Serviços Integrados de Faixa Larga (RDSI - FL) abre-se a perspectiva do surgimento de novas aplicações e serviços, como, por exemplo, a transmissão de sinais de TV digital/ HDTV (High Definition TV), vídeo-conferência, vídeo-fone, banco de dados de imagens, etc.

Apesar das suas grandes vantagens, tais como maior robustez a erros de transmissão e facilidade de manuseio, a imagem no formato digital implica em um maior volume de dados, exigindo uma maior capacidade de armazenamento e de transmissão. Sendo assim, as técnicas de compressão de imagens têm sido de fundamental importância na área de processamento digital de imagens e alvo de intensas pesquisas nos últimos anos.

As formas tradicionais de compressão de imagens baseiam-se, em sua maioria, na técnica de *codificação por transformada*. Esta consiste na aplicação de transformações lineares em um sinal, produzindo uma representação mais compacta do sinal em um domínio diferente. Entre as transformadas, destaca-se a DCT (Discrete Cosine Transform), que é uma variante da DFT (Discrete Fourier transform). O padrão JPEG (Joint Picture Experts Group) é, dentre os formatos de compressão de imagens estáticas que utilizam a DCT, um dos mais populares. Além da DCT, o JPEG utiliza uma quantização escalar e uma codificação entrópica baseada na codificação "run-length". Entretanto, uma das maiores desvantagens do JPEG é a introdução de distorções na imagem, em especial, os efeitos de "blocagem" [Rabbani].

Nos últimos anos surgiu uma nova alternativa à DCT: a *transformada wavelet*. Esta é uma técnica bastante recente, que se desenvolveu especialmente nos últimos quinze anos através de contribuições de pesquisadores nas mais diversas áreas, como a matemática, a física, a estatística, a computação gráfica e a engenharia.

Assim como a transformada de Fourier, a transformada wavelet decompõe o sinal em uma base de funções. Estas funções (wavelets) são geradas a partir de deslocamentos, compressões ou dilatações de uma função básica, de suporte compacto e oscilatória, denominada de *wavelet-mãe*. Esta estrutura permite que a transformada wavelet ofereça uma excelente resolução em tempo e freqüência, permitindo a sua utilização no processamento de sinais não-estacionários, como,

por exemplo, as imagens. Outra característica da transformada wavelet é a sua alta capacidade de concentrar a energia do sinal em um número reduzido de coeficientes, possibilitando a obtenção de uma representação mais compacta. Quanto à sua complexidade computacional, esta é inferior à da FFT (Fast Fourier Transform), o que motiva a sua utilização em sistemas práticos. Por fim, a transformada wavelet apresenta a vantagem de não introduzir os efeitos de "blocagem". O conjunto destas características fazem desta transformada uma excelente alternativa aos métodos tradicionais de compressão de imagens.

O objetivo deste trabalho é apresentar uma análise sobre as potencialidades de aplicação da transformada wavelet na compressão de imagens estáticas monocromáticas. Realizamos um estudo da teoria wavelet, enfatizando os pontos importantes para a área de compressão de imagens. Realizamos também um estudo sobre as principais técnicas de compressão de imagens estáticas, detendo-nos um pouco mais no estudo dos decodificadores com perdas. Utilizamos esta bagagem, para desenvolver alguns esquemas de compressão de imagens baseados na transformada wavelet, sendo alguns destes inéditos na literatura. Realizamos simulações para avaliação comparativa do desempenho de tais esquemas. Vislumbrando novas aplicações da transformada wavelet na área de processamento de imagens, estudamos também alguns esquemas de compressão de sinais de vídeo baseados na transformada wavelet, apresentando inclusive alguns conceitos sobre escalonabilidade.

Os capítulos foram, então, divididos da seguinte forma:

**Capítulo 2: Wavelets**

Neste capítulo apresentamos uma introdução à teoria wavelets. Fazemos um estudo detalhado dos aspectos da transformada wavelet que julgamos importantes para a compressão de imagens. Primeiramente, realizamos uma comparação entre a análise de Fourier, a "Short Time Fourier Transform" e a transformada wavelet. A seguir, definimos as transformadas wavelet contínua e discreta e fazemos uma breve introdução à teoria de frames. Apresentamos, então, um estudo da análise de multiresolução, uma teoria desenvolvida por Mallat que deu origem ao algoritmo utilizado para calcular a transformada wavelet discreta. Em seguida, apresentamos as wavelets biortogonais e a transformada wavelet bidimensional. Finalizamos este capítulo com alguns exemplos da utilização da transformada wavelet bidimensional, onde realizamos uma comparação visual entre o desempenho das wavelets ortogonais e biortogonais.

**Capítulo 3: Compressão de Imagens**

Um estudo sobre as principais técnicas de compressão de imagens foi realizado neste terceiro capítulo. Descrevemos alguns modelos de compressão com perdas e sem perdas, detalhando as diversas técnicas que podem ser empregadas nestes codificadores, como os tipos de quantizadores e codificadores entrópicos. Em seguida, apresentamos algumas das principais técnicas de codificação utilizadas atualmente, dando ênfase à codificação por transformada e por subbandas. Apresentamos também um exemplo numérico de uma codificação com perdas de um bloco $8 \times 8$ de uma imagem utilizando o padrão JPEG.

### Capítulo 4: Compressão de Imagens Utilizando a T.W.

Neste capítulo apresentamos modelos de codificadores com perdas baseados na transformada wavelet. Através de simulações, analisamos o desempenho destes modelos em função de parâmetros como o tipo de wavelet, o número de camadas da decomposição, o tipo de quantizador escalar e de varredura utilizado em cada sub-banda. No final do capítulo, são apresentadas as comparações entre os sistemas propostos.

### Capítulo 5: Técnicas Avançadas

A transformada wavelet também pode ser aplicada na compressão de sinais de vídeo. Esta aplicação é uma conseqüência dos bons resultados obtidos na compressão de imagens estáticas. No Capítulo 5 apresentamos um breve estudo sobre as possibilidades de aplicações da transformada wavelet nesta área. Apresentamos também uma introdução aos conceitos de escalonabilidade. A compressão de sinais de vídeo utilizando algoritmos de codificação escalonáveis oferecem uma série de funcionalidades, como a compatibilidade e a interoperabilidade. As wavelets se mostram adequadas à implementação de codificadores deste tipo, uma vez que geram naturalmente sinais com escalonabilidade espacial.

### Capítulo 6: Conclusões e Sugestões para Trabalhos Futuros

Finalizamos este trabalho com um balanço das conclusões e resultados obtidos, vislumbrando as perspectivas de continuação e avanços sobre a exposição aqui efetuada.

# Capítulo 2

# Wavelets

A teoria wavelet é bastante recente e se desenvolveu especialmente nos últimos quinze anos através de contribuições de pesquisadores das mais diversas áreas como matemática, física, estatística, computação gráfica e engenharia. Sob o ponto de vista de processamento de sinais, podemos pensar na transformada wavelet como uma alternativa para as técnicas tradicionais de análise de sinais, como a análise de Fourier e a "Short Time Fourier Transform". Bons resultados têm sido obtidos em diferentes áreas como compressão de imagens, processamento de sinais, processamento de voz, visão computacional e outras [Strang96] [Rioul] [Morettin].

O objetivo deste capítulo é dar uma introdução à teoria wavelets. Antes de definir formalmente a transformada wavelet, discutiremos as principais diferenças entre esta técnica e os métodos tradicionais de análise de sinais. Em seguida definiremos a forma contínua e discreta da transformada wavelet, dando uma pequena introdução à teoria de frames. Discutiremos, também a análise de multiresolução e apresentaremos os algoritmos de decomposição e síntese da transformada wavelet para o caso ortogonal e biortogonal. Finalizamos este capítulo com a generalização de toda este teoria para o caso bidimensional. Não apresentaremos neste trabalho, as demonstrações dos teoremas. Maiores detalhes sobre este assunto podem ser encontrados em [Daubechies92], [Chui92], [Vetterli95] e [Burrus].

## 2.1 Fourier e Wavelets

O objetivo da análise de sinais é extrair informação relevante de um sinal. Em geral, isto é feito através de uma transformação. A análise de Fourier é a ferramenta clássica e decompõe o sinal na base ortonormal formada pelas funções $e^{-j2\pi ft}$. A Transformada de Fourier (T F) de um sinal $f(t)$ é dada por:

$$F(f) = \int_{-\infty}^{\infty} f(t) \cdot e^{-j2\pi ft} dt \tag{2.1}$$

$F(f)$ representa o espectro médio a partir do qual podemos analisar as características do sinal. Entretanto, a TF de Fourier não é adequada para a análise de sinais não-estacionários e não é eficiente para a análise de sinais de duração finita, pois sua base tem duração infinita.

Esta deficiência foi observada por D. Gabor em 1946 [Gabor]. Para tentar amenizá-la, Gabor criou uma nova transformada através da introdução de uma "função janela" $g(t-\tau)$ na

transformada de Fourier, com o objetivo de extrair informação local do sinal. A transformada de Gabor utiliza a janela gaussiana, por motivos que ficarão mais claros nas próximas seções. Se generalizarmos a transformada de Gabor permitindo que $g(t)$ seja uma janela qualquer, obteremos uma nova transformada conhecida como "Short Time Fourier Transform" (STFT). Considerando um sinal $f(t)$ estacionário quando visto através da janela $g(t-\tau)$, definimos a sua STFT como:

$$STFT(\tau, f) = \int_{-\infty}^{\infty} f(t) \cdot g^*(t-\tau) \cdot e^{-j2\pi ft} dt \tag{2.2}$$

onde $g^*(t)$ é o complexo conjugado de $g(t)$.

Como o comprimento da janela $g(t)$ é fixo, independente do valor de $\tau$ ou $f$, a STFT não é uma técnica adequada para analisar sinais que apresentem ao mesmo tempo componentes de altas e baixas freqüências.

A transformada wavelet (TW) oferece uma solução para esta restrição da STFT. A wavelet $\psi(t)$, que é uma função com um papel semelhante ao da janela $g(t)$ na STFT, tem uma duração que pode ser ajustada em função do conteúdo de freqüência do sinal a ser analisado ou da ênfase que se deseja colocar na análise das baixas ou altas freqüências do sinal.

Seja a wavelet $\psi(t)$ uma função oscilatória e de curta duração. Como exemplo de uma wavelet, apresentamos na Figura 2.1 a wavelet de Morlet [Morlet]. A transformada wavelet contínua (TWC) é, então, definida pela seguinte expressão:

$$TWC(a,b) \triangleq a^{-1/2} \int_{-\infty}^{\infty} f(t) \cdot \psi^* \left( \frac{t-b}{a} \right) dt \tag{2.3}$$

Em (2.3), vemos que o parâmetro **b** é responsável pelo deslocamento de $\psi(t)$ ao longo do eixo do tempo, enquanto que o parâmetro **a** é responsável pela compressão ou dilatação. O parâmetro **a**, denominado escala, proporciona flexibilidade a $\psi(t)$, permitindo que a sua largura aumente ou diminua de acordo com o desejado. Além disso, é importante observar que a TW é expressa em termos dos parâmetros **a** e **b**. Assim, da mesma forma que a STFT, a TW depende do deslocamento da função janela. Porém, de forma distinta, não envolve implicitamente o conceito de freqüência, mas sim apresenta o resultado da análise para uma gama de valores de **a**, ou seja, dos valores da duração da "janela" $\psi(t)$. Apesar de haver uma relação entre escala e freqüência, estes conceitos são diferentes, o que modifica a forma de interpretação da TW em relação à STFT. A TW permite uma visualização simultânea dos resultados da transformada para os diversos valores de escala. Na próxima seção veremos os conceitos de escala e deslocamento com um pouco mais de detalhes.

Mas, quais as vantagens de se utilizar wavelets ao invés dos métodos tradicionais ? As funções da base na análise de Fourier são bem localizadas em freqüência, mas não em tempo. Dependendo da escolha da janela $g(t)$, que é fixa, a STFT pode apresentar uma boa localização no tempo, às custas de uma péssima localização em freqüência ou vice-versa. Já as wavelets possuem duração variável, o que significa que a análise do sinal é feita simultaneamente por todos os valores de escala e podemos escolher o mais adequado para cada parte do sinal.

Por fim, a TW se distingue pela sua capacidade de compressão de informação e pela rapidez dos seus algoritmos de implementação: muitas classes de sinais podem ser representadas de

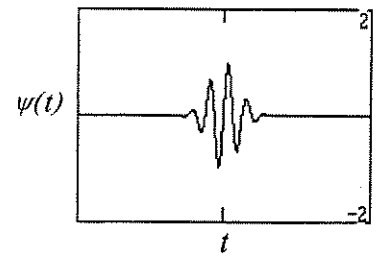



Figura 2.1: Wavelet de Morlet.

forma mais compacta através das wavelets e, enquanto que a complexidade computacional da FFT é da ordem de $O(n \cdot \log_2(n))$, a mesma é da ordem de $O(n)$ na TW.

## 2.2 Da Série de Fourier à Transformada Wavelet

O objetivo desta seção é definir a transformada wavelet (TW) e a sua inversa, partindo de uma comparação entre a representação utilizando a série de Fourier e a representação através da TW.

O espaço de funções $L^2(0, 2\pi)$ é definido como a classe de funções mensuráveis (Lebesgue), definidas no intervalo $(0, 2\pi)$, tais que:

$$\int_0^{2\pi} |f(x)|^2 \, dx < \infty \tag{2.4}$$

Toda função $f(x) \in L^2(0, 2\pi)$ possui uma representação por série de Fourier , na forma:

$$f(x) = \sum_{-\infty}^{\infty} C_n \, e^{inx} \tag{2.5}$$

onde os $C_n$ são os coeficientes de Fourier de $f(x)$, calculados através da expressão:

$$C_n = \frac{1}{2\pi} \int_0^{2\pi} f(x) \cdot e^{-inx} dx \tag{2.6}$$

A convergência da série (2.5) é definida em $L^2(0, 2\pi)$, isto é:

$$\lim_{M, N \to \infty} \int_0^{2\pi} \left| f(x) - \sum_{n=-M}^{N} C_n \, e^{inx} \right|^2 dx = 0 \tag{2.7}$$

A função $f(x)$ pode, portanto, ser decomposta em infinitas componentes ortonormais $g_n(x) = e^{inx}$, onde a ortonormalidade significa que:

$$\frac{1}{2\pi}\langle g_m, g_n\rangle = \frac{1}{2\pi}\int_0^{2\pi} g_m(x)\, g_n^*(x)\; dx = \begin{cases} 0, \text{ para todo } m \neq n \\ 1, \; m = n \end{cases}$$

Definimos $\langle g_m, g_n\rangle$ como o produto interno das funções $g_m(x)$ e $g_n(x)$.

Note que $g_n(x) = e^{inx}$ é uma base ortonormal para $L^2(0, 2\pi)$ gerada por uma única função $g(x) = e^{ix}$, através de um processo de dilatações e compressões. Valores grandes de $|n|$ correspondem a freqüências altas, enquanto que valores pequenos de $|n|$ correspondem a freqüências baixas.

Definimos o espaço de funções $L^2(\mathbb{R})$ como a classe de funções mensuráveis (Lebesgue), definidas na reta real, tais que a sua energia seja finita, ou seja:

$$\int_{-\infty}^{\infty} |f(x)|^2\, dx < \infty \tag{2.8}$$

Claramente, os espaços $L^2(\mathbb{R})$ e $L^2(0, 2\pi)$ são bem diferentes. Na busca de bases de expansão para $L^2(\mathbb{R})$, devemos nos ater a sinais que decaiam rapidamente a zero quando $x \to \pm\infty$. Assim como na série de Fourier, onde o espaço $L^2(0, 2\pi)$ é gerado a partir de dilatações e compressões de uma única função, na TW procuramos uma única função $\psi(x)$ capaz de gerar todo o espaço $L^2(\mathbb{R})$.

Da mesma forma, definimos o espaço de seqüências $l^2$ como a classe de seqüências quadrado somáveis, tais que a sua energia seja finita, ou seja:

$$\sum_{n=-\infty}^{\infty} |C_n|^2 < \infty \tag{2.9}$$

**Identidade de Parseval** A representação em série de Fourier satisfaz a seguinte propriedade:

$$\frac{1}{2\pi}\int_{-\infty}^{\infty} |f(x)|^2\, dx = \sum_{n=-\infty}^{\infty} |C_n|^2 \tag{2.10}$$

Em termos de processamento de sinais, esta propriedade significa que a energia do sinal $f(x)$ é conservada na decomposição.

> **Definição 2-1** A expressão do lado esquerdo da identidade de Parseval define a norma para $L^2(0, 2\pi)$, enquanto que a do lado direito define a norma para o espaço $l^2$. Por causa da igualdade em (2.10), os espaços $L^2(0, 2\pi)$ e $l^2$ são denominados isométricos. Note que toda função $2\pi$-periódica de energia finita pode ser expressa como uma combinação linear em $l^2$ das funções $g(nx) = e^{inx}$.

Como já dito, a função base na TF é comprimida/dilatada para que todo o espectro de freqüências possa ser gerado. Da mesma forma, na TW a função $\psi(x)$ deve ser comprimida e dilatada. Ao contrário de $e^{ix}$, a função $\psi(x)$ tem duração finita e para cobrir todo o espaço

$L^2(\mathbb{R})$ é necessário utilizar combinações lineares dos deslocamentos de $\psi(x)$ ao longo do eixo $x$.

Já definimos a transformada wavelet contínua (TWC) da função $f(x)$ como:

$$TWC(a,b) \triangleq a^{-1/2} \int_{-\infty}^{\infty} f(x) \cdot \psi^* \left( \frac{x-b}{a} \right) dx \tag{2.11}$$

Se definirmos:

$$\psi_{a,b}(x) \triangleq a^{-1/2} \cdot \psi \left( \frac{x-b}{a} \right) \tag{2.12}$$

podemos reescrever (2.11) como:

$$TWC(a,b) = \int_{-\infty}^{\infty} f(x) \cdot \psi_{a,b}^*(x)\, dt = \langle f(x), \psi_{a,b} \rangle \tag{2.13}$$

As funções $\psi_{a,b}(x)$ são as wavelets-filhotes, versões deslocadas e dilatadas ou comprimidas da wavelet-mãe $\psi(x)$ e capazes de gerar todo o espaço $L^2(\mathbb{R})$. Na expressão (2.12), **b** é o parâmetro responsável pelo deslocamento da wavelet e **a** é a escala, parâmetro responsável pela dilatação ou compressão. A constante $a^{-\frac{1}{2}}$ em (2.12) é um termo de normalização da energia da função em relação ao parâmetro **a**.

Na Figura 2.2 é apresentada a wavelet da Figura 2.1 para diversos valores de **a** e **b**. Observe que ao aumentarmos o valor de **a**, a wavelet é dilatada e a sua duração aumenta. Ao diminuirmos o valor de **a**, comprimimos a wavelet e a sua duração diminui. Para valores negativos de **b**, a wavelet é deslocada para a esquerda e para valores positivos para a direita.

Para obtermos a transformada wavelet inversa, isto é, recuperar $f(x)$ a partir dos seus coeficientes wavelet, é necessário que a seguinte relação seja satisfeita [Chui92] [Daubechies92]:

$$C_\psi \triangleq \int_{-\infty}^{\infty} \frac{|\Psi(w)|}{|w|}\, dw < \infty \tag{2.14}$$

onde $C_\psi$ é uma constante e $\Psi(w)$ é a TF de $\psi(x)$. Essa restrição é conhecida como *condição de admissibilidade* e limita a classe de funções que podem ser utilizadas como wavelets-mãe. Além desta restrição, não se pode esquecer que $\psi(x)$ é uma função janela, ou seja:

$$\int_{-\infty}^{\infty} |\psi(x)|\, dx < \infty \tag{2.15}$$

de tal forma que $\Psi(w)$ é contínua em $\mathbb{R}$ [Daubechies92].

Da expressão (2.14), vemos que $\Psi(w)$ deve se anular na origem. Assim, dado que:

$$\Psi(w) = \int_{-\infty}^{\infty} \psi(x)\ e^{-j\,2\pi f x} dx$$

a exigência:

$$\Psi(0) = 0$$

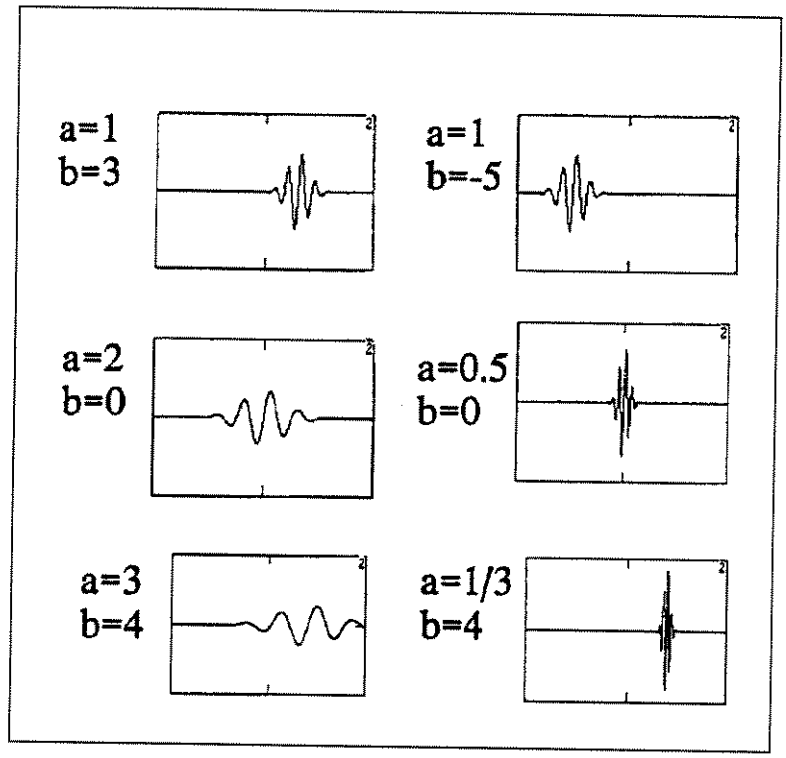



Figura 2.2: Wavelets de Morlet deslocadas e transladadas: $\Psi\left(\frac{x-b}{a}\right)$.

implica em:

$$\int_{-\infty}^{\infty} \psi(x)\ dx = 0$$

o que significa que $\psi(x)$ deve ser oscilante e ter média nula.

De forma resumida, podemos dizer que a wavelet é uma função oscilante de média nula e que tem um decaimento rápido a zero.

Se a constante $C_\psi$ em (2.14) é finita, então, existe uma TW inversa, que é definida por:

$$f(x) = \frac{1}{C_\psi} \int\int_{\mathbb{R}^2} \{TWC(a,b)\} \left\{ a^{-1/2} \psi\left(\frac{x-b}{a}\right) \right\} \frac{dadb}{a^2} \tag{2.16}$$

onde $f \in L^2(\mathbb{R})$ e $\mathbb{R}^2 = \mathbb{R} \times \mathbb{R}$.

## 2.3 Comparação entre STFT e TW

Como já visto Seção 2.1, antes de calcular a STFT de um sinal, é necessário escolher uma janela $g(t)$ de comprimento fixo. O deslocamento desta janela sobre o eixo do tempo divide o sinal em várias partes de comprimentos iguais. As listras verticais da Figura 2.3 ilustram este deslocamento. Calcular a STFT de um sinal corresponde, então, a calcular várias TF, uma para cada parte do sinal.

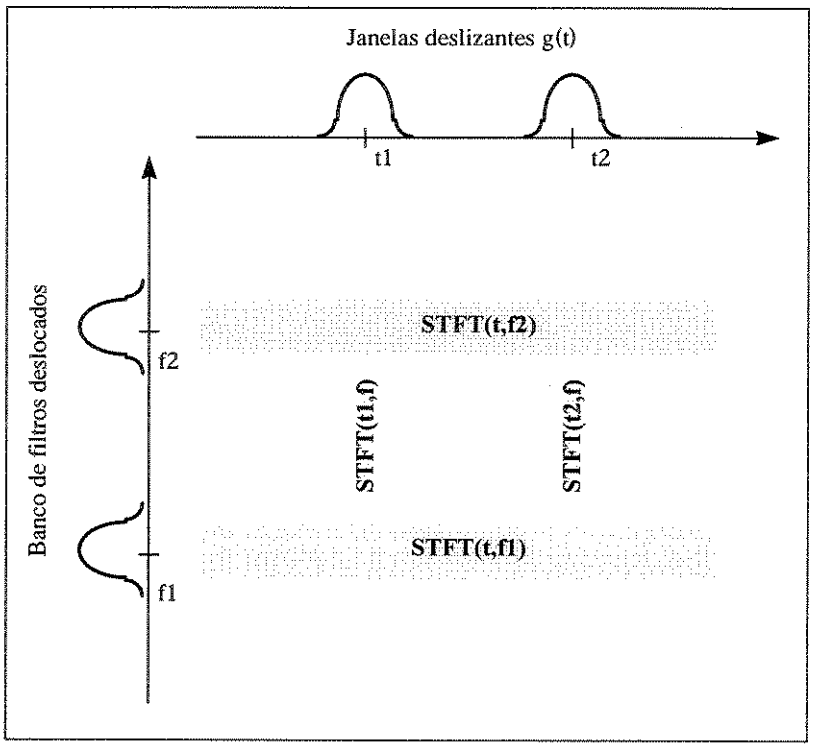



Figura 2.3: Plano tempo-freqüência correspondente a STFT.

Podemos interpretar a STFT como um banco de filtros infinito. Fazendo algumas manipulações na expressão (2.2), chegamos ao seguinte resultado:

$$\begin{aligned} STFT\left(t_0, f_0\right) &= e^{-j2\pi f_0 t_0} \int f\left(t\right) \cdot g^*\left(t - t_0\right) \cdot e^{-j2\pi f_0 (t - t_0)} dt \\ &= e^{-j2\pi f_0 t_0} \left\{ f\left(t\right) * \left[ g^*\left(t\right) e^{-j2\pi f_0 t} \right] dt \right\} \end{aligned} \tag{2.17}$$

Observe que $g^*\left(t\right) e^{-j2\pi f_0 t}$ pode ser interpretada como a resposta impulsiva de um filtro linear, com resposta em freqüência $G^*\left(-f - f_0\right)$. Tomando a TF da expressão entre chaves em (2.17), obtemos:

$$T.F. \left\{ f\left(t\right) * \left[ g^*\left(t\right) e^{-j2\pi f_0 t} \right] \right\} = F\left(f\right) \cdot G^*\left(-f - f_0\right) \tag{2.18}$$

Concluímos que a STFT de um sinal $f\left(t\right)$ pode ser interpretada como uma filtragem, utilizando o filtro de resposta em freqüência $G^*\left(-f - f_0\right)$ multiplicada pela exponencial $e^{-j2\pi f_0 t_0}$.

Todos os filtros da STFT são obtidos a partir de um único filtro $G^*\left(f\right)$ através de deslocamentos em freqüência. As listras horizontais da Figura 2.3 ilustram o banco de filtros obtido desta forma. Como os deslocamentos em freqüência não alteram a banda de freqüência, todos os filtros têm a mesma banda, o que pode ser visto na Figura 2.4.

Com a STFT adquirimos a habilidade de localizar freqüências, mas também adquirimos novos problemas. Um deles, inerente à técnica, é o fato da STFT ser uma função tridimensional.

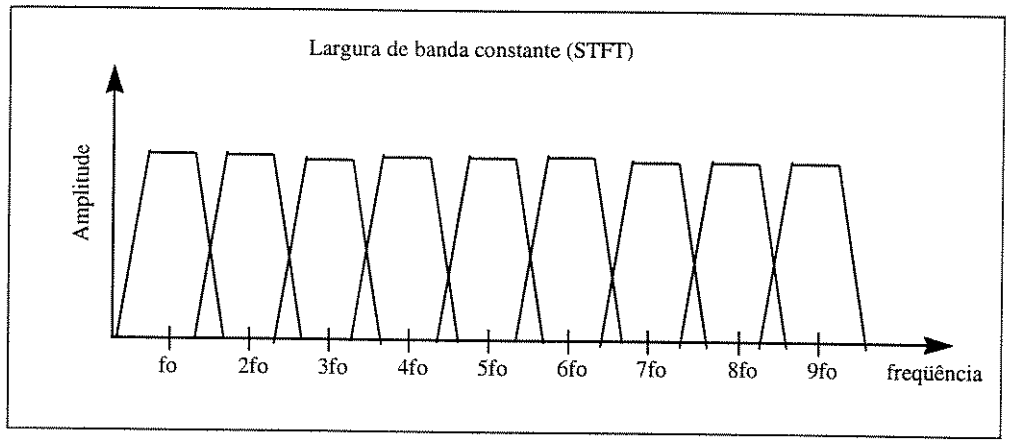



Figura 2.4: Bandas de freqüência das janelas para STFT.

Outro problema decorre da impossibilidade de obtermos uma alta precisão simultaneamente no domínio do tempo e da freqüência. Definimos a faixa de freqüências, ou espalhamento em freqüência, da janela g(t) como [Young92]:

$$\Delta f^2 = \frac{\int f^2 |G(f)|^2 df}{\int |G(f)|^2 df} \tag{2.19}$$

onde $G(f)$ é a transformada de Fourier de $g(t)$. Da mesma forma, o espalhamento no tempo é dado por:

$$\Delta t^2 = \frac{\int t^2 |g(t)|^2 dt}{\int |g(t)|^2 dt} \tag{2.20}$$

As resoluções no tempo e na freqüência não podem ser arbitrariamente pequenas, pois o seu produto tem um limitante inferior, dado pelo princípio da incerteza ou desigualdade de Heisenberg [Gabor]:

$$\Delta f \times \Delta t \geq \frac{1}{4\pi} \tag{2.21}$$

As janelas gaussianas são frequentemente empregadas pois elas satisfazem (2.21) com a igualdade [Gabor].

Considerando uma faixa de freqüências $\Delta f$, dois impulsos $\delta(t)$ e $\delta(t-\tau)$ só poderão ser discriminados pela STFT se distantes de um intervalo maior que $\Delta t$. Da mesma forma, considerando um intervalo de tempo $\Delta t$, duas senóides $sen(wt)$ e $sen(w-wo)t$ só poderão ser discriminadas se separadas por um intervalo em freqüência maior que $\Delta f$. Uma vez escolhida a janela $g(t)$, as resoluções no tempo e na freqüência são fixas para todo o plano tempo-freqüência. Na tentativa de resolver o problema de limitação da resolução da STFT, surge a TW que tem a largura do espectro da janela variável. Com isso, a resolução $\Delta t$ no tempo também varia. Porém, $\Delta t$ e $\Delta f$ ainda satisfazem a desigualdade de Heisenberg (2.21).

As Figuras 2.5 e 2.6 nos dão uma idéia do comportamento da resolução no plano tempo-freqüência para a STFT e a TW. A STFT possui janelas de comprimento fixo e, como con-

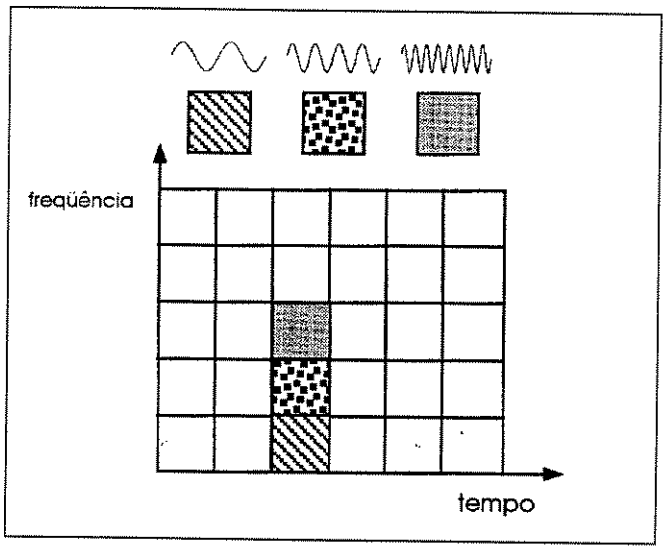



Figura 2.5: Resolução no plano tempo-freqüência da STFT.

seqüência, a sua resolução permanece constante em todo plano tempo-freqüência, conforme podemos ver na Figura 2.5. Isto significa que não há adaptabilidade na análise.

A TW possui janelas de comprimentos variáveis. Podemos escolher uma janela mais adequada para a análise de cada parte do sinal. Na Figura 2.6 escolhemos janelas estreitas para a análise de altas freqüências e janelas largas para a análise de baixas freqüências. Como conseqüência, obtemos uma maior resolução no tempo para as altas freqüências e uma menor para as baixas. Já a resolução em freqüência é maior para as baixas freqüências e menor para as altas.

Da mesma forma que a STFT, a TWC também pode ser interpretada como uma filtragem. A equação (2.11) pode ser expressa da seguinte forma:

$$TWC\left(a,b\right) = f\left(b\right) * \frac{1}{\sqrt{a}}\psi^{*}\left(\frac{b}{a}\right) \tag{2.22}$$

ou seja, a TWC de um sinal $f\left(x\right)$ corresponde à filtragem deste através de um filtro linear com resposta impulsiva $\frac{1}{\sqrt{a}}\psi^{*}\left(\frac{t}{a}\right)$. A resposta em freqüência deste filtro é dada por:

$$\Psi_{a}\left(f\right) = \sqrt{a}\Psi\left(af\right) \tag{2.23}$$

O valor da integral da magnitude quadrática de tais respostas é constante em relação ao parâmetro **a**. Enquanto que o banco de filtros da STFT é caracterizado por apresentar faixa constante, o banco de filtros da TWC é caracterizado por apresentar uma largura de faixa relativa constante, conforme é apresentado na Figura 2.7. Tais filtros são denominados de filtros com fator $Q$ constante [Young92].

Reduzindo o valor de **a** em (2.23), deslocamos a faixa do filtro para freqüências superiores. Como conseqüência, $\Delta f$ aumenta e $\Delta t$ diminui. Para a análise de altas freqüências podemos escolher valores pequenos de **a** e com isso obter uma boa localização no tempo.

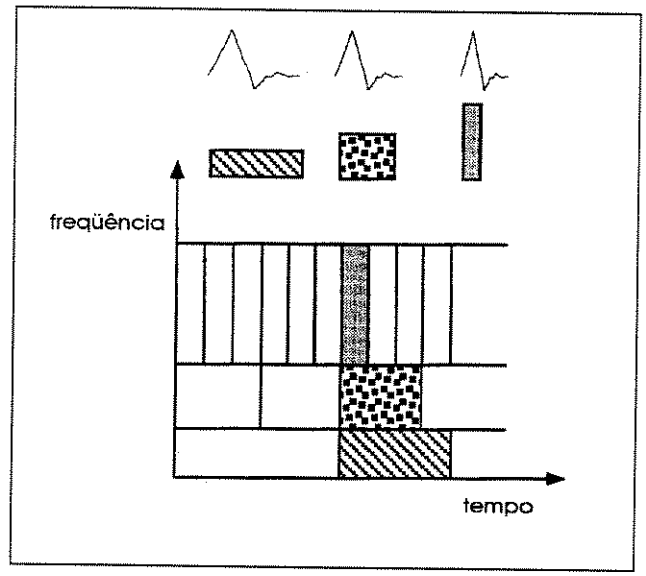



Figura 2.6: Resolução no plano tempo-freqüência da CWT.

A escala utilizada na transformada wavelet tem o mesmo significado da escala utilizada nos mapas geográficos, onde as escalas maiores fornecem visões globais enquanto que as menores nos permitem observar detalhes. Na transformada wavelet, escalas grandes nos permitem analisar trechos longos do sinal, enquanto que escalas menores, trechos mais curtos de forma mais detalhada.

## 2.4 A Transformada Wavelet Discreta ( TWD)

Analogamente à teoria de Fourier, na teoria wavelet a transformada wavelet contínua não é empregada tão frequentemente quanto a transformada wavelet discreta. As formas discretas são necessárias para a maioria das implementações. Para obtermos a transformada wavelet discreta ($TWD$), os parâmetros de deslocamento e escala são discretizados. Já a variável independente, isto é, o tempo, permanece contínua.

A transformada discreta nos fornece um conjunto enumerável de coeficientes, os quais correspondem a pontos em uma grade bidimensional no domínio deslocamento-escala. Essa grade será indexada por dois números inteiros. O primeiro inteiro, **m**, está relacionado à escala, enquanto que o segundo inteiro, **n**, está relacionado ao deslocamento. O parâmetro escala **a** é discretizado de forma exponencial, $\mathbf{a} = a_0^m$, enquanto que o parâmetro **b** é discretizado proporcionalmente a **a**, $\mathbf{b} = n \cdot b_0 \cdot a_0^m$. As constantes $a_0$ e $b_0$ são os comprimentos dos passos discretos de escala e deslocamento, respectivamente. A transformada wavelet discreta de um sinal $f(x)$ é definida como [Chui92]:

$$TWD(m,n) = \frac{1}{\sqrt{a_0^m}} \int_{-\infty}^{\infty} f(x) \cdot \psi^* \left( a_0^{-m} \cdot x - n \cdot b_0 \right) dx \tag{2.24}$$

Ao contrário da TWC, a TWD só está definida para valores positivos da escala, ou seja, para $a_0 > 0$. Isto não chega a ser uma condição restritiva, já que podemos utilizar uma wavelet

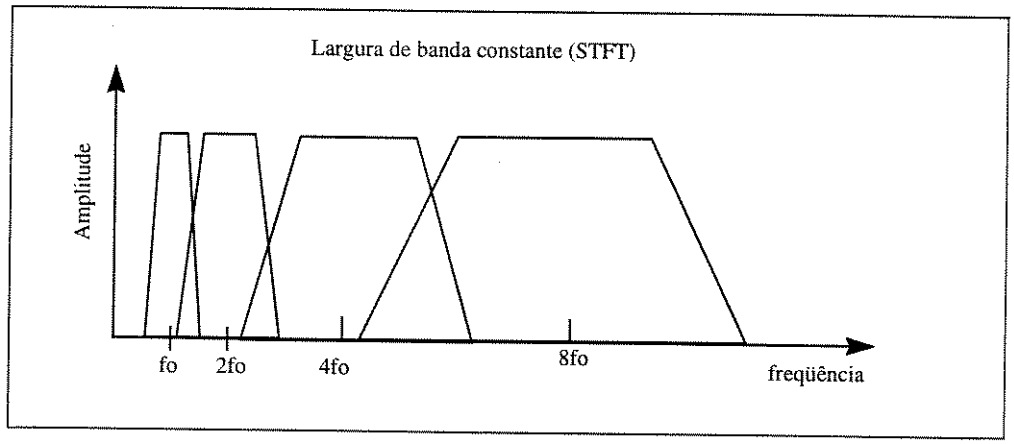



Figura 2.7: Bandas de freqüência das janelas para a CWT.

refletida para cobrir as escalas negativas.

Para valores grandes de **a**, a resolução no tempo é pequena e, conseqüentemente, os passos de deslocamento são grandes. Para valores pequenos de **a**, a resolução no tempo é grande e os passos de deslocamento são pequenos. Isto justifica o fato do passo de deslocamento ser proporcional à escala. Definindo:

$$\mathbf{a} = a_o^j \qquad \text{e} \qquad \mathbf{b} = n \cdot b_o \cdot a_0^j = k.a_o^j, \qquad j, k \in \mathbb{Z}. \tag{2.25}$$

podemos reescrever a expressão (2.24) da seguinte forma:

$$TWD\,(j,k) = a_0^{-j/2} \int_{-\infty}^{\infty} f\,(x) \cdot \psi^* \left(a_0^{-j} \cdot x - k\right) dx \tag{2.26}$$

Podemos comparar a análise de um sinal através das wavelets com a análise de um material em um microscópio. Para analisar um material, a primeira coisa que devemos fazer é nos mover para o local escolhido, o que seria equivalente na análise wavelets a escolher o valor para $k$ (2.25). Em seguida, devemos escolher o fator de ampliação, ou seja, $a_o^{-j}$. Se desejarmos ver detalhes muito pequenos do material a ampliação deve ser grande, o que corresponde a $j < 0$. Como conseqüência, o valor do produto $k \cdot a_0^j$ em (2.25) será pequeno produzindo deslocamentos pequenos. Se quisermos visualizar uma área maior, devemos escolher uma ampliação menor e, portanto, $k \cdot a_0^j$ será maior, levando a deslocamentos maiores.

Na Figura 2.8 vemos um exemplo de uma possível grade bidimensional de amostragem do plano deslocamento-escala da TW. Neste, cada ponto corresponde a uma wavelet $\psi_{a,b}\,(x)$ e, portanto, a um coeficiente wavelet. Como podemos ver, à medida que **a** aumenta, o número de coeficientes para um dado intervalo de tempo diminui exponencialmente. Conseqüentemente, o número de passos necessário para cobrir o intervalo diminui, já que o tamanho da janela aumentou.

Freqüentemente, o tamanho do passo da escala discreta é escolhido como sendo igual a 2, ou seja, temos $a_0 = 2$. Esse valor facilita a implementação, já que dilatar um sinal por um fator de 2 corresponde simplesmente a tomar uma amostra sim e outra não do sinal [Oppenheim].

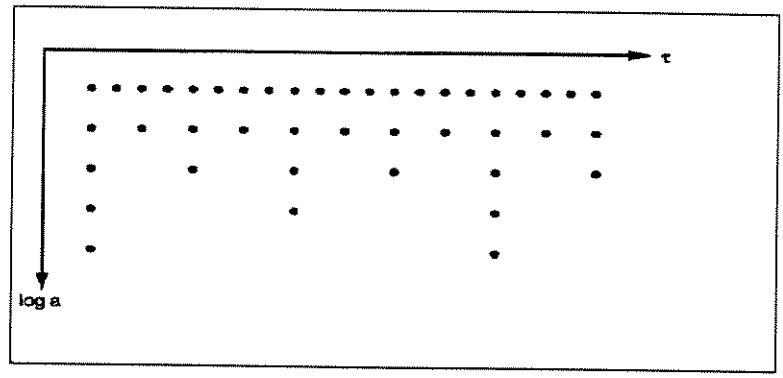



Figura 2.8: Grade de amostragem diádica do plano deslocamento-escala da TW. Cada nó corresponde a uma função wavelet $\psi_{jk}(x)$ com escala $2^{-j}$ e deslocamento $2^{-j}k$.

Na Figura 2.8 $a_0 = 2$ e, portanto, cada vez que aumentamos **a** por um fator de 2 o número de coeficientes diminui também por um fator de 2.

Quando $a_0 = 2$, dizemos que as freqüências são particionadas em oitavas consecutivas. Assim, as wavelets-filhotes são obtidas através de dilatações ou compressões binárias (de $2^j$) e deslocamentos diádicos ( de $k/\ 2^j$) de uma única wavelet-mãe, $\psi(x)$. Esta família de wavelets é conhecida como wavelets diádicas e definida pela seguinte expressão:

$$\psi_{j,k}(x) = 2^{j/2}\ \psi(2^j x - k), \qquad j,k \in \mathbb{Z} \tag{2.27}$$

Escolhendo uma wavelet-mãe de norma unitária, isto é , $\|\psi\| = 1$, é fácil verificar que as wavelets-filhotes, $\psi_{jk}(x)$, também possuem normas unitárias, isto é:

$$\|\psi_{j,k}\| = \|\psi\| = 1, \qquad j,k \in \mathbb{Z}.$$

onde $\|f\| = \left[\int |f(x)|^2\, dx\right]^{1/2}$.

Como já enfatizado, é possível reconstruir uma função de energia finita a partir dos seus coeficientes da TWC. No caso da TWD, entretanto, é preciso tomar cuidado ao escolher a wavelet-mãe e a densidade da grade, de modo que a reconstrução não se torne instável [Young80].

A reconstrução do sinal $f(x)$, a partir dos seus coeficientes wavelet, é dada pela seguinte expressão:

$$f(x) = c \sum_j \sum_k C_{j,k} \psi_{j,k}(x) \tag{2.28}$$

onde os $C_{j,k}$ são os coeficientes da TWD e c é uma constante.

Se a grade é esparsa (por exemplo, $a_0 = 2$), a representação wavelet se torna menos redundante. Como conseqüência, a equação de reconstrução pode não ser mais válida, ou seja, a wavelet e os coeficientes correspondentes podem não ser mais adequados para a reconstrução do sinal. Assim, para garantir a reconstrução neste caso é preciso fazer uma série de restrições à wavelet mãe $\psi(x)$ .

Por outro lado, se a grade é densa ($a_0 = 1$), as wavelets são redundantes e a reconstrução se dá com maior fidelidade. Além disso, não há muitas restrições acerca da wavelet $\psi(x)$ a ser utilizada. A representação redundante, entretanto, não é eficiente.

O ideal é que as grades sejam densas apenas o suficiente para que a reconstrução seja realizada com um mínimo de perda e um máximo de eficiência. É usual buscar uma base ortonormal para representar um sinal de forma eficiente.

**Definição 2-2** Uma wavelet é ortonormal, se a sua família $\{\psi_{j,k}\}$, definida em (2.27), formar uma base ortonormal para $L^2(\mathbb{R})$, isto é, se [Daubechies92]:

$$\langle \psi_{j,k}, \psi_{l,m} \rangle = \delta(j,l) \cdot \delta(k,m), \qquad j,k,l,m \in \mathbb{Z}.$$

e se toda função $f(x) \in L^2(\mathbb{R})$ puder ser representada por:

$$f(x) = \sum_{j,k=-\infty}^{\infty} C_{j,k}\psi_{j,k}(x) \tag{2.29}$$

onde, tendo em vista que as funções $\psi_{j,k}$ são ortonormais, os coeficientes $C_{j,k}$, podem ser facilmente calculados através de:

$$C_{j,k} = < f, \psi_{j,k} > . \tag{2.30}$$

A série (2.29) converge em $L^2(\mathbb{R})$, isto é:

$$\lim_{M_1, N_1, M_2, N_2 \to \infty} \left\| f - \sum_{j=-M_2}^{N_2} \sum_{j=-M_1}^{N_1} C_{j,k}\psi_{j,k} \right\| = 0$$

## 2.5 Frames

Um frame é um conjunto de elementos funcionais, tais que qualquer função não-nula deve ter uma projeção não nula em pelo menos um desses elementos. Além disso, qualquer função finita pode ser representada através de uma soma finita das suas projeções.

**Definição 2-3** Seja $\{g_k, k \in K\}$ um conjunto de funções em $L^2(\mathbb{R})$. Este conjunto é um frame se existirem números $0 \leq A \leq B < \infty$ , tais que:

$$A\|f\|^2 \leq \sum_{k \in K} |\langle g_k, f \rangle|^2 \leq B\|f\|^2, \forall f \in L^2(\mathbb{R}) \tag{2.31}$$

onde os números A e B são conhecidos como os limitantes do frame.

Os elementos de um frame podem ser linearmente dependentes, ao contrário do que acontece com os elementos de uma base. Um frame é necessariamente um conjunto completo em $L^2(\mathbb{R})$. Entretanto, pelo fato do frame ser geralmente redundante, ele não necessariamente constitui uma base para $L^2(\mathbb{R})$. A dependência entre os elementos do frame cria uma redundância, o que sugere que a representação pode não ser única.

Frames não-redundantes são conhecidos como **exatos**. Um frame é **exato** se, e apenas se, ele é uma base para $L^2(\mathbb{R})$ (não necessariamente uma base ortonormal). Um frame é dito ser **estreito** ("tight") se os seus limitantes, A e B, são iguais.

**Exemplo 2-1** Considere $\mathbb{R}^2$ e sua base ortonormal elementar $\overrightarrow{e_1} = [0,1]$ e $\overrightarrow{e_2} = [1,0]$. Definindo três elementos normalizados em $\mathbb{R}^2$:

$$\overrightarrow{\beta_1} = \overrightarrow{e_1}; \qquad \overrightarrow{\beta_2} = -\frac{1}{2}\overrightarrow{e_1} + \frac{\sqrt{3}}{2}\overrightarrow{e_2}; \qquad \overrightarrow{\beta_3} = -\frac{1}{2}\overrightarrow{e_1} - \frac{\sqrt{3}}{2}\overrightarrow{e_2}$$

temos:

$$\begin{aligned}
\sum_{j=1}^{3} |\langle f, \beta_j\rangle|^2 &= |\langle f, \overrightarrow{e_1}\rangle|^2 + \left|\left(-\frac{1}{2}\right)\langle f, \overrightarrow{e_1}\rangle + \left(\frac{\sqrt{3}}{2}\right)\langle f, \overrightarrow{e_2}\rangle^2\right| \\
&\quad + \left|\left(-\frac{1}{2}\right)\langle f, \overrightarrow{e_1}\rangle + \left(-\frac{\sqrt{3}}{2}\right)\langle f, \overrightarrow{e_2}\rangle\right|^2 \\
&= \frac{3}{2}|\langle f, \overrightarrow{e_1}\rangle|^2 + \frac{3}{2}|\langle f, \overrightarrow{e_2}\rangle|^2 = \frac{3}{2}\left\{|\langle f, \overrightarrow{e_1}\rangle|^2 + \frac{3}{2}|\langle f, \overrightarrow{e_2}\rangle|^2\right\} \\
&= \frac{3}{2}|\langle f, \overrightarrow{e}\rangle|^2
\end{aligned}$$

Conclui-se, então, que o conjunto de vetores $\left\{\overrightarrow{\beta_1}, \overrightarrow{\beta_2}, \overrightarrow{\beta_3}\right\}$ é um frame estreito, mas definitivamente não é uma base ortonormal. A = B = $\frac{3}{2}$ nos dá a razão de redundância. Se a razão de redundância é igual a 1, então o frame estreito é uma base ortonormal, como enunciado a seguir.

**Lema 2-1** Se o frame é estreito, A = B = 1 e todos os componentes do frame, $\{g_k, k \in K\}$, têm norma unitária, $\|g_k\| = 1$, então o frame é uma base ortonormal [Daubechies92].

A qualquer frame $\{g_k, \quad k \in K\}$, podemos associar um "operador frame" $T : L^2(\mathbb{R}) \rightarrow L^2(\mathbb{R})$, definido por :

$$Tf = \sum_{k \in K} \langle f, g_k\rangle\, g_k \tag{2.32}$$

ou seja, a transformação $Tf$ consiste na expansão em série de $f$, onde o frame $\{g_k\}$ é utilizado como base e os coeficientes são dados por $\langle f, g_k\rangle$.

Demonstra-se [Porat] que o operador T é inversível. Representaremos a operação inversa por $T^{-1}$.

**Lema 2-2** Seja $\{\gamma_k, \quad k \in K\}$ um conjunto de funções obtidas a partir do frame $\{g_k\}$ através da seguinte operação:

$$\gamma^k = T^{-1} g_k \tag{2.33}$$

O conjunto $\{\gamma_k\}$ é um frame, que definimos como frame dual de $\{g_k\}$ [Porat].

Quando o frame $\{g_k\}$ é exato, os conjuntos $\{g_k\}$ e $\{\gamma_k\}$ são biortogonais, ou seja:

$$\langle g_k, \gamma_l\rangle = \delta(k - l).$$

**Teorema 2-1** Para qualquer frame $\{g_k\}$ e o seu dual $\{\gamma_k\}$, podemos expressar qualquer função $f(x) \in L^2(\mathbb{R})$ através de qualquer uma das seguintes formas:

$$f = \sum_{k \in K} \langle f, \gamma_k \rangle \, g_k \tag{2.34}$$

e

$$f = \sum_{k \in K} \langle f, g_k \rangle \, \gamma_k \tag{2.35}$$

Estas equações mostram que podemos expressar qualquer função de $L^2(\mathbb{R})$ como uma combinação linear dos elementos de um frame. O conjunto dos coeficientes de (2.34) e (2.35) é quadrado somável, o que é conseqüência de $\{g_k\}$ e $\{\gamma_k\}$ formarem um par de frames duais. Assim, dado um frame $\{g_k\}$, para obtermos $f(x)$ precisamos apenas calcular o seu dual $\{\gamma_k\}$, o que é feito através da expressão (2.33) [Chui92].

No caso de um frame estreito, o operador frame T é simplesmente igual a $A \cdot \mathbf{1}$, onde $\mathbf{1}$ é o operador identidade [Porat]. Nesse caso, $\gamma_k = A^{-1} g_k$ e o frame dual é obtido de forma trivial. No caso do frame não ser estreito, o frame dual pode ser obtido através de algoritmos recursivos.

A expressão (2.34) é bastante semelhante a uma expansão em base ortogonal, mas suas propriedades são diferentes. Em particular, os seus coeficientes em (2.34) são obtidos a partir de produtos internos da função com os elementos do frame dual, ao invés de produtos internos da função com os elementos do frame, como acontece na expansão ortogonal.

Se uma wavelet $\psi(x)$ gera um frame, ou seja, se $\psi(x)$ satisfaz a seguinte relação:

$$A \|f\|^2 \leq \sum_{j,k \in \mathbb{Z}} |\langle f, \psi_{j,k} \rangle|^2 \leq B \|f\|^2 \tag{2.36}$$

então, uma função $f(x) \in L^2(\mathbb{R})$ pode ser recuperada a partir dos coeficientes da sua transformada wavelet [Daubechies92].

Seja $\{\psi_{j,k}\}$ um frame. Substituindo $\{g_k\} = \{\psi_{j,k}\}$ em (2.32), obtemos a seguinte expressão:

$$Tf = \sum_{j,k \in \mathbb{Z}} \langle f, \psi_{j,k} \rangle \cdot \psi_{j,k}, \qquad f \in L^2(\mathbb{R}) \tag{2.37}$$

e, de acordo com o Teorema 2-1, a função $f(x)$ pode ser expressa da seguinte forma:

$$f = T^{-1} T \cdot f = \sum_{j,k \in \mathbb{Z}} \langle f, \psi_{j,k} \rangle \cdot T^{-1} \psi_{j,k} \tag{2.38}$$

Definindo, então, $\tilde{\psi}_{j,k}$ como o frame dual de $\psi_{j,k}$ :

$$\tilde{\psi}_{j,k} \triangleq T^{-1} \cdot \psi_{j,k} \qquad j, k \in \mathbb{Z} \tag{2.39}$$

a fórmula de reconstrução da função $f(x)$ pode ser reescrita da seguinte forma :

$$f = \sum_{j,k \in \mathbb{Z}} \langle f, \psi_{j,k} \rangle \, \tilde{\psi}_{j,k} \tag{2.40}$$

ou

$$f = \sum_{j,k\in\mathbb{Z}} \left\langle f, \tilde{\psi}_{j,k} \right\rangle \psi_{j,k} \tag{2.41}$$

No caso de frames estreitos, o frame dual é composto através de versões escaladas e deslocadas da wavelet-mãe, multiplicadas por uma constante, isto é:

$$\tilde{\psi}_{j,k} = A^{-1}\psi_{j,k}$$

e, para este caso, podemos reescrever (2.40) da seguinte forma :

$$f = A^{-1} \sum_{j,k\in\mathbb{Z}} \langle f, \psi_{j,k} \rangle \psi_{j,k} \tag{2.42}$$

Os frames que obedecem às condições impostas pelo Lema 2-1 formam bases ortonormais. Para estes (2.42) pode ser simplificada e expressa da seguinte forma:

$$f = \sum_{j,k\in\mathbb{Z}} \langle f, \psi_{j,k} \rangle \psi_{j,k} \tag{2.43}$$

Uma possível vantagem dos frames redundantes sobre os ortonormais é que os primeiros permitem que uma classe maior de funções possam ser utilizadas como wavelet-mãe. Os vetores de um frame redundante podem, ainda, "casar" melhor com um sinal e, portanto, um número menor de coeficientes será necessário para representar o conteúdo de informação deste sinal.

Os frames ortonormais, por outro lado, impõem restrições à wavelet-mãe, o que pode não ser desejável em algumas aplicações. Entretanto, para representação, processamento e codificação de imagens, as transformadas wavelets ortonormais são mais adequadas, por serem mais facilmente implementáveis [Daubechies92].

## 2.6 Análise de Multiresolução

A análise de multiresolução nos fornece uma base para interpretação da informação contida em um sinal. Esta informação é organizada em um conjunto de detalhes com diferentes resoluções. Para um sinal $f(x) \in L^2(\mathbb{R})$, definido no intervalo $0 < x < x_0$, a sua resolução nos fornece a quantidade de pontos para os quais este sinal é conhecido. Dizemos que conhecemos um sinal com resolução completa, ou com um nível de resolução infinita, quando conhecemos $f(x)$ para todos os pontos do intervalo. Da mesma forma, dizemos que conhecemos um sinal com resolução $r_j$ quando conhecemos $r_j$ pontos do intervalo.

Dada uma seqüência crescente de resoluções $(r_j)_{j\in\mathbb{Z}}$, os detalhes com resolução $r_j$ de um sinal são definidos como a diferença de informação entre a sua aproximação com resolução $r_j$ e a sua aproximação com resolução $r_{j-1}$, mais baixa.

O nosso objetivo nesta seção é estudar as propriedades do operador que obtém uma aproximação com resolução $2^j$ de uma função e mostrar que a diferença de informação entre duas aproximações, com resoluções $2^{j+1}$ e $2^j$, é extraída decompondo-se a função em uma base wavelet ortogonal. Esta decomposição define uma representação por multiresolução ortogonal denominada **representação wavelet**. A partir desta representação construiremos um algoritmo para implementação da transformada wavelet ortogonal.

### 2.6.1 A Aproximação por Multiresolução em $L^2(\mathbb{R})$

Seja $f(x) \in L^2(\mathbb{R})$ o nosso sinal original e $A_{2^j}$ o operador que gera uma aproximação deste sinal, $A_{2^j} f(x)$, com resolução $2^j$. Este operador apresenta as seguintes propriedades:

**Propriedade 1** $A_{2^j}$ é um operador linear. Se $A_{2^j} f(x)$ é a aproximação de uma função $f(x)$ com resolução $2^j$, então $A_{2^j} f(x)$ não é modificada se repetirmos a operação no nível de resolução $2^j$. Isto significa que $A_{2^j} \circ A_{2^j} = A_{2^j}$. O operador $A_{2^j}$ é, então, um operador de projeção no espaço vetorial $\mathbf{V}_{2^j} \subset L^2(\mathbb{R})$. O espaço vetorial $\mathbf{V}_{2^j}$ pode ser interpretado como o conjunto de todas as possíveis aproximações das funções $L^2(\mathbb{R})$ no nível de resolução $2^j$.

**Propriedade 2** Dentre todas as aproximações da função $f(x)$ no nível de resolução $2^j$, $A_{2^j} f(x)$ é a função que mais se aproxima de $f(x)$, ou seja:

$$\forall\, g(x) \in \mathbf{V}_{2^j}, |g(x) - f(x)| \geq |A_{2^j} f(x) - f(x)| \tag{2.44}$$

Portanto, o operador $A_{2^j}$ calcula a projeção ortogonal de uma função no espaço vetorial $V_{2^j}$.

**Propriedade 3** A aproximação de um sinal no nível de resolução $2^{j+1}$ contém toda a informação necessária para calcular a sua aproximação no nível de resolução $2^j$. Este é o princípio da causalidade. Como $A_{2^j}$ é um operador de projeção em $\mathbf{V}_{2^j}$, esta propriedade é equivalente a:

$$\forall\, j \in \mathbb{Z},\ V_{2^j} \subset V_{2^{j+1}} \tag{2.45}$$

Aplicando (2.45) recursivamente obtemos a seguinte hierarquia de subespaços:

$$\ldots \subset \mathbf{V}_{2^{-1}} \subset \mathbf{V}_{2^0} \subset \mathbf{V}_{2^1} \subset \mathbf{V}_{2^2} \subset \ldots \tag{2.46}$$

**Propriedade 4** A operação aproximação independe do nível de resolução. Os subespaços das funções aproximadas podem ser obtidos a partir dos outros subespaços, através da dilatação ou compressão das aproximações das funções na razão dos valores das suas resoluções. Isto significa que:

$$\forall\, j \in \mathbb{Z},\ A_{2^j} f(x) \in V_{2^j} \Rightarrow A_{2^j} f(2x) \in V_{2^{j+1}} \tag{2.47}$$

**Propriedade 5** A aproximação $A_{2^j} f(x)$ pode ser caracterizada por $2^j$ amostras por unidade de comprimento. Quando $f(x)$ é transladada por um comprimento proporcional a $2^{-j}$, $A_{2^j} f(x)$ é transladada pela mesma quantidade e é caracterizada pelas mesmas amostras que foram deslocadas Como uma conseqüência, é suficiente expressar a Propriedade 5 para $j = 0$.

