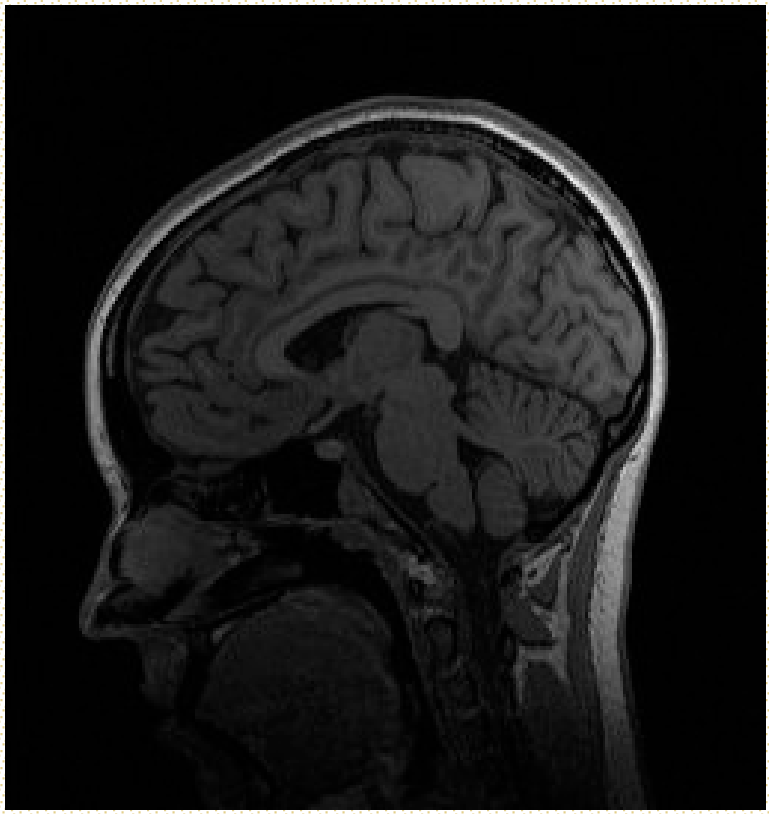
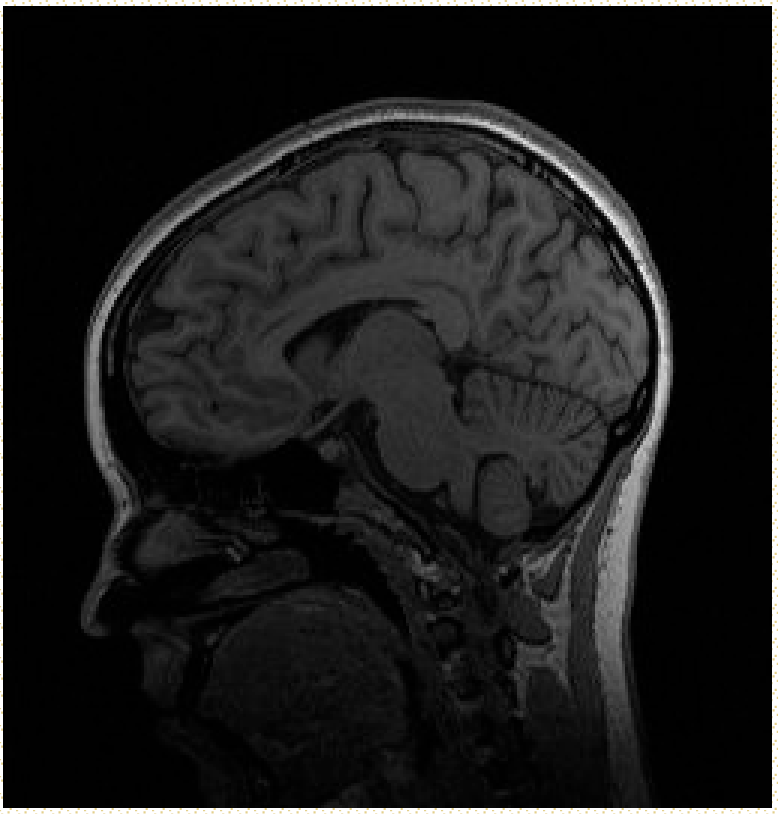
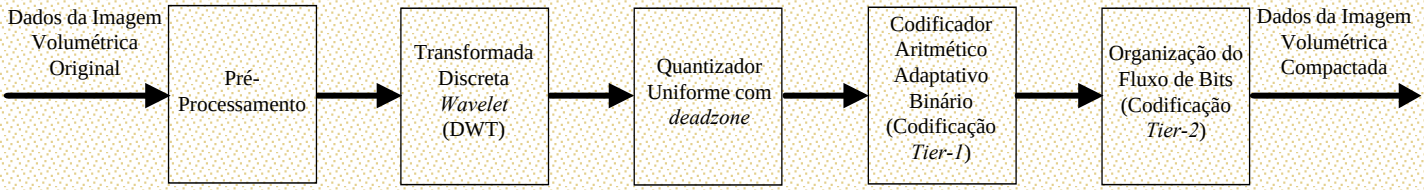






Rui Daniel Ribeiro Peliteiro

Estudo de compressão de dados médicos volumétricos usando codificadores normalizados de imagem

Dissertação de Mestrado em Engenharia Eletrotécnica e de Computadores

Fevereiro/2016

· U C ·

UNIVERSIDADE DE COIMBRA

Estudo de compressão de dados médicos volumétricos usando codificadores normalizados de imagem

Rui Daniel Ribeiro Peliteiro

Dissertação para obtenção do Grau de Mestre em

Engenharia Electrotécnica e de Computadores

Orientador: Doutor Luís Alberto da Silva Cruz

Júri

Presidente:	Doutor Henrique José Almeida da Silva
Orientador:	Doutor Luís Alberto da Silva Cruz
Vogal:	Doutor Vítor Manuel Mendes da Silva
	Doutor Fernando José Pimentel Lopes

Fevereiro de 2016

Agradecimentos

Gostaria de começar por agradecer ao professor Luís Cruz, pela orientação e pelo conhecimento por ele transmitido fazendo com que o desenvolvimento desta dissertação tornasse bastante enriquecedora a nível profissional e pessoal.

Quero também agradecer a Peter Schelkens, professor na Universidade VUB (Vrije Universiteit Brussel), pela disponibilização do *software* IRIS-JP3D.

Agradeço também aos meus pais e irmã pelo constante apoio durante todos estes anos. Sem eles não seria a pessoa que sou hoje.

Agradeço do mesmo modo ao meus amigos e colegas pelo apoio e motivação ao longo dos últimos anos, quer nos bons quer nos maus momentos, permitindo que estes cinco anos e meio da minha vida universitária tenham sido cheios de momentos inesquecíveis.

Muito Obrigado.

Abstract

The increasing use of technology in aiding to medicine, allowed the development of techniques and procedures to create visual representations of the interior of a human body for medical purposes using techniques such as computed tomography (CT), magnetic resonance imaging (MRI), positron emisson tomography (PET), among others. These imaging techniques create very large amounts of data, forming in this way what is know as medical images. Since these images must be stored, it becomes imperative to its compression, since otherwise it becomes rapidly prohibitive storing this data locally or with the use of the cloud. In this sense, it is necessary to use an image encoding system, more specifically the encoder JPEG 2000. The choise fell on this encoder seen that the international standard for medical images and related information (ISO 12052), also known as DICOM, set as default encoder JPEG 2000 Part 1, i.e. the core of the JPEG 2000 encoding system. However, as this dissertation the medical images used are of volumetric type is also of great interest the utilization of Part 10 of this encoder, since this adds the ability to compression of volumetric images. In this sense it is initially presented an overview of the encoder already with the necessary changes in order to be able to handle with volumetric images. However, along the dissertation is given more attention to step two of the encoder. This step defines the use of transforms based on wavelet technology. In this follow-up is presented a detailed description of wavelet transform implemented in JPEG 2000, the discrete wavelet transform (DWT). In the same segment is show in detail an alternative to the latter, defined as directional adaptive discrete wavelet transform (DADWT). Finally, it is performed an analysis on the different possibilities of using the encoder settings for compression of medical images. The settings used range from the choise of lossy compression or lossless compression to the number of decompositions to perform in each slice in three orthogonal directions. With this is done an analysis of the results obtained, allowing to draw conclusions as varied as, if compensates use axial decomposition in the case of medical images and to that type, the gain obtained with the axial decomposition, the optimal number of decompositions in the three directions, the loss of information when using lossy compression.

Keywords

JPEG 2000, JP3D, DWT, Directional Wavelets, Volumetric Medical Images, Error Metrics

Resumo

A crescente utilização de tecnologia no auxílio à medicina, permitiu o desenvolvimento de técnicas e processos para a criação de representações visuais do interior do corpo humano para propósitos clínicos, usando técnicas como a tomografia computorizada (CT), imagem por ressonância magnética (MRI), tomografia por emissão de positrões (PET), entre outros. Estas técnicas imagiológicas geram uma grande quantidade de informação, formando deste modo o que é conhecido como imagens médicas. Uma vez que estas imagens devem ser armazenadas, torna-se imperativo a sua compressão, pois de outro modo torna-se rapidamente incomportável o armazenamento destes dados localmente ou com a utilização da *cloud*. Neste sentido, é necessário a utilização de um sistema de codificação de imagens, mais concretamente o codificador JPEG 2000. A escolha recaiu neste codificador visto que a norma internacional para imagens médicas e informações relacionadas (ISO 12052), também conhecida como DICOM, definiu como codificador padrão o JPEG 2000 Parte 1, ou seja, o sistema de codificação central ao JPEG 2000. No entanto, como nesta dissertação as imagens médicas utilizadas são do tipo volumétrico é também de grande interesse a utilização da Parte 10 deste codificador, já que este adiciona a capacidade de compressão de imagens volumétricas. Neste sentido é inicialmente apresentada uma visão geral do codificador já com as alterações necessárias a fim de conseguir lidar com imagens volumétricas. No entanto, ao longo da dissertação é dada mais atenção à etapa dois do codificador. Esta etapa define a utilização de transformadas baseadas em tecnologia *wavelets*. Neste seguimento é apresentada uma descrição detalhada da transformada *wavelet* implementada no JPEG 2000, a transformada discreta *wavelet* (DWT). No mesmo seguimento é apresentado minuciosamente uma alternativa a este último, definido como transformada discreta *wavelet* direcional adaptativa (DADWT). Por fim, é efetuada uma análise sobre as diferentes possibilidades de configurações de utilização do codificador para compressão de imagens médicas. As configurações utilizadas variam desde a escolha da compressão com perdas ou compressão sem perdas ao número de decomposições a efetuar em cada *slice* em três direções ortogonais. Com isto é feito uma análise dos resultados obtidos permitindo tirar conclusões tão variadas como, se compensa utilizar decomposição axial para o caso das imagens médicas e para que tipo, o ganho obtido com a decomposição axial, o número ótimo de

decomposições a efetuar nas três direções, a perda de informação ao utilizar compressão com perdas.

Palavras-Chave

JPEG 2000, JP3D, DWT, *Wavelets* Direcionais, Imagens Médicas Volumétricas, Métricas de Erro

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

DADWT	Directional adaptive discrete wavelet transform
DCT	Discrete Cosine Transform
DICOM	Digital Imaging Communications in Medicine
DWT	Discrete Wavelet Transform
DWT 1-D	Unidimensional discrete wavelet transform
DWT 2-D	Bidimensional discrete wavelet transform
DWT 3-D	Tridimensional discrete wavelet transform
EBCOT	Embedded block coding with optimized truncation
FIR	Finite impluse response
ICT	Irreversible color transform
IDADWT	Inverse directional adaptive discrete wavelet transform
IDWT	Inverse discrete wavelet transform
IDWT 1-D	Unidimensional inverse discrete wavelet transform
JP3D	JPEG 2000 - Parte 10
JPEG	Joint Photographic Experts Group
LWT	Lazy Wavelet Transform
MSE	Mean Square Error
PSNR	Peak SNR
RCT	Reversible color transform
YCbCr	Luminance; Chroma: Blue; Chroma: Red

1

Introdução

Imagens volumétricas são amplamente utilizadas em aplicações médicas. Estas definem-se como um conjunto de dados de uma imagem tridimensional, e pode ser considerado como uma sequência de imagens bidimensionais, ou *slices*. A gestão de imagens volumétricas digitais define-se pela aplicação de diversos processos, desde a sua representação, processamento e aplicação de técnicas de compressão. Além disso, estas operações são geralmente obrigatórias para a correta interpretação e armazenamento adequado dos dados. No entanto, no que diz respeito à sua compressão ainda não existe nenhuma estrutura normalizada que permita obter um sistema de processamento e armazenamento de dados volumétricos que seja aceite de forma unânime na comunidade científica. Contudo, o comité *The Joint Photographic Experts Group* (ISO/IEC JTC1/SC29/WG1) [4] criou a norma JPEG 2000 bem como a extensão JPEG 2000 - Parte 10 (JP3D), que possibilita a codificação de imagens volumétricas. Similarmente, o comité *Digital Media and Communications in Medicine* (DICOM) [5] está a trabalhar na integração e melhoria de diferentes técnicas essenciais para lidar com imagens volumétricas. É importante citar que o comité DICOM adicionou ao seu *standard* o sistema de codificação JPEG 2000.

Os sistemas de digitalização atuais permitem obter progressivamente imagens médicas volumétricas com melhor qualidade, isto é, imagens tridimensionais com um número crescente de *slices* com uma resolução espacial e profundidade de bit dos píxeis também crescente. Como tal, torna-se crucial uma otimização no processo de compressão e armazenamento destas imagens. Para atingir este objetivo a comunidade internacional optou pela utilização de diversos sistemas normalizados de codificação de imagens. Sendo que para esta dissertação optou-se por escolher o codificador normalizado de imagens, JPEG 2000 - Parte 1 [6] [7] [2], com a extensão tridimensional, JPEG 2000 - Parte 10 [2] [8], visto que este último consiste numa extensão volumétrica do JPEG 2000 - Parte 1, adicionando-lhe suporte para a decomposição *wavelet* [9] ao longo da dimensão axial.

1.1 Objetivos e Principais Contribuições

A investigação desenvolvida ao longo desta dissertação tem em conta a utilização de um sistema normalizado de codificação de imagens, o JPEG 2000 - Parte 10, que tem como finalidade a compressão de dados volumétricos com e sem perdas. Estes dados volumétricos correspondem a imagens médicas volumétricas. Um dos objetivos desta tese consiste num levantamento bibliográfico e descrição teórica dos processos necessários para a correta implementação das transformadas *wavelets*, DWT e DADWT. Um outro objetivo traduz-se na implementação da transformada *wavelet* DADWT e sua inversa no codificador. Outra contribuição consistiu na pesquisa e recolha de imagens médicas volumétricas no formato DICOM. Por fim, fez-se uma análise para conseguir determinar a

melhor configuração dos parâmetros do codificador.

Resumindo, as principais contribuições desta tese, são:

1. Levantamento bibliográfico da DWT e DADWT e respetiva descrição matemática.
2. Recolha de um conjunto de imagens médicas volumétricas com formato DICOM.
3. Procura dos parâmetros ótimos na utilização do codificador em imagens médicas volumétricas.
4. Implementação parcial da transformada baseada na DADWT e sua inversa.

1.2 Estrutura da dissertação

Esta tese é composta por cinco capítulos. No seguimento da Introdução, o capítulo 2 introduz a constituição do codificador de imagens normalizado, JPEG 2000 - Parte 1 e a sua extensão para dados volumétricos, JPEG 2000 Parte 10. O capítulo 3 começa por descrever de uma forma detalhada a transformada discreta *wavelet* não-direcional (DWT) e de seguida é descrito uma alternativa à anterior que corresponde à transformada discreta *wavelet* direcional adaptativa (DADWT) e o seu processo de implementação. Relativamente ao capítulo 4 começa por descrever as métricas de avaliação que serão utilizadas nos resultados. De seguida, apresenta uma descrição do conjunto de imagens médicas volumétricas, bem como uma compilação de resultados obtidos pela utilização do codificador no conjunto de imagens acima referidas. Além disso é feita uma apreciação crítica destes resultados. Finalmente, o capítulo 5 apresenta as principais conclusões do trabalho desenvolvido nesta tese assim como algumas sugestões para trabalho futuro.

2

Codificador normalizado: JPEG 2000

Neste capítulo será explicada de uma forma pormenorizada a constituição do sistema de codificação JPEG 2000 - Parte 10, que tem como base o JPEG 2000 - Parte 1, para codificação de imagens volumétricas.

2.1 JPEG 2000 Parte 10

O comité *Joint Photographic Experts Group* (JPEG) decidiu no ano de 2001 iniciar o desenvolvimento da parte 10 do sistema de codificação JPEG 2000 [2] [8], também mencionado como JP3D. Esta extensão volumétrica para JPEG 2000 oferece um suporte para manipulação de dados tridimensionais com múltiplas componentes. O JP3D é inteiramente baseado em técnicas que também constam no JPEG 2000 Parte 1 [7].

