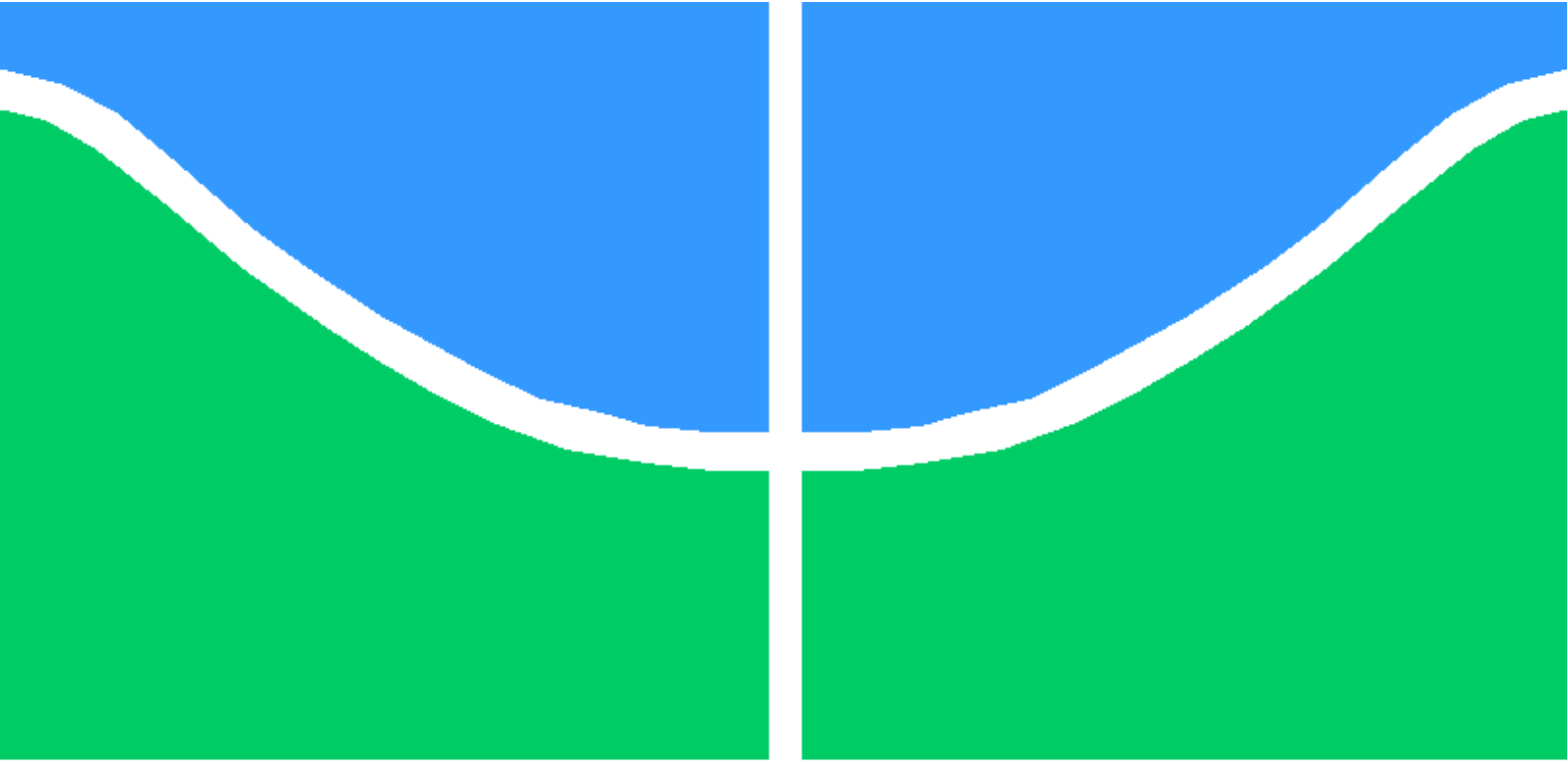


DISSERTAÇÃO DE MESTRADO EM ENGENHARIA ELÉTRICA

AVALIAÇÃO DO DESEMPENHO DE TRANSFORMADAS SOBREPOSTAS E WAVELETS NOS CODIFICADORES PADRÃO JPEG2000 E H.264/AVC

Rafael Galvão de Oliveira

Brasília, março de 2008

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia

DISSERTAÇÃO DE MESTRADO EM ENGENHARIA ELÉTRICA

AVALIAÇÃO DO DESEMPENHO DE TRANSFORMADAS SOBREPOSTAS E WAVELETS NOS CODIFICADORES PADRÃO JPEG2000 E H.264/AVC

Rafael Galvão de Oliveira

Dissertação de mestrado submetida ao Departamento de Engenharia Elétrica da Faculdade de Tecnologia da Universidade de Brasília, como parte dos requisitos necessários para a obtenção do grau de mestre.

Banca Examinadora

Prof. Ricardo Lopes de Queiroz, PhD. UnB/ ENE (Orientador) _____

Prof. Lúcio Martins Silva, Dr. UnB/ ENE (Examinador Interna) _____

Prof. Eduardo Antônio Barros da Silva, PhD. UFRJ/ COPPE (Examinador Externo) _____

FICHA CATALOGRÁFICA

DE OLIVEIRA, RAFAEL GALVÃO
Avaliação do Desempenho de Transformadas Sobrepostas nos Codificadores Padrão JPEG2000 E H.264/AVC. [Distrito Federal] 2008.
xvii, 82p., 297 mm (ENE/FT/UnB, Mestre, Telecomunicações Processamento de Sinais, 2008). Dissertação de Mestrado.
Universidade de Brasília. Faculdade de Tecnologia.
Departamento de Engenharia Elétrica.

1. Compressão de imagens	2. JPEG2000
3. H.264	4. Transformada *Wavelet*
5. Transformadas Sobrepostas	6. Processamento de Sinais
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

OLIVEIRA, R. G.(2008) Avaliação do Desempenho de Transformadas Sobrepostas nos Codificadores Padrão JPEG2000 E H.264/AVC. Dissertação de Mestrado em Engenharia Elétrica com enfase em Telecomunicações, Publicação MTARH.DM - 336/08, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 82p

CESSÃO DE DIREITOS

NOME DO AUTOR: Rafael Galvão de Oliveira.

TÍTULO DA DISSERTAÇÃO DE MESTRADO: Avaliação do Desempenho de Transformadas Sobrepostas nos Codificadores Padrão JPEG2000 E H.264/AVC.

GRAU / ANO: Mestre / 2008

Rafael Galvão de Oliveira
SQS 310 Bloco F Ap. 302
70.363-060 Brasília - DF - Brasil.

Dedicatória

À Érica

Rafael Galvão de Oliveira

Agradecimentos

Agradeço ao meu orientador Ricardo Lopes de Queiroz pela sua supervisão e paciência.
Aos meus colegas do GPDS Bruno, Diogo, Eduardo, Karen, Mintsu, Renan, Tiago e Zaghetto pela amizade e pelo agradável ambiente de trabalho.
À minha famílha Carlos, Eleonora, Gabriel e Larissa.
E à Érica que, além da ajuda e apoio neste últimos meses, esteve ao meu lado todos estes anos.

Rafael Galvão de Oliveira

RESUMO

O presente trabalho consiste em uma comparação do desempenhos de diversas transformadas, transformadas sobrepostas, *Wavelet* e DCT nos codificadores padrão JPEG2000 e H.264/AVC. Embora essas comparações já tenham sido realizada para diversos codificadores, não há muita informação a respeito do desempenhos relativos nesse nos codificadores utilizados neste trabalho. Além disso, no H.264/AVC, compara-se o desempenho com um outra forma de sobreposição, a predição intra quadros;

ABSTRACT

This work aims to evaluate the performance presented by lapped transforms, wavelets and DCT in standards JPEG2000 and H.264/AVC coders. Even though there is much of information about the comparison of those transforms in many coders, there is little information available in literature relative to the performs of the performs of these transforms in the coders used. It is also compared to other forms of overlaping as intra frame prediction, used in H.264/AVC coder.

LISTA DE SIGLAS, ABREVIAÇÕES E ACRÔNIMOS

Abreviações, Acrônimos e Siglas

AVC	*Advanced Video Coding*
CABAC	*Context-Based Adaptive Binary Arithmetic Coding*
CAVLC	*Context-Adaptive Variable Length Coding*
DCT	*Discret Fourier Transform*
DFT	*Discrete Fourier transform*
DVD	*Digital Versatile Disk*
DWT	*Discrete wavelet transform*
EBCOT	*Embedded Block Coding with Optimized Truncation*
FRExt	*Fidelity Range Extensions*
GB	Giga bytes (1073741824 bytes)
GenLot	*Generalized Lapped Orthogonal Transform*
GLBT	*Generalized Lapped Biorthogonal Transform*
HD	*High Definition*
IEC	*International Electrotechnical Commission*
ISO	*International Standards Organization*
ITU	*International Telecommunication Union*
JPEG	*Joint Photographic Experts Group*
JVT	*Joint Video Team*
LBT	*Lapped Biorthogonal Transform*
Lot	*Lapped Orthogonal Transform*
MP3	*MPEG-1 Audio Layer3*
MPEG	*Motion Picture Experts Group*
MSE	*mean square error*
MSE	*Mean squared error*
PSNR	*Peak Signal to Noise Ratio*
SAD	*Sum of absolute differences*
SD	*Standard Definition*
SPHIT	*Set Partitioning in Hierarchical Trees*
VCEG	*Video Coding Experts Group*
VLC	*Variable Length Code*

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO

Nos últimos anos muita pesquisa tem sido feita em compressão de imagens e vídeo. Diversas técnicas desenvolvidas são aplicadas tanto no armazenamento quanto na transmissão e o estado da arte são H.264/AVC[1] na compressão de vídeo e JPEG2000[2] na compressão de imagens.

Em 1956, a IBM lançou RAMAC 305 (*Random Access Method of Accounting and Control*), primeiro computador da história a possuir um disco rígido. Ele possuía 50 discos magnéticos que giravam a 1.200 rotações por minuto, media cerca de um metro e meio de largura, um metro e setenta centímetros de altura e setenta centímetros de profundidade e tinha a capacidade de armazenamento de cinco milhões de bytes (aproximadamente 4,77MB). O sistema completo custava cerca de cento e sessenta mil dólares americanos. Foi somente em 1958 que o RAMAC 305 pode utilizar um disco adicional dobrando a capacidade de armazenamento.

Atualmente, por algumas dezenas de reais é possível carregar no bolso dispositivos de armazenamento com capacidade de alguns gigabytes. Disco rígidos com capacidade da ordem de terabytes podem ser adquiridos por algumas centenas de dólares americanos. Discos ópticos como o DVD (*Digital Versatile Disc*) têm capacidade de até 8,5GB (caso apenas um lado seja utilizado com camada dupla) e está sendo considerado uma mídia ultrapassada. Os prováveis sucessores do DVD, o HD-DVD (*High Density Digital Versatile Disc*) e o Blu-ray tem capacidade ainda mais impressionantes. Para camadas duplas, o blu-ray tem capacidade de 50GB e o HD-DVD 30 GB. Isso sem mudar os 12cm de diâmetro do DVD. Recentemente, a Toshiba oficialmente desistiu de continuar a produção do HD-DVD, indicando uma possível vitória do Blu-ray como sucessor do DVD.

Sem dúvida houve uma imensa evolução dos dispositivos de armazenamento tanto em aumento de capacidade, como em diminuição de tamanho e custo. Isso levanta novamente o questionamento sobre a real necessidade de contínuo estudo sobre compressão de dados. Mais especificamente, no caso deste trabalho, sobre a compressão de imagens. Porém, o aumento de capacidade de armazenamento, foi acompanhado, senão superado, por um aumento na geração de informação. Exemplos disso são a popularização do uso de câmeras digitais, vídeos transmitidos por internet, a adoção de padrões de televisão digital, entre outros.

Um vídeo em definição padrão ou SD (do inglês, *Standard Definition*), 720×576 *pixels*, utilizada no sistema PAL europeu (*Phase Alternating Line*) com 25 quadros por segundos em um DVD de dupla face, considerando ainda que a imagem esteja em cores e que as crominâncias (detalhes sobre transformação de cores serão vistas a seguir) sejam subamostradas, possuindo apenas um quarto da definição do vídeo e que cada amostra ocupasse apenas um *byte*, poderia ter cerca de 690 segundos, aproximadamente 11 minutos. Mesmos utilizando um blu-ray esse mesmo vídeo não poderia ultrapassar 67 minutos. Todavia, discos Blu-ray tipicamente contém vídeos de até 1920×1080 *pixels* a 30 quadros por segundo. Sendo assim, a capacidade seria de apenas cerca de 10 minutos. Para um vídeo de duas horas em alta definição seriam necessários cerca de 625Gb, quantidade impressionante até para os padrões atuais.

Outro exemplo muito ilustrativo sobre a necessidade de compressão é o caso de imagens estáticas, nas quais costuma-se armazenar três amostras por *pixel*, uma para cada cor, vermelho, verde e azul. Considerando que cada amostra é armazenada utilizando-se oito bits, um byte, em uma câmera de sete *megapixels*, com capacidade de armazenamento de 1 Gb, somente 37 fotografias poderiam ser guardadas. Embora essa capacidade não deixe a desejar comparando-se com câmeras convencionais, nem se aproxima das centenas de imagens que poderiam ser armazenadas utilizando-se compressão.

Os dois exemplos apresentados demonstram a necessidade de utilização de técnicas de compressão de imagem e vídeo.

1.2 DEFINIÇÃO DO PROBLEMA

Ao utilizar técnicas de compressão, procura-se uma representação mais econômica de um sinal digital, ou seja, uma forma que consuma menos recursos, no caso, *bits*. Visando que essa representação seja eficiente, é interessante que, baseado em características conhecidas do sinal sendo codificado, sejam utilizados modelos. Por exemplo, caso o sinal sendo comprimido seja um texto escrito em língua portuguesa, sabe-se que é muito mais provável a ocorrência de uma letra “a” que de uma letra ‘w”. Assim pode-se, na codificação, representar o simbolo “a” de forma mais econômica que “w”. No mesmo exemplo, caso queira-se um modelo mais complexo, é possível dizer que, após a letra “p”, é mais provável a ocorrência de uma vogal “e” que a letra “t” (embora não seja impossível como visto na palavra “apto”). Da mesma forma, podem ser escolhidas representações para os símbolos de acordo com a vizinhança, no exemplo apresentado a letra “e” seria representada utilizando menos *bits* que “t”. Esse ultimo tipo de codificação, no qual a vizinhança já codificada é utilizada, é conhecido como contextual. É importante

notar que o modelo depende do sinal em questão, um modelo ótimo para textos em língua portuguesa não seria ótimo para um texto em língua inglesa, um sinal de áudio ou uma imagem.

As técnicas de codificação podem ser divididas em duas grandes classes: com perdas e sem perdas. Claro que não seria aceitável que quando se está comprimindo um texto que a versão disponível após a compressão fosse diferente da versão original. Caso apenas uma palavra mudasse, isso poderia modificar o sentido de uma frase ou mesmo de todo o texto, passando uma mensagem diferente da pretendida pelo autor. Da mesma forma, caso queira-se comprimir um arquivo executável, a mudança de apenas uma instrução poderia ocasionar resultados catastróficos. Nestes casos, utiliza-se um tipo de compressão conhecida como compressão sem perdas. Entretanto, na codificação de um sinal de áudio, como uma música, ou uma imagem é possível que se obtenha, após a codificação, uma versão ligeiramente diferente do original, compressão com perdas. Um exemplo pode ser apresentado na Figura 1.1. Obviamente a verão original é bem mais agradável, contudo, na versão, comprimida a uma taxa bem baixa para que se percebam diferenças entre a versão original, é possível ver e, inclusive, reconhecer todos os elementos pertencentes à imagem.

Embora não seja a única forma, o paradigma mais comum para compressão com perdas envolve transformadas. O sinal original é transformado com veremos no Capítulo 2 e os coeficientes gerados são armazenados com menor precisão do que a precisão com que são gerados (origem das perdas). Esse tipo de técnica permite que se explore características de como diversos sinais são percebidos por seres humanos. Por exemplo, no caso de sinais de áudio, pode ocorrer um fenômeno conhecido como mascaramento em freqüência, que ocorre quando um som que normalmente poderia ser ouvido é mascarado por outro, de maior intensidade, que encontra-se em uma freqüência próxima. Desta forma, pode não ser necessário armazenar dois coeficientes que representam freqüências muito próximas caso uma tenha maior intensidade. Uma técnica de compressão que explora esse fenômeno do sistema auditivo humano é o MP3 (*MPEG-1 Audio Layer 3*)[3]. O sistema visual humano possui percepções diferentes para as diferentes freqüências espaciais, permitindo que isso seja explorado pelas diversas técnicas de compressão de imagens.

1.3 OBJETIVOS

O presente trabalho consiste em uma comparação do desempenhos de algumas famílias de transformadas utilizadas em padrões de compressão de imagens como *Wavelets* e DCT com transformadas

(a)

(b)

Figura 1.1: Imagem Barbara: (a) imagem original (b) versão após compressão a uma taxa média de 0,2bpp, utilizando o padrão JPEG2000.

sobrepostas [4] [5]. Este tipo de comparação tem sido feito desde o inicio do desenvolvimento das transformadas sobrepostas em diversos codificadores como o JPEG [6] [7], SPIHT [8] [9], entre outros. Foi ainda uma comparação feita entre as transformadas citadas e o esquema de predição intra quadros, aliado a DCT inteira, utilizado no codificador padrão H.264/AVC. Não há muita informação disponível na literatura sobre esta ultima comparação.

1.4 APRESENTAÇÃO DO MANUSCRITO

O presente trabalho está dividido em seis capítulos sendo o primeiro esta introdução. No Capítulo 2, é apresentada, de forma introdutória, transformações que podem ser utilizadas em compressão de imagens. Em seguida, no Capítulo 3, os codificadores padrão utilizados no trabalho, o JPEG2000 e o H.264/AVC, são exibidos. O Capítulo 4 trata das modificações realizadas nos codificadores JPEG2000 e o H.264/AVC para permitir a utilização de transformadas sobrepostas e transformadas *wavelet*. Os resultados experimentais são analisados no Capítulo 5. Finalmente, no Capítulo 6, são apresentadas as conclusões.

2 TRANSFORMAÇÕES EM IMAGENS

Transformações são utilizadas em processamento de sinais para os mais diversos fins. Estimação de espectro, detecção de bordas e remoção de ruídos são apenas alguns exemplos. Em compressão de sinais estamos interessados na capacidade provida pela transformada de descorrelacionar as amostras, levando a um novo espaço e concentrando a energia do sinal.

2.1 TRANSFORMADAS EM IMAGENS

As transformadas que veremos podem ser abordadas como uma transformação linear, tipo particular de função entre dois espaços vetoriais. O núcleo da transformação possui as bases desse novo espaço vetorial e os valores dos coeficientes transformados são as projeções nessas bases. Uma forma bem geral de definir transformações em imagens é apresentada em:

$$\mathcal{F}(m_1, m_2) = \sum_{n_1=1}^{N_1} \sum_{n_2=1}^{N_2} F(n_1, n_2) A(n_1, n_2; m_1, m_2), \tag{2.1}$$

onde F é uma imagem de dimensão $N_1 \times N_2$ e $A(n_1, n_2; m_1, m_2)$ representa o núcleo da transformação direta.

É interessante notar que, na definição apresentada, as dimensões da matriz dos coeficientes transformados pode ser diferente da imagem. Além disso, não existe garantia de que exista uma transformação inversa. Para aplicação em compressão de imagens, é importante que essa transformação seja inversível. Vamos supor que exista transformação inversa e vamos defini-la como:

$$F(n_1, n_2) = \sum_{m_1=1}^{N_1} \sum_{m_2=1}^{N_2} \mathcal{F}(m_1, m_2) B(n_1, n_2; m_1, m_2), \tag{2.2}$$

onde $B(n_1, n_2; m_1, m_2)$ é o núcleo da transformação inversa.

Existe um tipo particular de transformada de duas dimensões bem interessante, as transformadas separáveis. Nestas, o núcleo da transformação direta e inversa estão descritas, respectivamente em:

$$A(n_1, n_2; m_1, m_2) = A_C(n_1, m_1) A_R(n_2, m_2) \tag{2.3}$$

$$B(n_1, n_2; m_1, m_2) = B_C(n_1, m_1) B_R(n_2, m_2). \tag{2.4}$$

Isso permite que as linhas e colunas sejam processadas separadamente como sinais unidimensionais. Dessa forma, pode-se dividir a transformação em duas operações:

$$P(m_1, n_2) = \sum_{n_1=1}^{N_1} F(n_1, n_2) A_C(n_1, m_1) \tag{2.5}$$

$$\mathcal{F}(m_1, m_2) = \sum_{n_2=1}^{N_2} P(m_1, n_2) A_R(n_2, m_2). \tag{2.6}$$

As equações 2.5 e 2.6 são facilmente reescritas na forma matricial e unidas em:

$$\mathcal{F} = A_C F A_R^T \tag{2.7}$$

e a inversa

$$F = B_C \mathcal{F} B_R^T. \tag{2.8}$$

2.2 TRANSFORMADA DISCRETA DE FOURIER

Como primeiro exemplo de transformada, será apresentada a transformada discreta de Fourier ou DFT (do inglês, *Discret Fourier Transform*). Contudo, primeiramente, deve-se lembrar da versão contínua. A transformada de Fourier de um sinal é o produto interno do mesmo em uma função oscilatória exponencial complexa, $e^{j\omega t}$, onde $j = \sqrt{-1}$. Como o produto interno entre duas funções pode ser definido como:

$$\langle f, g \rangle = \int_{-\infty}^{\infty} f(t)\, g^*(t)\, dt, \tag{2.9}$$

A transformada de Fourier é

$$\mathcal{F}(\omega) = \int_{-\infty}^{\infty} f(t)\, e^{-j\omega t} dt, \tag{2.10}$$

e a transformada inversa é dada por

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \mathcal{F}(\omega) e^{j\omega t} d\omega, \tag{2.11}$$

Em processamento digital de sinais, utiliza-se a versão discreta dessa transformada como

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn} \tag{2.12}$$

onde N é o número de amostras do sinal, X_k é o k-ésimo coeficiente transformado e x_n é a n-ésima amostra do sinal. A transformada inversa é definida por:

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{+\frac{2\pi i}{N} kn} \tag{2.13}$$

Apesar de ser uma transformada separável, pode se processar linhas e colunas independentemente, é costumeiro definir ainda a transformada discreta de Fourier bidimensional como

$$X_{p,k} = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x_{n,m} e^{-\frac{2\pi i}{N} pn} e^{-\frac{2\pi i}{M} km} \tag{2.14}$$

$$x_{n,m} = \frac{1}{NM} \sum_{p=0}^{N-1} \sum_{k=0}^{M-1} X_{p,k} e^{+\frac{2\pi i}{N} pn} e^{+\frac{2\pi i}{M} km} \tag{2.15}$$

2.3 TRANSFORMADAS DE BLOCO

Daqui em diante todas as transformadas consideradas serão separáveis e apenas a versão unidimensional será apresentada. Para extensão para duas dimensões as colunas e linhas deverão ser processadas independentemente conforme as equações 2.7 e 2.8.

As transformadas de bloco têm imensa importância tanto no processamento de sinais como na codificação de sinais digitais. A transformada discreta de co-senos, DCT [10], que talvez seja a mais difundida, é utilizada em diversos padrões de compressão de imagem e vídeo, tais como JPEG [6], MPEG2 [11], H.263 [12], entre outros. Como será visto a seguir, o H.264/AVC [1] utiliza uma transformação baseada na DCT.

Nesse tipo de transformada, o sinal $x(n)$ é agrupado em blocos antes do processamento. O m-ésimo bloco pode ser encontrado de acordo com:

$$\mathbf{x_m} = \left[\begin{array}{cccc} x(mM) & x(mM-1) & ... & x(mM-M+1) \end{array} \right]^T. \tag{2.16}$$

Cada um desses blocos é transformado independentemente, não havendo sobreposição. O bloco transformado resultante tem a mesma dimensão do bloco original. Observando as regras das operações matriciais, deve-se notar que a matriz A, núcleo da transformação direta, deve ser quadrada para que isso aconteça. Caso definamos o tamanho do bloco como sendo M, a matriz $\mathbf{A}$ deverá uma matriz de dimensões $M \times M$. A transformada direta é dada por:

$$\mathbf{y_m} = \mathbf{A}\mathbf{x_m}, \tag{2.17}$$

A transformada inversa é

$$\mathbf{x_m} = \mathbf{B}\mathbf{y_m} = \mathbf{A}^{-1}\mathbf{y_m}, \tag{2.18}$$

Considerando o caso específico de transformações reais e unitárias ou ortonormais (no caso de transformadas de bloco, os dois termos podem ser utilizados com sinônimo), temos que $\mathbf{A}^{-1} = \mathbf{A}^T$. Uma forma de verificar a ortogonalidade de uma transformada de bloco é verificar se

$$\langle \mathbf{a}_i, \mathbf{a}_j \rangle = \begin{cases} 0, & se \quad i \neq j \\ 1, & se \quad i == j \end{cases} \tag{2.19}$$

onde $\mathbf{a}_k$ é a k-ésima linha da matriz $\mathbf{A}$.