**Propriedade 6** Ao calcularmos uma aproximação de $f(x)$ no nível de resolução $2^j$, alguma informação sobre $f(x)$ é perdida. À medida que a resolução aumenta em direção a $+\infty$, o sinal aproximado converge para o sinal original. Da mesma forma, à medida que a resolução diminui, se aproximando do zero, o sinal aproximado passa a conter cada vez menos informação e, portanto, converge para zero. Como a aproximação $A_{2^j} f(x)$ é igual

à projeção ortogonal de $f(x)$ no espaço vetorial $\mathbf{V}_{2^j}$, esta propriedade pode ser resumida através das seguintes relações:

$$\lim_{j\to+\infty} \mathbf{V}_{2^j} = \bigcup_{j=-\infty}^{+\infty} \mathbf{V}_{2^j} \tag{2.48}$$

Ou seja, $\lim_{j\to+\infty} \mathbf{V}_{2^j} = L^2(\mathbb{R})$.

$$\lim_{j\to-\infty} \mathbf{V}_{2^j} = \bigcap_{j=-\infty}^{+\infty} \mathbf{V}_{2^j} = \{0\} \tag{2.49}$$

**Definição 2-4** Denominamos de aproximação por multiresolução de $L^2(\mathbb{R})$, qualquer conjunto de espaços vetoriais $\{\mathbf{V}_{2^j}\}_{j\in\mathbb{Z}}$ que satisfazem as Propriedades 1-6. O conjunto de operadores $A_{2^j}$ associados a este conjunto de espaços vetoriais gera as aproximações de qualquer função $f(x) \in L^2(\mathbb{R})$ em todos os níveis de resolução $2^j$, para $j \in \mathbb{Z}$.

**Teorema 2-2** Seja $(\mathbf{V}_{2^j})_{j\in\mathbb{Z}}$ uma aproximação por multiresolução em $L^2(\mathbb{R})$. Existe uma única função $\phi(x) \in L^2(\mathbb{R})$, denominada função escala, tal que se definirmos:

$$\phi_{j,k}(x) = 2^{j/2}\phi\left(2^j x - k\right) \tag{2.50}$$

para $j \in \mathbb{Z}$, o conjunto $\left(2^{j/2}\phi(2^j x - n)\right)_{n\in\mathbb{Z}}$ é uma base ortonormal para $\mathbf{V}_{2^j}$ [Mallat89c].

Este teorema mostra que, dilatando-se ou comprimindo-se uma função $\phi(x)$ por um fator $2^j$ e deslocando-se a função resultante de intervalos proporcionais a $2^{-j}$, podemos construir uma base ortonormal em qualquer espaço vetorial $\mathbf{V}_{2^j}$. As funções $\phi_{j,k}(x)$ são normalizadas com relação a $L^1(\mathbb{R})$. O termo $2^{-j/2}$ em (2.50) tem o objetivo de normalizar as funções em $L^2(\mathbb{R})$. Para uma dada aproximação por multiresolução $(\mathbf{V}_{2^j})_{j\in\mathbb{Z}}$ existe uma única função escala $\phi(x)$ que satisfaz o Teorema 2-2. Entretanto, para diferentes aproximações existem diferentes funções escala.

O Teorema 2-2 mostra que a aproximação por multiresolução $(V_{2^j})_{j\in\mathbb{Z}}$ é completamente caracterizada pela função escala $\phi(x)$. Esta pode ser definida como uma função $\phi(x) \in L^2(\mathbb{R})$, tal que, para qualquer $j \in \mathbb{Z}$, $\left(2^{j/2}\phi(2^j x - n)\right)_{n\in\mathbb{Z}}$ é uma família ortonormal. Além disso, se $V_{2^j}$ é o espaço vetorial gerado por esta família, $(V_{2^j})_{j\in\mathbb{Z}}$ é uma aproximação por multiresolução em $L^2(\mathbb{R})$. A função $\phi(x)$ deve ser continuamente diferenciável. O decaimento assintótico de $\phi(x)$ e da sua derivada $\phi'(x)$ no infinito deve satisfazer as seguintes relações:

$$\phi(x) = O\left(x^{-2}\right) \;\; e \;\; |\phi'(x)| = O\left(x^{-2}\right)$$

A projeção ortogonal de uma função $f(x)$ no espaço $\mathbf{V}_{2^j}$ pode ser calculada decompondo esta função na base ortogonal definida pelo Teorema 2-2, ou seja:

$$\forall f(x) \in L^2(\mathbb{R}),\; A_{2^j} f(x) = \sum_{n=-\infty}^{\infty} \langle f(u), \phi_{j,n}(u)\rangle \phi_{j,n}(x) \tag{2.51}$$

A aproximação do sinal $f(x)$ no nível de resolução $2^j$, $A_{2^j} f(x)$, é, então, caracterizada pelo seguinte conjunto de produtos internos:

$$A_{2^j}^d f = \left(\langle f(u), \phi_{j,n}(u)\rangle\right)_{n\in\mathbb{Z}} \tag{2.52}$$

que denominamos aproximação discreta do sinal $f(x)$ no nível de resolução $2^j$.

### 2.6.2 A Implementação de uma Transformada de Multiresolução

Na prática, os sinais utilizados possuem uma resolução máxima finita. Por uma questão de normalização, suporemos que esta resolução máxima é igual a 1 e, portanto, $A_1^d f$ é a sua aproximação discreta nesta resolução. O princípio da causalidade nos diz que a partir de $A_1^d f$ podemos calcular todas as aproximações discretas $A_{2^j}^d f$ para $j < 0$. Nesta seção, descreveremos algoritmos iterativos simples para calcular estas aproximações discretas.

Seja $(\mathbf{V}_{2^j})_{j\in\mathbb{Z}}$ uma aproximação por multiresolução e $\phi(x)$ a função escala correspondente. De acordo com o Teorema 2-2 , a família de funções $\left(2^{j/2}\phi\left(2^j x - n\right)\right)_{n\in\mathbb{Z}}$ é uma base ortonormal para $\mathbf{V}_{2^{j+1}}$. Sabemos que para qualquer $n \in \mathbb{Z}$ , a função $\phi_{j,n}(x)$ pertence ao espaço $\mathbf{V}_{2^j}$ que, por sua vez, está contido em $\mathbf{V}_{2^{j+1}}$. Podemos, então, expandir a função $\phi_{j,n}(x)$ na base ortonormal de $\mathbf{V}_{2^{j+1}}$ :

$$\phi_{j,n}(x) = \sum_{k=-\infty}^{\infty} \langle \phi_{j,n}(u), \phi_{j+1,k}(u)\rangle \, \phi_{j+1,k}(x) \tag{2.53}$$

Fazendo uma pequena mudança de variáveis no produto interno em (2.53), obtemos:

$$\begin{aligned}\left\langle 2^{j/2}\phi\left(2^j u - n\right), 2^{(j+1)/2}\phi\left(2^{j+1}u - k\right)\right\rangle &= 2^{j/2+(j+1)/2}\int_{-\infty}^{\infty}\phi\left(2^{-1}y\right)\phi\left(y-(k-2n)\right)2^{-j-1}dy \\ &= 2^{-1/2}\left\langle\phi\left(2^{-1}u\right), \phi\left(u-(k-2n)\right)\right\rangle \end{aligned} \tag{2.54}$$

Substituindo (2.54) em (2.53), obtemos a seguinte expressão para $\phi_{j,n}(x)$:

$$\phi_{j,n}(x) = \sum_{k=-\infty}^{\infty} \langle \phi_{-1,0}(u), \phi_{0,k-2n}(u)\rangle \cdot \phi_{j+1,k}(x) \tag{2.55}$$

Calculando o produto interno de $f(x)$ com ambos os lados de (2.55), obtemos:

$$\langle f(x), \phi_{j,n}(x)\rangle = \sum_{k=-\infty}^{\infty} \langle \phi_{-1,0}(u), \phi_{0,k-2n}(u)\rangle \cdot \langle f(u), \phi_{j+1,k}(x)\rangle \tag{2.56}$$

Sob o ponto de vista de processamento de sinais, podemos interpretar (2.56) como uma filtragem e definir H como o filtro discreto de resposta impulsiva:

$$h(n) = \langle \phi_{-1,0}(u), \phi_{0,n}(u)\rangle \qquad \forall n \in \mathbb{Z} \tag{2.57}$$

Substituindo (2.57) na expressão (2.56), obtemos:

$$\langle f(x), \phi_{j,n}(x)\rangle = \sum_{k=-\infty}^{\infty} h(k-2n)\cdot\langle f(x), \phi_{j+1,k}(x)\rangle \tag{2.58}$$

A expressão (2.58) mostra que $A_{2^j}^d f$ pode ser calculada convoluindo $A_{2^{j+1}}^d f$ com $h(-n)$ e tomando uma amostra sim e outra não do sinal resultante (subamostragem por um fator 2). No ramo superior da Figura 2.9, podemos visualizar esta operação. Observe que o símbolo $h(-n)$ indica a convolução do sinal com o filtro $h(-n)$ e o $\downarrow 2$ indica a subamostragem do resultado. Todas as aproximações $A_{2^j}^d f$, para $j<0$, podem ser calculadas a partir de $A_1^d f$ através da aplicação recursiva desta operação, que é denominada transformada piramidal [Adelson].

Vamos supor que a aproximação discreta do sinal $f(x)$ no maior nível de resolução é completamente caracterizada por N coeficientes, ou seja, $A_1^d f = (\alpha_n)_{1\le n\le N}$. Cada aproximação discreta $A_{2^j}^d f$ $(j<0)$ tem, portanto, $2^j N$ amostras. De maneira a evitar problemas de borda ao calcular as aproximações discretas $A_{2^j}^d f$, vamos supor que $A_1^d f$ seja simétrica em relação a $n=0$ e $n=N-1$, ou seja:

$$\alpha_n = \begin{cases} \alpha_{-n} & \text{se } -N-1<n<0 \\ \alpha_{2N-n} & \text{se } 0<n<N-1 \end{cases} \tag{2.59}$$

Se a resposta impulsiva do filtro $H$ for par, isto é, $h(-n)=h(n)$, cada aproximação discreta $A_{2^j}^d f$ também será simétrica com relação a $n=0$ e $n=2^{-j}N$ [Mallat89b].

Como já visto, podemos expandir a função $\phi_{j,n}(x)$ como apresentado em (2.55). Utilizando a definição (2.57), podemos reescrever (2.55) da seguinte forma:

$$\phi_{j,n}(x) = \sum_{k=-\infty}^{\infty} h(k-2n)\cdot\phi_{j+1,k}(x) \tag{2.60}$$

Fazendo $j=0$ e $n=0$ em (2.60) obtemos:

$$\phi(x) = \sum_{k=-\infty}^{\infty} h(k)\cdot 2^{1/2}\phi(2x-k) \tag{2.61}$$

Esta é a equação fundamental da análise de multiresolução e levará ao algoritmo de decomposição da transformada wavelet. Tomando a transformada de Fourier de (2.61), obtemos:

$$\Phi(\omega) = \sum_{k=-\infty}^{\infty} h(k)\,\frac{\Phi(0.5\omega)}{2^{1/2}}\,e^{-j\,0.5\omega k} = \frac{\Phi(0.5\omega)}{2^{1/2}}H(0.5\omega) \tag{2.62}$$

Uma conseqüência imediata de (2.62) é obtida para $\omega=0$:

$$H(0) = \sum_{k=-\infty}^{\infty} h(k) = \sqrt{2} \tag{2.63}$$

Aplicando (2.62) recursivamente, obtemos:

$$\Phi(\omega) = \Phi(0)\prod_{m=1}^{\infty}\frac{H(w\cdot 2^{-m})}{2^{1/2}} \tag{2.64}$$

Quando (2.64) converge, podemos obter o seguinte algoritmo recursivo para construção de $\phi(x)$:

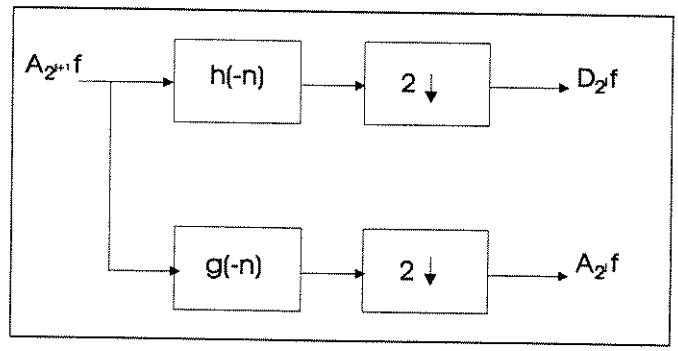



Figura 2.9: Diagrama em blocos do esquema de decomposição.

1. Seja $w(x)$ uma janela retangular em [0,1]
2. Para $j \geq 0$ , faça

$$w_j(x) = \sum_{k=-\infty}^{\infty} h(k) \cdot 2^{1/2} \cdot w_{j+1}(2x-k) \tag{2.65}$$

Então o $\lim_{j\to\infty} w_j(x) = \phi(x)$ .

A prova da convergência deste algoritmo se encontra em [Daubechies88]. Concluímos que se conhecermos os coeficientes $h(k)$ de (2.61) poderemos obter a função $\phi(x)$ através deste algoritmo.

**Lema 2-3** Se a seqüência $\{h(k)\}$ é não nula apenas para $0 \leq k \leq N-1$, ou seja, se $h(k)$ é um filtro FIR, então $\phi(x)$ é não nula apenas para $0 \leq x \leq N-1$ [Porat].

Concluímos que podemos construir uma função escala com duração limitada impondo um conjunto de restrições aos coeficientes $\{h(k)\}$ .

**Lema 2-4** A função $H(\omega)$ satisfaz [Porat]:

$$|H(\omega)|^2 + |H(\omega+\pi)|^2 = 2, \text{ para todo } \omega \tag{2.66}$$

A propriedade enunciada pelo Lema 2-4 é fundamental na construção da análise de multiresolução e das bases wavelets.

**Lema 2-5** Para qualquer análise de multiresolução, $\Phi(w) = 1$ [Porat].

Este lema mostra que as funções $\{\phi_{j,k}(x)\} = \left\{2^{j/2}\phi(2^j x - k)\right\}$ não são wavelets, uma vez que não obedecem a condição de admissibilidade (2.14).

A partir do que foi visto, podemos enunciar o seguinte teorema:

**Teorema 2-3** Seja $\phi(x)$ uma função escala e $H$ o filtro discreto com resposta impulsiva $h(n) = \langle \phi_{-1,0}(u), \phi_{0,n}(u) \rangle$. Seja $H(\omega)$ a série de Fourier de $h(n)$, definida por:

$$H(\omega) = \sum_{n=-\infty}^{\infty} h(n)\, e^{-in\omega} \tag{2.67}$$

onde $H(\omega)$ satisfaz as seguintes propriedades:

$$H(0) = \sqrt{2} \qquad \text{e} \qquad h(n) = O\left(x^{-2}\right) \text{ no infinito} \tag{2.68}$$

$$|H(\omega)|^2 + |H(\omega + \pi)|^2 = 2 \tag{2.69}$$

Estas propriedades indicam que $H(\omega)$ é um filtro passa-baixas. Se $H(\omega)$ satisfaz as igualdades (2.68) e (2.69) e, ainda apresenta a seguinte característica:

$$|H(\omega)| \neq 0 \qquad \text{para } \omega \in [0, \pi/2] \tag{2.70}$$

a função definida por:

$$\Phi(\omega) = \Phi(0) \prod_{p=1}^{\infty} \frac{H(2^{-p}\omega)}{2^{1/2}} \tag{2.71}$$

é a transformada de Fourier da função escala [Mallat89c].

A partir de um filtro conjugado $H(\omega)$, que satisfaça (2.68) e (2.70), podemos calcular a transformada de Fourier da função escala correspondente através de (2.71). É possível, então, escolher $H(\omega)$ de maneira a obter uma função escala $\phi(x)$ com boas propriedades de localização, tanto no domínio da freqüência quanto no domínio espacial [Daubechies92].

### 2.6.3 O Sinal de Detalhamento

Seja $\mathbf{W}_{2^j}$ o complemento ortogonal de $\mathbf{V}_{2^j}$, ou seja:

$$\mathbf{W}_{2^j} \text{ é ortogonal a } \mathbf{V}_{2^j},$$

$$\mathbf{V}_{2^j} \oplus \mathbf{W}_{2^j} = \mathbf{V}_{2^{j+1}}.$$

O sinal formado a partir da diferença de informação entre os níveis de resolução $2^j$ e $2^{j+1}$ é denominado sinal de detalhamento no nível de resolução $2^j$. Como já mencionado, as aproximações de um sinal nos níveis de resolução $2^{j+1}$ e $2^j$ são, respectivamente, iguais às suas projeções ortogonais em $\mathbf{V}_{2^{j+1}}$ e $\mathbf{V}_{2^j}$. Como conseqüência, o sinal de detalhamento no nível de resolução $2^j$ é obtido através da projeção ortogonal do sinal original em $\mathbf{W}_{2^j}$.

Para calcular a projeção ortogonal de uma função $f(x)$ em $\mathbf{W}_{2^j}$, é necessário encontrar uma base ortonormal para gerar o espaço $\mathbf{W}_{2^j}$. Procuramos, então, uma função $\psi(x)$ que tenha para $W_{2^j}$ o mesmo papel que $\phi(x)$ tem para $V_{2^j}$. Queremos que a família definida por:

$$\left\{\psi_{j,k}(x) = 2^{j/2}\, \psi\left(2^j x - k\right), k \in \mathbb{Z}\right\} \tag{2.72}$$

seja uma base ortonormal para $W_{2^j}$, o que significa que $\{\psi_{j,k}(x), j,k \in \mathbb{Z}\}$ é uma base ortonormal para $L^2(\mathbb{R})$. Demonstraremos que a base que procuramos pode ser formada a partir de uma família de wavelets, ou seja, que as wavelets podem ser utilizadas como funções geradoras dos subespaços $W_{2^j}$.

Vamos assumir a existência desta função $\psi(x)$ e verificar quais as conseqüências dessa suposição. Este procedimento nos dará uma algoritmo para construção de $\psi(x)$. Para qualquer $n \in \mathbb{Z}$, a função $\psi_{j,n}(x)$ pertence a $\mathbf{W}_{2^j} \subset \mathbf{V}_{2^{j+1}}$. Portanto, esta função pode ser expandida na base ortonormal de $\mathbf{V}_{2^{j+1}}$ :

$$\psi_{j,n}(x) = \sum_{k=-\infty}^{\infty} \langle \psi_{j,n}(u), \phi_{j+1,k}(u) \rangle \phi_{j+1,k}(x) \tag{2.73}$$

Fazendo uma pequena mudança de variáveis no produto interno em (2.73), obtemos:

$$\begin{aligned} \langle \psi_{j,n}(u), \phi_{j+1,k}(u) \rangle &= 2^{j/2+(j+1)/2} \int_{-\infty}^{\infty} \psi\left(2^{-1}y\right) \cdot \phi\left(y-(k-2n)\right) 2^{-j-1} dy \\ &= 2^{-1/2} \left\langle \psi\left(2^{-1}u\right), \phi\left(u-(k-2n)\right) \right\rangle \end{aligned} \tag{2.74}$$

Substituindo (2.74) em (2.73):

$$\psi_{j,n}(x) = \sum_{k=-\infty}^{\infty} \langle \psi_{-1,0}(u), \phi_{0,k-2n}(u) \rangle \phi_{j+1,k}(x) \tag{2.75}$$

Podemos interpretar (2.75) como uma filtragem e definir G como o filtro discreto de resposta impulsiva dada por:

$$g(n) = \langle \psi_{-1,0}(u), \phi_{0,n}(u) \rangle \tag{2.76}$$

Substituindo (2.76) em (2.75):

$$\psi_{j,n}(x) = \sum_{k=-\infty}^{\infty} g(k-2n) \cdot \phi_{j+1,k}(x) \tag{2.77}$$

Fazendo $j = 0$ e $n = 0$ em (2.77), obtemos a seguinte expressão:

$$\psi(x) = \sum_{k=-\infty}^{\infty} g(k) \cdot 2^{1/2} \cdot \phi(2x-k) \tag{2.78}$$

onde $\{g(k)\}$ é uma seqüência quadrado somável. Tomando a transformada de Fourier de ambos os lados de (2.78), obtemos:

$$\Psi(\omega) = \sum_{k=-\infty}^{\infty} g(k) \frac{\Phi(0.5\omega)}{2^{1/2}} e^{-j\,0.5\omega n} = \frac{\Phi(0.5\omega)}{2^{1/2}} G(0.5\omega) \tag{2.79}$$

É possível provar, exatamente como no Lema 2-4, que

$$|G(\omega)|^2 + |G(\omega+\pi)|^2 = 2, \text{ para todo } \omega \tag{2.80}$$

Além disso, $G(\omega)$ e $H(\omega)$ satisfazem a seguinte relação [Porat]:

$$H(\omega)G^*(\omega) + H(\omega+\pi)G^*(\omega+\pi) = 0 \tag{2.81}$$

Para mostrar que $\{\psi_{j,k}(x),\ k \in \mathbb{Z}\}$ é uma base wavelet, resta-nos apenas verificar se $\psi(x)$ satisfaz a condição de admissibilidade, ou seja, se $\Psi(0) = 0$. A equação (2.63) mostra que $H(0) = \sqrt{2}$. Substituindo (2.63) em (2.66), obtemos:

$$\begin{aligned} |H(0)|^2 + |H(\pi)|^2 &= 2 + |H(\pi)|^2 = 2 \\ |H(\pi)|^2 &= 0 \end{aligned} \tag{2.82}$$

Substituindo (2.82) em (2.81), obtemos os seguintes resultados:

$$G(0) = 0$$

e

$$\begin{aligned} \Psi(0) &= \frac{\Phi(0)}{\sqrt{2}} G(0) \\ &= 0 \end{aligned}$$

Concluímos que $\Psi(\omega)$ satisfaz a condição de admissibilidade e, portanto, $\{\psi_{j,k}(x),\ k \in \mathbb{Z}\}$ é uma base wavelet.

Uma vez fixado $H(\omega)$, $G(\omega)$ não é mais arbitrário. Para provar esta afirmação fazemos a seguinte definição:

$$H_i = H(\omega) \quad \text{e} \quad H_j = H(\omega+\pi) \tag{2.83}$$

$$G_i = G(\omega) \quad \text{e} \quad G_j = G(\omega+\pi) \tag{2.84}$$

Substituindo (2.83) e (2.84) em (2.81):

$$\frac{G_i}{G_j} = -\left(\frac{H_j}{H_i}\right)^*$$

Dividindo ambos os lados desta expressão por $\left[1 + \left|\frac{G_i}{G_j}\right|^2\right]^{1/2} = \left[1 + \left|\frac{H_j}{H_i}\right|^2\right]^{1/2}$:

$$\frac{\frac{G_i}{G_j}}{\left[1 + \left|\frac{G_i}{G_j}\right|^2\right]^{1/2}} = \frac{-\left(\frac{H_j}{H_i}\right)^*}{\left[1 + \left|\frac{H_j}{H_i}\right|^2\right]^{1/2}} \tag{2.85}$$

Multiplicando o numerador e o denominador do lado esquerdo de (2.85) por $|G_j|$, os do lado direito por $H_i$ e substituindo (2.66) e (2.80) no resultado, obtemos:

$$G_i = -\frac{|H_i|}{|G_j|}\frac{G_j}{H_i^*}H_j^* = -A(\omega)H_j^* \tag{2.86}$$

onde $A(\omega)$ é uma função passa-tudo, pois $|A(\omega)| = 1,\ \forall\ \omega$.

Substituindo (2.86) em (2.81), constatamos que a condição:

$$A(\omega) + A(\omega + \pi) = 0 \tag{2.87}$$

é necessária e suficiente para que (2.81) seja válida.

A escolha $A(\omega) = e^{-j\omega}$ é muito frequente. Substituindo este valor de $A(\omega)$ em (2.86), obtemos:

$$G(\omega) = e^{-j\omega} H^*(\omega + \pi) \tag{2.88}$$

o que corresponde no domínio temporal a:

$$g(k) = (-1)^k \cdot h^*(1-k)$$

A equação (2.88) fornece uma relação entre os filtros $H$ e $G$ utilizados no processo de decomposição e síntese da TW. Qualquer par de filtros que obedeçam às condições apresentadas nesta seção pode ser utilizado. Entretanto, do ponto de vista de Processamento de Sinais, um filtro passa-baixas é uma escolha interessante para $H$. Para esta escolha, $G(\omega)$ é automaticamente um passa-altas [Fournier].

Com base nos resultados acima, podemos, então, enunciar o próximo teorema .

**Teorema 2-4** Sejam $(\mathbf{V}_{2^j})_{j\in\mathbb{Z}}$ a seqüência geradora do conjunto de espaços vetoriais de uma análise de multiresolução, $\phi(x)$ a função escala e $H$ o filtro conjugado correspondente. Seja $\psi(x)$ a função cuja transformada de Fourier $\Psi(\omega)$ é dada por:

$$\Psi(\omega) = G\left(\frac{\omega}{2}\right)\Phi\left(\frac{\omega}{2}\right)2^{-1/2} \tag{2.89}$$

$$\text{onde} \qquad G(\omega) = e^{-i\omega} H^*(\omega + \pi)$$

e $\psi_{j,k}(x) = 2^{j/2}\psi(2^j x - k)$, então:

$$\left(2^{j/2}\psi_{j,k}\left(2^j x - k\right)\right)_{k\in\mathbb{Z}}$$

é uma base ortonormal em $\mathbf{W}_{2^j}$ e $\psi(x)$ é uma wavelet ortogonal.

Este teorema estabelece uma relação entre a wavelet e a função escala. Observe que a partir de um par de filtros $H$ e $G$, que satisfaçam as condições (2.68) - (2.70), podemos encontrar uma função escala e uma wavelet, utilizando as expressões (2.71) e (2.89), respectivamente. Dependendo da escolha de $H$, a função escala $\phi(x)$ e a wavelet $\psi(x)$ podem ter boa localização tanto no domínio espacial quanto temporal [Daubechies92].

Seja $\mathbf{P}_{\mathbf{W}_{2^j}}$ o operador projeção ortogonal no espaço vetorial $\mathbf{W}_{2^j}$. Como conseqüência do Teorema 2-3, a seguinte relação é válida:

$$\mathbf{P}_{\mathbf{W}_{2^j}} f(x) = \sum_{n=-\infty}^{\infty} \langle f(u), \psi_{j,n}(u)\rangle \cdot \psi_{j,n}(x) \tag{2.90}$$

$\mathbf{P}_{\mathbf{W}_{2^j}} f(x)$ fornece o sinal de detalhamento de $f(x)$ com resolução $2^j$ e é caracterizado pelo conjunto de produtos internos:

$$\mathbf{D}_{2^j} f = \left(\langle f(u), \psi_{j,n}(u)\rangle\right)_{n\in\mathbb{Z}} \tag{2.91}$$

que contém a diferença de informação entre $A^d_{2^{j+1}} f$ e $A^d_{2^j} f$.

### 2.6.4 Implementação de uma Representação Wavelet Ortogonal

Nesta seção descreveremos o algoritmo utilizado para calcular a representação wavelet. Mostraremos que $\mathbf{D}_{2^j} f$ pode ser calculado através da convolução de $A^d_{2^{j+1}} f$ com o filtro discreto $G$, cuja forma iremos caracterizar.

Calculando o produto interno de $f(x)$ com ambos os lados de (2.75), obtemos:

$$\langle f(x), \psi_{j,n}(x)\rangle = \sum_{k=-\infty}^{\infty} \langle \psi_{-1,0}(u), \phi_{0,k-2n}(u)\rangle \cdot \langle f(x), \phi_{j+1,k}(x)\rangle \tag{2.92}$$

Substituindo (2.76) em (2.92):

$$\langle f(x), \psi_{j,n}(x)\rangle = \sum_{k=-\infty}^{\infty} g(k-2n) \cdot \langle f(x), \phi_{j+1,k}(x)\rangle \tag{2.93}$$

A equação (2.93) mostra que podemos calcular o sinal de detalhamento $\mathbf{D}_{2^j} f$ através da convolução de $A^d_{2^{j+1}} f$ com o filtro discreto de resposta impulsiva $g(-n)$, jogando fora uma amostra sim e outra não, conforme mostrado no ramo superior da Figura 2.9.

O processo de decomposição é definido pelas equações (2.58) e (2.93). A representação wavelet ortogonal do sinal discreto $A^d_1 f$ pode ser calculada, portanto, através de sucessivas decomposições de $A^d_{2^{j+1}} f$ em $A^d_{2^j} f$ e $D_{2^j} f$ para $-J \leq j \leq -1$. Podemos provar, por indução, que para qualquer $J > 0$, o sinal original $A^d_{2^1} f$ pode ser representado por:

$$\left(A^d_{2^{-J}} f, (\mathbf{D}_{2^j} f)_{-J \leq j \leq -1}\right) \tag{2.94}$$

Este conjunto de coeficientes é denominado de *representação wavelet ortogonal*, e é formado pela aproximação do sinal $A^d_{2^{-J}} f$ com uma resolução mais grosseira, e pelos sinais de detalhamento $\mathbf{D}_{2^j} f(x)$ nas resoluções $2^j$, para $-J \leq j \leq -1$. Se o sinal original tem $N$ amostras, então os sinais discretos $D_{2^j} f$ e $A^d_{2^j} f$ têm $2^j N$ amostras cada e a representação wavelet dada por (2.94) tem o mesmo número de amostras de $A^d_1 f$. Isto se deve a ortogonalidade da representação.

Este conjunto pode ser interpretado como uma decomposição do sinal original em uma base ortonormal. Podemos, então, reescrever (2.94) da seguinte forma:

$$A^d_{2^1} f = A^d_{2^{-J}} f + \sum_{j=-J}^{-1} \mathbf{D}_{2^j} f \tag{2.95}$$

Observe que a expressão (2.95) representa o sinal decomposto em dois termos: o primeiro corresponde a aproximação da função $f(x)$ no nível de resolução $-J$ e o segundo aos sinais de detalhamento para todos os níveis de resolução $-J \leq j \leq -1$. Sob o ponto de vista de processamento de sinais, podemos interpretar o primeiro termo de (2.95) como a componente de baixas freqüências do sinal e o segundo termo como a componente de altas freqüências.

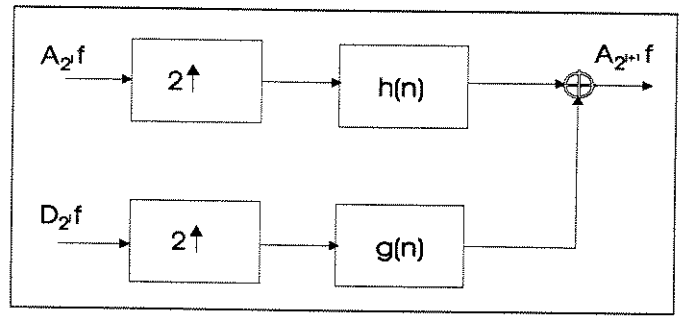



Figura 2.10: Diagrama em bloco do esquema de reconstrução da representação wavelet.

### 2.6.5 Reconstrução do Sinal de uma Representação Wavelet Ortogonal

Vimos que a representação wavelet é completa. Queremos mostrar agora que o sinal discreto original pode ser reconstruido. Como $\mathbf{W}_{2^j}$ é o complemento ortogonal de $\mathbf{V}_{2^j}$ em $\mathbf{V}_{2^{j+1}}$, $(\phi_{j,n}(x), \psi_{j,n}(x))_{n\in\mathbb{Z}}$ é uma base ortonormal em $\mathbf{V}_{2^{j+1}}$ e podemos decompor $\phi_{j+1,n}(x)$ em $\mathbf{V}_{2^{j+1}}$:

$$\phi_{j+1,n}(x) = \sum_{k=-\infty}^{\infty} \langle \phi_{j+1,n}(u), \phi_{j,k}(u) \rangle \cdot \phi_{j,k}(x) + \sum_{k=-\infty}^{\infty} \langle \phi_{j+1,n}(u), \psi_{j,k}(u) \rangle \cdot \psi_{j,k}(x) \tag{2.96}$$

Calculando o produto interno de ambos os lados da equação (2.96) com a função $f(x)$:

$$\begin{aligned} \langle f(x), \phi_{j+1,n}(x) \rangle = & \sum_{k=-\infty}^{\infty} \langle \phi_{j+1,n}(u), \phi_{j,k}(u) \rangle \cdot \langle f(x), \phi_{j,k}(x) \rangle + \\ & \sum_{k=-\infty}^{\infty} \langle \phi_{j+1,n}(u), \psi_{j,k}(u) \rangle \cdot \langle f(x), \psi_{j,k}(x) \rangle \end{aligned} \tag{2.97}$$

Fazendo uma pequena mudança de variáveis nos produtos internos em (2.97), obtemos:

$$\langle \phi_{j+1,n}(u), \phi_{j,k}(u) \rangle = \langle \phi_{-1,0}(u), \phi_{0,n-2k}(u) \rangle = h(n-2k) \tag{2.98}$$

$$\langle \phi_{j+1,n}(u), \psi_{j,k}(u) \rangle = \langle \psi_{-1,0}(u), \phi_{0,n-2k}(u) \rangle = g(n-2k) \tag{2.99}$$

Substituindo as equações (2.98) e (2.99) em (2.97):

$$\begin{aligned} \langle f(u), \phi_{j+1,n}(x) \rangle = & \sum_{k=-\infty}^{\infty} h(n-2k) \cdot \langle f(x), \phi_{j,k}(x) \rangle + \\ & \sum_{k=-\infty}^{\infty} g(n-2k) \cdot \langle f(x), \psi_{j,k}(x) \rangle \end{aligned} \tag{2.100}$$

Esta equação mostra que $A^d_{2^{j+1}} f(x)$ pode ser reconstruída inserindo-se um zero entre cada amostra de $A^d_{2^j} f$ e $\mathbf{D}_{2^j} f$ e convoluindo os resultados com os filtros $H$ e $G$, respectivamente. Na Figura 2.10 podemos ver o diagrama em blocos do processo de reconstrução.

Observe que as expressões (2.66), (2.80), (2.81) e (2.88) são equações características de dois filtros conjugados em quadratura, com respostas em freqüência $G(\omega)$ e $H(\omega)$[Vaidayanathan87]. Comparando os algoritmos de decomposição e síntese, apresentados nas Figuras 2.9 e 2.10, respectivamente, podemos observar que na síntese utilizamos os filtros de resposta impulsiva $h(n)$ e $g(n)$, ao passo que na decomposição utilizamos as versões invertidas ($h(-n)$ e $g(-n)$) destes filtros.

### 2.6.6 O Algoritmo de Daubechies

Em 1987, Ingrid Daubechies propôs um método para construção de bases wavelets de suporte compacto baseado em uma série de restrições ao filtro $H(\omega)$ [Daubechies88]. Estas restrições, expressas algebricamente, são tais que $H(\omega)$ gera uma análise de multiresolução, mas não estão baseadas em uma definição a priori dos espaços $(\mathbf{V}_{2^j})_{j\in\mathbb{Z}}$.

Pelo Lema 2-3, qualquer resposta impulsiva com duração finita de um filtro espelhado nos fornece uma função escala de suporte compacto $\phi(x)$. É imediato de (2.71) que a wavelet $\psi(x)$ também apresenta suporte compacto. Escolhas arbitrárias de filtros podem levar, entretanto, a wavelets bastante irregulares. Podemos, entretanto, impor algumas condições a esses filtros de forma a garantir que as wavelets $\psi(x)$ sejam regulares. A seguir, apresentamos o algoritmo de Daubechies para construção de filtros FIRs CQFs que gerem wavelets regulares. Para simplificar a notação, utilizaremos a transformada Z nos procedimentos a seguir.

1. Escolha um número inteiro $N > 0$
2. Escolha um polinômio ímpar $R(x)$, sujeito às restrições especificadas abaixo.
3. Faça

$$C(z) = \sum_{n=0}^{N-1} \binom{N-1+n}{n} \left(\frac{2-z-z^{-1}}{4}\right)^n + \left(\frac{2-z-z^{-1}}{4}\right)^N R\left(\frac{z+z^{-1}}{4}\right) \tag{2.101}$$

   onde $C(z)$ é um polinômio real simétrico em $z$ e $z^{-1}$. O polinômio $R(x)$ deve ser escolhido de forma que $C(\omega)$ ( que é real, dado a simetria de $C(z)$ ) seja não-negativo $\forall\ \omega$. $R(x) = 0$, por exemplo, satisfaz essa exigência.
4. Fatore $C(z)$ como $C(z) = D(z)\,D(z^{-1})$, onde $D(z)$ é um polinômio em $z$. Isso é possível de $2^K$ diferentes maneiras, onde $K$ é o grau de $C(z)$. Por exemplo, podemos tomar $D(z)$ como o polinômio com raízes iguais às raízes de $C(z)$ que se encontram dentro do círculo unitário.
5. Tome $H(z)$ como

$$H(z) = \left(\frac{1+z^{-1}}{2}\right) D(z) \tag{2.102}$$

   e $G(z)$ como em (2.88).
6. Construa $\phi(t)$ com um grau de precisão arbitrário, utilizando (2.65) de forma recursiva, e $\psi(t)$ através de (2.89).

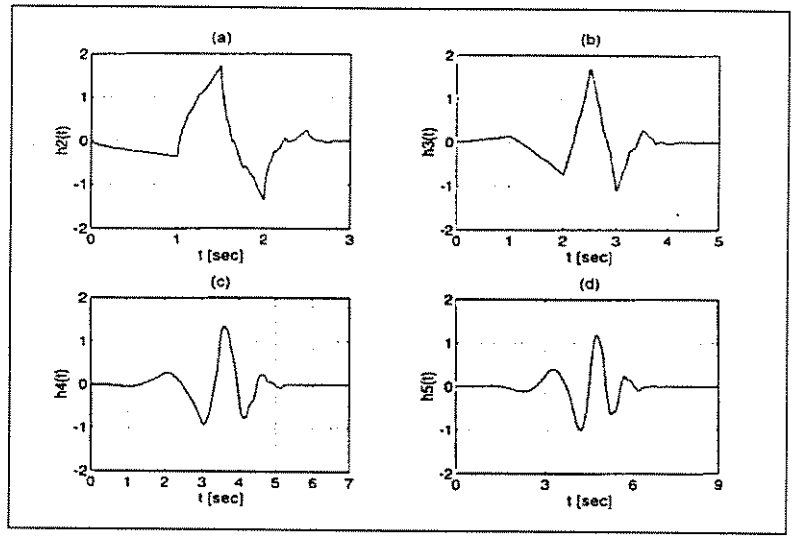



Figura 2.11: Wavelets de Daubechies para (a)$N = 2$ , (b)$N = 3$, (c)$N = 4$, (d)$N = 5$.

### Wavelets de Daubechies

O parâmetro N do algoritmo de Daubechies controla a regularidade de $\phi(t)$ e $\psi(t)$. Utilizando o algoritmo dado e fazendo $R(z) = 0$ e $D(z)$ conforme sugerido no passo 4, podemos obter uma família de wavelets ortogonais, regulares, de comprimento finito e com reconstrução exata, denominadas "wavelets de Daubechies ". Algumas das wavelets desta família são apresentadas na Figura (2.11). Nas Tabelas 2.1, 2.2 e 2.3 apresentamos os coeficientes dos filtros passa-baixas das wavelets de Daubechies 2, 3 e 4 [Daubechies88].

| | |
|---|---|
| $h_0$ | 0.482962913145 |
| $h_1$ | 0.836516303738 |
| $h_2$ | 0.224143868042 |
| $h_3$ | -0.129409522551 |

Tabela 2.1: Coeficientes do filtro passa baixas Daubechies 2.

## 2.7 Wavelets Biortogonais

Assim como na análise de Fourier, muitas propriedades interessantes podem ser obtidas se utilizarmos uma base ortonormal para expansão de sinais. Projeções ortogonais permitem que um sinal seja decomposto em um conjunto de coeficientes independentes, onde cada coeficiente corresponde a um elemento desta base ortogonal. Como já visto, estes coeficientes são calculados através da projeção da função em cada um dos elementos da base. Neste tipo de representação não há redundância.

| | |
|---|---|
| $h_0$ | 0.332670552950 |
| $h_1$ | 0.806891509311 |
| $h_2$ | 0.459877502118 |
| $h_3$ | -0.135011020010 |
| $h_4$ | -0.085441273882 |
| $h_5$ | 0.035226291882 |

Tabela 2.2: Coeficientes do filtro passa baixas Daubechies 3.

| | |
|---|---|
| $h_0$ | 0.230377813309 |
| $h_1$ | 0.714846570553 |
| $h_2$ | 0.630880767930 |
| $h_3$ | -0.027983769417 |
| $h_4$ | -0.187034811719 |
| $h_5$ | -0.030841381836 |
| $h_6$ | 0.032883011667 |
| $h_7$ | -0.010597401785 |

Tabela 2.3: Coeficientes do filtro passa baixas Daubechies 4.

Apesar da ortogonalidade apresentar muitas vantagens, em algumas aplicações, como por exemplo no processamento de imagens, é importante que a transformada utilizada apresente certas características como simetria, regularidade e suporte compacto [Antonini]. A regularidade das wavelets está relacionada com o tipo de função que a transformada pode representar. Quanto maior a regularidade, maior o número de sinais possíveis de serem representados com maior fidelidade. A simetria das wavelets, muito importante para o processamento de imagens, exige a utilização de filtros de fase linear. Tais filtros permitem cascateamento de bancos de filtros sem a necessidade de compensação de fase. Além disso, filtros de fase linear não propagam os erros de cada estágio. O suporte compacto exige que os filtros apresentem resposta impulsiva de duração finita ( filtros FIR ) e proporciona algoritmos eficientes. Por fim, os filtros utilizados no esquema de decomposição/reconstrução da transformada wavelet devem garantir a reconstrução perfeita do sinal, ou seja, nenhuma informação deve ser perdida.

Infelizmente, a única wavelet ortogonal que obedece a todas essas exigências é a wavelet de Haar [Daubechies92]. Esta wavelet, entretanto, não é contínua e, portanto, não pode ser utilizada em aplicações práticas. Para contornar este problema, relaxamos a condição de ortogonalidade, dando lugar à **Biortogonalidade**. Existem wavelets biortogonais que atendem a todas as exigências citadas. Além disso, os filtros utilizados para a sua implementação são bem mais flexíveis e fáceis de projetar.

### 2.7.1 Wavelets Ortogonais x Biortogonais

As bases wavelets biortogonais são uma generalização das bases wavelets ortogonais [Cohen92]. Segundo a teoria de Frames, apresentada na Seção 2.5, se uma wavelet $\psi$ gera um frame, podemos expressar qualquer função $f(x) \in L^2(\mathbb{R})$ através de qualquer uma das seguintes formas:

$$f(x) = \sum_{j,k} \left\langle f, \tilde{\psi}_{j,k} \right\rangle \cdot \psi_{j,k}(x) \tag{2.103}$$

ou

$$f(x) = \sum_{j,k} \langle f, \psi_{j,k} \rangle \cdot \tilde{\psi}_{j,k}(x) \tag{2.104}$$

onde $\{\psi_{j,k}(x)\}$ e $\left\{\tilde{\psi}_{j,k}(x)\right\}$ são frames duais ou bases wavelets duais, conforme definido em (2.39). Quando $\{\psi_{j,k}(x)\}$ é um frame exato, os conjuntos $\{\psi_{j,k}(x)\}$ e $\left\{\tilde{\psi}_{j,k}(x)\right\}$ são biortogonais, ou seja:

$$\left\langle \psi_{j,k}(t), \tilde{\psi}_{l,m}(t) \right\rangle = \int_{-\infty}^{\infty} \psi_{j,k}(t) \cdot \tilde{\psi}_{l,m}(t)\, dt = \delta_{jl}\delta_{km}, \qquad j,k,l,m \in \mathbb{Z} \tag{2.105}$$

(2.105) é a condição necessária para que as wavelets sejam biortogonais [Cohen92].

Se os frames $\{\psi_{j,k}(x)\}$ e $\left\{\tilde{\psi}_{j,k}(x)\right\}$ são estreitos e todos os seus componentes têm norma unitária, as wavelets formam uma base ortogonal. Como conseqüência, $\tilde{\psi}_{jk} = \psi_{j,k}$ e a função $f(x)$ pode ser expressa seguinte forma:

$$f(x) = \sum_{j,k} \langle f, \psi_{j,k} \rangle \cdot \psi_{j,k}(x) \tag{2.106}$$

Fazendo $\tilde{\psi}_{jk} = \psi_{j,k}$ em (2.103) obtemos (2.106). Podemos, então, perceber que de fato as wavelets ortogonais são apenas um caso especial das wavelets biortogonais.

Em resumo, as principais diferenças entre as wavelets ortogonais e biortogonais são:

- Os filtros ortogonais possuem necessariamente o mesmo comprimento, que deve ser par. Esta restrição não é imposta aos filtros biortogonais.
- As biortogonalidade permite que wavelets e funções escala simétricas sejam construídas. Esta é uma das principais vantagens das wavelets biortogonais.
- A identidade de Parseval (Seção 2.2) não é válida para os sistemas biortogonais, ou seja, a norma da soma dos coeficientes não é idêntica à norma da função. Esta é uma das principais desvantagens das wavelets biortogonais.

### 2.7.2 Análise de Multiresolução para Wavelets Biortogonais

A análise de multiresolução para wavelets biortogonais tem uma implementação um pouco mais complicada do que para wavelets ortogonais. Basicamente, no caso biortogonal há dois espaços de aproximação $\{V_{2^j}\}_{j\in\mathbb{Z}}$ e $\left\{\tilde{V}_{2^j}\right\}_{j\in\mathbb{Z}}$, um par de funções escala duais $\phi$ e $\tilde{\phi}$, dois espaços de detalhamento $\{W_{2^j}\}_{j\in\mathbb{Z}}$ e $\left\{\tilde{W}_{2^j}\right\}_{j\in\mathbb{Z}}$ e um par de wavelets duais $\psi$ e $\tilde{\psi}$.

Os subespaços de aproximação $V_{2^j}$ e $\widetilde{V}_{2^j}$ obedecem às propriedades apresentadas na Seção 2.5.1 e, portanto, à seguinte hierarquia:

$$\ldots \subset V_{2^{-1}} \subset V_{2^0} \subset V_{2^1} \subset V_{2^2} \subset V_{2^3} \subset \ldots \tag{2.107}$$

$$\ldots \subset \widetilde{V}_{2^{-1}} \subset \widetilde{V}_{2^0} \subset \widetilde{V}_{2^1} \subset \widetilde{V}_{2^2} \subset \widetilde{V}_{2^3} \subset \ldots \tag{2.108}$$

Para qualquer $j \in \mathbb{Z}$, as bases $\{\phi_{j,k}\}$ e $\left\{\widetilde{\phi}_{j,k}\right\}$ geram os subespaços $V_{2^j}$ e $\widetilde{V}_{2^j}$, respectivamente. $\phi_{j,k}$ e $\widetilde{\phi}_{j,k}$ são bases não-ortogonais construídas a partir de deslocamentos, dilatações e compressões das funções escala $\phi$ e $\widetilde{\phi}$, respectivamente. Estas funções, por sua vez, obedecem à seguinte relação:

$$\left\langle \phi(t), \widetilde{\phi}(t-m) \right\rangle = \delta(m) \tag{2.109}$$

O subespaço $W_{2^j}$ é o complemento não-ortogonal de $V_{2^j}$ em $V_{2^{j+1}}$ e, da mesma forma, $\widetilde{W}_{2^j}$ é o complemento não-ortogonal de $\widetilde{V}_{2^j}$ em $\widetilde{V}_{2^{j+1}}$, ou seja:

$$\begin{array}{rcll} V_{2^{j+1}} & = & V_{2^j} \oplus W_{2^j}, & V_{2^j} \text{ não é ortogonal à } W_{2^j} \\ \widetilde{V}_{2^{j+1}} & = & \widetilde{V}_{2^j} \oplus \widetilde{W}_{2^j}, & \widetilde{V}_{2^j} \text{ não é ortogonal à } \widetilde{W}_{2^j} \end{array} \tag{2.110}$$

Os subespaços de aproximação $W_{2^j}$ e $\widetilde{W}_{2^j}$ obedecem à seguinte hierarquia:

$$\ldots \subset W_{2^{-1}} \subset W_{2^0} \subset W_{2^1} \subset W_{2^2} \subset W_{2^3} \subset \ldots \tag{2.111}$$

$$\ldots \subset \widetilde{W}_{2^{-1}} \subset \widetilde{W}_{2^0} \subset \widetilde{W}_{2^1} \subset \widetilde{W}_{2^2} \subset \widetilde{W}_{2^3} \subset \ldots \tag{2.112}$$

Do mesmo modo, para qualquer $j \in \mathbb{Z}$, as bases $\psi_{j,k}$ e $\widetilde{\psi}_{j,k}$ geram os subespaços $\{W_{2^j}\}$ e $\left\{\widetilde{W}_{2^j}\right\}$, respectivamente. As bases $\psi_{j,k}$ e $\widetilde{\psi}_{j,k}$ também são não-ortogonais e construídas a partir de deslocamentos, dilatações e compressões das wavelets $\psi$ e $\widetilde{\psi}$, respectivamente. Estas, por sua vez, obedecem à seguinte relação:

$$\left\langle \psi(t), \widetilde{\psi}(t-m) \right\rangle = \delta(m) \tag{2.113}$$

Além destas propriedades, a seguinte também é válida para a análise de multiresolução biortogonal [Daubechies92]:

$$V_{2^j} \perp \widetilde{W}_{2^j} \quad e \quad \widetilde{V}_{2^j} \perp W_{2^j}$$

que implica nas seguintes relações:

$$\left\langle \phi(t), \widetilde{\psi}(t-m) \right\rangle = \left\langle \widetilde{\phi}(t), \psi(t-m) \right\rangle = \delta(m) \tag{2.114}$$

Sejam $\{\phi_{j,k}\}_{j\in\mathbb{Z}}$ e $\left\{\widetilde{\phi}_{j,k}\right\}_{j\in\mathbb{Z}}$ os dois frames duais geradores dos subespaços $V_{2^j}$ e $\widetilde{V}_{2^j}$. De acordo com a teoria de frames, podemos calcular a projeção de uma função $f(x) \in L(\mathbb{R}^2)$ na base $\{\phi_{j,k}\}_{j\in\mathbb{Z}}$. Então, a aproximação da função $f(x)$ no nível de resolução $2^j$ é dada pela seguinte expressão:

$$A_{2^j} f(x) = \sum_{n=-\infty}^{\infty} \left\langle f(u), \widetilde{\phi}_{j,n}(u) \right\rangle \phi_{j,n}(x) \tag{2.115}$$

Da mesma forma, sejam $\{\psi_{j,k}\}$ e $\left\{\widetilde{\psi}_{j,k}\right\}$ os dois frames duais geradores dos subespaços $\{W_{2^j}\}$ e $\left\{\widetilde{W}_{2^j}\right\}$. Podemos também calcular a projeção de uma função $f(x) \in L(\mathbb{R}^2)$ na base $\{\psi_{j,k}\}_{j\in\mathbb{Z}}$ geradora do espaço $W_{2^j}$. Esta projeção, no nível de resolução $2^j$, é dada pela seguinte expressão:

$$\mathbf{P}_{\mathbf{W}_{2^j}} f(x) = \sum_{n=-\infty}^{\infty} \left\langle f(u), \widetilde{\psi}_{j,n}(u) \right\rangle \cdot \psi_{j,n}(x) \tag{2.116}$$

Como $\widetilde{\phi}_{j,k} \in \widetilde{V}_{2^j} \subset \widetilde{V}_{2^{j+1}}$, podemos decompô-la em $\widetilde{V}_{2^{j+1}}$ e chegar à seguinte expressão:

$$\widetilde{\phi}_{j,n}(x) = \sum_{k=-\infty}^{\infty} \left\langle \widetilde{\phi}_{j,n}(u), \phi_{j+1,k}(u) \right\rangle \cdot \widetilde{\phi}_{j+1,k}(x) = \sum_{k=-\infty}^{\infty} \left\langle \widetilde{\phi}_{-1,0}(u), \phi_{0,k-2n}(u) \right\rangle \cdot \widetilde{\phi}_{j+1,k}(x) \tag{2.117}$$

Observe que (2.117) pode ser interpretada como uma filtragem e, assim, podemos definir $\widetilde{h}(n)$ como o filtro discreto com resposta impulsiva definida por:

$$\widetilde{h}(n) = \left\langle \widetilde{\phi}_{-1,0}(u), \phi_{0,n}(u) \right\rangle \tag{2.118}$$

Da mesma forma, como $\widetilde{\psi}_{j,k} \in \widetilde{W}_{2^j} \subset \widetilde{V}_{2^j+1}$, podemos decompô-la em $\widetilde{V}_{2^{j+1}}$ e obter a seguinte expressão:

$$\widetilde{\psi}_{j,n}(x) = \sum_{k=-\infty}^{\infty} \left\langle \widetilde{\psi}_{j,n}(u), \phi_{j+1,k}(u) \right\rangle \cdot \widetilde{\phi}_{j+1,k}(x) = \sum_{k=-\infty}^{\infty} \left\langle \widetilde{\psi}_{-1,0}(u), \phi_{0,k-2n}(u) \right\rangle \cdot \widetilde{\phi}_{j+1,k}(x) \tag{2.119}$$

Observe que (2.119) também pode ser interpretado como uma filtragem e podemos definir $\widetilde{g}(n)$ como o filtro discreto com resposta impulsiva definida por:

$$\widetilde{g}(n) = \left\langle \widetilde{\psi}_{-1,0}(u), \phi_{0,n}(u) \right\rangle \tag{2.120}$$

Substituindo (2.118) e (2.120) nas equações (2.117) e (2.119), obtemos:

$$\widetilde{\phi}_{j,k}(x) = \sum_{k=-\infty}^{\infty} \widetilde{h}(k-2n) \cdot \widetilde{\phi}_{j+1,k}(x) \tag{2.121}$$

$$\widetilde{\psi}_{j,k}(x) = \sum_{k=-\infty}^{\infty} \widetilde{g}(k-2n) \cdot \widetilde{\phi}_{j+1,k}(x) \tag{2.122}$$

Convoluindo as expressões (2.121) e (2.122) com a função $f(x)$, obtemos:

$$\left\langle f(x), \widetilde{\phi}_{j,k}(x) \right\rangle = \sum_{k=-\infty}^{\infty} \widetilde{h}(k-2n) \cdot \left\langle f(x), \widetilde{\phi}_{j+1,k}(x) \right\rangle \tag{2.123}$$

$$\left\langle f(x), \widetilde{\psi}_{j,k}(x) \right\rangle = \sum_{k=-\infty}^{\infty} \widetilde{g}(k-2n) \cdot \left\langle f(x), \widetilde{\phi}_{j+1,k}(x) \right\rangle \tag{2.124}$$

Assim, substituindo (2.123) em (2.115) e (2.124) em (2.116):

$$A_{2^j}^d f = \left( \left\langle f(x), \widetilde{\phi}_{j,k}(x) \right\rangle \right)_{n\in\mathbb{Z}} \tag{2.125}$$

$$D_{2^j}^d f = \left(\left\langle f(x), \widetilde{\psi}_{j,k}(x)\right\rangle\right)_{n\in\mathbb{Z}} \tag{2.126}$$

A aproximação discreta do sinal $f(x)$ no nível de resolução $2^j$ é dada, portanto, pela seguinte expressão:

$$A_{2^j}^d f = \left(\left\langle f(x), \widetilde{\phi}_{j,k}(x)\right\rangle\right)_{n\in\mathbb{Z}} \tag{2.127}$$

Da mesma forma, o sinal de detalhamento de $f(x)$ é dado por:

$$D_{2^j}^d f = \left(\left\langle f(x), \widetilde{\psi}_{j,k}(x)\right\rangle\right)_{n\in\mathbb{Z}} \tag{2.128}$$

A partir das expressões (2.123) e (2.124), observamos que podemos calcular $A_{2^j}^d f$ e $D_{2^j}^d f$ através da convolução de $A_{2^{j+1}}^d f$ com os filtros $\widetilde{h}$ e $\widetilde{g}$, respectivamente, descartando uma amostra sim e outra não do resultado. As equações (2.123) e (2.124) caracterizam o processo de decomposição da análise de multiresolução biortogonal.

Novamente, de acordo com a teoria de frames, podemos calcular a projeção da função $f(x)$ na base $\left\{\widetilde{\phi}_{j+1,k}\right\}_{j\in\mathbb{Z}}$. Ou seja, de forma semelhante ao que foi realizado em (2.115), a aproximação da função $f(x)$ no nível de resolução $2^{j+1}$ (um nível acima do que foi feito em (2.115) ) é dada pela seguinte expressão:

$$A_{2^{j+1}} f(x) = \sum_{n=-\infty}^{\infty} \langle f(u), \phi_{j+1,n}(u)\rangle \widetilde{\phi}_{j+1,n}(x) \tag{2.129}$$

O subespaço $W_{2^j}$ é o complemento não-ortogonal de $V_{2^j}$ em $V_{2^{j+1}}$. Portanto, o espaço $V_{2^{j+1}}$ pode ser gerado pela base $(\phi_{j,k}, \psi_{j,k})_{j\in\mathbb{Z}}$ e $\phi_{j+1,k}$ ser expressa da seguinte forma:

$$\begin{aligned}\phi_{j+1,n}(x) &= \sum_{k=-\infty}^{\infty} \left\langle \phi_{j+1,n}(u), \widetilde{\phi}_{j,k}(u)\right\rangle \cdot \phi_{j,k}(x) + \sum_{k=-\infty}^{\infty} \left\langle \phi_{j+1,n}(u), \widetilde{\psi}_{j,k}(u)\right\rangle \cdot \psi_{j,k}(x) \\ &= \sum_{k=-\infty}^{\infty} \left\langle \phi_{0,n-2k}(u), \widetilde{\phi}_{-1,0}(u)\right\rangle \cdot \phi_{j,k}(x) + \sum_{k=-\infty}^{\infty} \left\langle \phi_{0,n-2k}(u), \widetilde{\psi}_{-1,0}(u)\right\rangle \cdot \psi_{j,k}(x)\end{aligned} \tag{2.130}$$

Observe que a expressão (2.130) pode ser interpretada como a soma de duas filtragens. Definimos, então, $h(n)$ e $g(n)$ como os filtros digitais de resposta impulsiva:

$$h(n) = \left\langle \widetilde{\phi}_{-1,0}(u), \phi(u-n)\right\rangle \tag{2.131}$$

$$g(n) = \left\langle \widetilde{\psi}_{-1,0}(u), \phi(u-n)\right\rangle \tag{2.132}$$

Substituindo (2.131) e (2.132) em (2.130), obtemos:

$$\phi_{j+1,k}(x) = \sum_{k=-\infty}^{\infty} h(n-2k)\cdot\phi_{j,k}(x) + \sum_{k=-\infty}^{\infty} g(n-2k)\cdot\psi_{j,k}(x) \tag{2.133}$$

Convoluindo a expressão (2.133) com a função $f(x)$, obtemos:

$$\langle f(x), \phi_{j+1,k}(x)\rangle = \sum_{k=-\infty}^{\infty} h(n-2k)\cdot\langle f(x), \phi_{j,k}(x)\rangle + \sum_{k=-\infty}^{\infty} g(n-2k)\cdot\langle f(x), \psi_{j,k}(x)\rangle \tag{2.134}$$

Assim, o processo de reconstrução da análise de multiresolução é caracterizado pela seguinte relação:

$$A_{2^{j+1}}^{d} f(x) = A_{2^j}^{d} f(x) + D_{2^j}^{d} f(x) \tag{2.135}$$

A partir da expressão (2.135), podemos calcular a aproximação $A_{2^{j+1}}^{d} f$ através da convolução de $A_{2^j}^{d} f$ e $D_{2^j} f$ com os filtros $H$ e $G$, respectivamente, interpolando o resultado por um fator de 2.