2.1.1 Codificador JP3D

Na figura 2.1 pode-se observar o esquema do codificador JP3D, que é constituído fundamentalmente por cinco blocos. Este esquema define igualmente a arquitetura de codificação definida na Parte 1, no entanto, os blocos diferem internamente. O esquema de blocos descreve os cinco passos seguidos na compressão dos dados da imagem. Estas etapas correspondem ao pré-processamento, transformação por transformada discreta *wavelet* (DWT), quantização, codificação *tier-1*, e codificação *tier-2*.

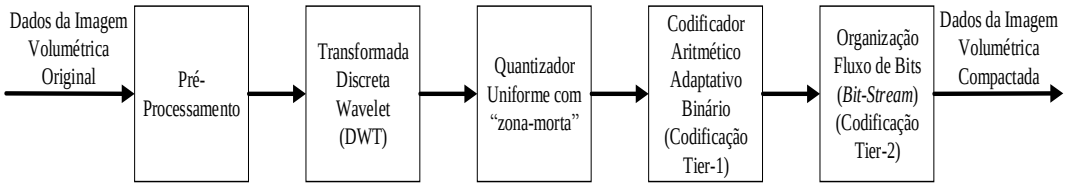


Figura 2.1: Esquema de blocos do codificador JP3D.

Os dados volumétricos de uma imagem que dão entrada no codificador correspondem a um conjunto de imagens bidimensionais, em que cada imagem bidimensional é denominada por *slice*. Na figura 2.2, pode-se observar uma representação esquemática de uma imagem tridimensional formada por um conjunto de *slices* e a orientação das direções X, Y e Z.

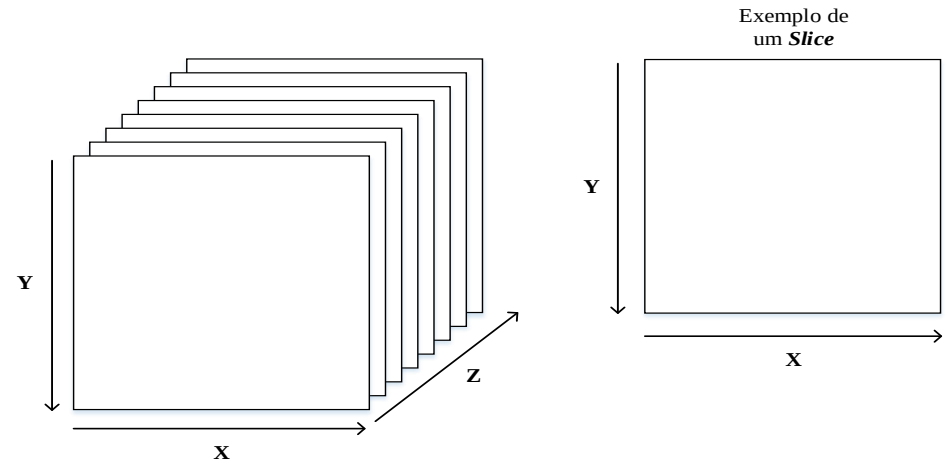


Figura 2.2: Representação esquemática de uma imagem volumétrica que dá entrada no codificador JP3D.

2.1.2 Pré-processamento

A primeira das cinco etapas que compõem o esquema de blocos do codificador JP3D é designado como pré-processamento. Esta etapa tem início com a possibilidade de partição da imagem volumétrica de entrada em sub-cubos de igual tamanho e não sobrepostos como representado na figura 2.3. Este processo é denominado de *tilling* e os sub-cubos são designados por *tiles*. A única situação em que as *tiles* podem ter diferentes tamanhos, são nas *tiles* localizadas nas bordas da imagem. O tamanho da *tile* utilizado nesta dissertação é igual à imagem de entrada do codificador, no entanto o codificador permite a alteração deste tamanho até à situação extrema de uma *tile* ser constituída unicamente por um píxel. O codificador normalizado JP3D é utilizado em cada *tile* de modo independente em relação às restantes *tiles* existentes na imagem. A utilização do mecanismo de *tilling* permite obter um leque de vantagens tal como a redução na memória requerida, visto que a codificação é realizada *tile* em *tile*, ou a possibilidade de descodificação de partes específicas da imagem em vez de toda a imagem, uma vez que as *tiles* são independentemente reconstruídas.

Após o processo de divisão da imagem em *tiles*, é efetuada uma operação designada por *DC-Level shifting*. Este processo é utilizado caso os valores das amostras originais da imagem com profundidade de *bit* B sejam não-negativas, então é adicionado-lhes o valor de -2^{B-1}. Com este deslocamento, as amostras passam a terem uma representação com sinal na gama de valores

$$-2^{B-1} \leq x_{\{i\}} < 2^{B-1} \tag{2.1}$$

Caso os valores das amostras já estejam definidas nesta gama de valores, não é necessário

realizar nenhum ajustamento. Caso a imagem a ser comprimida seja constituída por multi-componentes o processo *DC-Level shifting* é realizado de modo independente em todas as componentes. "A motivação para a aplicação deste processo deve-se ao facto de que quase todas as sub-bandas obtidas pela aplicação da DWT envolvendo filtragem passa-alto têm uma distribuição simétrica relativamente a 0. E sem a aplicação deste deslocamento, a sub-banda LL obtida por filtragem passa-baixo seria a única exceção a esta distribuição simétrica", como é mencionado no livro [9].

Por último, existe a possibilidade de aplicação do processo designado de transformação de componentes de cor. Este processo é opcional e é somente aplicado no caso em que a imagem a ser comprimida seja constituída por três ou mais componentes de cor e estas tenham todas o mesmo tamanho e profundidade de bit. Assumindo que as três primeiras componentes contêm os valores das amostras de vermelho, verde e azul de uma imagem de cor a aplicação desta transformação converte os valores destas componentes para o espaço de cor YCbCr. Esta transformação de cor permite explorar a redundância existente nas três componentes de cor, caso respeitem as condições mencionadas acima.
A aplicação deste processo é conseguido através de duas transformadas de cor. Uma é a transformada de cor irreversível (ICT) que apenas pode ser aplicada em casos que a codificação utiliza o banco de filtros Daubechies (9,7), e é definida matricialmente como

$$\begin{pmatrix} x^{Y}_{\{i\}} \\ x^{C_b}_{\{i\}} \\ x^{C_r}_{\{i\}} \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.168736 & -0.331264 & 0.5 \\ 0.5 & -0.418688 & -0.081312 \end{pmatrix} \times \begin{pmatrix} x^{R}_{\{i\}} \\ x^{G}_{\{i\}} \\ x^{B}_{\{i\}} \end{pmatrix} \tag{2.2}$$

E durante a descompressão, a transformação de cor tem que ser invertida utilizando a relação recíproca

$$\begin{pmatrix} x^{R}_{\{i\}} \\ x^{G}_{\{i\}} \\ x^{B}_{\{i\}} \end{pmatrix} = \begin{pmatrix} 1.0 & 0 & 1.402 \\ 1.0 & -0.344136 & -0.714136 \\ 1.0 & 1.772 & 0 \end{pmatrix} \times \begin{pmatrix} x^{Y}_{\{i\}} \\ x^{C_b}_{\{i\}} \\ x^{C_r}_{\{i\}} \end{pmatrix} \tag{2.3}$$

Ao passo que a outra transformada corresponde à transformação de cor reversível (RCT), em que esta tem a vantagem de ser invertível. Esta transformada de cor pode ser utilizada quando a compressão é realizada com a utilização do banco de filtros Le Gall (5,3), sendo definida matricialmente por

$$x^{\tilde{Y}}_{\{i\}} = \left\lfloor \frac{x^{R}_{\{i\}} + 2x^{G}_{\{i\}} + x^{B}_{\{i\}}}{4} \right\rfloor, \quad x^{D_b}_{\{i\}} = x^{B}_{\{i\}} - x^{G}_{\{i\}}, \quad x^{D_r}_{\{i\}} = x^{R}_{\{i\}} - x^{G}_{\{i\}}, \tag{2.4}$$

enquanto a RCT inversa é

$$x^{G}_{\{i\}} = x^{\tilde{Y}}_{\{i\}} - \left\lfloor \frac{x^{D_b}_{\{i\}} + x^{D_r}_{\{i\}}}{4} \right\rfloor, \quad x^{B}_{\{i\}} = x^{D_b}_{\{i\}} + x^{G}_{\{i\}}, \quad x^{R}_{\{i\}} = x^{D_r}_{\{i\}} + x^{G}_{\{i\}}. \tag{2.5}$$

Nas expressões 2.4 e 2.5 a função *floor* denotada por $\lfloor x \rfloor$, converte o número real x no maior número inteiro que seja menor ou igual a x.

O processo de transformação de cor não é utilizado nesta dissertação, uma vez que as imagens volumétricas utilizadas têm apenas uma componente que indica os tons em escala de cinza (*grayscale*).

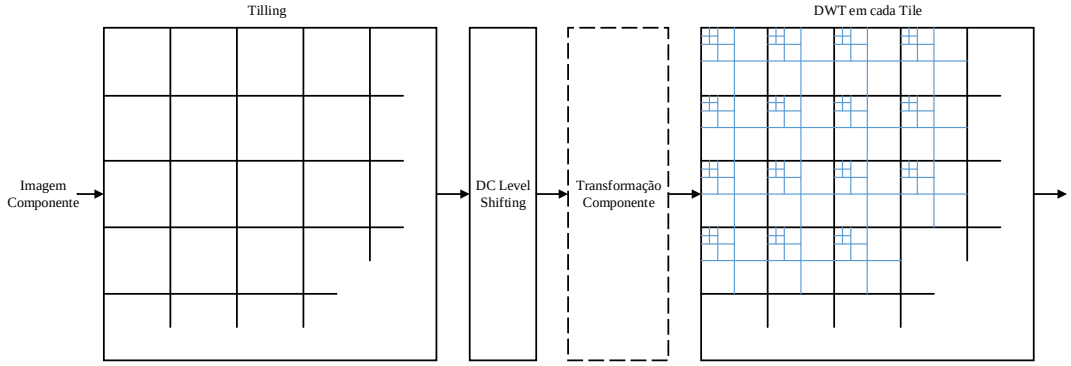


Figura 2.3: Passos existentes no pré-processamento para cada *slice*: *Tilling*, *DC-level shifting*. Seguida da aplição da DWT em cada *Tile*. [1]

2.1.3 Transformada Discreta *Wavelet* (DWT)

No sistema de codificação normalizado JPEG 2000 é utilizada uma transformação de sub-banda, a transformada discreta *wavelet* (DWT). A aplicação desta transformada no JPEG 2000 em detrimento da transformada DCT implementada no JPEG, codificador antecessor ao JPEG 2000, deveu-se ao facto desta última ter diversas desvantagens. Uma das principais justificações para a utilização de transformação de sub-banda, DWT, no lugar de transformação de bloco, DCT, deve-se à presença de artefactos de blocos após compressão quando se utiliza a DCT.

2.1.3.1 Transformada discreta *wavelet* unidimensional

A estrutura que se encontra na figura 2.4 é constituída por um banco de filtros de análise e por um banco de filtros de síntese. O banco de filtros de análise define a transformada discreta *wavelet* unidimensional, DWT 1-D, já o banco de filtros de síntese define a transformada discreta *wavelet* unidimensional inversa, IDWT 1-D.
No codificador a utilização da DWT 1-D num sinal x(n) traduz-se na aplicação sucessiva

de um par de filtros passa-baixo ($h_0(n)$) e passa-alto ($h_1(n)$), seguindo-se de decimação por um fator de 2 (eliminação das amostras com índice ímpar) após cada operação de filtragem. O par de filtros utilizado na codificador é designado por banco de filtros de análise. As amostras filtradas obtidas pela aplicação deste banco de filtros são designadas de coeficientes passa-baixo ou coeficientes *wavelet* passa-alto conforme se aplique os filtros passa-baixo ou passa-alto, respetivamente. Pode-se considerar que a aplicação da DWT 1-D divide o sinal $x(n)$ em duas sub-bandas, a sub-banda L que contém os coeficientes passa-baixo e a sub-banda H que contém os coeficientes *wavelet* passa-alto.
Já no descodificador, a reconstrução do sinal x(n) a partir dos coeficientes é conseguida com a utilização de um outro par de filtros passa-baixo $g_0(n)$ e passa-alto $g_1(n)$, conhecidos como banco de filtros de síntese. O sinal reconstruído é designado por $\hat{x}$(n) e é obtido pela soma das amostras reconstruídas passa-baixo e passa-alto. As amostras passa-baixo são obtidas ao aplicar nos coeficientes passa-baixo uma expansão por fator de 2 (inserção de um zero entre cada duas amostras), seguindo-se de uma filtragem através do filtro passa-baixo g_0(n). Do mesmo modo, as amostras passa-alto reconstruídas são obtidas pela aplicação de uma expansão por fator de 2 seguido da aplicação de um filtro passa-alto g_1(n) nos coeficientes *wavelet* passa-alto.

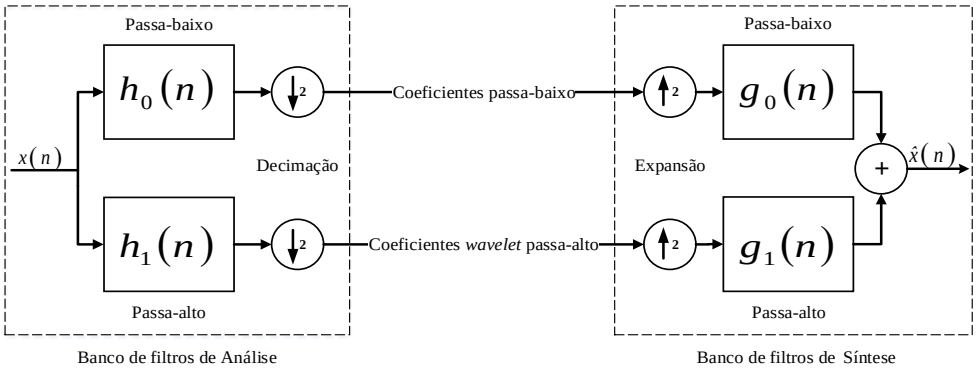


Figura 2.4: Banco de filtros de análise e de síntese, representativo da DWT e IDWT, respetivamente.

Com a finalidade de assegurar que o sinal original $x(n)$ e o sinal reconstruído $\hat{x}(n)$ sejam iguais, é preciso garantir a propriedade de reconstrução perfeita nos filtros de análise e de síntese, isto é, que satisfaçam as duas seguintes condições

$$H_0(z)G_0(z) + H_1(z)G_1(z) = 2, \tag{2.6}$$

$$H_0(-z)G_0(z) + H_1(-z)G_1(z) = 0, \tag{2.7}$$

onde $H_0(z)$, $G_0(n)$, $H_1(z)$ e $G_1(z)$ é a transformada Z de $h_0(n)$, $g_0(n)$, $h_1(n)$ e $g_1(n)$, respetivamente.

O JPEG 2000 especifica duas transformadas *wavelet*, estas são denominadas como o banco de filtros Le Gall (5,3) ou 5x3 [10] e o banco de filtros Daubechies (9,7) ou 9x7 [11]. A designação dos bancos de filtros anteriores como (5,3) e (9,7) deve-se ao facto que os seus filtros de análise são formados por dois filtros FIR passa-baixo de 5 e 9 coeficientes e por dois filtros FIR passa-alto de 3 e 7 coeficientes, respetivamente. Os valores dos coeficientes dos bancos de filtros de análise e síntese da transformada *wavelet*, Daubechies (9,7) encontram-se descritos na tabela 2.1. Enquanto os coeficientes dos filtros de análise e síntese do banco de filtros Le Gall (5,3) encontram-se definidos na tabela 2.2.