Uma característica muito interessante nas transformações unitárias é a preservação da energia do sinal transformado. considerando que o sinal x seja estacionário, temos que:

$$M\sigma_x^2 = \sum_{i=0}^{M-1} \sigma_i^2, \tag{2.20}$$

onde σ_x^2 é a variância do sinal x e σ_i^2 a variância da subbanda $y_i(m)$.

2.3.1 Fatorização de Transformadas Discretas

As transformadas de bloco são representadas por matrizes quadradas que podem ser fatorizadas. No caso de matrizes ortogonais, isso pode ser feito em produtos de rotações planos de Givens. Sendo assim, qualquer matriz ortogonal A pode ser escrita como:

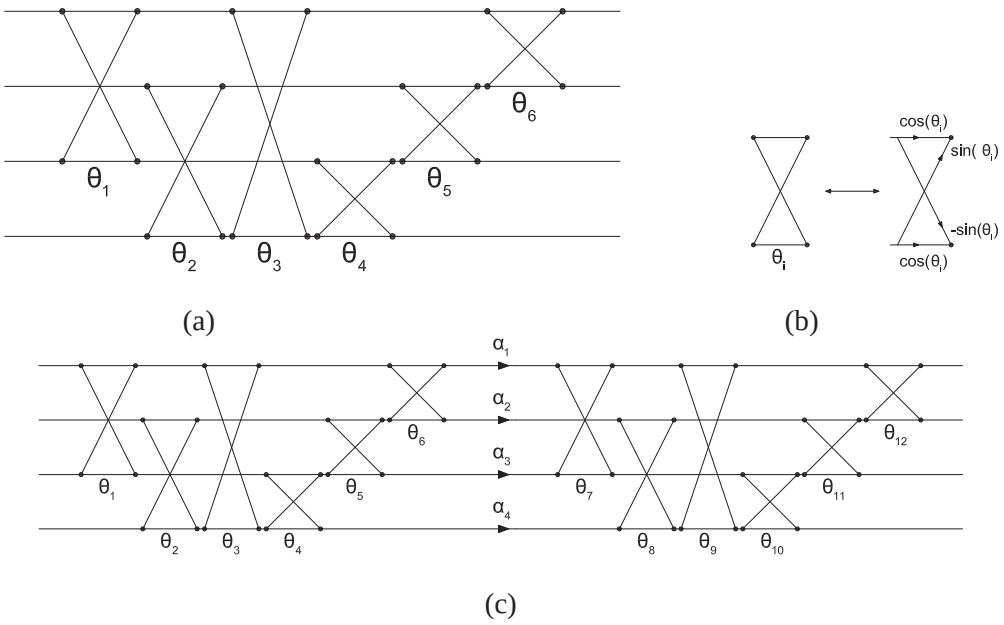


Figura 2.1: Fatoração de uma matriz 4×4. (a) Fatoração de uma matriz ortogonal de dimensão 4×4 por rotações de Givens. (b) Detalhe da rotação de um plano. (c) Fatoração completa de uma matriz de dimensão 4×4.

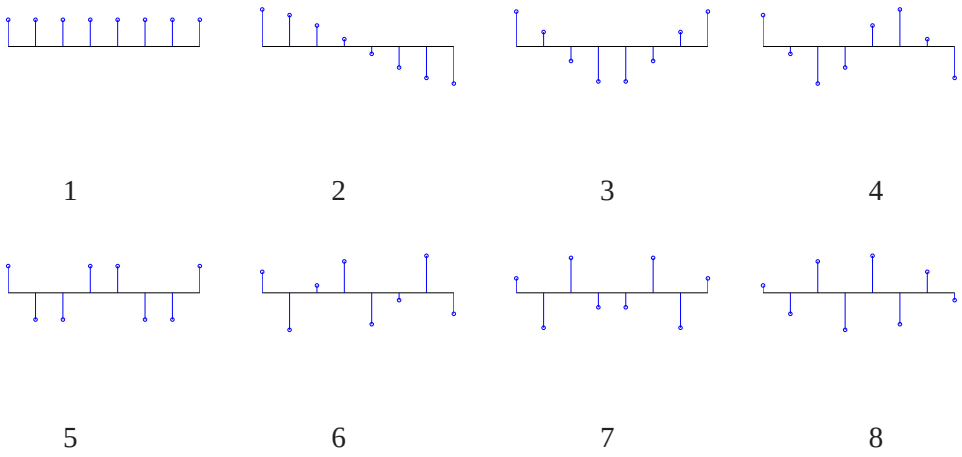


Figura 2.2: Bases da DCT unidimensional.

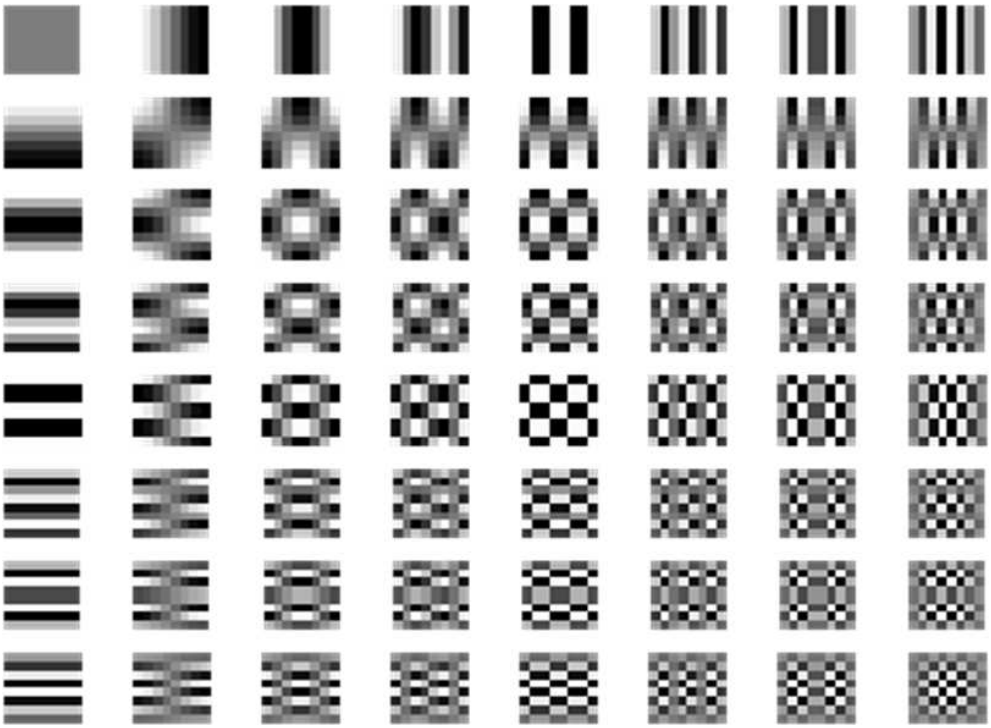


Figura 2.3: Bases da DCT bidimensional.

$$\mathbf{A} = \prod_{i=0}^{M-2} \prod_{j=i+1}^{M-1} \Theta(i, j, \theta_n), \tag{2.21}$$

onde $\Theta(i, j, \theta_n)$ representa a uma rotação de em torno do eixo normal a i-ésimo e j-ésimo eixos. Essa matriz é idêntica a identidade exceto por quatro posições que são:

$$\Theta_{ii} = cos\,(\theta) \quad \Theta_{jj} = cos\,(\theta) \quad \Theta_{ij} = sen\,(\theta) \quad \Theta_{ji} = -sen\,(\theta) \tag{2.22}$$

Um exemplo muito ilustrativo é apresentado a seguir, fatorização de uma transformada ortogonal de dimensão $M = 2$.

$$\mathbf{A} = \begin{bmatrix} cos\,(\theta) & sen\,(\theta) \\ cos\,(\theta) & -sen\,(\theta) \end{bmatrix} \tag{2.23}$$

Como veremos a seguir uma transformada discreta de cossenos de dimensão $M = 2$ é

$$\mathbf{A} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.24}$$

Isso equivale a uma rotação de $45°$. Todas as transformada de bloco ortogonais são rotações de eixos.

Contudo, isso somente se aplica para fatoração de matrizes ortogonais. Em um caso mais geral, pode-se utilizar a decomposição do valor singular, SVD (do inglês, *Singular Value Decomposition*). Neste tipo de decomposição, a matriz é separada em três matrizes $\mathbf{U}$ e $\mathbf{V}$, matrizes ortogonais, e $\mathbf{\Lambda}$, matriz diagonal contendo valores singulares da matriz $\mathbf{A}$.

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{V} \tag{2.25}$$

Variando os ângulos de $\mathbf{U}$ e $\mathbf{V}$ em uma Rotação de Givens e os valores singulares contidos na matriz $\mathbf{\Lambda}$, temos uma fatorização completa de A como em:

$$\mathbf{A} = \left(\prod_{i=0}^{M-2} \prod_{j=i+1}^{M-2} \Theta(i,j,\theta_n) \right) \mathbf{\Lambda} \left(\prod_{i=0}^{M-2} \prod_{j=i+1}^{M-2} \Theta(i,j,\theta_n) \right) \tag{2.26}$$

Na Figura 2.1 é possível ver um exemplo de fatoração completa de uma matriz de 4×4

2.3.2 Transformada Discreta de Cossenos

A transformada discreta de cossenos, a partir daqui referida somente pela sigla DCT (do inglês, *Discret cosine transform*) é uma transformada unitária e separável, definada por:

$$y(k) = w(k) \sum_{i=1}^{N} x(n) \cos\left\{ \frac{\pi (2n-1)(k-1))}{2N} \right\}, \tag{2.27}$$

onde

$$w(k) = \begin{cases} \frac{1}{\sqrt{N}}, & se \quad k == 1 \\ \sqrt{\frac{2}{N}}, & se \quad k == 2, ..., N \end{cases} \tag{2.28}$$

Assim, o núcleo da transformada pode ser definida como:

$$a_{ij} = w(j) \cos\left\{ \frac{\pi (2j-1)(i-1))}{2N} \right\} \tag{2.29}$$

Com intuito ilustrativo, cada uma das funções de base da DCT é apresentada na Figura 2.2. Na Figura 2.3 são ilustrados as bases para a DCT bidimensional.

2.3.3 Transformada Hadamard

A transformada Hadamard também é uma transformada unitária como a DCT. Contudo, contrariamente a esta ultima, somente está definida para dimensões que sejam potência de 2, ou seja, $M = 2^m$. Ela pode ser definida recursivamente por:

$$H_M = \frac{1}{\sqrt{2}} \begin{bmatrix} H_{\frac{M}{2}} & H_{\frac{M}{2}} \\ H_{\frac{M}{2}} & -H_{\frac{M}{2}} \end{bmatrix} \tag{2.30}$$

Para completar a definição, temos que $H_1 = 1$. Assim, para $M = 2$ temos:

$$H_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.31}$$

Neste caso, a transformada coincide com a DCT de dimensão 2. Para $M = 4$:

$$H_4 = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \tag{2.32}$$

2.4 TRANSFORMADAS SOBREPOSTAS

Como foi visto anteriormente, as transformadas de bloco apresentadas não possuem nenhum tipo de sobreposição. Desta forma, cada bloco é independente. Assim, quando os coeficientes são quantizados para a codificação, seja de um sinal ou uma imagem, podem aparecer descontinuidades entre um bloco e outro. Este efeito é conhecido como efeito de blocos. Uma das principais motivações para o uso de transformadas sobrepostas é amenizar esse efeito já que os coeficientes da transformada não dependem somente do bloco sendo codificado, mas também dos vizinhos.

De forma geral, uma transformada sobreposta tem suas bases, ou filtros, de tamanho L maior que M, estendendo-se além das fronteiras do bloco. De forma mais específica podemos concentrar nossas atenções nos casos em que

$$L = NM, \tag{2.33}$$

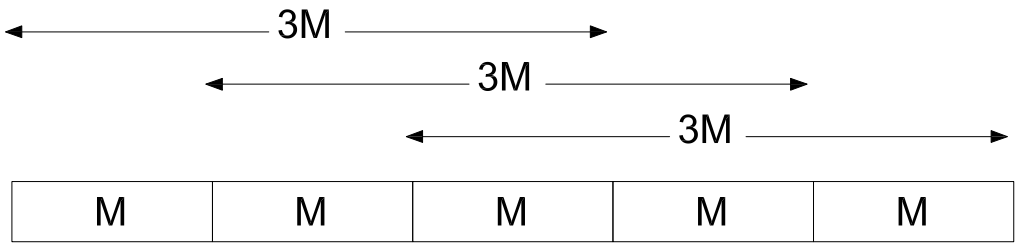


Figura 2.4: Sinal dividido em blocos e a vizinhança utilizada na transformação. No exemplo em questão, o fator de sobreposição $N = 3$

onde N, fator de sobreposição, é um numero inteiro. Inicialmente, vamos considerar que o sinal é infinito, ou pelo menos, que a porção sendo processada é suficientemente longe do inicio e fim do sinal, garantindo que haja sempre blocos vizinhos.

A matriz de transformação $\mathbf{P}$ pode ser separada em N matrizes quadradas:

$$\mathbf{P} = [\mathbf{P}_0 \mathbf{P}_1 \cdots \mathbf{P}_{N-1}], \tag{2.34}$$

2.4.1 Transformadas sobrepostas ortogonais

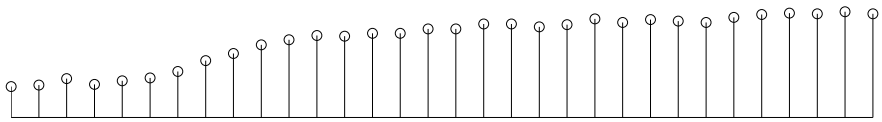
A ortogonalidade, como vista para transformadas de bloco, não é mais aplicável já que a matriz $\mathbf{P}$ não é nem sequer quadrada. O critério para determinar a ortogonalidade é substituído pelo critério de reconstrução perfeita:

$$\sum_{i=0}^{N-1-l} \mathbf{P}_i \mathbf{P}_{i+l}^T = \sum_{i=0}^{N-1-l} \mathbf{P}_{i+l}^T \mathbf{P}_i = \delta(l)\mathbf{I}_M, \tag{2.35}$$

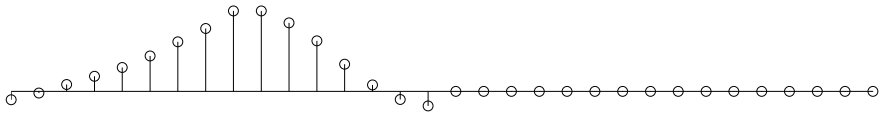
onde $\delta(l)$ é a função delta de Kronecker definida por

$$\delta(l) = \begin{cases} 1, & se \quad l = 0 \\ 0, & se \quad l \neq 0. \end{cases} \tag{2.36}$$

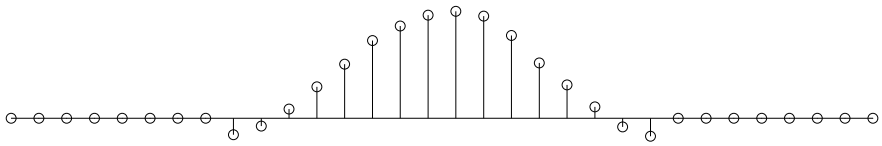
Uma forma de realizar a transformação é dividir as amostras do sinal em blocos como feito nas transformadas de bloco. Contudo, diferentemente destas, esses blocos devem ter L amostras e não somente M. O próximo bloco, deslocado em M, contém amostras pertencentes ao bloco anterior. O m-ésimo bloco, denominado $\mathbf{v}_m$, é definido na equação 2.37. Na prática é uma versão do bloco $\mathbf{x}_m$ estendido em $(L - M)$ em cada lado, ilustrado na Figura 2.4.



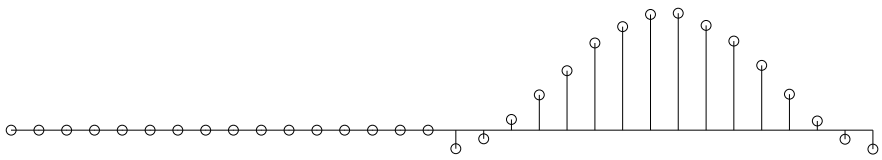
(a)



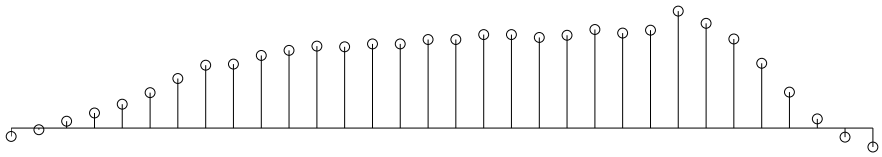
(b)



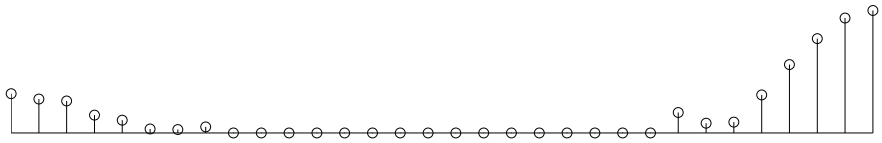
(c)



(d)



(e)



(f)

Figura 2.5: (a) Sinal original; (b) Sinal reconstruído a partir do primeiro bloco; (c) Sinal reconstruído a partir do segundo bloco; (d) Sinal reconstruído a partir do terceiro bloco; (e) Sinal reconstruído a partir dos três blocos (f) Erro absoluto na reconstrução.

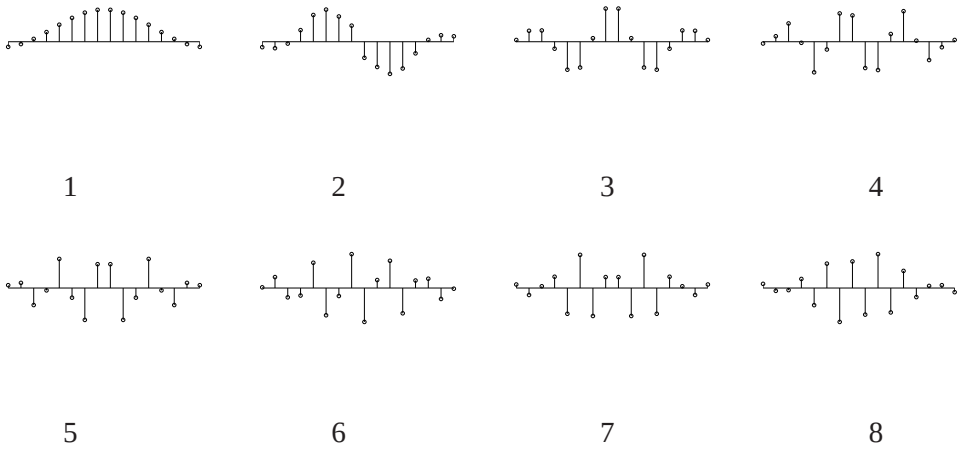


Figura 2.6: Bases da LOT unidimensional.

$$\mathbf{v}_m = \left[\begin{array}{ccc} x\left(mM - (N-1)\,\frac{M}{2}\right) & \cdots & x\left(mM + (N-1)\,\frac{M}{2} - 1\right) \end{array} \right] \tag{2.37}$$

A transformação de $\mathbf{v}_m$ é realizada conforme:

$$\mathbf{y}_m = \mathbf{P}\mathbf{v}_m \tag{2.38}$$

e a transformada inversa, no caso real, por:

$$\widehat{\mathbf{v}}_m = \mathbf{P}^T\mathbf{y}_m. \tag{2.39}$$

O resultado da transformação inversa é denotado por $\widehat{\mathbf{v}}_m$, pois é diferente do vetor $\mathbf{v}_m$. Para recuperar o sinal original é necessário acumular as contribuições de todos os blocos dentro de uma vizinhança vizinhança como é ilustrado na figura 2.5. Até agora, foram considerados apenas sinais infinitos ou suficientemente longe das fronteiras do sinal. Sem tratamento especial não é possível reconstruir os blocos da fronteira, motivo pelo qual as amostras no início e final do sinal reconstruído no exemplo são diferentes do sinal original.

As bases utilizadas no exemplo, são mostradas na Figura 2.6.

2.4.2 Transformadas sobrepostas não ortogonais

Como no caso ortogonal, o sinal é transformado a partir da projeção nas bases contidas na matriz $\mathbf{P}$. Contudo, para a reconstrução, diferentemente do caso ortogonal, é necessário a definição de uma nova matriz, de mesmas dimensões $M \times L$:

$$\mathbf{Q} = [\mathbf{Q}_0 \mathbf{Q}_1 \cdots \mathbf{Q}_{N-1}]. \tag{2.40}$$

A transformação inversa é dada por:

$$\widehat{\mathbf{v}}_m = \mathbf{Q}^T \mathbf{y}_m \tag{2.41}$$

E para garantir que haja reconstrução, é necessário que:

$$\sum_{i=0}^{N-1-l} \mathbf{Q}_i^T \mathbf{P}_{i+l} = \sum_{i=0}^{N-1-l} \mathbf{Q}_{i+l}^T \mathbf{P}_i = \delta(l)\mathbf{I}_M, \tag{2.42}$$

Diferentemente do caso ortogonal, há necessidade que ambas condições sejam atendidas.

2.4.3 Transformada ortogonal sobreposta

Uma forma de fatorização de transformadas sobrepostas ortogonais proposta por Malvar [5] é

$$\mathbf{P}_{LOT} = \begin{bmatrix} \mathbf{I}_M & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_R \end{bmatrix} \begin{bmatrix} \mathbf{D}_e - \mathbf{D}_o & \mathbf{J}_{M/2}\left(\mathbf{D}_e - \mathbf{D}_o\right) \\ \mathbf{D}_e - \mathbf{D}_o & -\mathbf{J}_{M/2}\left(\mathbf{D}_e - \mathbf{D}_o\right) \end{bmatrix}, \tag{2.43}$$

onde I_M é a matriz identidade de dimensão $M \times M$, $\mathbf{D}_e$ e $\mathbf{D}_o$ são matrizes de dimensão $M/2 \times M$ contendo as linhas pares e ímpares do núcleo da DCT, $\mathbf{V}_r$ é uma matriz ortogonal de dimensão $M/2 \times M/2$ e $\mathbf{J}_{M/2}$ é uma matriz quadrada esparsa contendo apenas uns na diagonal secundária.

$$\mathbf{J}_{M/2} = \begin{bmatrix} 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & \dots & 1 & 0 \\ \vdots & \dots & \ddots & & \vdots \\ 0 & 1 & \dots & 0 & 0 \\ 1 & 0 & \dots & 0 & 0 \end{bmatrix}, \tag{2.44}$$

e $\mathbf{V}_r$ é fatorizada de acordo com:

$$\mathbf{V}_r = \prod_{i=0}^{\frac{M}{2}-2} \Theta(i, i+1, \theta_i). \tag{2.45}$$

Essa fatorização não é completa como a apresentada na Equação 2.21. Contudo, essa transformada produz resultados muito interessantes. Essa foi a primeira transformada sobreposta útil e ficou conhecida como transformada ortogonal sobreposta ou LOT (*Lapped orthogonal transform*)

A LOT somente produz transformadas de $M \times 2M$. Uma forma mais geral de definição de transformada ortogonal sobreposta é transformada ortogonal sobreposta generalizada ou GenLot (*Generalized Lapped Orthogonal Transform*). Esta ultima produz bancos de filtros de $M \times NM$.