A transformada wavelet ortogonal utiliza os mesmos filtros, $h$ e $g$, tanto para a reconstrução, como para a decomposição. Como estudado nesta seção, no caso biortogonal, os filtros de reconstrução, $h$ e $g$, são diferentes dos filtros de decomposição, $\widetilde{h}$ e $\widetilde{g}$. Além disso, os filtros correspondentes à wavelet biortogonal são mais flexíveis, fáceis de projetar e podem ser simétricos [Antonini], o que é impossível para o caso ortogonal. Na Figura 2.12 é apresentado um diagrama em blocos dos esquemas de decomposição e síntese da transformada wavelet biortogonal.

### 2.7.3 Exemplos de Wavelets Biortogonais

Apresentaremos aqui alguns exemplos de wavelets biortogonais simétricas, regulares, de comprimento finito e reconstrução exata. Daubechies [Daubechies92] provou que uma regularidade alta pode ser obtida, tanto para $\psi$ como para $\widetilde{\psi}$, desde que escolhamos filtros suficientemente longos. Sejam $H(\omega)$ e $\widetilde{H}(\omega)$ as T.F. de $h(n)$ e $\widetilde{h}(n)$. Se as funções $\psi$ e $\widetilde{\psi}$ forem, respectivamente, diferenciáveis $(k-1)$ e $\left(\widetilde{k}-1\right)$ vezes, então as suas respostas em freqüência ($H(\omega)$ e $\widetilde{H}(\omega)$) são divisíveis por $(1+e^{-j\omega})^{k}$ e $(1+e^{-j\omega})^{\widetilde{k}}$, respectivamente, de tal forma que os filtros correspondentes ($h$ e $\widetilde{h}$) tenham um comprimento maior que $k$ e $\widetilde{k}$, respectivamente.

A divisibilidade de $\widetilde{H}(\omega)$ por $(1+e^{-j\omega})^{\widetilde{k}}$ significa que $\psi$ tem $\widetilde{k}$ momentos nulos consecutivos [Daubechies92], ou seja:

$$\int x^{l}\psi(x)\,dx = 0, \qquad para\ l = 0, 1, ..., \widetilde{k}-1. \tag{2.136}$$

É conhecido, e pode ser provado utilizando as séries de Taylor [Daubechies92], que se $\psi$ tem $\widetilde{k}$ momentos nulos, então, os coeficientes $\langle f, \psi_{mn}\rangle$ irão representar funções, $\widetilde{k}$ vezes diferenciáveis.

Muitos exemplos de wavelets biortogonais razoavelmente regulares podem ser construídos. Para nossas aplicações, a regularidade de $\widetilde{\psi}_{mn}$, que está diretamente ligada ao número de momentos de $\psi$, é mais importante que a regularidade de $\psi_{mn}$ ou ao número de momentos nulos de $\widetilde{\psi}$. Dentro dos limites impostos pelo suporte compacto, tentaremos escolher $\widetilde{k}$ o maior possível [Antonini].

Em termos de $H(\omega)$ e $\widetilde{H}(\omega)$, a equação de reconstrução perfeita é expressa da seguinte forma [Daubechies92]:

$$H(\omega)\cdot\widetilde{H}(\omega) + H(\omega+\pi)\cdot\widetilde{H}(\omega+\pi) = 2 \tag{2.137}$$

Combinando as expressões (2.136), (2.137) e a imposição de divisibilidade de $H(\omega)$ e $\widetilde{H}(\omega)$ por $(1+e^{-j\omega})^{k}$ e $(1+e^{-j\omega})^{\widetilde{k}}$, chega-se [Cohen92], após algumas manipulações, à seguinte expressão:

$$H(\omega)\cdot\widetilde{H}(\omega) = \cos(\omega/2)^{2l}\left[\sum_{p=0}^{l-1}\binom{l-1+p}{p}\cdot \operatorname{sen}(\omega/2)^{2p} + \operatorname{sen}(\omega/2)^{2l} R(\omega)\right] \tag{2.138}$$

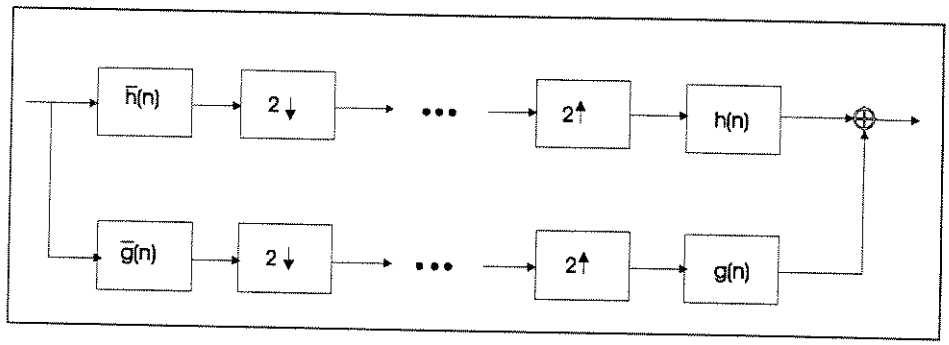



Figura 2.12: Diagrama em bolcos do algoritmo de decomposição e síntese da transformada wavelet biortogonal.

onde $R(\omega)$ é um polinômio ímpar em $\cos(\omega)$, e $2l = k + \widetilde{k}$ ( o fato dos filtros $h$ e $\widetilde{h}$ serem simétricos garante que $k + \widetilde{k}$ seja par ).

A partir da equação (2.138) é possível construir muitos exemplos de filtros. Estudaremos, em particular, três exemplos pertencentes a três diferentes famílias.

### Família de Wavelets Spline Biortogonais Cohen-Daubechies-Feauveau

Se $R(\omega) = 0$ em (2.138) e a resposta em freqüência do filtro passa-baixas dual for definida pela seguinte expressão:

$$\widetilde{H}(\omega) = \cos(\omega/2)^{\widetilde{k}} \cdot e^{-jk\omega} \tag{2.139}$$

onde $k = 0$, se $\widetilde{k}$ é par, e $k = 1$, se $\widetilde{k}$ é ímpar, obtemos uma família de filtros conhecida como filtros spline, conforme apresentado em [Daubechies92]. Os filtros splines são simétricos, suaves e tem coeficientes diádicos. Para estes, a resposta em freqüência do filtro passa-baixas é dada pela seguinte expressão:

$$H(\omega) = \cos(\omega/2)^{2l-\widetilde{k}}\, e^{jk\omega/2} \left[ \sum_{p=0}^{l-1} \binom{l-1+p}{p} \cdot \operatorname{sen}(\omega/2)^{2p} \right] \tag{2.140}$$

Na Tabela 2.4 apresentamos os coeficientes dos filtros passa-baixas para $l = 3$ e $\widetilde{k} = 2$. Observe que os filtros deste exemplo são simétricos com relação a 0, e possuem um comprimento ímpar. Esta é uma característica dos filtros spline

### Variante Família de Wavelets Spline biortogonais, com Comprimentos Semelhantes

Para construir esta família de filtros fazemos $R(\omega) = 0$ em (2.138) e fatoramos o lado direito da expressão quebrando o polinômio, em $\operatorname{sen}(\omega/2)$ em um produto de dois polinômios em $\operatorname{sen}(\omega/2)$. Na tentativa de tornar o comprimento do filtro $h$ mais próximo ao comprimento do filtro $\widetilde{h}$, um dos polinômios resultantes, que possui coeficientes reais, é definido como $H$ e o outro como $\widetilde{H}$.

O exemplo que vemos na Tabela 2.5 é o filtro mais curto desta família. Corresponde à $l = k = 4$.

### Família de Wavelets Próximas às Ortonormais

Existem muitos exemplos de wavelets biortogonais para as quais $R(\omega) \neq 0$. Em particular, existe uma escolha de $R(\omega)$ para a qual os filtros em (2.138) estão muito próximos e, conseqüentemente, muito próximos a filtros wavelets ortonormais. Um destes casos é o filtro piramidal Laplaciano, para o qual $l = k = 2$ e $R(\omega) = 48 \cdot \cos(\omega/2)/175$. Os coeficientes deste filtro podem ser vistos na Tabela 2.6.

| n | 0 | ±1 | ±2 | ±3 | ±4 |
|---|---|---|---|---|---|
| $2^{-1/2} \cdot h_n$ | 45/64 | 19/64 | -1/8 | 3/64 | 3/128 |
| $2^{-1/2} \cdot \tilde{h}_n$ | 1/2 | 1/4 | 0 | 0 | 0 |

Tabela 2.4: Coeficientes dos filtros spline para $l = 3$ e $\tilde{k} = 2$.

| n | 0 | ±1 | ±2 | ±3 | ±4 |
|---|---|---|---|---|---|
| $2^{-1/2} \cdot h_n$ | 0.602949 | 0.266864 | -0.078223 | -0.016864 | 0.026749 |
| $2^{-1/2} \cdot \tilde{h}_n$ | 0.557543 | 0.295636 | -0.028772 | -0.045636 | 0 |

Tabela 2.5: Coeficientes do filtro variante de um filtro spline com comprimentos semelhantes para $l = k = 4$.

| n | 0 | ±1 | ±2 | ±3 | ±4 |
|---|---|---|---|---|---|
| $2^{-1/2} \cdot h_n$ | 0.6 | 0.25 | -0.05 | 0 | 0 |
| $2^{-1/2} \cdot \tilde{h}_n$ | 17/28 | 73/280 | -3/56 | -3/280 | 0 |

Tabela 2.6: Coeficientes do filtro variante de um filtro spline com comprimentos mais semelhantes para $l = k = 2$ e $R(\omega) = 48 \cdot \cos(\omega/2)/175$.

Os dois filtros biortogonais deste exemplo são ambos muito próximos a um filtro wavelet ortonormal de comprimento 6 apresentado em [Daubechies90], onde eles foram denominados de coiflets. Sendo um filtro ortogonal, o coiflet não é simétrico. Os próximos ($k = l = 4$) filtros $h$ e $\tilde{h}$ desta família possuem comprimentos 9 e 15 e são ambos próximos ao coiflet de comprimento 12.

## 2.8 Extensão da Transformada Wavelet para o Caso Bidimensional

A teoria wavelet pode ser facilmente estendida para qualquer dimensão $n > 1$. Para que possamos definir a transformada wavelet bidimensional, vamos considerar que a wavelet bidimen-

sional $\psi(x,y)$ seja uma função oscilatória, de curta duração e de energia finita, ou seja:

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\psi(x,y)\,dx\,dy<\infty \tag{2.141}$$

Vamos ainda supor que $\psi(x,y)$ satisfaça a seguinte propriedade:

$$C_\psi=\int_{-\infty}^{\infty}\frac{|\Psi(a^{-1}\cdot\omega_x,a^{-1}\cdot\omega_y)|^2}{a^{-1}}da<\infty,\qquad\forall\,(\omega_x,\omega_y)\in\mathbb{R}^2 \tag{2.142}$$

onde $\Psi(\omega_x,\omega_y)$ é a transformada de Fourier de $\psi(x,y)$. A expressão (2.142) é a condição de admissibilidade para o caso bidimensional. Esta condição é satisfeita, por exemplo, por uma wavelet isotrópica, ou seja, uma wavelet para a qual $\psi(x,y)=\rho\sqrt{x^2+y^2}$ e cuja transformada de Fourier é nula na origem. Por uma questão de normalização, vamos supor ainda que $|\psi|=1$.

Definimos, então, a transformada wavelet contínua bidimensional como:

$$CWT(a;b_x,b_y)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(x,y)\cdot\psi_{a;b_x,b_y}(x,y)\,dxdy \tag{2.143}$$

onde $\psi_{a;b_x,b_y}(x,y)$ são as wavelets-filhotes geradas a partir da wavelet-mãe $\psi(x,y)$ através da seguinte relação:

$$\psi_{a;b_x,b_y}(x,y)=a^{-1}\cdot\psi\left(\frac{x-b_x}{a},\frac{y-b_y}{a}\right) \tag{2.144}$$

A CWT bidimensional (2-D) apresenta as mesmas propriedades da CWT unidimensional (1-D) como, por exemplo, a conservação de energia:

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}|CWT(a;b_x,b_y)|^2\cdot a^{-1}da\,db_xdb_y=C_\psi\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}|f(x,y)|^2\,dx\,dy \tag{2.145}$$

Assim como no caso 1-D, a igualdade (2.145) pode ser demonstrada utilizando o Teorema de Parseval.

Se $\psi(x,y)$ satisfaz (2.142), então existe uma CWT bidimensional inversa, definida por:

$$f(x,y)=\frac{1}{C_\psi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}CWT(a;b_x,b_y)\cdot a^{-1}\psi\left(\frac{x-b_x}{a},\frac{y-b_y}{a}\right)da\,db_xdb_y \tag{2.146}$$

A transformada wavelet discreta bidimensional é semelhante à transformada unidimensional. Fixamos os valores de $a_0$ e $b_0$ e fazemos $a=a_0^{-j}$, $b_x=n\cdot b_0\cdot a_0^j=k_x\cdot a_0^j$ e $b_y=m\cdot b_0\cdot a_0^j=k_y\cdot a_0^j$. A DWT é, então, definida por:

$$DWT(j;k_x,k_y)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(x,y)\cdot a^{-1}\cdot\psi^*\left(a_0^{-j}x-k_x,a_0^{-j}y-k_y\right)dxdy \tag{2.147}$$

As wavelets diádicas ( $a_0=2$) bidimensionais são definidas pela seguinte expressão:

$$\psi_{j;k_x,k_y}(x,y)=2^{-j}\cdot\psi\left(2^{-j}x-k_x,2^{-j}y-k_y\right) \tag{2.148}$$

### 2.8.1 Análise de Multiresolução Bidimensional

Uma análise de multiresolução em $\mathbf{L}^2(\mathbb{R}^2)$ é formada por uma seqüência de subespaços em $\mathbf{L}^2(\mathbb{R}^2)$ que satisfaz às propriedades 1-8 (Seção 2.6.1) para o caso bidimensional. Seja $\left(\mathbf{V}_{2^j}^2\right)_{j\in\mathbb{Z}}$ a seqüência de subespaços em $\mathbf{L}^2(\mathbb{R}^2)$. A aproximação do sinal $f(x,y)$ no nível de resolução $2^j$ é igual a sua projeção no espaço vetorial $\mathbf{V}_{2^j}^2$. Neste capítulo consideraremos a utilização de wavelets ortogonais, mas os resultados podem ser facilmente estendidos para casos não-ortogonais, como, por exemplo, para wavelets biortogonais.

#### A Aproximação por Multiresolução em $\mathbf{L}^2(\mathbb{R}^2)$

O Teorema 2-1 é válido para o caso bidimensional e podemos demonstrar que há uma única função escala $\phi(x,y)$ bidimensional cujos deslocamentos, dilatações e compressões geram uma base ortonormal para cada espaço $\mathbf{V}_{2^j}^2$. Seja $\phi_{j;k_x,k_y}(x,y) = 2^j\phi\left(2^{-j}x-k_x, 2^{-j}y-k_y\right)$. A família de funções:

$$\left(2^j\phi\left(2^{-j}x-k_x, 2^{-j}y-k_y\right)\right)_{(k_x,k_y)\in\mathbb{Z}^2}$$

é uma base ortonormal em $\mathbf{V}_{2^j}^2$. Para uma aproximação por multiresolução em $\mathbf{L}^2(\mathbb{R}^2)$, existe apenas uma função $\phi(x,y)$.

Nesta seção, descreveremos o caso da aproximação por multiresolução bidimensional [Mallat89b]. Nesta, cada espaço vetorial $\mathbf{V}_{2^j}^2$ pode ser decomposto como um produto tensorial de dois subespaços idênticos em $\mathbf{L}^2(\mathbb{R})$ :

$$\mathbf{V}_{2^j}^2 = \mathbf{V}_{2^j}^1 \otimes \mathbf{V}_{2^j}^1 \tag{2.149}$$

A seqüência de espaços vetoriais $\left(\mathbf{V}_{2^j}^2\right)_{j\in\mathbb{Z}}$ forma uma aproximação por multiresolução em $\mathbf{L}^2(\mathbb{R}^2)$ se, e apenas se, $\left(\mathbf{V}_{2^j}^1\right)_{j\in\mathbb{Z}}$ é uma aproximação por multiresolução em $\mathbf{L}^2(\mathbb{R})$, conforme definido. Prova-se [Mallat89b] facilmente que a função escala $\phi(x,y)$ é uma função separável e, portanto, pode ser expressa da seguinte forma:

$$\phi(x,y) = \phi(x)\,\phi(y) \tag{2.150}$$

onde $\phi(x)$ é a função escala unidimensional relativa a $\left(\mathbf{V}_{2^j}^1\right)_{j\in\mathbb{Z}}$. Da mesma forma, a base ortogonal em $\mathbf{V}_{2^j}^2$ pode ser expressa da seguinte forma :

$$\left(\phi_{j;k_x,k_y}(x,y)\right)_{(k_x,k_y)\in\mathbb{Z}^2} = \left(\phi_{j,k_x}(x)\cdot\phi_{j,k_y}(y)\right)_{(k_x,k_y)\in\mathbb{Z}^2} \tag{2.151}$$

Para o processamento de imagens, a utilização de uma aproximação por multiresolução separável dá ênfase às direções verticais e horizontais da imagem, representadas pelas variáveis $x$ e $y$. A projeção de uma função $f(x,y)\in\mathbf{L}^2(\mathbb{R}^2)$ no espaço $\mathbf{V}_{2^j}^2$ é dado por:

$$A_{2^j}f(x,y) = \sum_{k_x=-\infty}^{\infty}\sum_{k_x=-\infty}^{\infty}\left\langle f(u,v),\phi_{j;k_x,k_y}(u,v)\right\rangle\cdot\phi_{j;k_x,k_y}(x,y) \tag{2.152}$$

Logo, a aproximação de um sinal $f(x,y)$ no nível de resolução $2^j$ é caracterizada pelo seguinte conjunto de produtos internos:

$$A_{2^j}^d f = \left(\left\langle f(x,y),\phi_{j,k_x}(x)\cdot\phi_{j,k_y}(y)\right\rangle\right)_{(k_x,k_y)\in\mathbb{Z}^2} \tag{2.153}$$

que denominaremos de aproximação discreta do sinal $f(x,y)$ no nível de resolução $2^j$.

### Implementação de uma Transformada de Multiresolução Bidimensional

Na prática, os sinais bidimensionais $f(x,y)$ que iremos trabalhar são imagens discretas com resolução finita. Assim como no caso unidimensional, vamos supor a resolução máxima é igual a 1 e, portanto, $A_1^d f(x,y)$ é a aproximação discreta com resolução máxima. Se $A_1^d f(x,y)$ tem $N$ píxeis, pode-se mostrar que para $j<0$, a sua aproximação discreta com resolução $2^j$ $\left(A_{2^j}^d f\right)$ tem $2^j N$ píxeis. Os possíveis problemas de bordas são amenizados supondo-se que a imagem original é simétrica com relação às bordas horizontais e verticais.

Seja $\left(\mathbf{V}_{2^j}^2\right)_{j\in\mathbb{Z}}$ uma aproximação por multiresolução bidimensional e $\phi(x,y)$ a função escalar correspondente. Sabemos que $\phi_{j;k_x,k_y}(x,y)$ pertence a $\mathbf{V}_{2^j}^2 \subset \mathbf{V}_{2^{j+1}}^2$. Logo, podemos expandir a função $\phi_{j;k_x,k_y}(x,y)$ na base ortonormal $\phi_{j+1;k_x,k_y}(x,y)$ do espaço $\mathbf{V}_{2^{j+1}}^2$:

$$\phi_{j;k_x,k_y}(x,y) = \sum_{k_x=-\infty}^{\infty}\sum_{k_x=-\infty}^{\infty}\left\langle \phi_{j;n_x,n_y}(u,v), \phi_{j+1;k_x,k_y}(u,v)\right\rangle \cdot \phi_{j+1;k_x,k_y}(x,y) \tag{2.154}$$

Fazendo uma pequena substituição de variáveis em (2.154), obtemos:

$$\phi_{j;k_x,k_y}(x,y) = \sum_{k_x=-\infty}^{\infty}\sum_{k_x=-\infty}^{\infty}\left\langle \phi_{-1;0,0}(u,v), \phi_{0;(k_x-2n_x),(k_y-2n_y)}(u,v)\right\rangle \cdot \phi_{j+1;k_x,k_y}(x,y) \tag{2.155}$$

Podemos interpretar a equação (2.155) como uma filtragem e definir $H_2$ como o filtro discreto bidimensional de resposta impulsiva dada por:

$$\begin{aligned} h_2(n,m) &= \langle \phi_{-1;0,0}(x,y), \phi_{0;n,m}(x,y)\rangle \\ &= \langle \phi_{-1;0}(x), \phi_{0;n}(x)\rangle \cdot \langle \phi_{-1;0}(y), \phi_{0;m}(y)\rangle \\ &= h(n)\cdot h(m) \end{aligned} \tag{2.156}$$

Observe que a resposta impulsiva de $H_2$ pode ser decomposta no produto das respostas impulsivas de dois filtros digitais unidimensionais, de resposta impulsiva $h(.)$ conforme definido em (2.57). Portanto, podemos dizer que o filtro $H_2$ é separável nas direções $x$ e $y$.

Substituindo (2.156) em (2.155) e convoluindo o resultado com $f(x,y)$, obtemos:

$$\left\langle f(x,y), \phi_{j;k_x,k_y}(x,y)\right\rangle = \sum_{k_x=-\infty}^{\infty} h(k_y-2n_y)\cdot \sum_{k_x=-\infty}^{\infty} h(k_x-2n_x)\cdot \left\langle f(x,y), \phi_{j+1;k_x,k_y}(x,y)\right\rangle \tag{2.157}$$

Comparando (2.157) e (2.153), podemos observar que $A_{2^j}^d f$ pode ser calculada através de filtragens sucessivas de $A_{2^{j+1}}^d f$ em ambas as direções. Inicialmente, filtra-se $A_{2^{j+1}}^d f$ na direção $x$ (linhas) com o filtro unidimensional $h(.)$, descartando-se uma amostra sim e outra não nesta direção, ou seja, uma linha sim e outra não. Em seguida, o resultado é filtrado na direção $y$ (colunas) com o mesmo filtro unidimensional, descartando-se uma amostra sim e outra não, nesta direção, ou seja, uma coluna sim e outra não.

### O Sinal de Detalhamento em $\mathbf{L}^2(\mathbb{R}^2)$

Assim como no caso unidimensional, o sinal de detalhamento no nível de resolução $2^j$ é igual à projeção ortogonal do sinal no subespaço $\mathbf{W}_{2^j}^2$, que é o complemento ortogonal de $\mathbf{V}_{2^j}^2$ em $\mathbf{V}_{2^{j+1}}^2$. O teorema seguinte nos dá uma extensão simples do Teorema 2-3 e estabelece que podemos construir uma base ortonormal $\mathbf{W}_{2^j}^2$ através do escalonamento e deslocamento das três funções wavelets bidimensionais, $\psi^1(x,y)$, $\psi^2(x,y)$ e $\psi^3(x,y)$.

**Teorema 2-5** Seja $(\mathbf{V}_{2^j}^2)_{j\in\mathbb{Z}}$ uma aproximação por multiresolução separável em $\mathbf{L}^2(\mathbb{R}^2)$. Seja $\phi(x,y)=\phi(x)\phi(y)$ a função escala bidimensional separável e $\psi(x)$ a wavelet unidimensional associada à função escala $\phi(x)$ através da expressão (2.95). Então, as três wavelets definidas como:

$$\begin{aligned}\psi^1(x,y) &= \phi(x)\psi(y)\\ \psi^2(x,y) &= \psi(x)\phi(y)\\ \psi^3(x,y) &= \psi(x)\psi(y)\end{aligned} \tag{2.158}$$

são tais que

$$\begin{aligned}&\left(2^j\psi^1\left(2^jx-k_x,2^jy-k_y\right),2^j\psi^2\left(2^jx-k_x,2^jy-k_y\right),\right.\\ &\left.2^j\psi^3\left(2^jx-k_x,2^jy-k_y\right)\right)_{(k_x,k_y)\in\mathbb{Z}^2}\end{aligned} \tag{2.159}$$

é uma base ortonormal em $\mathbf{W}_{2^j}^2$.

A diferença de informação entre $A_{2^{j+1}}^d f$ e $A_{2^j}^d f$ é igual à projeção ortogonal de $f(x,y)$ em $\mathbf{W}_{2^j}^2$. Baseado no Teorema 2-5, podemos afirmar que a diferença de informação é dada pelas três imagens de detalhes:

$$D_{2^j}^1 f=\left(\left\langle f(x,y),2^j\psi^1\left(2^jx-k_x,2^jy-k_y\right)\right\rangle\right)_{(k_x,k_y)\in\mathbb{Z}^2} \tag{2.160}$$

$$D_{2^j}^2 f=\left(\left\langle f(x,y),2^j\psi^2\left(2^jx-k_x,2^jy-k_y\right)\right\rangle\right)_{(k_x,k_y)\in\mathbb{Z}^2} \tag{2.161}$$

$$D_{2^j}^3 f=\left(\left\langle f(x,y),2^j\psi^3\left(2^jx-k_x,2^jy-k_y\right)\right\rangle\right)_{(k_x,k_y)\in\mathbb{Z}^2} \tag{2.162}$$

Como as três wavelets $\psi^1(x,y)$, $\psi^2(x,y)$ e $\psi^3(x,y)$ são dadas pelos produtos separáveis das funções $\psi$ e $\phi$, (2.160), (2.161) e (2.162) podem ser expressas da seguinte forma:

$$D_{2^j}^1 f=\left(f(x,y)*2^{j/2}\phi\left(2^jx-k_x\right)\cdot 2^{j/2}\psi\left(2^jy-k_y\right)\right)_{(k_x,k_y)\in\mathbb{Z}^2} \tag{2.163}$$

$$D_{2^j}^2 f=\left(f(x,y)*2^{j/2}\psi\left(2^jx-k_x\right)\cdot 2^{j/2}\phi\left(2^jy-k_y\right)\right)_{(k_x,k_y)\in\mathbb{Z}^2} \tag{2.164}$$

$$D_{2^j}^3 f=\left(f(x,y)*2^{j/2}\psi\left(2^jx-k_x\right)\cdot 2^{j/2}\psi\left(2^jy-k_y\right)\right)_{(k_x,k_y)\in\mathbb{Z}^2} \tag{2.165}$$

### Implementação de uma Representação Wavelet Bidimensional Ortogonal

Assumindo que o Teorema 2-5 é válido, existe um conjunto de wavelets $\psi^i_{j;k_x,k_y}(x,y)_{1\leq i\leq 3}$, separáveis conforme definido em (2.158). Cada wavelet $\psi^i_{j;k_x,k_y}(x,y)$ pertence ao espaço $\mathbf{W}^2_{2^j} \subset \mathbf{V}^2_{2^{j+1}}$. Logo, podemos expandir $\psi^i_{j;k_x,k_y}(x,y)$ na base ortonormal $\phi_{j+1;k_x,k_y}(x,y)$ do espaço $\mathbf{V}^2_{2^{j+1}}$, da seguinte forma:

$$\psi^i_{j;k_x,k_y}(x,y) = \sum_{k_x=-\infty}^{\infty}\sum_{k_x=-\infty}^{\infty}\left\langle \psi^i_{j;k_x,k_y}(u,v), \phi_{j+1;k_x,k_y}(u,v)\right\rangle \cdot \phi_{j+1;k_x,k_y}(x,y) \tag{2.166}$$

Fazendo uma substituição de variáveis em (2.166), obtemos:

$$\psi^i_{j;k_x,k_y}(x,y) = \sum_{k_x=-\infty}^{\infty}\sum_{k_x=-\infty}^{\infty}\left\langle \psi^i_{-1;0,0}(u,v), \phi_{0;(k_x-2n_x),(k_y-2n_y)}(u,v)\right\rangle \cdot \phi_{j+1;k_x,k_y}(x,y) \tag{2.167}$$

Podemos interpretar a equação (2.167) como uma filtragem e definir $G^i_2$ como o filtro discreto bidimensional de resposta impulsiva dada por: :

$$g^i(n,m) = \left\langle \psi^i_{-1;0,0}(u,v), \phi_{0;n,m}(u,v)\right\rangle \tag{2.168}$$

onde o parâmetro $i$ especifica o índice da wavelet utilizada em (2.168), de acordo com o definido em (2.158). Para cada wavelet $i$, existe um filtro correspondente de resposta impulsiva $g^i(n,m)$. Assim, para $1 \leq i \leq 3$, os seguintes filtros bidimensionais são definidos:

$$\begin{aligned} g^1(n,m) &= \langle \phi_{-1,0}(u)\,\psi_{-1,0}(v), \phi_{0,n}(u)\,\phi_{0,m}(v)\rangle \\ &= \langle \phi_{-1,0}(u), \phi_{0,n}(u)\rangle \cdot \langle \psi_{-1,0}(v), \phi_{0,m}(v)\rangle \\ &= h(n)\cdot g(m) \end{aligned} \tag{2.169}$$

$$\begin{aligned} g^2(n,m) &= \langle \psi_{-1,0}(u)\,\phi_{-1,0}(v), \phi_{0,n}(u)\,\phi_{0,m}(v)\rangle \\ &= \langle \psi_{-1,0}(u), \phi_{0,n}(u)\rangle \cdot \langle \phi_{-1,0}(v), \phi_{0,m}(v)\rangle \\ &= g(n)\cdot h(m) \end{aligned} \tag{2.170}$$

$$\begin{aligned} g^3(n,m) &= \langle \psi_{-1,0}(u)\,\psi_{-1,0}(v), \phi_{0,n}(u)\,\phi_{0,m}(v)\rangle \\ &= \langle \psi_{-1,0}(u), \phi_{0,n}(u)\rangle \cdot \langle \psi_{-1,0}(v), \phi_{0,m}(v)\rangle \\ &= g(n)\cdot g(m) \end{aligned} \tag{2.171}$$

Observe que a resposta impulsiva dos filtros bidimensionais definidos pelas equações (2.169), (2.170) e (2.171) podem ser decompostos como produtos das respostas impulsivas de dois filtros unidimensionais $h(.)$ e $g(.)$, definidos em (2.57) e (2.76). Ou seja, estes filtros bidimensionais são separáveis e resultam em produtos de filtros unidimensionais que operam em duas direções ($x$ e $y$) distintas.

Convoluindo (2.167) com $f(x,y)$, e substituindo as expressões (2.169), (2.170) e (2.171) no resultado, obtemos:

$$\left\langle f(x,y), \psi^1_{j;k_x,k_y}(x,y)\right\rangle = \sum_{k_x=-\infty}^{\infty} g(k_y-2n_y)\sum_{k_x=-\infty}^{\infty} h(k_x-2n_x)\cdot\left\langle f(x,y), \psi^1_{j+1;k_x,k_y}(x,y)\right\rangle \tag{2.172}$$

$$\left\langle f(x,y), \psi^2_{j;k_x,k_y}(x,y)\right\rangle = \sum_{k_x=-\infty}^{\infty} h(k_y-2n_y) \sum_{k_x=-\infty}^{\infty} g(k_x-2n_x)\cdot\left\langle f(x,y), \psi^2_{j+1;k_x,k_y}(x,y)\right\rangle \tag{2.173}$$

$$\left\langle f(x,y), \psi^3_{j;k_x,k_y}(x,y)\right\rangle = \sum_{k_x=-\infty}^{\infty} g(k_y-2n_y v) \sum_{k_x=-\infty}^{\infty} g(k_x-2n_x)\cdot\left\langle f(x,y), \psi^3_{j+1;k_x,k_y}(x,y)\right\rangle \tag{2.174}$$

Comparando estas expressões com (2.163), (2.164) e (2.165), podemos concluir que as diferentes imagens de detalhes $D^i_{2^j} f$ podem ser calculadas através de filtragens sucessivas com os filtros unidimensionais em ambas as direções. Inicialmente, $A^d_{2^{j+1}} f$ é filtrado na direção $x$ (linhas) com um filtro unidimensional, descartando-se uma amostra sim e outra não nesta direção. Em seguida, o resultado é filtrado na direção $y$ (colunas) com outro filtro unidimensional, descartando-se uma amostra sim e outra não, nesta direção. O filtro utilizado em cada direção, $g(.)$ ou $h(.)$, vai depender do valor de $i$ conforme definido pelas equações (2.172), (2.173) e (2.174). Nestas, $n_y$ corresponde à direção $y$, enquanto que $n_x$ corresponde à direção $x$.

A transformada wavelet bidimensional decompõe $A^d_{2^{-j+1}} f$ em $A^d_{2^j} f$, $D^1_{2^j} f$, $D^2_{2^j} f$ e $D^3_{2^j} f$. Este algoritmo é ilustrado pelo diagrama de blocos apresentado na Figura 2.13. A representação wavelet bidimensional pode, então, ser calculada através do cascateamento de dois algoritmos piramidais unidimensionais. Inicialmente, aplicamos o algoritmo unidimensional nas linhas da imagem e depois nas suas colunas. O algoritmo consiste em realizarmos a filtragem das linhas de $A^d_{2^{-j+1}} f$ com um filtro unidimensional, descartando-se uma linha sim e outra não. A seguir, realizamos a filtragem das colunas do sinal resultante com um outro filtro unidimensional, descartando-se uma coluna sim e outra não. Os filtros unidimensionais utilizados nesta decomposição são os filtros conjugados em quadratura $H^*$ e $G^*$, que possuem respostas impulsivas $h(-n)$ e $g(-n)$, respectivamente.

A imagem $A^d_{2^j} f$ corresponde às freqüências mais baixas de $f(x)$, $D^1_{2^j} f$ às freqüências verticais altas, $D^2_{2^j} f$ às freqüências horizontais altas e $D^3_{2^j} f$ às altas freqüências em ambas às direções. O arranjo das imagens resultantes para três estágios de decomposição pode ser visto na Figura 2.14

Para qualquer $J > 0$, uma imagem $A^d_1 f(x,y)$ é completamente representada pelas $3J+1$ imagens discretas:

$$\left(A^d_{2^{-J}} f, \left(D^1_{2^j} f\right)_{-J\le j\le -1}, \left(D^2_{2^j} f\right)_{-J\le j\le -1}, \left(D^3_{2^j} f\right)_{-J\le j\le -1}\right) \tag{2.175}$$

Este conjunto de imagens é denominado representação wavelet em duas dimensões. A imagem $A^d_{2^{-J}} f$ é a aproximação de $A^d_1 f$ com nível de resolução $2^{-J}$. $\left\{D^k_{2^j} f\right\}_{1\le k\le 3}$ são as imagens de detalhamento para as diferentes orientações e níveis de resoluções. Se a imagem original tem $N$ píxeis, cada imagem $A^d_{2^{-J}} f$, $D^1_{2^j} f$, $D^2_{2^j} f$ e $D^3_{2^j} f$ tem $2^j N$ píxeis ($j < 0$ ). O número total de píxeis desta nova representação é igual ao número de píxeis da imagem original. Por causa da ortogonalidade da representação, o volume de dados não foi aumentado.

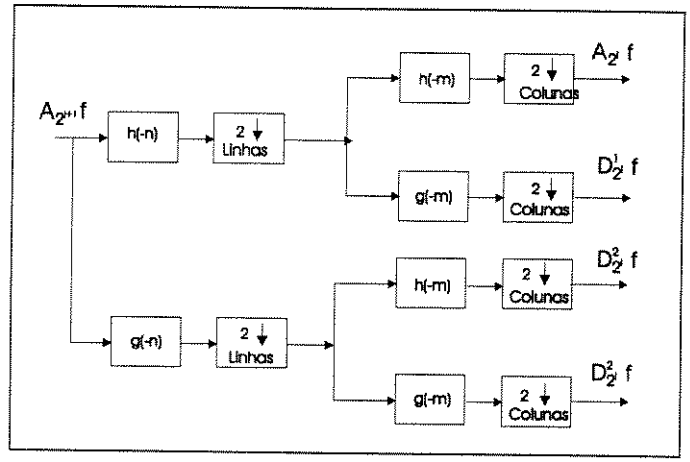



Figura 2.13: Diagrama de blocos do algoritmo de decomposição de uma imagem.

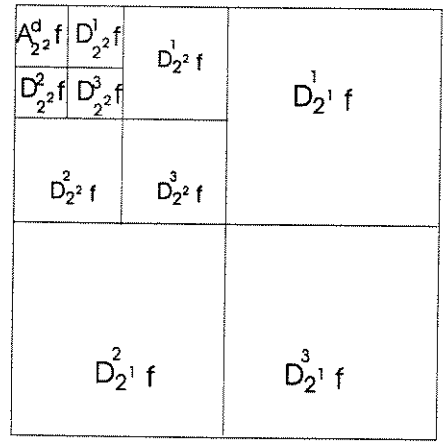



Figura 2.14: Imagens resultantes de uma decomposição através da transformada wavelet.

### Reconstrução do Sinal

A imagem decomposta $\left(A^d_{2^{-J}} f, (D^1_{2^j} f)_{-J\leq j\leq -1}, (D^2_{2^j} f)_{-J\leq j\leq -1}, (D^3_{2^j} f)_{-J\leq j\leq -1}\right)$ pode ser reconstruída utilizando a transformada wavelet 2-D. Como $\mathbf{W}^2_{2^j}$ é o complemento ortogonal de $\mathbf{V}^2_{2^j}$ em $\mathbf{V}^2_{2^{j+1}}$, $\left(\phi_{j;k_x,k_y}(x,y), \left(\psi^i_{j;k_x,k_y}(x,y)\right)_{1\leq i\leq 3}\right)_{n\in\mathbb{Z}}$ é a base ortonormal de $\mathbf{V}^2_{2^{j+1}}$ e a função $\phi_{j+1;k_x,k_y}(x,y)$ pode ser decomposta da seguinte forma:

$$\begin{aligned}\phi_{j+1;k_x,k_y}(x,y) &= \sum_{k_x=-\infty}^{\infty}\sum_{k_x=-\infty}^{\infty}\left\langle \phi_{j+1;n_x,n_y}(u,v), \phi_{j;k_x,k_y}(u,v)\right\rangle \cdot \phi_{j;k_x,k_y}(x,y) \\ &+\sum_{i=1}^{3}\sum_{k_x=-\infty}^{\infty}\sum_{k_x=-\infty}^{\infty}\left\langle \phi_{j+1;n_x,n_y}(u,v), \psi^i_{j;k_x,k_y}(u,v)\right\rangle \cdot \psi^i_{j;k_x,k_y}(x,y)\end{aligned} \tag{2.176}$$

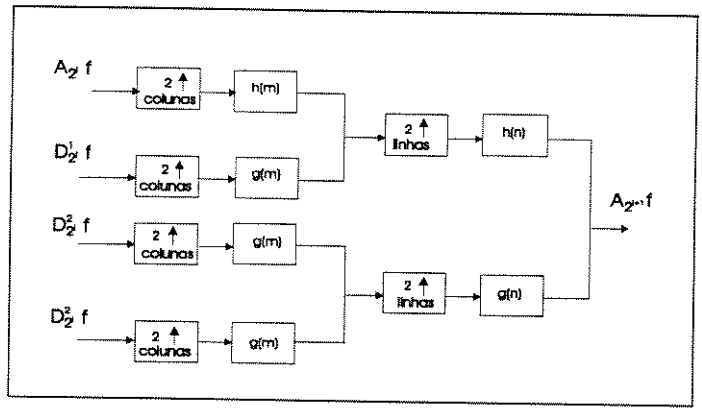



Figura 2.15: Diagrama de blocos do algoritmo de reconstrução de uma imagem.

Fazendo algumas substituições em (2.176) e convoluindo o resultado com $f(x,y)$ obtemos:

$$\begin{aligned}\left\langle f(x,y), \phi_{j+1;k_x,k_y}(x,y)\right\rangle &= \sum_{k_x=-\infty}^{\infty} h(2n_y-k_y) \sum_{k_x=-\infty}^{\infty} h(2n_x-k_x)\cdot\left\langle f(x,y), \phi_{j;k_x,k_y}(x,y)\right\rangle \\ &+ \sum_{k_x=-\infty}^{\infty} g(2n_y-k_y) \sum_{k_x=-\infty}^{\infty} h(2n_x-k_x)\cdot\left\langle f(x,y), \psi^1_{j;k_x,k_y}(x,y)\right\rangle \\ &+ \sum_{k_x=-\infty}^{\infty} h(2n_y-k_y) \sum_{k_x=-\infty}^{\infty} g(2n_x-k_x)\cdot\left\langle f(x,y), \psi^2_{j;k_x,k_y}(x,y)\right\rangle \\ &+ \sum_{k_x=-\infty}^{\infty} g(2n_y-k_y) \sum_{k_x=-\infty}^{\infty} g(2n_x-k_x)\cdot\left\langle f(x,y), \psi^3_{j;k_x,k_y}(x,y)\right\rangle\end{aligned} \tag{2.177}$$

O diagrama de blocos deste algoritmo é ilustrado na Figura 2.15. Comparando (2.177) com (2.153), (2.164), (2.165) e (2.166), conclui-se que a cada passo, a imagem $A^d_{2^{-j+1}} f$ é reconstruída a partir de filtragens unidimensionais das imagens $A^d_{2^j} f$, $D^1_{2^j} f$, $D^2_{2^j} f$ e $D^3_{2^j} f$. Primeiramente, a cada coluna das imagens $A^d_{2^j} f$, $D^1_{2^j} f$, $D^2_{2^j} f$ e $D^3_{2^j} f$, é interpolada com uma coluna de zeros e realizamos a convolução das linhas resultantes com um filtro unidimensional. A seguir, cada linha da imagem é interpolada com uma linha de zeros e realizamos a convolução das colunas resultantes com um outro filtro unidimensional. Os filtros unidimensionais utilizados neste processo de reconstrução são os filtros conjugados em quadratura $H$ e $G$, com respostas impulsivas $h(n)$ e $g(n)$. A imagem $A^d_1 f$ é reconstruída a partir da sua transformada wavelet, repetindo-se este processo para $-1 \leq j \leq -J$.

## 2.8.2 Wavelets e o Processamento de Imagens

A natureza da imagem e o mecanismo de visão humana exigem que a transformada utilizada no processamento de imagens apresente certas características para que se consiga um bom desempenho sem que haja distorções visíveis. A transformada utilizada deve, entre outras

coisas, aceitar a não-estacionariedade da imagem. Conforme apresentado nas primeiras seções deste capítulo, a TW satisfaz a esta exigência.

As aplicações em processamento de imagens exigem também que a transformada e, portanto, os filtros responsáveis pela sua implementação, apresentem certas características, algumas das quais são descritas a seguir:

1. **Suporte Compacto:** Se a função escala e a wavelet têm suporte compacto, isto é, são funções não-nulas em um intervalo finito, as respostas impulsivas dos filtros correspondentes têm comprimento finito (filtros FIR). Assim, o esforço computacional da implementação é inferior àquele exigido por funções que não apresentam suporte compacto.
2. **Coeficientes racionais:** Quando utilizamos filtros com coeficientes racionais ou, ainda melhor, racionais diádicos, as operações de ponto flutuante são evitadas e um número menor de erros são introduzidos.
3. **Filtros de fase linear:** Os filtros FIR utilizados devem ter fase linear (wavelets simétricas), evitando, assim, problemas nas bordas das imagens e simplificando a implementação.
4. **Suavidade ou regularidade:** A compressão é obtida anulando coeficientes da transformada que apresentem amplitude $d_{j,l}$ inferior a um determinado valor. Numa compressão de imagens, se a wavelet não é suave, o erro devido à sua eliminação é mais facilmente detectado (visualmente). Além disso um maior grau de suavidade corresponde a uma melhor localização em freqüência dos filtros.
5. **Comprimento dos filtros:** Obviamente, sob o aspecto da implementação, é preferível que os filtros tenham comprimentos menores. Entretanto, há um compromisso entre o comprimento dos filtros e a suavidade e o número de momento desvanecentes da wavelet e da função escala. Quanto maior a suavidade e o número de momentos desvanecentes da wavelet, maior o comprimento dos filtros correspondentes.

Resumindo, como as imagens são, na sua maioria, suaves, parece apropriado utilizar wavelets de reconstrução exata ortogonais, com uma boa regularidade. Além disso, devemos utilizar filtros FIR de comprimento curto e fase linear. Infelizmente, nem todas estas condições podem ser satisfeitas simultaneamente, pois não existem filtros FIR de fase linear ortonormais de reconstrução exata.

As wavelets que atendem à maioria destas exigências simultaneamente são as wavelets biortogonais, as quais têm apresentado um desempenho bastante interessante na compressão de imagens [Antonini]. Uma grande variedade de bases wavelets biortogonais podem ser construídas a partir de filtros suficientemente longos. Neste trabalho vamos utilizar as wavelets apresentadas na Seção 2.7.3, que apresentam um desempenho muito bom nesta aplicação. Além disso, estas wavelets introduzem uma quantidade de distorções muito inferior à introduzida pelas ortogonais, não apresentando os problemas de bordas introduzidos pelas ortogonais. Com o objetivo de fazer uma comparação visual entre o desempenho das wavelets ortogonais e biortogonais, apresentaremos na seção seguinte alguns exemplos da decomposição de uma imagem utilizando estes dois tipos de wavelets.

### 2.8.3 Experiências

Nesta seção apresentaremos alguns exemplos da utilização transformada wavelet bidimensional. No Apêndice B podem ser encontradas todas as imagens resultantes destas experiências. Com o intuito de realizar uma primeira comparação entre os desempenhos das wavelets ortogonais e biortogonais, utilizamos nestes exemplos a wavelet ortogonal Daubechies 3 e a variante da wavelet biortogonal spline ($l = k = 4$). Os coeficientes dos filtros passa-baixas destas wavelets foram fornecidos nas seções anteriores.

Como o nosso interesse é medir as perdas introduzidas pela TW, não foi realizado nenhum tipo de processamento entre os estágios de decomposição e síntese. As perdas na qualidade das imagens reconstruídas se devem unicamente aos processos de síntese e decomposição da TW. Nestas experiências, avaliamos a qualidade visual das imagens reconstruídas para 1 e 5 estágios de decomposição/síntese.

Para um estágio de decomposição/síntese ($J = 1$ em (2.175)), a imagem original $A_1^d f$ é decomposta em 4 subimagens através da TW:

$$\left(A_{2^{-1}}^d f, D_{2^{-1}}^1 f, D_{2^{-1}}^2 f, D_{2^{-1}}^3 f\right) \tag{2.178}$$

Se a imagem original tem $N$ píxeis, cada imagem $\left(A_{2^{-1}}^d f,\ D_{2^{-1}}^1 f,\ D_{2^{-1}}^2 f \text{ ou } D_{2^{-1}}^3 f\right)$ possui $2^{-1}N$ píxeis. O número total de píxeis desta nova representação é igual ao número de píxeis da imagem original. Portanto, cada subimagem desta representação wavelet tem $\frac{1}{4}$ do tamanho da imagem original. Cada uma destas é uma sub-banda de freqüências da imagem original.

Na Figura 2.16-(a), apresentamos a imagem "quadrado", que utilizaremos como imagem original na primeira parte da nossa experiência. Na Figura 2.16-(b) apresentamos a sua transformada wavelet. Observe que a sub-banda do canto superior esquerdo da transformada é uma aproximação da imagem original, denominada sub-banda de baixas freqüências. As outras sub-bandas são as imagens de detalhes. No canto superior direito temos a sub-banda de detalhes horizontais, no canto inferior esquerdo a sub-banda de detalhes verticais e no canto inferior direito a sub-banda de detalhes diagonais ou sub-banda de altas freqüências.

Na Figura 2.17-(a), apresentamos a imagem reconstruída utilizando a wavelet ortogonal Daubechies-3. Vemos que esta imagem apresenta problemas nas bordas do quadrado interno e uma coloração um pouco diferente da original. Na Figura 2.17-(b), apresentamos a mesma imagem reconstruída utilizando a wavelet biortogonal. A Figura 2.17-(b) é aparentemente idêntica à imagem original, não apresentando nenhuma deterioração visível. Esta diferença deve-se ao fato da wavelet biortogonal utilizada ser simétrica, o que reduz os problemas de reconstrução das bordas das imagens a cada estágio.

Com o objetivo verificar como se comporta a imagem reconstruída quando aumentamos o número de estágios de decomposição/síntese da transformada wavelet, repetimos a experiência anterior utilizando agora cinco estágios em cascata. Na Figura 2.18-(a) apresentamos a imagem reconstruída utilizando a wavelet ortogonal. Observe que esta imagem apresenta problemas ainda mais graves nas bordas, além de estar muito borrada. A qualidade da imagem reconstruída piorou bastante quando aumentamos o número de níveis de decomposição.

A imagem reconstruída utilizando a wavelet biortogonal é apresentada na Figura 2.18-(b). A diferença da imagem reconstruída 2.18-(b) para a imagem original 2.16-(a) é ainda imperceptível, mesmo para 5 estágios de decomposição/síntese.

Apesar dos resultados obtidos nas experiências 1 e 2 serem bastante favoráveis ao uso das wavelets biortogonais, precisamos ainda examinar o comportamento destas wavelets no processamento de imagens mais suaves, como as encontradas em aplicações práticas. Como um dos grandes problemas das wavelets ortogonais são as bordas, é esperado que o desempenho destas seja um pouco melhor.

Utilizaremos a imagem apresentada na Figura 2.19-(a) como imagem original nesta segunda parte da experiência. A transformada wavelet desta imagem é apresentada na Figura 2.19-(b). Repetiremos, então, os procedimentos utilizados na primeira parte da experiência, utilizando agora a imagem 2.19-(a). As mesmas wavelets são utilizadas e as experiências são repetidas para um e cinco estágios de decomposição/síntese.

A imagem reconstruída após um estágio de decomposição/síntese utilizando a wavelet ortogonal, é apresentada na Figura 2.20-(a). Observa-se que esta imagem, apesar de ser bastante semelhante à original, apresenta algumas falhas nas bordas. Na Figura 2.20-(b) apresentamos a imagem reconstruída utilizando a wavelet biortogonal. Esta, é novamente aparentemente idêntica à original, não apresentando nenhuma deterioração visível.

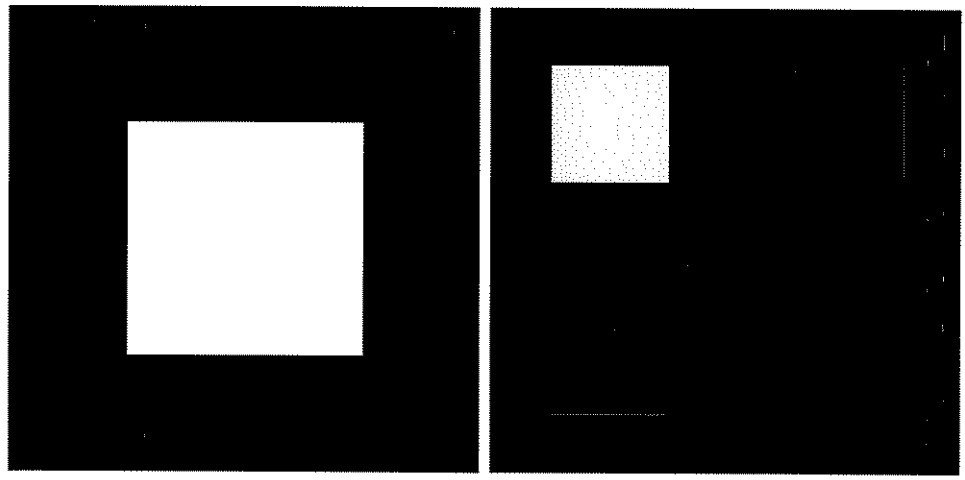
Na Figura 2.21-(a) apresentamos a imagem reconstruída utilizando a wavelet ortogonal, após cinco estágios de decomposição/síntese em cascata. Esta imagem apresenta graves problemas e está completamente borrada. Na Figura 2.21-(b) apresentamos a imagem reconstruída utilizando a wavelet biortogonal. Esta, por sua vez, é aparentemente idêntica à imagem original 2.19-(a), mesmo após 5 estágios de decomposição/síntese.

### Conclusões

O principal objetivo destes exemplos foi realizar uma comparação simples entre o desempenho das wavelets ortogonais e biortogonais. A wavelet biortogonal apresentou uma qualidade de reconstrução superior à apresentada pela ortogonal. Quando aumentamos o número de níveis de decomposições utilizadas, esta diferença tornou-se ainda maior.

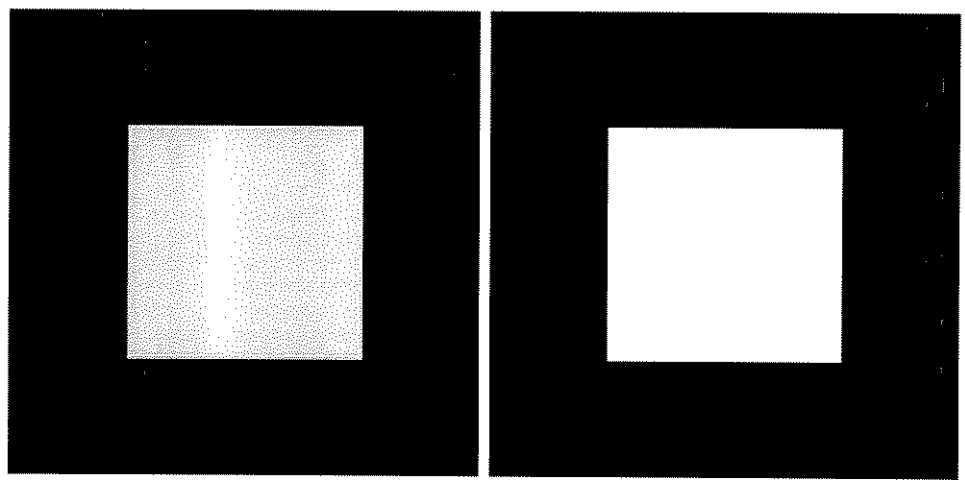
As wavelets Daubechies 3 e a variante da wavelet spline biortogonal não foram as únicas wavelets testadas. Realizamos as mesmas experiências para as wavelets ortogonais Daubechies 2 e Daubechies 4 e as wavelets biortogonais splines e wavelets biortogonais próximas à ortonormais. Os resultados obtidos para a wavelet Daubechies 4 foram bastante semelhantes aos obtidos para a wavelet Daubechies 3. A wavelet Daubechies 2 apresentou uma imagem reconstruída de qualidade um pouco inferior às outras duas wavelets ortogonais. Todas wavelets biortogonais testadas apresentaram imagens reconstruídas de excelente qualidade e desempenho bastante semelhante.

Acreditamos que a decomposição em vários níveis é bastante útil na compressão de imagens. Daí a importância da qualidade da imagem, mesmo após vários níveis de decomposição. As imagens reconstruídas utilizando wavelets biortogonais não apresentaram nenhum problema grave. Podemos dizer que toda elas são visualmente muito semelhantes às imagens originais correspondentes. Concluímos, então, que no contexto das experiências aqui realizadas, a melhor opção para o processamento de imagens, e especificamente para a compressão, é de fato a utilização das wavelets biortogonais.



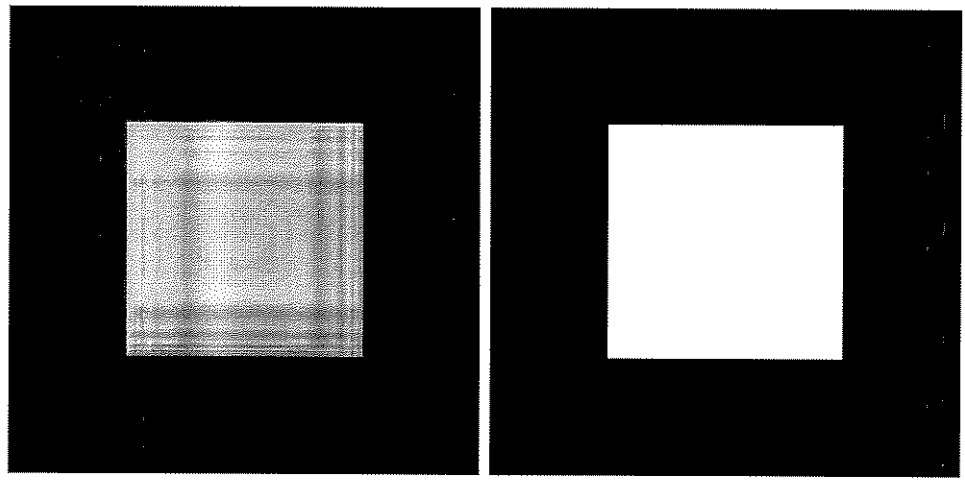
(a) (b)

Figura 2.16: (a) Imagem utilizada nas experiências 1 e 2 e (b) sua TW.



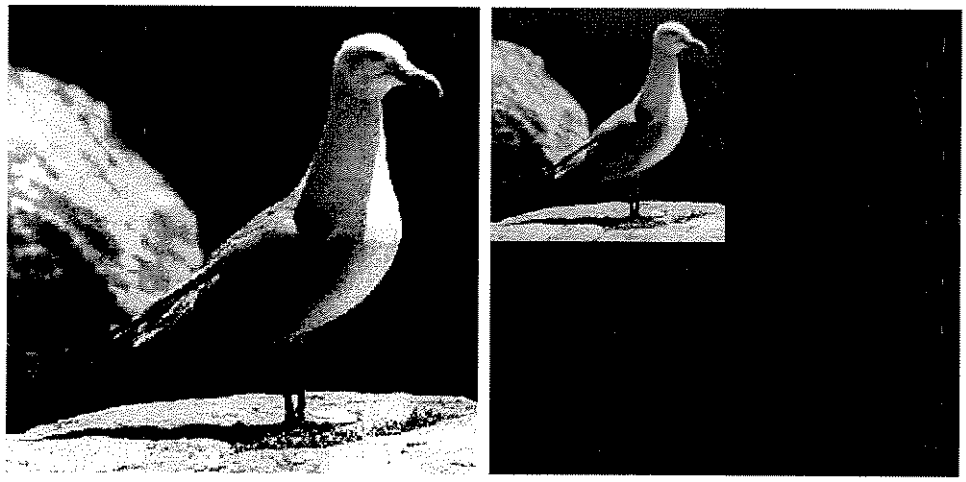
(a) (b)

Figura 2.17: Imagens reconstruídas utilizando a wavelet (a) ortogonal e a (b) biortogonal, para um estágio de decomposição/síntese.