Tabela 2.1: Valores dos coeficientes dos filtros de análise e síntese para o banco de filtros Daubechies (9,7)

Coeficientes dos filtros de análise e de síntese de Daubechies (9,7)		
Coeficientes dos filtros de análise		
n	Passa-baixo, $h_0(n)$	Passa-alto, $h_1(n)$
0	+0.6029490182363579	+1.115087052456994
$\pm$ 1	+0.2668641184428723	-0.591271763114250
$\pm$ 2	-0.07822326652898785	-0.05754352622849957
$\pm$ 3	-0.01686411844287495	0.09127176311424948
$\pm$ 4	+0.02674875741080976	
Coeficientes dos filtros de síntese		
n	Passa-baixo, $g_0(n)$	Passa-alto, $g_1(n)$
0	+1.115087052456994	+0.6029490182363579
$\pm$ 1	+0.5912717631142470	-0.2668641184428723
$\pm$ 2	-0.05754352622849957	-0.07822326652898785
$\pm$ 3	-0.09127176311424948	+0.01686411844287495
$\pm$ 4		+0.02674875741080976

Tabela 2.2: Valores dos coeficientes dos filtros de análise e de síntese para o banco de filtros Le Gall (5,3)

Coeficientes dos filtros de análise e de síntese de Le Gall (5,3)				
	Coeficientes dos filtros de análise		**Coeficientes dos filtros de síntese**	
n	Passa-baixo, $h_0(n)$	Passa-alto, $h_1(n)$	Passa-baixo, $g_0(n)$	Passa-alto, $g_1(n)$
0	+3/4	+1	+1	+3/4
$\pm$ 1	+1/4	-1/2	+1/2	-1/4
$\pm$ 2	-1/8			-1/8

2.1.3.2 Transformada Discreta *Wavelet* 2-D (DWT 2-D)

A transformada discreta *wavelet* bidimensional (DWT 2-D) é obtida com a aplicação da DWT 1-D nas direções espaciais X e Y de um modo independente. O esquema que

traduz esta ideia encontra-se exposto na figura 2.6, onde se verifica a aplicação do banco de filtros de análise da DWT 1-D primeiro na direção Y e depois na direção X. A aplicação da DWT 2-D numa imagem bidimensional tem como resultado a sua divisão em quatro sub-bandas, a sub-banda LL, LH, HL e HH, onde L indica a utilização de um filtro passa-baixo e H a de um filtro passa-alto. Esta divisão é conseguida com a convolução vertical ao longo de todas as linhas da imagem utilizando o banco de filtros de análise formado por $h_0(n)$ e $h_1(n)$, obtendo as sub-bandas L e H. De seguida, em cada sub-banda anterior aplica-se novamente o mesmo banco de filtros de análise para permitir a convolução horizontal ao longo das colunas, fazendo com que se obtenha as quatro sub-bandas finais, LL, LH, HL e HH. Este procedimento descreve a decomposição da imagem em um nível, como é visível na figura 2.5. No entanto, caso se queira aplicar novo nível de decomposição a DWT 2-D é apenas aplicada na sub-banda de menor frequência (LL), dividindo-a em quatro novas sub-bandas.

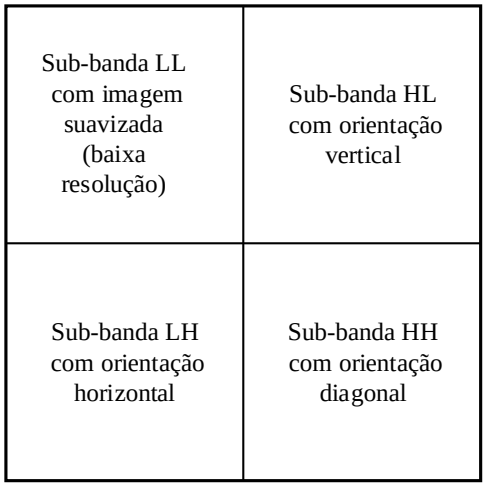


Figura 2.5: Decomposição de um nível de uma imagem bidimensional através da aplicação da DWT 2-D.

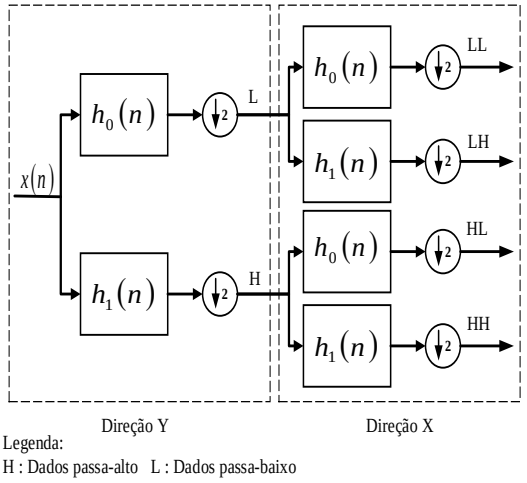


Figura 2.6: Esquema do banco de filtros de análise para DWT 2-D.

Uma vez que a aplicação de um filtro passa-baixo numa imagem permite preservar as baixas frequências e eliminar as altas, obtém-se como resultado uma imagem suavizada relativamente à original. Inversamente, a aplicação de um filtro passa-alto permite preservar as altas e eliminar as baixas frequências, mantendo deste modo a textura da imagem, como é o caso das extremidades entre outros detalhes da imagem. Estes detalhes são necessários para que a imagem reconstruída tenha a melhor qualidade possível. Ao ter em conta a informação acima referida, pode-se constatar que a sub-banda LL corresponde a uma versão suavizada da imagem original, ou seja, com pior resolução. Já a sub-banda

HL contém detalhes orientados verticalmente, pois o filtro passa-alto permite filtrar as regiões suavizadas e os detalhes orientados na direção horizontal. No caso da sub-banda LH o filtro elimina os detalhes orientados verticalmente, ou seja, esta sub-banda contém principalmente detalhes orientados na horizontal. Por último, a sub-banda HH responde principalmente a características da imagem orientadas diagonalmente.

2.1.3.3 Transformada Discreta *Wavelet* 3-D (DWT 3-D)

A transformada discreta *wavelet* tridimensional (DWT 3-D) pode ser considerada como a combinação de três transformadas discretas *wavelet* unidimensionais aplicadas nas direções espaciais X, Y e Z. O esquema que ilustra a DWT 3-D, isto é, a aplicação de um modo independente do banco de filtros de análise definido na DWT 1-D nas três direções, encontra-se na figura 2.7. Da figura, verifica-se que a aplicação da DWT 3-D de um nível em uma imagem volumétrica resulta na sua decomposição em oito sub-bandas, as sub-bandas LLL, LLH, LHL, LHH, HLL, HLH, HHL, HHH. Cada uma destas sub-bandas representa uma versão filtrada e decimada do volume original.

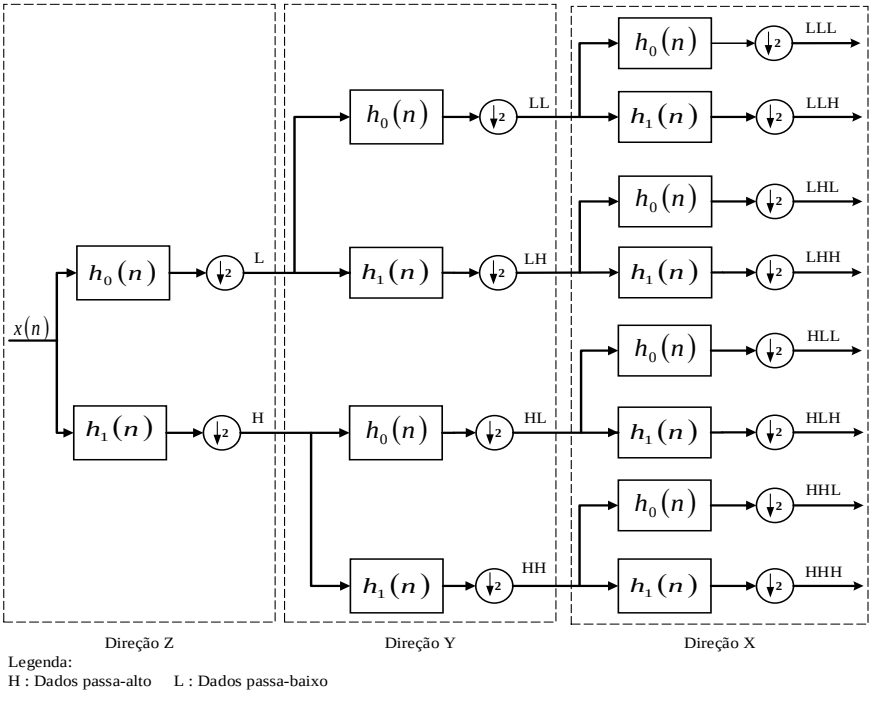


Figura 2.7: Estrutura DWT 3-D de um nível.

A decomposição de uma imagem volumétrica inicia-se com a aplicação independente da DWT 1-D em todas as linhas contidas nas direções espaciais Z, Y e X. As linhas que se encontram orientadas na direção Z têm as coordenadas Y e X em comum. Já as linhas

definidas na direção Y têm como coordenadas em comum X e Z. Enquanto as linhas compreendidas na direção X têm as coordenadas Y e Z em comum.
A decomposição de uma imagem volumétrica tem início com a aplicação do banco de filtros de análise definido na DWT 1-D a cada linha orientada segundo o eixo Z da imagem volumétrica. Com esta primeira filtragem seguida de decimação a imagem volumétrica original é dividida em duas sub-bandas, as sub-bandas L e H como se observa na figura 2.8 de 'a' para 'b'. Neste seguimento, é aplicado novamente a DWT 1-D na direção Y aos dados filtrados e decimados que formam cada uma das sub-bandas obtidas anteriormente, fazendo com que cada sub-banda se divida em duas novas sub-bandas. Totalizando até este momento as quatro sub-bandas LL, LH, HL e HH, como se constata na figura 2.8 de 'b' para 'c'. Por fim, é aplicado pela última vez a DWT 1-D, agora na direção X alcançando deste modo a decomposição final da imagem volumétrica em oito sub-bandas, as sub-bandas LLL, LLH, LHL, LHH, HLL, HLH, HHL e HHH, figura 2.8 de 'c' para 'd'. Porém, a imagem volumétrica pode ser continuamente decomposta, aplicando para isso a DWT 3-D na sub-banda de menor frequência (LLL), até que esta seja demasiada pequena para uma nova decomposição.

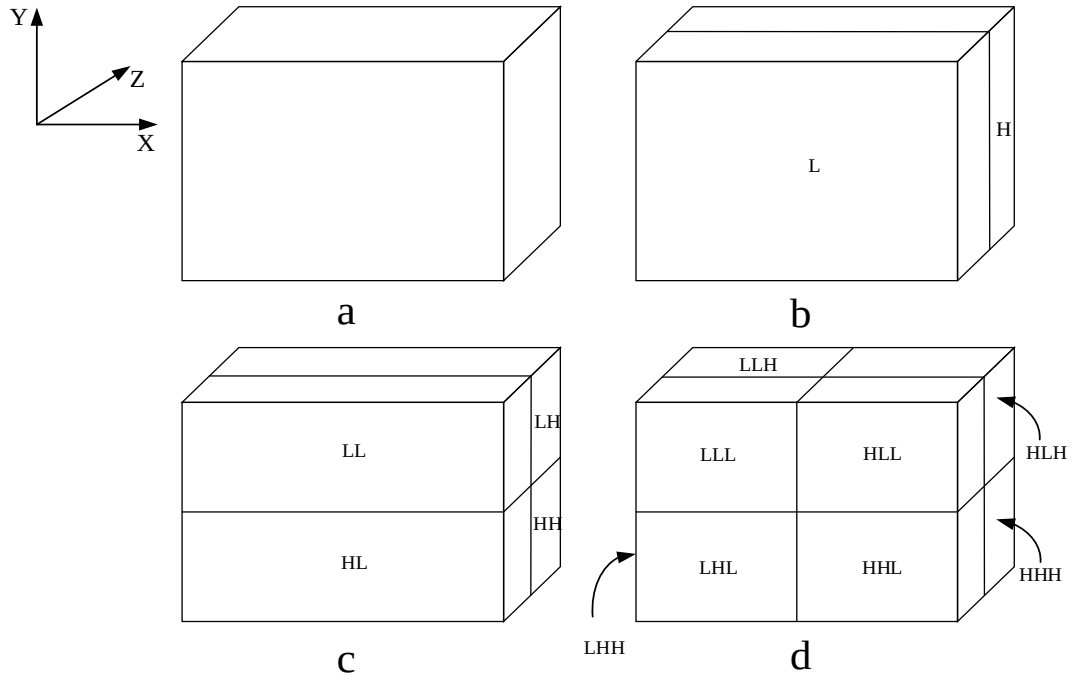


Figura 2.8: Decomposição um nível após aplicação da DWT 3-D.

A transformada discreta *wavelet* tridimensional (DWT 3-D) encontra-se implementada no codificador normalizado JP3D, já que este codificador tem como funcionalidade

a compressão de imagens volumétricas. Porém, este codificador também permite a compressão de imagens bidimensionais. Os bancos de filtros utilizados neste codificador são os mesmos que no JPEG 2000 - Parte 1, ou seja, utilizam os bancos de filtros Daubechies (9,7) e o Le Gall (5,3) para compressão.

2.1.3.4 Implementação da DWT 1-D com base no esquema de *lifting*

O codificador normalizado JPEG 2000 permite a implementação de dois modos de filtragem: o baseado na convolução e o baseado no esquema de *lifting* [12] [13] [14] [15]. Os coeficientes obtidos quer pela utilização do esquema de *lifting* quer pela utilização da convolução através dos bancos de filtros têm valores idênticos. Porém, só a implementação do esquema de *lifting* com base no 5x3 é que permite obter uma transformação reversível. A implementação da estrutura de *lifting* requer uma menor utilização de memória e um menor número de cálculos aritméticos em comparação ao método de convolução.

A estrutura de *lifting* é constituída por um conjunto de etapas. A primeira etapa consiste na divisão da sequência de dados de entrada x_i em duas sub-bandas, a $s_i^{\{0\}}$ e a $d_i^{\{0\}}$, onde a sub-banda $s_i^{\{0\}}$ contém as amostras pares de x_i, enquanto a sub-banda $d_i^{\{0\}}$ contém as amostras ímpares, ou seja $s_i^{\{0\}} = x_{2i}$ e $d_i^{\{0\}} = x_{2i+1}$, respetivamente. Esta transformação é conhecida por transformada *wavelet* "preguiçosa"(LWT). Na etapa seguinte, é aplicado sucessivamente a cada uma das sub-bandas de amostras obtidas anteriormente uma sequência de passos, denominados por passos de *lifting*. Os passos de *lifting* são constituídos pela etapa de predição e atualização. Estas duas etapas acima descritas encontra-se ilustradas através do diagrama de blocos presente na figura 2.9.

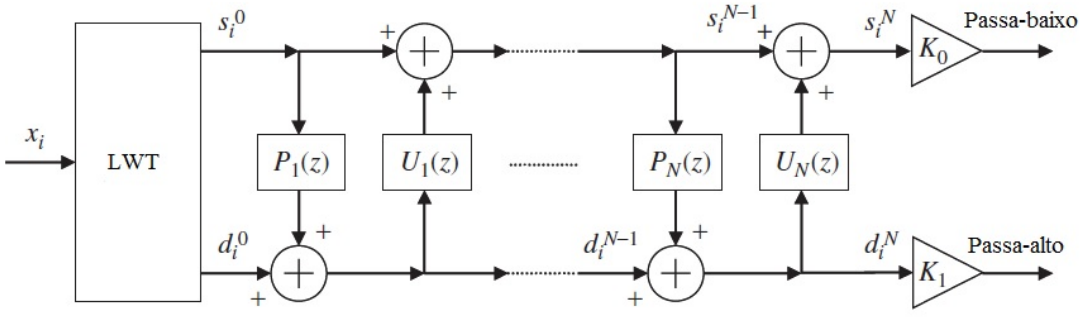


Figura 2.9: Diagrama de blocos do processo de *lifting*. [2]

Os passos de *lifting* são executados N vezes, e o índice 'n' indica o número do passo de *lifting* a ser executado onde $n \in \{1,2,...,N\}$. Para as amostras que se encontram na sub-banda $d_i^{\{n-1\}}$ aplica-se a etapa de predição que consiste na predição de cada amostra com índice ímpar $d_i^{\{n\}}$ através de uma combinação linear com amostras com índice par

$s_{i+k}^{\{n-1\}}$ e somando a este resultado a amostra com índice ímpar $d_i^{\{n-1\}}$, obtendo $d_i^{\{n\}}$.

$$d_i^{\{n\}} = d_i^{\{n-1\}} + \sum_k P_n(k) s_{i+k}^{\{n-1\}}, \quad k \in \{0,1\}, \tag{2.8}$$

Para as amostras com índice par $s_i^{\{n-1\}}$, é efetuada uma etapa de *lifting* conhecida por etapa de atualização. Esta etapa calcula as amostras $s_i^{\{n\}}$ utilizando uma combinação linear com as amostras de índice ímpar $d_{i-k}^{\{n\}}$ e somando logo de seguida a este a amostra com índice par $s_i^{\{n-1\}}$, obtendo $s_i^{\{n\}}$.

$$s_i^{\{n\}} = s_i^{\{n-1\}} + \sum_k U_n(k) d_{i-k}^{\{n\}}, \quad k \in \{0,1\}, \tag{2.9}$$

Após completar todos os 'N' passos de *lifting*, as sub-sequências $d_i^{\{n\}}$ e $s_i^{\{n\}}$ apenas têm que ser multiplicadas pelos fatores K_0 e K_1, a fim de obter a sub-banda passa-baixo $s_i^{\{n\}}$ e sub-banda passa-alto $d_i^{\{n\}}$.