2.4.4 Transformada Biortogonal Sobreposta

Embora a ortognalidade seja uma característica muito interessante, ela também é restritiva. Em algumas situações uma transformada biortogonal pode ter um desempenho superior a uma transformada ortogonal semelhante. No caso da LOT, há ainda o problema de que a primeira base não decai a zero, por isso, apesar de diminuído, o efeito de blocos continua existindo. Para resolver esse problema, de forma semelhante a LOT, define-se a transformada biortogonal sobreposta ou LBT (*Lapped Biorthogonal Transform*)[5], cuja fatorização é apresentada na seguinte equação:

$$\mathbf{P}_{LBT} = \begin{bmatrix} \mathbf{I}_M & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_R \end{bmatrix} \begin{bmatrix} \mathbf{D}_e - \boldsymbol{\Upsilon}\mathbf{D}_o & \mathbf{J}_{M/2}\left(\mathbf{D}_e - \boldsymbol{\Upsilon}\mathbf{D}_o\right) \\ \mathbf{D}_e - \boldsymbol{\Upsilon}\mathbf{D}_o & -\mathbf{J}_{M/2}\left(\mathbf{D}_e - \boldsymbol{\Upsilon}\mathbf{D}_o\right) \end{bmatrix}, \tag{2.46}$$

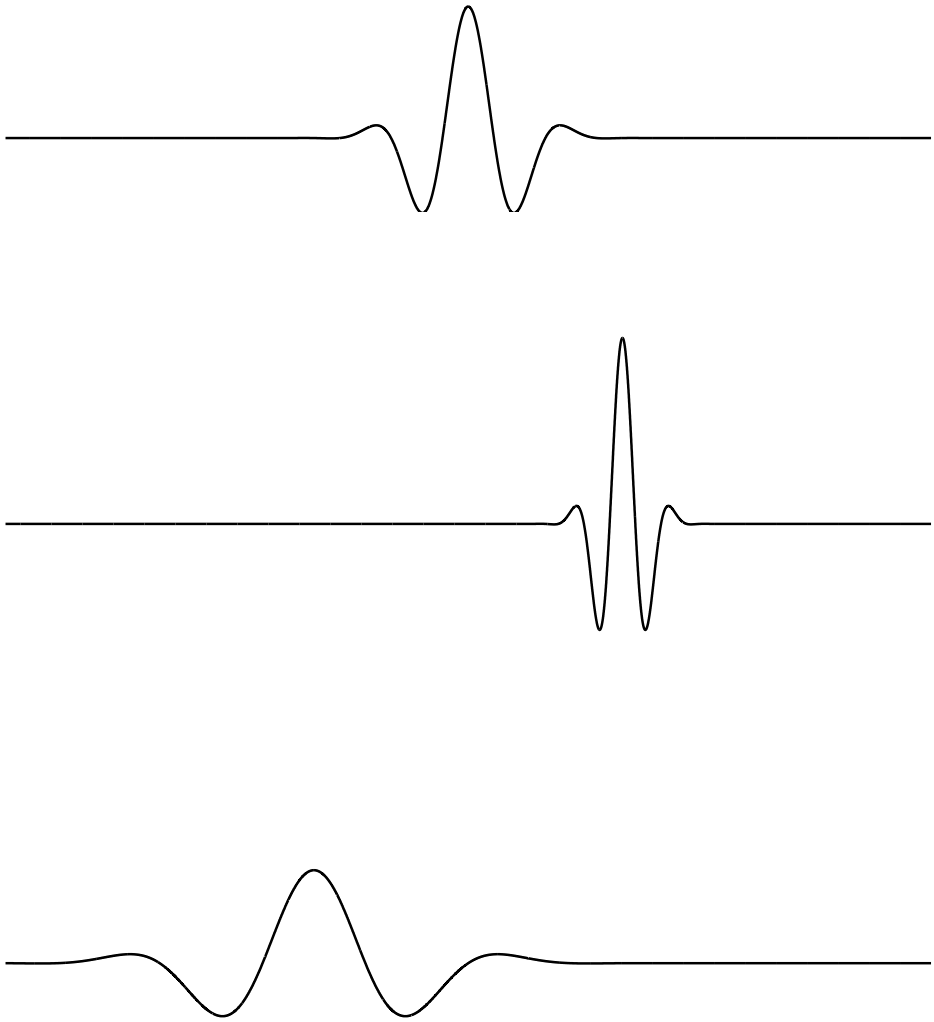
onde $\boldsymbol{\Upsilon}$ é uma matriz diagonal dada por $\boldsymbol{\Upsilon} = diag\left\{\sqrt{2}, 1, 1 \ldots, 1\right\}$.

É interessante notar que apenas a saída da primeira subbanda da DCT é multiplicada $\sqrt{2}$. Isso torna a transformada transformada biortogonal mas facilmente inversível, $\mathbf{Q}_{LBT}$ seria fatorada de forma semelhante a $\mathbf{P}_{LBT}$ com a diferença que $\boldsymbol{\Upsilon}$ seria trocado por $\boldsymbol{\Upsilon}_{INV} = diag\left\{1/\sqrt{2}, 1, 1 \ldots, 1\right\}$. O resultado da inserção desse fator é a diminuição nas coeficientes laterais na primeira base, causando uma redução adicional do efeito de blocos.

2.5 *WAVELETS*

Uma das desvantagens da DFT, vista anteriormente, é que, embora ela tenha apenas resolução em freqüência, não possui nenhuma resolução no tempo. Assim, embora se conheça o conteúdo de freqüência do sinal, é impossível determinar o intervalo de tempo em que determinado componente de freqüência ocorreu. Esse tipo de conhecimento pode ser essencial para alguns tipos de análise de sinal. As transformadas *wavelets* podem ser uma alternativa nestes casos.

Antes de uma introdução sobre transformadas *wavelets* discretas, de maior interesse neste trabalho, será dada uma breve olhada na versão contínua. Para a definição de uma *wavelet*, escolhe-se uma função $\psi(t)$ denominada wavelet-mãe. Desta função, gera-se uma família de funções utilizando escalonamentos e



(c)

Figura 2.7: Funções *wavelet*. (a) Função wavelet-mãe. (b) Função *wavelet* dilatada ($a > 1$) e deslocada à direita ($b > 0$). (c) Função *wavelet* dilatada ($a < 1$) e deslocada à esquerda ($b < 0$)

deslocamentos como em:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{|a|}}\psi\left(\frac{t-b}{a}\right), \tag{2.47}$$

onde a é o fator de escalonamento, deve ser diferente de zero, e b é deslocamento. Para valores de a maiores que a unidade, obteremos uma versão dilatada da mesma forma que valores de a menores que a unidade fornecerão uma versão comprimida. A fator b controla o deslocamento da função *wavelet*. Valores positivos de b deslocam a função para esquerda e valores negativos para a direita. Na Figura 2.7, são ilustradas algumas versões de deslocamento e escalonamento.

Algumas critérios devem ser atendidos para que uma função possa ser usada como *wavelet*. Primeira-

mente, o seu valor médio deve ser nulo, ou seja:

$$\int_{-\infty}^{\infty} \psi(t)\, dt \equiv 0 \tag{2.48}$$

Além disso, deve existir o par de transformada de Fourier: $\psi(t) \leftrightarrow \Psi(w)$ e

$$\int_{-\infty}^{\infty} \frac{|\Psi(w)|}{|w|} dw < \infty \tag{2.49}$$

Garantidas essas condições, a transformada *wavelet* é definida como o produto interno entre a função sendo analisada e $\psi_{a,b}(t)$

$$\mathcal{W}(a,b) = \int_{-\infty}^{\infty} f(t)\, \psi_{a,b}(t)\, dt. \tag{2.50}$$

transformada inversa é dada por:

$$f(t) = \frac{1}{C} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{1}{|a|^2} \mathcal{W}(a,b) \left\{ \frac{1}{\sqrt{|a|}} \bar{\psi}\left(\frac{t-b}{a}\right) \right\} da\, db \tag{2.51}$$

onde

$$C = \int_{-\infty}^{\infty} \frac{|\Psi(w)|}{|w|} dw \tag{2.52}$$

Caso a wavelet seja ortogonal e real, $\bar{\psi}$ e ψ são iguais.

Basicamente, essa transformada mapeia um sinal unidimensional em uma representação bidimensional ($tempo \times escala$) extremamente redundante. Para evitar isso pode-se discretizar os parâmetros a e b. A forma mais popular de discretização é

$$a = a_0^m \tag{2.53}$$

$$b = n b_0 a_0^m, \tag{2.54}$$

onde n e m são números inteiros. Dessa forma a Equação 2.47 pode ser reescrita como:

$$\psi_{m,n}(t) = a_0^{-m/2} \psi\left(\frac{t - n b_0 a_0^m}{a_0^m}\right) \tag{2.55}$$

Uma escolha interessante pode ser: $a_0 = 2$ e $b_0 = 1$. A discretização com essa escolha de parâmetros é conhecida como diádica. Na Equação 2.56.

$$\psi_{m,n}(t) = 2^{-m/2}\psi\left(\frac{t - n2^m}{2^m}\right) \tag{2.56}$$

O resultado da transformada da Equação 2.50, reescrita na Equação 2.57, deixa de ser uma função bidimensional contínua. A função f agora é mapeada em um conjunto de pontos em um reticulado bidimensional. Essa nova transformada é conhecida como série de *wavelet*:

$$c_{m,n}(f) = \int_{-\infty}^{\infty} f(t)\,\psi_{m,n}(t)\,dt \tag{2.57}$$

A transformada inversa é:

$$f(t) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} c_{m,n}(f)\,\psi_{m,n}(t) \tag{2.58}$$

Contudo, a função f ainda é contínua. Para aplicações de processamento digital de sinais, é interessante que a variável independente dessa função, t, também seja discretizada. Essa nova transformada é conhecida como transformada discreta de *wavelet* ou DWT (do inglês, *Discrete wavelet transform*). A forma diádica da DWT é apresentada na Equação 2.59.

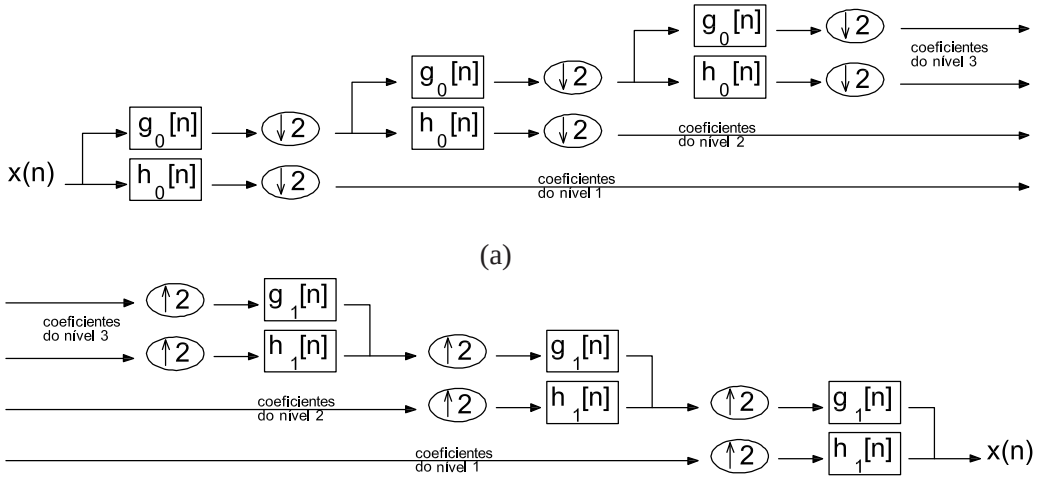
$$DFT_{m,n}(f) = 2^{-m/2}\sum_{-\infty}^{\infty} f(k)\,\psi\left(\frac{k - n2^m}{2^m}\right) \tag{2.59}$$

O mais interessante da DWT é que ela pode ser implementada por meio de um algoritmo de filtragens de forma piramidal como na Figura 2.8. Verificando a Equação 2.58, é possível notar que seriam necessários infinitos níveis de decomposição *wavelet*. Contudo, em situações práticas, são utilizados apenas uma quantidade finita de níveis. O erro ocasionado com esse truncamento pode ser compensado com a adoção de uma nova função $\phi(t)$, conhecida como função de escala

$$f(t) = \sum X_{S,n}2^{-\frac{S}{2}}\psi\left(2^{-S}t - n\right) + \sum_{m=-\infty}^{S-1}\sum_{n=-\infty}^{\infty} c_{m,n}(f)\,\psi_{m,n}(t)\,. \tag{2.60}$$

Dessa forma os filtros de análise e síntese seriam

$$h_0(n) = \int_{-\infty}^{\infty} \sqrt{2}\phi(t)\,\overline{\phi}(2t + n)\,dt, \tag{2.61}$$



(a)

(b)

Figura 2.8: Implemenstação da transformada discreta de *wavelet* por meio de filtragens. (a) banco de filtros de análise (b) banco de filtro de síntese.

$$g_0(n) = \int_{-\infty}^{\infty} \overline{\phi}(t)\sqrt{2}\phi(2t-n)\,dt, \tag{2.62}$$

$$h_1(n) = \int_{-\infty}^{\infty} \sqrt{2}\psi(t)\overline{\phi}(2t+n)\,dt, \tag{2.63}$$

$$g_1(n) = \int_{-\infty}^{\infty} \overline{\psi}(t)\sqrt{2}\phi(2t-n)\,dt. \tag{2.64}$$

3 CODIFICADORES PADRÃO

3.1 PADRÃO JPEG2000

O JPEG2000[2][13][14] é considerado por muitos o padrão estado da arte em compressão de imagens. Embora, como será apresentado na próxima seção, o padrão H.264/AVC[1][15][16] de compressão de vídeos possa ter desempenho superior na compressão de imagens estáticas, ele não foi originalmente proposto para compressão de imagens estáticas e não possui diversos recursos atrativos presentes no JPEG2000[17]. Este padrão é resultado do esforço conjunto do ISO/IEC (*International Standards Organization/ International Electrotechnical Commission*) e ITU (do inglês *International Telecommunication Union*) ou União Internacional de Telecomunicações. O próprio nome JPEG (*Joint Photographic Experts Team Group*) é uma referencia à união dessas organizações.

Em março de 1997, uma chamada de contribuições técnicas foi feita contendo várias características desejáveis, algumas das quais são apresentadas a seguir. As contribuições foram avaliadas em uma reunião em Sidney em novembro do mesmo ano.

- Alto desempenho em baixas taxa. Obviamente desejava-se um desempenho superior aos codificadores de imagem predecessores em todas as faixas de taxa. Contudo, os ganhos em baixas taxas deveriam ser significativamente maiores.
- Compressão de imagens com amostras de 1 a 16 *bits*.
- Transmissão progressiva, tanto em qualidade como em resolução.
- Compressão com e sem perdas.
- Acesso espacial aleatório sem a necessidade de descompressão de toda imagem.
- Robustez a erros.
- Processamento seqüencial. Não havendo necessidade de armazenamento da imagem inteira em *buffer*.

O resultado foi um padrão que não somente atendia todos os requisitos iniciais como apresentava características inovadoras, algumas das quais são apresentadas a seguir.

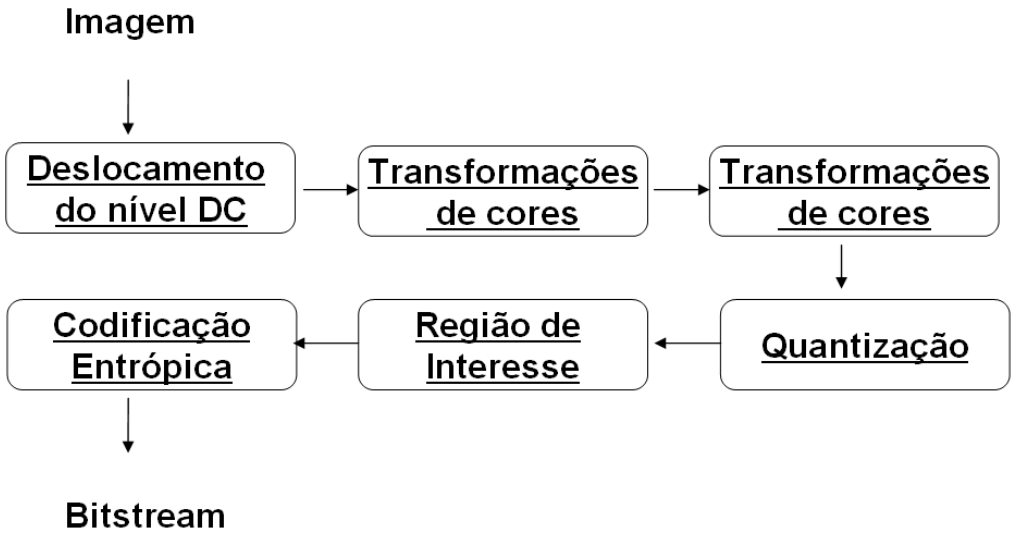


Figura 3.1: Diagrama simplificado do JPEG2000.

- Tamanho do arquivo pode ser definido antes do início da codificação.
- Região de interesse.
- Inclusão de metadados, essa característica foi inserida na parte 2 do padrão.
- Suporte a imagens grandes. O padrão prevê suporte a imagens de até $(2^{32}-1) \times (2^{32}-1)$ *pixels*.

A Figura 3.1 apresenta um diagrama simplificado do JPEG2000. Cada um dos módulos será apresentado a seguir.

3.1.1 Deslocamento do nível DC

Costumeiramente, as imagens são armazenadas utilizando números inteiros não negativos. Contudo, por questões de precisão numérica na transformação, é interessante que as amostras estejam em uma faixa dinâmica centrada em zero. Esse é o objetivo do Deslocamento do nível DC, que pode, opcionalmente, ser realizado em todos os elementos da imagem de acordo com:

$$I_i^{'}(x,y) = I_i(x,y) - 2^{N-1}, \tag{3.1}$$

onde N é o numero de bits por amostra.

Por exemplo, para imagens de 256 níveis, 8 bits, a Equação 3.1 se reduz a:

$$I_i^{'}(x,y) = I_i(x,y) - 128 \tag{3.2}$$

3.1.2 Transformação de Cores

Embora o padrão codifique cada componente de uma imagem multiespectral independentemente, a parte 1 do padrão admite dois tipos de transformações de cor, uma reversível e outra irreversível. A aplicação dessas transformações pode reduzir redundância entre os componentes espectrais aumentando, assim, a eficiência de codificação.

As transformações, opcionalmente, podem ser aplicadas nos três primeiros componentes da imagem, que serão considerados como vermelho (R), verde (G) e azul (B). Estes deverão ter a mesma dimensão e mesmo número de bits de resolução. A primeira transformação, apresentada nas Equações 3.3 a 3.5, na forma direta, e nas Equações 3.6 a 3.8, para a transformação inversa, é reversível e utilizada quando se deseja codificação de imagens sem perda. A outra transformação, irreversível, é utilizada na codificação de imagens com perda, apresentada nas equações 3.9, no caso da transformação direta, e 3.10, no caso da transformação inversa.

$$x_{Y'}[n] = \left\lfloor \frac{x_R[n] + 2x_G[n] + x_B[n]}{4} \right\rfloor \tag{3.3}$$

$$x_{Db}[n] = x_B[n] - x_G[n] \tag{3.4}$$

$$x_{Dr}[n] = x_R[n] - x_G[n] \tag{3.5}$$

$$x_G[n] = x_{Y'}[n] - \left\lfloor \frac{x_Db[n] + x_Dr[n]}{4} \right\rfloor \tag{3.6}$$

$$x_B[n] = x_{Db}[n] + x_G[n] \tag{3.7}$$

$$x_R[n] = x_{Dr}[n] + x_G[n] \tag{3.8}$$

$$\begin{pmatrix} x_y[n] \\ x_{Cb}[n] \\ x_{Cr}[n] \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.168736 & -0.331264 & 0.5 \\ 0.5 & -0.418688 & -0.081312 \end{pmatrix} \begin{pmatrix} x_R[n] \\ x_G[n] \\ x_B[n] \end{pmatrix} \tag{3.9}$$

$$\begin{pmatrix} x_R[n] \\ x_G[n] \\ x_B[n] \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1.402 \\ 1 & -0.344136 & -0.714136 \\ 1 & 1.772 & 0 \end{pmatrix} \begin{pmatrix} x_y[n] \\ x_{Cb}[n] \\ x_{Cr}[n] \end{pmatrix} \tag{3.10}$$

O operador $\lfloor \rfloor$ arredonda o seu argumento para o inteiro mais próximo na direção de zero.

3.1.3 Transformada *Wavelet*

O JPEG2000 utiliza transformada *(*wavelet) discreta como apresentada na Seção 2.5. Contudo, há duas opções disponíveis. É possível utilizar um banco conhecido como Daubechies 9/7, que deve ser utilizado quando se pretende realizar compressões com perda. Os números 9 e 7 designam o tamanhos dos filtros passa-altas e passa-baixas da análise, apresentados a seguir.

$$\begin{aligned} h_0(z) = 0.602949018236 &+ 0.266864118443\left(z^1 + z^{-1}\right) \\ &- 0.078223266529\left(z^2 + z^{-2}\right) \\ &- 0.016864118443\left(z^3 + z^{-3}\right) \\ &+ 0.026748757411\left(z^4 + z^{-4}\right) \end{aligned} \tag{3.11}$$

$$\begin{aligned} h_1(z) = 1.11508705 &- 0.591271763114\left(z^1 + z^{-1}\right) \\ &- 0.057543526229\left(z^2 + z^{-2}\right) \\ &+ 0.091271763114\left(z^3 + z^{-3}\right) \end{aligned} \tag{3.12}$$

Como alternativa a essa transformada, para compressão sem perdas, pode-se utilizar uma transformação inteira baseada no banco *spline* 5/3 que será omitido aqui já que não foi utilizado nas comparações feitas neste trabalho.

3.1.4 Quantização

Após o cálculo da transformada wavelet é aplicado uma quantização escalar de faixa morta (*deadzone*). Nesse tipo de quantização, a faixa central tem o dobro da largura das outras faixas, como é ilustrado na Figura 3.2.

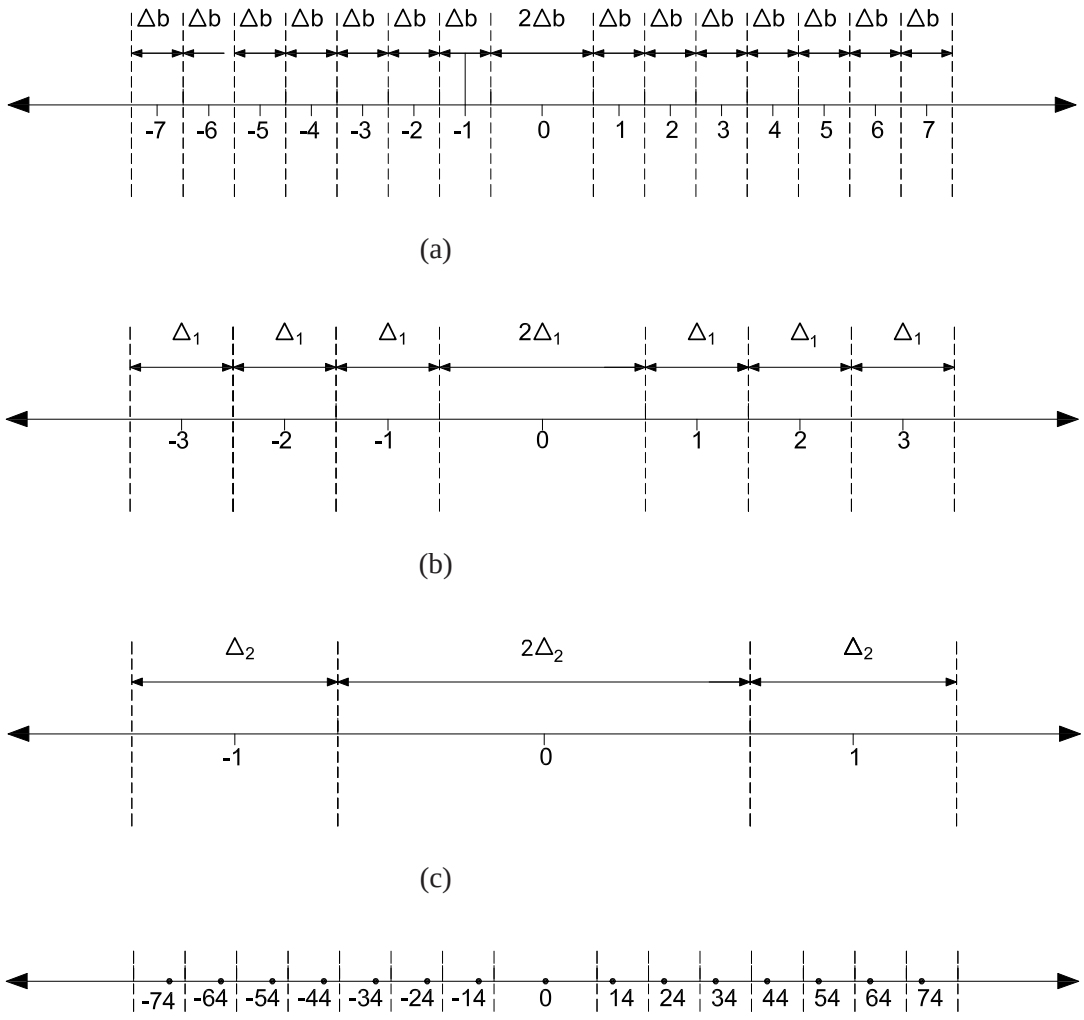


Figura 3.2: (a) Quantizador de faixa morta. (b) Quantizador equivalente com um bit faltando ser decodificado. (c) Quantizador equivalente com dois bits faltando serem decodificados. (d) Exemplo de reconstrução para $\Delta = 10$ e $\delta = 0,4$.

A Equação 3.13 define a operação.

$$q_b[j] = sign(y_b[j]) \left\lfloor \frac{|y_b[j]|}{\Delta_b} \right\rfloor, \tag{3.13}$$

onde $y_b[j]$ é um coeficiente da subbanda b e Δ_b é o passo de quantização que pode ser escolhido diferentemente para cada subbanda. A operação $\lfloor x \rfloor$ arredonda o valor de x retornando o inteiro mais próximo de zero.

A magnitude e sinal são separados de acordo com as equações 3.14 e 3.15, pois são codificados separadamente como será visto a seguir.

$$s_b[j] = sign(y_b[j]), \tag{3.14}$$

$$m_b[j] = \left\lfloor \frac{|y_b[j]|}{\Delta_b} \right\rfloor \tag{3.15}$$

O processo de dequantização é definido como:

$$\widehat{y}_b[j] = \begin{cases} 0, & se \quad m_b[j] = 0 \\ s_b[j](m_b[j] + \delta)\Delta_b, & se \quad m_b[j] \neq 0 \end{cases} \tag{3.16}$$

onde δ define o ponto em que ocorrerá a reconstrução dentro de uma faixa definida pelo passo de quantização, Δ_b. Caso o sinal sendo quantizado tenha distribuição uniforme, o erro de quantização seria minimizado quando a reconstrução fosse feita no centro da faixa, $\delta = 0,5$. Como veremos a seguir, coeficientes de transformadas, em geral, têm distribuição centrada em zero, sendo melhor a escolha de um ponto de reconstrução menor.