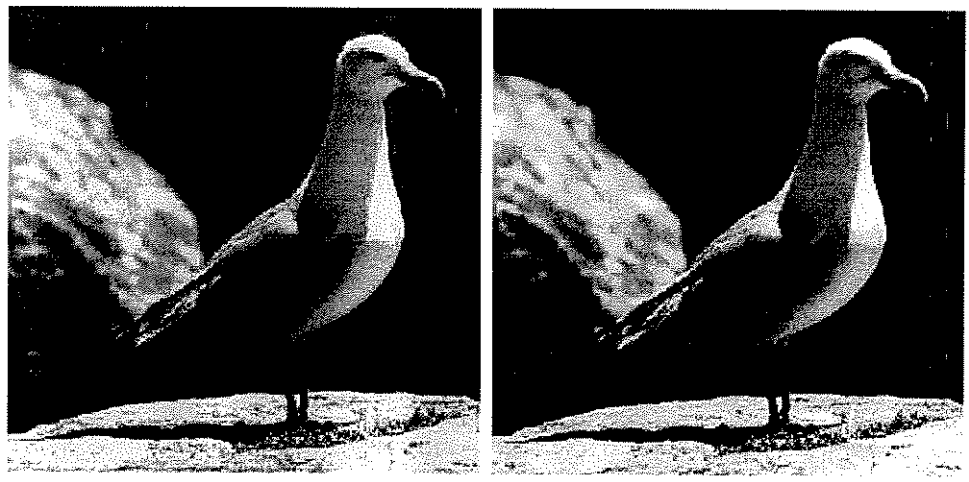
(a) (b)

Figura 2.18: Imagens reconstruídas utilizando a wavelet (a) ortogonal e a (b) biortogonal, para um estágio de decomposição/síntese.



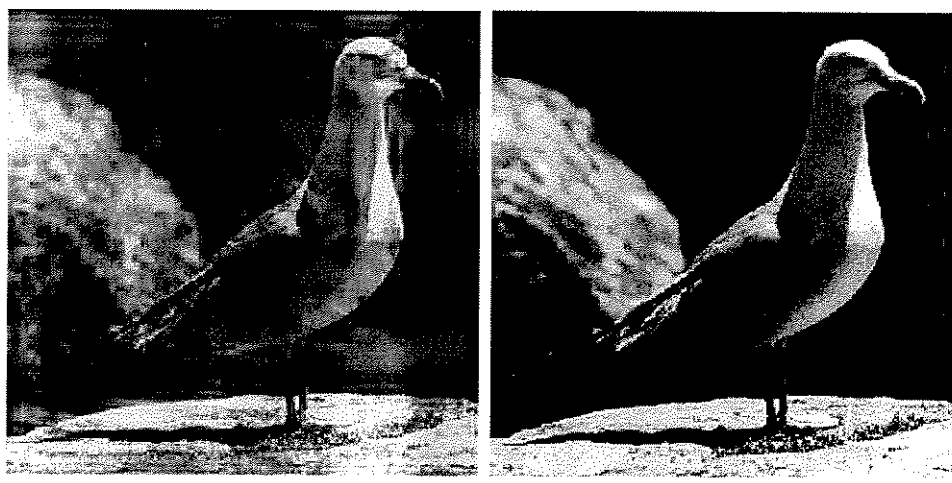
(a) (b)

Figura 2.19: (a) Imagem utilizada nas experiências 3 e 4 e (b) sua TW.



(a) (b)

Figura 2.20: Imagens reconstruídas para a wavelet (a) ortogonal e (b) biortogonal, para um estágio de decomposição/síntese.



(a) (b)

Figura 2.21: Imagens reconstruídas para a wavelet (a) ortogonal e (b) biortogonal, para cinco estágios de decomposição/síntese.

# Capítulo 3

# Compressão de imagens

Apesar das vantagens, o grande desafio da utilização de imagens digitais é a grande quantidade de bits necessários para a sua representação. Felizmente, elas possuem, na sua representação original, uma grande quantidade de informação redundante, o que possibilita a compressão dos dados. Técnicas específicas são necessárias e o ramo do processamento de imagens que se dedica a estudar estas técnicas é a compressão de imagens. Neste capítulo apresentaremos um resumo das principais técnicas de compressão de imagens existentes.

## 3.1 Introdução

Compressão de imagens é o processo de redução da quantidade de bits necessária para representar uma imagem. A compressão, em geral, é possível porque o número de bits realmente necessários para representar a informação contida em uma imagem pode ser reduzido, devido à redundância natural da imagem. Em geral, podemos identificar os seguintes tipos de redundância:

- Redundância espacial ⇒ Existe uma grande correlação entre valores de píxeis vizinhos. Esta correlação é denominada redundância espacial.
- Redundância espectral ⇒ Entre diferentes planos de cores ou entre as diferentes bandas espectrais, também existe uma correlação que denominamos de correlação espectral. Este tipo de redundância é mais facilmente identificável em imagens coloridas.
- Redundância temporal ⇒A correlação entre os diferentes quadros de uma seqüência de imagens é denominada redundância temporal. Este tipo de redundância é facilmente percebida, pois geralmente as imagens de uma seqüência apresentam uma grande quantidade de informação que permanece invariável de um quadro para outro. Os algoritmos de compressão de seqüências de imagens, ou seja, vídeo, tiram proveito deste tipo de redundância para obter uma compressão mais eficiente.
- Redundância de codificação ⇒ Uma codificação é considerada ótima se o número de símbolos resultante for mínimo. Este limitante, estabelecido por Shannon [Abramson], fornece o número mínimo de bits por símbolo necessários para codificar uma fonte. Uma

imagem codificada apresenta uma redundância de codificação se o número de símbolos utilizados para codificá-la for maior que este limitante.

- Redundância psico-visual ⇒ O olho humano não reage com igual intensidade a todas as informações visuais contidas em uma imagem. Algumas informações têm menor importância relativa, ou seja, são psico-visualmente redundantes, podendo ser eliminadas sem que nenhuma perda na qualidade da imagem possa ser percebida.

A redundância não é um conceito abstrato, mas uma definição matemática quantitativa. Sejam $n_1$ e $n_2$ os números de unidades de informação contidas em dois conjuntos de dados, que representam a mesma informação. Para aplicações em compressão de imagens, o primeiro conjunto ($n_1$) representa a imagem original e o segundo ($n_2$) representa a imagem comprimida. A redundância de dados relativa ao primeiro conjunto é definida como:

$$R_D = 1 - \frac{1}{C_R} \tag{3.1}$$

onde $C_R$, denominada de taxa de compressão relativa, é definida por:

$$C_R = \frac{n_1}{n_2} \tag{3.2}$$

Para o caso em que $n_1 = n_2$, $C_R = 1$ e $R_D = 0$, indicando que a primeira representação não contém redundância em relação à segunda. Quando $n_2 \ll n_1$, $C_R \to \infty$ e $R_D \to 1$, significando que houve uma compressão significativa. Para $n_2 \gg n_1$, $C_R \to 0$ e $R_D \to -\infty$, indicando que o segundo conjunto de dados contém um número de unidades de informação muito maior que o primeiro conjunto. Houve, portanto, uma expansão de dados, que é uma situação indesejável para a compressão. Para os casos de interesse prático, $C_R$ e $R_D$ estão dentro dos intervalos $(1, \infty)$ e $(0, 1)$, respectivamente.

### 3.1.1 Padrões de Compressão Existentes

A adoção de padrões não só facilita a utilização das imagens nas mais diversas aplicações, mas também ajuda a reduzir sensivelmente o custo do *hardware* dos sistemas de compressão de imagens. Os padrões relacionados à codificação e transmissão de sinais através de canais de telecomunicações são desenvolvidos com o apoio do setor de padronização em telecomunicações da União Internacional de Telecomunicações ( "International Telecommunications Union"- ITU-T). Este setor é conhecido como CCITT. Atualmente, os esforços para a padronização de esquemas de compressão de imagens têm sido dirigidos para as três seguintes áreas:

- Imagens de dois níveis:

  Um comitê conhecido como JBIG ( "Joint Bilivel Imaging Group") foi formado em 1988 com o objetivo de trabalhar no desenvolvimento de um padrão de compressão e descompressão de imagens de dois níveis. Como já existia um padrão para este tipo de imagens, o objetivo do grupo foi encontrar um algoritmo que satisfizesse o padrão existente para as aplicações já especificadas e estendesse a sua utilidade a outras aplicações.

- Imagens de tons contínuos, estáticas, monocromáticas ou coloridas.

  Um comitê conhecido como JPEG ( "Joint Photographic Experts Group" ) foi formado no final do ano de 1986 com o propósito de desenvolver um padrão internacional para a compressão e descompressão de imagens de tons contínuos, estáticas, monocromáticas ou coloridas. O objetivo deste comitê foi criar um padrão para aplicações tão diversas como foto-videotexto, publicações em computadores pessoais, artes gráficas, fac-símiles coloridos, sistemas médicos, foto-jornalismo, etc.

- Imagens seqüenciais de tons contínuos

  Desde 1988, um grupo de padronização conhecido como MPEG ("Moving Picture Experts Group") vem trabalhando para desenvolver um padrão para armazenamento e recuperação de imagens em movimento e sons, usando meios digitais de armazenamento com uma taxa combinada de 1,0 -1,5 Mbits/s. O padrão MPEG é uma técnica de propósitos gerais para aplicações tão diversas como publicações eletrônicas, guia de viagens, vídeo-texto, educação, jogos, diversões, vídeo-mail e treinamento através de teleseminários.

## 3.2 Modelos de Compressão de Imagens

Como mostra a Figura 3.1, um sistema de compressão pode ser dividido em dois blocos básicos: um codificador e um decodificador. A imagem de entrada, $f(x,y)$, é alimentada no codificador, que cria um conjunto de símbolos. No caso de uma imagem, os símbolos representam os níveis de amplitude de cada píxel da imagem. Após a transmissão através do canal, a representação codificada é alimentada no decodificador, onde a imagem reconstruída, $\widehat{f}(x,y)$, é gerada. Se $\widehat{f}(x,y) = f(x,y)$, o sistema é livre de erros e a compressão é *sem-perdas*. Caso contrário, a compressão é *com-perdas* e algum nível de distorção está presente na imagem reconstruída.

A compressão *sem-perdas* apresenta a vantagem de não introduzir erros. Por outro lado, pouca compressão pode ser obtida desta forma.

A imagem reconstruída na compressão *com-perdas* contém degradações em relação à original. Um nível de compressão muito maior pode ser obtido, ao custo de uma maior degradação. É importante frisar que essas degradações podem ou não ser visíveis, o que torna este tipo de compressão bastante atrativa.

Tanto o codificador, quanto o decodificador são formados por dois sub-blocos relativamente independentes, como pode ser visto na Figura 3.1. O codificador é formado por um codificador de fonte e um codificador de canal. Da mesma forma, o decodificador é formado por um decodificador e um decodificador de fonte. Se o canal não apresenta ruído, o codificador e decodificador de canal são desnecessários.

### 3.2.1 O Codificador e Decodificador de Fonte

O codificador de fonte é responsável pela redução ou eliminação de redundâncias da imagem de entrada. A codificação utilizada irá depender da aplicação. É importante frisar que a imagem de entrada é uma imagem digital, ou seja, com um número de níveis de amplitude finito.

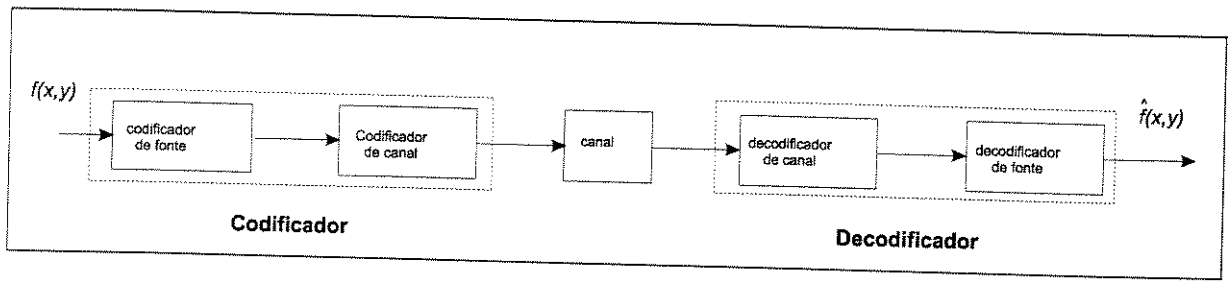



Figura 3.1: Modelo de um sistema de compressão genérico.

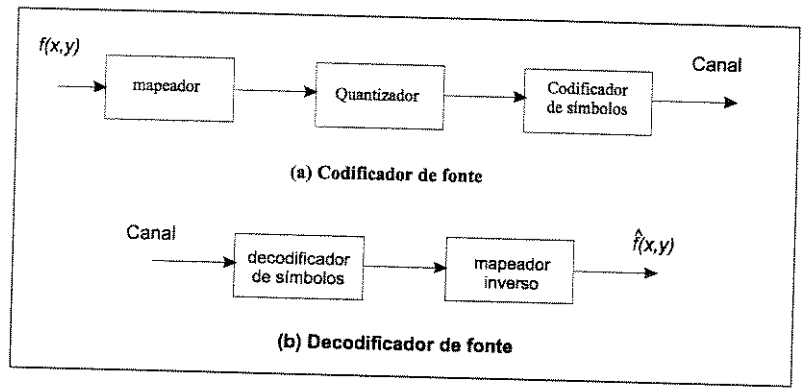



Figura 3.2: Esquema de blocos de um codificador/decodificador de fonte.

Normalmente, o codificador e o decodificador podem ser construídos de acordo com o modelo da Figura 3.2. Este modelo possui um valor didático, pois nem sempre será possível distinguir tão claramente em um codificador/ decodificador de fonte todos os estágios deste diagrama.

O primeiro estágio do codificador de fonte, o mapeador, modifica o formato dos dados de entrada, de maneira a reduzir as redundâncias entre píxeis da imagem. Esta operação geralmente é reversível e não necessariamente reduz a quantidade de dados necessários para representar a imagem.

O segundo estágio, o quantizador, reduz a redundância psico-visual da imagem. Parte da informação contida na imagem, que não representa uma perda significativa na qualidade, é eliminada através da redução da quantidade de níveis de amplitude da imagem. A quantização é uma operação irreversível e não deve ser utilizada quando se deseja uma compressão *sem-perdas*.

O terceiro estágio, o codificador de símbolos, cria um código para representar a saída do quantizador e mapeia esta saída de acordo com este código. Este codificador tem o objetivo de reduzir a redundância de codificação. Na maioria dos casos é utilizado um código de comprimento variável, que atribui palavras-código de comprimento mais curto a símbolos mais freqüentes e vice-versa. Esta operação é reversível e tem o objetivo de reduzir ou eliminar as redundâncias de codificação.

O diagrama de blocos do decodificador de fonte pode ser visto na Figura 3.2-(b). Observe que o decodificador possui apenas dois estágios: o decodificador de símbolos e o mapeador

inverso, que realizam, respectivamente, as operações inversas do codificador de símbolos e do mapeador. Como a quantização é uma operação que resulta em perda de informação, não há um estágio que realize a sua operação inversa.

### 3.2.2 O Codificador e Decodificador de Canal

Se o canal é ruidoso, o codificador e o decodificador de canal têm um papel importante no esquema de compressão, pois eles são projetados de maneira a reduzir o impacto de ruído, o que é feito através da inserção de redundância de forma controlada. Como a informação codificada tem pouca redundância, sua sensibilidade ao ruído é grande.

Como exemplo, vamos abordar os aspectos básicos do código de Hamming [Hamming], que é uma das técnicas mais utilizadas. A estratégia deste código consiste em adicionar uma certa quantidade de bits à mensagem de forma a garantir que um número mínimo de bits diferencie as palavras-código válidas. Chamamos este número mínimo de distância entre as palavras-código. Hamming mostrou que se, por exemplo, três bits são adicionados a uma palavra de 4 bits, de forma que a distância mínima seja igual a três, todos os erros de um bit podem ser detectados e corrigidos. A palavra-código de 7 bits $h_1 \ldots h_5 h_6 h_7$ do código de Hamming (7,4), correspondente ao número binário de 4 bits $b_3 b_2 b_1 b_0$, é calculada através das seguintes expressões:

$$\begin{aligned} h_1 &= b_3 \oplus b_2 \oplus b_0 \\ h_2 &= b_3 \oplus b_1 \oplus b_0 \\ h_4 &= b_2 \oplus b_1 \oplus b_0 \\ h_3 &= b_3 \\ h_5 &= b_2 \\ h_6 &= b_1 \\ h_7 &= b_0 \end{aligned} \tag{3.3}$$

onde $\oplus$ indica a operação OU exclusivo. Observe que os bits $h_1$, $h_2$ e $h_4$ são os bits de paridade par associados aos blocos de bits $b_3 b_2 b_0$, $b_3 b_1 b_0$ e $b_2 b_1 b_0$, respectivamente.

Na decodificação, a paridade dos bits codificados é verificada. Um único erro em um dos bits da palavra é indicado por uma "palavra de paridade " $(c_4 c_2 c_1)$ não-nula. Os bits da palavra de paridade são calculados através das seguintes expressões:

$$\begin{aligned} c_1 &= h_1 \oplus h_3 \oplus h_5 \oplus h_7 \\ c_2 &= h_2 \oplus h_3 \oplus h_6 \oplus h_7 \\ c_4 &= h_4 \oplus h_5 \oplus h_6 \oplus h_7 \end{aligned} \tag{3.4}$$

Se um valor não-nulo é obtido em uma das expressões em (3.4), um erro foi encontrado. Um valor não-nulo em $c_i$ corresponde a um erro no $c_4 c_2 c_1$-ésimo bit da palavra codificada. Calcula-se o complemento deste bit e o erro é corrigido. O valor binário decodificado é, então, extraído da palavra-código corrigida. Observe que a mensagem original corresponde aos bits $h_3 h_5 h_6 h_7 = b_3 b_2 b_1 b_0$.

Neste trabalho, não abordaremos as técnicas de codificação de canal. O modelo que iremos considerar resume-se ao codificador e decodificador de fonte.

## 3.3 Técnicas de Compressão Sem-Perdas

Em algumas aplicações, a compressão *sem-perdas* é a única maneira aceitável de compressão de imagens. Nestas, perdas na qualidade da imagem não são permitidas. Uma dessas aplicações é o arquivo de documentos médicos ou de negócios, onde a compressão *com-perdas* é proibida por razões legais. Nesta seção faremos uma introdução às principais estratégias de compressão *sem-perdas*.

O modelo do sistema de compressão *sem-perdas* é composto por apenas dois estágios: o mapeador e o codificador de símbolos. Estudaremos alguns tipos de mapeadores e codificadores de símbolos, que podem ser utilizados tanto para compressões *sem-perdas* como para compressões *com-perdas*. As taxas de compressão obtidas normalmente variam de 2 a 10. A escolha entre uma destas técnicas vai depender do tipo de aplicação.

### 3.3.1 Codificadores de Símbolos

A redução da redundância de codificação é a forma mais simples de se obter uma compressão *sem-perdas*. Se utilizarmos uma codificação que minimize o comprimento médio das palavras-código, será possível obter esta redução. Para isso, é necessário construir um código de comprimento variável, que atribua palavras-código de comprimentos mais curtos a símbolos mais prováveis e vice-versa. Estes códigos serão o alvo da nossa discussão nesta seção.

#### Código de Huffman

O código de comprimento variável mais conhecido é o código de Huffman [Abramson], um código de bloco unicamente decodificável e instantâneo. Para um dado conjunto de símbolos da fonte, este código gera o menor número possível de símbolos do código por símbolo da fonte. A entropia da fonte ($H(z)$) é o limitante do comprimento médio das palavras-código[Abramson]. De acordo com o teorema da codificação sem ruído [Abramson], o código de Huffman é ótimo, desde que os símbolos sejam codificados um de cada vez.

Na Figura 3.3 podemos ver um exemplo do procedimento de geração de um código de Huffman a partir de um conjunto de símbolos de fonte e as suas respectivas probabilidades (segunda coluna da Figura 3.3). A terceira coluna da Figura 3.3 mostra o código final. Depois que o código é criado, a codificação e a decodificação das mensagens são realizadas através de consultas à tabela criada.

#### Códigos de Comprimento Variável Quase-Ótimos

A complexidade do algoritmo do código de Huffman aumenta muito quando o número de símbolos da fonte aumenta. Para codificar uma fonte com $J$ símbolos, um total de $J-2$ reduções e $J-2$ atribuições devem ser realizadas. A construção de um código de Huffman para uma imagem com 256 níveis de cinzas, por exemplo, exige que 254 reduções e 254 atribuições sejam feitas.

Existem, entretanto, algumas alternativas ao código de Huffman. Na Tabela 3.1 apresentamos uma comparação entre vários códigos [Gonzalez]: o código binário, o código de Huffman e quatro outros códigos de comprimento variável, que definiremos a seguir. Na primeira coluna desta tabela são apresentados o conjunto de símbolos da fonte e as suas probabilidades. Nas

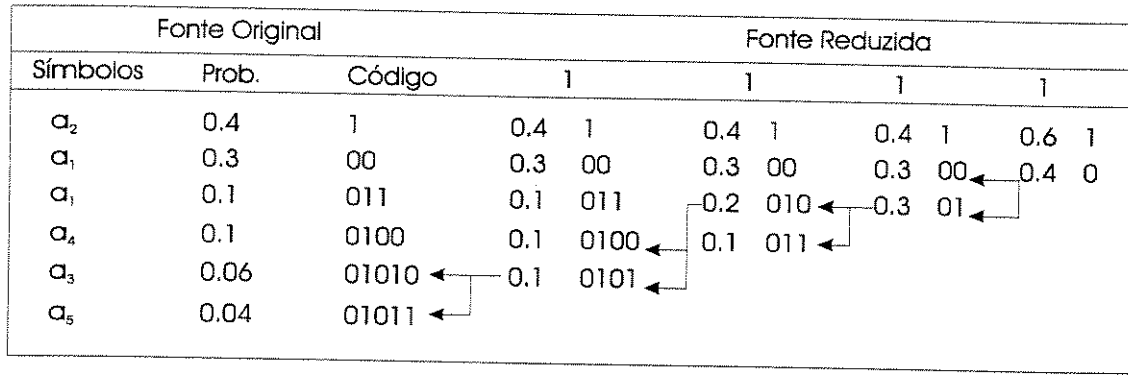



Figura 3.3: Procedimento para codificação de um conjunto de símbolos utlizando o código de Huffman.

demais colunas, são apresentados as palavras-código e os comprimentos médios correspondentes a cada um dos códigos. Observe que o código binário (natural) é o que possui o maior comprimento médio. O código de Huffman (quarta coluna da Tabela 3.1) é o que possui o menor comprimento médio [Abramson], com valor bem próximo à entropia da fonte. Assim como o código de Huffman, os outros códigos da Tabela 3.1 também atribuem palavras-código de comprimentos mais curtos aos símbolos mais prováveis e vice-versa. Entretanto, nenhum destes códigos possui um comprimento médio menor que o código de Huffman. Por outro lado, estes códigos apresentam uma complexidade computacional bem menor que a do código de Huffman e são quase-ótimos, ou seja, possuem comprimentos médios bem próximos à entropia $H(z)$ da fonte.

O código de Huffman truncado (quinta coluna da Tabela 3.1) é uma modificação do código de Huffman. Este código é gerado utilizando-se o código de Huffman para codificar apenas os $\psi$ símbolos mais prováveis da fonte. O número $\psi$ é um inteiro positivo menor que o número total de símbolos da fonte $(J)$. Na Tabela 3.1 $\psi = 12$. Um código-prefixo, seguido por um código de comprimento fixo adequado, é, então, utilizado para codificar todos os outros símbolos da fonte.

O código-prefixo é gerado a partir da décima terceira palavra-código do código de Huffman. Assim, construímos um símbolo fictício, cuja probabilidade é igual à soma das probabilidades dos símbolos $a_{13}$ até $a_{21}$. Este símbolo é definido como o décimo terceiro símbolo durante a codificação dos 12 símbolos mais prováveis. O código gerado para este símbolo será utilizado como código-prefixo na codificação dos 9 símbolos restantes. O código-prefixo, neste caso, é $(10)$. A este é adicionado um número binário de 4 bits que corresponde ao subíndice do símbolo correspondente (no segundo grupo de símbolos) subtraído de 13. Por exemplo, o décimo-sétimo símbolo foi codificado como $(100100)$, onde $(10)$ é o código-prefixo e $(0100)_2 = (4)_{10} = (17 - 13)_{10}$.

O código B é um código quase-ótimo de comprimento variável. Este código se aproxima do código ótimo, quando as probabilidades dos símbolos da fonte são expressas da seguinte forma:

$$P(a_j) = cj^{-\beta} \tag{3.5}$$

| Símbolos da fonte | prob. | Código Binário | Huffman | Huffman truncado | Código $B_2$ | Binário deslocado | Huffman deslocado |
|---|---|---|---|---|---|---|---|
| $a_1$ | 0,2 | 00000 | 10 | 11 | C00 | 000 | 10 |
| $a_2$ | 0,1 | 00001 | 110 | 011 | C01 | 001 | 11 |
| $a_3$ | 0,1 | 00010 | 111 | 0000 | C10 | 010 | 110 |
| $a_4$ | 0,06 | 00011 | 0101 | 0101 | C11 | 011 | 100 |
| $a_5$ | 0,05 | 00100 | 00000 | 00010 | C00C00 | 100 | 101 |
| $a_6$ | 0,05 | 00101 | 00001 | 00011 | C00C01 | 101 | 1110 |
| $a_7$ | 0,05 | 00110 | 00010 | 00100 | C00C10 | 110 | 1111 |
| $a_8$ | 0,04 | 00111 | 00011 | 00101 | C00C11 | 111000 | 0010 |
| $a_9$ | 0,04 | 01000 | 00110 | 00110 | C01C00 | 111001 | 0011 |
| $a_{10}$ | 0,04 | 01001 | 00111 | 00111 | C01C01 | 111010 | 00110 |
| $a_{11}$ | 0,04 | 01010 | 00100 | 01000 | C01C10 | 111011 | 00100 |
| $a_{12}$ | 0,03 | 01011 | 01001 | 01001 | C01C11 | 111100 | 00101 |
| $a_{13}$ | 0,03 | 01100 | 01110 | 100000 | C10C00 | 111101 | 001110 |
| $a_{14}$ | 0,03 | 01101 | 01111 | 100001 | C10C01 | 111110 | 001111 |
| $a_{15}$ | 0,03 | 01110 | 01100 | 100010 | C10C10 | 111111000 | 000010 |
| $a_{16}$ | 0,02 | 01111 | 010000 | 100011 | C10C11 | 111111001 | 000011 |
| $a_{17}$ | 0,02 | 10000 | 010001 | 100100 | C11C00 | 111111010 | 0000110 |
| $a_{18}$ | 0,02 | 10001 | 001010 | 100101 | C11C01 | 111111011 | 0000100 |
| $a_{19}$ | 0,02 | 10010 | 001011 | 100110 | C11C10 | 111111100 | 0000101 |
| $a_{20}$ | 0,02 | 10011 | 011010 | 100111 | C11C11 | 111111101 | 00001110 |
| $a_{21}$ | 0,01 | 10100 | 011011 | 101000 | C00C00C00 | 111111110 | 00001111 |
| comprimento medio | | 5,00 | 4,05 | 4,24 | 4,65 | 4,59 | 4,13 |
| $H(Z)$ | 3,98 | | | | | | |

Tabela 3.1: Códigos de comprimento variável.

onde $\beta$ é uma constante positiva e $c$ é uma constante de normalização ($c = 1/\sum_{j=0}^{J} j^{-\beta}$). A distribuição de comprimentos em uma representação binária de um texto escrito, por exemplo, é aproximadamente exponencial. Cada palavra-código é construída com bits de continuação, indicados por $C$, e bits de informação, que são os bits binários naturais. O único propósito dos bits de continuação é separar os bits de informação, assumindo valores alternados, 0 ou 1, de acordo com a posição da palavra código na seqüência. O código apresentado na sexta coluna da Tabela 3.1 é denominado de código $B_2$, porque a cada dois bits de informação ocorre um bit de continuação. A seqüência de códigos $B_2$, correspondente a seqüência de símbolos da fonte $a_{11}a_2a_7$, por exemplo, é igual a (001 010 101 000 010) ou a (101 110 001 100 110), dependendo do valor do primeiro bit de continuação.

Os dois códigos restantes da Tabela 3.1 são denominados códigos de deslocamento. Um código de deslocamento é gerado da seguinte forma:

1. Arranja-se os símbolos da fonte de forma que suas probabilidades sejam monotonicamente crescentes.

2. Divide-se o número total de símbolos em blocos de símbolos de tamanho igual.

3. Codifica-se os elementos individuais dentro de blocos idênticos.

4. Adiciona-se símbolos de deslocamento para cima e/ou para baixo, de forma que se possa identificar cada bloco. Cada vez que um símbolo de deslocamento é reconhecido no decodificador, o bloco é movido para cima ou para baixo em relação a um bloco de referência pré-estabelecido.

Para se gerar um código de deslocamento binário de 3 bits (sétima coluna da Tabela 3.1), ordenamos os 21 símbolos da fonte de acordo com as suas probabilidades e, em seguida, dividimos conjunto de símbolos em 3 blocos de 7 símbolos cada. Os símbolos do primeiro bloco ($a_1$até $a_7$), considerado como bloco de referência, são, então,codificados utilizando-se códigos binários com valores entre (000) e (110) . O oitavo código binário (111) , que não é incluído no bloco de referência, é utilizado como controle de deslocamento para cima, de maneira que os blocos restantes possam ser identificados. Os símbolos dos dois blocos restantes são codificados por um ou dois símbolos de deslocamento para cima, seguidos do código binário de mesmo índice no bloco de referência. Por exemplo, o símbolo da fonte $a_{19}$ é codificado como (111 111 100).

O código de deslocamento de Huffman (oitava coluna da Figura 3.1) é gerado de uma maneira semelhante. A principal diferença está na atribuição das probabilidades aos símbolos de deslocamentos. Normalmente esta atribuição é realizada somando-se todas as probabilidades dos símbolos da fonte que não pertençam ao bloco de referência, ou seja, o mesmo conceito utilizado para definir o código-prefixo no código de Huffman truncado. Neste caso, a soma é realizada considerando-se todos os símbolos de $a_8$ até $a_{21}$. O resultado é igual a 0,39. Ao símbolo de deslocamento (um dos símbolos mais prováveis) é atribuído uma das palavras do código de Huffman de menor comprimento.

### Codificação Aritmética

A codificação aritmética [Langdon84] [Rissanen] é uma técnica de compressão *sem-perdas* que associa uma única palavra-código a cada conjunto de dados da fonte. Cada palavra-código válida corresponde a um subintervalo dentro do intervalo semi-aberto $[0, 1)$. Atribuindo uma precisão suficiente a cada palavra-código, pode-se distinguir os subintervalos e, assim, decodificar corretamente os dados. À medida que o número de símbolos na mensagem aumenta, o intervalo utilizado para representá-la diminui e o número de unidades de informação utilizado para representar o intervalo aumenta. Cada símbolo adicional na mensagem reduz o tamanho do intervalo proporcionalmente ao valor da sua probabilidade. Como este código não exige que os símbolos sejam codificados um de cada vez, o limitante estabelecido pelo teorema da codificação sem ruído pode, pelo menos teoricamente, ser atingido.

A Figura 3.4 ilustra o processo básico da codificação aritmética. Neste exemplo, uma mensagem de cinco símbolos ( $a_1$ $a_2$ $a_3$ $a_3$ $a_4$ ) de uma fonte de quatro símbolos é codificada. A probabilidade dos símbolos desta fonte é apresentada na Tabela 3.2. Para codificar esta seqüência seguiremos os seguintes passos:

1. No início do processo, consideramos que a mensagem corresponde ao intervalo $[0, 1)$. Este intervalo é dividido em vários subintervalos, um para cada símbolo do nosso alfabeto. O

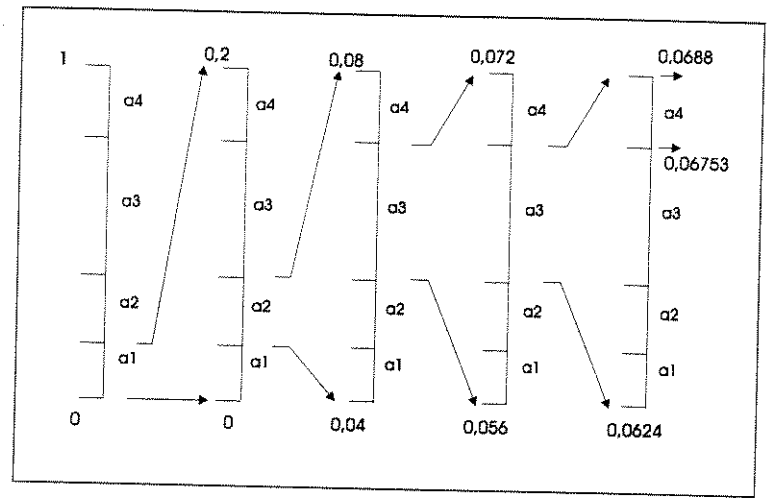



Figura 3.4: Procedimento para codificação aritmética.

limite inferior de cada subintervalo é igual à probabilidade acumulada, definida como a soma das probabilidades dos símbolos considerados até o momento. O limite superior é igual a soma da probabilidade do símbolo com a probabilidade acumulada. O símbolo $a_1$, por exemplo, é associado ao subintervalo [0;0,2), $a_2$ é associado ao subintervalo [0,2;0,4) e assim por diante. Os intervalos de todos os símbolos podem ser vistos na Tabela 3.2.

2. Quando o primeiro símbolo da nossa mensagem, $a_1$, aparece, selecionamos o subintervalo correspondente e o transformamos no intervalo atual. Os intervalos dos símbolos do alfabeto são, então, reescalonados. Intuitivamente, igualamos o tamanho deste subintervalo ao tamanho do subintervalo original. Sejam $\text{Ant}_{\text{inf}}$ e $\text{Ant}_{\text{sup}}$ os limites inferior e superior do intervalo anterior. Neste exemplo, $\text{Ant}_{\text{inf}} = 0$ e $\text{Ant}_{\text{sup}} = 1$. Seja Int = $\text{Ant}_{\text{sup}}$-$\text{Ant}_{\text{inf}}$, o tamanho do subintervalo anterior. Após $a_1$, o limite inferior do novo intervalo é igual a $\text{Ant}_{\text{inf}}$ + Int×$\text{Sub}_{\text{inf}}$, onde $\text{Sub}_{\text{inf}}$ é o limite inferior do subintervalo do símbolo. Da mesma forma, o limite superior do novo intervalo é igual a $\text{Ant}_{\text{inf}}$ +Int×$\text{Sub}_{\text{sup}}$, onde $\text{Sub}_{\text{sup}}$ é o limite superior do subintervalo do símbolo. No nosso exemplo, o novo intervalo corresponde a $[0; 0, 2)$.

3. Após a entrada do segundo símbolo, $a_2$, os limites inferior e superior do novo intervalo são recalculados e obtemos $[0, 04; 0, 08]$. Continuamos fazendo o mesmo para todas as entradas restantes. Para o terceiro símbolo $(a_3)$, obtemos o subintervalo $[0, 056; 0, 072]$. Para o quarto $(a_3)$, $[0, 0624; 0, 0688]$. Após o quinto e último símbolo $(a_4)$, o subintervalo final é calculado. O resultado é o intervalo $[0, 06753; 0, 0688]$.

4. Não há necessidade de transmitir os dois limites do intervalo final. Qualquer número dentro do intervalo $[0, 06753\ 0, 0688]$ pode ser utilizado para representar a mensagem $(\ a_1\ a_2\ a_3\ a_3\ a_4\ )$.

Na mensagem aritmeticamente codificada da Figura 3.4, três dígitos decimais são utilizados para representar uma mensagem de cinco símbolos. Obtemos uma taxa de 0,6 dígitos deci-

| símbolo da fonte | probabilidade | subintervalo inicial |
|---|---|---|
| $a_1$ | 0,2 | [0,0;0,2) |
| $a_2$ | 0,2 | [0,2;0,4) |
| $a_3$ | 0,4 | [0,4;0,8) |
| $a_4$ | 0,2 | [0,8;1,0) |

Tabela 3.2: Intervalos associados a um conjunto de símbolos e suas respectivas probabilidades.

mais por símbolo da fonte, valor que está bastante próximo da entropia da fonte (0,58 dígitos decimais/símbolo).

À medida que o comprimento da mensagem aumenta, o código aritmético resultante se aproxima do limitante estabelecido pelo teorema da codificação sem ruído. Na prática, entretanto, dois fatores contribuem para que o desempenho da codificação aritmética esteja um pouco abaixo do limitante:

- A adição de um indicador no fim da mensagem, que é necessário para separar uma mensagem da outra.
- O uso de uma precisão aritmética finita.

A precisão é necessária para que se possa distiguir as palavras-códigos. Uma precisão finita limita o número de palavras. Implementações práticas da codificação aritmética amenizam o problema da precisão finita introduzindo uma estratégia de escalamento e arredondamento [Jones] [Witten] [Langdon81]. A estratégia de escalonamento normaliza cada intervalo para [0,1), antes de sub-dividí-lo, de acordo com as probabilidades dos símbolos. A estratégia de arredondamento garante que os truncamentos de precisão aritmética finita não impeçam que as codificações dos sub-intervalos sejam apresentadas com precisão. Uma implementação da codificação aritmética bastante interessante é apresentada em [Pennebaker].

Os codificadores de comprimento variável não podem se adaptar às mudanças na estatística da fonte. A codificação aritmética é mais eficiente porque permite uma adaptação a estas mudanças. Entretanto, a sua implementação é bem mais complicada que a dos códigos de comprimento variável, o que dificulta a sua utilização.

### 3.3.2 Mapeadores

Tendo examinado os principais métodos de remoção da redundância de codificação, consideraremos agora algumas das várias técnicas de compressão sem-perdas que atacam a redundância entre os píxeis de uma imagem. Os mapeadores mudam o formato da imagem original. A própria codificação binária já é um mapeamento. O objetivo deste estágio, entretanto, é representar a imagem em um formato mais conveniente. A codificação será, então, mais eficiente e maiores taxas de compressão poderão ser obtidas. A escolha do mapeador adequado vai depender das características da imagem e do tipo de codificador de símbolos utilizados.

### Decomposição em Planos de Bits

O princípio desta técnica é baseada no conceito de decomposição de imagens (com vários níveis de cinza) em uma série de imagens binárias. Cada imagem binária pode, então, ser comprimida através de um dos métodos de compressão para imagens binárias já estudados. Os níveis de cinza de uma imagem de m bits podem ser representados na forma de um polinômio na base 2:

$$a_{m-1}2^{m-1} + a_{m-2}2^{m-2} + ... + a_1 2^1 + a_0 2^0 \tag{3.6}$$

Baseado nesta propriedade, um método simples de decomposição da imagem em uma coleção de imagens binárias consiste em separar os $m$ bits da imagem original em $m$ imagens, ou planos de bits, onde cada elemento tem 1 bit. O plano de bits de ordem zero é gerado agrupando-se os coeficientes $a_0$, ou bits menos significativos de cada píxel da imagem, enquanto que o plano de bits de ordem $(m-1)$ agrupa coeficientes, ou os bits mais significativos $a_{m-1}$. Em geral, os planos de bits são numerados de 0 a $m-1$ e construídos igualando-se o valor dos seus píxeis aos dos bits correspondentes na imagem original.

Uma desvantagem que este tipo de decomposição apresenta é que, ao contrário do que seria esperado, níveis sucessivos de amplitude podem implicar em grandes diferenças entre os planos de bits. Tomemos, por exemplo, um píxel de intensidade 127 (0111111) adjacente a um píxel de intensidade 128 (1000000) na imagem original. Como todos os bits dos códigos binários (0111111) e (1000000) são diferentes em todas as posições, todos os planos de bits terão um píxel de valor zero próximo a um píxel de valor 1, criando uma transição de 0 para 1 nesta posição. Isto significa que uma pequena mudança entre palavras-código vizinhas no alfabeto da fonte podem afetar todos os planos de bits.

Uma maneira simples de contornar este problema consiste em codificar a imagem utilizando o código Gray antes de realizar a decomposição em planos de bits. O código Gray da palavra de m-bits $g_{m-1}...g_2g_1g_0$ ( o polinômio (3.6)) é calculado através das seguintes expressões:

$$\begin{aligned} g_i &= a_i \oplus a_{i+1} \qquad 0 \leq i \leq m-2 \\ g_{m-1} &= a_{m-1}. \end{aligned} \tag{3.7}$$

onde $\oplus$ indica a operação OU exclusivo. Este código possui uma propriedade importantíssima: palavras sucessivas do código diferem em apenas uma posição de bits. Assim, pequenas mudanças nos níveis de cinza terão uma menor probabilidade de afetar todos os planos de bits. Quando, por exemplo, os níveis de cinza 127 e 128 forem adjacentes em uma imagem, apenas o sétimo plano de bits irá conter uma transição de 0 para 1, já que o código Gray dos números 127 e 128 corresponde a (11000000) e (01000000), respectivamente. Após a imagem ter sido decomposta em várias imagens binárias, pode-se codificar estas novas imagens com uma das técnicas de codificação já apresentadas. A seguir apresentaremos outras técnicas de codificação específicas para este tipo de decomposição.

**Codificação Por área Constante** Um método simples, mas eficiente, para comprimir uma imagem binária, ou um plano de bits, é a utilização de um código especial para identificar grandes áreas contíguas de 1's e 0's [Huanga]. Nesta abordagem, denominada codificação por área constante (CAC), a imagem é dividida em blocos de $m \times n$ píxeis, classificados como todos brancos, todos pretos ou de intensidade mista. À categoria de ocorrência mais frequente é

atribuída a palavra-código de 1 bit (0). Às duas outras categorias são atribuídas as palavras-código (10) e (11). Ao representar-se os $m \cdot n$ bits, que normalmente seriam necessários para representar cada bloco, por uma palavra-código de 1 ou 2 bits obtem-se uma compressão. A palavra-código atribuída a regiões mistas é, entretanto, utilizada como um prefixo e deve ser seguida pelo padrão $m \cdot n$ do bloco.

Para comprimir textos de documentos predominantemente brancos pode se utilizar uma abordagem mais simples, que consiste em codificar as áreas sólidas brancas como (0) e todos os outros blocos por (1), seguido do padrão do bloco. Esta abordagem, denominada de "white block skipping" (WBS), tira proveito da estrutura da imagem. Como há uma expectativa que haja poucas áreas sólidas pretas, elas serão classificadas como regiões de intensidade mista, permitindo que uma palavra-código de 1 bit possa ser utilizada para codificar blocos brancos, que são altamente prováveis.

Uma modificação particularmente eficaz para este procedimento consiste em codificar as linhas brancas sólidas como (0) e todas as outras linhas como (1), seguido pela seqüência do código WBS. Uma outra possibilidade é empregar uma abordagem iterativa na qual o plano de bits é decomposto em blocos sucessivamente menores. Para blocos 2-D, uma imagem branca sólida é codificada como (0). Todas as outras imagens são divididas em sub-blocos aos quais são atribuídos um prefixo (1) e o código resultante de uma nova codificação. Por exemplo, se um sub-bloco é todo branco, ele é representado pelo prefixo (1), indicando que ele resulta de uma única iteração, seguido por (0), indicando que ele é um sólido branco. Se o sub-bloco não é branco, o processo de decomposição é repetido até que um tamanho pré-definido de sub-bloco é atingido e codificado como (0), se ele é todo branco, ou (1) seguido pelo padrão de bits do bloco.

**Codificação "Run-Length" Unidimensional** Uma alternativa eficaz para a codificação por área constante consiste na representação de cada linha do plano de bits por uma seqüência de comprimentos que descreve as sucessivas aparições de 1's e 0's. Esta técnica, conhecida como codificação "run-length", foi desenvolvida nos anos 50 [Huangb] e tornou-se, juntamente com a sua extensão para o caso bidimensional, a abordagem padrão na compressão em facsímile (FAX) [Yamazaki]. O conceito básico consiste em codificar cada grupo contíguo de 1's e 0's utilizando um código de comprimento variável. Os bits são lidos da esquerda para a direita em uma linha e codificados de acordo com o seu comprimento.

Apesar da codificação "run-length" ser um método eficiente de compressão, uma compressão adicional pode ser obtida codificando-se a sua saída com um código de comprimento variável. Na realidade, as seqüências pretas e brancas podem ser codificadas separadamente com códigos de comprimento variável específicos para a sua estatística. Por exemplo, seja $a_j$ uma seqüência preta de comprimento $j$. Pode-se estimar a probabilidade de que o símbolo $a_j$ seja emitido por uma fonte de seqüência pretas, dividindo-se o número de seqüências pretas de comprimento $j$ em toda a imagem pelo número total de seqüências pretas. Uma estimativa da entropia desta fonte de seqüências pretas, indicada por $H_0$, pode então ser calculada. Um argumento semelhante é válido para calcular a entropia das seqüências brancas, indicada por $H_1$. A entropia aproximada da fonte das seqüências é dada por:

$$H_{RL} = \frac{H_0 + H_1}{L_0 + L_1} \tag{3.8}$$

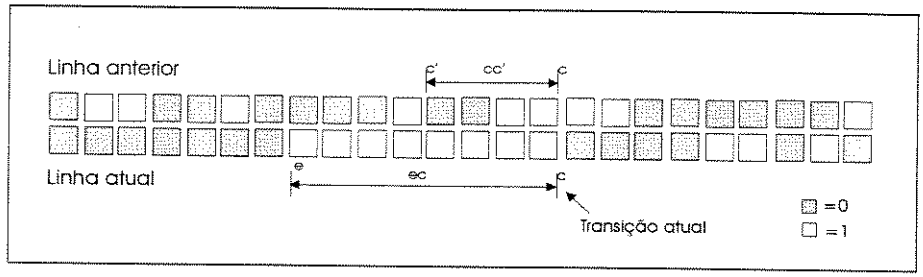



Figura 3.5: Ilustração do processo de endereçamento relativo.

| distância medida | distância | código | intervalo da distância | código $h(d)$ |
|---|---|---|---|---|
| $cc'$ | 0 | 0 | 1-4 | 0xx |
| $ec$ ou $cc'$ | 1 | 100 | 5-20 | 10xxxx |
| $cc'$ (direita) | 1 | 101 | 21-84 | 110xxxxxx |
| $ec$ | $d\ (d > 1)$ | 111 $h(d)$ | 85-340 | 1110xxxxxxxx |
| $cc'$ ($c'$ p/esquerda) | $d\ (d > 1)$ | 1100 $h(d)$ | 341-1364 | 11110xxxxxxxxxx |
| $cc'$ ($c'$ p/direita) | $d\ (d > 1)$ | 1101 $h(d)$ | 1365-5460 | 111110xxxxxxxxxx |

Tabela 3.3: Ilustração da codificação por endereçamento relativo.

onde $L_0$ e $L_1$ indicam os comprimentos médios das seqüências pretas e brancas, respectivamente. A equação (3.8) fornece uma estimativa do número médio de bits por píxel necessários para codificar as seqüências em uma imagem binária, usando um código de comprimento variável.

**Codificação "Run-Length" Bidimensional** Os conceitos da codificação *run-length* unidimensionais são facilmente estendíveis de forma a criar um procedimento bidimensional [Yamazaki]. Um dos melhores resultados obtidos para este caso é a codificação por endereçamento relativo (RAC), que é baseada no princípio de rastreamento das transições que ocorrem ao início e ao final de cada seqüência preta e branca. A Figura 3.5 ilustra uma implementação desta abordagem. Note que **ec** é a distância entre a transição **c** e a última transição da linha corrente **e** e **cc'** é a distância entre **c** e a última transição da linha anterior **c'**. Se $\mathbf{ec} \leq \mathbf{cc'}$, a distância codificada pela RAC, **d**, é igual à **ec** e é utilizada para representar a transição corrente em **c**. Caso contrário, **d** é igual a **cc'**.

Assim como no caso da codificação *run-length* unidimensional, a RAC exige a adoção de uma convenção para determinação dos valores das seqüências. Além disso, deve-se supor que existem transições no começo e no fim de cada linha, assim como uma linha de início. Desta forma, as bordas da imagem podem ser tratadas adequadamente. Na prática, como a distribuição de probabilidade das distâncias da RAC, na maioria das imagens, não é uniforme, o passo final do processo é a codificação da distância RAC selecionada (ou seja, a mais curta) e da sua distância **d**, utilizando-se um código de comprimento variável adequado. Como mostra a Tabela 3.3, um código semelhante ao código $B_1$ pode ser utilizado. As menores distâncias são atribuídas às

menores palavras-código. Todas as outras distâncias são codificadas utilizando-se um prefixo para indicar a menor distância RAC, um segundo prefixo que atribui **d** a um intervalo específico de distâncias e a representação binária de **d** subtraída da distância base do próprio intervalo (indicada por $\times \times \times \times ... \times$ na Tabela 3.3). Se **ec** e **cc'** são iguais a +8 e +4 como na Tabela 3.3, a palavra código RAC apropriada é (1100011). Se $\mathbf{d} = 0$, $c$ está diretamente abaixo de $\mathbf{c}'$. Se $\mathbf{d} = 1$, o decodificador pode precisar determinar o ponto de transição mais próximo, já que o código (100) não especifica se a medida é relativa à linha corrente ou à linha anterior.

## 3.4 Técnicas de Compressão Com-Perdas

A grande diferença entre os esquemas de codificação *sem-perdas* e *com-perdas* é a inclusão do estágio de quantização neste último. Ao quantizar os dados, o número de possíveis símbolos de saída é reduzido. O tipo de quantização tem uma grande influência na taxa de bits e na qualidade do esquema *com-perdas*.

A relativa importância de cada um dos estágios do esquema de compressão varia de uma técnica para outra. O mapeador e o codificador de símbolos do esquema de compressão *com-perdas* são semelhantes aos do esquema *sem-perdas*. As mesmas técnicas apresentadas nas seções anteriores podem, então, ser utilizadas neste esquema. Como regra geral, quanto mais sofisticado o sistema, melhor a qualidade da imagem reconstruída para uma certa taxa de bits.

Em esquemas de compressão *com-perdas*, uma redução extra na taxa de bits é conseguida a custo de degradações na imagem reconstruída. Estas degradações podem ou não ser visíveis na imagem reconstruída. Quanto maior for a degradação permitida, maior será a compressão obtida. Seja $f$ a imagem original $N \times N$ e $\hat{f}$ a imagem reconstruída, a diferença entre estas duas imagens é denominada imagem erro e definida pela expressão:

$$e(i,j) = f(i,j) - \hat{f}(i,j) \tag{3.9}$$

Para avaliar a qualidade da imagem reconstruída, faremos uso do erro RMSE e da amplitude de pico da relação sinal-ruído (PSNR) como métricas do erro. O RMSE é definido como:

$$RMSE = \sqrt{\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left[f(i,j) - \hat{f}(i,j)\right]^2} \tag{3.10}$$

e representa o desvio médio da imagem erro. A medida PSNR (em dB), para uma imagem de 8 bits (255 níveis), é definida como:

$$PSNR = 20\log_{10}\left(\frac{255}{RMSE}\right) \tag{3.11}$$

É importante lembrar que um valor de RMSE baixo (ou um PSNR alto) não implica necessariamente em uma boa qualidade de imagem reconstruída. As medidas de erros, apesar de fornecerem alguma medida da qualidade relativa, nem sempre traduzem de forma adequada a qualidade visual da imagem. Por esta razão, as imagens reconstruídas são sempre apresentadas juntamente com as medidas de erro.

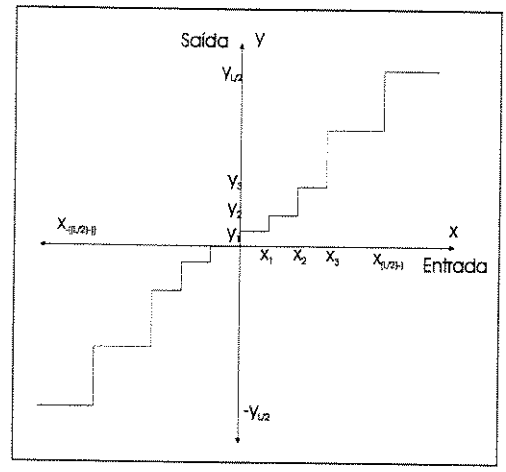



Figura 3.6: Quantizador escalar uniforme.

### 3.4.1 Quantização

Seja $x$ uma quantidade escalar contínua representando a intensidade de um píxel. Para representar $x$ com um número finito de bits, apenas um número finito de níveis de quantização pode ser utilizado. Assumiremos que um total de $L$ níveis são utilizados para representar $x$. O processo de atribuição de um dos $L$ níveis a cada valor de $x$ é denominado quantização da amplitude, ou simplesmente, quantização [Jayant]. De forma grosseira, pode-se dizer que se cada escalar é quantizado de forma independente, o procedimento é denominado quantização escalar. Se dois ou mais escalares são quantizados conjuntamente, o procedimento é denominado quantização vetorial.

#### Quantização Escalar

Existem várias formas de quantização escalar. Dentre elas, a quantização uniforme e a quantização Llyod-Max. Uma quantização escalar, em geral, pode ser interpretada como um mapeamento $Q[\cdot]$ de um conjunto infinito $X = \{x \in \mathbb{R}\}$ em um conjunto finito $Y = \{y_0, y_1, \cdots, y_{L-1} | \; y_i \in \mathbb{R}\}$. Um exemplo do mapeamento de um quantizador escalar é apresentado na Figura 3.6.

O mapeamento $Q[\cdot]$ é sempre definido através de uma partição do conjunto $X$ em intervalos não-sobrepostos $\{X_i\} = (x_{i-1}, x_i)$. Os limites deste intervalo, $x_{i-1}$ e $x_i$, são denominados níveis de decisão. Os intervalos devem satisfazer às seguintes condições:

$$\begin{aligned} \bigcup_{i=0}^{L-1} X_i &= X \\ X_i \cap X_j &= 0 \end{aligned} \tag{3.12}$$

(3.14)

A cada intervalo $X_i$ é associado um nível de representação $y_i$. Portanto, o conjunto $X$ for dividido em $L$ intervalos, haverá $L$ níveis de representação. Os valores do eixo vertical da

Figura 3.6 correspondem aos níveis de representação $\{y_i, i = 0, 1, ..., L-1\}$ do quantizador. Os valores associados ao eixo horizontal correspondem aos níveis de decisão $\{x_i\}$.

Seja $\widehat{x}$ o valor quantizado de $x$. Se $x \in X_i$, $\widehat{x}$ pode ser expresso de acordo com a seguinte definição:

$$x \in \mathbb{R}_i \Rightarrow Q[x] = y_i, \quad i = 0, 1, ..., L-1 \tag{3.15}$$

O desempenho de um quantizador é medido através do erro introduzido pela quantização. Este erro é denominado ruído de quantização e definido pela seguinte expressão:

$$e_Q[n] = \widehat{x}[n] - x[n]$$

onde $x$ e $\widehat{x}$ são, respectivamente, a entrada e a saída do quantizador. O valor $e_Q^2$ pode ser interpretado como um caso especial da medida de distorção $d(\widehat{x}, x)$, que é uma medida da distância entre $\widehat{x}$ e $x$. Outros exemplos de medidas de distorção são $|\widehat{x} - x|$ e $||\widehat{x}|^p - |x|^p|$.

Os níveis de representação e decisão são frequentemente determinados através da minimização de algum critério de erro baseado em $d(\widehat{x}, x)$. Um exemplo é a média da distorção, definida por:

$$D = E[d(\widehat{x}, x)] = \int_{x_0=-\infty}^{\infty} d(\widehat{x}, x_0) \cdot p_X(x_0) \cdot dx_0$$

onde $p_X$ é a função densidade de probabilidade de $x$. Seja $d(\widehat{x}, x) = |\widehat{x} - x|^2$. O erro médio quadrático (MSE) da saída do quantizador é definido pela seguinte expressão:

$$D = E\left(|x - \widehat{x}|^2\right) = \sum_{i=0}^{L-1} \int_{x_{i-1}}^{x_i} (x - y_i)^2 p_X(x_0)\, dx_0 \tag{3.16}$$

Quando $p_X(x_0)$ é uniforme, a quantização ótima possui níveis de representação e decisão uniformemente espaçados e definidos através das seguintes expressões:

$$\begin{aligned} x_{i-1} - x_i &= \Delta, \quad 1 \leq i \leq L \\ y_i &= \frac{(x_{i-1} + x_i)}{2}, \quad 1 \leq i \leq L \end{aligned}$$

onde $\Delta$ é o tamanho do passo, ou seja, o espaçamento entre dois níveis de representação consecutivos ou entre dois níveis de decisão consecutivos. Para este tipo de quantização, denominada quantização uniforme, o valor do MSE é igual a $\Delta^2/2$ [Jayant].

A quantização uniforme é muito simples e fácil de implementar. Os parâmetros do projeto são o passo de quantização ($\Delta$) e o número de intervalos ($N$), ou os limites do sinal ($a$ e $b$), que correspondem à região de sobre-carga. Estes limites são, frequentemente, escolhidos como múltiplos do desvio padrão $\sigma_x$ da função densidade de probabilidade do sinal de entrada. Sejam $a$ e $b$ estes limites, $\Delta = \frac{(b-a)}{N}$ e $\frac{1}{N} = \frac{1}{2^R}$, onde $R$ é o número de bits do quantizador. Então, a distorção para este tipo de quantização é definida por:

$$D = \Delta^2/2 = \frac{(b-a)^2}{2 \cdot N^2} = \sigma_x^2 \cdot 2^{-2R} = C \cdot 2^{-2R} \tag{3.17}$$

Se a densidade de probabilidade não é uniforme, a quantização ótima também não será uniforme. Um quantizador ótimo será aquele que minimizar (3.16) para um número fixo de

níveis. A minimização de (3.16) foi realizada por Max [Max] e Lloyd [Lloyd], que obtiveram as seguintes expressões para os níveis de decisão e representação:

$$\begin{aligned} x_{k,opt} &= \frac{1}{2}\left(y_{k,opt} + y_{k-1,opt}\right), \quad k = 1, 2, ..., L-1. \\ x_{0,opt} &= -\infty \\ x_{L,opt} &= \infty \\ y_{k,opt} &= \frac{\int_{x_{k,opt}}^{x_{k+1,opt}} x \cdot p_X(x)\, dx}{\int_{x_{k,opt}}^{x_{k+1,opt}} p_X(x)\, dx}, \quad k = 1, 2, ..., L-1. \end{aligned} \tag{3.18}$$

Estas expressões não fornecem uma forma explícita de mapear a quantização. A primeira expressão indica que os níveis de decisão devem estar no ponto médio entre níveis de representação vizinhos. A última afirma que os níveis de representação ótimos são os centróides da função densidade de probabilidade do intervalo apropriado. Estas expressões podem ser utilizadas de forma iterativa para obter os níveis de representação e decisão ótimos. Este quantizador escalar ótimo é conhecido como quantizador Llyod-Max.