Como qualquer transformada de sub-banda pode ser implementada através da aplicação sucessiva de um determinado número de passos de *lifting* nas sequências de amostras pares e ímpares. Será apresentado o esquema de *lifting* para o banco de filtros Daubechies (9,7) e também para o banco de filtros Le Gall (5,3). A implementação do esquema de *lifting* segundo o banco de filtros Daubechies (9,7) consiste na aplicação de dois passos de *lifting* e pode ser observado na figura 2.10. Enquanto que o valor dos coeficientes de *lifting* das variáveis que constam nas equações 2.8 e 2.9 que definem a etapa de predição e atualização, estão definidas na tabela 2.3.

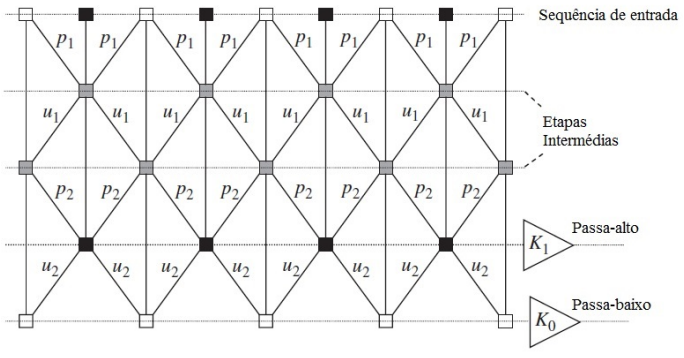


Figura 2.10: Esquema de *lifting* como alternativa ao banco de filtros Daubechies (9,7). [2]

O esquema de *lifting* que permite obter os mesmos coeficientes *wavelet* ao utilizar o banco de filtros Le Gall (5,3) está definido na figura 2.11, e verifica-se que neste esquema só é necessário aplicar um único passo de *lifting*. Os coeficientes de *lifting* designados para este esquema figuram na tabela 2.4.

Tabela 2.3: Valores dos coeficientes de *lifting* para o esquema de *lifting* que retrata o banco de filtros Daubechies (9,7).

N	2
p_1	-1.586134342059924
u_1	-0.052980118572961
p_2	+0.882911075530934
u_2	+0.443506852043971
K_1	+1.230174104914001
$K_0 = 1/K_1$	+0.812893066100000

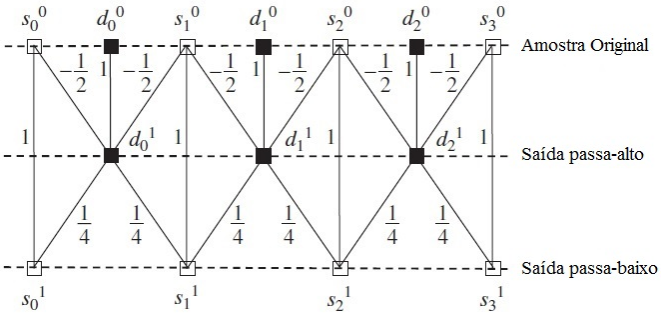


Figura 2.11: Esquema de *lifting* como alternativa ao banco de filtros Le Gall (5,3). [2]

Tabela 2.4: Valores variáveis do esquema de *lifting* com base no banco de filtros Le Gall reversível (5,3).

N	1
K_0	1
K_1	1
p_1	-1/2
u_1	1/4

Como foi enunciado acima, a aplicação do esquema de *lifting* permite garantir a reversibilidade da compressão sem obter perdas, ou seja, ao descomprimir a imagem comprimida obtém-se uma imagem igual à original. Esta reversibilidade é demonstrada em [16], ao indicar que "um esquema de *lifting* que descreve a implementação de uma qualquer transformada de sub-banda de dois canais com filtros FIR é possível sintetizar uma transformada reversível". No entanto, para isto acontecer é necessário eliminar os fatores de ganho de sub-banda K_0 e K_1 e substituir as expressões 2.8 e 2.9 pelas seguintes aproximações não lineares,

$$d_i^{\{n\}} = d_i^{\{n-1\}} + \left\lfloor \sum_k P_n(k) s_{i+k}^{\{n-1\}} \right\rfloor, \quad k \in \{0,1\}, \tag{2.10}$$

$$s_i^{\{n\}} = s_i^{\{n-1\}} + \left\lfloor \frac{1}{2} + \sum_k U_n(k) d_{i-k}^{\{n\}} \right\rfloor, \quad k \in \{0,1\}, \tag{2.11}$$

garantindo deste modo que os valores de $d_i^{\{n\}}$ e de $s_i^{\{n\}}$ sejam inteiros. As alterações introduzidas nestas expressões relativamente às expressões 2.8 e 2.9 consistem na introdução de duas funções *floor* definidas por $-\lfloor -x \rfloor$ e $\lfloor \frac{1}{2} + x \rfloor$, respetivamente. A função *floor* permite o arredondamento ao maior número inteiro que seja menor ou igual a x no primeiro caso e igual a $x + \frac{1}{2}$ no segundo caso. Das transformadas utilizadas com base no esquema de *lifting* no codificador JPEG 2000 apenas a designada por Le Gall (5,3) é uma transformada reversível.

2.1.4 Quantização

A quantização corresponde à terceira etapa no processo de codificação, quer na utilização do codificador JPEG 2000 - Parte 1 ou Parte 10. Este processo é aplicado após a transformada *wavelet* e apenas no caso de compressão com perdas, pois esta operação resulta em perda de informação, a menos que o passo de quantização seja 1 e os coeficientes *wavelet* sejam inteiros, como se verifica caso se utilize a transformada wavelet reversível (5,3). A etapa de quantização tem como dados de entrada os coeficientes *wavelet*, e a aplicação resulta numa redução de precisão destes coeficientes, permitindo deste modo um aumento na compressão. O quantizador utilizado pelo JPEG 2000 - Parte 10 é conhecido como quantizador uniforme escalar com "*deadzone*".

2.1.4.1 Quantização no Codificador

O processo de quantização de cada sub-banda b utiliza o tamanho de passo Δ_b para quantizar todos os coeficientes *wavelet* pertencentes à sub-banda b,isto é, converte os coeficientes com valores reais para o valor inteiro mais próximo. É possível definir um tamanho de passo Δ_b para cada sub-banda. A quantização é dependente do valor de Δ_b, e este pode ser ajustado de forma a atingir uma determinada taxa de bit ou nível de distorção relativamente à imagem a ser codificada.
A operação de quantização é executada no codificador e é implementada de acordo com a expressão 2.12, onde os coeficientes y_b(u,v) pertencentes à sub-banda b são mapeados relativamente a um valor quantizado de $q_b(u, v)$, como se pode observar na figura 2.12.

$$q_b(u,v) = sign(y_b(u,v)) \left\lfloor \frac{|y_b(u,v)|}{\Delta b} \right\rfloor \tag{2.12}$$

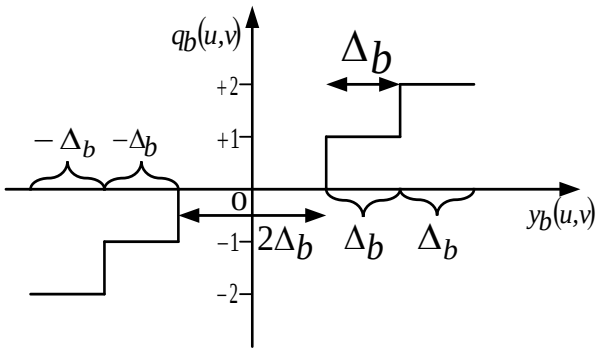


Figura 2.12: Esquema do quantizador escalar com "*deadzone*".

O tamanho de passo Δ_b para cada sub-banda é representado com um total de dois *bytes*, uma mantissa μ_b de 11-bits e um expoente ε_b de 5 bits, de acordo com a expressão

$$\Delta_b = 2^{R_b - \varepsilon b}\left(1 + \frac{\mu_b}{2^{11}}\right),\ onde\ 0 \leq \varepsilon_b < 2^5,\ 0 \leq \mu_b < 2^{11} \tag{2.13}$$

e R_b representa o número máximo de bits que os coeficientes da sub-banda b podem ter.

2.1.4.2 Quantização Inversa no Descodificador

A operação de quantização inversa tem como objetivo tentar reverter da melhor forma possível o processo de quantização aplicado no codificador. No entanto, como este processo não é totalmente invertível existe a possibilidade de perda de informação. A recuperação dos coeficientes *wavelet* quantizados é conseguida através das seguintes expressões

$$\hat{q}_b(u,v) = \begin{cases} (q_b(u,v) + \gamma)\Delta_b,\ se\, q_b(u,v) > 0, \\ (q_b(u,v) - \gamma)\Delta_b,\ se\, q_b(u,v) < 0, \\ 0, \quad outro\ caso, \end{cases} \tag{2.14}$$

onde $0 \leq \gamma < 1$ é um parâmetro de reconstrução escolhido arbitrariamente pelo descodificador. Caso o valor de γ seja de $\frac{1}{2}$, $\gamma = \frac{1}{2}$, isto indica que corresponde a uma reconstrução de ponto médio.

2.1.5 Modelação de Bit e Codificação Entrópica

Uma vez que esta tese centra-se na transformada *wavelet*, ou seja, a etapa dois do codificador normalizado, as etapas relativas à codificação *tier-1* e *tier-2* vão ser explicadas de uma maneira muito sucinta. Pois estas etapas requeriam uma minuciosa explicação a

fim de perceber todo o seu processo, no entanto será dada uma visão geral dos passos aplicados.

Posteriormente à quantização dos coeficientes *wavelet*, ocorre a codificação entrópica destes coeficientes, que permite a criação de um fluxo de bits (*bit-stream*). Este processo de codificação entrópico é conhecido como codificação *tier-1*. O codificador JP3D utiliza uma versão tridimensional do algoritmo EBCOT (*Embedded block coding with optimized truncation*) para a etapa de codificação entrópica. Este procedimento inicia-se com a divisão de cada sub-banda em unidades de menor dimensão chamadas de *code-blocks* como se encontra ilustrado na figura 2.13. Estes *code-blocks* correspondem a um volume tridimensional, onde as suas dimensões são definidas no codificador e podem ser livremente escolhidas, mediante a verificação de certas condições. Estas condições definem que a largura, altura e profundidade dos *code-blocks* sejam números inteiros de potência dois, e que apenas possam conter entre 4 a 2^{18} coeficientes *wavelet* quantizados. Habitualmente os *code-blocks* têm dimensões de $32 \times 32 \times 32$ ou $64 \times 64 \times 64$.

Após cada sub-banda ter sido particionada em *code-blocks*, cada um destes *code-blocks* é independentemente codificado, obtendo no final um fluxo de bits para cada *code-block*. Esta codificação é realizada *bit-plane* a *bit-plane*, começando com o *bit-plane* mais significativo com pelo menos um elemento não negativo até ao *bit-plane* menos significativo. Onde cada *bit-plane* agrupa os bits que estejam na mesma posição de bit de todos os coeficientes *wavelet* quantizados.

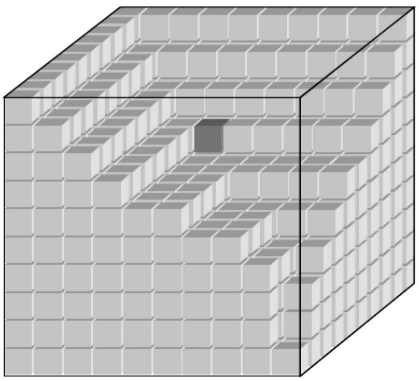


Figura 2.13: Divisão de uma sub-banda em *code-blocks* de igual tamanho. [3]

Para a codificação de cada *bit-plane* é executado três passos de codificação, (1) passo propagação de significância, (2) passo refinamento de magnitude e (3) passo de limpeza. Todos os três tipos de codificação de passos, utilizam o mesmo padrão de pesquisa dos coeficientes nos *code-blocks*, como se observa na figura 2.14. Cada "coeficiente-bit"pertencente a um dado *bit-plane* só é codificado num dos três passos acima referidos.

Este processo de codificação por *bit-plane* gera um conjunto de símbolos binários (informação contextual), em que estes são posteriormente codificados utilizando um codificador aritmético. O codificador adotado é designado codificador aritmético MQ.

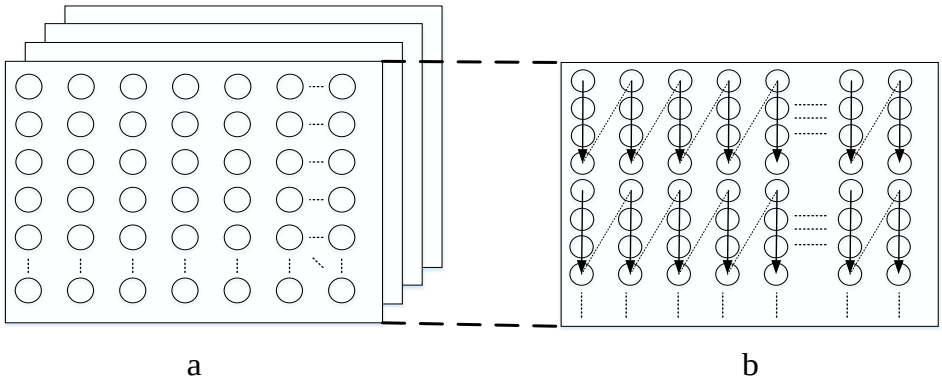


Figura 2.14: Divisão dos coeficientes *wavelet* num esquema de *bit-plane* e padrão de pesquisa efetuado no interior de um *bit-plane*.

A execução de cada passo de codificação relativamente a um dado "coeficiente-bit", baseia-se na "significância"da localização deste bit e dos seus vizinhos. Caso uma localização seja considerada significante será atribuído um "um"para essa localização no *bit-plane* para criação de informação contextual, caso contrário será atribuído o valor um zero.
Durante o primeiro passo de codificação, passo propagação de significância, em cada *bit-plane*, o bit só é codificado caso a sua localização seja não significante, e se pelo menos um dos seus oito bits vizinhos seja significante.

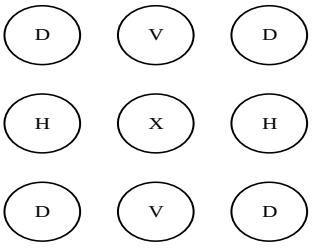


Figura 2.15: Método que verifica a significância de um coeficiente-bit através da sua localização dos oito coeficientes-bit vizinhos.

O segundo passo é designado como passo de refinamento de magnitude, onde são codificados os coeficientes-bits em que o coeficiente-bit anterior mais significativo (mesma posição mas no *bit-plane* acima) se tornara significante.

O terceiro e último passo para todos os *bit-planes* é o "passo de limpeza". Neste passo é codificado os restantes coeficientes-bits que não tenham sido codificados nos dois primeiros passos.

2.1.6 Organização do Fluxo de Bits

A organização do fluxo de bits proveniente da codificação independente dos *code-blocks* corresponde à última etapa do codificador, e permite obter o fluxo de bits final. Esta etapa também é referida como *tier-2*. Este processo além de organizar o fluxo de bits de todos os *code-blocks* permite truncar estes fluxo de bits, possibilitando deste modo obter quer, taxas de bits, limites de distorção ou níveis de qualidade visual pré-definidos. Este processo facilita a criação de uma representação multi-camada do fluxo de bits final, sendo deste modo possível observar por exemplo uma imagem com diferentes níveis de qualidade ou resolução.

Os vários fluxos de bits provenientes dos *code-blocks* codificados na etapa *tier-1* são primeiro agrupados em *precints*. *Precint* pode ser considerado com uma estrutura que basicamente agrupa um conjunto de fluxos de bits de *code-blocks* espacialmente adjacentes de todas as sub-bandas de um nível de resolução em particular. De seguida, cada *precint* é organizado em uma ou mais unidades, chamadas de *pacotes*. Neste seguimento, os pacotes são reorganizados em uma nova estrutura chamada de *camada*, onde em cada *camada* está presente dados compactados de todos os *code-blocks* de todas as sub-bandas. Por fim, o conjunto de todas as *camadas* são organizadas de modo a obter o fluxo de bits final. O processo de agrupamento dos fluxos de bits nas unidades e por fim no fluxo de bit final do codificador encontra-se ilustrado na figura 2.16. Dado à informação anterior, um *pacote* pode ser interpretado como um incremento de qualidade para um nível de resolução em uma determinada região espacial, enquanto uma *camada* pode ser interpretado como um incremento de qualidade na resolução de toda a imagem.