O JPEG2000, como foi mencionado anteriormente, tem como característica a transmissão progressiva. Isso pode ser feito com aumento de resolução ou qualidade. No último caso, nem todos os bits provenientes da codificação são enviados em uma única vez. Além disso, é possível que, para se atingir um tamanho desejado de arquivo, apenas parte dos bits sejam armazenados, ou seja, haja um truncamento no *bitstream*. Esse fenômeno ocorre pelo fato da codificação ser baseada em planos de bits. Nesse tipo de codificação os bits mais significativos, pertencentes aos planos de bits mais altos, são codificados primeiro, seguidos do sinal. Somente após a codificação completa de um plano de bits, novos bits de um mesmo coeficiente serão codificados. Essa segunda passagem é conhecida como passagem de refinamento. Esse processo é ilustrado na figura na Figura 3.3.

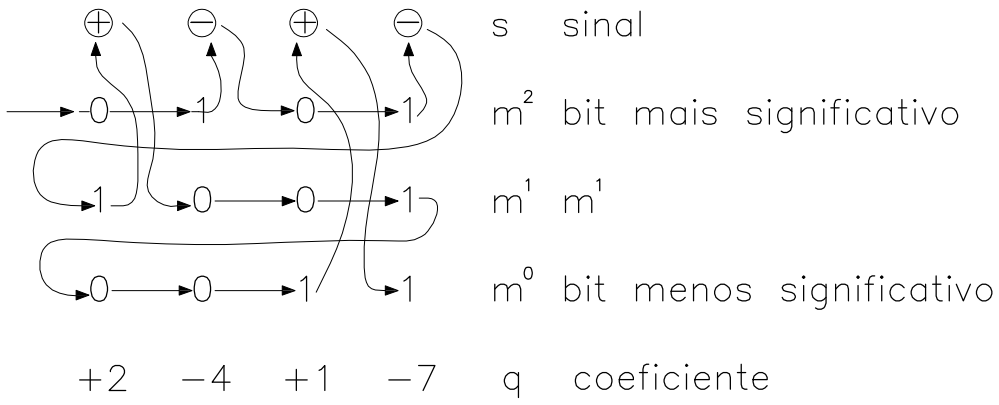


Figura 3.3: Codificação por plano de bits. As setas indicam a ordem na qual cada elemento dos coeficientes (*bits* e sinal) é codificado.

Ou seja, é possível que o decodificador não possua a mesma versão de coeficiente quantizado que o codificador, não possua todos os planos de bits. Dessa forma, eventualmente a reconstrução será dada por:

$$\widehat{y}_i\left[j\right] = \begin{cases} 0, & se \quad \widehat{m}_i\left[j\right] = 0 \\ s_i\left[j\right]\left(\left\lfloor \frac{\widehat{m}_i[j]}{2^{p_i[\mathbf{j}]}} \right\rfloor + \delta\right)\Delta_i, & se \quad \widehat{m}_i\left[j\right] \neq 0 \end{cases}, \tag{3.17}$$

onde $\widehat{m}_i$ e s_i são as versões disponíveis ao decodificador de m_b e s_b, $p_i[\mathbf{j}]$ é o numero de bits menos significativos não decodificados e Δ_i é o passo de quantização efetivo dado o número de bits decodificados, definido pela Equação 3.18

$$\Delta_i = 2^{p_i[\mathbf{j}]}\Delta_b. \tag{3.18}$$

3.1.5 Codificação Entrópica

A codificação entrópica do JPEG2000, que fica a cargo de um codificador aritmético binário conhecido como *MQ-coder*, é realizada em três passagens: codificação de significância, refinamento de magnitude e *cleanup* . A inclusão de cada amostra em um desses passos dependerá do seu estado de significância e de seus vizinhos. Em todos os casos a codificação é feita segundo a seqüência e contexto mostrados na Figura 3.4.

O primeiro passo para qualquer plano de bits é a passagem de significância. Nela estão incluídos todos os coeficientes que ainda permanecem insignificantes (o bit mais significativo ainda não foi codificado) e possui pelo menos um vizinho significante. Essa condição é feita para que apenas os coeficientes mais

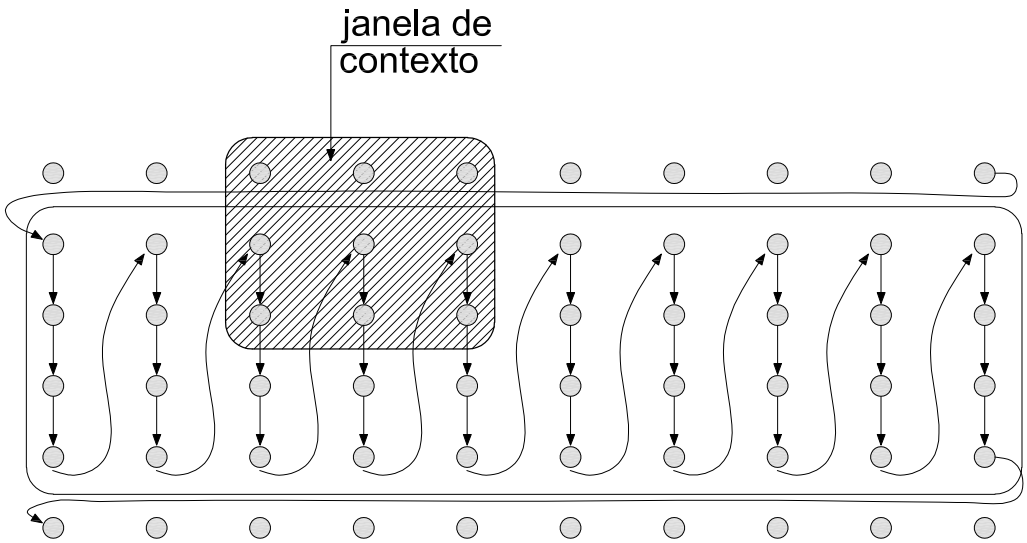


Figura 3.4: Seqüência de codificação utilizada nos blocos de codificação do JPEG2000. Os coeficientes são separados de quatro em quatro linhas e a são codificados de cima para baixo e da esquerda para direita. A janela de contexto disponível para a codificação pode conter até oito vizinhos

prováveis de se tornarem significantes sejam considerados. Após a codificação da significância, o sinal é codificado. Para isso, utiliza-se como contexto o sinal de coeficientes vizinhos que eventualmente já tenham sido codificados.

Em seguida, é realizada a passagem de refinamento de magnitude, na qual são incluídas todas as amostras que passaram a ser significantes em planos de bits anteriores. Além do contexto disponível a qualquer uma das passagens, há a informação que coeficientes de subbandas tendem a ter distribuição de probabilidade simétrica e concentrada em zero como ilustrado na Figura 3.5. Assim temos que a probabilidade do próximo bit ser zero, independente do valor já codificado (q^{p+1}), é sempre maior que meio:

$$p_{m^p|q^{p+1}}\left(0\left|q^{p+1}\right.\right) > \frac{1}{2}. \tag{3.19}$$

É interessante notar que, como a função é simétrica, essa probabilidade não depende do sinal do coeficiente.

Por último, é realizado a passagem de *cleanup*. Nele são codificados os coeficientes que não foram incluídos nas passagens anteriores, ou seja, coeficientes ainda insignificantes e que não possuem nenhum vizinho que tenha se tornado significante em planos de bits anteriores. Da mesma forma que na passagem de significância, caso o coeficiente se torne significante, em seguida o sinal é codificado.

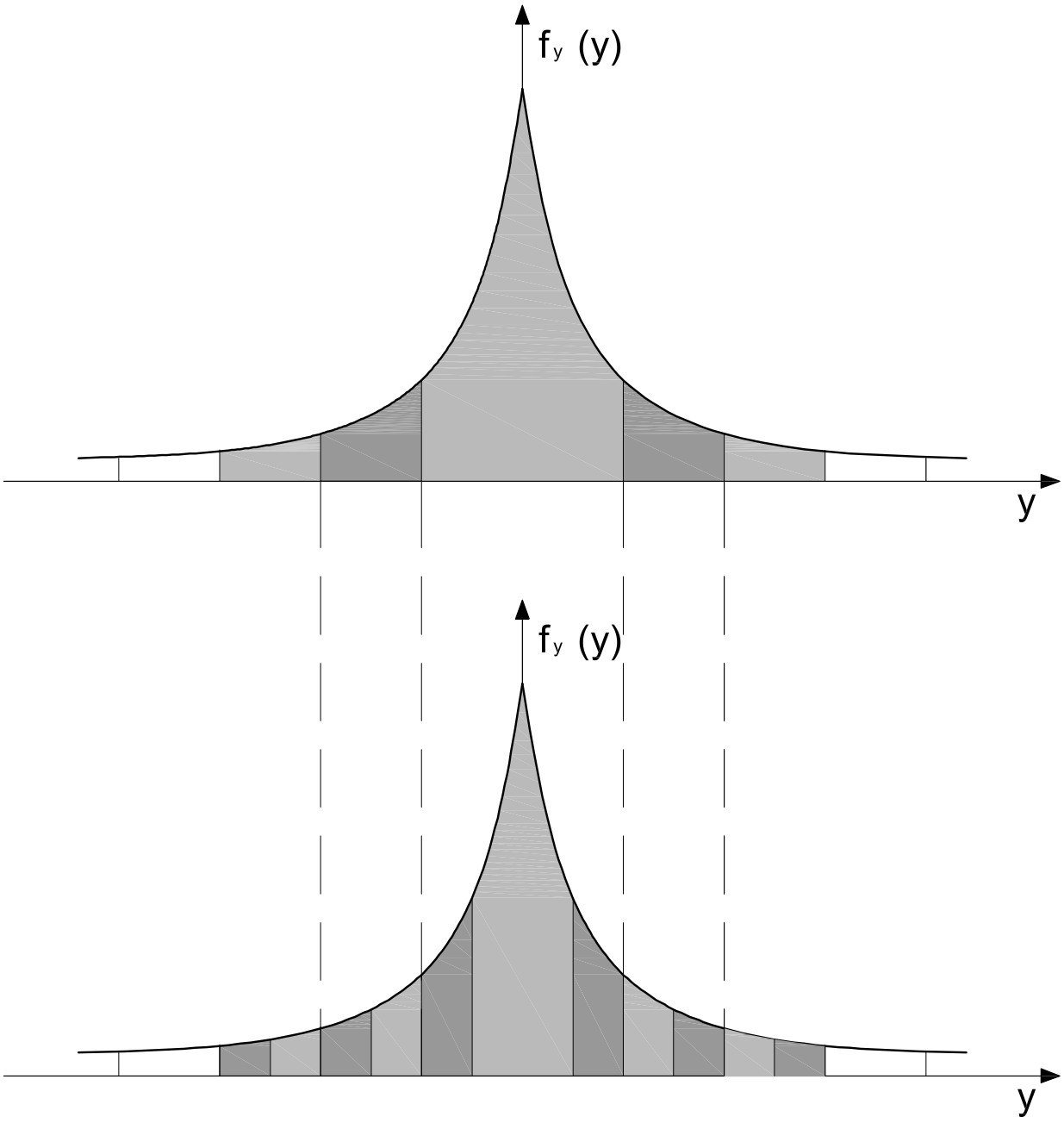


Figura 3.5: Função de densidade de probabilidade típica para os coeficientes de uma transformada. No refinamento da precisão de um coeficiente, é sempre mais provável que o coeficiente esteja mais próximo de zero (próximo *bit* seja zero).

3.1.6 Região de Interesse

A utilização de região de interesse permite a escolha de uma região da imagem para qual será alocada uma maior quantidade de bits, em detrimento do resto da imagem. Com isto, é possível que partes diferentes de uma imagem sejam codificadas com fidelidades diferentes, preservando a região escolhida. Um exemplo pode ser visto na Figura 3.6.

O processo é bem ilustrado na Figura 3.7. Nele uma máscara binária da região de interesse é gerada e os coeficientes no interior são deslocados para planos de bits superiores. No caso do JPEG2000, é utilizada uma técnica denominada MAXSHIFT. Nesta técnica os coeficientes são deslocados de forma que o plano de bits menos significativos da região de interesse pelo menos coincida com o plano de bits mais significativos dos coeficientes do resto da subbanda. Com isso, o decodificador receberá primeiramente os coeficientes do interior da região de interesse. Garantindo a eles uma maior qualidade na reconstrução em qualquer ponto de truncamento. A técnica utilizada é muito interessante por não ser necessário o envio da máscara binária que define a região de interesse. Apenas o quanto a região de interesse foi escalonada.

3.1.7 EBCOT

O JPEG2000 é baseado em um conceito conhecido como codificação progressiva de bloco com truncamento ótimo ou EBCOT (do inglês, *embedded block coding with optimal truncation*). Nele, os coeficientes das subbandas são divididos em blocos relativamente pequenos, tipicamente de 32×32 ou 64×64 coeficientes, conforme mostrado na Figura 3.8. Cada bloco é codificado independentemente gerando um *bitstream* progressivo, que pode ser ser truncado em qualquer ponto e ainda seja decodificádavel (obtendo distorções diferentes). Contudo, é interessante que esse ponto seja escolhido de forma que maximize uma função de custo do tipo:

$$J = D + \lambda R, \tag{3.20}$$

onde D é a distorção da imagem após a codificação, R é a taxa da imagem codificada e λ é um multiplicador de Lagrange.

Vamos denotar o comprimento em bits do bloco i por L_i. Assim podemos estimar o tamanho final do arquivo codificado como sendo:

(a)

(b)

Figura 3.6: Imagem Lena comprimida a 0,1 bpp. (a)Compressão utilizando de região de interesse. A região de interesse escolhida foi o rosto. (b) compressão sem utilização de região de interesse na mesma taxa.

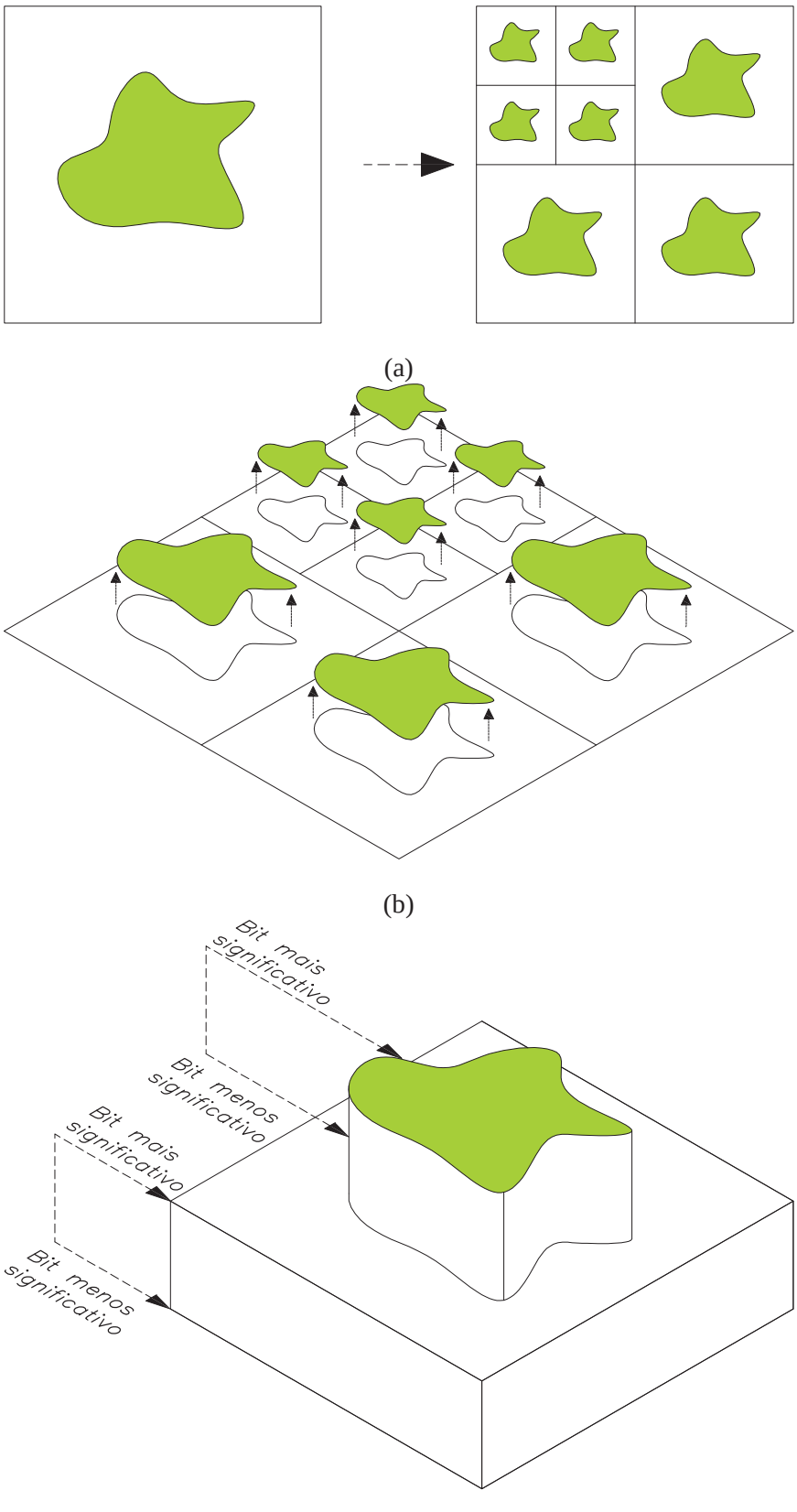


Figura 3.7: (a) Máscara binária da região de interesse escolhida na imagem e nas subbandas. (b) Escalonamento dos coeficientes da subbanda. (c) Escalonamento utilizando técnica MAXSHIFT, adotada pelo JPEG2000.

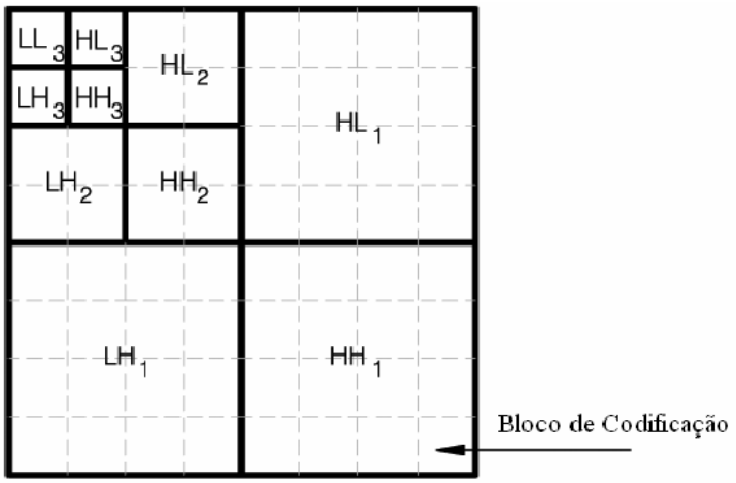


Figura 3.8: Blocos de codificação espacialmente organizados dentro de subbandas provenientes de uma decomposição wavelet de três níveis

$$R = \sum_i L_i. \tag{3.21}$$

A distorção global também pode ser calculada a partir da soma das contribuições das distorções de cada bloco de codificação

$$D = \sum_i D_i. \tag{3.22}$$

Para computar a distorção devida a um determinado bloco, como a transformada não é ortogonal, é necessário levar em conta os filtros utilizados para geração da subbanda. Assim a distorção do bloco i é dada por:

$$D_i = G_{b_i} \sum_j \left(y_i \hat{[}j] - y_i [j] \right)^2, \tag{3.23}$$

onde G_{b_i} é a energia, norma $\mathcal{L}_2$, do filtros de análise da subbanda ao qual o bloco pertence.

3.2 PADRÃO H.264/AVC

Assim como JPEG2000, para a definição de um novo padrão de codificação de vídeo, ITU-T VCEG (do inglês, *Video Coding Experts Group*) e o ISO/IEC MPEG (do inglês, *Moving Picture Experts Group*) se uniram em uma parceria que ficou conhecida como JVT (do inglês, *Joint Video Experts*). A norma

emitida pela ISO/IEC foi denominada MPEG-4 parte 10 ou MPEG-4/AVC e pela ITU-T H.264. A partir desse momento o padrão será denominado apenas por H.264/AVC. Atualmente, ele representa o estado da arte em codificadores de vídeo, tendo sido adotado pelo Sistema Brasileiro de Televisão Digital e sistemas de codificação de vídeo em alta definição, como por exemplo, HD-DVD (High-Definition DVD) e Blu-ray, ambos candidatos à sucessão do DVD.

Esse padrão trouxe uma grande melhoria no desempenho em codificação de vídeo quando comparado com seus predecessores. Especificamente com relação ao MPEG-2, o H.264/AVC chega a ter um desempenho duas vezes superior, ou seja, cerca da metade da distorção para uma mesma taxa[18][19]. No entanto, essa melhoria veio acompanhada de imenso aumento de complexidade computacional, principalmente com relação à estimação de movimento, responsável por até 99% do custo computacional[20].

Talvez o fato mais surpreendente sobre o H.264/AVC seja a superioridade dele na codificação de imagens estáticas. Tendo desempenho superior, inclusive, ao JPEG2000. Isso pode ser observado nas curvas de PSNR (do inglês, *Peak signal-to-noise ratio* ou razão pico sinal-ruído), definida na equação 3.24, por taxa apresentadas na Figura 3.9. No exemplo apresentado, o desempenho do H.264/AVC se mostra maior em todas as taxas.

$$PSNR = 20log\left(\frac{MAX}{\sqrt{MSE}}\right) \tag{3.24}$$

onde MAX é o valor máximo que uma amostra pode atingir e MSE esta definido como:

$$MSE = \frac{1}{M}\sum_{i}\sum_{j}\left(P_{ij} - Pr_{ij}\left(x,y\right)\right)^2 \tag{3.25}$$

3.2.1 Predição

Como será visto adiante, o H.264/AVC utiliza uma transformada de bloco baseada na DCT. Contudo, para que haja uma diminuição da redundância entre blocos e conseqüente aumento na eficiência de codificação, no padrão está prevista a utilização de blocos previamente codificados e reconstruídos para a predição do bloco sendo codificado. Apenas o resíduo de predição, diferença entre valor predito e as amostras reais, é codificado. A predição deve ser feita com a versão reconstruída, obtida a partir de um decodificador local, para evitar diferenças nas amostras disponíveis ao codificador e ao decodificador. Isto ocasionaria uma acumulação nos erros obtidos em cada bloco, fenômeno conhecido como escorregamento (*drift*, em inglês). Da mesma forma que padrões anteriores, uma das fontes de predição são *pixels*

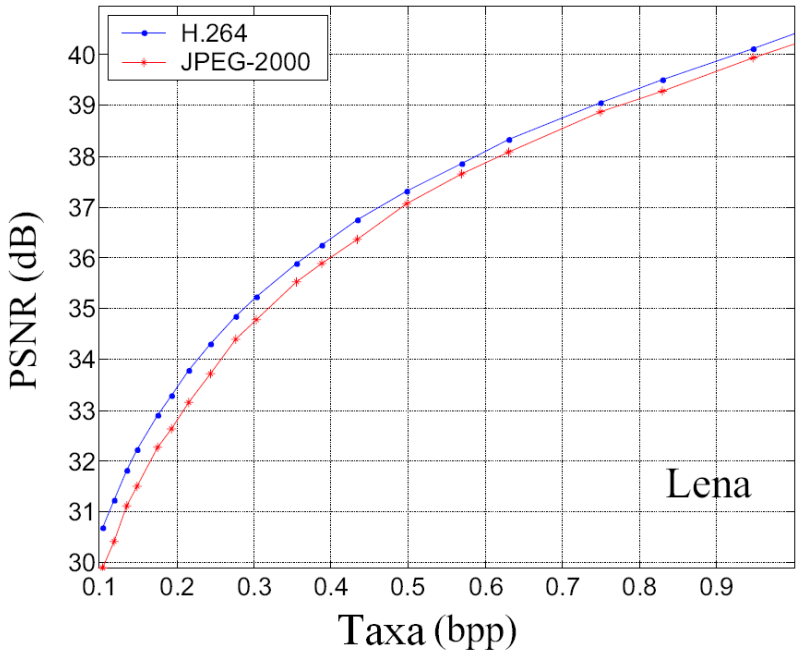


Figura 3.9: Comparação entre H.264/AVC e JPEG2000 para a imagem Lena em tons de cinza.

pertencentes a blocos de quadros anteriormente codificados e reconstruídos (predição entre quadros). O H.264/AVC utiliza, adicionalmente, blocos vizinhos do mesmo quadro, predição intra quadro.

3.2.1.1 Predição entre quadros

Em uma seqüência típica de vídeo, excluindo-se mudanças de cena, entre quadros vizinhos há uma redundância muito grande já que tendem a ser muito semelhantes. Isso acontece por esses quadros serem obtidos com uma diferença de tempo muito pequena. Considerando que a câmera permaneça fixa, as diferenças serão devidas a pequenas movimentações de objetos, e o fundo permanece constante. Um exemplo disso pode ser visto na Figura 3.10, onde vemos que as diferenças entre os dois quadros ocorrem devido a uma pequena movimentação da cabeça da mãe.