### Quantização Vetorial

O conceito da quantização vetorial é uma generalização da quantização escalar. O quantizador vetorial $Q[.]$ mapeia um vetor $\mathbf{x} \in \mathbb{R}^N$ em um conjunto finito $\mathbf{Y} = \left[\mathbf{y}_0, ..., \mathbf{y}_{L-1} \mid \mathbf{y}_i \in \mathbb{R}^N\right]$ de vetores de representação, ou seja:

$$Q : \mathbb{R}^N \rightarrow \mathbf{Y}$$

Enquanto que a quantização escalar mapeia um valor $x$, que pertence a um conjunto infinito, em um conjunto finito $Y$, através da seguinte definição:

$$\mathbf{x} \in C_i \Rightarrow Q[\mathbf{x}] = \mathbf{y}_i, \quad i = 0, 1, 2, ..., L-1.$$

Os vetores $\{\mathbf{y}_i\}$ correspondem aos níveis de representação em um quantizador escalar. Em uma quantização vetorial, a coleção de níveis de representação é conhecida como "codebook". As células $C_i$ correspondem aos níveis de decisão e podem ser interpretadas como polígonos sólidos no espaço $N$-dimensional $\mathbb{R}^N$. No caso escalar, é fácil testar se uma amostra do sinal pertence ou não ao intervalo. No caso da quantização vetorial, uma abordagem indireta é necessária, onde é utilizado o critério de fidelidade ou a medida de distorção. Assim, $\mathbf{x} \in C_i$ se a distorção de $\mathbf{x}$ em relação a $\mathbf{y}_i$ for inferior ou igual à distorção em relação à qualquer $\mathbf{y}_j$ para $j \neq i$, ou seja:

$$Q[\mathbf{x}] = \mathbf{y}_i \Leftrightarrow d(\mathbf{x}, \mathbf{y}_i) \leq d\left(\mathbf{x}, \mathbf{y}_j\right), \quad j = 1, 2, ..., L-1. \tag{3.19}$$

Quando a melhor aproximação $\mathbf{y}_i$ foi encontrada, o índice $i$ é codificado como representação eficiente do vetor. O receptor pode reconstruir $\mathbf{y}_i$ simplesmente procurando a representação do vetor na célula de número $i$ do "codebook". A taxa de bits por amostra deste esquema é igual a $\frac{1}{N}\log_2 L$, quando o índice $i$ é codificado diretamente.

Apesar de apresentar resultados superiores aos da quantização escalar, a quantização vetorial implica em um grande aumento na complexidade do sistema. A sua implementação exige o cálculo de um "codebook", o que pode é mais custoso. Maiores detalhes desta técnica podem ser encontrados em [Gersho]. No nosso trabalho, como queremos focalizar o desempenho da transformada wavelet, utilizaremos apenas a quantização escalar, por uma questão de simplicidade.

### 3.4.2 Codificação Preditiva Com-Perdas

Em geral, qualquer um dos estágios do esquema de codificação *com-perdas* pode ser implementado de forma adaptativa ou não. Um esquema de compressão é adaptativo se a estrutura dos seus estágios, ou dos seus parâmetros, muda localmente dentro da imagem com o objetivo de tirar proveito das variações da estatística local da imagem. A adaptabilidade permite uma melhor codificação ao custo de um aumento na complexidade do sistema.

A adaptabilidade pode ser causal ou não causal. Em sistemas com adaptabilidade causal, os parâmetros do codificador são baseados apenas nos valores dos píxeis previamente reconstruídos. Qualquer processo que leve a uma decisão no codificador é repetido no decodificador. Estes sistemas oferecem a vantagem de não exigir que um cabeçalho, contendo bits de informação adicionais, seja enviado ao decodificador. Infelizmente, há duas desvantagens associadas a este esquema. A primeira é que o sistema pode falhar na tentativa de prever mudanças muito bruscas na estatística do sinal, que não possam ser inferidas a partir dos valores anteriores do sinal. A segunda desvantagem é o fato da adaptabilidade causal aumentar a complexidade do codificador e do decodificador na mesma proporção, já que todos o processos de tomada de decisão do codificador devem ser repetidos no decodificador.

Em sistemas com adaptabilidade não-causal, os parâmetros do codificador são baseados nos valores dos píxeis anteriores (originais ou reconstruídos) e também nos seus valores futuros. Como os últimos não estão disponíveis no decodificador, o codificador deve enviar bits adicionais ao decodificador para informá-lo das suas adaptações. Apesar do aumento na taxa de bits, o desempenho deste sistema é melhor que o desempenho do sistema causal. Além disso, o aumento da complexidade no decodificador devido à adaptação é mínimo, já que é não preciso repetir todos os processos de tomada de decisão do codificador.

#### Modulação por Código de Pulsos Diferencial ( DPCM )

Em um esquema de codificação preditivo, a correlação entre os píxeis vizinhos de uma imagem é utilizada no cálculo da predição de cada píxel. A abordagem mais comum é o DPCM (*Differencial Pulse code modulation* - Modulação por código de pulsos diferencial ), que subtrai a predição do valor real do píxel, gerando uma imagem diferencial menos correlacionada que a imagem original. Esta imagem diferencial é, então, quantizada e codificada. O processo de quantização determina a taxa de bits resultante e a qualidade da imagem.

Em um esquema DPCM, os $m$ píxeis dentro da vizinhança causal de um píxel são utilizados no cálculo do valor da sua predição linear. Na Figura 3.7, apresentamos um exemplo da ordenação dos píxeis dentro de uma imagem. Para o píxel $x_m$, o valor da sua predição linear

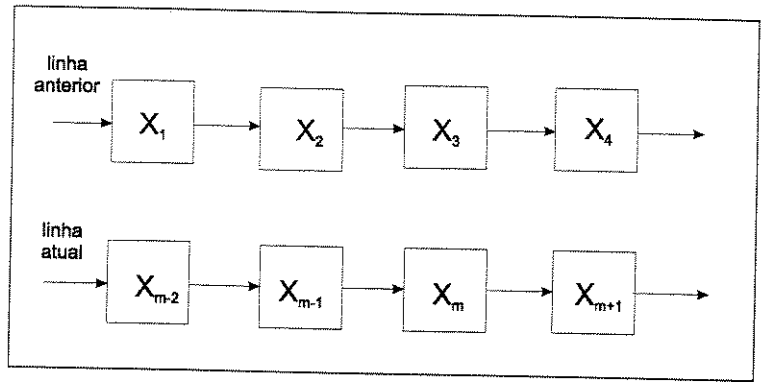



Figura 3.7: Modelo da ordenação dos píxeis dentro de uma imagem.

indicada por $\tilde{x}_m$ é calculado através da seguinte expressão:

$$\tilde{x}_m = \sum_{i=0}^{m-1} \alpha_i x_i \tag{3.20}$$

onde os $\alpha_i$ são os coeficientes do preditor. Este valor será aproximado pelo inteiro mais próximo, com o objetivo de reduzir a complexidade do preditor. Para uma imagem de $n$ bits, este valor é mantido dentro do intervalo $[0, 2^n - 1]$.

A imagem diferencial, $e_m$, é calculada a partir da diferença entre o valor da predição e o valor real, isto é:

$$e_m = x_m - \tilde{x}_m \tag{3.21}$$

A imagem diferencial tem uma variância bem menor e é significativamente menos correlacionada que a imagem original. Além disso, tem um histograma estável que pode ser aproximado por uma distribuição gaussiana.

Nas Figuras 3.8 e 3.9 apresentamos os diagramas de blocos do transmissor e do receptor DPCM básico, respectivamente. Observe que o receptor tem acesso apenas aos valores dos píxeis reconstruídos. Como a quantização da imagem diferencial introduz erros, os valores da imagem reconstruída geralmente diferem dos valores da imagem original. Para garantir que predições idênticas sejam utilizadas tanto no preditor quanto no receptor, o transmissor irá basear as suas predições nos valores reconstruídos, o que é feito utilizando uma realimentação. Com isso, a expressão (3.21) é modificada:

$$\tilde{x}_m = \sum_{i=0}^{m-1} \alpha_i x'_i \tag{3.22}$$

onde $x'_i$ é o valor do píxel reconstruído na posição $i$ e os $\alpha_i$s são os coeficientes do preditor.

O projeto de um DPCM consiste, então, na otimização das componentes do preditor e do quantizador. Por causa da inclusão do quantizador no ramo de realimentação, há uma dependência entre o erro de quantização e o erro de predição. Uma otimização que levasse em conta os dois erros seria interessante. Entretanto, para evitar a complexidade de um modelo deste tipo, as componentes são geralmente otimizadas separadamente.

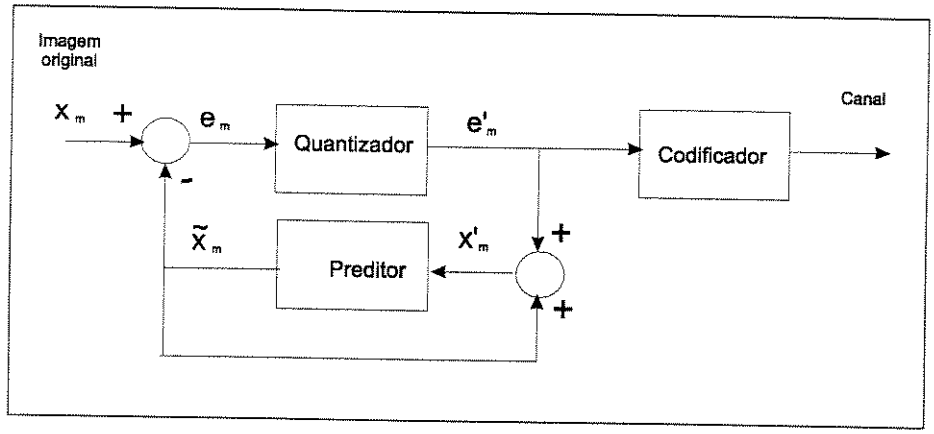



Figura 3.8: Diagrama em blocos do transmissor DPCM.

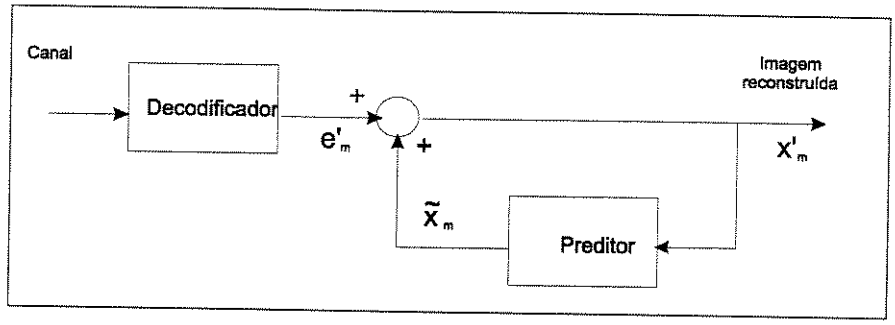



Figura 3.9: Diagrama em blocos do receptor DPCM.

### DPCM Adaptativo

A maior limitação do sistema DPCM é o fato do quantizador e do preditor serem fixos para toda a imagem. Os esquemas DPCM podem ser adaptativos em relação ao quantizador, ao preditor ou a ambos. Uma predição adaptativa geralmente reduz o erro de predição devido a quantização e, assim, para uma mesma taxa de bits um erro de quantização menor é obtido e, consequentemente, uma melhor qualidade de imagem reconstruída.

## 3.4.3 Codificação por Transformação

Uma alternativa para o esquema básico de compressão apresentado nas Figuras 3.1 e 3.2 pode ser visto na Figura 3.10. Este esquema, denominado de esquema de codificação através de uma transformada genérica, apresenta além dos estágios de quantização e codificação de símbolos, um estágio de segmentação da imagem em blocos e um estágio de transformada. A transformada utilizada neste tipo de esquema é unitária, ou seja, é linear, inversível e sua base é formada por um conjunto completo de funções discretas ortonormais. Esta transformada faz o papel do mapeador no esquema de compressão, pois é através desta operação que os dados da imagem original são mapeados em um novo domínio, denominado domínio da transformada.

O objetivo da transformada é descorrelacionar o sinal original. Esta descorrelação geral-

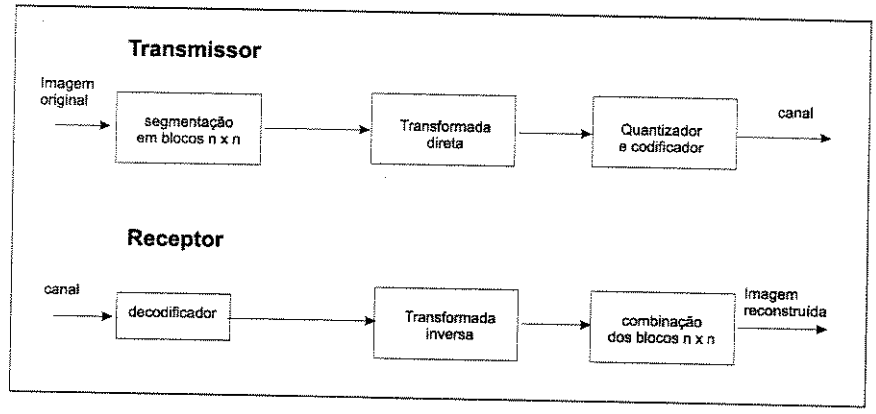



Figura 3.10: Diagrama da codificação por transformada.

mente resulta em um agrupamento da maior parte da energia do sinal em um pequeno grupo de coeficientes. Desta maneira, muitos coeficientes podem ser descartados entre os estágios de quantização e codificação. Além disso, uma compressão sem degradações visíveis pode ser obtida incorporando a função sensibilidade de contraste do sistema visual humano (HSVT) na quantização dos coeficientes [Gonzalez].

Uma transformada é unidimensional ( 1-D ) se aplicada em uma única direção da imagem, isto é, na direção das linhas ou das colunas. Uma transformada 1-D definida para $n$ píxeis é denominada transformada de $n$-pontos. Uma transformada é bidimensional ( 2-D ) se aplicada simultaneamente em duas direções (linhas e colunas ou horizontal e vertical, por exemplo). Todas as transformada 2-D consideradas neste trabalho são separáveis. Portanto, a transformada 2-D em um bloco de $n \times n$ píxeis, pode ser dividida em duas transformadas 1-D de $n$ pontos, sendo que uma destas é aplicada nas linhas da imagem, enquanto que a outra é aplicada nas colunas.

### Transformadas para Imagens

Há muitas características que são desejáveis em uma transformada utilizada com o propósito de comprimir imagens. Algumas destas características são:

- Descorrelação da imagem: A transformada ideal descorrelaciona completamente os dados em um bloco, ou seja, ela agrupa a maior parte da energia em um grupo pequeno de coeficientes.

- Bases de funções independentes da imagem utilizada: Devido às grandes variações estatísticas entre as imagens, a transformada ótima geralmente depende da imagem. Infelizmente, encontrar a transformada ótima é computacionalmente inviável. Isto é particularmente um problema se os blocos de imagens forem altamente não-estacionários, o que torna necessário a utilização de mais de um conjunto de funções de base para atingir uma alta descorrelação. Além disso, a transmissão destas bases de funções exige a utilização de bits adicionais, o que pode prejudicar a eficiência da compressão. Como resultado, é

geralmente desejável abrir mão de um desempenho de codificação ótimo, optando pela utilização de funções de base independentes da imagem.

- Implementação rápida: O número de operações necessárias para uma transformada de $n$-pontos é geralmente da ordem de $O(n^2)$. Algumas transformadas têm implementações rápidas, com um número de operações da ordem de $O(n \log n)$. Implementar uma transformada 2-D $n \times n$ utilizando duas transformadas 1-D rápidas, uma aplicada nas linhas e a outra nas colunas, reduz o número de operações de $O(n^4)$ para $O(2n^2 \log n)$.

**Karhunen-Loève Transform - KLT** A KLT é a transformada ótima em relação ao empacotamento de energia, isto é, os coeficientes KLT apresentam uma maior concentração de energia quando comparados a qualquer outra transformada. Infelizmente, as funções da base da KLT são dependentes da imagem. Para calculá-las é preciso fazer uma estimação da função covariância da imagem. Além disso, não existe um algoritmo KLT rápido. Estas desvantagens limitam seriamente o uso desta transformada para compressão de imagens .

**Discrete Fourier Transform - DFT** A DFT é comumente utilizada para análise espectral e filtragem. Para um bloco de $n \times n$ píxeis de um sinal, a DFT 2-D é definida como:

$$F(u,v) = \frac{1}{n} \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} f(j,k) \exp\left(-\frac{2\pi i (uj + vk)}{n}\right) \tag{3.23}$$

e a sua inversa como:

$$f(j,k) = \frac{1}{n} \sum_{u=0}^{n-1} \sum_{v=0}^{n-1} F(u,v) \exp\left(\frac{2\pi i (uj + vk)}{n}\right) \tag{3.24}$$

onde $i = \sqrt{-1}$. A DFT decompõe cada bloco da imagem em suas componentes espectrais. Os índices $u$ e $v$ são denominados freqüências espaciais da transformada. O núcleo 2-D da DFT é separável, ou seja:

$$\exp\left(\frac{2\pi i (uj + vk)}{n}\right) = \exp\left(\frac{2\pi i \cdot uj}{n}\right) \exp\left(\frac{2\pi i \cdot vk}{n}\right)$$

o que permite que a transformada 2-D seja implementada como duas transformadas 1-D. Algoritmos rápidos para o cálculo da DFT existem e exigem uma complexidade computacional da ordem de $O(n \log_2 n)$ para uma transformada de $n$ pontos.

Em geral, os coeficientes gerados a partir da DFT são números complexos. Portanto, o armazenamento e manipulação destes coeficientes é um pouco mais trabalhoso, o que é uma desvantagem. Há na realidade um total de $2n^2$ componentes da transformada, mas devido à propriedade da simetria conjugada [Lim] [Jain88] [Rabbani], a seguinte relação é válida:

$$F(u,v) = F(-u + l \cdot n, -v + m \cdot n), \qquad l, m = 0, 1, 2... \tag{3.25}$$

Logo, quase metade das componentes da transformada são redundantes e podem ser calculados a partir dos outros componentes.

Uma outra desvantagem da DFT é a geração de componentes espúrios no espectro devido à periodicidade implícita dos blocos de imagem. Quando codificados à baixas taxas, estas componentes espúrias dão origem à elementos de blocagem. Este fenômeno será melhor discutido no próximo item.

**"Discrete Cosine Transform" - DCT** Para aplicações em processamento de imagens, a DCT 2-D direta de um bloco de $n \times n$ píxeis é definida como [Jain88] [Lim]:

$$F(u,v) = \frac{4C(u)C(v)}{n^2} \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} f(j,k) \cos\left(\frac{u\pi(2j+1)}{2n}\right) \cos\left(\frac{v\pi(2k+1)}{2n}\right) \tag{3.26}$$

e a sua inversa como:

$$f(j,k) = \sum_{u=0}^{n-1} \sum_{v=0}^{n-1} C(u)C(v)F(u,v) \cos\left(\frac{u\pi(2j+1)}{2n}\right) \cos\left(\frac{v\pi(2k+1)}{2n}\right) \tag{3.27}$$

onde

$$C(l) = \begin{cases} \frac{1}{\sqrt{2}} & \text{para } l = 0 \\ 1 & \text{para } l = 1, 2, .., n-1 \end{cases} \tag{3.28}$$

Para imagens típicas, que exibem uma alta correlação píxel a píxel, o desempenho da DCT é comparável ao da KLT. Na realidade, pode ser demonstrado que se o sinal puder ser modelado por uma fonte de Markov de primeira ordem, a base de funções da DCT é idêntica à base de funções da KLT [Lim].

Assim como a DFT, a DCT tem uma implementação rápida com uma complexidade matemática da ordem de $O(n \log n)$, para uma transformada de n-pontos. Entretanto, a DCT apresenta uma eficiência de compressão maior, já que ela evita a geração de componentes espúrias. A DFT é a representação em série de Fourier de uma seqüência de duração finita. Logo, há uma periodicidade implícita na definição, como pode ser visualizado na Figura 3.11-(a). Esta periodicidade é o resultado de uma amostragem no domínio da freqüência. Replicando a seqüência original desta maneira, descontinuidade severas são criadas entre os segmentos. Estas descontinuidades geram componentes espúrias de alta freqüência que podem reduzir consideravelmente a eficiência da transformada. Apesar destas componentes espúrias não fazerem parte da seqüência original, elas são necessárias para reconstruir as bordas de uma seqüência periódica. A sua eliminação, com o intuito de aumentar a eficiência da transformada, pode acarretar em erros na reconstrução das bordas. Na codificação de imagens, a imagem é quebrada em vários blocos de píxeis para formar seqüências 2-D. Os erros de reconstrução resultam, então, em elementos de blocagem, isto é, as bordas entre blocos adjacentes são altamente visíveis.

Para eliminar as descontinuidades das bordas, a seqüência original de n pontos pode ser estendida em uma seqüência de 2n pontos, através da sua reflexão em torno do eixo vertical. A seqüência estendida é então replicada de maneira a formar uma seqüência periódica, necessária para o cálculo da série de Fourier discreta. Como apresentado na Figura 3.11-(b), esta seqüência periódica não tem qualquer descontinuidade nas bordas e, portanto, nenhuma componente espectral espúria é introduzida na DFT. Pode-se perceber que a seqüência estendida resultante de 2n pontos é simétrica e par. Portanto, a DFT de 2n pontos resultante também é par e simétrica e apenas n pontos são necessários para representá-la. Como neste caso estamos lidando apenas com quantidades reais, a DFT requer apenas cálculos reais. O processo de

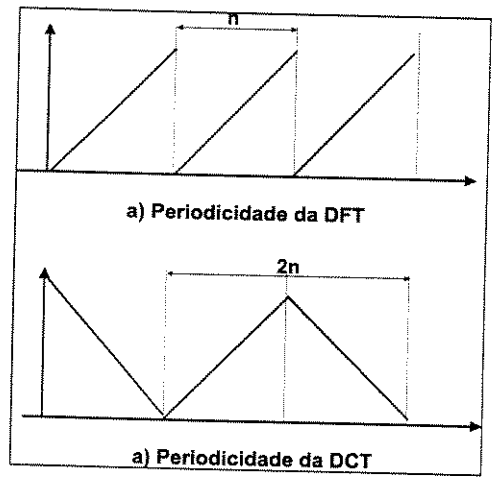



Figura 3.11: Periodicidade da (a) DFT e (b) DCT.

tomar uma DFT de 2n pontos da seqüência estendida de n pontos é equivalente a tomar a DCT da seqüência de n-pontos original. Na realidade, a DCT pode ser calculada como uma FFT de 2n pontos. Apesar da DFT e da DCT estarem intimamente relacionadas, a simetria implícita da DCT resulta em duas grandes vantagens sobre a DFT:

- A DCT não gera componentes espectrais espúrias. Portanto, a eficiência da codificação continua alta e os componentes de blocagem são reduzidos.
- Apenas cálculos reais são utilizados para a DCT.

Por causa dessas vantagens, a DCT se tornou a transformada mais utilizada para compressão de imagens.

**Walsh-Hadmard Transform - WHT** Apesar da WHT deixar muito a desejar com relação ao empacotamento de energia, sua implementação simples tornou-a popular especialmente para implementações em hardware [Lim]. A base de funções da WHT contém valores iguais a +1 ou -1, e podem ser interpretados como linhas das matrizes de Hadamard. A menor matriz de Hadamard ortonormal tem tamanho $2 \times 2$:

$$\mathbf{H}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{3.29}$$

Para construir matrizes de Hadamard maiores, podemos usar a seguinte relação recursiva:

$$\mathbf{H}_{2n} = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{H}_n & \mathbf{H}_n \\ \mathbf{H}_n & -\mathbf{H}_n \end{bmatrix} \tag{3.30}$$

onde $\mathbf{H}_{2n}$ é a matriz de Hadamard de tamanho $2n \times 2n$.

### Estratégias de Seleção dos Coeficientes

Um esquema de codificação por transformada utiliza uma transformada, geralmente independente, como por exemplo a DCT ou a WHT. Em seguida emprega-se uma certa estratégia de amostras que define a seleção dos coeficientes da transformada. Estes coeficientes são, então, quantizados e codificados. A escolha de uma estratégia adequada pode aumentar a eficiência destes dois últimos estágios. Nesta seção estudaremos duas das estratégias de seleção de amostras mais utilizadas: a seleção de amostras por zona e a seleção de amostras por limiar.

**Seleção de Amostras por Zonas** A amostragem por zonas consiste basicamente na retenção dos coeficientes que estejam localizados em zonas pré-estabelecidas do bloco transformado e na anulação de todos os outros coeficientes do bloco. Como a maioria das imagens tem uma concentração de energia nas baixas freqüências, os coeficientes de baixas freqüências são geralmente retidos, enquanto que os de freqüências mais altas são descartados. Cada coeficiente retido é então quantizado e codificado como uma palavra-código de comprimento fixo. É possível utilizar o mesmo quantizador para todos os coeficientes de um bloco, ou ainda, utilizar vários quantizadores para os diferentes grupos de coeficientes. Por exemplo, podemos utilizar um quantizador de 8 bits para os três coeficientes de freqüência mais baixa , um quantizador de 7 bits para os 5 coeficientes seguintes e assim por diante. Como as posições das amostras selecionadas e os quantizadores são especificados previamente, não há necessidade de enviar nenhuma informação adicional.

Uma abordagem um pouco mais refinada consiste na alocação de bits dentro de uma zona pré-estabelecida de maneira a minimizar o erro total de quantização. Isto é feito atribuindo bits na proporção do logaritmo das variâncias dos coeficientes. A variância atribuída à cada posição dos coeficientes é encontrada através do cálculo da energia dos coeficientes correspondentes e da média desta com relação à todos os blocos da imagem. Esta abordagem divide a codificação em duas etapas. Na primeira etapa, calculam-se as variâncias de todas as posições dos coeficientes e determina-se os quantizadores apropriados. A informação do quantizador será transmitida ao receptor como informação adicional (cabeçalho), na forma de uma matriz de alocação de bits. Na segunda etapa, os blocos são quantizados e codificados.

Na técnica descrita, a matriz de alocação de bits foi calculada de acordo com a estatística global da imagem. Devido à natureza não estacionária das imagens, pode haver variações significativas entre as estatísticas dos diversos blocos. Se apenas uma matriz de alocação de bits é utilizada para todos os blocos, quando o coeficiente diferir muito do valor esperado os erros podem atingir valores muito grandes. Este problema é atenuado quando utiliza-se uma versão adaptativa localmente desta técnica. Nesta versão, cada bloco é classificado como pertencendo a uma classe e, à cada classe, uma matriz de alocação de bits diferente é utilizada. A classificação é baseada na variância total do bloco, que é uma boa medida da sua atividade. Esta versão também divide a codificação em duas etapas. Na primeira etapa, as diferentes classes são formadas através do cálculo das estatísticas de todos os blocos da imagem. Geralmente, um número igual de blocos é atribuído a cada classe. A seguir, a zona de retenção e a matriz de alocação de bits para cada classe são determinadas e transmitidas ao receptor. Na segunda etapa, esta informação é utilizada para classificar cada bloco e codificá-lo e quantizá-lo de acordo com a matriz de alocação de bits correspondente.

Os esquemas apresentados tem como objetivo a codificação de imagens a uma taxa de

bits constante. Uma taxa de bits menor, mas variável, pode ser obtida se utilizarmos codificadores de comprimento variável. A distribuição das amplitudes dos coeficientes é geralmente não-uniforme. Como resultado, os níveis de saída do quantizador apresentam freqüências de ocorrência bastante variáveis e podem ser codificados, de forma vantajosa, utilizando um código de comprimento variável. A redução na taxa de bits é particularmente significativa quando o quantizador é uniforme e as freqüências de ocorrências variam significativamente entre si. No caso de um quantizador não-uniforme, as freqüências de ocorrência não variam tanto e um código de Huffman global pode ser utilizado, já que as freqüências dos símbolos são semelhantes de imagem para imagem.

**Seleção de Amostras por Limiar** A maior desvantagem da estratégia de seleção de amostras por zona é o possível descarte de coeficientes com energia significativa, que estejam fora da zona de retenção, o que pode resultar em erros de reconstrução grandes. Para solucionar este problema, pode ser utilizado um processo de seleção por limiar das amostras, onde um nível de limiar é escolhido e apenas os coeficientes cujos valores estão acima deste limiar são quantizados e codificados. Uma quantização uniforme ou não-uniforme pode ser utilizada. Se um quantizador uniforme é utilizado, os níveis de saída são geralmente codificados entropicamente.

Uma desvantagem desta técnica de seleção é a necessidade de transmissão do endereçamento dos coeficientes selecionados, já que o número e posição dos coeficientes acima do limiar varia de bloco para bloco. As técnicas para codificação e endereçamento são geralmente baseadas na codificacão runlength de ordem zero.

Um refinamento desta técnica de amostragem por limiar pode ser obtido através da transmissão de apenas os L maiores coeficientes de cada bloco. O valor de L pode ser fixo ou variar dependendo de algum atributo do bloco, como, por exemplo, o valor da sua variância. A estrutura do quantizador pode ser uma função da posição dos coeficientes, de modo que a quantidade de distorção introduzida por cada coeficiente possa ser controlada. Para simplificar a implementação, os quantizadores podem ser projetados de modo a diferir apenas por uma fator de escala. Logo, um único quantizador pode ser utilizado para todos os coeficientes, desde que, antes da quantização, os coeficientes sejam escalonados para se obter o tamanho de passo desejado. Um método útil para a determinação da normalização para cada coeficiente consiste em utilizar a função de sensibilidade de contraste do sistema visual humano (HSTV). Desta maneira, o erro de quantização para cada coeficiente pode ser definido como sendo função da importância visual deste coeficiente.

A quantização no algoritmo DCT JPEG, a ser descrito a seguir, utiliza uma técnica de seleção por limiar que permite que uma matriz de normalização específica para cada aplicação seja aplicada aos coeficientes.

### Algoritmo JPEG DCT

A CCITT e ISO colaboraram para desenvolver o mais popular padrão de compressão para imagens estáticas de níveis de cinza - o JPEG . O padrão JPEG define três sistemas de codificação diferentes:

1. Um sistema "básico" de codificação *com-perdas* - Este sistema é baseado na DCT e é adequado à maioria das aplicações em compressão de imagens estáticas.

2. Um sistema aperfeiçoado de codificação *com-perdas* - Com este sistema é possível obter maiores taxas de compressão e uma maior precisão. Entre as características deste sistema se encontram a compressão de imagens de 12 bits/píxel, a codificação aritmética e a construção progressiva seqüencial e hierárquica.

3. Um sistema de codificação *sem-perdas* - Este sistema se destina a aplicações que não toleram perdas.

Para ser considerado compatível com o padrão JPEG, um sistema de compressão deve apresentar no mínimo as características do sistema básico de codificação (item 1). Nenhum formato ou resolução espacial do arquivo de entrada precisa ser especificado. Em seguida, apresentaremos uma descrição breve do sistema básico de codificação do JPEG:

**Sistema Básico do JPEG** O sistema básico do JPEG apresenta as seguintes características:

- A imagem original (8 bits/píxel) é particionada em blocos de $8 \times 8$ píxeis e cada bloco é transformado de forma independente utilizando a DCT.

- Todos os coeficientes da transformada são normalizados através de uma matriz de normalização, pré-definida, e fixa para todos os blocos. Cada componente da matriz de normalização é um número inteiro de 8 bits transmitido ao receptor como parte do cabeçalho de cada imagem. Até quatro matrizes de normalização diferentes podem ser especificadas, isto é, podemos utilizar, por exemplo, diferentes matrizes de normalização para as diferentes componentes de cor de uma imagem colorida. Os coeficientes normalizados são então uniformemente quantizados através de um arredondamento para o inteiro mais próximo. A matriz de normalização pode ser vista como um escalonamento do quantizador de maneira a controlar a quantidade de erro de quantização em cada coeficiente. A função de sensibilidade ao contraste do sistema visual humano (HSVT) pode ser utilizada como guia para desenvolvimento de uma matriz de normalização que trate cada coeficiente de acordo com a sua importância visual.

- O coeficiente superior esquerdo da matriz resultante da aplicação da DCT 2-D é conhecido como o coeficiente DC. O seu valor é proporcional à média de brilho do bloco. Após a quantização, este coeficiente é codificado através de um esquema DPCM, utilizando o coeficiente DC quantizado do bloco anterior como predição. Para o sistema básico, até duas tabelas de Huffman podem ser utilizadas para codificar o sinal diferença resultante. Estas tabelas devem ser especificadas nos bits de cabeçalho.

- A quantização dos coeficientes AC gera muitos coeficientes nulos, especialmente nas altas freqüências. Para tirar proveito desta característica, a matriz dos coeficientes DCT é transformada em um vetor através de um reordenamento em zig-zag. Esta operação ordena os coeficientes aproximadamente de acordo com a sua energia média e, conseqüentemente, cria uma grande quantidade de seqüências de zeros.

- Para codificar os coeficientes AC, cada coeficiente não-nulo é inicialmente descrito através de um número $I$ de 8 bits composto, que apresenta a seguinte forma:

$$I = NNNNSSSS$$

Os dígitos menos significativos, $SSSS$, definem a categoria da amplitude dos coeficientes. Os valores da categoria $k$ estão dentro do intervalo $\left(2^{k-1}, 2^k - 1\right)$ ou $\left(-2^k + 1, -2^{k-1}\right)$, onde $1 \leq k \leq 10$ para o sistema básico. Os valores dos coeficientes contidos em cada categoria são apresentados na Tabela 3.4. Especificada a categoria é necessário enviar $k$ bits adicionais para especificar completamente o sinal e a amplitude do coeficiente dentro da categoria. Os quatro bits mais significativos de $I$, $NNNN$, fornecem a posição do coeficiente com relação ao coeficiente não-nulo anterior. O comprimento especificado por $NNNN$ pode variar de 0 a 15. Um símbolo adicional, $I = (11110000)$, é definido para representar uma distância de 16 coeficientes nulos. Se esta distância excede 16 coeficientes nulos, ela é então codificada utilizando vários símbolos. Outro símbolo especial, $I = 0$, é utilizado para codificar o final do bloco, sinalizando que os coeficientes restantes são todos nulos. Portanto, o conjunto total de símbolos contém 162 membros ( 10 categorias $\times$ 16 valores de distâncias + dois símbolos adicionais ). Em seguida, os símbolos de saída para cada bloco são codificados utilizando o código de Huffman e ao resultado é acrescentado símbolos de cabeçalho, necessários para especificar o sinal e a magnitude do coeficiente dentro da categoria. Os símbolos de cabeçalho possuem uma distribuição uniforme e, portanto, comprimento fixo. Até duas tabelas de Huffman separadas podem ser especificadas no sistema básico.

| Categoria | AC coeficiente |
|---|---|
| 1 | -1,1 |
| 2 | -3,-2,2,3 |
| 3 | -7,...,-4,4,...,7 |
| 4 | -15,...,-8,8,...,15 |
| 5 | -31,...,-16,16,...,31 |
| 6 | -63,...,-32,32,...,63 |
| 7 | -127,...,-64,64,...,127 |
| 8 | -255,...,-128,128,...,255 |
| 9 | -511,...,-256,256,...,511 |
| 10 | -1023,...,-512,512,...,1023 |

Tabela 3.4: Agrupamento de coeficientes AC.

- Cada componente de uma imagem colorida é codificada independentemente. Por exemplo, uma imagem representada no espaço YIQ [Gonzalez] é codificada essencialmente como três imagens separadas.

- No decodificador, após a cadeia de bits ser codificada e a matriz de coeficientes quantizados ser recuperada, cada coeficiente é desnormalizado, através da multiplicação da matriz de normalização inversa pela componente correspondente. À matriz resultante é aplicada a DCT inversa. Obtemos, então, uma aproximação do bloco da imagem original. O erro de reconstrução resultante é controlado pela matriz de normalização.

**Exemplo**

No exemplo a seguir ilustramos os passos na codificação de um bloco utilizando o sistema básico do algoritmo JPEG. Considere o seguinte bloco de $8 \times 8$ píxeis de uma imagem:

$$\mathbf{f} = \begin{bmatrix} 139 & 144 & 149 & 153 & 155 & 155 & 155 & 155 \\ 144 & 151 & 153 & 156 & 159 & 156 & 156 & 156 \\ 150 & 155 & 160 & 163 & 158 & 156 & 156 & 156 \\ 159 & 161 & 162 & 160 & 160 & 159 & 159 & 159 \\ 159 & 160 & 161 & 162 & 162 & 155 & 155 & 155 \\ 161 & 161 & 161 & 161 & 160 & 157 & 157 & 157 \\ 162 & 162 & 161 & 163 & 162 & 157 & 157 & 157 \\ 162 & 162 & 161 & 161 & 163 & 158 & 158 & 158 \end{bmatrix} \tag{3.31}$$

O sistema JPEG encontrado nas aplicações reais utiliza uma DCT ligeiramente diferente. A DCT JPEG direta é definida como:

$$F(u,v) = \frac{C(u)\,C(v)}{4} \sum_{j=0}^{7} \sum_{k=0}^{7} f(j,k) \cos\left(\frac{u\pi(2j+1)}{16}\right) \cos\left(\frac{v\pi(2k+1)}{16}\right) \tag{3.32}$$

A equação (3.32) é idêntica à (3.23) para $n = 8$, exceto por um fator de 4. Este escalonamento tem o objetivo de preservar as distâncias na transformada direta, o que não aconteceria com a aplicação direta de (3.23). Para reduzir a complexidade de hardware e software, várias implementações de (3.32) foram desenvolvidas com o intuito de minimizar o número de multiplicações e adições. Todos os algoritmos DCT práticos utilizam uma aritmética inteira de precisão finita. O bloco resultante da aplicação de (3.32) em (3.31) é:

$$\mathbf{F} = \begin{bmatrix} 1260 & -1 & -12 & -5 & 2 & -2 & -3 & 1 \\ -23 & -17 & -6 & -3 & -3 & 0 & 0 & -1 \\ -11 & -9 & -2 & 2 & 0 & -1 & -1 & 0 \\ -7 & -2 & 0 & 1 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 2 & 0 & -1 & 1 & 1 \\ 2 & 0 & 2 & 0 & -1 & 1 & 1 & -1 \\ -1 & 0 & 0 & -1 & 0 & 2 & 1 & -1 \\ -3 & 2 & -4 & -2 & 2 & 1 & -1 & 0 \end{bmatrix} \tag{3.33}$$

O coeficiente superior esquerdo da matriz resultante, que chamamos de nível DC da imagem, é igual a oito vezes a média do bloco. A energia do bloco está concentrada apenas em alguns coeficientes de baixa freqüência. Os coeficientes da DCT são então normalizados e quantizados utilizando uma matriz de normalização definida de acordo com a aplicação e fixa para todos os blocos da imagem. Cada componente da matriz de normalização, $Q(u,v)$, é um número inteiro de 8 bits que determina o tamanho do passo de quantização, ou seja, valores maiores correspondem a passos maiores e vice-versa. A taxa de bits de uma imagem codificada pode ser variada modificando esta matriz. A matriz pode ser, por exemplo, multiplicada por um fator

constante. Uma matriz de normalização típica utilizada no JPEG é apresentada em (3.34):

$$\mathbf{Q} = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix} \tag{3.34}$$

Os coeficientes normalizados e quantizados resultantes, $F^*(u,v)$, são calculados através da expressão:

$$F^*(u,v) = \textit{inteiro mais próximo}\left(\frac{F(u,v)}{Q(u,v)}\right) \approx \left\lfloor \frac{F(u,v) + \left\lfloor \frac{Q(u,v)}{2} \right\rfloor}{Q(u,v)} \right\rfloor \tag{3.35}$$

Onde $\lfloor x \rfloor$ indica o maior inteiro menor que $x$. Quantizando a matriz DCT inteira deste exemplo, de acordo com (3.35) obtemos:

$$\mathbf{F}^* = \begin{bmatrix} 79 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.36}$$

Note que o processo de quantização e normalização produziu muitos coeficientes nulos.

Em seguida, os coeficientes DCT quantizados são reordenados em um formato unidimensional usando uma matriz zig-zag, onde o valor de cada elemento indica a posição que o elemento correspondente em $F^*(u,v)$ deve ocupar no vetor resultante. Esta matriz é definida como:

$$\begin{bmatrix} 0 & 1 & 5 & 6 & 14 & 15 & 27 & 28 \\ 2 & 4 & 7 & 13 & 16 & 26 & 29 & 42 \\ 3 & 8 & 12 & 17 & 25 & 30 & 41 & 43 \\ 9 & 11 & 18 & 24 & 31 & 40 & 44 & 53 \\ 10 & 19 & 23 & 32 & 39 & 45 & 52 & 54 \\ 20 & 22 & 33 & 38 & 46 & 51 & 55 & 60 \\ 21 & 34 & 37 & 47 & 50 & 56 & 59 & 61 \\ 35 & 36 & 48 & 49 & 57 & 58 & 62 & 63 \end{bmatrix} \tag{3.37}$$

Para o nosso exemplo, o resultado da reordenação é:

$$79 \quad 0 \quad -2 \quad -1 \quad -1 \quad -1 \quad 0 \quad 0 \quad -1 \quad \text{EOB} \tag{3.38}$$

onde o símbolo EOB indica o fim do bloco, ou seja, que todas as amostras seguintes são iguais a zero. O valor DC e os coeficientes AC na seqüência 1-D (3.38) são codificados utilizando tabelas de Hufman separadas, locais ou globais, que são transmitidas ao receptor como parte do cabeçalho da imagem. Como mencionado anteriormente, ao invés de codificar o valor DC diretamente, calculamos primeiro a diferença entre este valor e o valor DC do bloco anterior. Em seguida, codificamos o resultado utilizando uma tabela do código Huffman semelhante à Tabela 3.5. O comprimento máximo da palavra código para cada símbolo é restrito a 16 bits e nenhuma palavra-código formada apenas por 1's pode existir. Na Tabela 3.5 apresentamos os símbolos cujas palavras-código têm comprimento menor que 10.

| número de zeros | categoria | comprimento do código | palavra código |
|---|---|---|---|
| 0 | 1 | 2 | 00 |
| 0 | 2 | 1 | 01 |
| 0 | 4 | 3 | 100 |
| 0 | 5 | 4 | 1011 |
| ... | ... | ... | ... |
| 1 | 1 | 4 | 1100 |
| 1 | 2 | 6 | 111001 |
| ... | ... | ... | ... |
| 2 | 1 | 5 | 11011 |
| 2 | 2 | 8 | 11111000 |
| ... | ... | ... | ... |
| 4 | 1 | 6 | 111011 |
| 5 | | | 1111010 |
| 6 | | | 1111011 |
| 7 | | | 11111001 |
| ... | ... | ... | ... |
| Fim do bloco (EOB) | | 4 | 1010 |

Tabela 3.5: Exemplo de tabela para o código de Huffman.

Para o nosso exemplo, o primeiro coeficiente AC não-nulo tem valor $-2$ e está separado do coeficiente nulo anterior por uma distância de 1. De acordo com a Tabela 3.4, vemos que $-2$ se encontra na categoria $k = 2$. Da Tabela 3.5, vemos que um coeficiente da categoria 2 precedido de 1 zero é representado pela palavra código de Huffman (111001). Esta palavra código é então acrescida do bit de sinal (0 para coeficientes negativos e 1 para os positivos) e dos $k - 1$ bits adicionais, que servem para identificar a magnitude do coeficiente dentro da sua categoria. Desta forma, a cadeia de bits gerada pela concatenação das palavras-código é formada da seguinte forma:

$$\underset{\substack{\text{palavra código de Huffman} \\ \text{da diferença DC}}}{} \quad 11100101 \quad 000 \quad 000 \quad 000 \quad 110110 \quad 1010 \tag{3.39}$$

Assumindo que 8 bits foram utilizados para codificar a diferença DC, um total de 35 bits são necessários para codificar o bloco de $8 \times 8$ píxeis do exemplo. A taxa de bits resultante é igual a 35 bits/ 64 píxeis = 0.55 bits/ píxeis. No receptor, os coeficientes quantizados são reconstruídos pelo procedimento de decodificação de Huffman e a seguir são desnormalizados de acordo com a seguinte relação:

$$\widehat{F}(u,v) = F^*(u,v)\, Q(u,v) \tag{3.40}$$

Calculando-se a transformada inversa do bloco desnormalizado, obtemos o bloco reconstruído:

$$\widehat{\mathbf{f}} = \begin{bmatrix} 144 & 146 & 149 & 152 & 154 & 156 & 156 & 156 \\ 148 & 150 & 152 & 154 & 156 & 156 & 156 & 156 \\ 155 & 156 & 157 & 158 & 158 & 157 & 156 & 155 \\ 160 & 161 & 161 & 162 & 161 & 159 & 157 & 155 \\ 163 & 163 & 164 & 163 & 162 & 160 & 158 & 156 \\ 163 & 163 & 164 & 164 & 162 & 160 & 158 & 157 \\ 160 & 161 & 162 & 162 & 162 & 161 & 159 & 158 \\ 158 & 159 & 161 & 161 & 162 & 161 & 159 & 158 \end{bmatrix} \tag{3.41}$$

Os erros introduzidos no bloco reconstruído devido a compressão são dados por:

$$\mathbf{e} = \begin{bmatrix} -5 & -2 & 0 & 1 & 1 & -1 & -1 & -1 \\ -4 & 1 & 1 & 2 & 3 & 0 & 0 & 0 \\ -5 & -1 & 3 & 5 & 0 & -1 & 0 & 1 \\ -1 & 0 & 1 & -2 & -1 & 0 & 2 & 4 \\ -4 & -3 & -3 & -1 & 0 & -5 & -3 & -1 \\ -2 & -2 & -3 & -3 & -2 & -3 & -1 & 0 \\ 2 & 1 & -1 & 1 & 0 & -4 & -2 & -1 \\ 4 & 3 & 0 & 0 & 1 & -3 & -1 & 0 \end{bmatrix} \tag{3.42}$$

onde $e(j,k) = f(j,k) - \widehat{f}(j,k)$. O RMSE resultante da codificação deste bloco é:

$$RMSE = \sqrt{\frac{1}{64}\sum_{j=0}^{7}\sum_{k=0}^{7} e^2(j,k)} = 2.26 \tag{3.43}$$

### 3.4.4 Codificação por Sub-Banda

A codificação por sub-banda também pode ser interpretada como uma alternativa aos esquemas apresentados nas Figuras 3.1 e 3.2. Este tipo de codificação não é nada mais que uma classe da codificação por transformada [Rabbani]. Tradicionalmente, entretanto, os codificadores baseados em transformadas lineares são divididos em duas classes: codificadores por transformada e codificadores por sub-banda. Esta distinção é devido apenas à natureza dos métodos computacionais empregados. Na codificação por transformada, obtemos como saída um conjunto de coeficientes. Na codificação por sub-bandas, obtemos um conjunto de sub-bandas.

As transformadas por sub-banda correspondem a sistemas formados por bancos de filtros, contendo um estágio de decomposição e outro de síntese, conforme pode ser visto na Figura 3.12. A notação $k_i \downarrow$, utilizada na Figura 3.12, representa a operação de dizimação da seqüência por um fator $k_i$, ou seja, a cada $k_i$ amostras da seqüência uma é preservada e as outras $k_i - 1$

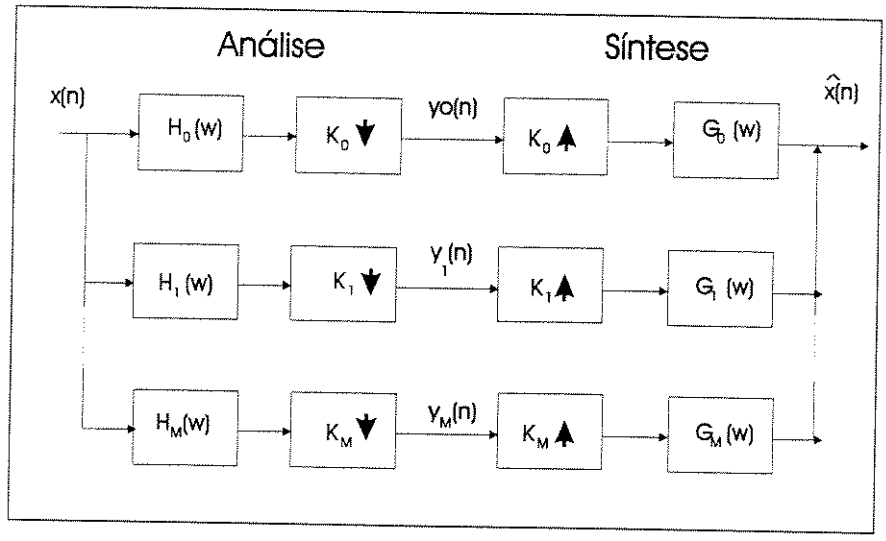



Figura 3.12: Banco de filtros de um sistema análise/ síntese

são eliminadas. Da mesma forma, $k_i \uparrow$ representa a operação de interpolação de zeros por um fator $k_i$, ou seja, $k_i - 1$ zeros são inseridos entre cada par de amostras. Assumimos que $k_i$ é um número inteiro e divisor de $N$ (número de amostras do sinal). $M$ é o número de filtros do sistema.

Estes sistemas são denominados sistemas A/S. Neles, o sinal de entrada é filtrado por um conjunto de filtros passa-faixa cujas respostas são subamostradas, dando origem a um conjunto de sub-bandas. Este processo de filtragem e subamostragem é denominado estágio de análise. A seguir, os sinais das sub-bandas são quantizados e codificados. Diferentes taxas de bits ou mesmo diferentes técnicas de codificação podem ser utilizadas para cada sub-banda, tirando proveito das propriedades da sub-banda e/ou permitindo que os erros de codificação sejam distribuídos pelas sub-bandas de uma maneira visualmente ótima.

A reconstrução da imagem original é obtida interpolando com zeros as sub-bandas decodificadas, aplicando os filtros apropriados e adicionando as sub-bandas reconstruídas. Este estágio é denominado estágio de reconstrução ou síntese. Note que a simples formação de sub-bandas não gera nenhuma compressão. O sinal decomposto possui a mesma quantidade de bits do sinal original. A vantagem desta abordagem é que as sub-bandas podem ser codificadas de forma mais eficiente que a imagem original.

### Notação Matricial

A notação matricial é uma alternativa para a notação no domínio da freqüência [Oppenheim]. Uma imagem de tamanho finito amostrada por uma grade discreta, pode ser representada por um vetor coluna $\mathbf{x}$ de comprimento finito e que corresponde a um ponto no espaço $\mathbf{R}^N$, onde $N$ é o comprimento deste vetor $\mathbf{x}$. Uma transformação linear da imagem corresponde à multiplicação de $\mathbf{x}$ por uma matriz $\mathbf{M}$ com $N$ colunas.

Como cada estágio de síntese e de análise do sistema da Figura 3.12 corresponde a uma transformação linear, podemos representar a mesma transformação utilizando uma notação

matricial. Utilizando a definição de convolução e assumindo, por simplicidade, que estamos lidando com um sistema unidimensional, podemos representar o sinal intermediário e saída do sistema A/S da seguinte forma:

$$y_i(m) = \sum_{l=0}^{N-1} x(l)\, h_i(k_i \cdot m - l); \qquad 0 \leq i \leq M \text{ e } 0 \leq m \leq \frac{N}{k_i} \tag{3.44}$$

$$\widehat{x}(n) = \sum_{i=0}^{M-1} \sum_{m=0}^{\frac{N}{k_i}-1} y_i(m)\, g_i(n - k_i \cdot m) \tag{3.45}$$

onde as posições $(k_i m - l)$ do filtro e $(n - k_i m)$ da imagem são calculadas em álgebra módulo $N$. As matrizes $\mathbf{H}$ e $\mathbf{G}$ são construídas da seguinte forma:

$$\mathbf{H} = \begin{bmatrix} h_0(0) & h_0(k_0) & \ldots & h_1(0) & h_1(k_1) & \ldots \\ h_0(-1) & h_0(k_0-1) & \ldots & h_1(-1) & h_1(k_1-1) & \ldots \\ h_0(-2) & h_0(k_0-2) & \ldots & h_1(-2) & h_1(k_1-2) & \ldots \\ \ldots & h_0(k_0-3) & \ldots & \ldots & h_1(k_1-3) & \ldots \\ h_0(2) & \ldots & \ldots & h_1(2) & \ldots & \ldots \\ h_0(1) & \ldots & \ldots & h_1(1) & \ldots & \ldots \end{bmatrix} \tag{3.46}$$

e

$$\mathbf{G} = \begin{bmatrix} g_0(0) & g_0(k_0) & \ldots & g_1(0) & g_1(k_1) & \ldots \\ g_0(1) & g_0(k_0+1) & \ldots & g_1(1) & g_1(k_1+1) & \ldots \\ g_0(2) & g_0(k_0+2) & \ldots & g_1(2) & g_1(k_1+2) & \ldots \\ \ldots & g_0(k_0+3) & \ldots & \ldots & g_1(k_1+3) & \ldots \\ g_0(-2) & \ldots & \ldots & g_1(-2) & \ldots & \ldots \\ g_0(-1) & \ldots & \ldots & g_1(-1) & \ldots & \ldots \end{bmatrix} \tag{3.47}$$

As colunas de $\mathbf{G}$, compostas por cópias dos núcleos dos filtros de síntese, deslocados por incrementos $k_i$, são as funções da base da transformação. Da mesma forma, as colunas de $\mathbf{H}$, compostas por cópias dos núcleos dos filtros de análise invertidos no tempo, deslocados por incrementos $k_i$, são as funções de amostragem da transformação.

As expressões (3.44) e (3.45) podem então ser expressas em notação matricial da seguinte forma:

$$\mathbf{y} = \mathbf{H}^t \mathbf{x} \tag{3.48}$$

e

$$\widehat{\mathbf{x}} = \mathbf{G}\mathbf{y} \tag{3.49}$$

Combinando estas duas equações, obtemos:

$$\widehat{\mathbf{x}} = \mathbf{G}\mathbf{H}^t \mathbf{x} \tag{3.50}$$

onde $\mathbf{y}$ e $\widehat{\mathbf{x}}$ são vetores de comprimento $N$, $\mathbf{H}^t$ é a transposta da matriz $\mathbf{H}$.

Da discussão, fica claro que podemos expressar qualquer sistema linear A/S na forma matricial. O inverso também é verdadeiro: há um sistema A/S linear correspondente a qualquer transformada linear e sua inversa, definidas por uma matriz inversível $\mathbf{M}$.

Uma das vantagens da notação matricial é a facilidade com que podemos expressar a inversabilidade da transformada. Da equação (3.49), vemos que para que o sistema A/S reconstrua perfeitamente o sinal original, as matrizes correspondentes devem obedecer a seguinte relação:

$$\mathbf{G}\mathbf{H}^t = \mathbf{I} \tag{3.51}$$

onde $\mathbf{I}$ é a matriz identidade. Se $\mathbf{H}$ tem posto $N$ e é quadrada, podemos escolher como matriz síntese:

$$\mathbf{G} = \left(\mathbf{H}^{-1}\right)^t \tag{3.52}$$

que também é quadrada e possui posto igual a $N$. Assim, expressar a transformada inversa na notação matricial é simples e será bastante útil na análise dos sistemas A/S. Fica bastante claro que $\mathbf{G}$ e $\mathbf{H}$ podem ser intercambeados, o que corresponde a usar as funções da base como funções amostrais e vice-versa.

Se a matriz $\mathbf{H}$ tem posto $N$, mas não é quadrada, ainda podemos construir um sistema de reconstrução perfeita escolhendo $\mathbf{G}$ como sendo a pseudo-inversa de $\mathbf{H}$ :

$$\mathbf{G} = \left(\mathbf{H}\mathbf{H}^t\right)^{-1}\mathbf{H} \tag{3.53}$$

Se $\mathbf{H}$ é quadrada, a equação (3.53) se reduz a (3.52). Da mesma maneira, se utilizarmos a matriz $\mathbf{G}$ de posto $N$ como matriz de análise, podemos escolher como matriz de síntese a matriz $\mathbf{H} = \left(\mathbf{G}\mathbf{G}^t\right)^{-1}\mathbf{G}$.

### Transformadas Ortogonais

Como já mencionado, a ortogonalidade é uma propriedade importante para a codificação de imagens. Uma matriz $\mathbf{M}$, correspondente a uma transformação ortogonal, é uma matriz quadrada de posto cheio com a propriedade:

$$\mathbf{M}\mathbf{M}^t = \mathbf{M}^t\mathbf{M} = \mathbf{I} \tag{3.54}$$

Isto significa que o produto interno de duas colunas distintas deve ser nulo e o produto interno de uma coluna com ela própria deve ser igual a unidade. Como já observado, as colunas da matriz transformação são formadas pelas funções da base desta transformação. Portanto, estas funções da base são ortogonais.

A condição de ortogonalidade impõe algumas restrições ao sistema A/S. Como a matriz transformação $\mathbf{M}$ deve ser quadrada, o número de amostras do sinal transformado deve ser igual a $N$, que é o número de amostras da imagem original. O tamanho do sinal original é, portanto, preservado. Para o sistema A/S isto significa que:

$$\sum_{i=0}^{M-1} \frac{1}{k_i} = 1 \tag{3.55}$$

onde assumimos que $N$ é divisível por todos os $k_i$.

Uma segunda restrição, ainda mais importante, imposta ao sistema A/S pela ortogonalidade é dada pela combinação das equações (3.51) e (3.54):

$$\mathbf{G} = \mathbf{H} \tag{3.56}$$

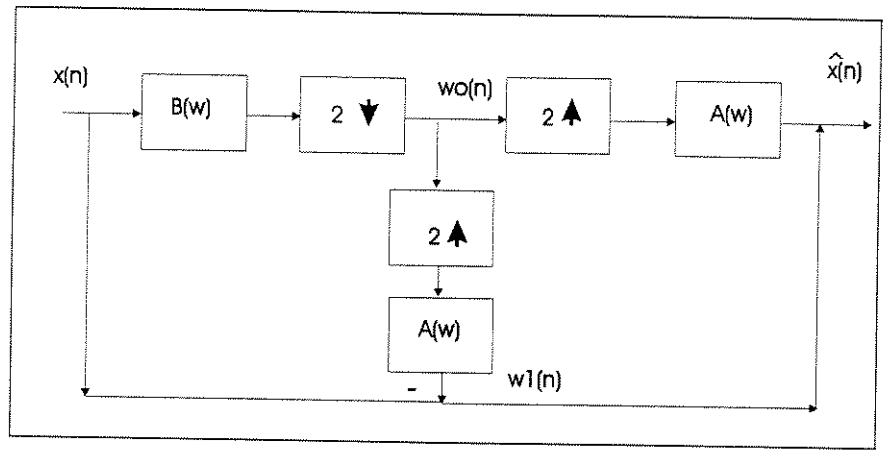



Figura 3.13: Diagrama de blocos do esquema de decomposição piramidal laplaciana para 1 nível.

Se considerarmos as relações entre os filtros $h$ e $g$ e as matrizes $\mathbf{G}$ e $\mathbf{H}$, descritas pelas expressões (3.46) e (3.47), a equação (3.56) implica que os filtros devem obedecer a seguinte relação:

$$g_i(n) = h_i(-n), \qquad \text{para todo } i \tag{3.57}$$

Em outras palavras, os filtros de síntese de uma transformada ortogonal são versões invertidas no tempo dos filtros de análise.

### Técnicas de Decomposição em Sub-Bandas

Existem diversas técnicas de decomposição em sub-bandas [Lim] [Jain88] [Gharavi]. Descreveremos nesta seção, com o objetivo de exemplificação, duas destas técnicas: a pirâmide laplaciana e a transformada wavelet.

**A Pirâmide Lapalaciana** Uma das técnicas para decomposição em sub-bandas de comprimento em oitavas foi desenvolvido por Burt [Burt81] e aplicado na codificação de imagens por Burt e Adelson [Burt83]. Eles utilizaram um cascateamento piramidal de pequenos filtros gaussianos para criar uma decomposição redundante, que foi denominada pirâmide laplaciana. Um sistema unidimensional para construção de um nível desta pirâmide é apresentado na Figura 3.13. O sinal é filtrado por um passa-baixa, $B(\omega)$, e depois é subamostrado gerando a sub-banda de baixas freqüências $W_0(\omega)$. Uma sub-banda de altas freqüências, $W_1(\omega)$, é formada interpolando $W_0(\omega)$ por um fator de 2, filtrando o resultado através de $A(\omega)$ e subtraindo o resultado do sinal original. O sinal é reconstruído interpolando $W_0(\omega)$ por um fator de 2, filtrando com $A(\omega)$ e adicionando o resultado a $W_1(\omega)$. Esta reconstrução é exata, independente da escolha dos filtros $A(\omega)$ e $B(\omega)$. A pirâmide completa é construída aplicando o algoritmo recursivamente às sub-bandas de baixas freqüências.