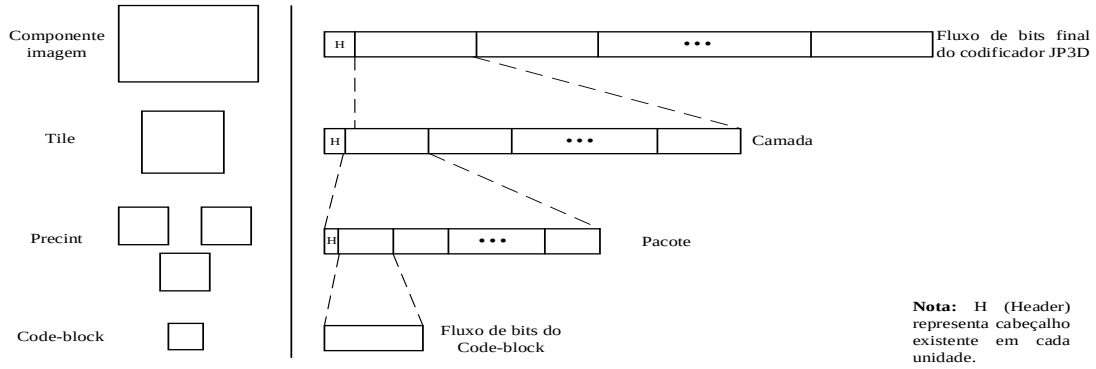


Figura 2.16: Ilustração da correspondência entre as representações espaciais (esquerda) e os respetivos fluxo de bits (direita).

2.1.7 Extensão Simétrica Periódica

Uma imagem volumétrica é formada por um conjunto de imagens bidimensionais também conhecidas por *slices*, e estas têm dimensões finitas. Portanto, na aplicação da transformada *wavelet* nos *slices* é necessário lidar com o problema da existência de extremos em cada imagem bidimensional. Neste sentido, e como a aplicação da transformada *wavelet* numa imagem bidimensional é conseguida com a execução separável da transformada discreta *wavelet* unidimensional, primeiro em todas as linhas e em seguida em todas as colunas. Isto resume-se à sua aplicação num vetor unidimensional que contém amostras de uma determinada linha ou coluna, logo o problema anterior é resolvido com a extensão das fronteiras de cada vetor unidimensional.

Assumindo que $x_{\{i\}}$ corresponde a um vetor unidimensional com amostras compreendidas no intervalo $I \leq i < F$, em que I = 0, e por conseguinte F define o comprimento do vetor. Enquanto $\hat{x}_{\{i\}}$ corresponde ao sinal $x_{\{i\}}$ juntamente com a extensão das amostras nas suas fronteiras, em que,

$$x_{\{i\}} = \hat{x}_{\{i\}},\ I \leq i < F \tag{2.15}$$

A extensão das extremidades do sinal $x_{\{i\}}$ são obtidas com a aplicação de uma extensão simétrica periódica relativamente as amostras existentes entre I e $F-1$. Caso se verifique as seguintes condições, $F-I \geq 2$, $E_{esquerda} < F-I$ e $E_{direita} < F-I$, onde $E_{esquerda}$ e $E_{direita}$ define o número de elementos a serem acrescentados na fronteira esquerda e direita, respetivamente, a extensão simétrica num vetor unidimensional com dimensão finita é definida de acordo com as seguintes expressões

$$\hat{x}_{\{I-i\}} = x_{\{I+i\}},\ \forall i \in \mathbb{Z} \tag{2.16}$$

$$\hat{x}_{\{F-1+i\}} = x_{\{F-1-i\}},\ \forall i \in \mathbb{Z} \tag{2.17}$$

No entanto, se o número de elementos a acrescentar na extremidade esquerda e direita do vetor unidimensional for maior que o número de elementos pertencestes ao próprio vetor, ou seja, $E_{esquerda} \geq F-I$ e $E_{direita} \geq F-I$, as expressões que passam a definir a extensão simétrica periódica são

$$\hat{x}_{\{I-i\}} = x_{\{I+min(mod(-I+i-I,2(F-I-1)),2(F-I-1)-mod(-I+i-I,2(F-I-1)))\}},\ \forall i \in \mathbb{Z} \tag{2.18}$$

$$\hat{x}_{\{F-1+i\}} = x_{\{I+min(mod(F-1+i-I,2(F-I-1)),2(F-I-1)-mod(F-1+i-I,2(F-I-1)))\}},\ \forall i \in \mathbb{Z} \tag{2.19}$$

onde a função $mod(y,x) = z$, devolve o valor z, onde z é tal que $0 \leq z < x$, e $y-z$ é um múltiplo de x. Já a função $min(y,x) = z$ devolve o valor z que corresponde ao menor número inteiro entre y e x.

Por fim, se o número de elementos do vetor unidimensional for inferior a 2, $F-I<2$, não é aplicada nenhuma extensão simétrica e $\hat{x}_{\{i\}}=x_{\{i\}}$ se I for um índice par ou $\hat{x}_{\{i\}}=x_{\{i\}}/2$ se I corresponder a um índice ímpar.
Como exemplo, pode-se observar na figura 2.17 uma extensão simétrica de $E_{esquerda}$ elementos na extremidade esquerda e $E_{direita}$ elementos na extremidade direita em um vetor com dimensão finita. Nesta figura também é possível observar a extensão simétrica periódica caso verifique as condições $E_{esquerda}<F-I$, $E_{direita}<F-I$ ou as condições opostas, $E_{esquerda}\geq F-I$ e $E_{direita}\geq F-I$.

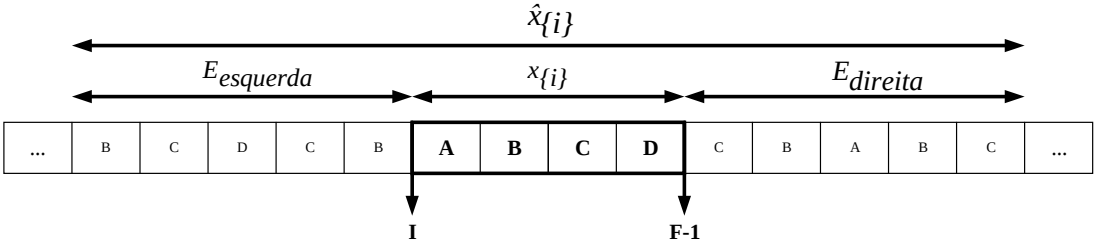


Figura 2.17: Exemplo da aplicação de uma extensão simétrica num vetor unidimensional com dimensão finita para os casos $E_{esquerda}<F-I$ e $E_{direita}<F-I$, bem como para $E_{esquerda}\geq F-I$ e $E_{direita}\geq F-I$.

Tal como é indicado em [6], a tabela 2.5 expressa o número mínimo de elementos necessários a acrescentar nas fronteiras de cada vetor unidimensional, caso a transformada aplicada seja o banco de filtros Le Gall (5,3) ou Daubechies (9,7),

Extensão para a esquerda		
I	$E_{esquerda(5,3)}$	$E_{esquerda(9,7)}$
Par	1	3
Ímpar	2	4

Extensão para a direita		
F	$E_{direita(5,3)}$	$E_{direita(9,7)}$
Par	2	4
Ímpar	1	3

Tabela 2.5: Tabela com o número mínimo de elementos a serem estendidos para a direita e para a esquerda num vetor unidimensional conforme a utilização do banco de filtros (5,3) ou (9,7).

3

Transformada Discreta *Wavelet* Direcional Adaptativa

Uma alternativa à utilização da transformada discreta *wavelet* (DWT) [2] na execução da etapa dois do codificador é a transformada discreta *wavelet* direcional adaptativa (DADWT) [17] [18] [19] [20]. A DADWT permite adaptar localmente as direções de filtragem relativamente ao conteúdo da imagem e a aplicação desta transformada é baseada no esquema de *lifting* utilizado na DWT. Porém, a maior capacidade direcional do esquema de *lifting* utilizado na DADWT permite uma representação mais eficiente para as caraterísticas direcionais da imagem, tais como arestas e linhas.
No decorrer deste capítulo será apresentado em primeiro lugar um esclarecimento do ponto de vista matemático da DWT unidimensional e sua extensão para aplicação em imagens volumétricas, sendo a notação matemática adotada vai de encontra à utilizada nos artigos [17–20]. De seguida, será apresentado as alterações necessárias nas expressões matemáticas que caraterizam a etapa de predição e atualização para permitirem adaptabilidade nas direções de *lifting*. Estas alterações são essenciais na definição da transformada discreta *wavelet* direcional adaptativa. Por fim, será descrito o processo de implementação da DADWT e IDADWT no codificador de imagens normalizado JPEG 2000.

3.1 Transformada Discreta *Wavelet* não-direcional

3.1.1 Transformada Discreta *Wavelet*

Como enunciado na secção 2.1.3, a aplicação da transformada discreta *wavelet* unidimensional numa sequência discreta de dados tem como resultado a sua decomposição em duas novas sub-sequências/sub-bandas, a sub-banda passa-baixo L que contém a informação de baixa frequência e a sub-banda passa-alto H que contém a informação de alta frequência.

Assumindo que $\mathbf{S} = \{s[\mathbf{l}], \mathbf{l} \in \Pi\}$, com $s[\mathbf{l}] = s[l_x, l_y, l_z]$ e $\mathbf{l} = (l_x, l_y, lz)$, representa uma amostra de uma imagem volumétrica definida numa grelha de amostragem ortogonal tridimensional, $\Pi = (l_x, l_y, l_z) \in \mathbb{Z}^3$, e que esta grelha Π é composta por oito sub-grelhas distintas, onde cada uma desta sub-grelhas é matematicamente definida por:

$$\Pi_{pqr} = (l_x, l_y, l_z) \in \Pi \mid p = (l_x \, mod \, 2),\ q = (l_y \, mod \, 2),\ r = (l_z \, mod \, 2), \tag{3.1}$$

Em que $a \, mod \, n$ corresponde à operação de módulo entre dois números positivos, que devolve o resto da divisão de a (dividendo) por n (divisor).

Para aplicar a DWT 1-D com base no esquema de *lifting* é primeiro necessário aplicar a transformada *wavelet* "preguiçosa"que divide todas as linhas existentes nas direções X, Y e Z nas suas componentes pares e ímpares. Estas componentes podem ser reunidas em seis conjuntos de amostras, dois conjuntos por cada direção espacial. Cada um destes seis

conjuntos de amostras com o auxílio das sub-grelhas anteriores podem ser expressas por:

$$\Pi_{\phi}^{H} = \bigcup_{q,r\in\mathbb{B}} \Pi_{\phi qr},\ \Pi_{\phi}^{V} = \bigcup_{p,r\in\mathbb{B}} \Pi_{p\phi r},\ \Pi_{\phi}^{A} = \bigcup_{p,q\in\mathbb{B}} \Pi_{pq\phi},\ |\ \phi\ \in\ \mathbb{B},\ \mathbb{B} = \{0;1\} \tag{3.2}$$

Onde Π_{ϕ}^{H}, Π_{ϕ}^{V} e Π_{ϕ}^{A} define o conjunto de amostras relativamente às direções X, Y e Z. Já ϕ identifica se o conjunto de amostras definido numa dada direção corresponde à componente com as amostras pares caso $\phi = 0$ ou ímpares se $\phi = 1$.
As componentes pares e ímpares da direção D assumindo que D = $\{H;V;A\}$ que constam em Π_{ϕ}^{D}, podem ser expressas por $\mathbf{S}_0^0 = \left\{s\left[\mathbf{l}_0\right]\ |\ \mathbf{l}_0\ \in\ \Pi_0^D\right\}$ e $\mathbf{S}_1^0 = \left\{s\left[\mathbf{l}_1\right]\ |\ \mathbf{l}_1\ \in\ \Pi_1^D\right\}$, respetivamente. De seguida, é aplicado sucessivamente a cada uma destas componentes o passo de predição ou o de atualização, conjuntamente estes passos são definidos como passos de *lifting*. O passo de predição é aplicado às componentes ímpares de **S**, enquanto o passo de atualização é aplicado às componentes pares de **S**. O passo de predição e atualização aplicados a **S** são definidos da seguinte forma:

$$\mathbf{S}_1^{(i)}\left[\mathbf{l}_1\right] = \mathbf{S}_1^{(i-1)}\left[\mathbf{l}_1\right]\ -\ P_{\mathbf{l}_1}^{d,(i)}\left(\mathbf{S}_0^{(i-1)}\right), \forall \mathbf{l}_1\ \in\ \Pi_1^D \tag{3.3}$$

$$\mathbf{S}_0^{(i)}\left[\mathbf{l}_0\right] = \mathbf{S}_0^{(i-1)}\left[\mathbf{l}_0\right]\ +\ U_{\mathbf{l}_0}^{d,(i)}\left(\mathbf{S}_1^{(i)}\right), \forall \mathbf{l}_0\ \in\ \Pi_0^D \tag{3.4}$$

Após a aplicação de M passos de *lifting* nas componentes pares e ímpares e multiplicando posteriormente este resultado pelo fatores G_L e G_H, obtém-se as sub-bandas passa-baixo e passa-alto expressas por:

$$\mathbf{H}_D\left[\mathbf{l}_1\right] = G_H \mathbf{S}_1^{(M)}\left[\mathbf{l}_1\right], \forall \mathbf{l}_1 \in \Pi_1^D \tag{3.5}$$

$$\mathbf{L}_D\left[\mathbf{l}_0\right] = G_L \mathbf{S}_0^{(M)}\left[\mathbf{l}_0\right], \forall \mathbf{l}_0 \in \Pi_0^D \tag{3.6}$$

As funções de predição $P_{\mathbf{l}_1}^{d,(i)}$ e atualização $U_{\mathbf{l}_0}^{d,(i)}$ são indicadas por:

$$\mathbf{P}_{\mathbf{l}_1}^{\mathbf{d},(i)}\left(\mathbf{S}_0^{(i-1)}\right) = \sum_{k=-K_P}^{K_P-1} c_{P,i,k} \mathbf{S}_0^{i-1}\left[\mathbf{l}_1 - (2k+1)\,\mathbf{d}\right] \tag{3.7}$$

$$\mathbf{U}_{\mathbf{l}_0}^{\mathbf{d},(i)}\left(\mathbf{S}_1^{(i)}\right) = \sum_{k=-K_U}^{K_U-1} c_{U,i,k} \mathbf{S}_1^{i}\left[\mathbf{l}_0 - (2k+1)\,\mathbf{d}\right] \tag{3.8}$$

Onde **d** indica a direcionalidade de aplicação do passo de predição e atualização, ou seja, $d = (d_x, d_y, d_z)$ e os seus valores para cada direção espacial são $d_H = (1,0,0)$, $d_V = (0,1,0)$ e $d_A = (0,0,1)$. Relativamente ao valor das variáveis existentes nas expressões 3.5, 3.6, 3.7 e 3.8, são determinados pelo *kernel* a ser utilizado na transformada discreta *wavelet*.

Caso se queira garantir a reversibilidade do esquema de *lifting* pode-se alterar as funções de predição 3.7 e atualização 3.8 com procedimento descrito na secção 2.1.3.4. Onde é aplicado a cada uma das funções anteriores um processo de arredondamente ao número inteiro mais próximo, garantindo que as funções devolvam desta forma unicamente valores inteiros. As funções 3.7 e 3.8 são redefinidas para:

$$\hat{\mathbf{P}}_{\mathbf{l}_1}^{\mathbf{d},(i)}\left(\mathbf{S}_0^{(i-1)}\right) = \left\lfloor \sum_{k=-K_P}^{K_P-1} c_{P,i,k}\mathbf{S}_0^{i-1}\left[\mathbf{l}_1-(2k+1)\,\mathbf{d}\right] \right\rfloor \tag{3.9}$$

$$\hat{\mathbf{U}}_{\mathbf{l}_0}^{\mathbf{d},(i)}\left(\mathbf{S}_1^{(i)}\right) = \left\lfloor \sum_{k=-K_U}^{K_U-1} c_{U,i,k}\mathbf{S}_1^{i}\left[\mathbf{l}_0-(2k+1)\,\mathbf{d}\right] + 0.5 \right\rfloor \tag{3.10}$$

Tal como é indicado na secção 2.1.3.3, a decomposição de uma imagem volumétrica é obtida ao aplicar a DWT 1-D nas direções Z, Y e X. Portanto, o procedimento acima descrito DWT_D 1-D ao ser aplicado a $\mathbf{S}$, primeiro na direção axial, DWT_A 1-D, seguido por DWT_V 1-D na direção vertical e por último DWT_H 1-D na horizontal, decompõem $\mathbf{S}$ em oito sub-bandas.