No caso apresentado pode-se utilizar um quadro previamente codificado como uma excelente fonte de predição para o quadro a ser codificado. Contudo, quando há mais movimento, seja por um número maior de objetos que modificam a posição, seja por movimentos maiores ou por movimento da câmera, as diferenças entre os dois quadros tendem a ser maiores. Para melhorar a predição costuma-se levar em conta esses movimentos na predição.

Inicialmente, deve-se estimar os movimentos ocorridos entre os dois quadros. Para isso, o quadro

(a)

(b)

(c)

Figura 3.10: Seqüência Mother_daughter. (a) Primeiro quadro(b) Segundo quadro (c) Diferença entre os quadros multiplicada por um gnaho de 5 acrescida de 128 para permitir a visualização.

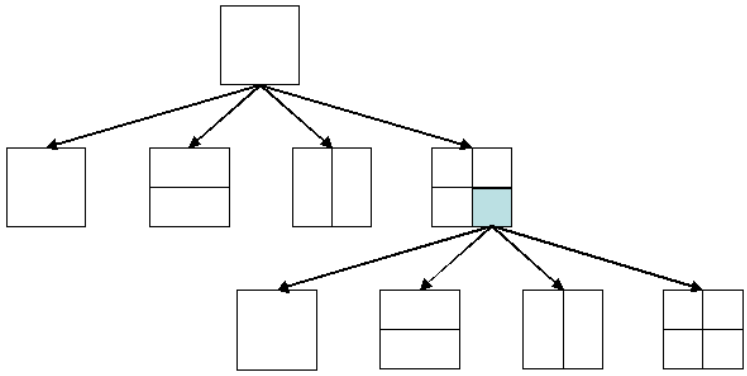


Figura 3.11: Partição *quadtree* utilizada na estimação de movimento.

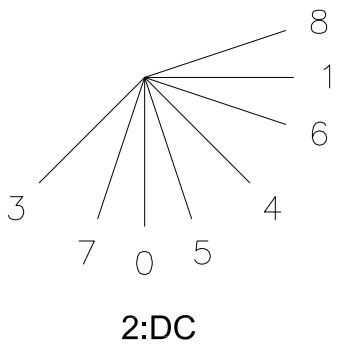


Figura 3.12: Direções utilizadas nos modos de predição para blocos de 4×4 *pixels*.

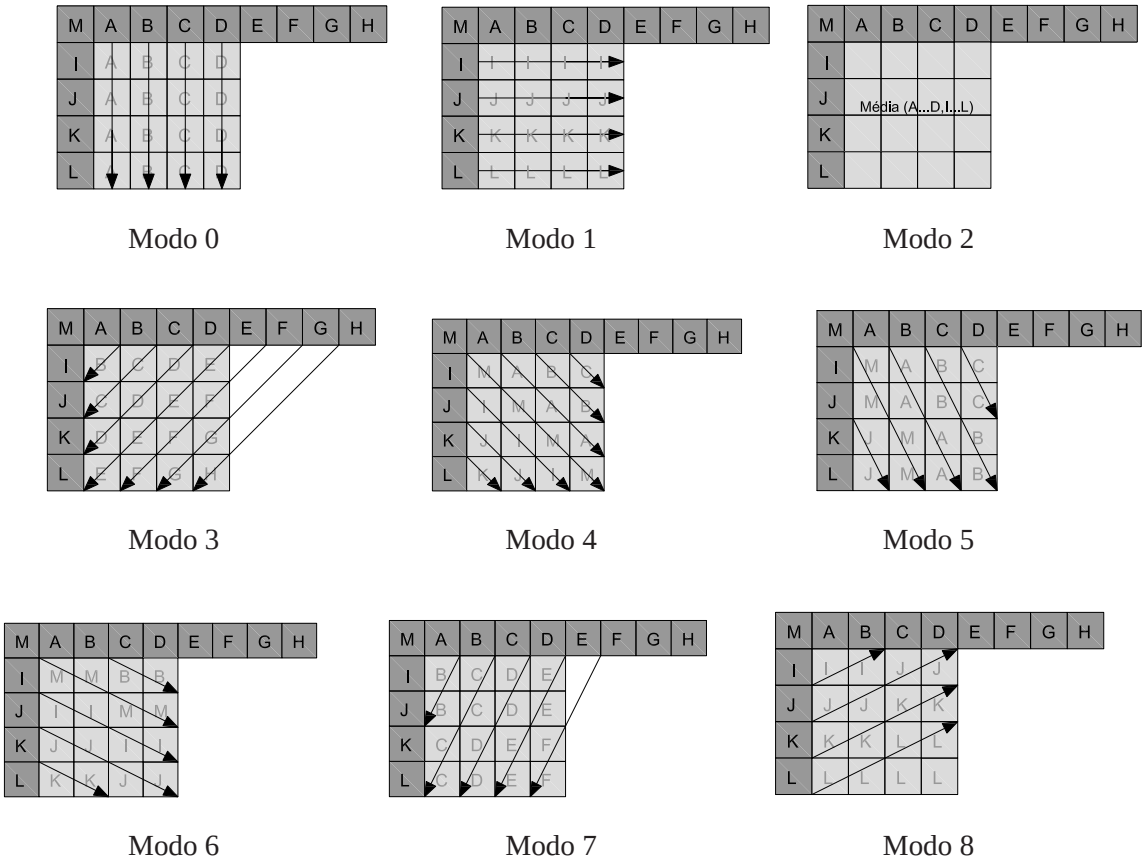
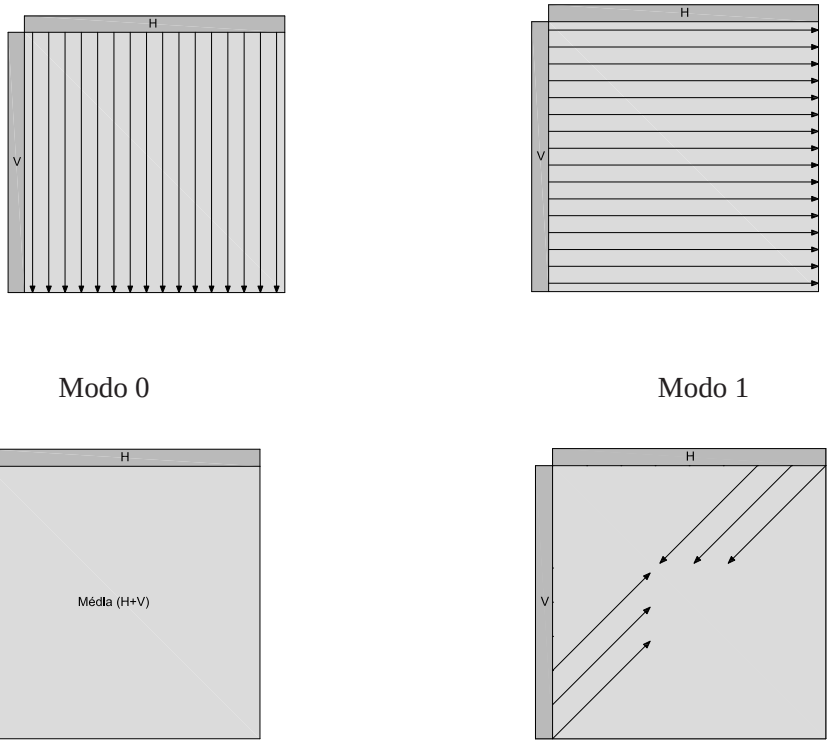


Figura 3.13: Modos de predição para blocos de 4×4 *pixels*.



Modo 0 Modo 1

Modo 2 Modo 3

Figura 3.14: Modos de predição para blocos 16×16

sendo codificado é dividido em blocos retangulares. Para cada bloco, procura-se um bloco que mais se assemelhe no quadro previamente codificado que está sendo utilizado como referência. Embora o critério a ser utilizado para medir a semelhança (ou diferença) não seja normativo, costuma-se utilizar como métrica erro médio quadrático, MSE (*Mean squared error*), definido na Equação 3.25, ou soma das diferenças absolutas, SAD (*Sum of absolute differences*), definida na Equação 3.26. O bloco escolhido é o que minimiza a função escolhida.

$$SAD = \sum_i \sum_j \left| P_{ij} - Pr_{ij}\left(x, y\right) \right|, \tag{3.26}$$

onde P_{ij} é um *pixel* do bloco do quadro atual, $Pr_{ij}\left(x, y\right)$ é um *pixel* do bloco no quadro de referência (previamente codificado e reconstruído) candidato a predição (deslocado em x e y) e N é o número de *pixels* pertencentes ao bloco. Pode-se utilizar funções de custo mais complexas que levam em conta tanto taxa quanto distorção.

A diferença de posição entre os blocos, denominado vetor de movimento, é codificado. Além disso o codificador possui um decodificador local fazendo com que o quadro de referência seja a mesma versão que estará disponível ao decodificador. Esses dois fatos permitem que o decodificador possa realizar a mesma predição obtendo o mesmo quadro predito que o codificador. Dessa forma somente a diferença entre a predição e o bloco atual necessita ser codificada. Todo esse processo é ilustrado na Figura 3.15.

No H.264/AVC é possível particionar os blocos para obter uma melhor compensação de movimento, técnica conhecida como *quadtree* Os macroblocos, blocos de 16×16 podem ser divididos em blocos de 16×8, 8×16, 8×8 *pixels*. Os blocos de 8×8 podem novamente se divididos em blocos de 8×4, 4×8 e 4×4 *pixels*. A técnica *quadtree* é ilustrada na Figura 3.11.

Em alguns casos, é possível que seja feita uma melhor predição pode ser feita a partir de amostras interpoladas do quadro de referência. Desta forma, o H.264/AVC interpola os quadros utilizados como referência por um fator de quatro em cada dimensão, o que permite uma estimação de movimento com precisão de um quarto de *pixel*.

Outra característica do O H.264/AVC é permitir que possam ser usados até 16 quadros de referência passados como futuros (a ordem de codificação é diferente da ordem de exibição para permitir a busca em quadros futuros).

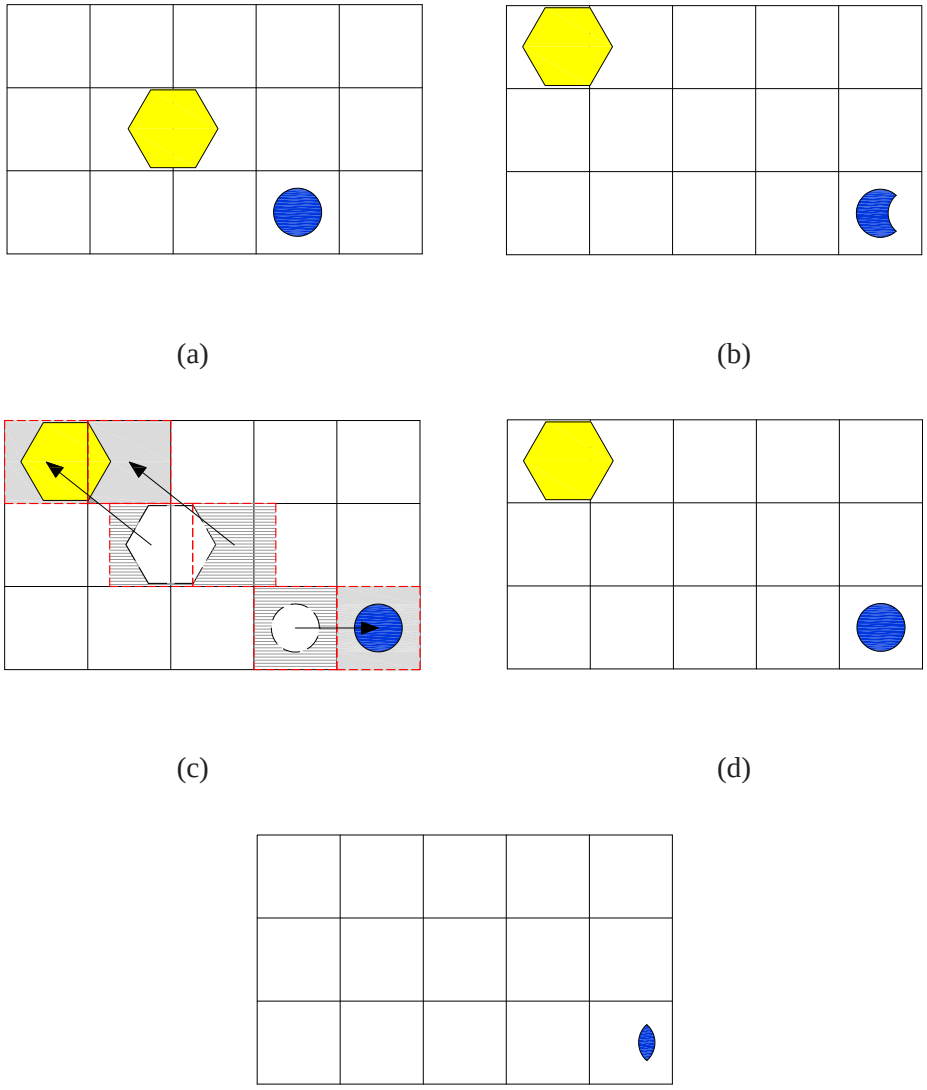


Figura 3.15: Estimação e compensação de movimento. (a) Quadro previamente codificado utilizado como referência para predição. (b) Quadro sendo codificado. (c) Estimação de movimento indicando de onde vieram os blocos utilizados para predizer o quadro atual no quadro de referência, vetores de movimento. (d) Quadro predito. (c) Resíduo de predição, diferença ente predição e o quadro a ser codificado

3.2.1.2 Predição Intra Quadro

Como mencionado anteriormente, o H.264/AVC utiliza *pixel*s de blocos vizinhos previamente codificados e reconstruídos para predizer o bloco a ser codificado. Assim como na estimação de movimento, o tamanho do bloco da predição intra pode variar para melhor se adaptar a situação da imagem. Os tamanhos possíveis para a luminância são 4×4, 8×8 (somente disponível no perfil FREext - *Fifelity Range Extensions*)[21] e 16×16.

Nas Figura 3.13 e 3.12, é possível ver os modos de predição para blocos de 4×4 que são descritos a seguir. Os ângulos são descritos a partir da direção do modo 1, predição horizontal, com valores positivos significando rotação no sentido horário.

- *Modo 0 - Vertical*: As amostras A-D são extrapoladas verticalmente, $90°$.
- *Modo 1 - Horizontal*: As amostras I-L são extrapoladas horizontalmente.
- *Modo 2 - DC*: Todas as amostras são preditas a partir da média das amostras A-D e I-L.
- *Modo 3 - Diagonal abaixo à esquerda*: As amostras são interpoladas em um ângulo $135°$
- *Modo 4 - Diagonal abaixo à direita*: As amostras são interpoladas em um ângulo de $45°$.
- *Modo 5 - Vertical direita*: As amostras são interpoladas em um ângulo de $63,4°$.
- *Modo 6 - Horizontal abaixo*: As amostras são interpoladas em um ângulo de $26,6°$.
- *Modo 7 - Vertical esquerda*: As amostras são interpoladas em um ângulo de $116,6°$.
- *Modo 8 - Horizontal acima*: As amostras são interpoladas em um ângulo de $-26,6°$.

A predição 16×16 possui apenas quatro modos apresentados na Figura 3.14.

- *Modo 0 - Vertical*: As amostras de H são extrapoladas verticalmente.
- *Modo 1 - Horizontal*: As amostras de V são extrapoladas horizontalmente.
- *Modo 2 - DC*: Todas as amostras são preditas a partir da média das amostras de H e V.
- *Modo 3 - Planar*: Um plano é ajustado às amostras de H e V.

A predição de blocos de 8×8 é muito semelhante a apresentada para blocos de 4×4.

3.2.2 Transformação

O H.264/AVC, assim como seus predecessores, utiliza transformação no resíduo de predição antes de realizar a codificação. Contrariamente a outros padrões, são implementadas transformações de tamanhos variáveis que podem ser escolhidas em cada bloco dependendo da otimização utilizada. Uma das grandes inovações do padrão é a utilização de transformadas inteiras que traz simplicidade computacional (apenas somas, subtrações e deslocamentos são necessários no núcleo da transformada). Isto permite a implementação em arquiteturas de ponto fixo com aritmética de apenas 16 bits.

Com intuito de ilustrar a transformação inteira utilizada no H.264/AVC, será apresentada adiante apenas a versão de 4×4. Reescrevendo a matriz da Equação 2.29 para o caso de 4×4, obtém-se:

$$A = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \sqrt{\frac{1}{2}}cos\left(\frac{\pi}{8}\right) & \sqrt{\frac{1}{2}}cos\left(\frac{3\pi}{8}\right) & -\sqrt{\frac{1}{2}}cos\left(\frac{3\pi}{8}\right) & -\sqrt{\frac{1}{2}}cos\left(\frac{\pi}{8}\right) \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ \sqrt{\frac{1}{2}}cos\left(\frac{3\pi}{8}\right) & -\sqrt{\frac{1}{2}}cos\left(\frac{\pi}{8}\right) & \sqrt{\frac{1}{2}}cos\left(\frac{\pi}{8}\right) & -\sqrt{\frac{1}{2}}cos\left(\frac{3\pi}{8}\right) \end{bmatrix} \tag{3.27}$$

Isso pode ser facilmente reescrito como:

$$A = \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix}, \tag{3.28}$$

onde $a = \frac{1}{2}$, $b = \sqrt{\frac{1}{2}}cos\left(\frac{\pi}{8}\right)$ e $c = \sqrt{\frac{1}{2}}cos\left(\frac{3\pi}{8}\right)$.

A transformada $Y = AXA^T$ pode ser reescrito como $Y = CXC^T \otimes E$. O símbolo $\otimes$ significa que cada matriz $\left(CXC^T\right)$ é multiplicado pelo elemento na mesma posição da matriz E.

$$C = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & d & -d & -1 \\ 1 & -1 & -1 & 1 \\ d & -1 & 1 & -1 \end{bmatrix} \tag{3.29}$$

e

$$E = \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix}. \tag{3.30}$$

onde $d = c/b$, que é aproximadamente 0.4142. Esse valor é aproximado para 0.5, o que permite que o núcleo da transformação, $Y = CXC^T$, contenha apenas adições, somas e deslocamentos. Essa mudança faria com que a transformação deixasse de ser ortogonal. Para evitar isso, altera-se o valor de b para $\sqrt{2/5}$. Assim, e com escalonamento de algumas linhas e colunas para evitar a perda de precisão, a transformada completa fica:

$$Y = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} X \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \right) \otimes \begin{bmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{bmatrix}. \tag{3.31}$$

O escalonamento pode ser incorporado pela quantização, economizando uma operação de multiplicação por coeficiente.

O H.264/AVC ainda possui uma transformada inteira semelhante a apresentada de 8×8. Para os modos de 16×16, utiliza-se transformada de 4×4 e uma decomposição adicional é realizada nos coeficientes DC com uma transformada Hadamard, apresentada na Equação 2.32

3.2.3 Codificação de entropia

A codificação de entropia do H.264/AVC fica a cargo de dois codificadores aritméticos disponíveis no padrão. O primeiro é denominado *Context-adaptive variable-length coding* ou simplesmente CAVLC. Ele está disponível em todos os perfis e tem complexidade relativamente baixa.

Há ainda o codificador conhecido como CABAC (*Context-adaptive binary arithmetic coding*). Comparado ao CAVLC, o CABAC tem complexidade computacional elevada. Ele está disponível apenas nos perfis *Main* e *High* e possui desempenho superior. Ambos são adaptativos e contextuais, ou seja, as tabelas de estatística se adaptam aos dados sendo codificados e podem usar vizinhança para a escolha da tabela mais adequada.

4 MODIFICAÇÕES NOS CODIFICADORES PADRÃO H.264/AVC E JPEG2000

O presente capítulo apresenta as modificações realizadas nos codificadores padrão JPEG2000 e H.264. No JPEG2000, foram comparados o desempenho de transformadas sobrepostas, *wavelets* e transformadas de bloco (DCT). No H.264, além das comparações entre as transformada antes mencionadas, avaliou-se o desempenho da predição intra quadro.

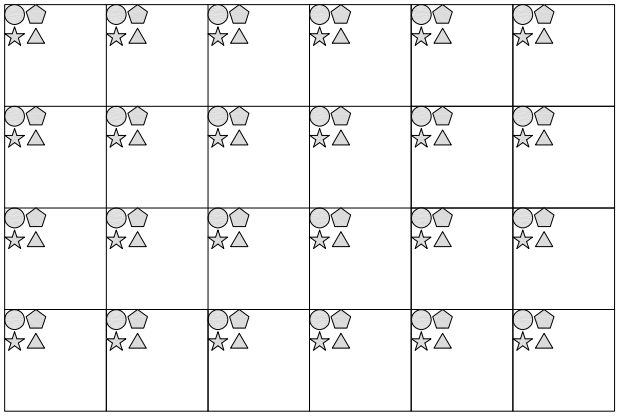
4.1 ALTERAÇÕES NO JPEG2000 PARA UTILIZAÇÃO DE TRANSFORMADAS DE BLOCO E TRANSFORMADAS SOBREPOSTAS

Para utilizar o JPEG2000 para a codificação de imagens utilizando outras transformadas diferentes da originalmente utilizada, algumas modificações tiveram que ser feitas.

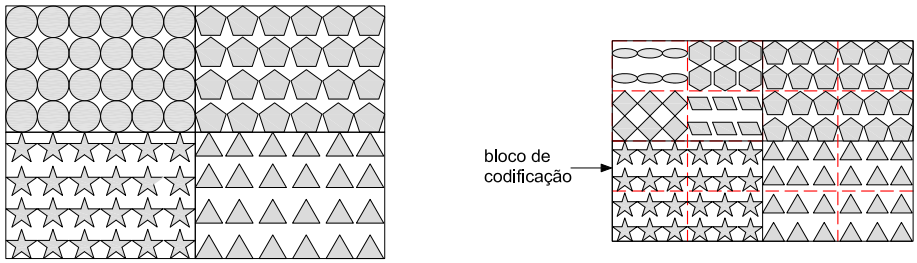
Como mencionado anteriormente, no JPEG2000, as subbandas são divididas em blocos que são codificados independentemente. Com o intuito de codificar os coeficientes provenientes de transformadas de bloco ou transformadas sobrepostas, faz-se necessário uma mudança na forma como os coeficientes são ordenados. A reordenação que apresentou melhor desempenho consistia em agrupar cada subbanda obtida nos diversos blocos de transformação. Assim, como no JPEG2000 original, cada bloco de codificação contém apenas coeficientes de uma única subbanda.

Visando melhorar o desempenho de codificação, nas transformadas de bloco e sobrepostas, foi realizada uma decomposição adicional no nível DC. Essa decomposição foi feita utilizando dois bancos de filtros diferentes. No primeiro caso, agrupava-se os coeficientes DC de todos blocos, mantendo-se as posições relativas conforme mostra a Figura 4.1 formando uma imagem de menor dimensão. Nessa imagem, aplica-se um nível de decomposição *wavelet*. Para os testes foi utilizado o banco de filtros padrão do JPEG2000, Daubechies 9/7. Como alternativa, a decomposição já apresentada, agrupou-se os coeficientes DC de quatro blocos em um novo bloco de 2×2 coeficientes. A este aplica-se uma decomposição Hadamard. Todo o processo é ilustrado na Figura 4.2. A reordenação final é mesma apresentada anteriormente.

Deve-se lembrar que o padrão JPEG2000 tem uma codificação progressiva e segundo o paradigma do



(a)



(b) (c)

Figura 4.1: Reordenação para transformadas de bloco. Apenas quatro subbandas são ilustradas para facilitar a visualização. (a) Blocos transformados. (b) Agrupamento da subbandas. (c) Agrupamento final das subbandas após a decomposição do DC por *wavelets* (cada nova subbanda é ilustrada por uma cor). Observe que os blocos de codificação possuem apenas coeficientes de uma subbanda.

EBCOT o truncamento deve ser feito de forma ótima. Isto é obtido minimizando a função de custo (taxa distorção) apresentada na Equação 3.20. Como foi apresentado nas Equações 3.22 e 3.23, a distorção é computada como sendo a soma das distorções de cada bloco de codificação ponderadas pela energia do filtro que gerou aquela subbanda. Para tornar o codificador independente da transformada utilizada, todos os pesos foram feitos iguais, fazendo com que todas as subbandas tenha a mesma importância na computação da distorção. Esta escolha torna a otimização taxa distorção sub-ótima para transformadas bi-ortogonais e privilegia o desempenho de transformadas ortonormais.

Os coeficientes originais, gerados a partir da decomposição *wavelet*, são substituídos pelos coeficientes reordenados da forma apresentada acima. As demais operações como quantização e codificação são realizadas sem maiores alterações pelo JPEG2000.