Além de ser adequada à compressão de dados, a natureza de múltiplas escalas da pirâmide a torna particularmente útil para transmissões progressivas. Transmissão progressiva é um processo pelo qual uma imagem é enviada por partes, ou em camadas. Ou seja, uma versão de

baixa resolução da imagem é enviada através de um canal de baixa capacidade, de maneira a garantir uma qualidade básica. Os detalhes da imagem, ou informações de resolução mais alta, são enviados de uma maneira gradual. No caso de uma pirâmide, este processo é facilmente implementado através do envio dos coeficientes da transformada ordenados da resolução mais baixa até a resolução mais alta.

Em comparação com outras transformadas por sub-banda, o esquema da pirâmide laplaciana pode ser visto como um sistema A/S de três bandas, $W_0(\omega)$, $Y_1(\omega)$ e $Y_2(\omega)$. As duas bandas $Y_1(\omega)$ e $Y_2(\omega)$ são, na verdade, sub-divisões do sinal $W_1(\omega)$, sendo que $Y_1(\omega)$ contém as amostras pares e $Y_2(\omega)$ contém as amostras ímpares. Os fatores de subamostragem são iguais à $k = 2$ para todos os três ramos do sistema A/S, produzindo assim uma representação redundante por um fator de 3/2. Os filtros apropriados para o sistema A/S são definidos em termos dos filtros originais $A(\omega)$ e $B(\omega)$ através das seguintes relações:

$$\begin{array}{ll} H_0(\omega) = B(\omega), & G_0(\omega) = A(\omega) \\ H_1(\omega) = \frac{1}{2}[1 - B(\omega)A(\omega) - B(\omega)A(\omega+\pi)], & G_1(\omega) = 1 \\ H_2(\omega) = \frac{e^{j\omega}}{2}[1 - B(\omega)A(\omega) + B(\omega)A(\omega+\pi)], & G_2(\omega) = e^{-j\omega} \end{array} \tag{3.58}$$

Note que se $A(\omega)$ e $B(\omega)$ forem filtros passa-baixas, as funções de amostragem terão uma banda limitada e as funções da base terão uma banda mais larga. Como o sistema resultante viola a restrição (3.57), a transformada é claramente não-ortogonal.

Burt e Adelson [Burt83] construíram pirâmides laplacianas bidimensionais utilizando o mesmo algoritmo, mas com filtros passa-baixas 2-D separáveis. As pirâmides bidimensionais podem ser reformuladas como sistemas A/S de 5 bandas, onde cada banda é subamostrada por um fator de 2, tanto horizontalmente como verticalmente.

A pirâmide Laplaciana apresenta algumas desvantagens para codificação de imagens. Uma das maiores desvantagens é que os erros de quantização das sub-bandas de freqüências mais altas não permanecem nestas sub-bandas. Ao invés disso, eles são visíveis na imagem reconstruída como ruído de banda larga. Assim como na transformada de Gabor, a não-ortogonalidade da transformada é uma fonte de dificuldades. Além disso, o conjunto de coeficientes da transformada piramidal é redundante, implicando em um aumento por um fator de $\frac{4}{3}$ no número de amostras da imagem original. Finalmente, as funções da base não são orientadas e, portanto, não irão extrair a redundância estrutural orientada encontrada em imagens. Apesar destas desvantagens, esta técnica tem sido utilizada em outras aplicações como codificação de vídeo com compensação de movimentos, onde a sua redundância a torna menos sensível a erros. [Adelson]

**A Transformada Wavelet** A transformada wavelet também é uma transformada por sub-bandas. A principal característica desta técnica é a utilização de filtros QMFs e CQFs [Herley] [Vetterli92]. Estes filtros devem satisfazer às condições impostas pelos teoremas 2-3 e 2-4, apresentados no capítulo 2. Os filtros utilizados pela transformada wavelet estão associados às funções escala e wavelet dos estágios de análise e síntese e, conforme já discutido, influenciam diretamente no desempenho do codificador.

Devido a sua grande capacidade de concentração de energia, a TW permite obter taxas de compressão maiores que asquelas oferecidas pela DCT. Além disso, a transformada wavelet não apresenta efeito de blocagem, muito comum em codificações por transformada, como a DCT.

No próximo capítulo, faremos um estudo da utilização desta transformada na compressão de imagens, apresentando alguns resultados obtidos a partir de simulações utilizando três modelos de compressão baseados na transformada wavelet.

# Capítulo 4

# Compressão de Imagens Utilizando a T.W.

Uma das aplicações da transformada wavelet que tem apresentado um grande sucesso é a compressão de imagens [Antonini] [Franca] [Brislawn] [Shapiro]. Além de apresentar uma grande capacidade de concentração de energia, a transformada wavelet possui uma forte relação com o sistema visual humano, o que permite obter uma alta taxa de compressão com menores níveis visíveis de degradação.

Uma das propriedades mais atrativas desta transformada é a sua flexibilidade com relação à duração das wavelets, o que permite que sinais com conteúdo de freqüência variado, como as imagens, possam ser bem representados com um número reduzido de funções. Além disso, a utilização da transformada wavelet diminui sensivelmente o efeito de blocagem, que é um dos principais problemas do JPEG. Wavelets mais longas podem ser utilizadas para representar, por exemplo, as regiões planas de uma imagem (baixa freqüência), enquanto que as wavelets mais estreitas são utilizadas para representar as regiões de textura (alta freqüência) [Lim] [Jain88].

Neste capítulo apresentaremos um estudo sobre a estrutura de um esquema básico de compressão baseado na transformada wavelet, detalhando cada um dos estágios deste esquema. Apresentaremos também alguns modelos inéditos, baseados em esquemas já existentes na literatura [Shapiro] [Antonini], assim como os resultados das suas simulações. Não faremos, entretanto, nenhum estudo sobre a complexidade computacional deste modelos, uma vez que o nosso interesse se restringe ao estudo da capacidade de compressão das wavelets.

## 4.1 Sistema Básico de Compressão Baseado na TW

O diagrama em blocos de um sistema básico de compressão de imagens baseado na transformada wavelet é apresentado na Figura 4.1. Este sistema pode ser dividido em três estágios: a transformada wavelet, a quantização e a codificação entrópica [Strang96]. Cada um destes estágios tem grande importância no desempenho do compressor e o seu projeto deve levar em conta as características da imagem, as taxas de compressão desejadas e as limitações inerentes ao processo de implementação.

No primeiro estágio, que corresponde ao mapeador do esquema de compressão apresentado na Figura 4.1, calcula-se a transformada wavelet bidimensional da imagem. Como já discutido,

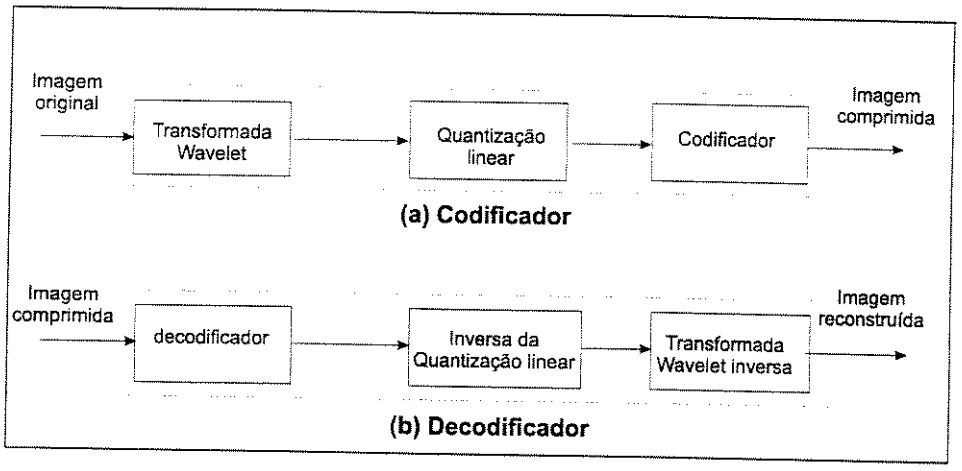



Figura 4.1: Diagrama de blocos do esquema de compressão baseado na Transformada Wavelet.

a imagem pode ser decomposta em uma, duas ou mais camadas. Além disso, existe uma grande variedade de famílias de wavelets que podem ser utilizadas. Cada uma delas apresenta características específicas como, por exemplo, suavidade, simetria, ortogonalidade, etc.

No segundo estágio, os coeficientes da transformada wavelet são quantizados. Neste trabalho, por uma questão de simplicidade, utilizaremos apenas quantizadores escalares.

No terceiro estágio, os coeficientes da transformada, já quantizados, são codificados entropicamente. Entre os códigos existentes, os mais utilizados são o código de Huffman (*run-length)* e o código aritmético. Neste trabalho utilizaremos o código *run-length* em todos os esquemas de compressão, porque acreditamos que este é um código simples e eficiente. O código *run-length* é um código unidimensional de comprimento variável, que apresenta bons resultados na codificação de dados que possuam grandes seqüências de zeros, como é o caso das sub-bandas de freqüências mais altas da transformada wavelet.

Entretanto, para que uma imagem possa ser codificada utilizando o código *run-length,* os seus píxeis (coeficientes wavelet) devem ser reagrupados de maneira a formar um vetor. A maneira como os coeficientes são agrupados é muito importante e pode influenciar diretamente no desempenho da codificação.

Nas próximas seções, faremos uma descrição de cada um destes estágios. Em seguida, apresentaremos alguns esquemas de compressão propostos. Para facilitar a análise dos resultados, as tabelas e imagens resultantes das simulações são apresentadas na última seção do capítulo. A imagem utilizada em todas as simulações foi a Lena, apresentada na Figura 4.2.

As tabelas e gráficos de desempenhos deste capítulo se encontram no apêndice A e as imagens reconstruídas no apêndice B.

## 4.2 A Transformada Wavelet

Uma das características que uma transformada deve apresentar para que possa ser utilizada em um esquema de compressão é uma grande capacidade de concentração da energia. A transformada deve concentrar a maior parte da energia da imagem em uma pequena porção dos

Figura 4.2: Imagem utilizada nas simulações: Lena.

coeficientes. Esta característica permite que muitos coeficientes sejam eliminados, obtendo-se assim uma primeira compressão.

A transformada wavelet apresenta uma capacidade de concentração de energia superior ao da transformada de Fourier e da DCT. Além disso, a forte relação da transformada wavelet com o mecanismo de visão humana permite que componentes da imagem de menor importância visual sejam identificadas e eliminadas mais facilmente [Strang96].

Como já enfatizado no Capítulo 2, as famílias de wavelets utilizadas no processamento de imagens e, especificamente, na compressão de imagens, devem apresentar certas características como filtros de comprimento finito e fase linear, suavidade, regularidade e um grande número de momentos desvanecentes. Ao escolher uma família de wavelets devemos levar em conta todos estes fatores [Antonini].

Além da família de wavelets utilizada, um outro parâmetro que pode interferir no desempenho do esquema de compressão é o número de camadas de decomposição/síntese. A taxa de compressão aumenta com o número de camadas, embora este aumento comece a saturar quando o número de camadas excede 4 [Strang96]. Por outro lado, ao aumentarmos o número de camadas, aumentamos também a complexidade do sistema. Como conseqüência deste compromisso entre a complexidade do sistema e o ganho em taxa de compressão, geralmente são utilizadas de duas a quatro camadas.

## 4.3 A Quantização

Este é um estágio muito importante do esquema de compressão, pois é nele que os distorções são introduzidas na imagem. Como cada sub-banda da transformada wavelet possui características próprias, melhores resultados podem ser obtidos se cada sub-banda for quantizada de forma

independente [Rabbani]. A transformada wavelet possui uma grande correlação com o sistema visual humano, o que permite que informações visualmente pouco importantes sejam mais facilmente identificadas e descartadas. Obviamente, quanto maior a quantidade de informação descartada, maior a compressão e a degradação da imagem resultante.

Nesta seção, descreveremos três tipos de quantizadores lineares: um quantizador com vetor de quantização pré-estabelecido, um quantizador com alocação ótima de bits e um quantizador com alocação ponderada de bits. Para medir o erro introduzido nas imagens reconstruídas para cada quantizador, utilizamos a medida de erro PSNR, definida no Capítulo 2. Apresentamos também algumas das imagens reconstruídas, para que se possa visualizar o tipo de distorção introduzida a cada configuração.

### 4.3.1 Quantizador com Vetor de Quantização Fixo

Este primeiro quantizador é o mais simples de todos. A sua construção baseia-se na escolha de um passo diferente para cada sub-banda. Cria-se um vetor $\mathbf{q}$, denominado vetor de quantização, cujos elementos $(q(i))$ têm valores diretamente proporcionais aos passos de quantização utilizados em cada sub-banda.

A escolha do vetor irá determinar a quantidade de erros presentes na imagem reconstruída e limitar a taxa de compressão possível. Quanto maior for o valor de $q(i)$, maior será o tamanho do passo e mais grosseira a quantização. Um exemplo de um possível vetor de quantização para uma decomposição em três camadas é apresentado a seguir:

$$\mathbf{q} = \mathbf{q}_4 = [4\ 16\ 16\ 16\ 32\ 32\ 32\ 64\ 64\ 64] \tag{4.1}$$

Uma vez fixado $\mathbf{q}$, cada coeficiente $x_i(k,l)$ pertencente a i-ésima sub-banda $\mathbf{X}_i$, é quantizado através da seguinte relação [Rabbani]:

$$y_i(k,l) = \lfloor (2 \cdot x_i(k,l) + \lfloor q(i)/2 \rfloor) \cdot q(i) \rfloor, \quad i = 1, 2, \ldots (3 \cdot N + 1). \tag{4.2}$$

onde $y_i(k,l)$ é o coeficiente quantizado resultante, $q(i)$ é a i-ésima componente do vetor de quantização $\mathbf{q}$ e $N$ é o número de camadas do esquema de decomposição. A notação $\lfloor x \rfloor$ representa o menor inteiro maior ou igual a $x$. Na Figura 4.3 apresentamos o modelo de ordenação para uma decomposição em três camadas. Este modelo pode ser facilmente generalizado para um número diferente de camadas.

Para se obter maiores taxas de compressão, sem modificar o vetor de quantização $\mathbf{q}$, basta multiplicar este vetor por um fator $\rho$, denominado fator de quantização, o qual assume valores inteiros maiores ou iguais a 1. Quanto maior o valor de $\rho$, maior a compressão e, conseqüentemente, maior a distorção.

A operação inversa à quantização é definida pela seguinte relação:

$$\mathbf{Z}_i = q(i) \cdot \mathbf{Y}_i, \qquad i = 1, 2, \ldots (3 \cdot N + 1). \tag{4.3}$$

onde $\mathbf{Z}_i$ é a i-ésima sub-banda reconstruída e $\mathbf{Y}_i$ é a i-ésima sub-banda quantizada. Esta operação consiste basicamente na decodificação dos índices de quantização em seus respectivos níveis de reconstrução, conforme definido no Capítulo 2. A operação de inversão da quantização não recupera, entretanto, as perdas introduzidas pelo processo quantização.

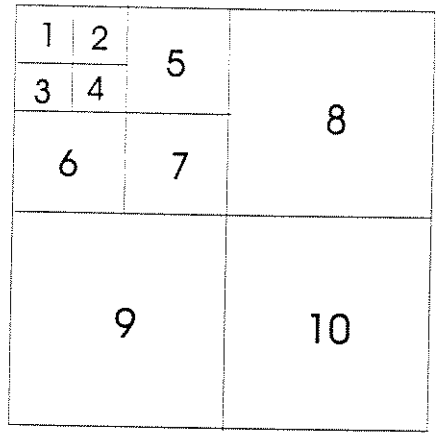



Figura 4.3: Subimagens resultantes de uma decomposição em três camadas através da Transformada Wavelet.

Realizamos algumas experiências utilizando este tipo de quantização. As imagens reconstruídas, obtidas após a quantização da imagem Lena utilizando o vetor $\mathbf{q}$ em (4.2), com fatores de quantização 1 e 4, são apresentadas nas Figuras B.3-(a), B.3-(b) do apêndice B. Os valores de PSNR correspondentes a estas imagens podem ser encontrados na Tabela A.4 do apêndice A. Podemos constatar que com o aumento do fator de quantização, a qualidade da imagem piora e diminui o valor do PSNR.

### 4.3.2 Quantização com Alocação de Bits

O segundo quantizador utiliza a estatística das sub-bandas como base para o processo de quantização. A cada sub-banda da transformada wavelet é alocada uma certa quantidade de bits para sua representação, de acordo com as suas características estatísticas e a taxa de bits total que se deseja obter.

O processo de alocação ótima de bits é realizada de acordo com um critério de minimização [Jain88] previamente escolhido. Existe um algoritmo de alocação de bits que permite fazer uma distribuição de bits entre as sub-bandas, de acordo com o valor das suas variâncias [Strang96]. Discutiremos, rapidamente, este algoritmo. Sejam:

- $N$ o número total de bits na imagem original.
- $M$ o número total de sub-bandas.
- $R$ a taxa de bits desejada.
- $\alpha_k = N_k/N$ o tamanho relativo da $k$-ésima sub-banda.
- $\mathbf{b} = (b_1, ..., b_M)$ o vetor contendo o número de bits alocados à cada sub-banda.
- $\epsilon_k$ o índice de desempenho do quantizador.
- $\sigma_k^2$ a variância da $k$-ésima sub-banda.

- $\omega_k$ o fator de ponderação da sub-banda. (Maiores detalhes serão dados na próxima seção. Por enquanto, assume-se que $\omega_k = 1, \forall k$.)

Definimos o erro de quantização como:

$$D(\mathbf{b}) = \sum_{k=1}^{M} \alpha_k \cdot \omega_k \cdot \epsilon_k^2 \cdot 2^{-2b_k} \cdot \sigma_k^2 \tag{4.4}$$

e a taxa de bits como:

$$R(\mathbf{b}) = \sum_{k=1}^{M} \alpha_k \cdot b_k \tag{4.5}$$

O problema da alocação de bits consiste em minimizar $D(\mathbf{b})$, mantendo fixa a taxa de bits total $R(\mathbf{b}) = R_C$. Este problema pode ser resolvido utilizando um multiplicador de Lagrange $\lambda$ [Jain88]:

$$\min\{D(\mathbf{b}) + \lambda \cdot R(\mathbf{b})\} = \min\left\{\sum_{k=1}^{M} \alpha_k \cdot \left(\omega_k \cdot \epsilon_k^2 \cdot 2^{-2b_k} \cdot \sigma_k^2 + \lambda \cdot b_k\right)\right\} \tag{4.6}$$

Assumindo que $\epsilon_k = \epsilon$ e definindo $\tilde{\lambda} = \lambda/\epsilon^2$, a expressão (4.6) se reduz a:

$$\min\left\{\sum_{k=1}^{M} \alpha_k \cdot \epsilon^2 \left(\omega_k \cdot 2^{-2b_k} \cdot \sigma_k^2 + \tilde{\lambda} \cdot b_k\right)\right\} \tag{4.7}$$

Diferenciando (4.7) em relação a $b_k$, obtemos a seguinte solução [Strang96]:

$$b_k = \frac{1}{2}\log_2\left(\frac{(2\log_e 2) \cdot \omega_k \cdot \sigma_k^2}{\tilde{\lambda}}\right) \tag{4.8}$$

Substituindo (4.8) em (4.5), obtemos:

$$R_C = \sum_{k=1}^{M} \alpha_k \cdot b_k = \frac{1}{2}\sum_{k=1}^{M} \alpha_k \cdot \log_2\left(\frac{(2\log_e 2) \cdot \omega_k \cdot \sigma_k^2}{\tilde{\lambda}}\right) \tag{4.9}$$

onde $\tilde{\lambda}$ é definido por:

$$\tilde{\lambda} = 2^{\left\{\sum_{k=1}^{M} \alpha_k \cdot \log_2\left((2\log_e 2)\cdot\omega_k\cdot\sigma_k^2\right) - R_C\right\}} \tag{4.10}$$

Primeiramente, o valor de $\tilde{\lambda}$ é calculado utilizando-se as variâncias $\sigma_k^2$ das sub-bandas. Em seguida, o número de bits alocado a cada sub-banda é calculado através da expressão (4.8).

Em algumas situações, quando a taxa de bits desejada $R_C$ é muito pequena, $b_k$ pode assumir valores negativos para sub-bandas que possuam variância muito pequena. Como não faz sentido um número negativo de bits, as sub-bandas para as quais $b_k < 0$ são descartadas. Assim, o número de sub-bandas que serão transmitidas diminui e alguns parâmetros do algoritmo precisam ser modificados. Após os devidos ajustes, o algoritmo é reinicializado. Caso ainda existam valores de $b_k$ negativos, este procedimento deve ser repetido até a sua eliminação completa.

Resumindo, o algoritmo de alocação ótima de bits é implementado a partir dos seguintes passos:

1. Encontram-se os comprimentos em bits $b_k$
2. Cancelam-se todos os valores negativos de $b_k$.
3. Reduz-se, apropriadamente, o número de sub-bandas, ou seja, $M_{atual} = M_{anterior} - M_{canceladas}$.
4. Calcula-se o novo valor da taxa $R_C = R_{C,atual}$.
5. Repete-se o algoritmo até que todos os $b_k$ sejam não-negativos.

Este algoritmo deve ser repetido cada vez que uma imagem nova for codificada.

No caso de filtros ortogonais [Coifman], a relação (4.4) é verdadeira e o algoritmo é ótimo. Entretanto, para filtros de fase linear não-ortogonais (como, por exemplo, os filtros biortogonais), a energia do sinal não é preservada pela transformada e a relação (4.4) não é válida. O ruído de quantização introduzido nas sub-bandas é, neste caso, amplificado pelo banco de filtros de síntese.

No caso de filtros não-ortogonais, precisamos fazer algumas considerações antes de utilizarmos este algoritmo. Assumindo que os erros de quantização são descorrelatos, definimos uma função de distribuição de energia nas sub-bandas $B_k = \alpha_k \sum_n f_k^2(n)$, onde $f_k(n)$ representa a energia da banda $k$ e $n$ é o índice do elemento desta sub-banda. A variância do erro de reconstrução é dada, então, pela seguinte expressão [Strang96]:

$$\sigma^2 = \sum_{k=0}^{M-1} B_k \cdot \sigma_k^2 \tag{4.11}$$

onde $B_k$ geralmente varia de $0,9\alpha_k$ a $1,1\alpha_k$ para sistemas práticos. Para sistemas ortogonais $B_k = 1 \cdot \alpha_k$, e a expressão (4.4) é válida. É razoável, entretanto, assumir que $B_k \simeq 1 \cdot \alpha_k$ e utilizar (4.4) para todos os casos práticos. Neste trabalho utilizaremos esta aproximação.

Um vetor de quantização, semelhante ao do esquema anterior, é construído a partir dos valores dos $b_k$. Sejam $x_{\max}$ e $x_{\min}$ os valores máximo e mínimo das amplitudes dos coeficientes da transformada wavelet. Após a quantização, a $k$-ésima sub-banda possui $2^{b_k}$ níveis de amplitude. Os elementos do vetor de quantização são calculados através da seguinte expressão:

$$q(i) = \frac{x_{\max} - x_{\min}}{2^{b_i}} \tag{4.12}$$

Nas Figuras B.7 e B.8 são apresentadas as imagens reconstruídas para $R_C$ = 2, 4, 6 e 10. Na segunda coluna da Tabela A.8 são apresentados os valores de PSNR para $R_C$ = 0,5, 1, 2, 4, 6, 10 e 12. O parâmetro $R_C$ funciona apenas como uma base para a compressão, uma vez que a taxa de bits final só será conhecida após a codificação entrópica. O fator $a$, que será definido na próxima seção, tem valor igual a 1 para estas simulações.

### 4.3.3 Quantização com Ponderação das Sub-bandas

O terceiro tipo de quantizador é um aperfeiçoamento do segundo. É bem conhecido que uma minimização do erro quadrático médio não garante resultados visualmente ótimos [Jain88]. Os olhos humanos são menos sensíveis a perdas em altas freqüências do que a perdas em

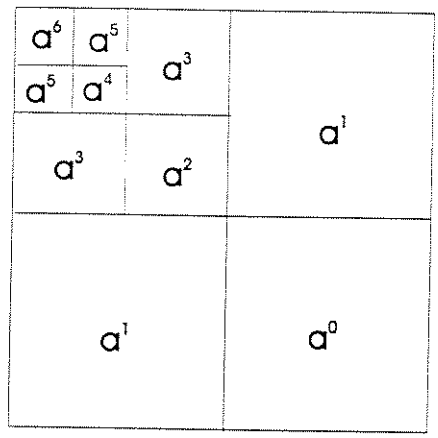



Figura 4.4: Modelo de distribuição dos pesos de ponderação para as sub-bandas de uma decomposição wavelet em três camadas.

médias e baixas freqüências, fator que não foi diretamente considerado nas seções anteriores. O desempenho do quantizador pode, então, melhorar se introduzirmos aspectos do sistema visual humano no modelo de alocação de bits. Isto é feito através da ponderação do ruído de quantização de cada sub-banda, de acordo com a sensibilidade do sistema visual humano.

Apresentamos aqui um esquema simples e eficaz para ponderação do ruído de quantização. Neste, mais bits são alocados às bandas de baixas e médias freqüências [Strang96]. O algoritmo de alocação de bits é o mesmo utilizado na seção anterior, porém os valores atribuídos aos pesos de ponderação das sub-bandas assumem valores diferentes de 1. O peso $\omega_k$, correspondente à $k$-ésima sub-banda, assume um valor igual à $a^j$, que irá depender do conteúdo em freqüência, ou seja, do índice desta sub-banda. Definimos $a$ como um número real maior que 1.

Na Figura 4.4 apresentamos um modelo de distribuição dos pesos $\omega_k$ para uma decomposição wavelet em três camadas. Para este caso, o peso de ponderação da primeira banda é igual a $a^6$, o da segunda banda é igual a $a^5$, o da terceira banda é igual a $a^4$ e assim por diante. Este modelo pode ser facilmente generalizado para um número diferente de camadas. Para calcularmos o número de bits alocados a cada sub-banda, substituímos os valores de $\omega_k$ nas expressões (4.8), (4.9) e (4.10) e utilizamos o algoritmo apresentado na seção anterior.

Nas Figuras B.9 até B.14 são apresentadas as imagens reconstruídas para vários valores de $R_C$ e $a$. Podemos perceber que a qualidade visual das imagens reconstruídas melhorou bastante em relação às imagens referentes à $a = 1$, o que permite concluir que a utilização de valores de $a$ maiores que 1 suaviza as distorções nas imagens reconstruídas.

Na Tabela A.7 são apresentados os valores de PSNR para estas simulações. À medida que o valor de $a$ cresce, a taxa de bits cresce. No esquema anterior, a alocação de bits era feita de forma uniforme entre todas as sub-bandas, independentemente do conteúdo de freqüência e da sua importância visual. Neste esquema, mais bits são alocados às médias e baixas freqüência, que são visualmente mais importantes. A ponderação garante que as sub-bandas sejam comprimidas levando em conta a sua importância visual. Porém, para $R_C = 12$ a taxa de bits será suficientemente alta para que a alteração no valor de $a$ não seja significativa.

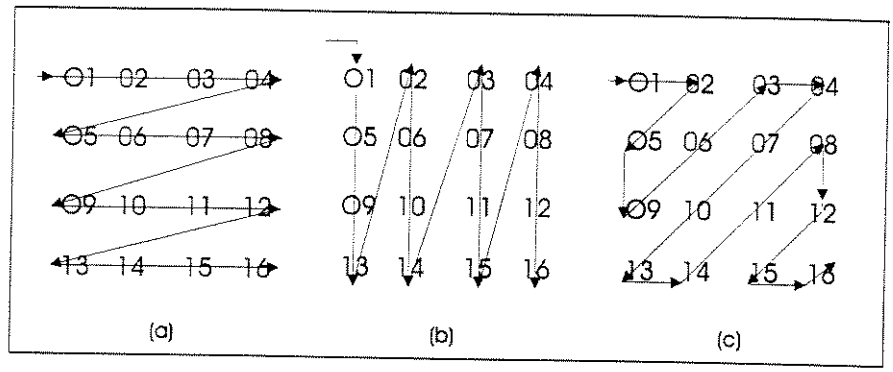



Figura 4.5: Tipos de varredura: (a) linha a linha, (b) coluna a coluna e (c) zig-zag.

## 4.4 Codificação Entrópica

Em uma imagem natural há grandes regiões de píxeis com amplitude nula e/ou idêntica aos píxeis da vizinhança. A transformada wavelet, assim como a DCT, tem a propriedade de concentrar a maior parte da energia de uma imagem em uma pequena porção de coeficientes. No caso da DCT, a energia fica concentrada nos coeficientes de baixa freqüência dos blocos 8x8. Na transformada wavelet, a energia fica concentrada principalmente na sub-banda de baixas freqüências. As outras sub-bandas apresentam, então, uma grande quantidade de coeficientes nulos.

Em um esquema de compressão, a varredura deve ser realizada após a quantização e antes da codificação entrópica. O objetivo da varredura é agrupar os coeficientes da transformada de maneira que a codificação entrópica seja eficiente. Se possível, os coeficientes devem estar ordenados em ordem decrescente de energia formando longas seqüências de coeficientes nulos, que podem ser codificados com poucos bits. Nesta seção estudaremos algumas formas de varredura adequadas aos codificadores utilizados.

Como já discutido, as sub-bandas de detalhes (médias e altas freqüências) são codificadas utilizando o *run length*. Já a sub-banda de baixas freqüências não apresenta uma quantidade significativa de zeros que justifique a utilização deste código. Um código Huffman tradicional é utilizado neste caso. Estudaremos estes tipos de codificadores no fim desta seção.

### 4.4.1 Varredura

A transformada wavelet apresenta algumas características particulares, que podem ser aproveitadas pela varredura. Entre estas características está a correlação espacial entre coeficientes de diferentes sub-bandas [Shapiro] [Barreto]. Se um grupo de píxeis estiver altamente correlacionado, há uma grande chance de que eles tenham amplitudes iguais. Se este valor for igual a zero, formamos uma seqüência de coeficientes nulos. Neste trabalho vamos apresentar apenas alguns tipos de varredura que julgamos ser interessantes para cada sub-banda da TW.

As formas mais simples de varreduras são aquelas linha a linha e coluna a coluna, apresentadas nas Figuras 4.5-(a) e 4.5-(b), respectivamente. Estas varreduras exploram a correlação existente entre elementos vizinhos em uma linha ou coluna. A varredura utilizada no JPEG é

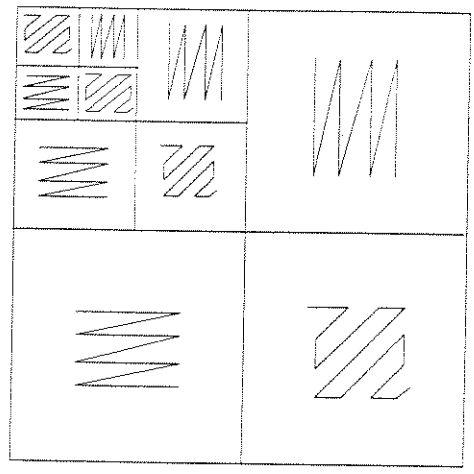


Figura 4.6: Tipos de varreduras utilizadas em cada sub-banda de uma decomposição wavelet em três camadas.

a zig-zag, apresentada na Figura 4.5-(c). Nesta varredura, os elementos da imagem são lidos de forma diagonal, o que permite explorar a correlação existente em ambas as direções de uma imagem.

As sub-bandas de detalhes horizontais da transformada wavelet apresentam mudanças apenas no sentido horizontal. No sentido vertical, os coeficientes são altamente correlacionados e, portanto, uma varredura eficiente para esta sub-banda é a varredura coluna a coluna. Já as sub-bandas de detalhes verticais apresentam mudanças apenas no sentido vertical e uma alta correlação dos coeficientes no sentido horizontal. Portanto, a varredura mais adequada é a linha a linha.

As sub-bandas de altas freqüências apresentam mudanças em ambos os sentidos da imagem (horizontal e vertical) e, conseqüentemente, não há uma correlação específica para estas sub-bandas. Por outro lado, a quantidade de coeficientes nulos destas sub-bandas é muito grande. Optamos por utilizar uma varredura em zig-zag para estas sub-bandas e, desta forma, não privilegiar qualquer direção.

Nas sub-bandas de baixas freqüências, a correlação entre coeficientes vizinhos é praticamente idêntica à da imagem original, que, em geral, é muito grande. Optamos, então, por uma varredura em zig-zag pelo mesmo motivo. O tipo de varreduras utilizadas para cada sub-banda em uma decomposição de três camadas é apresentado na Figura 4.6.

Além das correlações entre píxeis, a transformada wavelet apresenta uma característica bastante interessante: existe uma grande correlação espacial entre os coeficientes de sub-bandas em diferentes níveis de decomposição da TW [Shapiro]. Analisemos o seguinte exemplo: uma imagem 32x32 decomposta em 3 camadas através da TW. As sub-bandas 2, 5 e 8 (o ordenamento das sub-bandas foi apresentado na Figura 4.3) são altamente correlacionadas [Shapiro], uma vez que a sub-banda 2 é uma aproximação grosseira da sub-banda 5, e esta, por sua vez, é uma aproximação grosseira da sub-banda 8. O mesmo é válido para as sub-bandas 3, 6 e 9

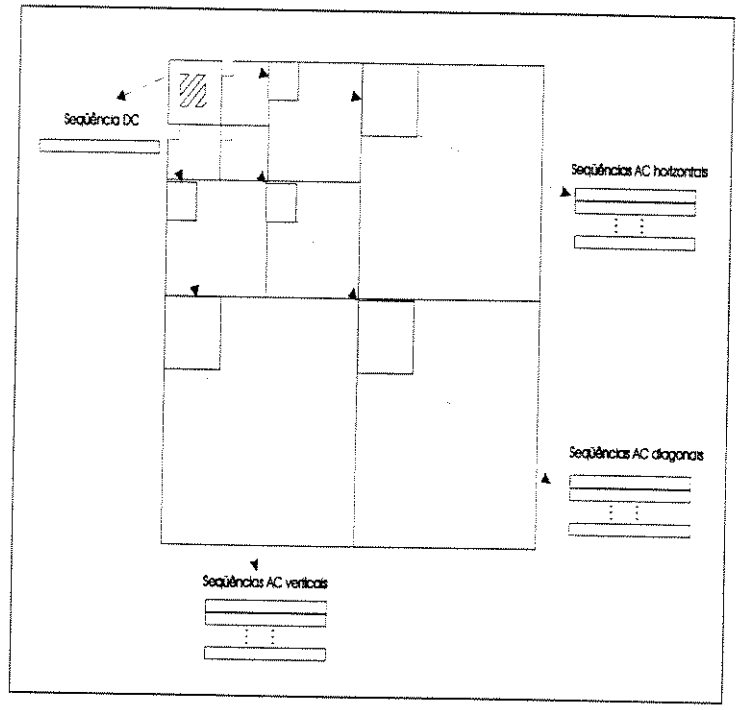



Figura 4.7: Varredura tipo 1 baseada na correlação espacial e espectral das sub-bandas. No total são gerados 3 conjuntos de seqüências AC e uma seqüência DC.

e as sub-bandas 4, 7 e 10. Suponha que o píxel no canto superior esquerdo da sub-banda 2 na Figura 4.7 seja nulo. Então, há uma grande probabilidade de que os píxeis do quadrado sombreado $2 \times 2$ da sub-banda 5 também sejam nulos. Da mesma forma, os píxeis do quadrado sombreado $4 \times 4$ da sub-banda 8 provavelmente também são nulos. Estas regiões sombreadas são denominadas regiões espacialmente correlacionadas.

Esta importante característica foi utilizada por Shapiro [Shapiro] no desenvolvimento de um algoritmo denominado EZW ( Embedded Zerotree Wavelet). Este é um algoritmo para compressão de imagens, cuja principal característica é o fato dos bits serem codificados pela sua ordem de importância visual. O EZW é formado por uma seqüência de decisões binárias que distinguem coeficientes nulos ou válidos. Neste trabalho utilizaremos uma simplificação deste tipo de algoritmo, eliminando as estruturas de testes e simplificando as árvores utilizadas pelo algoritmo. Estas simplificações permitiram o desenvolvimento de duas formas de agrupamento dos coeficientes (após a quantização), que apresentaremos em seguida.

Na Figura 4.7 apresentamos uma ilustração da primeira destas formas de agrupamento. O primeiro passo consiste em dividir as sub-bandas em três classes: detalhes horizontais, verticais e diagonais. Para cada classe existe um vetor de coeficientes que será formado a partir de pequenas seqüências de coeficientes. Estas seqüências, por sua vez, são formadas pela concatenação das regiões espacialmente correlacionadas desta classe, de acordo com a ordem da sub-banda na classe.

Podemos observar que cada conjunto de regiões sombreadas correlacionadas espacialmente vai gerar um vetor de coeficientes AC. Desta forma, para as sub-bandas e regiões de cada

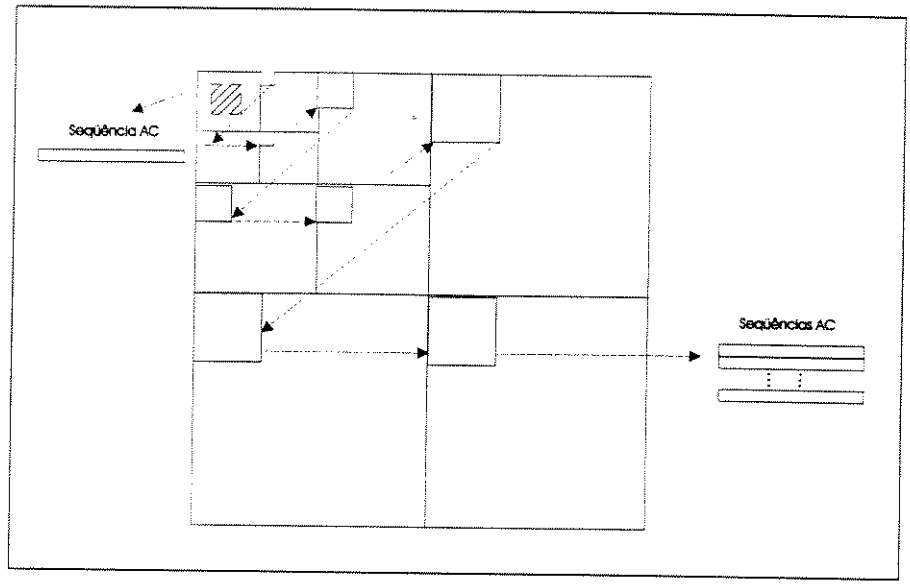



Figura 4.8: Varredura tipo 2 baseada na correlação espacial e espectral das sub-bandas. No total são gerados 1 conjunto de seqüências AC e uma seqüência DC.

conjunto, é utilizada uma varredura adequada à sub-banda (ver Figura 4.6). No total, três vetores de coeficientes AC são gerados, uma para cada classe de sub-banda (detalhes horizontais, verticais ou diagonais), que serão codificados individualmente. Denominaremos este modelo de agrupamento de varredura do tipo 1.

Na Figura 4.8 apresentamos a ilustração da segunda forma de agrupamento de coeficientes. A grande diferença é que neste caso apenas uma classe de sub-bandas é criada e, conseqüentemente, apenas um vetor de coeficientes AC. As seqüências formadas são, portanto, um pouco maiores, sendo formadas pela concatenação das regiões espacialmente correlacionadas de todas as sub-bandas (exceto a de baixas freqüências), de acordo com o índice desta na decomposição. Em cada região e sub-banda são utilizadas as varreduras adequadas. O objetivo desta modificação é tirar o máximo proveito da correlação espacial existente entre as sub-bandas da mesma camada de decomposição. Este procedimento foi denominado varredura do tipo 2.

A sub-banda de baixas freqüências passa pelo estágio de varredura em zig-zag e é, então, codificada utilizando o código de Huffman.

### 4.4.2 Codificação

Após a varredura das sub-bandas, temos em mãos um(três) vetor(es) AC (médias e altas freqüências) e um vetor DC (baixas freqüências) prontos para serem codificados. O(s) vetor(es) AC são codificados utilizando-se o código *run-length*. A codificação da seqüência DC exige, porém, maiores cuidados, pois carrega o conteúdo DC da imagem. A utilização do código *run-length* não é interessante porque este vetor não apresenta grandes cadeias de zeros. Uma solução bastante simples é utilizar o código PCM ou o código DPCM. Estes códigos, entretanto, possuem uma baixa capacidade de compressão pois codificam cada elemento da seqüência com

palavras de comprimentos fixos. Uma alternativa mais eficiente consiste em utilizar o código de Huffman.

## 4.5 Ensaios e Comparações

Nesta seção, apresentaremos três esquemas de compressão baseados na transformada wavelet. Estes esquemas utilizam as técnicas descritas anteriormente neste capítulo. Na Seção 4.5.4 são apresentados os resultados das simulações utilizadas para testar o desempenho destes esquemas. A implementação foi dividida em duas partes. A primeira, formada pelos estágios da transformada wavelet e quantização, foi implementada utilizando o Matlab® [Sanchez]. A segunda, formada pelo codificador/ decodificador entrópico, foi implementada em linguagem C.

### 4.5.1 Primeiro Esquema de Compressão

O primeiro esquema é bem simples. Utiliza uma quantização com vetor fixo e uma varredura linha a linha em cada uma das sub-bandas. A sub-banda de baixas freqüências é codificada utilizando o código Huffman. As outras sub-bandas são codificadas utilizando o código *run-length*.

Utilizamos uma decomposição wavelet com 2, 3 e 4 camadas, com o objetivo de medir o impacto de uma decomposição em várias camadas sobre desempenho. Além disso, realizamos simulações utilizando parte do conjunto de wavelets apresentado no Capítulo 2 para verificar o seu desempenho. Foram testadas duas wavelets ortogonais e três wavelets biortogonais. As ortogonais foram as wavelets Daubechies 6 e 8 [Daubechies92] que representaremos por daub6 e daub8. As biortogonais foram uma Spline, uma wavelet quase-ortonormal e uma variante da Spline [Antonini], as quais representaremos por $h_1$, $h_2$ e $h_3$, respectivamente.

Os vetores de quantização utilizados neste esquema foram:

$$\begin{aligned}
\mathbf{q}_1 &= [1\ 2\ 2\ 2\ 4\ 4\ 4\ 8\ 8\ 8] \\
\mathbf{q}_2 &= [2\ 4\ 4\ 4\ 10\ 10\ 10\ 20\ 20\ 20] \\
\mathbf{q}_3 &= [2\ 4\ 4\ 4\ 10\ 10\ 10\ 32\ 32\ 32] \\
\mathbf{q}_4 &= [4\ 16\ 16\ 16\ 32\ 32\ 32\ 64\ 64\ 64] \\
\mathbf{q}_5 &= [4\ 16\ 16\ 16\ 64\ 64\ 64\ 128\ 128\ 128]
\end{aligned}$$

Nas Tabelas A.1 A.2 A.3 A.4 e A.5 apresentamos o resultado (PSNRs e taxas de bits) da simulação de um esquema de compressão com 3 camadas de decomposição, utilizando os vetores de quantização ($\mathbf{q}_1$, $\mathbf{q}_2$, $\mathbf{q}_3$, $\mathbf{q}_4$ e $\mathbf{q}_5$), com fatores de quantização 1, 2, 4 e 8. Foram utilizadas as wavelets biortogonais ( $h_1$, $h_2$ e $h_3$ ) e as ortogonais daub6 e daub8. Nas Figuras A.1, A.2, A.3, A.4 e A.5 são apresentados os gráficos de desempenho deste esquema para os vetores propostos. A wavelet que apresentou o melhor desempenho quantitativo (PSNR) foi a biortogonal $h_3$. As wavelets daub6, daub8 e $h_1$ apresentaram desempenhos semelhantes e, na maioria dos casos, $h_1$ apresentou um melhor desempenho. A wavelet $h_2$ apresentou o pior desempenho quantitativo.

Nas Figuras B.1-(a), B.1-(b), B.2-(a), B.2-(b) e B.3-(a) apresentamos as imagens reconstruídas deste conjunto de experiências, utilizando o vetor de quantização $\mathbf{q}_4$ e as wavelets daub6, daub8, $h_1$, $h_2$ e $h_3$. A partir destas imagens pudemos perceber que a degradação introduzida pela wavelet biortogonal $h_3$ é inferior à introduzida pelas demais. Os experimentos

também mostraram que as wavelets biortogonais $h_1$ e $h_2$ introduzem uma distorção inferior à introduzida pelas wavelets *daub*6 e *daub*8, o que contraria aos resultados quantitativos. Isto se deve ao fato de qua as medidas de erro quantitativas não conseguirem expressar adequadamente a importância visual das degradações. Sendo assim, a qualidade visual é considerada um fator mais importante na análise de uma imagem que as medidas de erro quantitativas. Portanto, as wavelets biortogonais possuem um desempenho superior em relação às ortogonais.

Na Figura A.6 apresentamos uma comparação entre o desempenho dos diversos vetores de quantização utilizando a wavelet $h_3$. Os vetores $q_2$ e $q_4$ apresentaram os melhores desempenhos. Entretanto, uma quantidade maior de fatores de quantização deve ser utilizada para que se possa obter um intervalo maior de taxas de bits.

O último conjunto de simulações tem o objetivo de comparar o desempenho deste esquema com duas, três ou quatro camadas de decomposição. Para isto, utilizamos a wavelet biortogonal $h_3$ e os vetores de quantização $\mathbf{q}_1$, $\mathbf{q}_2$ e $\mathbf{q}_4$. Na Tabela A.7, podemos observar o resultado deste conjunto de simulações para duas, três e quatro camadas de decomposição. Nas Figuras B.5-(a), B.5-(b), B.6-(a) e B.6-(b) apresentamos imagens reconstruídas utilizando-se 2 e 4 camadas de decomposição, a wavelet biortogonal $h_3$ e os vetores de quantização $\mathbf{q}_2$ e $\mathbf{q}_4$, respectivamente.

Concluímos que a capacidade de compressão de um sistema aumenta com o número de camadas. Porém, o ganho ao se passar de três para quatro camadas é pequeno. Considerando que o aumento de camadas provoca um aumento da complexidade, consideramos que a decomposição em três camadas representa o melhor compromisso entre simplicidade e desempenho. Adotamos, então, este tipa decomposição nos esquemas do restante do trabalho.

### 4.5.2 Segundo Esquema de Compressão

O segundo esquema de compressão apresenta melhorias em relação ao primeiro. Neste esquema utilizamos o quantizador com alocação ótima de bits ponderada, que, conforme já discutido, apresenta bons resultados.

Utilizamos a varredura apresentada na Figura 4.7 e a wavelet biortogonal $h_3$. Assim como no primeiro esquema de codificação, as sub-bandas de médias e altas freqüências foram codificadas utilizando o código *run-length*, enquanto que a sub-banda de baixas freqüências foi codificada utilizando o código de Huffman.

Os resultados das simulações deste esquema para vários valores de $R_C$ e $a$ são apresentados na Tabela A.7. Na Figura A.7 apresentamos uma comparação entre os desempenhos deste esquema para estes valores de $a$. Observamos que o valor que apresentou o melhor desempenho quantitativo foi $a = 1$, seguido por $a = 1,5$, $a = 2$ e $a = 2,5$. O valor de PSNR diminui, portanto, com o valor de $a$. A diferença entre os resultados, entretanto, não é significativa.

As imagens reconstruídas para R= (2), (4), (6) e (10) e os mesmos valores de $a$ são apresentadas nas Figuras B.7 até B.14. Os resultados mostram que há uma diferença entre a qualidade objetiva (quantitativa) e subjetiva (visual) das imagens. Apesar do valor de $a$ não interferir de forma significativa no resultado quantitativo, a qualidade visual da imagem melhora consideravelmente com o aumento do valor de $a$.

### 4.5.3 Terceiro Esquema de Compressão

O terceiro esquema de compressão é bastante semelhante ao anterior. A principal diferença se encontra no tipo de varredura utilizada. Neste caso utilizamos a varredura apresentada na Figura 4.8. Os resultados das simulações deste esquema para vários valores de $R_C$ e $a$ são apresentados na Tabela A.8. Na Figura A.8 apresentamos uma comparação dos desempenhos deste esquema para estes valores de $a$. Observamos que o melhor desempenho quantitativo ocorreu para $a = 1$ e que não existe uma diferença significativa entre os desempenhos. As imagens reconstruídas são as mesmas do esquema anterior, já que apenas a codificação entrópica (que não introduz erros) foi modificada. Valem, então, as mesmas considerações sobre a qualidade subjetiva das imagens em relação ao parâmetro $a$.

Na Figura A.9 apresentamos uma comparação entre os desempenhos deste esquema e do esquema 2. Este esquema apresentou um desempenho ligeiramente superior. Podemos concluir que esta superioridade se deve à uma maior capacidade de compressão da varredura do tipo 2.

Na Figura A.10 apresentamos uma comparação entre os desempenhos deste esquema ( $a = 1$ e $a = 2$) e dos esquemas 1 (vetores $q_2$ e $q_4$) e 2 ( $a = 1$ e $a = 2$), simultaneamente. O esquema 1 apresenta um desempenho inferior aos esquemas 1 e 2. A qualidade visual das imagens obtidas com os esquemas 2 e 3 também é superior às obtidas com o esquema 1. Portanto, a técnica de alocação de bits é melhor que a quantização fixa e a varredura da Figura 4.8 deve ser utilizada.

### 4.5.4 Conclusões

Os esquemas de compressão apresentados neste capítulo apresentaram um desempenho muito bom. A estrutura destes esquemas é baseada em uma quantização escalar e uma codificação entrópica utilizando o código *run-length* e o código de Huffman. O esquema 1 é o mais simples e o que apresenta menor capacidade de compressão. Os esquemas 2 e 3, um pouco mais complexos, apresentam como vantagem um maior controle sobre a taxa de bits que se deseja obter. Além disso, como estes esquemas exploram a característica de correlação espacial da TW, o desempenho de codificação é maior do que no caso do esquema 1, tanto quantitavamente quanto visualmente.

O esquema de compressão 3 um desempenho ligeiramente superior ao do esquema 2. O melhor resultado quantitativo foi obtido para $a = 1$, enquanto que o melhor resultado visual foi obtido para $a = 2,5$.

As wavelets biortogonais apresentaram um desempenho superior em relação às ortogonais. Em particular a wavelet $h_3$ apresentou um desempenho bem melhor que as outras wavelets testadas. Quanto ao número de camadas de decomposição, concluímos que um esquema com três camadas atende ao compromisso de simplicidade e eficiência.

# Capítulo 5

# Compressão de Vídeo

Neste capítulo apresentaremos uma breve introdução às técnicas de compressão de vídeo, uma área de estudo que consideramos de fundamental importância no cenário atual das telecomunicações. Com a crescente demanda pelos serviços de vídeo e com o advento das redes ATM, cresce o interesse por algoritmos que sejam eficientes e satisfaçam as novas exigências dos serviços de telecomunicações.

Os atuais algortimos de compressão possuem uma estrutura híbrida, formada por um codificador DPCM/DCT com compensação de movimentos [Haskell96] [Mitchell]. A utilização da transformada wavelet nesta aplicação é uma conseqüência dos bons resultados obtidos na compressão de imagens estáticas. Já existem na literatura codificadores utilizando esta transformada com bons desempenhos [Said] [Ohta] [Martucci]. Neste capítulo, apresentaremos um modelo bastante simples de um codificador baseado na transformada wavelet.

Um outro tópico, que será abordado brevemente neste capítulo é a escalonabilidade. A utilização de algoritmos de codificação com escalonabilidade dá ao sistema uma maior robustez a erros de transmissão, além de permitir que funcionalidades como a compatibilidade e a interoperabilidade sejam implementadas entre sistemas com diferentes níveis de resolução ou qualidade. A transformada wavelet, por sua vez, provê uma maneira mais fácil de implementação, devido à sua natureza "escalonável" [Wang] [Comer]. Apresentaremos um modelo simples de codificação com escalonabilidade baseado na transformada wavelet.

## 5.1 Princípios da Compressão de Vídeo

As seqüências de vídeo contêm uma quantidade significativa de redundância estatística e subjetiva. O objetivo do codificador de vídeo é reduzir a taxa de bits necessária para o armazenamento e transmissão, o que é feito através da exploração das redundâncias, de maneira que apenas um conjunto mínimo de informações seja codificado entropicamente. O desempenho da compressão, entretanto, vai depender das técnicas utilizadas e do conteúdo das cenas. Em sistemas práticos, um compromisso entre o desempenho e a complexidade deve ser observado.

As seqüências de vídeo possuem redundâncias de caráter espacial e temporal [Haskell96] [Mitchell]. As correlações exploradas pelas técnicas de compressão são especialmente aquelas entre píxeis de um mesmo quadro ou entre píxeis de quadros adjacentes. Ou seja, assume-se que a magnitude de um píxel de uma imagem pode ser predito a partir dos píxeis vizinhos do

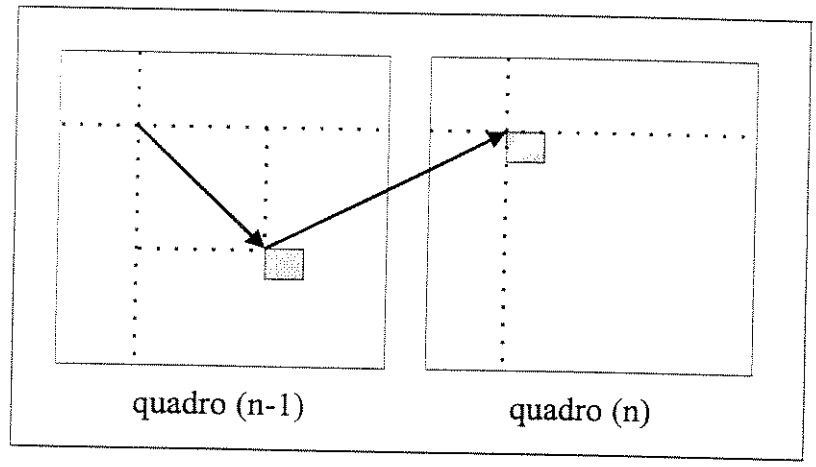



Figura 5.1: Estimação de movimento entre dois quadros consecutivos.

quadro (utilizando a codificação intra-quadros) ou de píxeis de um quadro adjacente (utilizando a codificação inter-quadros).

A codificação intra-quadros é a maneira mais simples, tanto conceitualmente como em termos de implementação, de se obter uma compressão de seqüências de vídeo. Nesta abordagem, cada quadro é tratado como uma imagem estática e uma das técnicas de compressão descritas no Capítulo 3 é utilizada. A codificação inter-quadros, um pouco mais elaborada, explora a dependência entre os quadros consecutivos do sinal de vídeo, ou seja, a redundância temporal da seqüência. Para isso é utilizada uma técnica denominada predição com compensação de movimentos [Haskell96] [Mitchell], à qual se associa a codificação DPCM.

O conceito de compensação de movimentos é baseado na estimação de movimentos entre quadros. Se uma cena apresenta uma quantidade de movimento relativamente baixa, a melhor predição é freqüentemente obtida a partir dos píxeis do quadro adjacente temporalmente. O movimento dos píxeis pode ser descrito por um número limitado de parâmetros de movimento, isto é, os vetores de movimento, que descrevem a trajetória de deslocamento dos píxeis entre quadros consecutivos.

O processo de determinação dos vetores de movimento (vm) é denominado estimação de movimento, e os vetores resultantes são transmitidos para o receptor. Neste, o decodificador identifica qual a área da imagem de referência utilizada para a predição através de uma compensação de movimento. Soma-se, então, esta predição com a diferença decodificada para obter a imagem reconstruída. A compensação de movimento também é realizada no codificador para calcular o erro de predição que será codificado.

Em geral, como é alta a correlação entre os vetores de movimento de uma região, o movimento é estimado para pequenos blocos da cena. Desta forma, um conjunto de vetores de movimento é utilizado para representar o movimento de cada bloco. Na Figura 5.1 apresentamos um modelo de uma compensação de movimento entre dois quadros.

Para cenas de pouco movimento é grande a correlação entre os píxeis de quadros adjacentes. O desempenho deste codificador é, então, superior ao dos codificadores intra-quadros. Por outro lado, a correlação entre píxeis de quadros adjacentes é pequena para cenas de muito movimento.

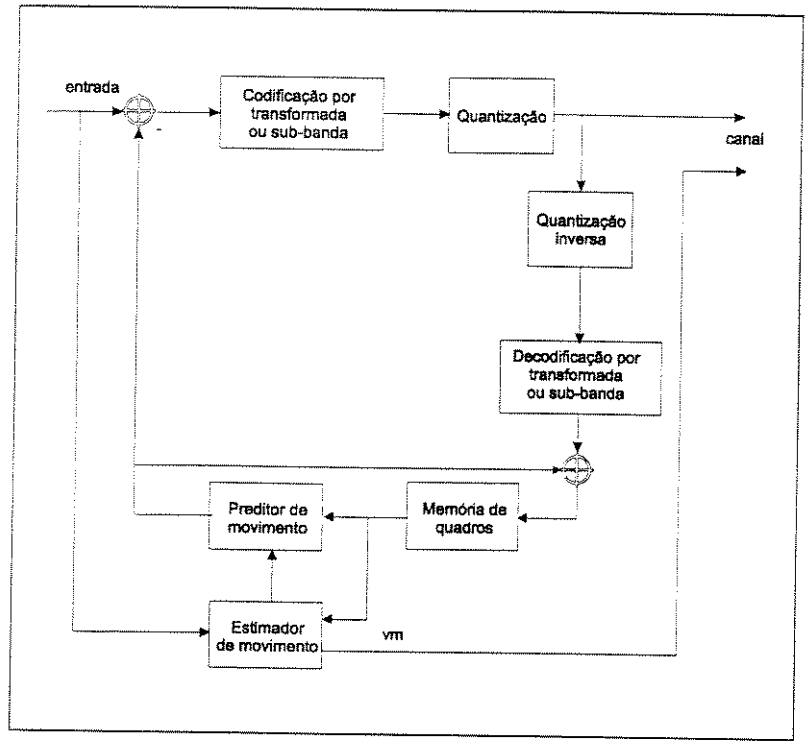



Figura 5.2: Codificador híbrido com laço interno e compensação de movimento.

Conseqüentemente, o desempenho dos codificadores inter-quadros é ruim. Nestas situações é mais interessante utilizar uma codificação intra-quadros.

Devido à correlação existente entre os quadros consecutivos de uma seqüência, os codificadores inter-quadros são mais sensíveis a erros de transmissão. Em geral, neste tipo de codificadores os erros podem se propagar por vários quadros. A codificação intra-quadros, por sua vez, é menos sensível a este tipo de erros, uma vez que os erros são confinados em apenas um quadro da seqüência, o que impede a sua propagação. Em geral, os compressores utilizam as duas codificações, de acordo com o conteúdo da cena.

Na Figura 5.2 apresentamos um esquema básico de um codificador de vídeo, denominado codificador híbrido. Esta é uma das técnicas mais tradicionais de codificação de vídeo e combina duas formas distintas de redução da taxa de bits: (1) codificação por transformada/ sub-bandas espacial e (2) codificação preditiva temporal [Jain88] [Netravali] [Walker]. Quase invariavelmente esta técnica consiste em aplicar o DPCM no domínio do tempo e uma codificação por transformada/ sub-banda no domínio espacial.