3.2 Transformada Discreta *Wavelet* Direcional

Uma forma de acrescentar a adaptabilidade das direções de filtragem e consequentemente obter uma melhoria na qualidade de compressão da imagem consiste no uso de transformadas *wavelets* direcionais. Estes factos estão demonstrados nos artigos [18–20]. De modo a permitir a adaptabilidade na escolha das direções de filtragem na transformada discreta *wavelet* unidimensional acima descrita é necessário que as funções de predição e atualização que constam nas expressões 3.7 e 3.8, respetivamente, tenham que aceitar vetores de direção mais genéricos que, $\mathbf{d} = \mathbf{d}_D$ onde D corresponde novamente a A, V ou H. Neste sentido, os novos vetores direcionais são descritos pelas seguintes expressões:

$$\mathbf{l}_1-(2k+1)\,\mathbf{d} \in \Pi_0^D,\ \forall \mathbf{l}_1 \in \Pi_1^D \wedge k \in [-K_P;K_P[\tag{3.11}$$

$$\mathbf{l}_0-(2k+1)\,\mathbf{d} \in \Pi_1^D,\ \forall \mathbf{l}_0 \in \Pi_0^D \wedge k \in [-K_U;K_U[\tag{3.12}$$

A escolha dos vetores direcionais, $\mathbf{d}$, têm que verificar as restrições definidas nas expressões 3.11 e 3.12, estas expressões implicam que as coordenadas dos vetores direcionais tenham que ser números inteiros, $\mathbf{d} \in \mathbb{Z}^3$, e que pelo menos uma dessas coordenadas tenha que ter valor ímpar.

A transformada discreta *wavelet* direcional adaptativa (DADWT) utiliza um conjunto discreto de nove vetores direcionais [17] que respeitam as restrições das expressões 3.11

e 3.12. Nesta transformada optou-se por predefinir um determinado número de vetores direcionais, em detrimento da possibilidade da transformada testar todos os vetores direcionais possíveis de um conjunto bem definido de ângulos de direção com o objetivo de determinar a melhor direção de filtragem, obtendo por conseguinte uma melhor otimização da filtragem às distintas características direcionais que constam na imagem. Esta opção de limitar o número de direções permite diminuir significativamente o número de operações computacionais bem como o tempo necessário para a seleção da direção. O conjunto de nove vetores direcionais utilizados em DWT$_H$-1D, DWT$_V$-1D e DWT$_A$-1D encontram-se descritos na figura 3.1. Observando esta figura verifica-se que no plano XY os vetores direcionais contíguos têm um intervalo de ângulo entre si na ordem dos 22.5° graus, enquanto que para as direções envolvendo a dimensão Z, estes estão desfasados entre si na ordem dos 45° graus.

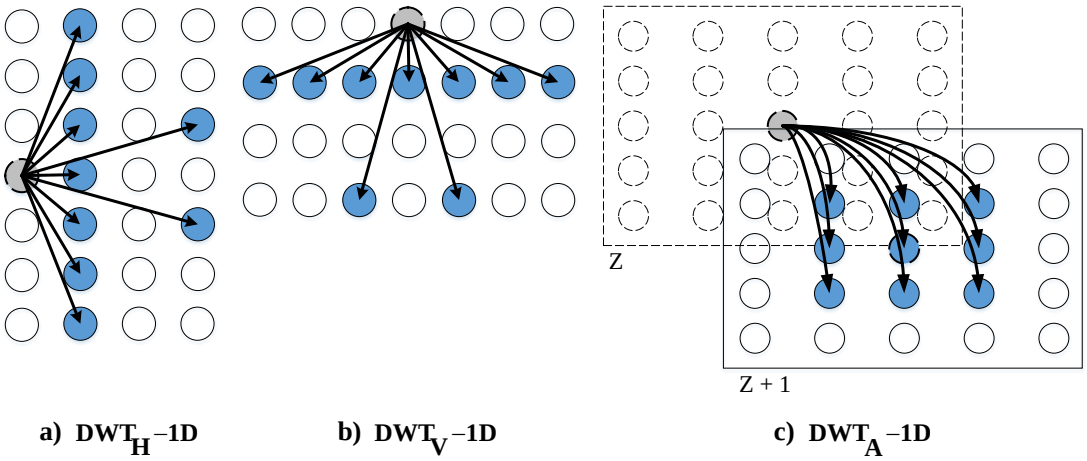


Figura 3.1: Vetores direcionais utilizados em DWT$_D$-1D

Visto que é praticamente impossível permitir a seleção de direção para aplicação da transformada por píxel, é necessário fazer uma segmentação da imagem, isto é, aplica-se a DWT direcional com uma determinada direção num dado conjunto de dados. Com a segmentação dos dados da imagem é possível obter um compromisso entre a escolha da direção e a sobrecarga computacional da escolha da mesma.

Na transformada discreta *wavelet* direcional adaptativa a segmentação utilizada corresponde à segmentação por blocos, isto é, os dados de entrada da transformada (um *slice*) são segmentados em blocos de tamanho fixo N_X * N_Y * N_Z, estes blocos denominam-se *blocos-DA*. Para cada *bloco-DA* é selecionado um vetor direcional.

A seleção de cada vetor direcional em cada *bloco-DA* é baseada na minimização de uma

função de custo lagrangiana definida por:

$$\mathbf{d}_b^D = arg\,min_{\mathbf{d}} \left\{ \sum_{\mathbf{l}_1 \in \Pi_1 \cap \mathbf{B}_b} D\left(S[\mathbf{l}_1] - P_{\mathbf{l}_1}^{\mathbf{d}}(\mathbf{S}_0)/G\right) + \lambda R_b^{\mathbf{d}} \right\} \tag{3.13}$$

onde $D(\cdot) = |\cdot|$ é a medida de distorção, $\lambda > 0$ é um multiplicador lagrangiano, $R_b^{\mathbf{d}}$ corresponde ao número de bits utilizados na sobrecarga para selecionar o vetor direcional $\mathbf{d}_b^D = d$ e $\mathbf{B}_b$ identifica um determinado *bloco-DA*. Na prática, esta função têm como objetivo dar preferência aos vetores direcionais que minimizem a soma dos valores absolutos dos coeficientes de predição calculados. Pois, "a minimização destes coeficientes tende a minimizar as contribuições de taxa das sub-bandas passa-alto após quantização e codificação entrópica" [19]. Notar que, embora a direção é selecionada por blocos, a filtragem no passo de predição e atualização pode, em certos casos, ser realizada para além dos (seus) limites/fronteiras dos blocos.

A codificação do vetor direcional não é realizada nesta tese, uma vez que este é realizado aquando da codificação aritmética. Sendo que o estudo e entendimento da codificação aritmética fica fora do âmbito de trabalho realizado ao longo desta dissertação. Portanto, optou-se por guardar os vetores direcionais escolhidos para cada *bloco-DA* ao longo da imagem em ficheiros textos que serão acedidos caso se queira efetuar a descodificação do fluxo de bits final, para obter a reconstrução da imagem.

3.2.1 Implementação Prática da DADWT

Após a apresentação teórica da DADWT, é necessário elucidar a sua implementação do ponto de vista prático. Neste sentido, pode-se encontrar no algoritmo 1 uma descrição do procedimento utilizado na execução da DADWT numa imagem volumétrica. Similarmente também será ilustrado o seu processo inverso, a IDADWT, que se encontra caracterizada no algoritmo 2.

A transformada discreta *wavelet* com direção adaptativa é executada na etapa dois do codificador JPEG 2000 - Parte 10, como alternativa à DWT tradicional. Uma vez que na DADWT optou-se por adotar a pesquisa da melhor direção através de blocos, também referidos como blocos-DA, definiu-se que o tamanho do bloco-DA seja de $32\times32\times1$. Para cada bloco-DA, a direção ótima é selecionada a partir de um conjunto de nove vetores direcionais que se encontram definidos em cada direção espacial na figura 3.1. A escolha de entre estes nove vetores é baseada na minimização de uma função de custo lagrangiana, a equação 3.13. Nesta equação o valor numérico do multiplicador lagrangiano, λ, é 0.5. Este multiplicador define se na escolha do melhor vetor direcional é tido mais em conta o número de bits utilizados na seleção da direção ou o cálculo do valor da distorção. Quanto mais elevado o seu valor, maior a relevância dada ao primeiro caso.

Por fim, é necessário referir que os vetores direcionais escolhidos para cada bloco-DA serão necessários no descodificador. Assim sendo, estes valores são colocados em ficheiros de texto, que serão lidos aquando da aplicação da transformada inversa. Caso isso não aconteça é impossível realizar a descodificação da imagem.

O procedimento descrito no algoritmo 1 encontra-se implementado na função principal *dadwt_forward_transform()*. Verifica-se que neste algoritmo ocorre a aplicação de quatro procedimentos em cada um dos *slices* que constituem a imagem de entrada do codificador. A cada um destes quatro procedimentos será associado o nome da função que executa essa tarefa. O algoritmo inicia-se com a execução do procedimento {1} que corresponde à aplicação da extensão simétrica nas fronteiras de um *slice* e está implementada na função *dadwt_full_point_symmetric_extend()*. O procedimento {2} diz respeito à escolha da melhor direção em cada bloco-DA, enquanto o procedimento {3} guarda essa escolha num ficheiro texto. Estas duas etapas estão implementadas na função *find_and_save_best_direction_and_apply_transform()*. Por fim, o procedimento {4} corresponde à aplicação da transformada direcional adaptativa em cada *slice* de acordo com o vetor direcional escolhido no procedimento {2}, é executado pela função com o nome *dadwt_one_dimensional_forward_transform_int()*. Todas estas funções encontram-se implementadas no ficheiro *dadwt.c*.

Algorithm 1 Algoritmo DADWT

```
1:  procedure DADWT(Direction, Level, LevelMax)
2:      Direction ← Z, Y ou X
3:      Level ← LevelZ, LevelY ou LevelX
4:      LevelMax ← MAX(LevelZ, LevelY, LevelX)
5:      i ← 0
6:      for all Tiles do
7:          while i < LevelMax do
8:              for all Direction do
9:                  if Level < LevelMax then
10:                     for all Slice do
11:                         • Extensão simétrica nas fronteiras de cada slice. {1}
12:                         • Procura da melhor direção em cada Bloco-DA. {2}
13:                         • Guardar o vetor direcional escolhido para cada
14:                         bloco-DA existente num slice. {3}
15:                         • Aplicar transformada de acordo com vetor direcional
16:                         escolhido para cada bloco-DA dentro de um slice. {4}
17:                     end for
18:                 end if
19:             end for
20:             i ← i + 1
21:         end while
22:     end for
23: end procedure
```

A sequência de etapas que se encontram especificadas no algoritmo 2 definem a aplicação da transformada discreta *wavelet* direcional adaptativa inversa (IDADWT) numa

imagem volumétrica. A implementação da transformada inversa realiza-se na função principal *dadwt_inverse_transform()*. As três funções que integram a função principal são aplicadas em cada um dos *slices* que constituem a imagem volumétrica de entrada. Em primeiro lugar é aplicado a função *dadwt_full_point_symmetric_extend()* que corresponde ao procedimento {1}, onde é realizada a extensão simétrica nas fronteiras de um determinado *slice*. Posteriormente é executado o procedimento {2} onde é efetuada a leitura de um ficheiro de texto para cada *slice*, em que o seu conteúdo informa o descodificador das direções escolhidas pelo codificador para cada bloco-DA pertencente a esse *slice*. A função que executa esta tarefa é *read_directions_and_apply_transform()*. Finalmente, a transformada discreta *wavelet* com direção adaptativa é aplicada a um *slice* através da execução da função *dadwt_one_dimensional_inverse_transform_int()* que corresponde à etapa {3}. Todas as funções acima citadas estão definidas no ficheiro com nome *idadwt.c*.

Algorithm 2 Algoritmo relativo à DADWT inversa

```
1:  procedure IDADWT(Direction, Level, LevelMax)
2:      Direction ← X, Y ou Z
3:      Level ← LevelX, LevelY ou LevelZ
4:      LevelMax ← MAX(LevelZ, LevelY, LevelX)
5:      i ← 0
6:      for all Tiles do
7:          while i < LevelMax do
8:              for all Direction do
9:                  if Level < LevelMax then
10:                     for all Slice do
11:                         • Extensão simétrica nas fronteiras de cada slice. {1}
12:                         • Leitura dos ficheiro de texto que contém as
13:                         direções escolhidas para todos os Bloco-DA. {2}
14:                         • Aplicar transformada inversa de acordo com vetor
15:                         direcional. {3}
16:                     end for
17:                 end if
18:             end for
19:             i ← i + 1
20:         end while
21:     end for
22: end procedure
```

O esquema JP3D+DADWT ilustrado na figura 3.2 descreve os novos procedimentos com a introdução da DADWT no codificador JP3D. O procedimento adotado na figura 3.2 inicia-se com a introdução da imagem no módulo da transformada, onde se pode escolher entre a DADWT baseada em blocos, a DWT não direcional, ou por fim a não

aplicação de qualquer transformada. Após a escolha, a sua saída gera sub-bandas que correspondem a um nível de decomposição. A sub-banda de menor frequência pode ser de novo reenviada para o módulo da transformada a fim de ser aplicado novo nível de decomposição. Caso contrário, é enviado para o módulo entrópico, onde se é realizado a codificação de entropia utilizando o algoritmo EBCOT. Por fim, obtém-se o fluxo de bits final que corresponde à codificação da imagem.

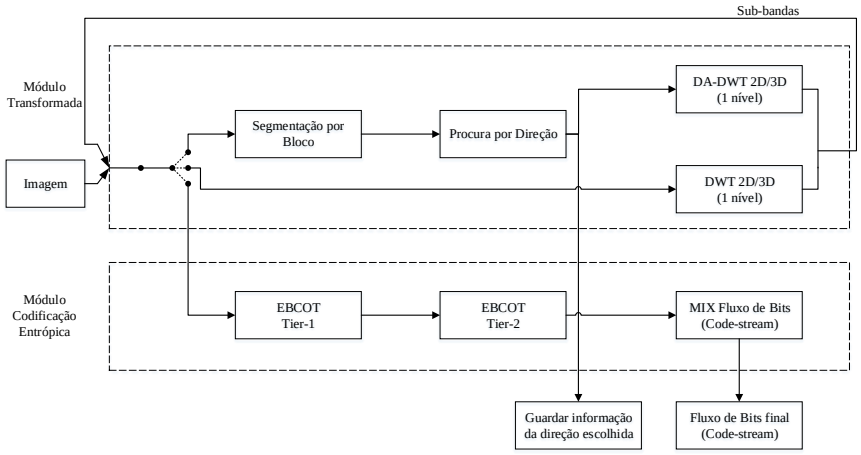


Figura 3.2: Representação esquemática do codificador JP3D com DADWT. [17]

4

Resultados

O Capítulo 4, começa com uma abordagem sobre as métricas utilizadas para avaliação de desempenho da codificação de imagens médicas volumétricas por parte do sistema de codificação de imagens JPEG 2000. De seguida, é apresentado a definição de medidas de compressão utilizadas nesta dissertação. Seguidamente é feito uma referência ao conjunto de imagens médicas, onde é descrito as suas principais características. Por fim, pode-se encontrar uma discussão de resultados relativamente ao valores obtidos na codificação das imagens que constam no conjunto referido anteriormente.

4.1 Métricas de Avaliação

No âmbito da avaliação do desempenho de algoritmos de codificação de imagens existem diversas métricas de erro. Entre estas, as métricas de erro geralmente utilizadas para comparar as diversas técnicas de compressão de imagem, são:

1. Erro Quadrático Médio, (MSE)
2. Relação Sinal-Ruído de Pico, (PSNR)

A MSE corresponde à medida de distorção mais usualmente aplicada e é calculada através do erro quadrático cumulativo entre a imagem original e a comprimida. A fórmula matemática que a define é

$$MSE = \frac{1}{Z \cdot Y \cdot X} \sum_{z=0}^{Z-1} \sum_{y=0}^{Y-1} \sum_{x=0}^{X-1} \left[I(x,y,z) - \hat{I}(x,y,z) \right]^2 \tag{4.1}$$

onde $I(x,y,z)$ é a imagem volumétrica original, $\hat{I}(x,y,z)$ é a versão aproximada, isto é, a imagem descompactada, e $X \times Y \times Z$ estabelece as dimensões da imagem volumétrica.

A relação sinal-ruído de pico (PSNR) define a correlação entre a máxima energia de um sinal e o ruído que afeta a sua representação fidedigna. Em que sinal refere-se aos valores originais de uma determinada imagem e ruído corresponde aos erros introduzidos na compressão da imagem. A PSNR é definida pela fórmula matemática

$$PSNR = 10 \cdot log_{10} \frac{\left(2^B - 1\right)^2}{MSE} \tag{4.2}$$

onde B identifica a profundidade de bit de uma imagem. O PSNR é expresso em dB (decibel).

Uma menor valor de MSE indica uma menor distorção existente entre a imagem original e a aproximada, e consequentemente isto traduz-se numa elevado valor de PSNR, devido às relação inversa existente entre MSE e PSNR. Portanto, pode dizer que quanto maior o valor da relação sinal-ruído de pico melhor a qualidade da imagem aproximada. Na

ausência de ruído, as duas imagens são idênticas, e assim o valor de MSE é zero, consequentemente o valor de PSNR é infinito. Embora que as métricas MSE e PNSR proporcione uma medida simples de cálculo sendo portanto matematicamente conveniente em contextos de otimização, elas não são totalmente fiáveis quando se trata de imagens. "Pois encontra-se demonstrado que MSE e PSNR carece de uma caraterística crucial: a capacidade de avaliar a semelhança de imagem para diferentes tipos de distorção" [21].