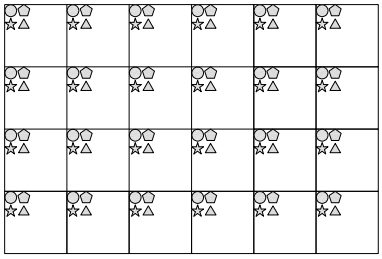
4.2 ALTERAÇÕES NO H.264 PARA UTILIZAÇÃO DE TRANSFORMADAS DE *WAVELETS* E TRANSFORMADAS SOBREPOSTAS

O fato do H.264, que utiliza uma transformada de bloco baseada na DCT, ter desempenho superior ao JPEG2000 impressiona, pois a literatura mostra que *wavelets* apresentam desempenho para codificação superior a transformadas de blocos. Contudo, como foi apresentado na Seção 3.2.1.2, o H.264 utiliza uma técnica inovadora (predição intra quadro) para aproveitar redundância entre blocos previamente codificados e o bloco sendo processado, melhorando a eficiência de codificação. Em princípio, isto é uma forma de sobreposição de blocos, já que os coeficientes transformados não dependem apenas dos píxels do bloco sendo codificado. Com a intenção de comparar a eficiência de codificação da predição intra quadro aliada com DCT modificada com transformadas sobrepostas e *wavelets*, diversas modificações tiveram que ser realizadas .

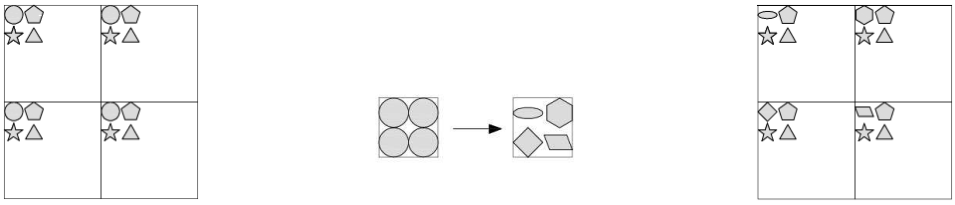
Inicialmente, para utilização de outras transformada, deveria-se forçar um único tamanho de bloco para todos os macroblocos, no caso, o tamanho escolhido foi de 8×8 píxels. Essa escolha foi feita por estar condizente com o tamanho de diversos codificadores como, por exemplo, H.263, MPEG-2 e JPEG, e, como será visto adiante, se adequar a 4 níveis de decomposição *wavelet*.

Foi necessário, ainda, desabilitar a predição intra quadro[1]. Dois principais motivos exigiram essa modificação. Primeiramente, o interesse é em uma comparação entre o esquema de predição intra

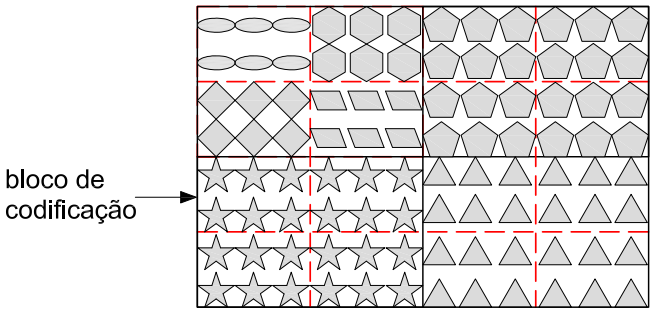
[1] A desabilitação foi realizada tornando os blocos vizinhos que poderiam ser utilizados para predição intra quadro indisponíveis, como se o bloco fosse sempre o do canto superior esquerdo.



(a)

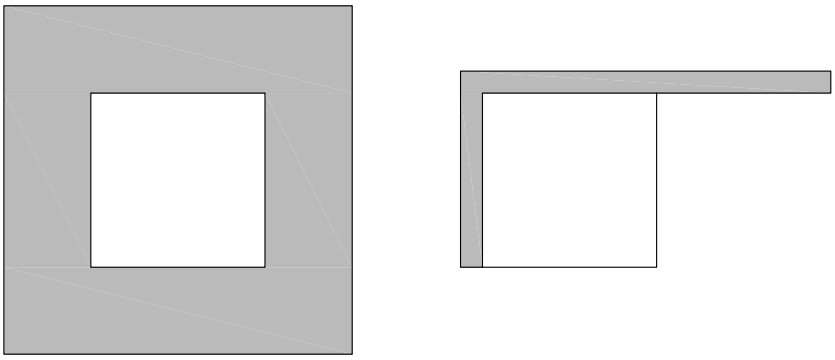


(b) (c) (d)



(e)

Figura 4.2: Agrupamento dos coeficientes DC de quatro blocos e transformação.(a) Coeficientes dos blocos. (b) Coeficientes de quatro blocos (c) Bloco de tamanho 2×2 de coeficientes DC. (d) Bloco após decomposição Hadamard. (e) Organização final.



(a) (b)

Figura 4.3: Comparação do suporte de predição intra quadro e transformadas sobrepostas. (a) Transformada sobreposta com fator de sobreposição 2. (b) Predição intra quadro

quadro utilizado no H.264 com outras formas de sobreposição. Somente a título ilustrativo o suporte das transformadas sobrepostas, *wavelets* e predição intra quadro é mostrado na Figura 4.3. O outro motivo é que a predição intra quadro é uma técnica recursiva e, por isso, incompatível com outras formas de sobreposição utilizadas nesse trabalho, pois, para evitar o fenômeno de escorregamento (*drift*, em inglês), a predição é realizada com a versão reconstruída dos blocos previamente codificados. Nas transformadas sobrepostas, por exemplo, é possível obter a reconstrução de um bloco antes que todos os vizinhos com o qual há sobreposição tenham sido transformados e quantizados.

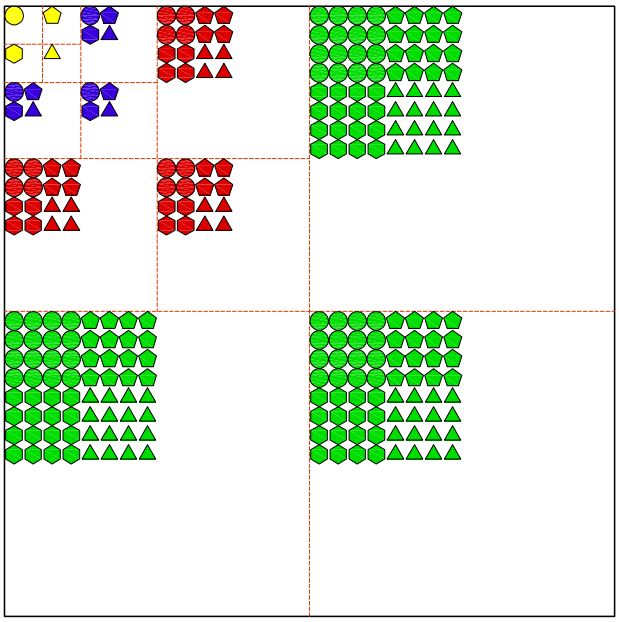
É importante salientar que as duas mudanças citadas anteriormente (utilizar apenas blocos de 8×8 *pixels* e não realizar predição intra quadro) não tornam o codificador incompatível com o padrão. Sendo assim, mesmo que haja apenas um modo possível, o codificador ainda sinaliza este modo. Esta sinalização utiliza *bits* de codificação desnecessariamente já que não traz informação útil. Com maiores modificações seria possível remover essa informação lateral aumentando a eficiência do codificador H.264 para a codificação em questão.

Outra adaptação necessária foi a mudança das tabelas de quantização. Como foi mencionado na Seção 3.2.2, o escalonamento da transformação é integrado com a quantização. Para separar esses dois processos,

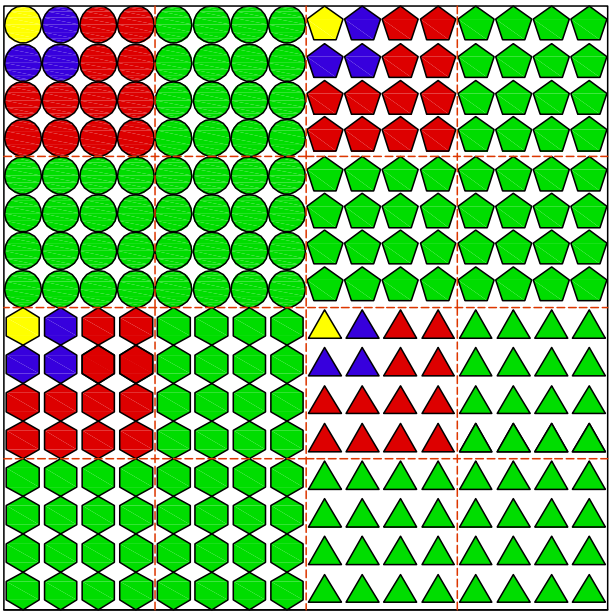
utilizou-se um quantizador escalar de faixa morta nos coeficientes transformados. Todos os passos de quantização foram feitos idênticos para todas as subbandas. Embora isso, eventualmente, possa privilegiar transformadas ortogonais, as transformadas bi-ortogonais utilizadas são aproximadamente ortogonais.

Por último, para que fosse feita uma comparação justa e levasse em conta apenas predição intra quadro/DCT, transformadas sobrepostas e *wavelets* (técnicas em comparação) a otimização taxa distorção do H.264 foi desabilitada.

Para as transformadas sobrepostas e de bloco, uma simples substituição dos coeficientes seria suficiente. Contudo, da mesma forma que foi feito no JPEG2000, para aumentar a eficiência de codificação uma decomposição Hadamard de 2×2 adicional foi feita no DC. Para os coeficientes da *wavelet*, uma simples substituição não seria suficiente. Foi necessário fazer uma reordenação. Os coeficientes foram agrupados de acordo com as suas respectivas posições espaciais formando blocos de 8×8 *pixels*. Como utilizou-se 4 níveis de decomposição, o coeficiente que subistituia o coeficiente DC da DCT era proveniente de subbandas diferentes (LL_4, LH_4, HL_4 e HH_4), como ilustrado na Figura 4.4.



(a)



(b)

Figura 4.4: Reordenação dos coeficientes *wavelets* para quatro blocos 8×8 . (a) Coeficientes organizados na forma original. Cada cor representa uma subbanda e cada figura geométrica representa o bloco de 8×8 *pixels* a qual o coeficiente pertencerá após a reordenação. (b) Agrupamento em blocos quatro blocos de 8×8.

5 RESULTADOS

Neste capítulo são apresentados os resultados para os experimentos descritos no Capítulo 4. Inicialmente, serão apresentados os resultados obtidos para um conjunto de imagens e diversas transformadas. Em seguida, são apresentados os resultados para as mesmas imagens utilizando o codificador H.264/AVC.

5.1 RESULTADOS PARA JPEG2000

Inicialmente, apresenta-se, na Figura 5.1, curvas Taxa vs. Distorção com intuito de comparar o desempenho das decomposições utilizadas no DC utilizando *wavelet* e Hadamard. Também são apresentados os resultados para a codificação sem a utilização de qualquer decomposição da subbanda DC. Neste ultimo caso, variou-se o tamanho do bloco de codificação 32×32 e 64×64 coeficientes, maiores blocos que poderiam ser utilizados. Isso acontece pelo tamanho da imagem, 512×512. Cada subbanda contém 64×64 coeficientes, já que a transformada é de 8×8. Caso seja realizado alguma decomposição na subbanda DC, as novas subbandas terão apenas 32×32 coeficientes. Os resultados indicam um melhor desempenho da decomposição *wavelet* quando ambos utilizaram blocos de codificação de mesmo tamanho, 32×32. Contudo, essa vantagem desaparece quando se compara com a codificação utilizando blocos maiores. A imagem utilizada para obtenção dos resultados apresentados, Lena, é a que apresenta maior diferença entre a utilização ou não de decomposição adicional no DC. Sendo assim, na obtenção dos resultados que serão apresentados a seguir, não foi utilizado decomposição.

Nos testes realizados, foram utilizados quatro imagens populares de 512×512: *pixels*, “Barbara”, “Lena”, “Goldhill” e “Baboon”. Utilizando as modificações apresentadas no Capitulo 4 para o codificador padrão JPEG2000, estas imagens foram codificadas utilizando *Wavelet* $9/7$ (3 níveis de decomposição), DCT, GenLot 8×32, GenLot 8×48 e Glbt 8×16. O resultado para a imagem “Barbara” é apresentado na Figura 5.2. Para permitir uma melhor visualização, os resultados são reapresentados na Figura 5.3 de forma diferencial, todos os resultados são apresentados com relação aos resultados da *wavelet*. Os resultados para as demais imagens são apresentados nas Figuras 5.4 a 5.6. Os mesmos testes foram realizados em imagens de maior resolução (1920×1080 *pixels*). Para isso, apenas o primeiro quadro de algumas seqüências de vídeo, “Pedestrian_area”, “Riverbed”, “Rush_hour”, “Station2” e “Sunflower” foram utilizados. Os resultados são apresentados nas Figuras 5.7 à 5.11. Para o caso de imagens de alta definição, as curvas de

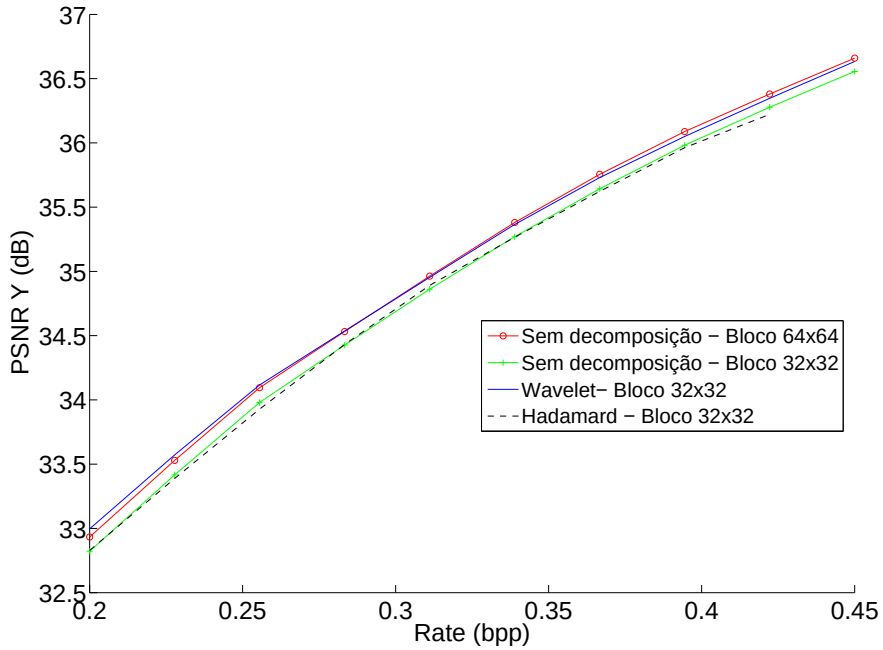


Figura 5.1: Curvas de Distorção vs. Taxa obtidas a partit da codificação da imagem “Lena”. imagem codificada utilizando transformada sobreposta (GenLot 8×48).

PSNR estão limitadas a 0.5 bpp, já que, para esta taxa, apresentavam uma qualidade satisfatoriamente boa, sempre acima de $41dB$.

Nas imagens de “Goldhill” e “Baboon” de 512×512 *pixels*, as transformadas GenLot 8×48 e GenLot 8×32 apresentam uma modesta vantagem com relação à *wavelet* $9/7$ utilizada no JPEG2000. No caso da imagem “Barbara”, esse surpreendente desempenho das transformadas sobrepostas é atípico e foi observado para diversos codificadores [4]. A imagem “Lena” foi a única imagem na qual a GenLot 8×48 obteve desempenho inferior à *Wavelet*, sendo que o desempenho foi semelhante até cerca de uma taxa de0.5 bpp. Como era esperado, a DCT foi a transformada que obteve o pior desempenho.

Para as imagens de alta definição, a vantagem comparativa das transformadas sobrepostas GenLot 8×48 e GenLot 8×32 se mostrou sólida e consistente. Apenas na imagem “Sunflower”, observou-se desempenho destas transformadas semelhantes a *wavelet*. Novamente, a DCT apresenta o pior desempenho. A transformada GenLot 8×32 foi incluída nos testes pelo fato de, em baixas taxas, apresentar menor efeito de *ringing* que a GenLot 8×48, já que os filtros são menores. Contudo, esta transformada obteve desempenho muito inferior para baixas taxas na imagens pedestrian, ““Rush_hour”” e “Sunflower”.

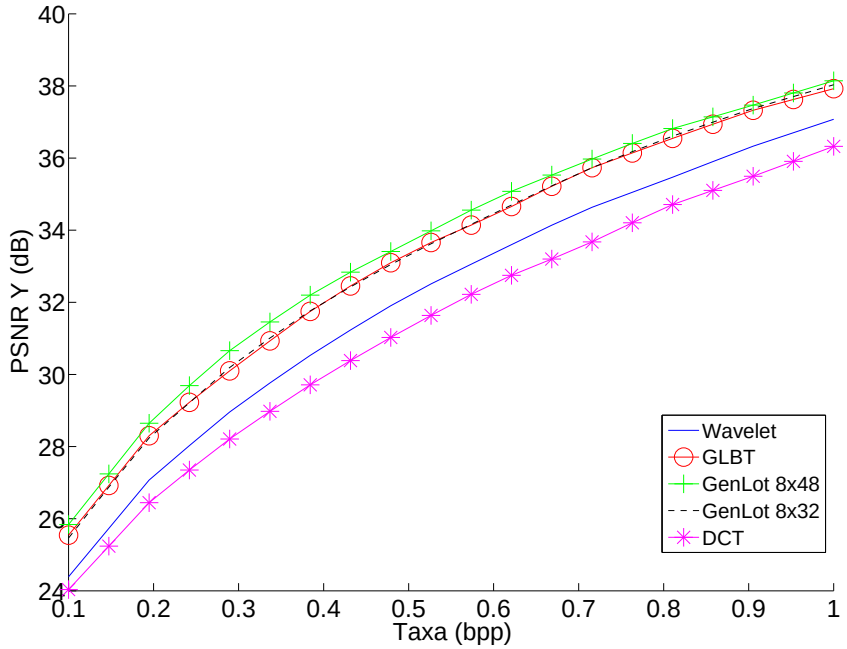


Figura 5.2: Curvas PSNR para a imagem “Barbara”. Codificação realizada utilizando JPEG2000.

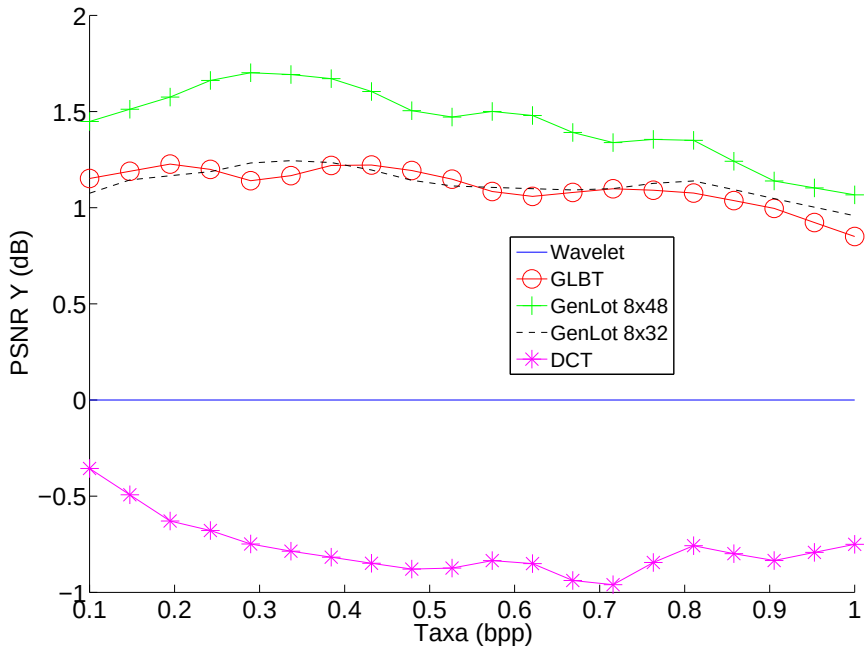


Figura 5.3: Curvas diferencias de PSNR para a imagem “Barbara”. Resultados apresentados de forma relativa à transformada *Wavelet* dB 9/7. Codificação realizada utilizando JPEG2000.

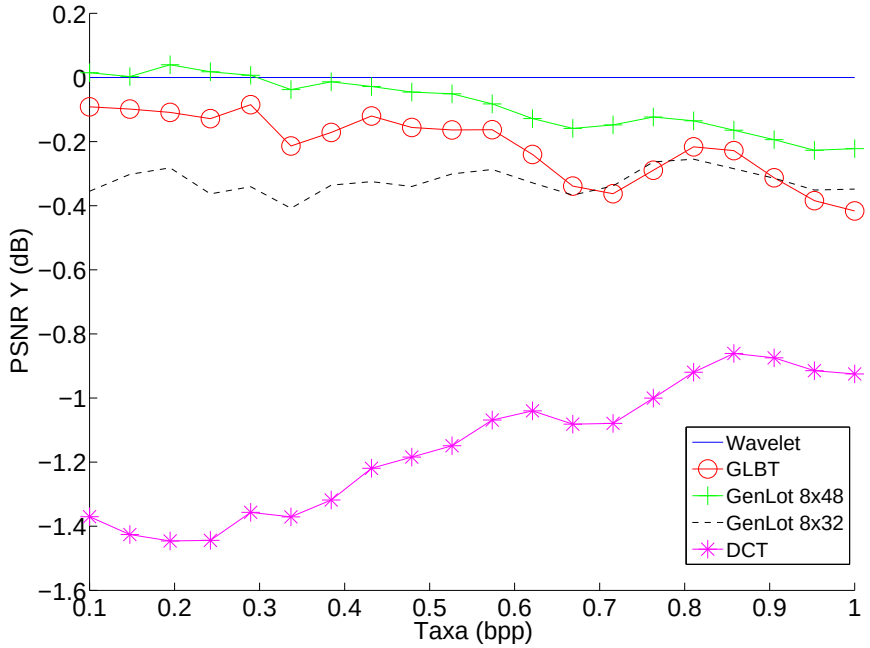


Figura 5.4: Curvas diferencias de PSNR para a imagem “Lena”. Resultados apresentados de forma relativa à transformada *Wavelet* dB 9/7. Codificação realizada utilizando JPEG2000.

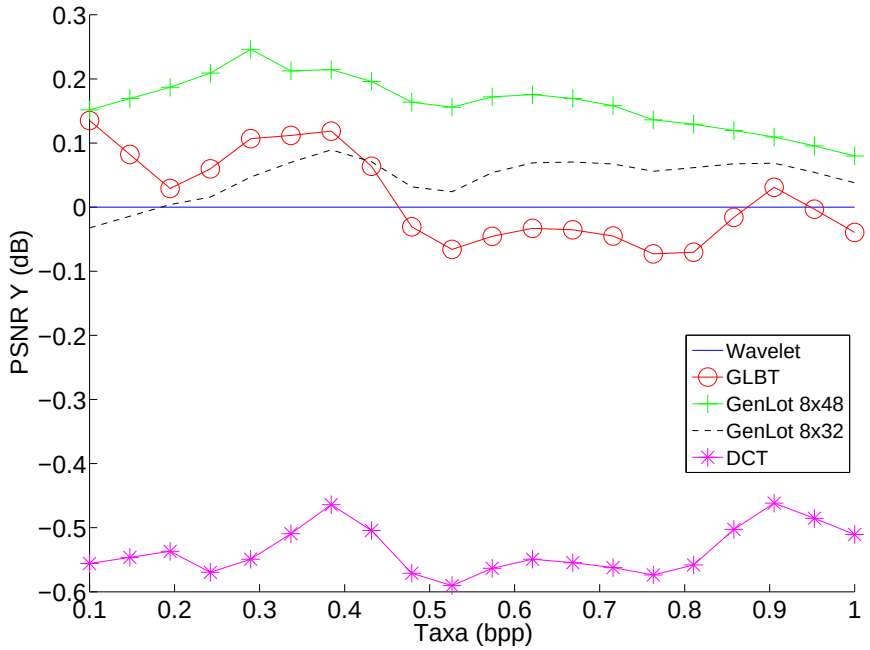


Figura 5.5: Curvas diferencias de PSNR para a imagem “Goldhill”. Resultados apresentados de forma relativa à transformada *Wavelet* dB 9/7. Codificação realizada utilizando JPEG2000.

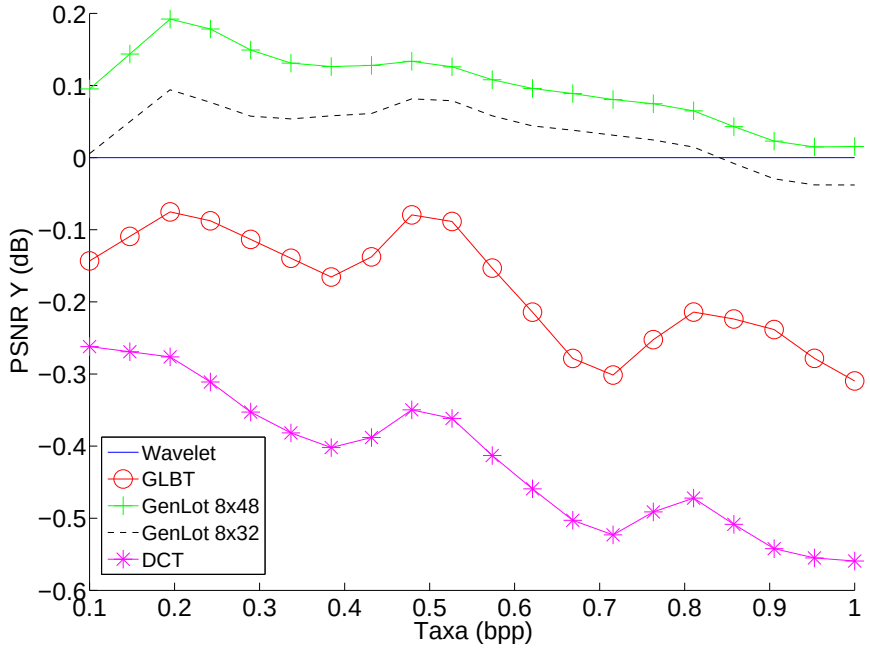


Figura 5.6: Curvas diferencias de PSNR para a imagem “Baboon”. Resultados apresentados de forma relativa à transformada *Wavelet* dB 9/7. Codificação realizada utilizando JPEG2000.