No codificador apresentado na Figura 5.2, o DPCM é utilizado com o objetivo de explorar a correlação entre píxeis de quadros adjacentes. Desta forma, apenas os elementos do quadro que apresentaram mudanças significativas em relação aos quadros anteriores são transmitidos. No receptor, a informação decodificada é utilizada para atualizar os quadros anteriores. Neste esquema é utilizada uma predição com compensação de movimentos. Em cenas de muito movimento, entretanto, o codificador opta por uma codificação intra-quadros.

O codificador híbrido apresenta um bom desempenho para serviços convencionais, como por

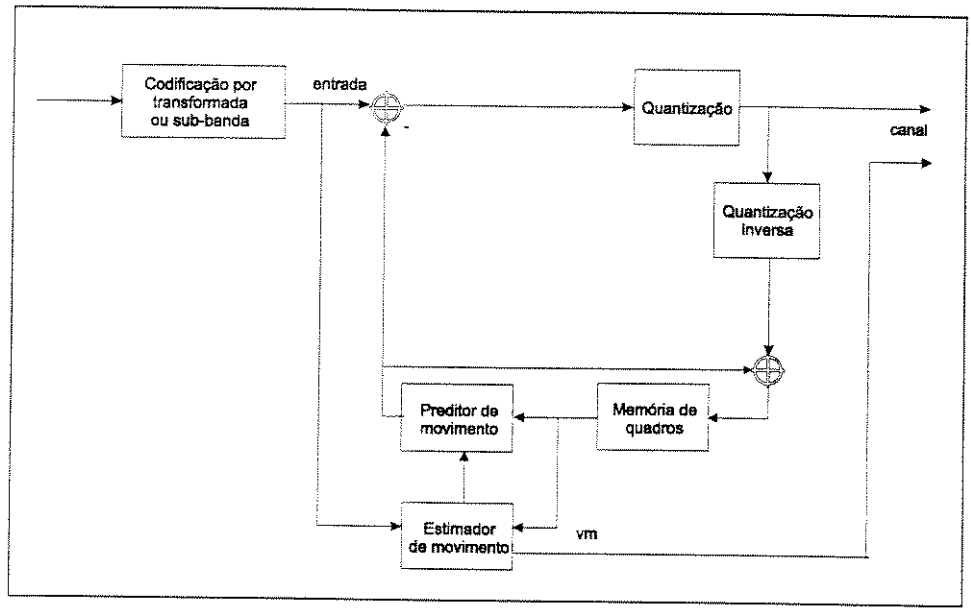



Figura 5.3: Codificador híbrido com laço externo e compensação de movimento.

exemplo, a distribuição do sinal de TV ou o armazenamento de dados de multi-mídia. Uma das maiores desvantagens desta técnica é, entretanto, a sua vulnerabilidade a erros de transmissão e a sua incompatibilidade com outros padrões.

Os novos serviços de telecomunicações exigem que os codificadores/ decodificadores sejam compatíveis com outros padrões e serviços e robustos a erros de transmissão. Uma solução consiste em utilizar a técnica de codificação com escalonabilidade, que se baseia na codificação da informação em camadas. Na seção seguinte, faremos um breve estudo sobre este tipo de codificador.

## 5.2 Codificação de Vídeo com Escalonabilidade

A escalonabilidade de vídeo é a habilidade de se obter sinais com diferentes resoluções ou níveis de qualidade. O objetivo dos sistemas escalonáveis é oferecer uma compatibilidade ou interoperabilidade entre diferentes serviços ou padrões. No atual contexto dos serviços de telecomunicações, esta habilidade é extremamente desejável em sistemas de codificação de vídeo.

A criação de novos serviços, ou mesmo, o aprimoramento dos atuais serviços de telecomunicações, aumenta a importância dos codificadores de vídeo. Com a futura transmissão de sinais de televisão digital e de televisão de alta definição - HDTV (High Definition TV) [Haskell96] [Mitchell], a compatibilidade entre os vários sistemas é vital. É importante que o usuário de TV convencional possa receber o sinal de HDTV e obter um sinal de menor resolução (correspondente a TV convencional) sem ter que decodificar todo o sinal recebido. Com o advento da rede digital de serviços integrados de faixa larga ( "Broadband Integrated Service Digital Network "- B-ISDN), que é uma evolução das atuais redes de telecomunicações, as exigências em relação aos codificadores de vídeo aumentam. Esta rede fornecerá vários tipos de serviços,

com diferentes níveis de resolução e diferentes taxas de bits. Os novos algoritmos devem lidar com os vários formatos de sinais pertencentes a uma dada família. O desafio é comprimir os sinais de vídeo com um mínimo de distorções possível, mantendo a compatibilidade entre os diversos sistemas.

A codificação de vídeo escalonável envolve a geração de uma representação codificada com mais de uma resolução ou qualidade, e também é denominada "codificação de vídeo em camadas" ou "codificação hierárquica". Existem, basicamente, duas formas de se obter seqüências de vídeo com escalonabilidade: a codificação simulcast e a codificação de vídeo com escalonabilidade. Na codificação simulcast cada camada de vídeo corresponde a um tipo de resolução ou qualidade e é codificada independentemente das outras. A banda de freqüências disponível é particionada de acordo com a resolução ou qualidade que se deseja atribuir a cada camada. Nesta codificação são utilizados decodificadores independentes para cada camada.

Na codificação de vídeo com escalonabilidade, a primeira camada de vídeo é codificada independentemente, enquanto que as demais são codificadas de acordo com a anterior. A primeira camada é denominada camada de base e transporta um sinal com qualidade ou resolução básica. Esta camada, pelo fato de ter sido codificada independentemente, pode ser decodificada por decodificadores não-escalonáveis. As outras camadas, denominadas camadas de enriquecimento, transportam as informações de refinamento ou enriquecimento e, quando adicionadas ou combinadas com a camada de base, geram um sinal com uma resolução ou qualidade mais alta.

Geralmente a codificação com escalonabilidade é mais eficiente que a simulcast, uma vez que, com exceção da camada de base, todas as outras camadas são codificadas utilizando parte da banda da informação atribuída à camada anterior. O aumento da eficiência, entretanto, vai depender do tipo de técnica utilizada, do número de camadas e da partição da banda de freqüências utilizada. Este aumento implica, porém, em um aumento da complexidade em relação a codificação simulcast. Em geral utiliza-se um codificador com escalonabilidade com duas camadas: uma camada de base e uma camada de enriquecimento.

As principais vantagens dos codificadores com escalonabilidade, em relação aos codificadores sem escalonabilidade, são:

1. **Maior poder de recuperação contra erros** - A codificação em camadas possui um poder de recuperação contra erros na perda de camadas, pois garante que os dados contidos na camada básica sejam recuperados, garantido desta forma uma qualidade mínima. Isto é muito importante, por exemplo, em redes do tipo ATM, que foram selecionadas como base para o sistema B-ISDN, e em aplicações em radiodifusão através de satélites e distribuição terrestre. Apesar de ser possível empregar códigos corretores de erros para reduzir os erros de transmissão, em termos de economia é mais interessante aceitar que alguns erros cheguem ao receptor de vídeo e tentar disfarçá-los. Os códigos corretores, devido ao seu alto custo, podem ser utilizados na proteção apenas dos dados mais sensíveis.

2. **Compatibilidade** - Provavelmente a maior vantagem da codificação em camadas é a sua capacidade de oferecer compatibilidade entre os padrões, recomendados pelo CCITT, de codecs para a ISDN (banda estreita) e para o B-ISDN (banda larga). A camada de maior prioridade do novo codificador B-ISDN deve ser idêntica ao sinal completo do codificador ISDN de banda estreita. Assim, o sinal emitido pelos codificadores mais novos pode ser decodificado pelos mais antigos, sem qualquer custo extra. As camadas de prioridade menor contém versões finamente quantizadas da diferença entre o sinal completo

e a primeira camada. Um exemplo de uma aplicação em telecomunicações onde a compatibilidade pode ser bastante interessante é a vídeo-telefonia. Nos atuais sistemas de vídeo-telefonia de múltiplos pontos, todos os participantes transmitem e recebem numa mesma taxa de bits, determinada pelo participante de menor capacidade. Com a implementação de um sistema compatível, cada participante terá acesso a um sinal condizente com a sua capacidade. Uma unidade de controle enviará o sinal completo para os participantes que possuem capacidade suficiente e uma versão do sinal com resolução menor para os participantes com menor capacidade.

### 5.2.1 Tipos de Escalonabilidade

Existem basicamente quatro tipos de Escalonabilidade [ITU-T]:

- Partição de dados
- Escalonabilidade SNR (Relação Sinal-Ruído)
- Escalonabilidade Espacial
- Escalonabilidade Temporal
- Escalonabilidade Híbrida

A partição de dados não é exatamente uma codificação com escalonabilidade, uma vez que esta técnica consiste simplesmente na partição de um sinal, já codificados, em duas camadas ou partições. À cada camada pode ser atribuída uma prioridade diferente, o que possibilita a implementação de uma transmissão menos sensível a erros. Na Figura 5.4 apresentamos o diagrama em blocos de um codec com partição de dados. As principais aplicações desta técnica são as transmissões de vídeo em redes ATM ou qualquer outro tipo de rede.

A escalonabilidade SNR gera camadas individuais de sinal codificado, com níveis de qualidade diferentes, mas uma mesma resolução espacial ou temporal. A camada de base transporta os coeficientes DCT com qualidade mais baixa, enquanto que a camada de enriquecimento transporta os coeficientes diferenciais de refinamento. Quando adicionados aos coeficientes DCT da camada mais baixa, estes coeficientes enriquecem a relação sinal ruído (SNR). Se apenas a camada de base for recuperada no receptor, a imagem reconstruída tem uma qualidade básica. Na Figura 5.5 apresentamos um esquema genérico de um codificador com escalonabilidade SNR com duas camadas.

Entre as importantes aplicações da escalonabilidade SNR se encontram a transmissão de vídeo em redes ATM ou qualquer outro tipo de rede, transmissão de sinais de TV digital (ou HDTV) com diferentes níveis de qualidade, serviços de vídeo com vários níveis de qualidade, etc.

A escalonabilidade espacial consiste em uma codificação em camadas, onde cada camada possui uma resolução espacial própria, mas uma mesma resolução temporal e qualidade. As camadas são fracamente correlacionadas, e existe uma considerável liberdade na utilização de resoluções e formatos de vídeo em cada camada. Uma outra característica importante é o fato da camada mais alta utilizar uma predição espacial inter-camadas, com relação às imagens decodificadas da camada inferior. Esta predição é responsável por um aumento no desempenho

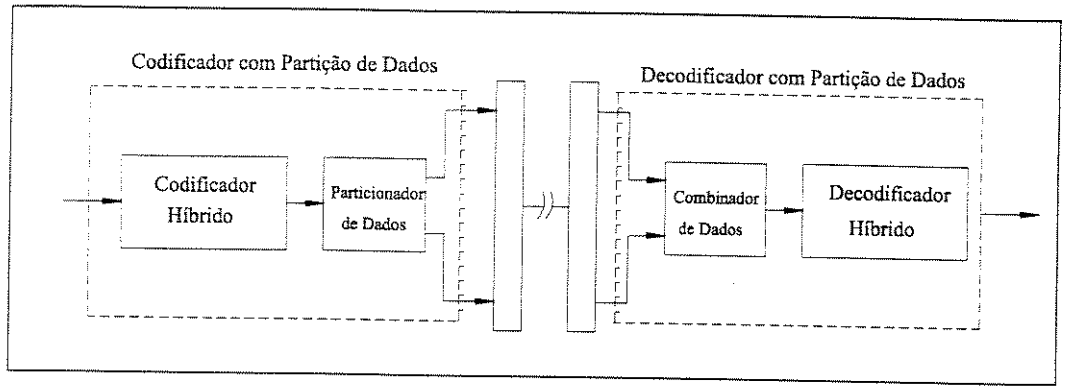



Figura 5.4: Codec com partição de dados.

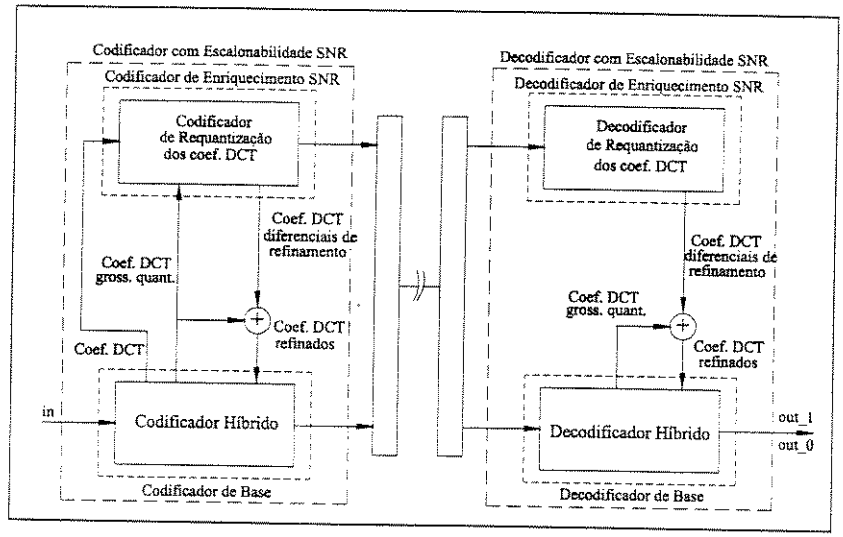



Figura 5.5: Codec com escalonabilidade SNR.

do codificador. Na Figura 5.6 apresentamos um esquema genérico de um codificador com escalonabilidade espacial com duas camadas.

A codificação com escalonabilidade espacial permite a interoperabilidade entre aplicações utilizando diferentes formatos de vídeo e diferentes padrões, aplicações em computadores e rádio-difusão, a possibilidade de integração de diversos serviços de telecomunicações, a criação de um sistema com duas camadas que permitam a seleção entre os sinais HDTV e TV (normal) na recepção, criação de sistemas com maior resistência a erros em redes ATM ou outras redes, etc.

A codificação com escalonabilidade temporal produz duas ou mais camadas com resoluções temporais próprias, de maneira que quando as camadas são combinadas obtém-se uma resolução temporal completa igual à do sinal de entrada. A taxa de entrada de quadros é simplesmente particionada entre a camada de base e as camadas de enriquecimento. Na Figura 5.7 apresentamos um esquema genérico de um codificador com escalonabilidade temporal com duas

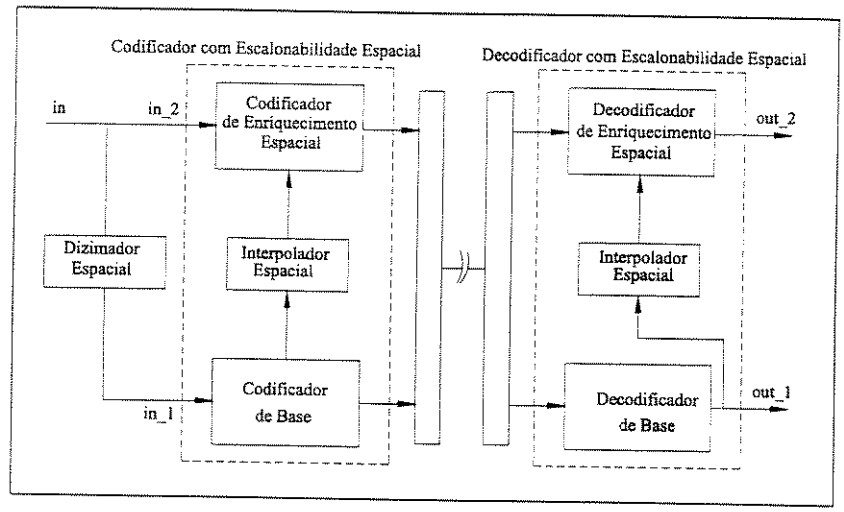



Figura 5.6: Codec com Escalonabilidade Espacial.

camadas. Neste, o decodificador tem uma complexidade semelhante àquela do decodificador não-escalonável.

Uma das aplicações da escalonabilidade temporal mais promissoras é para HDTV, para a qual a migração da primeira geração de HDTV entrelaçada para a versão de HDTV progressiva de alta resolução temporal é possível de maneira compatível. Há também outras aplicações, tais como, a geração de novos serviços de comunicação utilizando resolução de TV com um formato progressivo no dobro da taxa de quadros. Esta aplicação exige uma interoperabilidade com os "displays" de TV existentes. Além destas aplicações, a escalonabilidade temporal também é útil em ambientes onde o processador de decodificação não é tão poderoso a ponto de conseguir decodificar o sinal de vídeo a uma taxa de quadros completa ou, ainda, pode estar sendo compartilhando por diferentes usuários ou tarefas.

A escalonabilidade híbrida é a combinação de dois ou mais tipos diferentes de escalonabilidade. Se duas escalonabilidades são combinadas para formar uma escalonabilidade híbrida, pelo menos três camadas são obtidas: a camada de base, a camada de enriquecimento 1 e camada de enriquecimento 2. A camada de enriquecimento 1 é a camada imediatamente inferior à camada de enriquecimento 2, da mesma forma que a camada de base é a camada imediatamente inferior à camada de enriquecimento 1. Na Figura 5.8 apresentamos um exemplo de um codificador com escalonabilidade híbrida - um codificador com escalonabilidade SNR-temporal.

## 5.3 Codificação de Vídeo Baseada na T.W.

A transformada wavelet é uma ferramenta valiosa para o processamento de vídeo devido à sua flexibilidade na representação de sinais não-estacionários. A compressão de vídeo baseada na TW proporciona uma descorrelação eficiente da informação contida em cada quadro da seqüência, como acontece no caso das imagens estáticas. Além disso, o algoritmo de estimação de movimentos para a TW possui uma baixa complexidade, uma vez que é possível explorar a grande correlação espacial entre as sub-bandas.

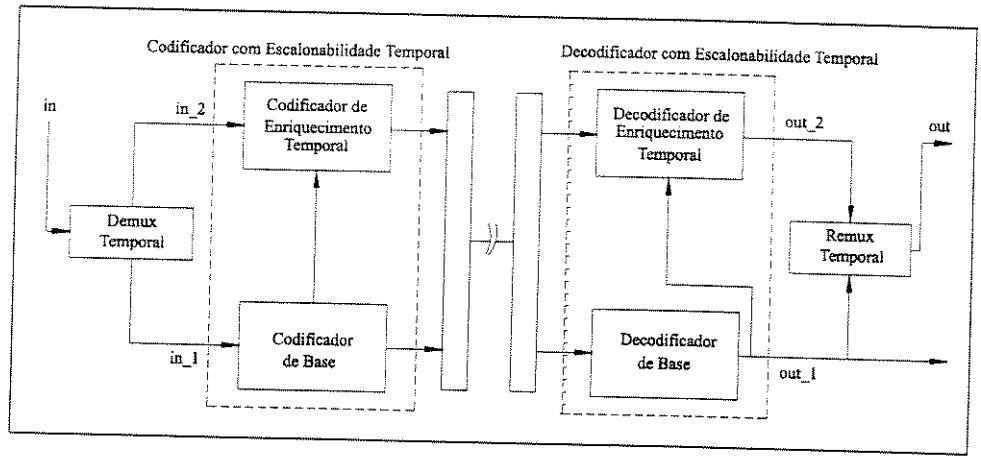



Figura 5.7: Codec com Escalonabilidade Temporal.

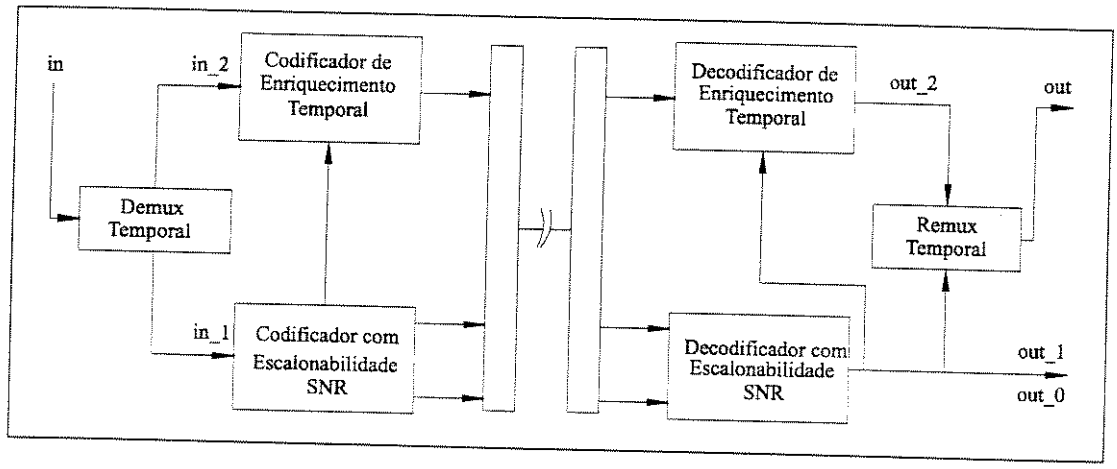



Figura 5.8: Codec com Escalonabilidade SNR-Temporal (híbrido).

Na Figura 5.9 apresentamos um diagrama em blocos de um codificador híbrido baseado na TW. O codificador é formado por 3 blocos principais: a TW, a compensação de movimento e a quantização. Em geral, assim como no caso da compressão de imagens estáticas, podem ser utilizadas de 2 a 5 camadas de decomposição.

A redundância temporal de uma seqüência de vídeo é eliminada pela compensação de movimentos. Os quadros de erro de predição são, então, quantizados e codificados. Evidentemente, existem outras formas de codificação além da apresentada na Figura 5.9. Por exemplo, a compensação de movimentos pode ser realizada antes da TW ou, ainda, os quadros de erro de predição podem ser mais decorrelacionados utilizando uma DCT. Entretanto, os resultados [Zhang] [VanDyck] [Jung] [Vlachos]mostram que o esquema da Figura 5.9 possui um desempenho de codificação superior.

Nesta estrutura, a escolha do algoritmo de estimação de movimentos é crucial e determina o desempenho e complexidade do codificador [Haskell96]. Zhang et al [Zhang] propuseram uma

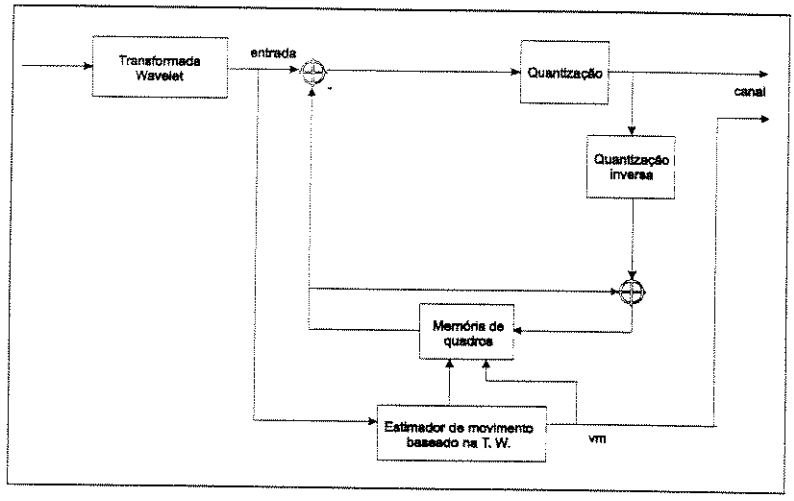



Figura 5.9: Codificador de vídeo híbrido baseado na TW.

técnica de estimação de movimentos com multiresolução (MRME) que estima os vetores de movimento no domínio da TW. Nesta técnica, a estimação dos vetores de movimento (vm) é realizada de acordo com o índice da camada de decomposição. Os vetores de movimento das sub-bandas de detalhes e a sub-banda de baixas freqüências da última camada de decomposição são calculados através de um algoritmo de estimação de movimentos tradicional, semelhante ao utilizado pelo MPEG [Haskell96] [ITU-T]. Os vetores de movimento das demais camadas são calculados através de uma predição que utiliza os vetores da próxima camada como referência.

Seja, por exemplo, uma decomposição wavelet em três camadas. Os vetores das sub-bandas $D^1_{2^2}$, $D^2_{2^2}$ e $D^3_{2^2}$, da segunda camada de decomposição são preditos a partir dos vetores das sub-bandas $D^1_{2^3}$, $D^2_{2^3}$ e $D^3_{2^3}$, da terceira camada.

Assim como nas técnicas tradicionais, a estimação de movimentos na MRME é realizada com base em blocos de píxeis. Entretanto, os blocos utilizados nesta técnica possuem tamanhos diferentes de acordo com a camada de decomposição. Assim, por exemplo, para uma decomposição em 3 camadas utilizando um bloco $n \times n$ de referência, as sub-bandas da primeira, segunda e terceira camadas são divididas em blocos de tamanho $n \times n$, $2n \times 2n$ e $4n \times 4n$. Desta forma, garantimos que o número de blocos em todas as sub-bandas seja idêntico.

A vantagem de se utilizar uma codificação baseada na TW é que, além de apresentar um alto desempenho de codificação, a qualidade subjetiva desta técnica é bem superior à das técnicas tradicionais baseadas na DCT, em particular, do padrão MPEG. A TW não introduz os efeitos de blocagem e reduz bastante o ruído "mosquito" [Said] [Ohta] [Martucci].

### 5.3.1 Codificação de vídeo com escalonabilidade baseado na TW

A transformada wavelet provê uma maneira fácil de se implementar a codificação com escalonabilidade espacial. Isto se deve ao fato da decomposição wavelet apresentar a propriedade de escalonabilidade de forma natural, uma vez que, ao decompormos uma imagem utilizando a transformada wavelet obtemos uma outra imagem com uma resolução espacial inferior e três imagens diferenciais de enriquecimento.

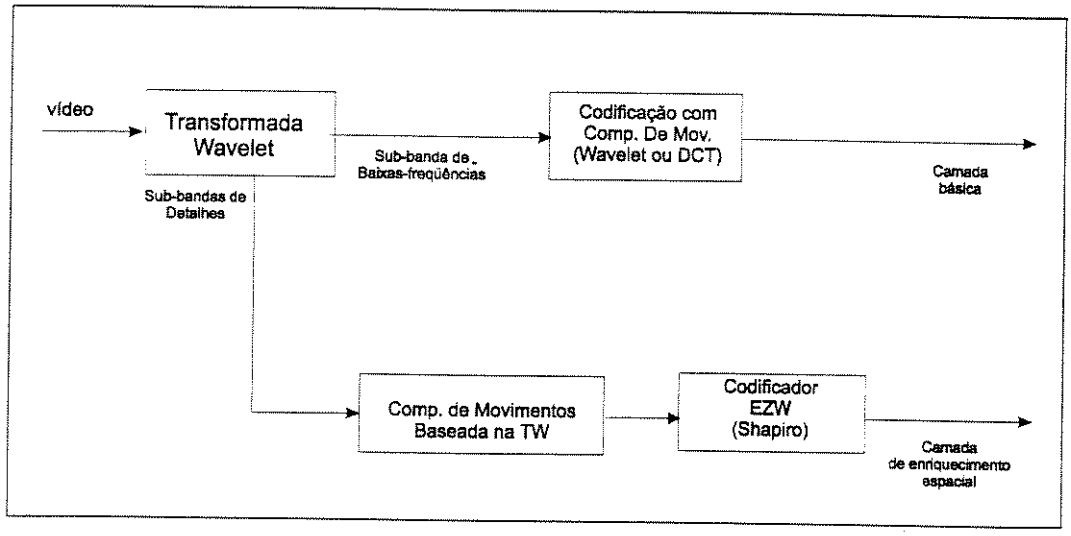



Figura 5.10: Codificador com escalonabilidade espacial baseado na TW.

É possível, então, construir um codificador de vídeo com escalonabilidade espacial baseado na TW acrescentando poucas modificações ao esquema da Figura 5.9 [Comer]. Na Figura 5.10 apresentamos um exemplo de tal codificador. O sinal de entrada é decomposto pela transformada wavelet, produzindo quatro imagens resultantes: a sub-banda de baixas-freqüências e as sub-bandas de detalhes. A sub-banda de baixas-freqüências é codificada utilizando um codificador de vídeo com compensação de movimentos, que pode ser baseado na TW ou na DCT. As imagens de detalhes passam por um estágio de compensação de movimentos semelhante ao MRME e , em seguida, são codificadas por um codificador EZW [Shapiro].

Em relação ao esquemas de codificação com escalonabilidade baseados na DCT, este tipo de codificador apresenta melhores valores de SNR e uma qualidade subjetiva bem superior [Comer]. As possibilidades de implementação de outro tipo de escalonabilidade ainda estão sendo estudadas.

# Capítulo 6

# Conclusão

## 6.1 Resumo

Neste trabalho realizamos uma análise sobre as potencialidades da aplicação da transformada wavelet na compressão de imagens, com ênfase em imagens estáticas e monocromáticas. Iniciamos com um estudo sobre a teoria wavelet, enfatizando os pontos importantes para compressão de imagens. Em particular, estudamos as wavelets biortogonais e analisamos as vantagens da sua utilização no processamento de imagens.

Em seguida, realizamos também um estudo sobre as principais técnicas de compressão de imagens, detendo-nos no estudo dos codificadores com perdas. Apresentamos alguns esquemas de compressão baseados na transformada wavelet, onde detalhamos os diversos estágios destes esquemas, expondo as técnicas que podem ser utilizadas em cada um deles. Analisamos os desempenhos dos codificadores utilizando wavelets biortogonais e ortogonais. Avaliamos também a utilização de duas, três ou quatro camadas de decomposição, levando em conta o compromisso entre eficiência e simplicidade.

Estudamos três modelos de quantizadores escalares: um quantizador com vetor fixo, um quantizador com alocação de bits ótima e um quantizador com alocação ponderada de bits. Apresentamos alguns modelos de varredura de coeficientes, baseados no trabalho de Shapiro [Shapiro], que tiram proveito da correlação espacial existente entre as sub-bandas da transformada wavelet. Combinando algumas das técnicas estudadas, construímos três esquemas de compressão baseados na transformada wavelet e analisamos os seus desempenhos.

O esquema 1 foi composto por um quantizador com vetor fixo e uma varredura dos coeficientes linha a linha em cada uma das sub-bandas.

No esquema 2 utilizamos um quantizador com alocação ótima de bits ponderada e uma varredura baseada na correlação espacial e espectral das sub-bandas.

O esquema 3 difere do esquema 2 apenas quanto ao tipo de varredura, que visa a concatenação das regiões espacialmente correlacionadas de todas as sub-bandas (exceto a de baixas freqüências).

Vislumbrando novas aplicações da transformada wavelet na área de processamento de imagens, apresentamos uma breve introdução à compressão de vídeo. Abordamos tópicos como os sistemas com escalonabilidade e codificadores de vídeo utilizando a transformada wavelet.

## 6.2 Conclusões

As wavelets biortogonais apresentaram um desempenho superior ao das ortogonais, especialmente em termos de qualidade visual, uma vez que as imagens reconstruídas utilizando wavelets biortogonais apresentam distorções menos perceptíveis. Esta diferença é ainda maior quando o número de camadas de decomposição aumenta, fazendo com que aumentem as distorções de bordas para imagens reconstruídas utilizando wavelets ortogonais.

O número de camadas de decomposição influencia diretamente na eficiência da codificação. Quanto maior o número de camadas, maior a concentração de energia obtida e, conseqüentemente, maior a eficiência de codificação. Entretanto, um número maior de camadas implica também em uma maior complexidade computacional. Analisamos os desempenhos de esquemas com 2, 3 e 4 camadas de decomposição e optamos por uma decomposição em três camadas, que atende ao compromisso entre simplicidade e eficiência de forma satisfatória.

Os esquemas de compressão estudados apresentaram um desempenho satisfatório. Os esquemas 2 e 3 apresentaram um desempenho subjetivo superior ao do esquema 1, demonstrando que a alocação ótima de bits, embora exigindo maior complexidade, oferece melhores resultados que a quantização fixa. O esquema 3 foi ligeiramente superior ao 2, demonstrando que o esquema de varredura correspondente é mais eficiente em proporcionar correlações entre blocos de imagens.

Os estudos sobre compressão de vídeo utilizando wavelets demonstraram a factibilidade de emprego desta transformada como alternativa à DCT. Foi possível também demonstrar as facilidades proporcionadas pela TW para a construção de sistemas de compressão de vídeo com escalonabilidade, uma vez que a escalonabilidade é uma característica intrínseca da TW.

Finalmente, podemos concluir que a TW é uma excelente alternativa às técnicas tradicionais de compressão de imagens, como por exemplo, a DCT. Devido à sua alta capacidade de compressão, aliada à sua baixa complexidade computacional, a TW apresenta-se como uma ferramenta competitiva e bastante promissora, tendo nos últimos anos sido alvo de diversas pesquisas. As pesquisas mais recentes têm se concentrado na busca de generalizações da TW que possam satisfazer melhor as exigências das diversas aplicações na área de procesamento de sinais. A utilização da TW em sistemas de compressão práticos já é uma realidade. O FBI americano, por exemplo, adotou a TW como padrão para compressão de impressões digitais. Além disso, o MPEG-4 vem estudando a possibilidade de empregar a TW na compressão de vídeo.

## 6.3 Sugestões para Trabalhos Futuros

Algumas modificações podem ainda ser implementadas nos esquemas de compressão apresentados neste trabalho. Como os esquemas são compostos por diversos estágios relativamente independentes, pode-se buscar o aperfeiçoamento de cada um deles isoladamente. Uma primeira sugestão é a utilização de uma quantização vetorial que, apesar da sua complexidade, apresenta um desempenho superior ao da quantização escalar. Uma outra sugestão é a utilização de uma codificação aritmética ao invés da codificação Huffman/*run-length*. Da mesma forma, outros métodos de alocação de bits podem ser estudados.

Com relação à própria transformada wavelet, algumas generalizações desta transformada,

como por exemplo, as *wavelet packets* [Coifman] e as *multiwavelets* [Strela96a] [Strela96b] podem ser aplicadas nos algoritmos apresentados neste trabalho. Já existem na literatura algumas propostas de algoritmos para compressão de imagens utilizando wavelet packets [Ramchandra] [Coifman] [Bradley] [Brislawn]. As multiwavelets, por sua vez, possuem todas as vantagens das wavelets tradicionais além de apresentarem, simultaneamente, características como ortogonalidade, simetria, maior ordem de aproximação e suporte compacto. Os sistemas que utilizam as multiwavelets prometem ser superiores àqueles que utilizam as wavelets tradicionais [Heller] [Lian] [Strela96a].

Este trabalho pode ser estendido no sentido de aprofundar os esudos na área de compressão de vídeo. Já existem na literatura bons codificadores utilizando esta transformada [Said] [Ohta] [Martucci], servindo como excelentes alternativas ao padrão de codificação de vídeo MPEG. Além disso, pode ser estudada a aplicação das multiwavelets e das wavelet packets no desenvolvimento de codificadores de vídeo com escalonabilidade.

# Apêndice A

# Tabelas e Gráficos

| | $h_1$ | $h_2$ | $h_3$ | *Daub6* | *Daub8* |
|---|---|---|---|---|---|
| $1 * \mathbf{q}_1$ | 3,5400 bpp<br>37,1984 dB | 3,5303 bpp<br>37,7574 dB | 3,3563 bpp<br>37,6526 dB | 3,5107 bpp<br>37,6333 dB | 3,4899 bpp<br>37,6924 dB |
| $2 * \mathbf{q}_1$ | 2,8300 bpp<br>34,0600 dB | 2,7820 bpp<br>33,9188 dB | 2,6990 bpp<br>34,4604 dB | 2,6782 bpp<br>34,1583 dB | 2,7662 bpp<br>34,2429 dB |
| $4 * \mathbf{q}_1$ | 2,1019 bpp<br>31,4949 dB | 1,9857 bpp<br>31,0304 dB | 1,9196 bpp<br>31,9421 dB | 1,9824 bpp<br>31,4796 dB | 1,9716 bpp<br>31,6482 dB |
| $8 * \mathbf{q}_1$ | 1,6190 bpp<br>29,8612 dB | 1,5325 bpp<br>29,2569 dB | 1,5018 bpp<br>30,1847 dB | 1,5408 bpp<br>29,6490 dB | 1,5465 bpp<br>29,7953 dB |

Tabela A.1: Valores de taxa de bits e PSNR para o primeiro esquema de codificação, utilizando o vetor $q_1$ e 3 camadas.

| | $h_1$ | $h_2$ | $h_3$ | *Daub6* | *Daub8* |
|---|---|---|---|---|---|
| $1 * \mathbf{q}_2$ | 2,5396 bpp<br>35,5493 dB | 2,4667 bpp<br>35,4318 dB | 32,3153 bpp<br>35,6673 dB | 2,4449 bpp<br>35,5400 dB | 2,4296 bpp<br>35,5877 dB |
| $2 * \mathbf{q}_2$ | 1,8945 bpp<br>32,2445 dB | 1,8538 bpp<br>31,7030 dB | 1,7542 bpp<br>32,2689 dB | 1,8285 bpp<br>31,9751 dB | 1,8279 bpp<br>32,0783 dB |
| $4 * \mathbf{q}_2$ | 1,4567 bpp<br>29,5062 dB | 1,4449 bpp<br>28,6133 dB | 1,3514 bpp<br>29,3861 dB | 1,4430 bpp<br>28,9740 dB | 1,4094 bpp<br>29,0435 dB |
| $8 * \mathbf{q}_2$ | 1,1923 bpp<br>27,2937 dB | 1,1421 bpp<br>26,2359 dB | 1,1195 bpp<br>26,9418 dB | 1,1453 bpp<br>26,5320 dB | 1,1526 bpp<br>26,4558 dB |

Tabela A.2: Valores de taxa de bits e PSNR para o primeiro esquema de codificação, utilizando o vetor $q_2$ e 3 camadas.

| | $h_1$ | $h_2$ | $h_3$ | *Daub6* | *Daub8* |
|---|---|---|---|---|---|
| $1 * q_3$ | 2,3859 bpp<br>33.7504 dB | 2,2469 bpp<br>33,4961 dB | 2,1410 bpp<br>34,0106 dB | 2,2406 bpp<br>33,7233 dB | 2,2244 bpp<br>33,7852 dB |
| $2 * q_3$ | 1,7460 bpp<br>30,9853 dB | 1,7457 bpp<br>30,3822 dB | 1,6805 bpp<br>31,2052 dB | 1,7321 bpp<br>30,7812 dB | 1,7416 bpp<br>30,9402 dB |
| $4 * q_3$ | 1,4319 bpp<br>28,9760 dB | 1,3724 bpp<br>28,1481 dB | 1,3291 bpp<br>28,9500 dB | 1,3771 bpp<br>28,4949 dB | 1,3801 bpp<br>28,5690 dB |
| $8 * q_3$ | 1,1090 bpp<br>27,1873 dB | 1,0795 bpp<br>26,1235 dB | 1,0637 bpp<br>26,8496 dB | 1,1135 bpp<br>26,3781 dB | 1,1187 bpp<br>26,3003 dB |

Tabela A.3: Valores de taxa de bits e PSNR para o primeiro esquema de codificação, utilizando o vetor $q_3$ e 3 camadas.

| | $h_1$ | $h_2$ | $h_3$ | *Daub6* | *Daub8* |
|---|---|---|---|---|---|
| $1 * q_4$ | 1,6794 bpp<br>30,9521 dB | 1,6672 bpp<br>30,3225 dB | 1,6165 bpp<br>31,1340 dB | 1,6646 bpp<br>30,7230 dB | 1,6631 bpp<br>30,8803 dB |
| $2 * q_4$ | 1,3715 bpp<br>28,8964 dB | 1,2979 bpp<br>28,0248 dB | 1,2708 bpp<br>28,7994 dB | 1,3013 bpp<br>28,3668 dB | 1,3085 bpp<br>28,4336 dB |
| $4 * q_4$ | 1,0530 bpp<br>27,0024 dB | 1,0270 bpp<br>25,8604 dB | 1,0156 bpp<br>26,5458 dB | 1,06241 bpp<br>26,1147 dB | 1,0646 bpp<br>26,0348 dB |
| $8 * q_4$ | 0,9276 bpp<br>24,9901 dB | 0,8944 bpp<br>23,8630 dB | 0,8904 bpp<br>24,3743 dB | 0,9041 bpp<br>24,0726 dB | 0,9008 bpp<br>24,0576 dB |

Tabela A.4: Valores de taxa de bits e PSNR para o primeiro esquema de codificação, utilizando o vetor $q_4$ e 3 camadas.

| | $h_1$ | $h_2$ | $h_3$ | *Daub6* | *Daub8* |
|---|---|---|---|---|---|
| $1 * q_5$ | 1,4497 bpp<br>28,9772 dB | 1,3906 bpp<br>28,1510 dB | 1,3470 bpp<br>28,9535 dB | 1,3951 bpp<br>28,4977 dB | 1,3966 bpp<br>28,5719 dB |
| $2 * q_5$ | 1,1471 bpp<br>27,1915 dB | 1,1174 bpp<br>26,1308 dB | 1,1019 bpp<br>26,8592 dB | 1,1514 bpp<br>26,3855 dB | 1,1577 bpp<br>26,3077 dB |
| $4 * q_5$ | 0,9976 bpp<br>25,3746 dB | 0,9647 bpp<br>24,3986 dB | 0,9602 bpp<br>25,0233 dB | 0,9771 bpp<br>24,6290 dB | 0,9708 bpp<br>24,5951 dB |
| $8 * q_5$ | 0,025 bpp<br>24,2496 dB | 0,8716 bpp<br>23,5388 dB | 0,8704 bpp<br>24,0165 dB | 0,8752 bpp<br>23,7249 dB | 0,8782 bpp<br>23,7882 dB |

Tabela A.5: Valores de taxa de bits e PSNR para o primeiro esquema de codificação, utilizando o vetor $q_5$ e 3 camadas.

| | 2 camadas | 3 camadas | 4 camadas |
|---|---|---|---|
| $\mathbf{q_1}$ | 4,3905 bpp<br>28,5709 dB | 3,3563 bpp<br>37,6526 dB | 2,2592 bpp<br>28,1110 dB |
| $\mathbf{q_2}$ | 3,991 bpp<br>28,3103 dB | 2,3153 bpp<br>35,667323 | 1,8018 bpp<br>28,0469 dB |
| $\mathbf{q_4}$ | 3,5975 bpp<br>27,83dB | 1,6165 bpp<br>31,1340 dB | 0,9407 bpp<br>27,6681 dB |

Tabela A.6: Valores de taxa de bits e PSNR para o primeiro esquema de compressão com 2, 3 e 4 camadas de decomposição, utilizando os vetores $q_1$, $q_2$ e $q_4$ e a wavelet $h_3$.

| Esq. 2 | $a = 1$ | | $a = 1,5$ | | $a = 2,0$ | | $a = 2,5$ | |
|---|---|---|---|---|---|---|---|---|
| R | PSNR | taxa | PSNR | taxa | PSNR | taxa | PSNR | taxa |
| 0,5 | 22,7842 | 0,6526 | 23,1924 | 0,6790 | 23,4113 | 0,6951 | 23,5787 | 0,7075 |
| 1 | 23,8731 | 0,7014 | 24,3313 | 0,7417 | 24,4482 | 0,7747 | 24,5302 | 0,7914 |
| 2 | 25,1405 | 0,7620 | 26,0834 | 0,8497 | 26,5432 | 0,8984 | 26,8073 | 0,9451 |
| 4 | 27,1303 | 0,8959 | 27,9693 | 1,0035 | 28,4124 | 1,0899 | 28,6342 | 1,1625 |
| 6 | 28,9641 | 1,0671 | 29,4830 | 1,1998 | 29,7370 | 1,3152 | 29,8997 | 1,3816 |
| 10 | 33,4066 | 1,7402 | 33,3852 | 1,8848 | 33,3556 | 1,9957 | 33,2898 | 2,0654 |
| 12 | 35,9780 | 2,2754 | 35,7512 | 2,4022 | 35,7166 | 2,5193 | 35,5608 | 2,5591 |
| | | | | | | | | |

Tabela A.7: Valores de taxa de bits e PSNR para o segundo esquema de compressão.

| Esq.3 | $a = 1$ | | $a = 1,5$ | | $a = 2,0$ | | $a = 2,5$ | |
|---|---|---|---|---|---|---|---|---|
| R | PSNR | taxa | PSNR | taxa | PSNR | taxa | PSNR | taxa |
| 0,5 | 22,7842 | 0,6538 | 23,1924 | 0,6800 | 23,4113 | 0,6954 | 23,5787 | 0,7083 |
| 1 | 23,8731 | 0,7001 | 24,3313 | 0,7394 | 24,4482 | 0,7719 | 24,5302 | 0,7880 |
| 2 | 25,1405 | 0,7572 | 26,0834 | 0,8427 | 26,5432 | 0,8893 | 26,8073 | 0,9353 |
| 4 | 27,1303 | 0,8865 | 27,9693 | 0,9923 | 28,4124 | 1,0782 | 28,6342 | 1,1500 |
| 6 | 28,9641 | 1,0553 | 29,4830 | 1,1869 | 29,7370 | 1,3004 | 29,8997 | 1,3656 |
| 10 | 33,4066 | 1,7166 | 33,3852 | 1,8597 | 33,3556 | 1,9677 | 33,2898 | 2,0968 |
| 12 | 35,9780 | 2,2463 | 35,7512 | 2,3716 | 35,7166 | 2,4867 | 35,5608 | 2,5964 |
| | | | | | | | | |

Tabela A.8: Valores de taxa de bits e PSNR para o terceiro esquema de compressão.

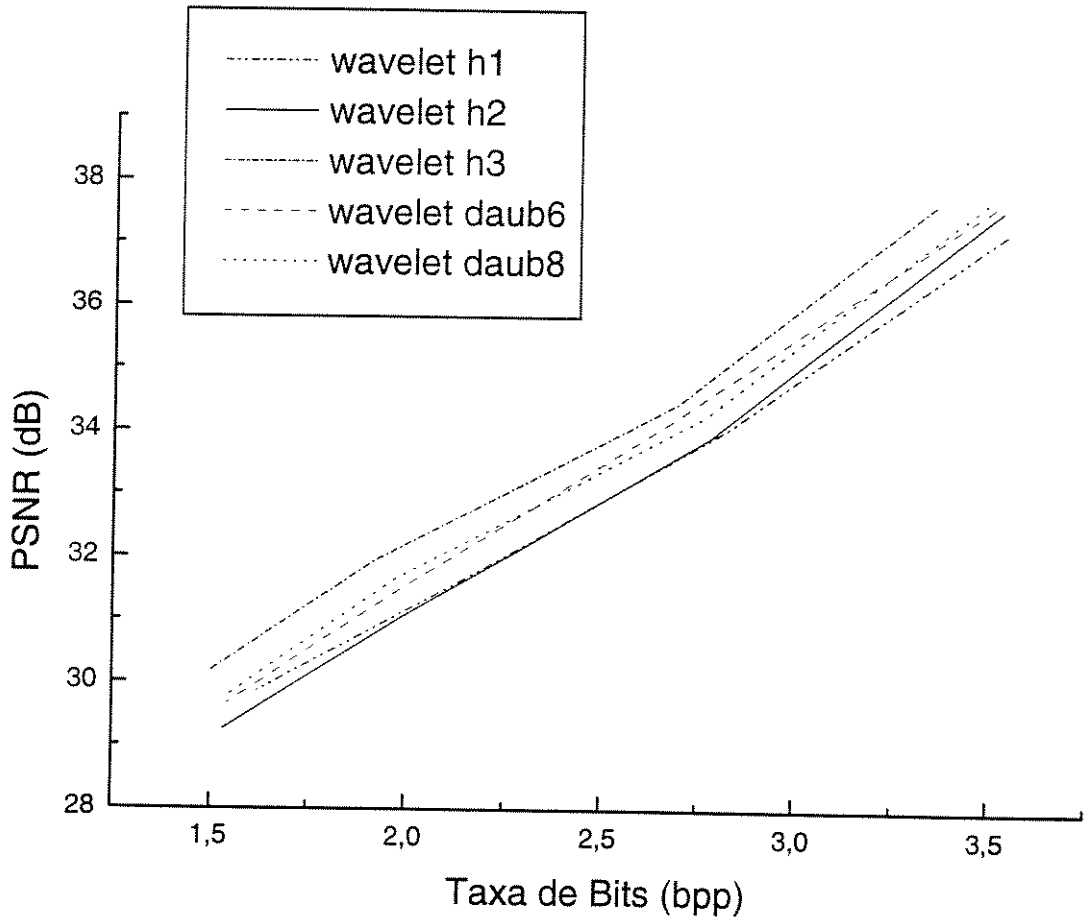



Figura A.1: Comparação entre os valores de PSNR para as wavelets $h_1$, $h_2$, $h_3$, *daub*6 e *daub*8, resultantes de um esquema com três camadas de decomposição e o vetor $q_1$.

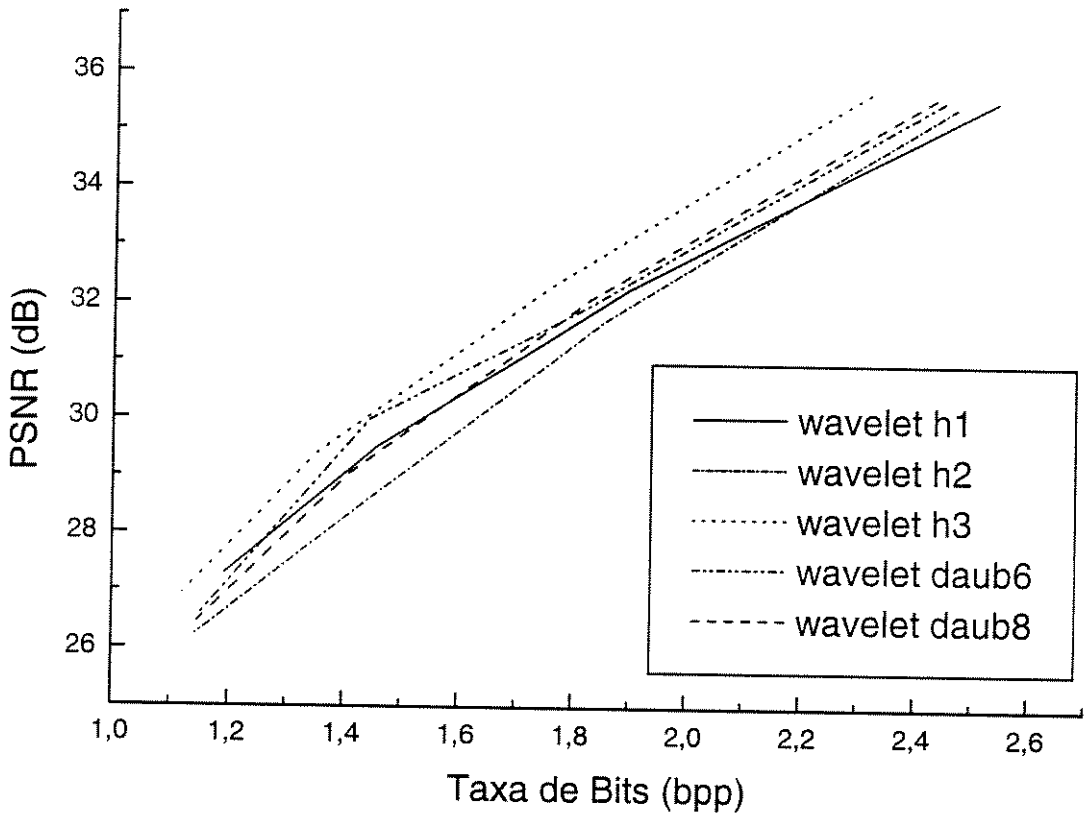



Figura A.2: Comparação entre os valores de PSNR para as wavelets $h_1$, $h_2$, $h_3$, *daub*6 e *daub*8, resultantes de um esquema com três camadas de decomposição e o vetor $q_2$.

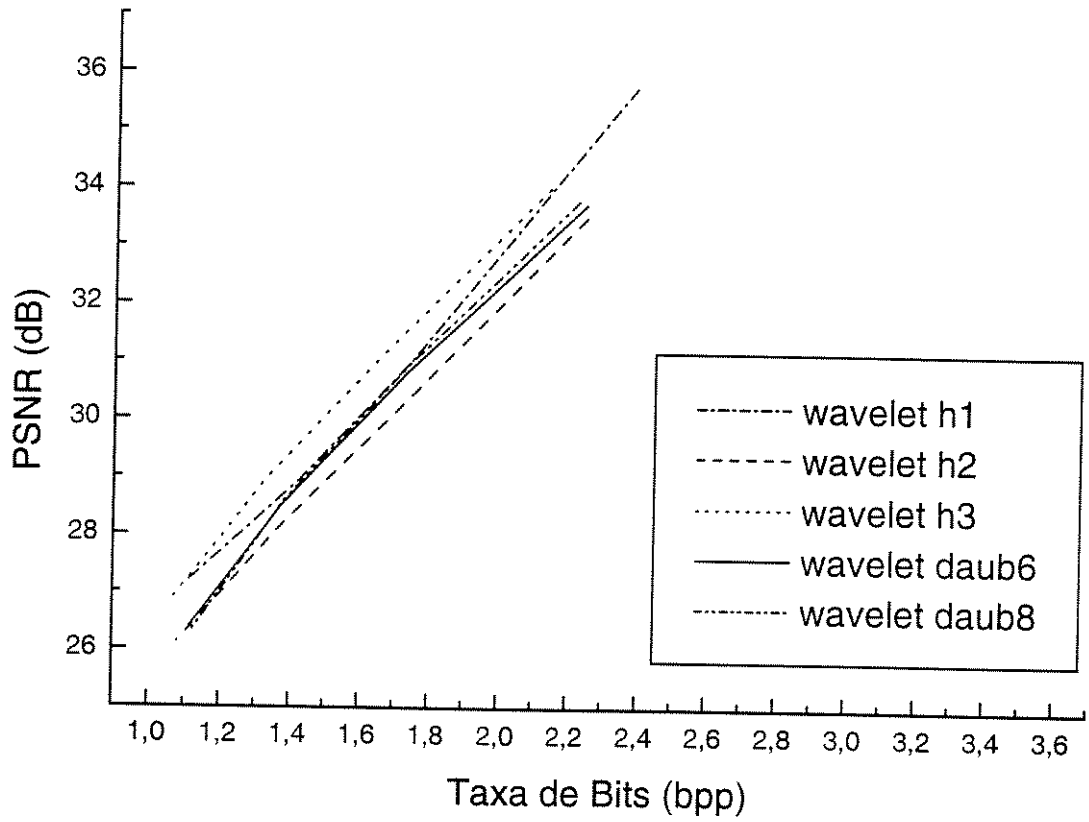



Figura A.3: Comparação entre os valores de PSNR para as wavelets $h_1$, $h_2$, $h_3$, *daub*6 e *daub*8, resultantes de um esquema com três camadas de decomposição e o vetor $q_3$.

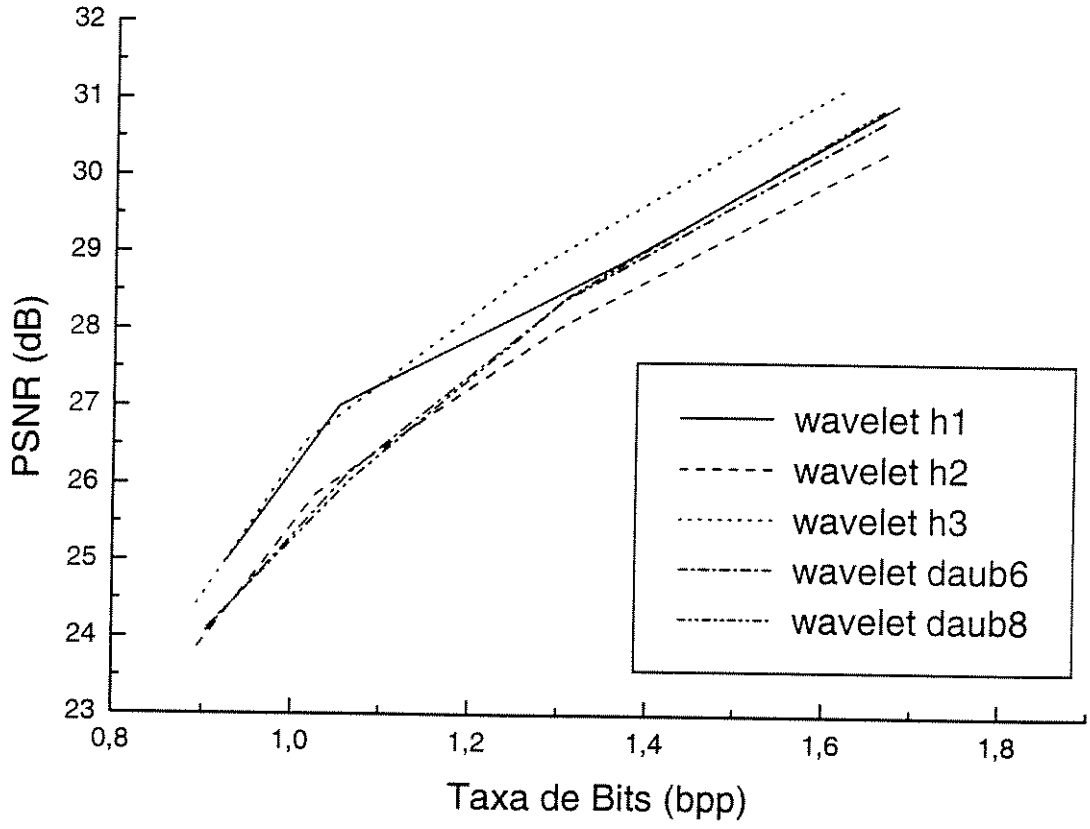



Figura A.4: Comparação entre os valores de PSNR para as wavelets $h_1$, $h_2$, $h_3$, *daub*6 e *daub*8, resultantes de um esquema com três camadas de decomposição e o vetor $q_4$.

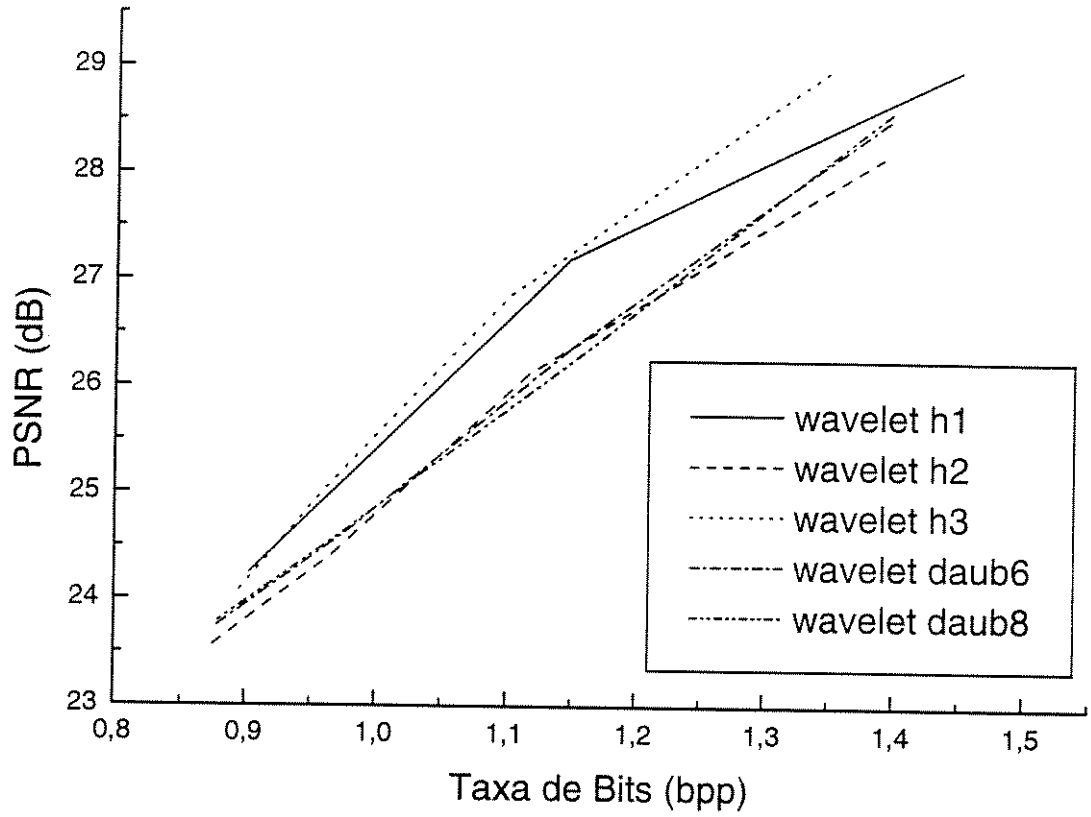



Figura A.5: Comparação entre os valores de PSNR para as wavelets $h_1$, $h_2$, $h_3$, *daub*6 e *daub*8, resultantes de um esquema com três camadas de decomposição e o vetor $q_5$.