4.2 Medidas de Compressão

Assumindo que uma imagem volumétrica tem dimensões de $X \times Y \times Z$ e profundidade de bit B, o número de bits necessários para a representar é de $X \cdot Y \cdot Z \cdot B$ bits. Ao comprimir esta imagem utilizando um codificador, este gera como resultado um fluxo de bits designado por F. Sabendo estas duas informações pode-se definir a taxa de compressão da imagem através da expressão matemática definida por

$$\textbf{Taxa de Compressão} = \frac{X \cdot Y \cdot Z \cdot B}{F} \tag{4.3}$$

Relativamente, a outra medida de compressão como é o caso do cálculo da taxa de bit, na qual representa o número de bits necessários para representar um único píxel é determinado por

$$\textbf{Taxa de bit} = \frac{F}{X \cdot Y \cdot Z} \tag{4.4}$$

4.3 Resultados Experimentais

4.3.1 Esquema de Codificação

As imagens que se encontram descritas na tabela 4.2, são codificadas seguindo o esquema de codificação presente na figura 4.1. Este esquema define o processo de codificação do JP3D, permitindo desta forma a codificação de dados bidimensionais ou tridimensionais. Inicialmente, a imagem de entrada é enviada para o módulo de transformada, onde é permitido escolher se a transformada é aplicada ou não. Caso seja escolhida a opção de aplicar a transformada, é utilizada a transformada discreta *wavelet* não-direcional DWT 1-D em cada uma das direções espaciais. Após aplicação deste módulo na sua saída encontra-se a imagem de entrada decomposta em oito sub-bandas correspondente a um nível de decomposição. Caso se queira mais um nível de decomposição, a sub-banda LLL é novamente enviada para a entrada do módulo de transformada. Apenas no caso de a aplicação da transformada não seja selecionada, os dados de entrada do módulo de transformada, inalterados, são enviados para o módulo entrópico de modo a que seja aplicado

nesses dados a codificação entrópica. O EBCOT [22] é o codificador entrópico utilizado. A transformada discreta *wavelet* é implementada segundo o esquema de *lifting* para os filtros *wavelet* 5x3 reversível e 9x7 irreversível. Os parâmetros e coeficientes de *lifting* dos kernels estão expostos na tabela 4.1.

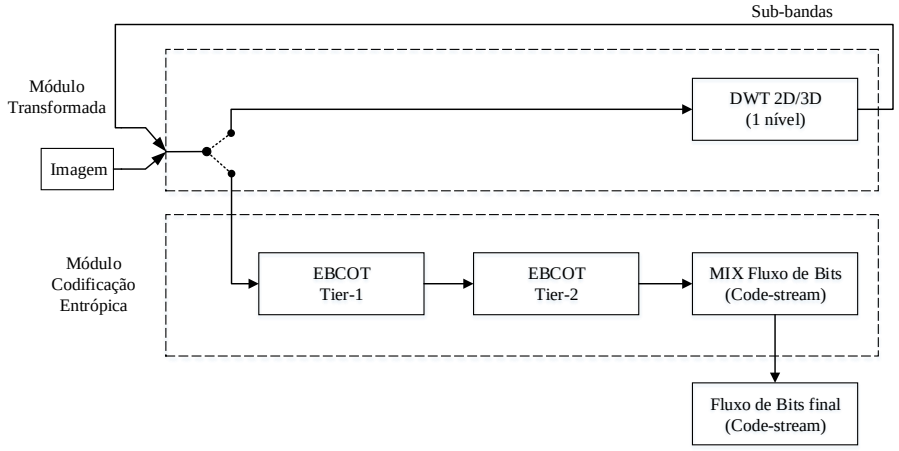


Figura 4.1: Representação esquemática do codificador JP3D. [17]

Tabela 4.1: Parâmetros e coeficientes de *lifting* dos kernels *wavelet* 5x3 e 9x7.

Kernel	M	K_P	K_U	G_L	G_H	i	k	$c_{P,i,k}$	$c_{U,i,k}$
5x3	1	1	1	1	1	0	-1,1	0.5	0.25
9x7	2	1	1	0.812893066	1.230174105	0	-1,1	1.586134342	-0.052980119
						1	-1,1,	-0.882911076	0.443506852

4.3.2 Conjunto de Imagens Volumétricas

Com o propósito de obter resultados relativamente à codificação de imagens médicas volumétricas, reuniu-se um conjunto de imagens obtidas através de várias modalidades frequentemente utilizadas em ambiente hospitalar, tais como CT, MRI e PET. Este conjunto é constituído por dezanove imagens, nove de tipo CT, nove de tipo MRI e finalmente uma de tipo PET. Cada uma destas dezanove imagens encontra-se descrita na tabela 4.2, onde consta as suas principais informações. Todas as imagens volumétricas são em tons de cinza (*grayscale*), porém a profundidade de bit (*bit-depth*) e resolução varia de imagem para imagem.

Todas as imagens que constam na tabela 4.1 estavam originalmente armazenadas em ficheiros DICOM, que correspondem a um dos vários formatos possíveis para armazenamento de imagens médicas. Optou-se por escolher este tipo de formato, uma vez que este

Tabela 4.2: Descrição do conjunto de imagens médicas volumétricas.

Nome	Resolução (X/Y/Z em pixeis)	Profundidade de bit (bits)	Tipo de cor	Conteúdo
CT1	512/512/209	12	*grayscale*	Digitalização cérebro
CT2	512/512/245	16	*grayscale*	Digitalização crânio
CT3	512/512/400	16	*grayscale*	Digitalização anca
CT4	512/512/140	16	*grayscale*	Digitalização pélvis
CT5	512/512/460	16	*grayscale*	Digitalização ombro
CT6	512/512/350	16	*grayscale*	Digitalização joelho
CT7	512/512/150	16	*grayscale*	Digitalização tornozelo
CT8	512/512/148	12	*grayscale*	Digitalização crânio até meio das coxas
CT9	512/512/113	12	*grayscale*	Digitalização cabeça/pescoço
MRI1	512/512/014	12	*grayscale*	Digitalização ombro (cima)
MRI2	512/512/012	12	*grayscale*	Digitalização ombro (frente)
MRI3	256/256/022	12	*grayscale*	Digitalização Neuro-crânio
MRI4	166/195/024	08	*grayscale*	Não disponível
MRI5	256/256/160	12	*grayscale*	Digitalização cabeça/pescoço
MRI6	320/320/045	12	*grayscale*	Digitalização próstata
MRI7	512/512/015	12	*grayscale*	Digitalização fémur
MRI8	256/256/020	12	*grayscale*	Digitalização cabeça
MRI9	256/256/016	08	*grayscale*	Digitalização coração
PET1	144/144/255	16	*grayscale*	Não disponível

é globalmente aceite no meio médico. No entanto, o codificador utilizado nesta dissertação só aceita imagens em que o formato seja *RAW*. Sendo assim, utilizou-se o programa *ImageJ* de modo a permitir a conversão de ficheiros com a extensão *DICOM* para a extensão *RAW*. Outra situação que foi necessário alterar é relativo à profundidade de bit (*bit-depth*), isto é, as imagens com profundidade de bit superior a oito bits foram manualmente modificadas de modo a redefinir a profundidade de bit para os oito bits, através de

$$\hat{I} = \frac{I - min}{max - min} \times 255 \tag{4.5}$$

Onde I corresponde ao valor de um determinado píxel, max corresponde ao maior valor de píxel da imagem e min ao menor.
Todas as imagens que constam na tabela 4.2, excetuando as imagens MRI4 e MRI9, sofreram as alterações acima descritas.

4.3.3 Resultados

De modo a obter resultados representativos na codificação de imagens médicas volumétricas, foi adotado o processo de codificação existente na figura 4.1 utilizando as

imagens da tabela 4.2. Inicialmente optou-se por avaliar qual o número desejável de níveis de decomposição a realizar nas três direções espaciais em cada uma das imagens de modo a obter uma compactação de energia quase-ótima. No entanto, é necessário alcançar um compromisso no número de níveis de decomposição a utilizar, uma vez que quanto maior o número de decomposições maior será o tempo de processamento e utilização de memória por parte da transformada discreta *wavelet*. Num outro ponto, será analisado a relação entre o número de decomposições aplicadas e a qualidade da imagem comprimida, obtida pela métrica de erro PSNR. Por último, será averiguado o ganho obtido na compressão das imagens com a aplicação da decomposição axial.

Na figura 4.2 encontram-se os valores de taxa de bit para cada uma das imagens médicas volumétricas, em que com taxa de bit quer-se dizer o número de bits utilizados para representar cada píxel pertencente a uma determinada imagem comprimida. Em cada uma destas imagens foi aplicado um diferente número de níveis de decomposição, definidos da seguinte forma, $(n,n,0)\ \forall n \in \{0..6\}$. Neste primeiro caso apenas é realizado níveis de decomposição nas direções espaciais X e Y, ou seja, dentro do *slice*. A transforma discreta *wavelet* aplicada, utilizou o *kernel* 5x3 permitindo desta forma obter compressão sem perdas.

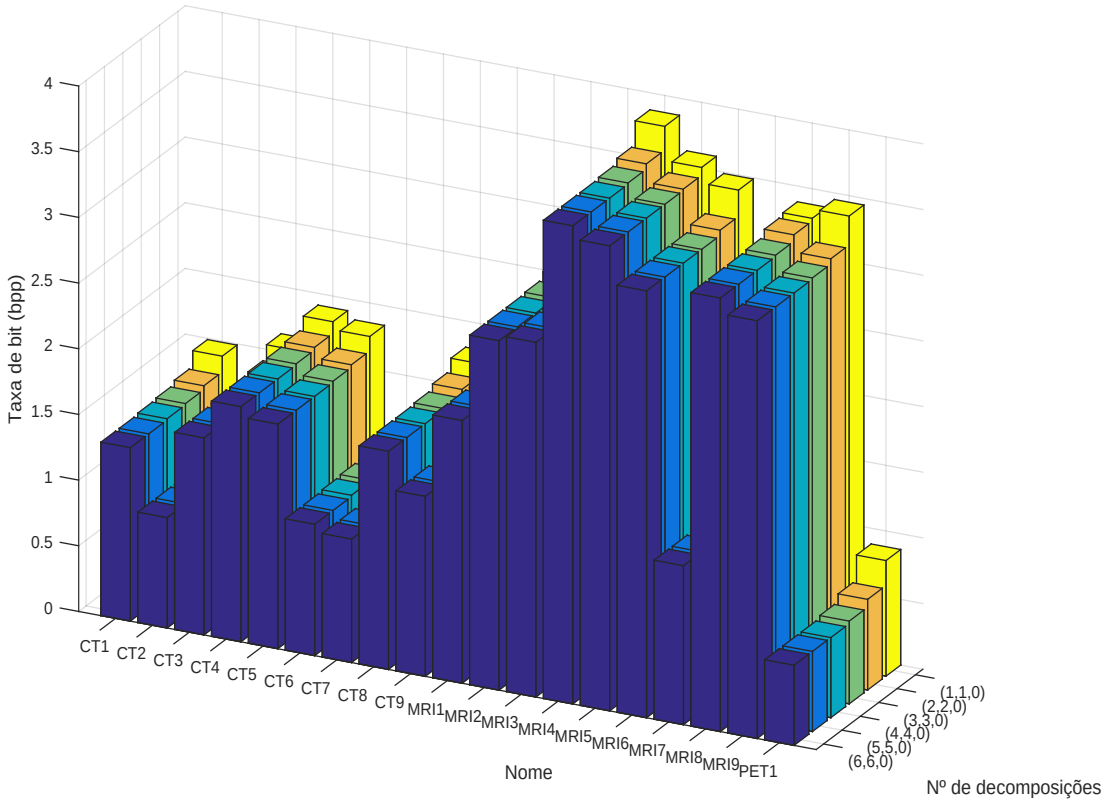


Figura 4.2: Resultados subsequentes da decomposição na direção X e Y, utilizando *kernel* 5x3.

Os resultados observados na figura 4.2 são obtidos através da compressão das imagens com rearranjo da profundidade de bit do seu valor original para um valor de 8 bits. Isto deve-se ao facto de o codificador não executar (somente quando é utilizado o *kernel* 5x3) com imagens com profundidade de bit superiores a 8 bits, então optou-se por esta solução.

Uma análise aos resultados da figura 4.2 permite concluir que, geralmente, quatro níveis de decomposição nas direções X e Y são suficientes para atingir um compressão de imagem para as imagens do tipo CT, MRI e PET significativa, uma vez que um maior número de decomposições não resulta num ganho suficiente que o justifique. Porém, verifica-se que em determinadas imagens CT a aplicação de cinco níveis de decomposição permite obter uma menor taxa de bit. Já nas imagens MRI verifica-se que não existe qualquer redução no valor de taxa de bit ao aplicar um número de decomposições superior a quatro e que em certos casos três níveis de decomposição são suficientes.

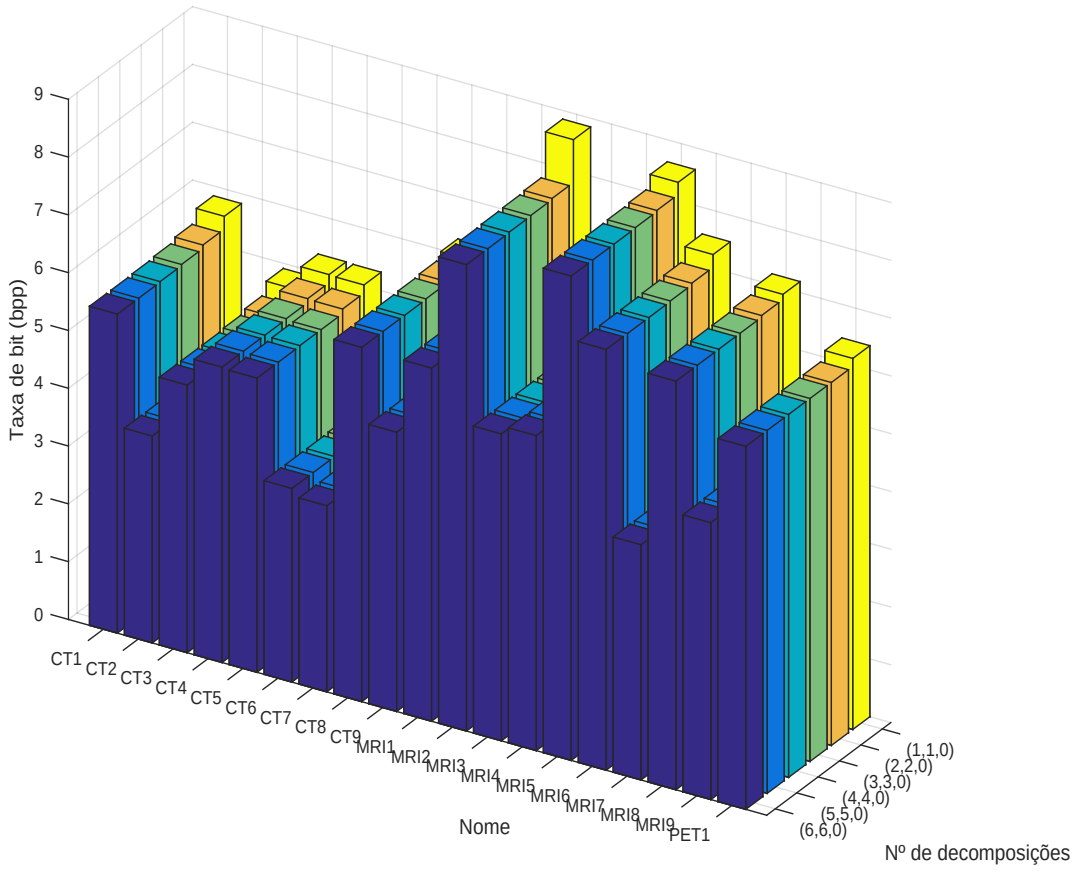


Figura 4.3: Resultados obtidos utilizando decomposição na direção X e Y, utilizando *kernel* 9x7.

A figura 4.3 retrata os valores de taxa de bit conseguidos com a codificação com perdas das imagens volumétricas. Considera-se esta codificação com perdas em virtude da

utilização do *kernel* 9x7 na DWT. As imagens codificadas têm como profundidade de bit o seu valor original, ou seja, não houve nenhum rearranjo no valor da profundidade de bit. Analisando os valores da figura 4.3 determina-se que o número de níveis de decomposições a aplicar nas direções X e Y nas imagens do tipo CT, MRI e PET são quatro.

Visto que já foi definido o número ótimo de níveis de decomposições relativamente à direção X e Y, têm-se agora que determinar qual o número ideal de decomposições no que respeita à direção axial. Para isso, optou-se que para cada imagem definida em 4.2 fosse codificada com variação nos níveis de decomposição de $(4,4,n)\ \forall n \in \{0...4\}$. Assim sendo, o objetivo de verificar o número ideal de níveis a aplicar quando ocorre codificação na direção axial pode ser obtido ao observar os resultados nas figuras 4.4 e 4.5 que correspondem à utilização do *kernel* 5x3 e 9x7, respetivamente.