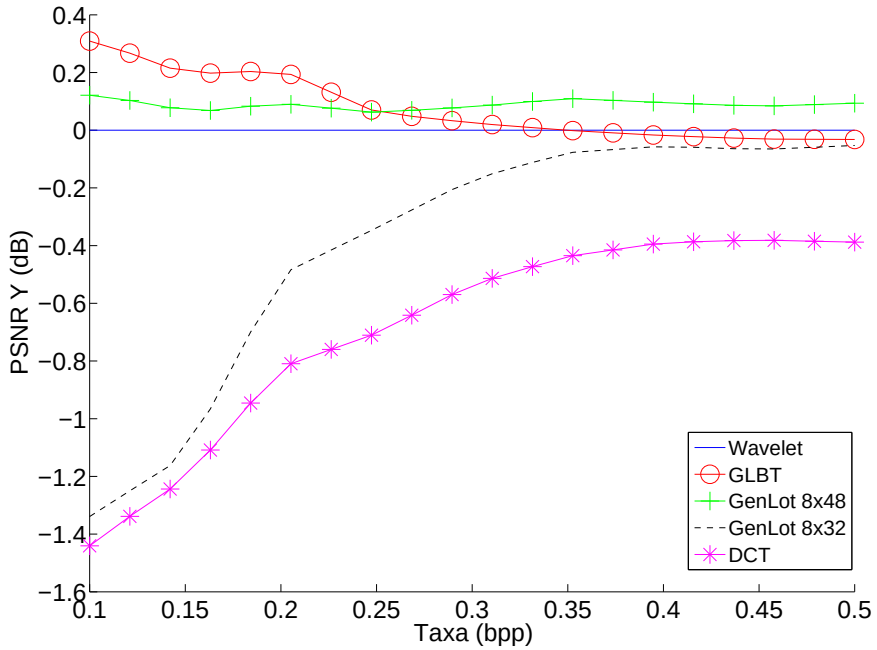


Figura 5.7: Curvas diferencias de PSNR para a imagem “Pedestrian_area”. Resultados apresentados de forma relativa à transformada *Wavelet* dB 9/7. Codificação realizada utilizando JPEG2000.

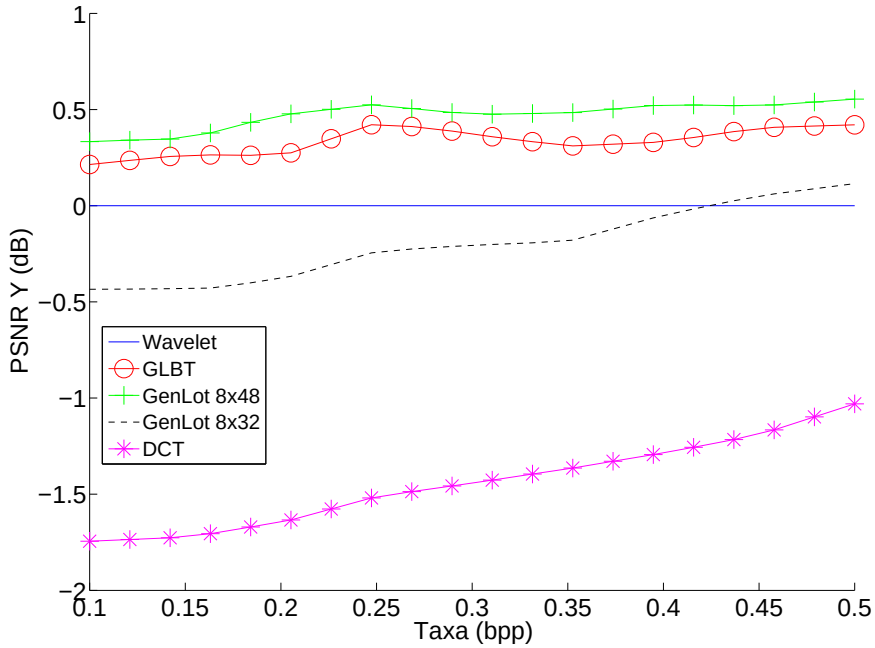


Figura 5.8: Curvas diferencias de PSNR para a imagem “Riverbed”. Resultados apresentados de forma relativa à transformada *Wavelet* dB 9/7. Codificação realizada utilizando JPEG2000.

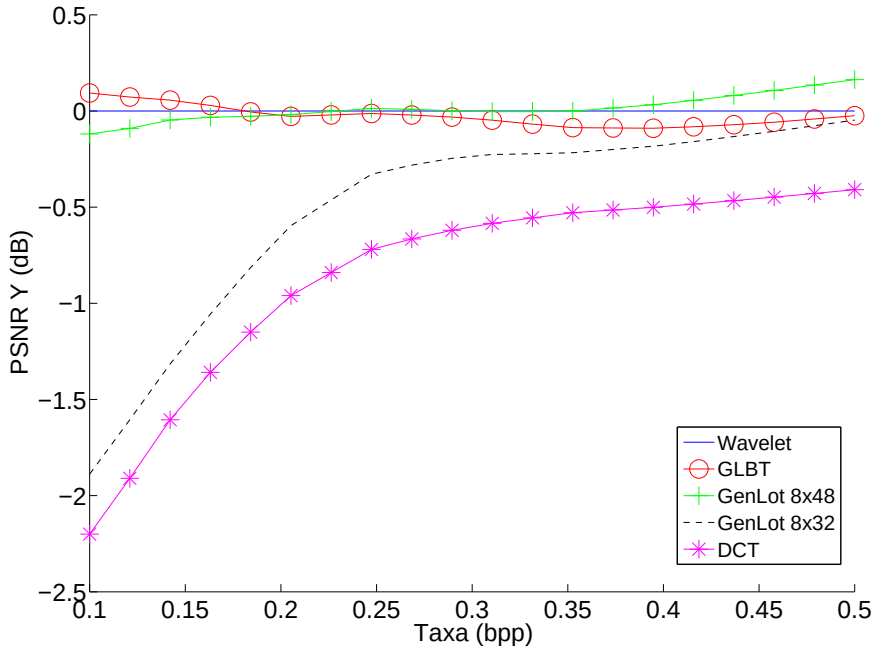


Figura 5.9: Curvas diferencias de PSNR para a imagem ““Rush_hour””. Resultados apresentados de forma relativa à transformada *Wavelet* dB 9/7. Codificação realizada utilizando JPEG2000.

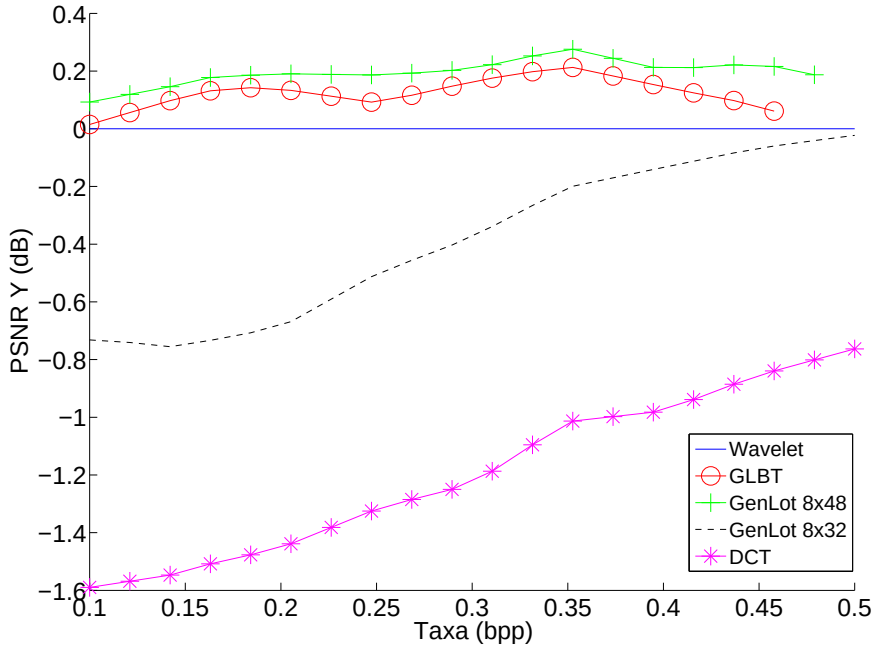


Figura 5.10: Curvas diferencias de PSNR para a imagem “Station2”. Resultados apresentados de forma relativa à transformada *Wavelet* dB 9/7. Codificação realizada utilizando JPEG2000.

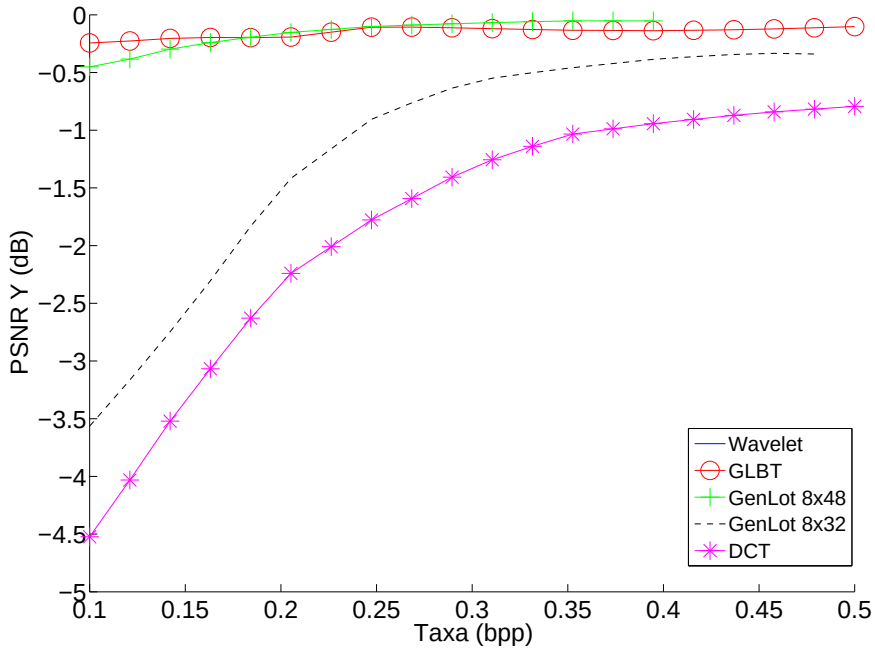


Figura 5.11: Curvas diferencias de PSNR para a imagem “Sunflower”. Resultados apresentados de forma relativa à transformada *Wavelet* dB 9/7. Codificação realizada utilizando JPEG2000.

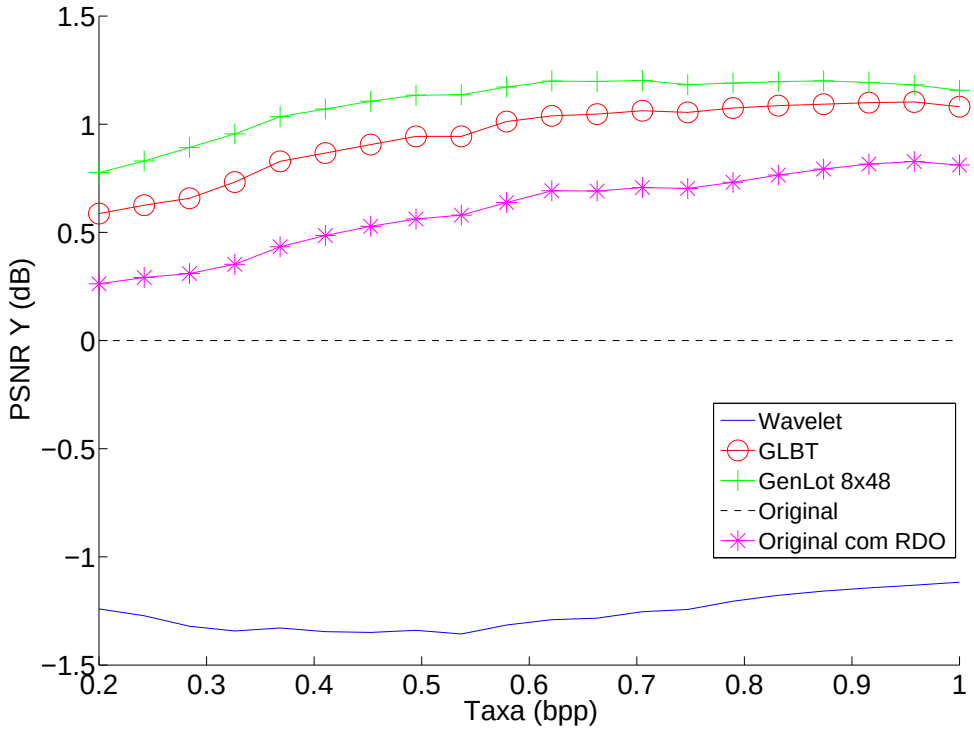


Figura 5.12: Curvas diferencias de PSNR para a imagem “Barbara”. Resultados apresentados de forma relativa à codificação com H.264/AVC original sem otimização taxa-distorção.

5.2 RESULTADOS PARA H.264/AVC

Para os testes com o codificador H.264/AVC, as transformadas DCT e GenLot 8×32 devido ao desempenho apresentado nos testes anteriores. As mesmas imagens foram utilizadas. Os resultados para imagens de 512×512 *pixels* são apresentados nas Figuras 5.12 a 5.15 e os resultados para as imagens de alta definição nas Figuras 5.16 a 5.20

As transformadas sobrepostas utilizadas nos testes, GenLot 8×48 e Glbt 8×16, apresentaram, nestes testes, vantagem mais consistente em relação à *wavelet*. Apenas na imagem “Sunflower”, esta ultima transformada obteve desempenho superior, e isso somente para taxas até 0.7 bpp. Contudo, a comparação mais interessante destes testes é entre as transformadas *wavelets* e sobrepostas com o esquema de predição intra quadros aliada à DCT, utilizado no H.264/AVC. Para as imagens menores, apenas na “Barbara”, as transformadas sobrepostas obtiveram desempenho superior a predição intra quadros. Como já foi mencionado, esse comportamento é atípico. Contudo, como no caso do uso do codificador JPEG2000, as transformadas sobrepostas apresentaram melhor desempenho nas imagens de alta definição. Neste caso, apenas na imagem Pedestrian_area, estas transformadas foram superadas pelo esquema de

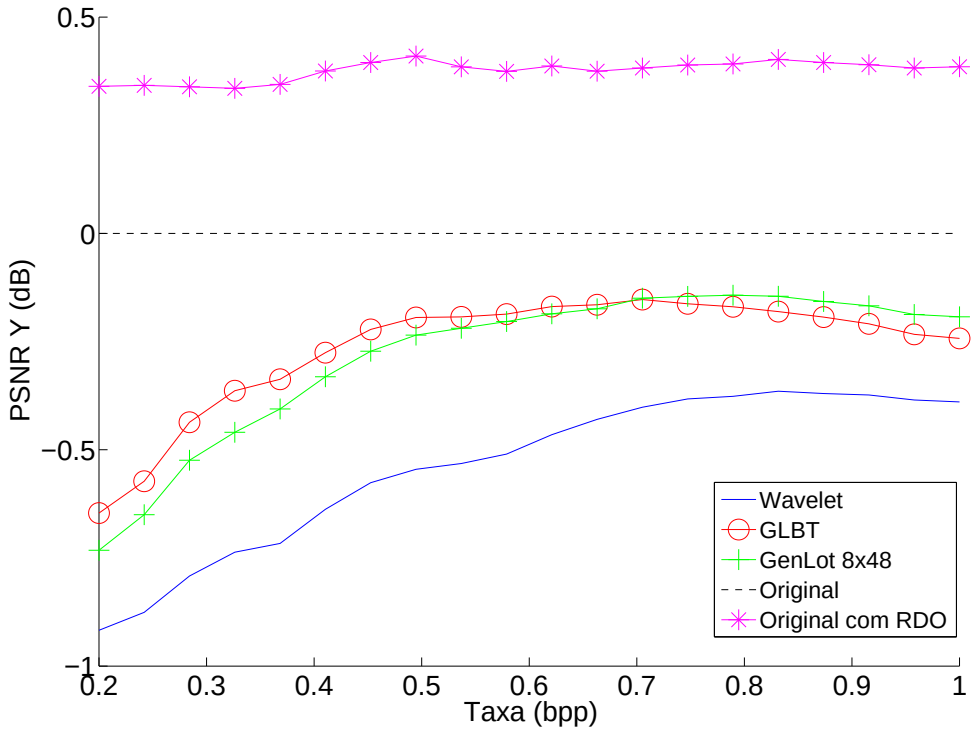


Figura 5.13: Curvas diferencias de PSNR para a imagem “Lena”. Resultados apresentados de forma relativa à codificação com H.264/AVC original sem otimização taxa-distorção.

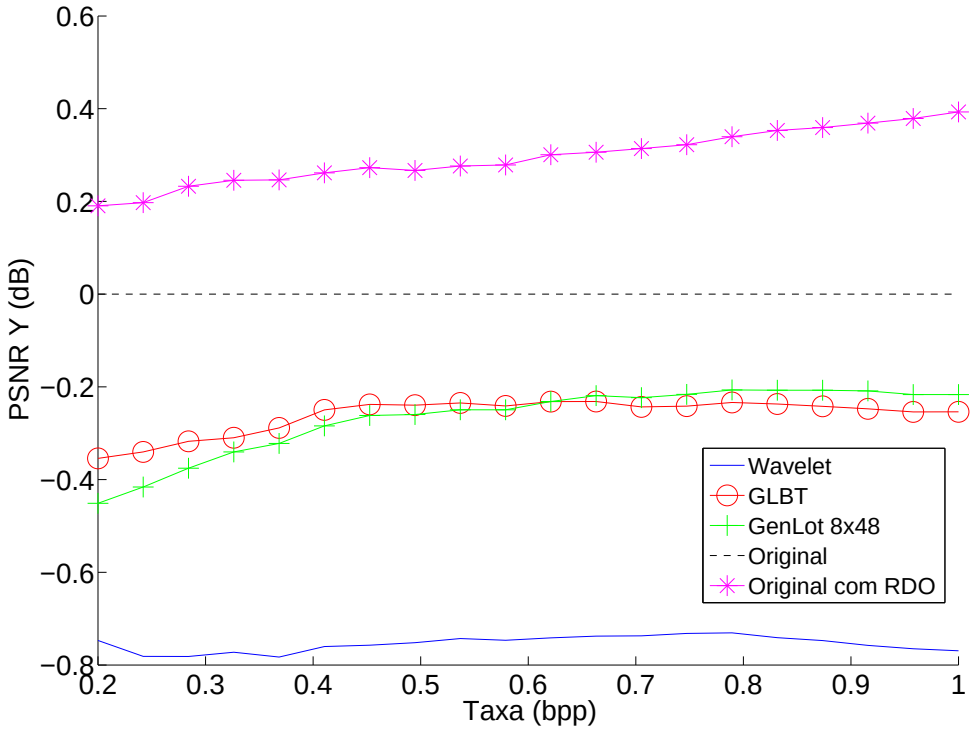


Figura 5.14: Curvas diferencias de PSNR para a imagem “Goldhill”. Resultados apresentados de forma relativa à codificação com H.264/AVC original sem otimização taxa-distorção.

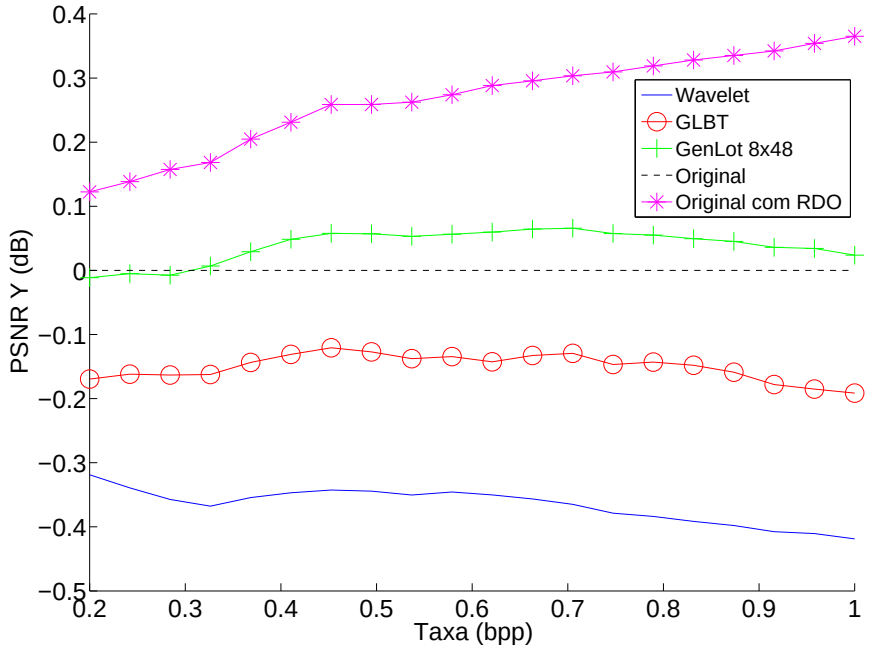


Figura 5.15: Curvas diferencias de PSNR para a imagem “Baboon”. Resultados apresentados de forma relativa à codificação com H.264/AVC original sem otimização taxa-distorção.

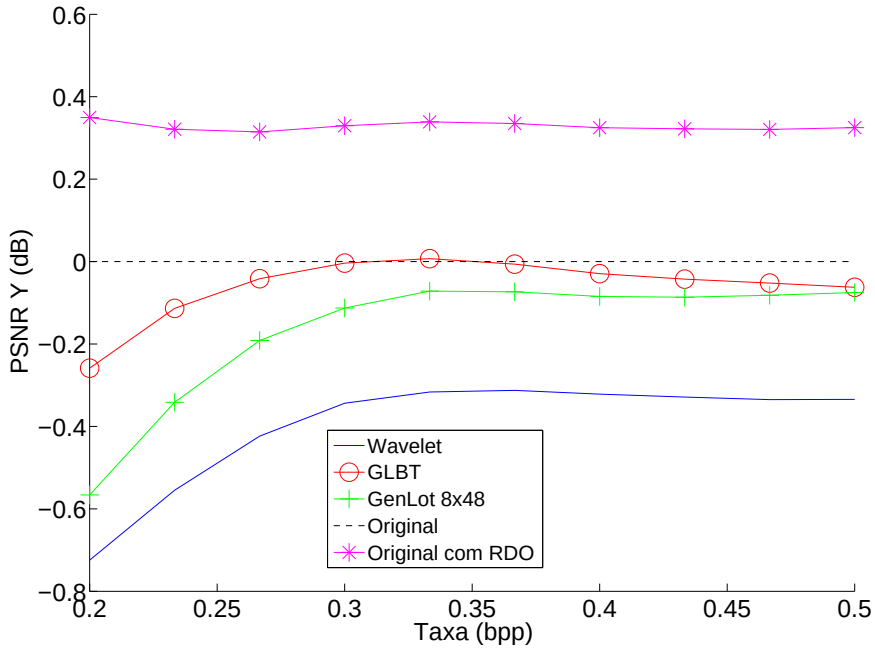


Figura 5.16: Curvas diferencias de PSNR para a imagem “Pedestrian_area”. Resultados apresentados de forma relativa à codificação com H.264/AVC original sem otimização taxa-distorção.

predição utilizado no H.264/AVC sem a utilização de otimização taxa-distorção. Nas imagens, “Riverbed” e “Sunflower”, as transformadas sobrepostas superaram inclusive o H.264/AVC original utilizando este tipo de otimização.

5.3 IMAGENS UTILIZADAS

Nas Figuras 5.21 a 5.29 são apresentadas as figuras utilizadas nos testes.