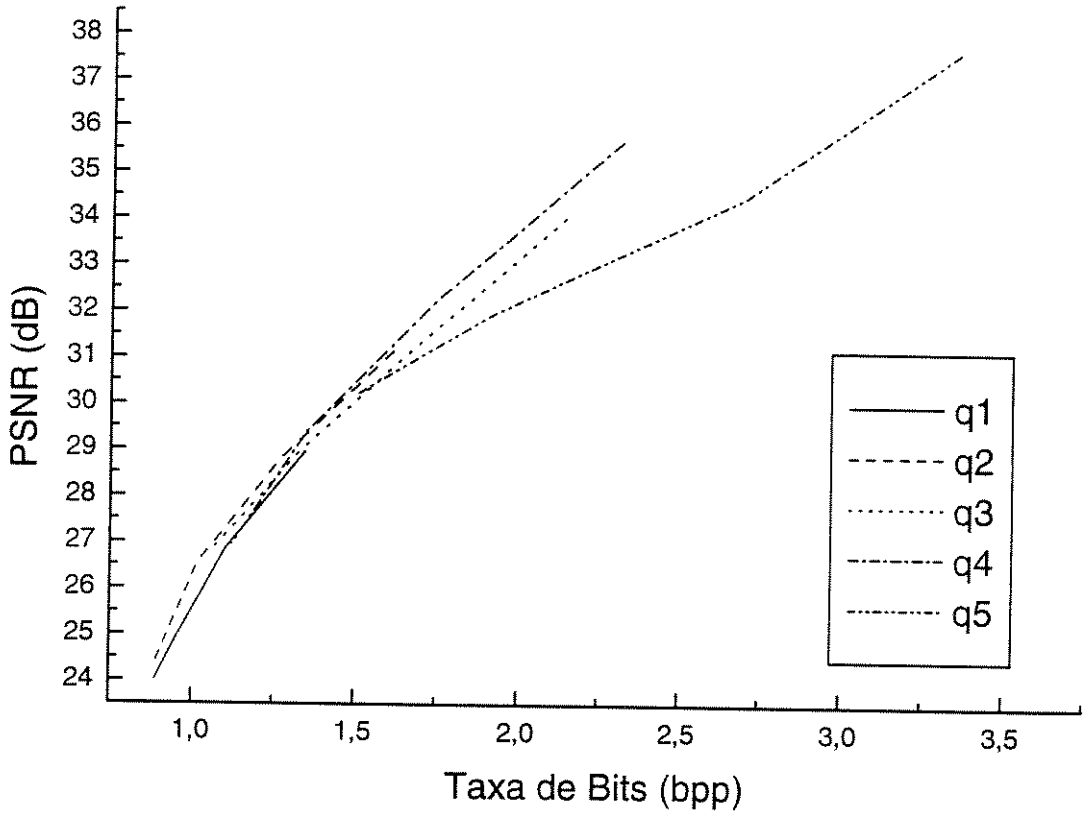



Figura A.6: Comparação entre os valores de PSNR resultantes de esquemas utilizando os vetores de quantização $q_1$, $q_2$, $q_3$, $q_4$ e $q_5$, com três camadas e a wavelet $h_3$.

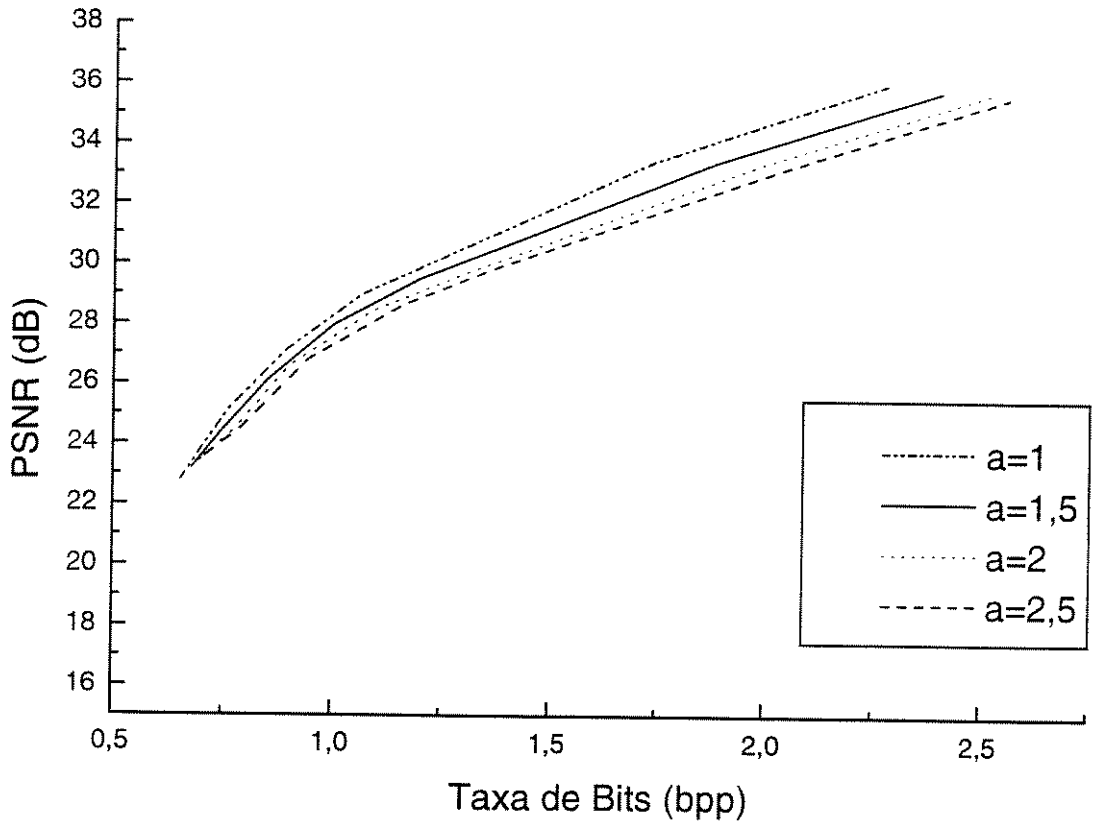



Figura A.7: Comparação entre os valores de PSNR para alguns valores de $a$, resultantes do esquema de compressão 2 com três camadas de decomposição.

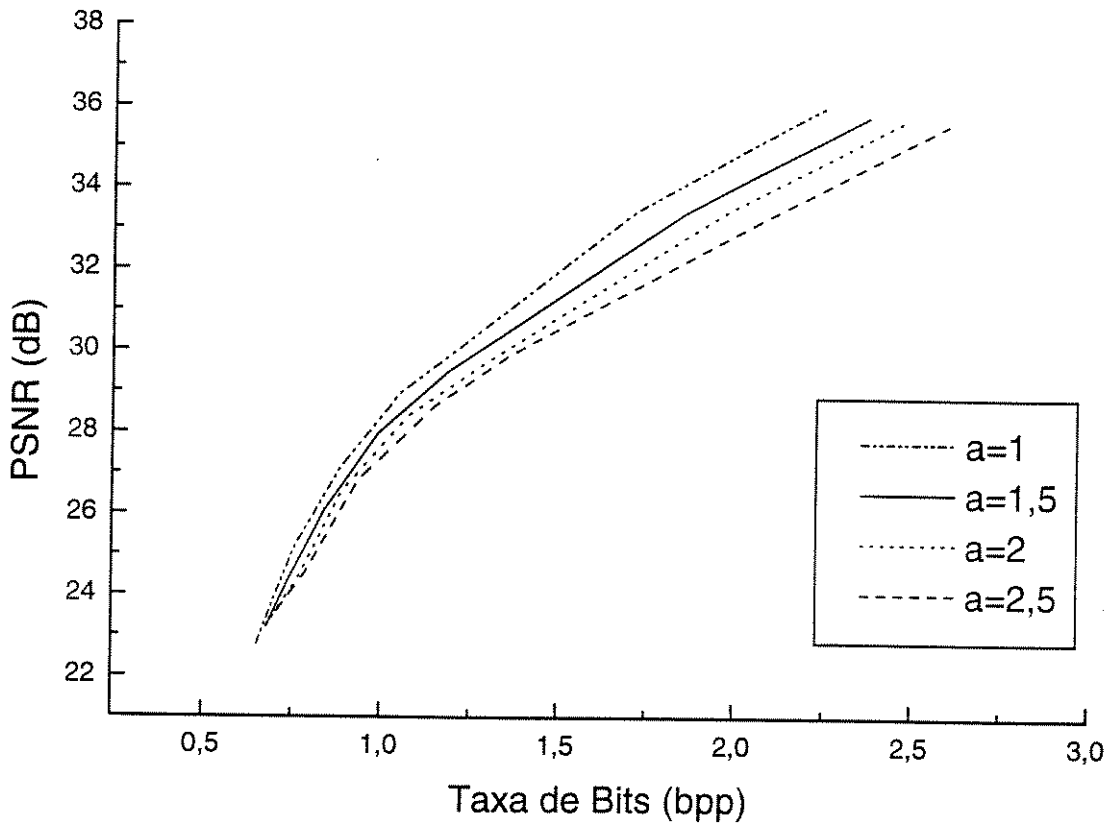



Figura A.8: Comparação entre os valores de PSNR para alguns valores de $a$, resultantes do esquema de compressão 3 com três camadas de decomposição.

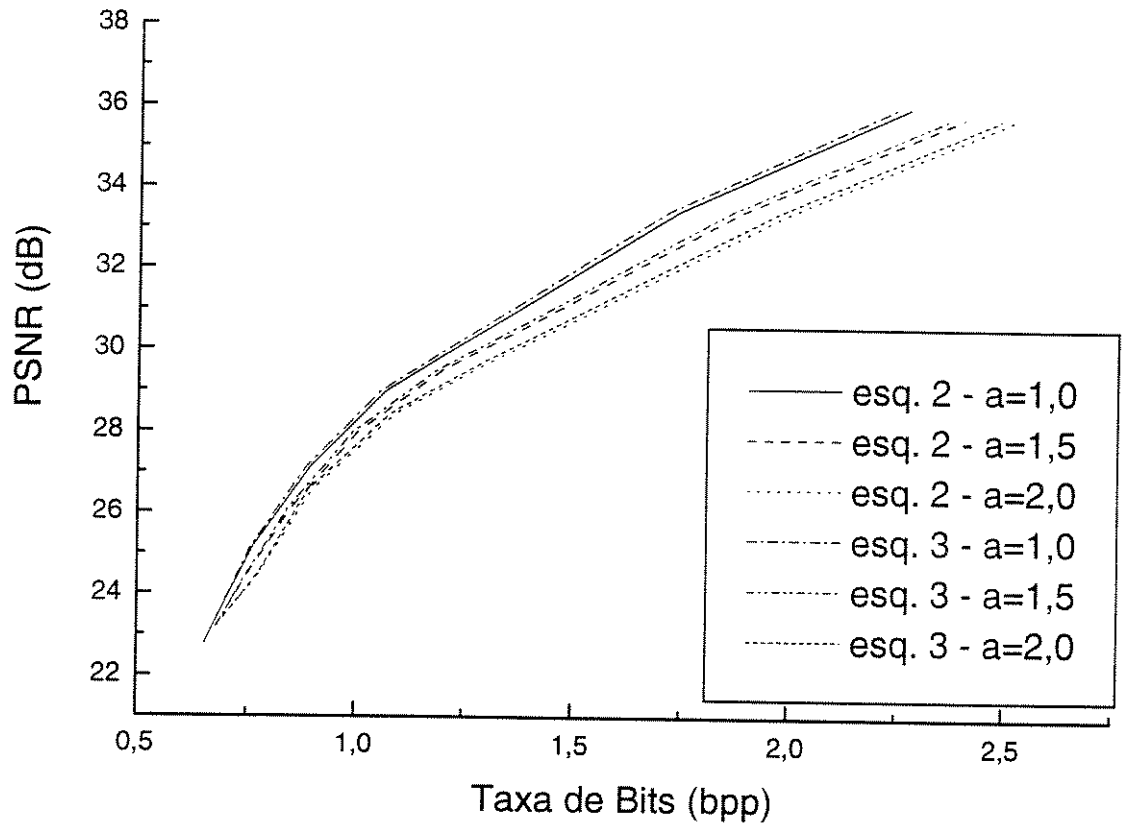



Figura A.9: Comparação entre os desempenhos dos esquemas de compressão 2 e 3.

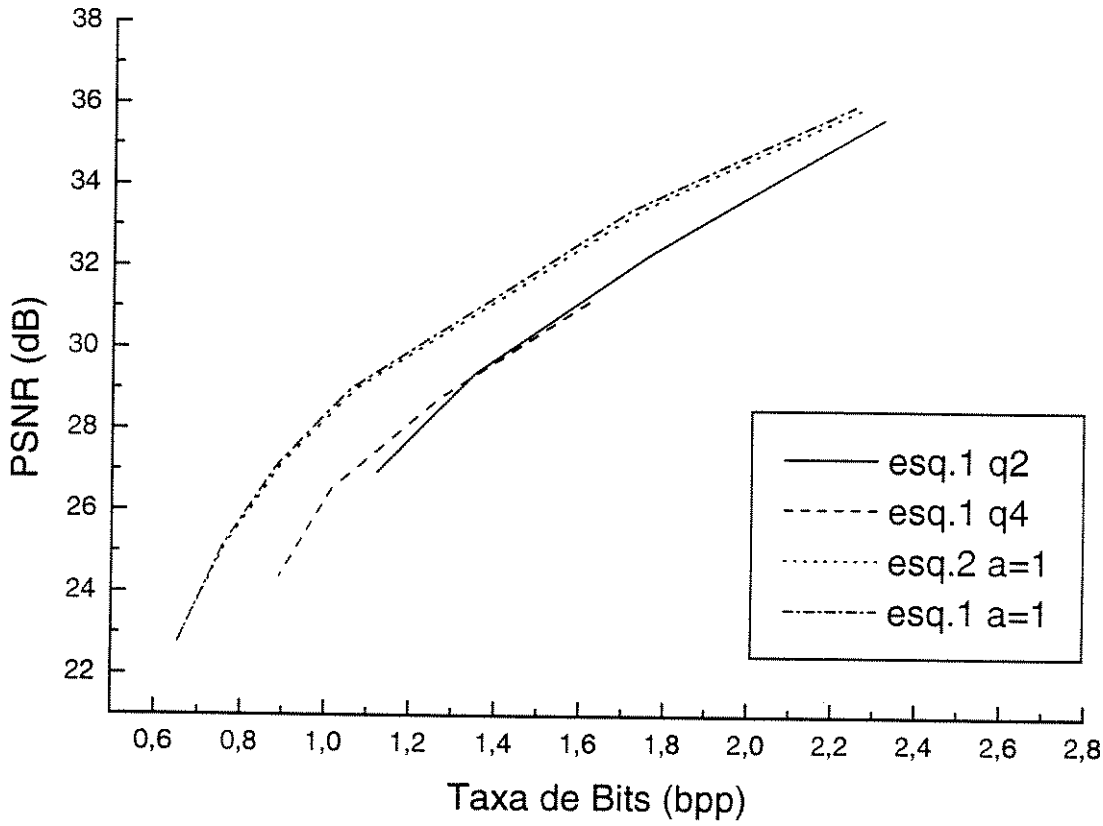
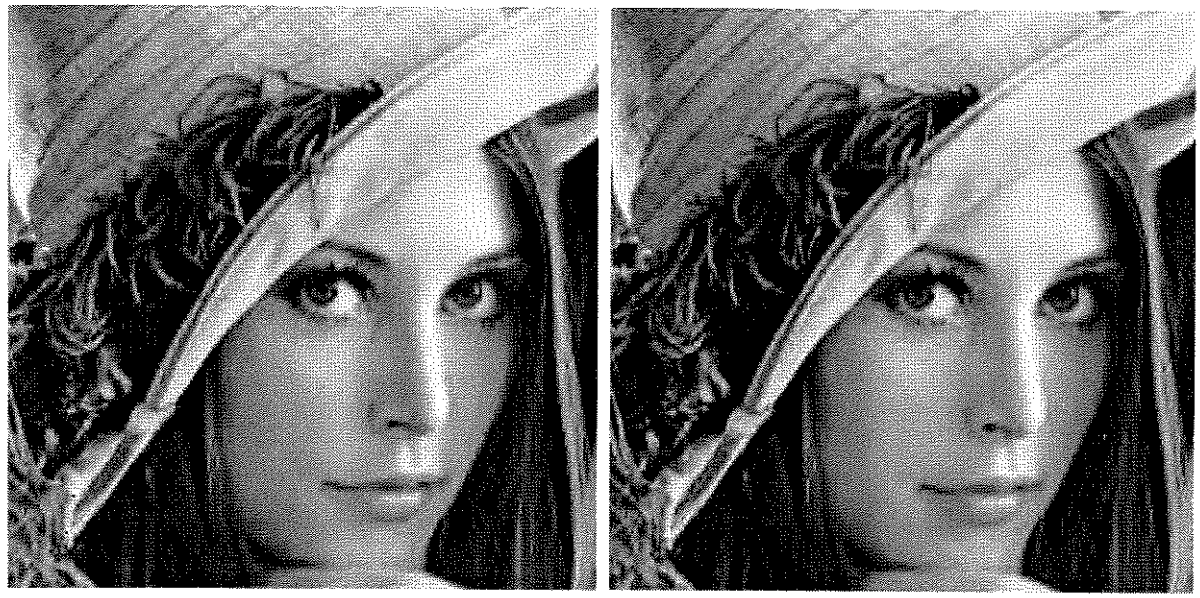



Figura A.10: Comparação entre os desempenhos dos esquemas de compressão 1, 2 e 3.

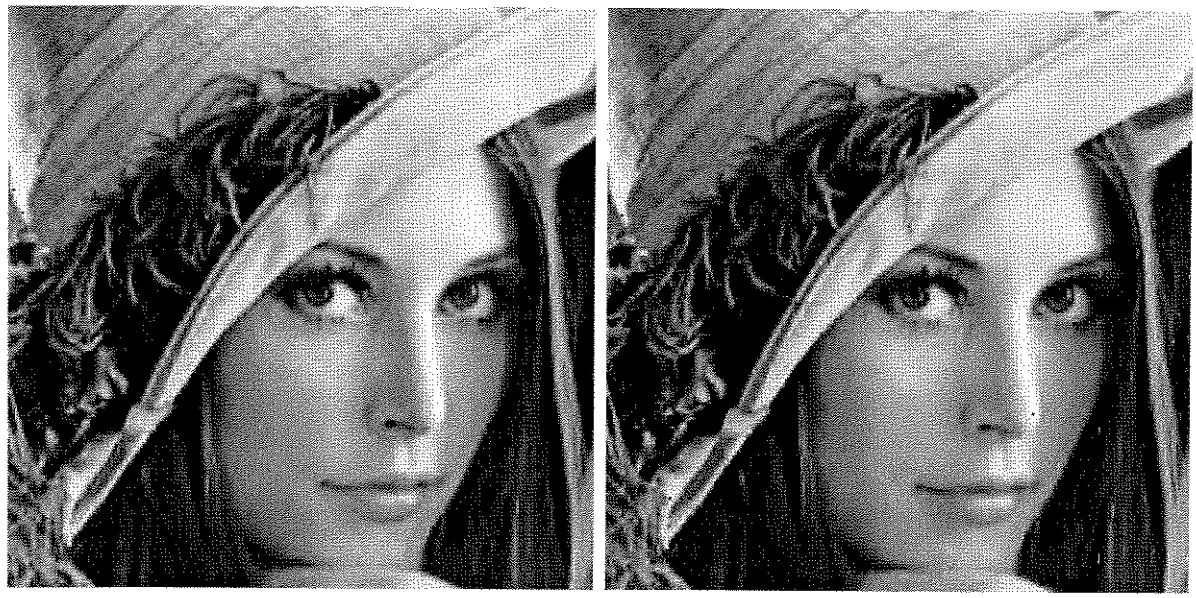
# Apêndice B

# Imagens



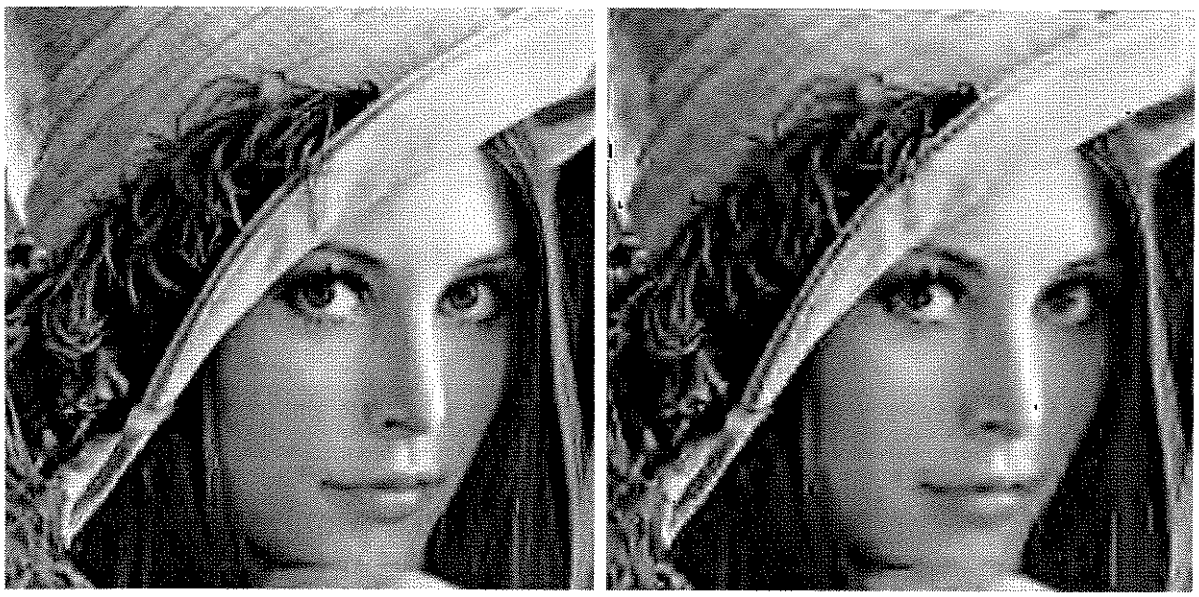
(a) (b)

Figura B.1: Imagens reconstruídas utilizando o vetor de quantização $\mathbf{q}_4$ e as wavelets (a) *daub*6 e (b) *daub*8.



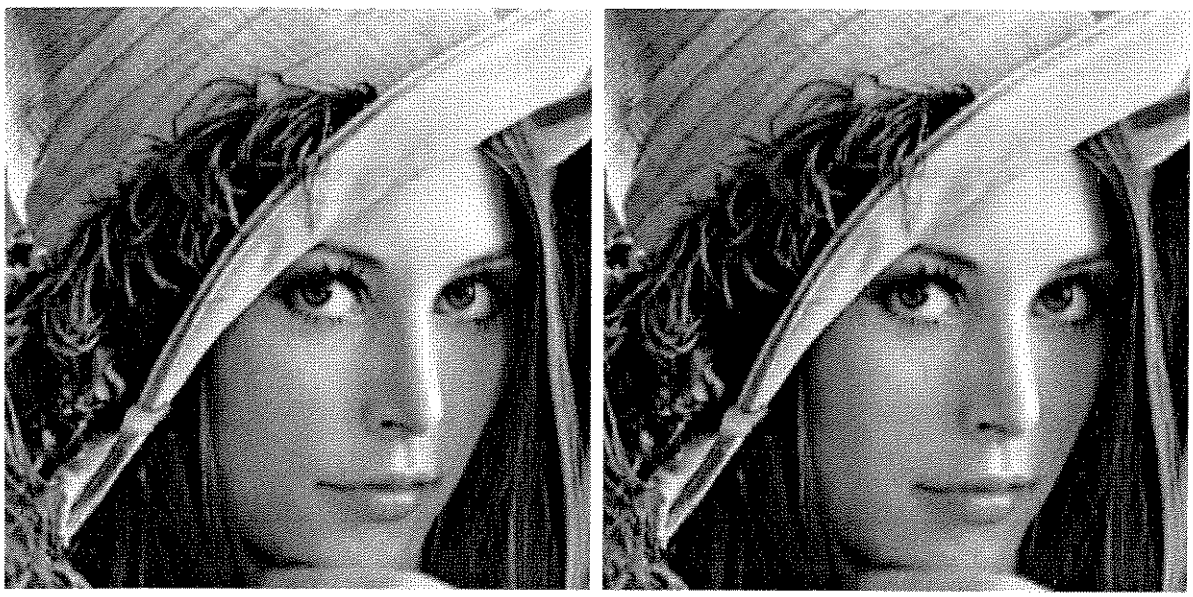
(a) (b)

Figura B.2: Imagens reconstruídas utilizando o vetor de quantização $\mathbf{q}_4$ e as wavelets (a) $h_1$ e (b) $h_2$.



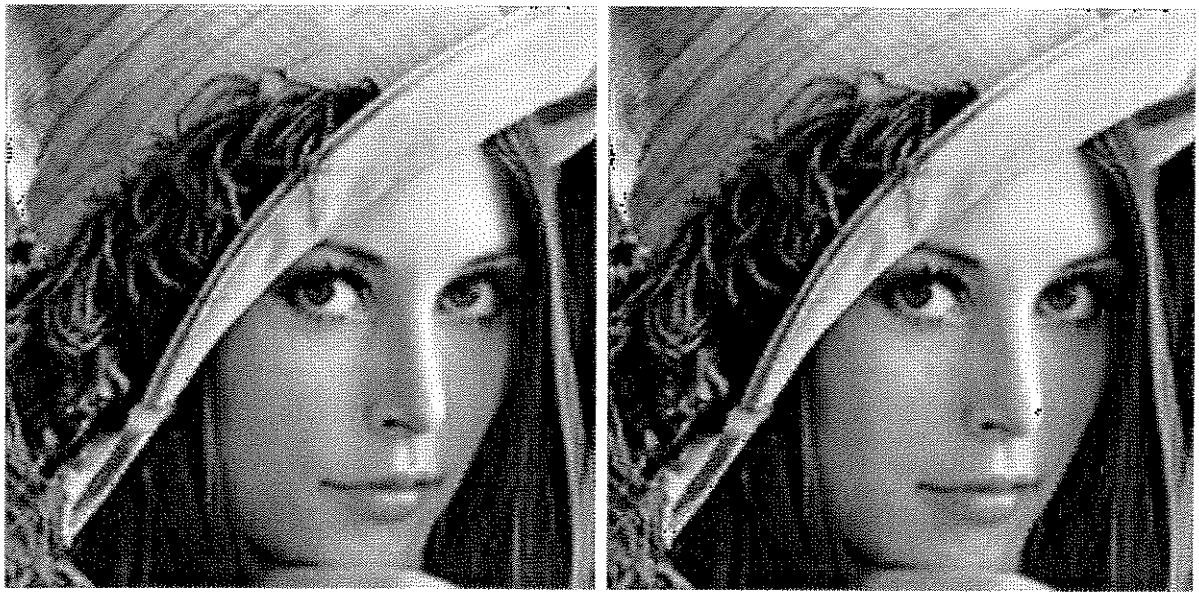
(a) (b)

Figura B.3: Imagens reconstruídas utilizando o vetor de quantização $q_4$, a wavelet $h_3$ e os fatores de quantização (a) 1 e (b) 4.



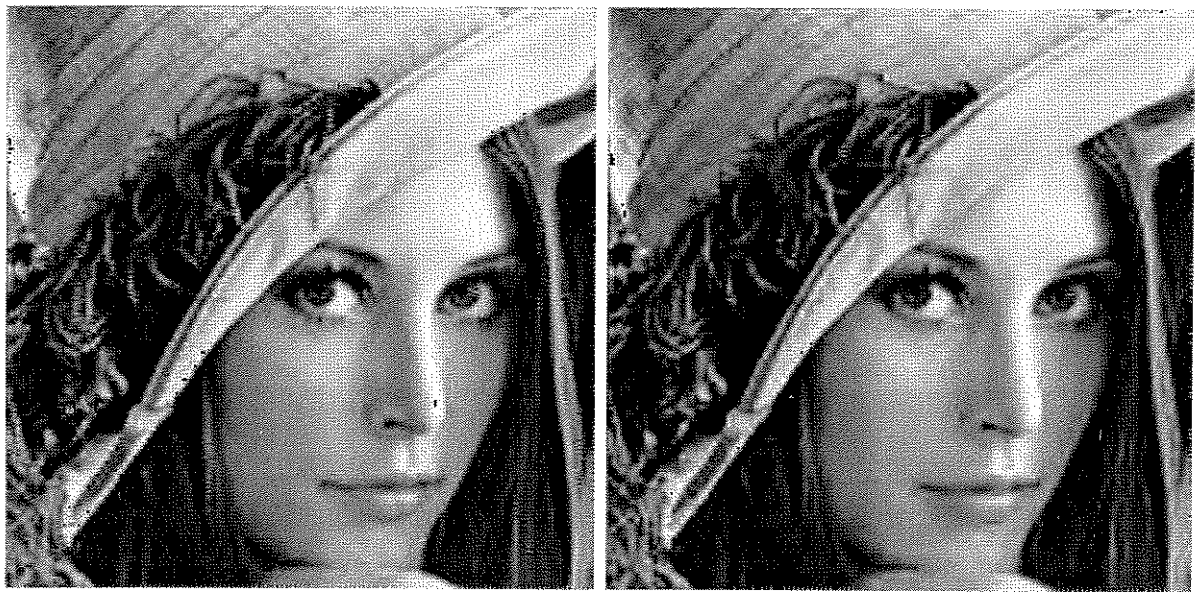
(a) (b)

Figura B.4: Imagens reconstruídas utilizando o vetor de quantização $q_3$ e $q_5$ e a wavelet $h_3$.



(a) (b)

Figura B.5: Imagens reconstruídas utilizando a wavelet $h_3$, 2 camadas e os vetores (a) $q_2$ e (b) $q_4$.

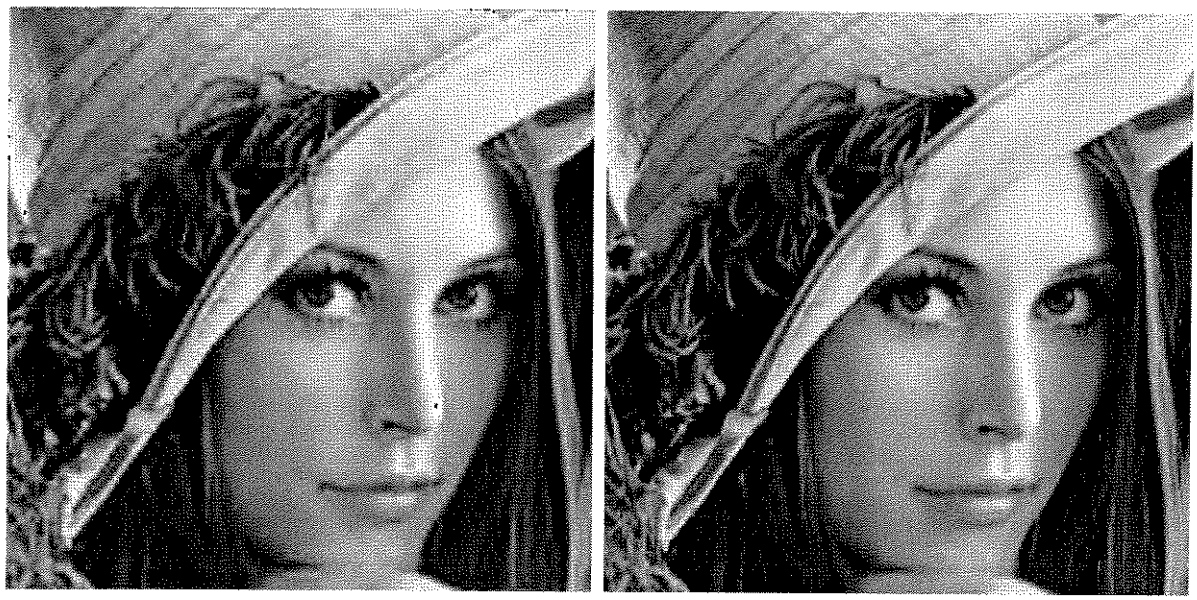


(a) (b)

Figura B.6: Imagens reconstruídas utilizando a wavelet $h_3$, 4 camadas e os vetores (a) $q_2$ e (b) $q_4$.

(a) (b)

Figura B.7: Imagens reconstruídas utilizando o alocador ótimo de bits, $a = 1$ e (a) $R_C = 2$ e (b) $R_C = 4$.

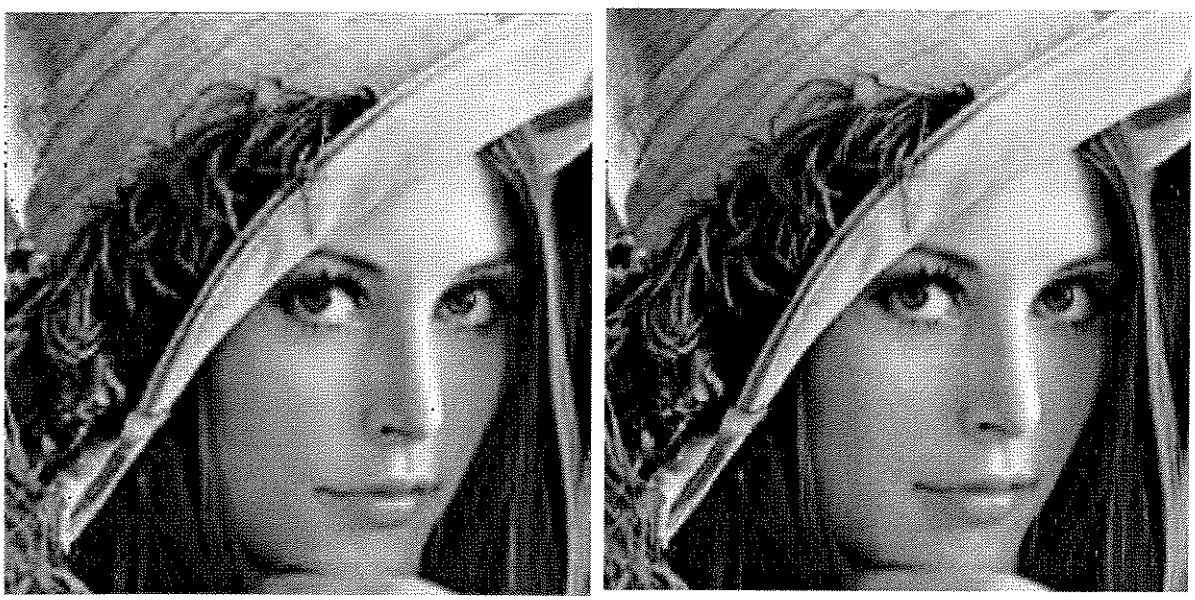


(a) (b)

Figura B.8: Imagens reconstruídas utilizando o alocador ótimo de bits, $a = 1$ e (a) $R_C = 6$ e (b) $R_C = 10$.

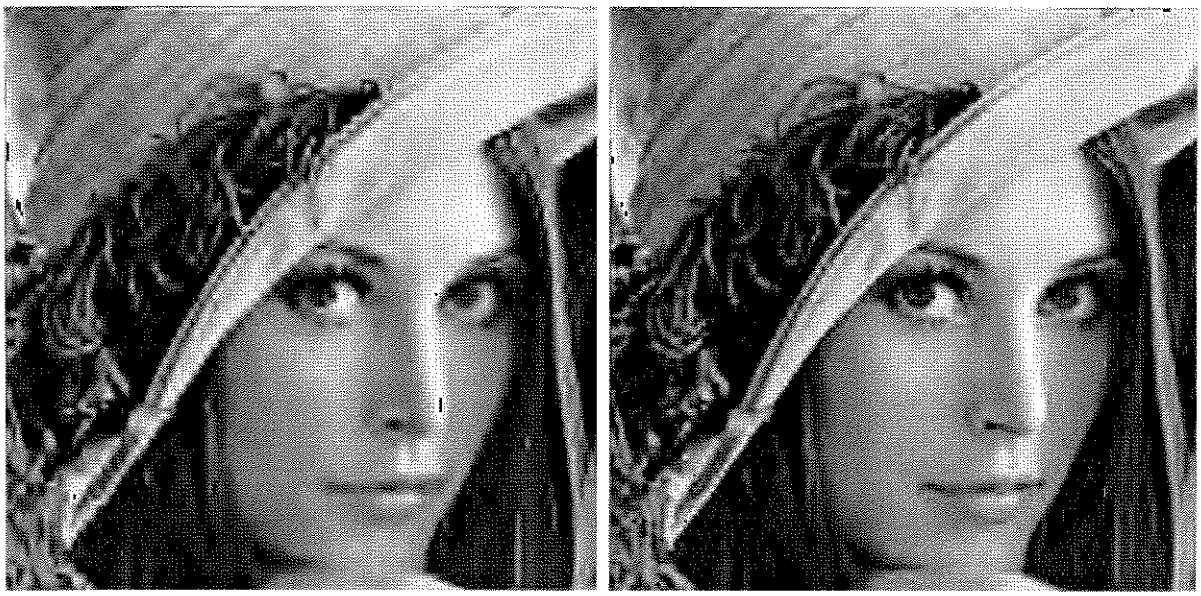
(a) (b)

Figura B.9: Imagens reconstruídas utilizando o alocador ótimo de bits, $a = 1,5$ e (a) $R_C = 2$ e (b) $R_C = 4$.



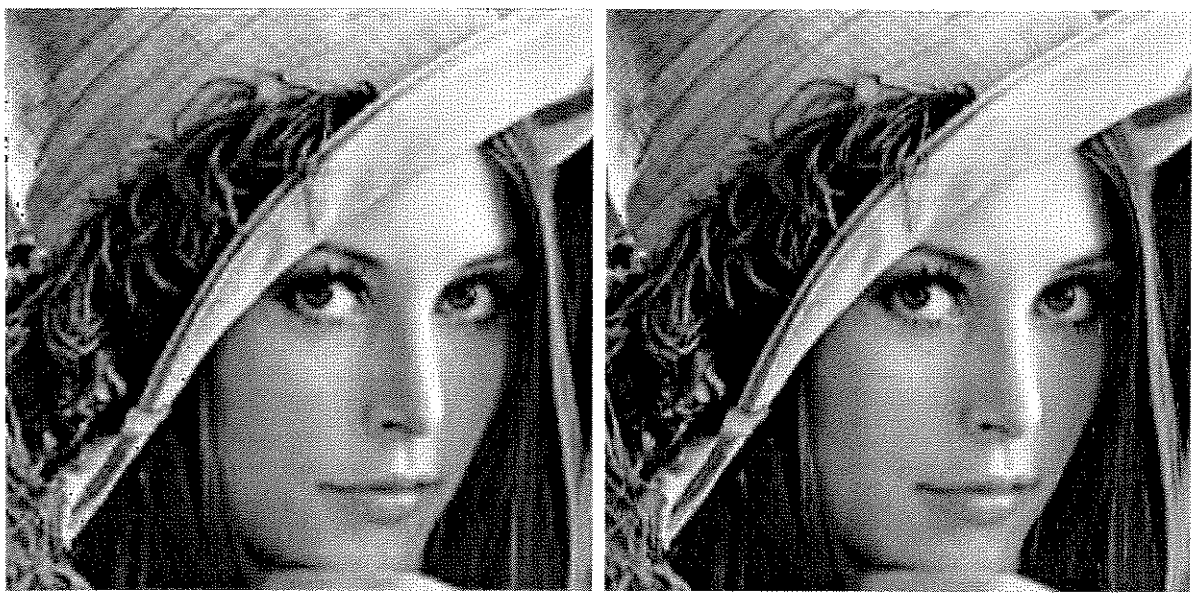
(a) (b)

Figura B.10: Imagens reconstruídas utilizando o alocador ótimo de bits, $a = 1,5$ e (a) $R_C = 6$ e (b) $R_C = 10$.



(a) (b)

Figura B.11: Imagens reconstruídas utilizando o alocador ótimo de bits, $a = 2$ e (a) $R_C = 2$ e (b) $R_C = 4$.

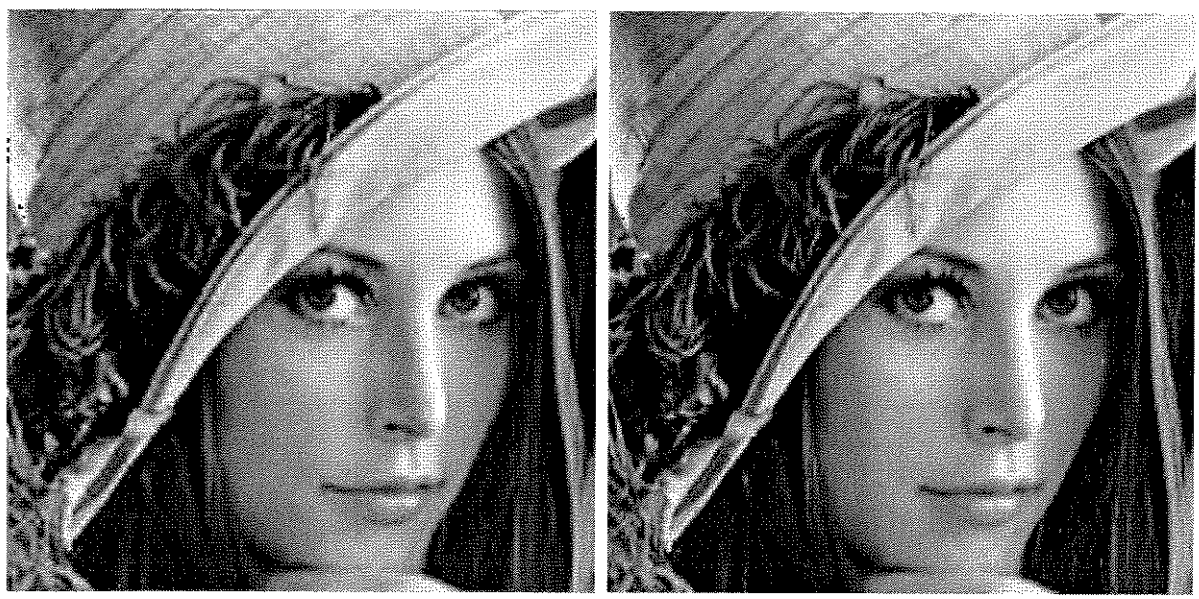


(a) (b)

Figura B.12: Imagens reconstruídas utilizando o alocador ótimo de bits, $a = 2$ e (a) $R_C = 6$ e (b) $R = 10$.

(a) (b)

Figura B.13: magens reconstruídas utilizando o alocador ótimo de bits, $a = 2,5$ e (a) $R_C = 2$ e (b) $R_C = 4$.



(a) (b)

Figura B.14: Imagens reconstruídas utilizando o alocador ótimo de bits, $a = 2,5$ e (a) $R_C = 6$ e (b) $R_C = 10$.

# Bibliografia


[Abramson] N. Ambranson, *Information Theory and Coding.* McGraw-Hill Book Company, 1963.

[Adelson] E. H. Adelson, E. Simoncelli e R. Hingorani, "Orthogonal Pyramid Transforms for Image Coding". In: *Proc. of SPIE.* (Cambridge, MA), Oct. 1987, vol. 845, pp. 50-58.

[Antonini] M. Antonini, M. Barlaud, P. Mathieu e I. Daubechies, "Image Coding Using Wavelet Transform". *IEEE Trans. on Image Processing*, vol. 1, no. 2, Apr. 1992.

[Barreto] C. S. Barreto e G. Mendonça, "Enhanced Zerotree Wavelet Transform Image Coding Exploiting Similarities Inside Bands". In: *Proc. of The IEEE International Conference on Image Processing.* (Lausanne, Switzerland), 16-19 Sep. 1996, vol. II, pp. 549-552.

[Bradley] J. N. Bradley, C. M Brinslawn e T. Hopper, "The FBI Wavelet/Scalar Quantization Standard for Gray-Scale Digitized Fingerprints Image Compression". In: *Visual Info. Process. II - SPIE.* (Orlando, FL), Apr. 1993.

[Brislawn] C. M Brislawn, J. N. Bradley e T. Hopper, "The FBI Compression Standard for Digitized Fingerprint Images". In: *Proceedings of The SPIE Conference.* 1996, Applications of Digital Image Processing XIX, pp. 2847.

[Burrus] C. S. Burrus, R. A. Gonipath, e H. Guo, *Introduction to Wavelet and Wavelet Transforms: A Primer.* Englewoods Cliffs, NJ: Prentice Hall, 1997.

[Burt81] P. J. Burt, "Fast Filter Transforms for Image Coding Processing". *Computer Graphics and Image Processing*, vol. 16, pp. 20-51, 1981.

[Burt83] P. J. Burt e E. H. Adelson, "The Laplacian Pyramid as a Compact Image Code". *IEEE Trans. Communications*, vol. 31, no. 4, pp. 532-540, Mar. 1983.

[Chui92] C. F. Chui, *An introduction to Wavelets.* San Diego: Academic Press Inc, 1992.

[Cohen92] A. Cohen, I. Daubechies e J.C. Feauveau, "Biorthogonal Bases of Compactly Supported Wavelets". *Communications on Pure and Applied Mathematics*, vol. 45, pp. 485-560, 1992.

[Coifman] R. Coifman e M. V. Wickerhauser, "Entropy-Based Algorithms for Best Basis Selection". *IEEE Trans. on Information Theory*, vol. 38, no. 2, pp. 713-718, Mar. 1992.

[Combes] J. M. Combes e A. Grossman, "Wavelets Time-Frequency Methods and Phase Space". In: *Proc. of the International Conference.* (Marseille, France), 14-18 Dec., 1987.

[Comer] M. L. Comer, K. Shen e E.J. Delp, "Rate-Scalable Coding Using a Zero-Tree Wavelet Approach". In: *Proceedings of The Ninth Image and Multidimensional Digital Signal Processing Workshop.* (Belize City, Belize), 3-6 Mar. 1996, pp. 162-163.

[Daubechies88] I. Daubechies, "Orthornomal Bases of Compactly Supported Wavelets". *Comm. Pure Appl. Math.*, vol. XLI, pp. 909-996, 1988.

[Daubechies90] I. Daubechies, "Orthornomal Bases of Compactly Supported Wavelets II - Variations on a Theme". *Tech.Report TM 11217-89111617*, ATT&Bell Lab., 1990.

[Daubechies92] I. Daubechies, *Ten Lectures on Wavelets.* Philadelphia, Pennsylvania: Siam, 1992.

[Farias] M. C. Q. de Farias e A. Lopes, "Introdução a Wavelets". Campinas: FEEC, UNICAMP, 1996. Relatório Técnico, Pub. FEEC.

[Farias96] M. C. Q. de Farias e A. Lopes, "Wavelets para Imagens". Campinas: FEEC, UNICAMP, 1996. Relatório Técnico, Pub. FEEC.

[Farias97] M. C. Q. de Farias e A. Lopes, "Wavelets para Compressão de Imagens". Campinas: FEEC, UNICAMP, 1997. Relatório Técnico, Pub. FEEC.

[Farias98] M. C. Q. de Farias e A. Lopes, "Arquiteturas Escalonáveis Segundo o Padrão MPEG-2/Vídeo". Campinas: FEEC, UNICAMP, 1998. Relatório Técnico, Pub. FEEC.

[Fournier] A. Fournier, *Wavelets: Algorithms and Aplications in Computer Graphics.* SIGGRAPH'94 Course Notes.

[Franca] A. C. França, "Um Codec para Imagens Paradas Baseado em Transformada Discreta de Wavelets e Quantização Linear Adaptativa". In: *Anais do XIV Simpósio Brasileiro de Telecomunicações.* (Curitiba, Brazil), Jul. 1996, vol. 1, p. 67-72.

[Gabor] D. Gabor, "Theory of Communication". *Journal of the IEEE*, vol. 33, pp. 429-457, 1946.

[Gersho] A. Gersho e R. M. Gray, "Vector Quantization and Signal Compression". Kluwer Academic Publishers, 1991.

[Gharavi] H. Gharavi, "Sub-Band Coding Algorithms for Video Applications: Videophone to HDTV Conferencing". *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 1, pp. 174-183, 1991.

[Gonzalez] R. C. Gonzalez e R. E. Woods, *Digital Image Processing.* Addison-Wesley Publishing Company, 1992.

[Grossman] A. Grossman e J. Morlet, "Decomposition of Hardy Functions Integrables Wavelets of Constant Shape". *SIAM J. Math. Anal.*, vol. 15, pp. 723-736, 1984.

[Hamming] R.W. Hamming, "Error Detecting and Error Correcting Codes". *Bell Sys. Tech. J.*, vol. 29, pp. 147-160.

[Haskell96] B. G. Haskell, A. Puri e A. Netravali, *Digital Video: An Introduction to MPEG-2.* International Thomson Publishing, 1996.

[Heller] P. N. Heller, V. Strela, G. Strang, P. Topiwala, C. Heil e L. S. Hills, "Multiwavelet Filter Banks for Data Compression". In: *Proc. of the IEEE International Symposium on Circuits and Systems.* pp. 1796-1799, 1995.

[Herley] C. Herley e M. Vetterli, "Wavelets and Recursive Filters Banks: Theory and Design". *IEEE Trans. on Signal Processing*, vol. 41, no. 8, Aug. 1993.

[Huanga] T. S. Huang e A. B. S. Hussian, "Facsimile Coding by Skipping White". *IEEE Trans. Comm.*, vol. COM-23, no. 12, pp. 1452-1466.

[Huangb] T. S. Huang, "Run-Length Coding and Its Extensions", *EG&G Tech. Report*, no. B-3742.

[ITU-T] International Organization for Standardisation - MPEG (Moving Pictures Expert Group), *ISSO/IEC 13818-2: Generic Coding of Moving Pictures and Associated Audio Information: Video.* 1994.

[Jain81] A. K. Jain, "Image Data Compression: A Review". *Proc. IEEE*, vol. 69, no. 3, Mar. 1981.

[Jain88] A. K. Jain, *Fundamental of Digital Image Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1988.

[Jayant] N. S. Jayant, *Waveform Quantization and Coding.* New York: IEEE Press, 1976.

[Jones] C. B. Jones, "An Efficient Coding System for Long Source Sequences". *IEEE Trans. Info. Theory*, vol. IT-27, no. 3, pp. 280-291, 1981.

[Jung] H. M. Jung, Y. Kim, S. Rhee, H. Song e K. T. Park, "HD-VCR Codec for Video Studio Application Using Quadtree Structured Binary Symbols in Wavelet Transform Domain". *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 6, no. 5, Oct. 1996.

[Langdon81] G. G. Langdon e J.J. Rissanen, "Compression of Black-White Images with Arithmetic Coding". *IEEE Trans. Comm.*, vol. COM-29, no. 6 , pp. 858-867, 1981.

[Langdon84] G. G. Langdon, "An Introduction to Arithmetic Coding". *IBM J. Res. Dev.*, vol. 28, no. 2, pp. 135-149, 1984.

[Lemarie] P. G. Lemarié, "Une Nouvelle Bases d'Ondelettes de $L^2(\mathbb{R})$". *J. de Math. Pure e Appl.*, vol. 67, pp. 227-236, 1988.

[Lian] K. C. Lian, J. Li e C. C. J. Kuo, "Image Processing with Embedded Multiwavelet Coding". In: *Proc. of SPIE.* (Orlando, FL), Apr. 1996, Wavelet Application III, pp. 165-176.

[Lim] J. S. Lim, *Two-Dimensional Signal and Image Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1989.

[Lloyd] S.P. Lloyd, "Least Squares Quantization in PCM". *IEEE Trans. on Information Theory*, vol. IT-28, pp. 129-137, Mar. 1982.

[Mallat89a] S.Mallat, "Multiresolution Approximation and Wavelet Orthonormal Bases of $L^2(\mathbb{R})$". *Trans. Amer. Math. Soc.*, Jun. 1989.

[Mallat89b] S. Mallat, "Multiresolution Approximation and Wavelets". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, Jul. 1989.

[Mallat89c] S. Mallat, "Multifrequency Channel Decompositions of Images and Wavelets Models", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 37, no. 12, Dec. 1989.

[Martucci] S. A. Martucci, I. Sodagar, T. Chiang e Y. Q.Zhang, "A Zerotree Wavelet Video Coder". *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 7, no. 1, pp. 109-118, Feb. 1997.

[Max] J. Max, "Quantizing for Minimum Distortion", *IEEE Trans. on Information Theory*, pp. 7-12, Mar. 1960.

[Meyer92] Y. Meyer, *Wavelets: algorithms and aplications.* Siam, 1992.

[Mitchell] J. L. Mitchell, W. B. Pennebaker, C. E. Fogg e D. J. LeGall, *MPEG Video Compression Standard.* In: Digital Multimedia Standard Series. Chapman & Hall, 1996.

[Morettin] P. A. Morettin, "Ondaletas e Seus Usos na Estatística". In: 1ª Escola de Séries Temporais e Econometria. (Canela - Brasil), Ago. 1997.

[Morlet] J. Morlet, *Sampling Theory and Wave Propagation.* In: C. H. Chen, Issues in Acoustic Signal/ Image Processing and Recognition, NATO Series, vol.1. Berlin: Springer-Verlag, pp.233-261.

[Netravali] A. N. Netravali e B. G. Haskell, *Digital Pictures: Representation and Compression Applications of Communications Theory.* In: R. W. Lucky, Series Editor. New York: Plenum, 1988.

[Ohta] M. Ohta e S. Nogaki, "Hybrid Picture Coding with Wavelet Transform and Overlapped Motion-Compensated Interframe Prediction Coding". *IEEE Trans. on Signal Processing*, vol. 41, no. 12, pp. 3416-3424, Dec. 1993.

[Oppenheim] A.V. Oppenleim e R.W. Schafer, *Digital Signal Processing.* Englewood Cliffs, NJ: Prentice Hall, 1975.

[Pennebaker] W. B. Pennebaker, J. L. Mitchel, G. G. Langdon e R. B. Arps, "An Overview of the Basic Principles of the Q-coder Adaptive Binary Arithmetic Coder". *IBM J. Res. Dev.*, vol. 32, no. 6, pp. 717-726.

[Porat] B. Porat, *Digital Processing of Random Signals.* Pennsylvania: Prentice Hall, 1992.

[Rabbani] M. Rabbani e P. W. Jones, *Digital Image Compression Techniques.* SPIE Optical Engeneering Press, 1991.

[Ramchandra] K. Ramchandra e M. Vetterli, "Best Wavelet Packet Bases in a Rate Distortion Sense", *IEEE Trans. on Image Processing*, vol. 2, no. 2, pp. 160-175, 1995.

[Rissanen] J. Rissanen e G. G. Langdon, "Arithmetic Coding". *IBM J. Res. Dev.*, vol. 23, no. 2 , pp. 149-162, 1979.

[Strela96a] V. Strela, P. N. Heller, G. Strang, P. Topiwala e C. Heil, "The Application of Multiwavelet Filter Banks to Image Processing". *Technical Report*, MIT, Jan. 1996.

[Rioul] O. Rioul e M. Vetterli, "Wavelet and Signal Processing". *IEEE Signal Processing Magazine*, vol. 8, no. 4, pp. 14-38, Oct. 1991.

[Sanchez] S. G. Sanchez, N. G. Prelcic e S. J. G. Galán, *Uvi-Wave Wavelet Toolbox for use in Matlab.* Departamento de Tecnoloxías das Comunicacións - Universidade de Vigo, Jul. 1996.

[Said] A. Said e W. A. Pearlman, "A New, Fast, and Efficiente Image Codec Based on Set Partitionin Hierarchical Trees". *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 6, no. 3, pp. 243-250, Jun. 1996.

[Shapiro] J. M. Shapiro, "Embedded Image Coding Using Zerotrees of Wavelet Coefficients". *IEEE Trans. Signal Processing*, vol. 41, pp. 3445-3462, Dec. 1993.

[Strang96] G. Strang e T. Nguyen, *Wavelets and Filter Banks.* Wellesley-Cambridge Press, 1996.

[Strela96b] V. Strela, *Multiwavelets: Theory and Applications.* PhD Thesis. Dept. of Mathematics, MIT, Jun. 1996.

[Vaidayanathan93] P.P. Vaidayanathan, *Multirate Systems and Signal Processing.* Englewoods Cliffs, NJ: Prentice Hall, 1993.

[Vaidayanathan87] P.P. Vaidayanathan, "Quadrature Mirror Filter Banks, M-band Extensions and Perfect Reconstruction Techniques". *IEEE ASSP Magazine*, pp. 4-20, Jul. 1987.

[VanDyck] R. E. Van Dyck, T. G. Marshall Jr., M. Chin e N. Moayeri, "Wavelet Video Coding With Ladder Structures and Entropy-Constrained Quantization". *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 6, no. 5, Oct. 1996.

[Vetterli92] M. Vetterli e C. Herley, "Wavelets and Filters Banks: Theory and Design". *IEEE Trans. on Signal Processing*, vol. 40, no. 9, Sep. 1992.

[Vetterli95] M. Vetterli e J. Kovacevic, *Wavelets and Subband Coding.* Englewood Cliffs, NJ:Prentice Hall, 1995.

[Vlachos] T. Vlachos, "Source Coding of Wavelet-Transformed Digital HDTV for Recording Applications". In: *IEEE Proc.-Vis Image Signal Processing.* vol. 143, no. 1, Feb. 1996.

[Walker] D. R. Walker e K. R. Rao, "Motion-Compensated Coder". *IEEE Trans. on Comm.*, vol. COM-35, no. 11, Nov. 1987.

[Wang] Q. Wang e M. Ghanbari, "Scalable Coding of Very High Resolution Video Using the Virtual Zerotree". *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 7, no. 5, pp.719-727, Oct. 1997.

[Witten] I. H. Witten, R. M. Neal e J. G. Cleary, "Arithmetic Coding for Data Compression". *Comm. of ACM*, vol. 30, no. 6, pp. 520-540, 1987.

[Yamazaki] Y. Yamazaki, Y. Wakahara e H. Teramura, "Digital Facsimile Equipment Quick-FAX': Using A New Redundancy Reduction Tecnique". In: *NTC '76.* 1976, pp. 621-625.

[Young80] R. M. Young, *An Introduction to Nonharmonic Fourier Series.* Ohio: Academic Press, 1980.

[Young92] R. K. Young, *Wavelet Theory and its Aplications*, Kluwer Academic Publishers, 1992.

[Zhang] Y. Q. Zhang e S. Zafar, "Motion-Compensated Wavelet Transform Coding for Color Video Compression". *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 2, no. 3, pp. 285-296, Sept. 1992.