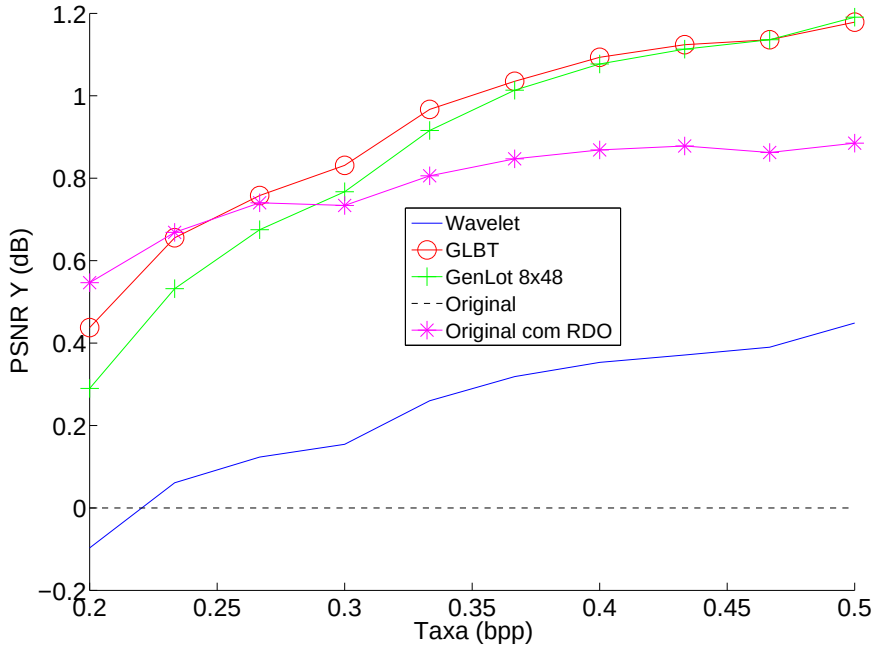


Figura 5.17: Curvas diferencias de PSNR para a imagem “Riverbed”. Resultados apresentados de forma relativa à codificação com H.264/AVC original sem otimização taxa-distorção.

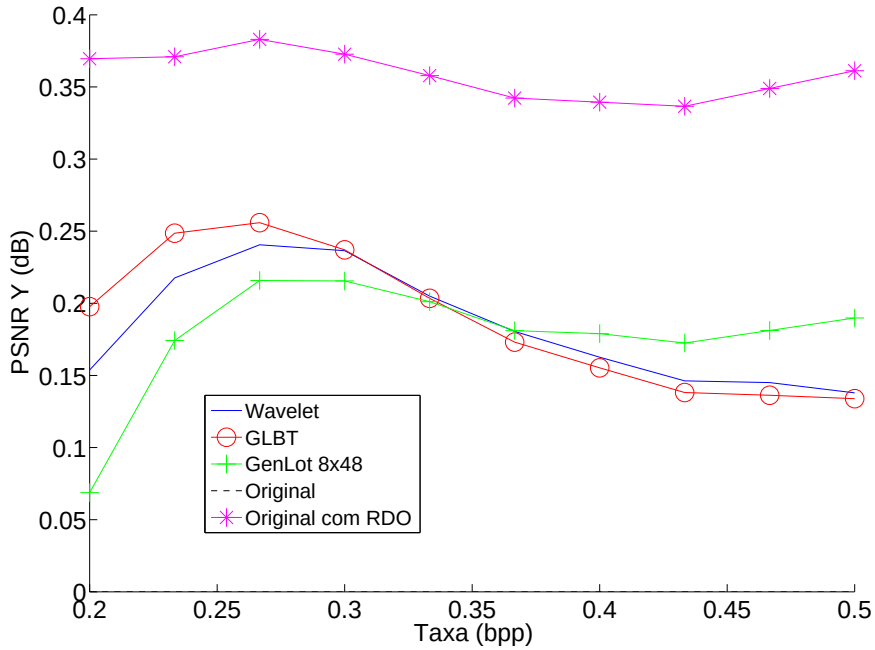


Figura 5.18: Curvas diferencias de PSNR para a imagem “Rush_hour” Resultados apresentados de forma relativa à codificação com H.264/AVC original sem otimização taxa-distorção.

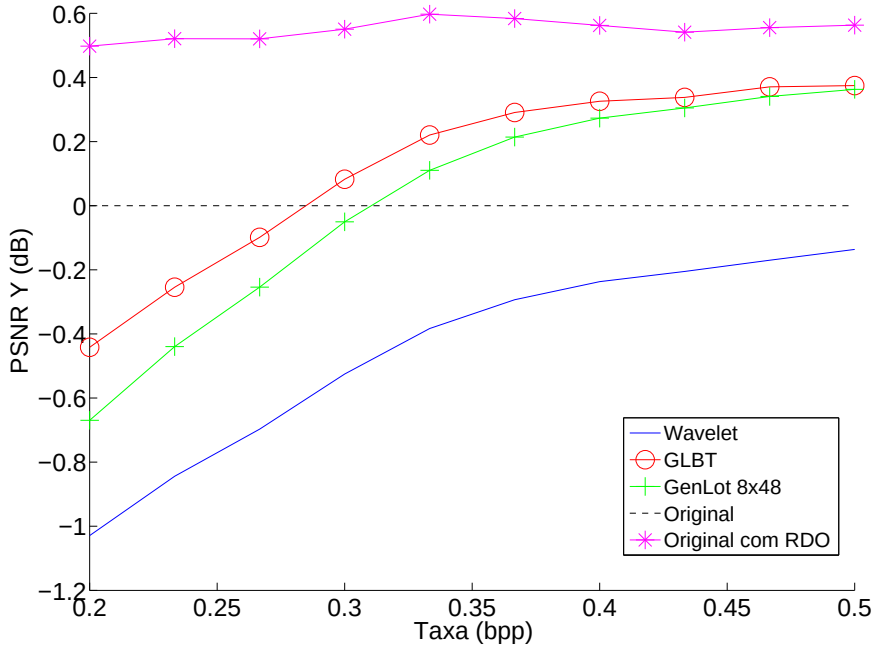


Figura 5.19: Curvas diferencias de PSNR para a imagem “Station2” Resultados apresentados de forma relativa à codificação com H.264/AVC original sem otimização taxa-distorção.

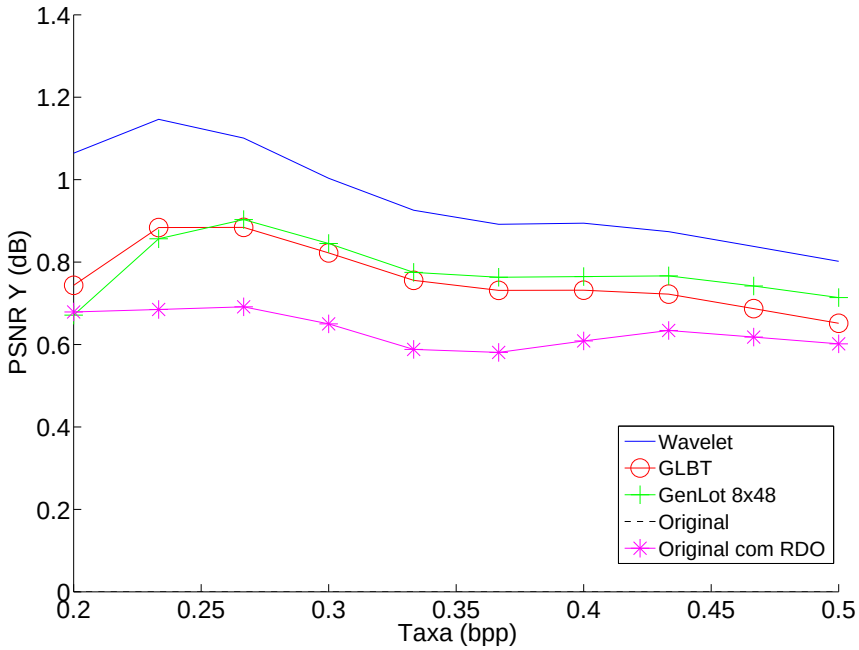


Figura 5.20: Curvas diferencias de PSNR para a imagem “Sunflower” Resultados apresentados de forma relativa à codificação com H.264/AVC original sem otimização taxa-distorção.

Figura 5.21: Imagem “Barbara”.

Figura 5.22: Imagem “Lena”.

Figura 5.23: Imagem “Goldhill”.

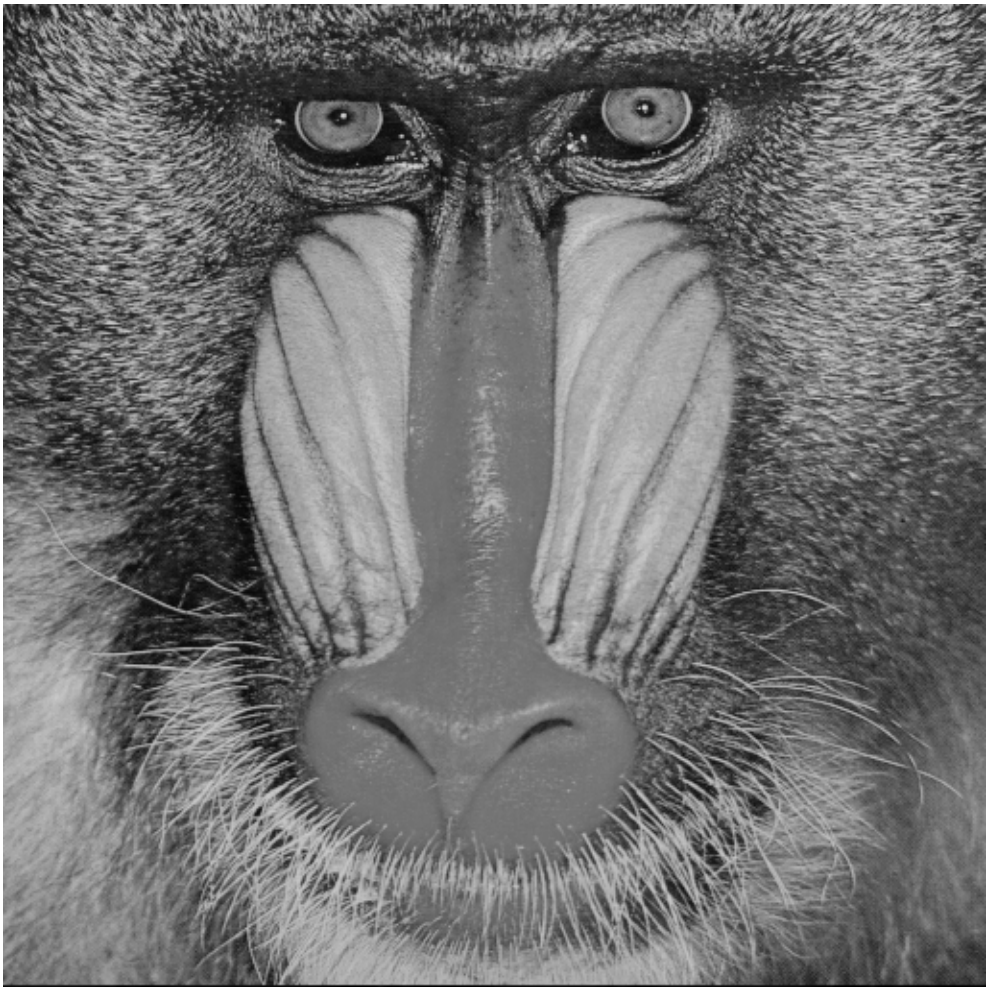


Figura 5.24: Imagem “Baboon”.

Figura 5.25: Imagem “Pedestrian_area”.

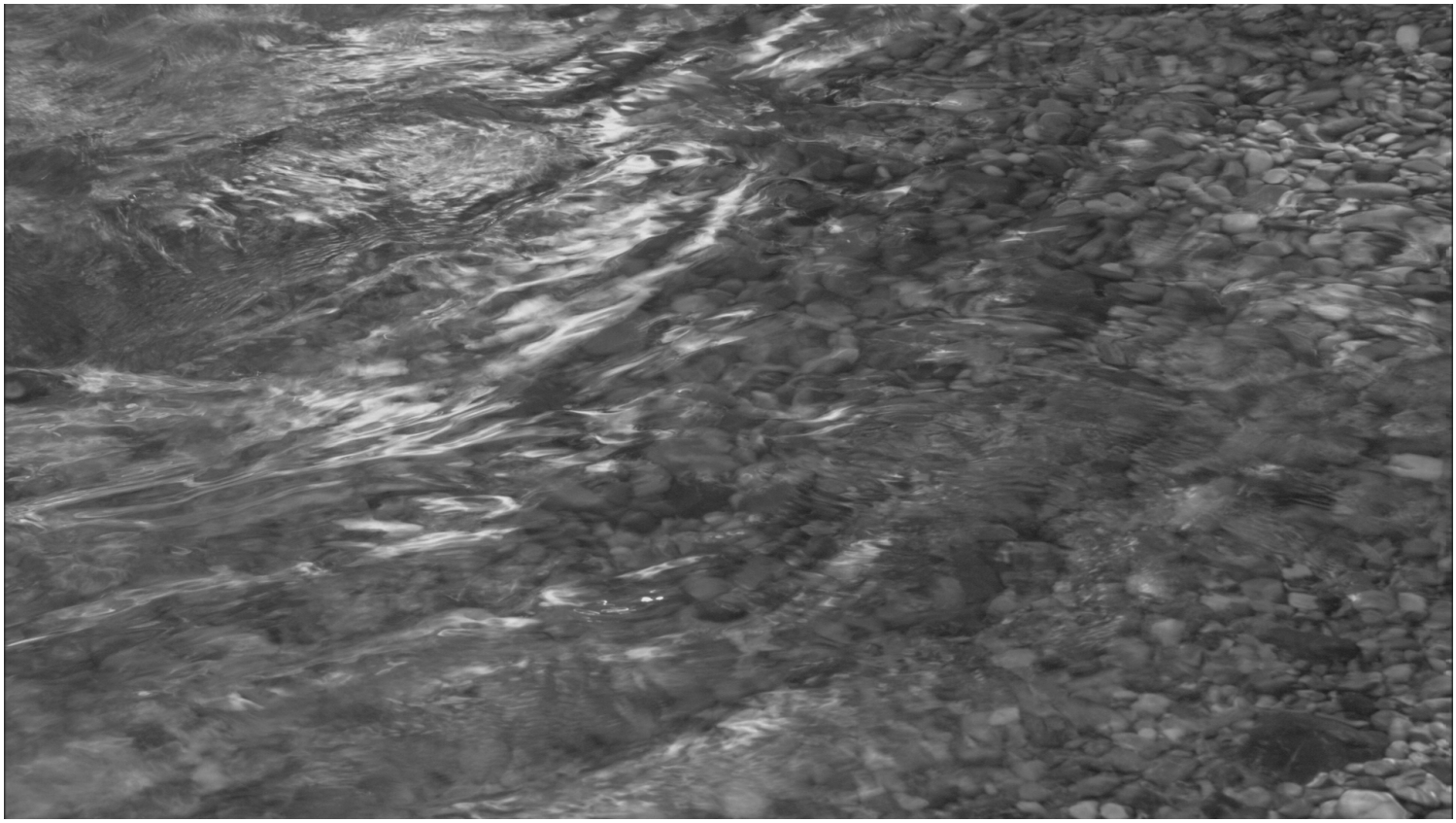


Figura 5.26: Imagem “Riverbed”.

Figura 5.27: Imagem “Rush_hour”.

Figura 5.28: Imagem “Station2”.

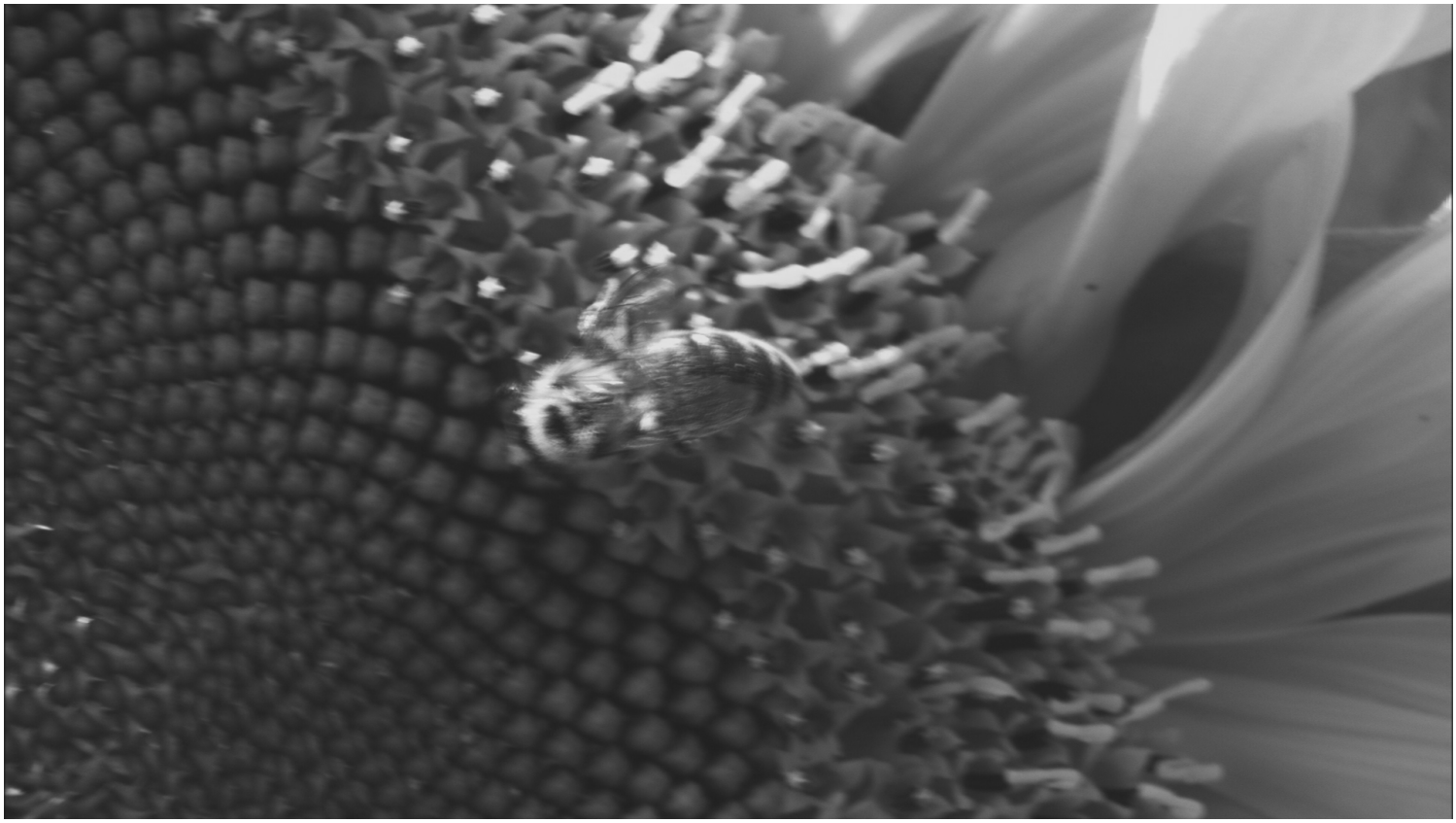


Figura 5.29: Imagem “Sunflower”.

6 CONCLUSÕES

Neste trabalho foram realizados testes comparativos do desempenho de algumas transformadas para a codificação de imagens estáticas, especialmente transformadas sobrepostas e *wavelets*, nos codificadores padrão JPEG2000 e H.264/AVC. Neste ultimo caso, foi feito ainda uma comparação com a predição intra quadro.

Confirmando diversos resultados disponíveis na literatura, as transformadas sobrepostas apresentaram um desempenho superior ao das transformadas *wavelets* e estas se mostraram superiores a DCT. Um fato interessante foi que a vantagem apresentada pelas transformadas sobrepostas foi maior em imagens de alta definição (1920×1080 *pixels*) tanto no H.264/AVC quanto no JPEG2000. Na comparação com a predição intra quadro, realizada apenas no H.264/AVC, as transformadas sobrepostas também se mostraram mais eficiente. Contudo, não tenham superado a versão original do H.264/AVC quando utilizando otimização de taxa e distorção. Novamente, as vantagens se mostraram maiores em alta definição.

Embora esta resolução de 1920×1080 *pixels* seja considerada alta para vídeo, ela possui apenas 2 *Megapixels*. Há câmera digitais disponíveis no mercado que superam 20 *Megapixels*. Sendo assim, é possível que, para imagens estáticas obtidas utilizando este tipo de câmera, a vantagem apresentada nesse trabalho seja ainda maior.

No H.264/AVC, além de um aumento na eficiência de codificação, há uma clara diminuição na complexidade computacional ao utilizar transformadas sobrepostas. Na codificação de um macrobloco de 16×16 *pixels*, na codificação com H.264/AVC original, há a necessidade do teste 4 modos de 16×16 *pixels*, 9 modos em 16 blocos de 4×4 *pixels* e 9 modos em 4 blocos de 8×8 *pixels* contra apenas 4 transformadas de $8 \times 8N$.

O padrão de cinema digital utiliza o Motion JPEG2000, parte 3 do padrão JPEG2000, no qual todos os quadros, que tem uma resolução de 4096×2048, são codificados independentemente, sem compensação de movimentos. A utilização de transformadas sobrepostas poderia tornar a codificação mais eficiente.

6.1 TRABALHOS FUTUROS

Embora a comparação entre as transformadas já esteja consolidada na literatura, muitos trabalhos ainda podem ser realizados.

Inicialmente, pode-se verificar o desempenho na compressão de resíduo de compensação de movimento para compressão de vídeo, o que, eventualmente, aumentaria a eficiência de codificação de vídeo e não somente aplicando a imagens estáticas, como utilizadas neste trabalho.

Outra pesquisa que pode ser realizada é a inclusão de otimização tanto de taxa como distorção na codificação de coeficientes obtidos com a transformadas sobrepostas utilizando o H.264/AVC. Para isso, poderia-se, por exemplo, permitir que se variasse o quantizador e escolher o que minimize uma função de custo.

Por ultimo, os bancos de filtro utilizados foram obtidos utilizando otimização para maximizar o ganho de codificação baseado em estatísticas colhidas em imagens pequenas. Provavelmente, essas estatísticas variem com a resolução da imagem. Outro estudo possível é o levantamento de estatísticas utilizando imagens de maior definição e obtenção de novos bancos de filtro.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] ITU-T Rec. H.264 and ISO/IEC 14496-10 (MPEG-4). *Advanced Video Coding for Generic Audiovisual Services*. [S.l.], Version 1, May 2003; Version 2, January 2004; Version 3, September 2004; Version 4, July 2005.

[2] IMAGE Coding System: Core Coding System (JPEG2000 Part 1). [S.l.], September 2000.

[3] GENERIC Coding of Moving Pictures and Associated Audio (Part 3: MPEG-Audio). [S.l.], Feburary 1997.

[4] QUEIROZ, R. L. de; TRAN, T. D. In: RAO, R.; YIP, P. (Ed.). *The Handbook on Transforms and Data Compression*. [S.l.: s.n.].

[5] MALVAR, H. S. In: *Signal Processing with Lapped Transforms*. [S.l.: s.n.].

[6] ITU-T (formalmente CCITT) and ISO/IEC JTC1. *Digital Compression and Coding of Continuous-Tone Sitll Images*. [S.l.], September 1992.

[7] QUEIROZ C. CHOI, Y. H. R. de; RAO, K. R. Wavelet transforms in a jpeg-like image coder. *IEEE Trans. on Circuits and Systems for Video Technology*, v. 7, n. 1, p. 419–424, April 1997.

[8] SAID, A.; PEARLMAN, W. A. *IEEE Trans on Circuits Syst. Video Tech.*, June.

[9] TRAN T.D.; NGUYEN, T. *Circuits and Systems, 1998. ISCAS '98. Proceedings of the 1998 IEEE International Symposium on*, May.

[10] RAO, K. R.; YIP, P. *Discrete Cosine Transform: Algorithms, Advantages*. [S.l.]: New York: Academic, 1990.

[11] ITU-T and ISO/IEC JTC 1 - ISO/EC 13818-2 (MPEG-2). *Generic coding of moving pictures and associating audio information - Part 2: Video*. [S.l.], November 1994.

[12] ITU-T. *ITU-T Recommendation H.263, Video coding for low bit rate communication*. [S.l.], November 2000.

[13] TAUBMAN, D. S.; MARCELLIN, M. W. *JPEG2000: Image Compression Fundamentals, Standards and Practice*. [S.l.]: Kluwer Academic, 2002.

[14] ACHARYA, T.; TSAI, P. *JPEG2000 Standard for Image Compression: Concepts, Algorithms and VLSI Architectures*. [S.l.]: John Wiley & Sons, 2005.

[15] RICHARDSON, I. E. G. *H.264 and MPEG-4 Video Compression*. [S.l.]: John Wiley & Sons Ltd, 2003.

[16] LUTHRA, A.; SULLIVAN, G. J.; WIEGAND, T. H.264/avc video coding standard. *IEEE Transactions on Circuits and Systems for Video Technologies*, v. 13, n. 7, July 2003.

[17] ISO/IEC JTC/SC29/WG1. *Call for contributions for JPEG 2000 (JTC 1.29.14, 15444): Image coding system. Tecnichal Report N505*. [S.l.], March 1997.

[18] QUEIROZ, R. L. de et al. Fringe benefits of the h.264/avc. In: *International Telecommunication Symposium*. [S.l.: s.n.], 2006. p. 208–212.

[19] MARPE, V. G. D.; WIEGAND, T. Performance evaluation of motion-jpeg2000 in comparison with h.264/avc operated in pure intra coding mode. In: *Proc. SPIE*. [S.l.: s.n.], 2004. p. 129–137.

[20] HUANG, Y.-Y. et al. Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 16, n. 4, p. 507–522, April 2006.

[21] SULLIVAN, G. J.; TOPIWALA, P.; LUTHRA, A. The H.264/AVC Advanced Video Coding Standard: Overview and Introduction to the Fidelity Range Extensions. *Proc. SPIE Conference on Applications of Digital Image Processing XXVII, Special Session on Advances in the New Emerging Standard: H.264/AVC*, August 2004.