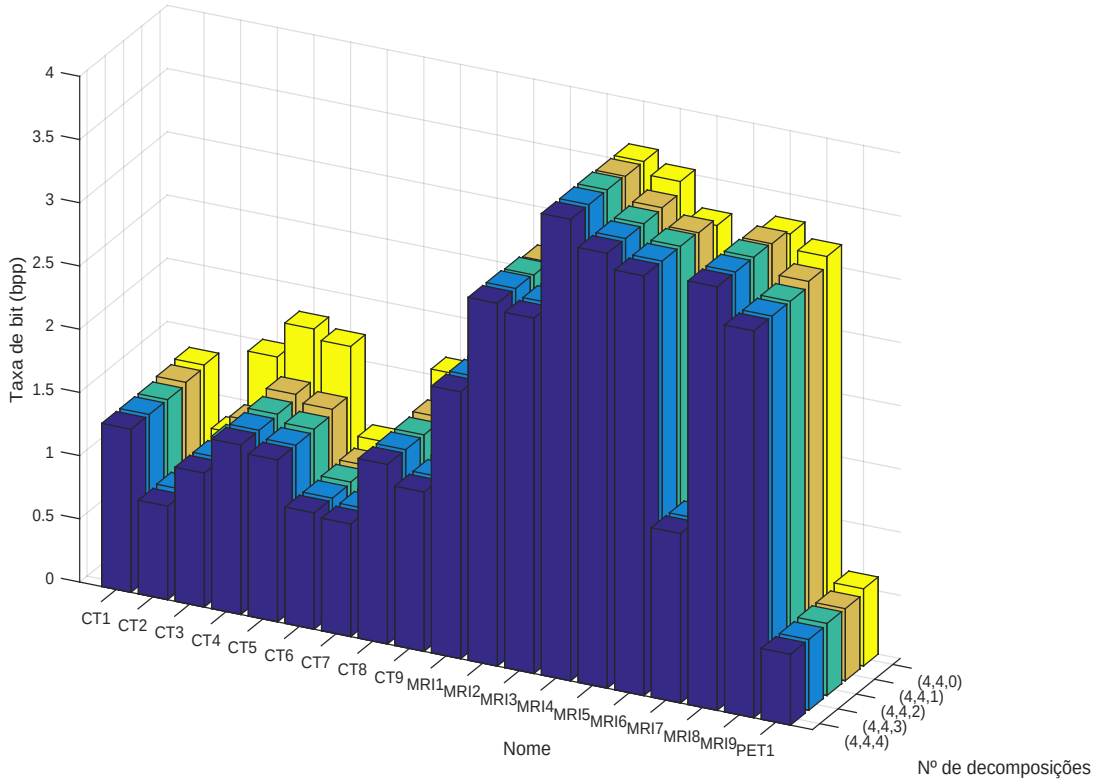


Figura 4.4: Resultados obtidos através da decomposição na direção X, Y e Z, utilizando *kernel* 5x3.

No que respeita à figura anterior, cada barra identifica o valor de taxa de bit obtida com a compressão de uma determinada imagem com a aplicação de um determinado número de decomposições. Posto isto, ao realizar uma análise cuidada dos seus valores verifica-se que a maior redução no valor da taxa de bit ocorre com a aplicação de um nível de decomposição axial, porém ainda ocorre um redução assinalável com a aplicação

de um outro nível de decomposição axial, já com a aplicação de três níveis só existe uma diminuição no valor de taxa de bit em algumas imagens, sendo que nesses casos a diminuição no valor de taxa de bit é diminuta relativamente às reduções obtidas nos casos anteriores. Portanto, conclui-se que o número ideal de níveis de decomposições na direção axial corresponde a dois para as imagens do tipo CT e uma no caso da imagem PET. Todavia, nos casos das imagens obtidas com a técnica MRI, mais propriamente as imagens MRI1, MRI2, MRI3, MRI4, MRI6, MRI7 e MRI8, indicam que não se obtém nenhuma vantagem na aplicação de qualquer nível de decomposição na direção Z, já que é na codificação utilizando decomposição (4,4,0) onde se obtém o menor valor de taxa de bit.

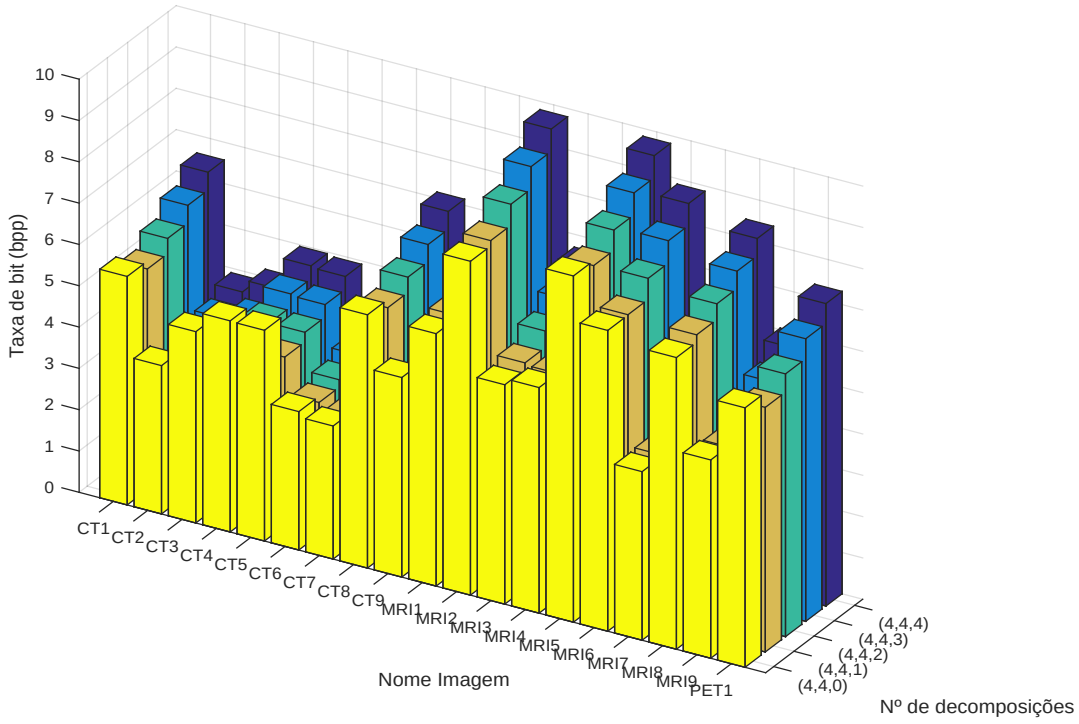


Figura 4.5: Resultados obtidos utilizando decomposição na direção X, Y e Z, utilizando *kernel* 9x7.

Comparativamente, os resultados obtidos com a codificação de todas as imagens com a profundidade de bit original e utilizando o *kernel* 9x7 na DWT, verifica-se, geralmente, que a melhor combinação de níveis de decomposição nas três direções é (4,4,1) nas imagens do tipo CT e PET. No entanto, as imagens do tipo MRI, mais precisamente as imagens MRI1, MRI2, MRI3, MRI6, MRI7 e MRI8 não exibem qualquer ganho com a utilização da decomposição axial na codificação. Portanto, pode-se afirmar que tanto no caso da utilização do *kernel* 5x3 ou 9x7 na DWT não se obtém melhoria na taxa de bit para as imagens MRI.

Agora que sabemos qual o número ideal de níveis de decomposição quando se codifica uma imagem médica volumétrica, irá ser apresentado a relação existente entre a aplicação dos níveis de decomposição e o valor obtido de PSNR da imagem. Recordar que o valor de PSNR indica a perda de qualidade da imagem obtida após codificação relativamente à imagem original e que o seu valor é dado em decibéis (dB). Importa também indicar que quanto maior o valor de PSNR menor é a perda de informação relativamente à imagem original. Neste seguimento, as figuras 4.6 e 4.7 mostram os valores obtidos de PSNR para a codificação das imagens com profundidade de bit igual a oito bits onde a DWT utiliza o *kernel* 9x7 e para os níveis de decomposição $(n,n,0)$ $\forall$ n $\in \{0...6\}$ e $(4,4,n)$ $\forall$ n $\in \{0...4\}$, respetivamente. Na obtenção dos valores de PSNR também utilizou-se o controlo de taxa de bit na compressão, isto é, o valor de taxa de bit seja igual ao obtido com a utilização do *kernel* 5x3.

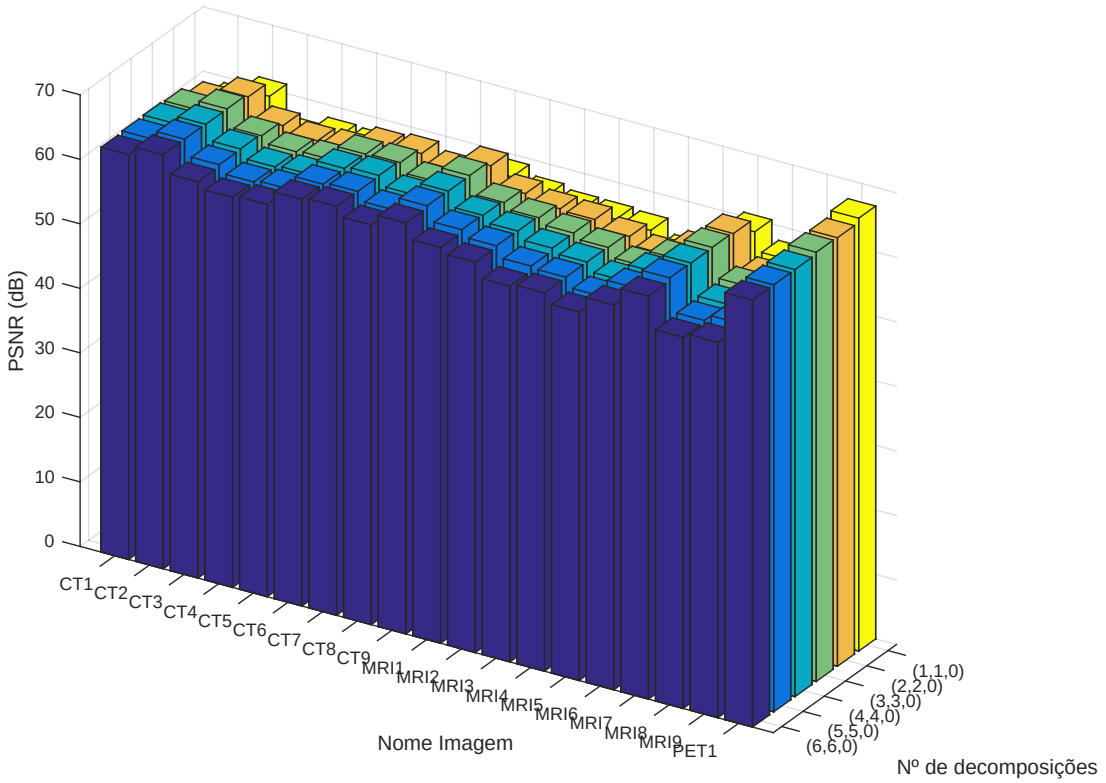


Figura 4.6: Resultados obtidos utilizando decomposição na direção X e Y, utilizando *kernel* 9x7.

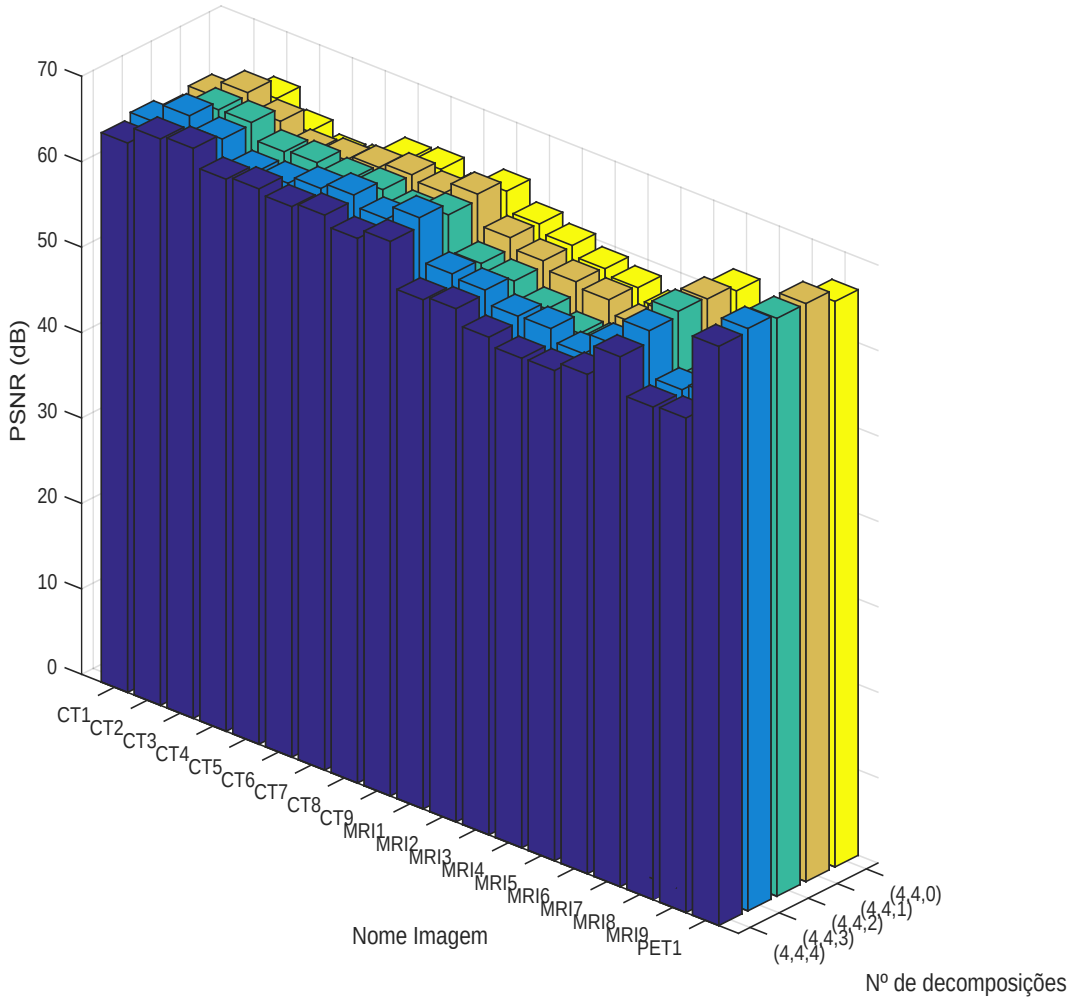


Figura 4.7: Resultados obtidos utilizando decomposição na direção X, Y e Z, utilizando *kernel* 9x7.

Nas tabelas 4.3 e 4.4 para o *kernel* 5x3 e 9x7, respetivamente, é apresentado o ganho de taxa de bit obtido pela utilização de decomposição axial na compressão do conjunto das imagens.

Relativamente aos valores apresentados na tabela 4.3, verifica-se que na ausência de decomposição axial em média são necessários 1.3522 bits por píxel para as imagens CT e 0.61 bpp para a imagem PET após compressão da imagem. Porém, ao ser aplicado uma codificação com aplicação do número ideal de níveis de decomposição para este caso específico, (4,4,2) e (4,4,1), verifica-se que são necessários 1.14 bpp e 0.57 bpp para as imagens CT e PET, respetivamente. Consequentemente, pode-se constatar que existe uma redução de 0.2122 bpp na taxa de bit para as imagens CT, enquanto para a imagem PET ocorre uma redução de taxa de bit de 0.04 bpp. Este resultados foram obtidos tendo apenas em conta as imagens CT1, CT2, CT3, CT4, CT5, CT6, CT7, CT8, CT9 e PET1, pois já

Tabela 4.3: Valores obtidos para os níveis de decomposição (4,4,0) e (4,4,2) nas imagens CT e (4,4,0) e (4,4,1) na imagem PET, para o *kernel* 5x3.

Nome	Taxa de bit (bpp)	
	Níveis de decomposição	
	(4,4,0)	(4,4,2)
CT1	1.33	1.29
CT2	0.84	0.75
CT3	1.51	1.07
CT4	1.79	1.35
CT5	1.71	1.29
CT6	1.01	0.93
CT7	0.95	0.89
CT8	1.66	1.42
CT9	1.37	1.27
Nome	Níveis de decomposição	
	(4,4,0)	(4,4,1)
PET1	0.61	0.57

Tabela 4.4: Valores obtidos para níveis de decomposição (4,4,0) e (4,4,1) nas imagens CT e PET, para o *kernel* 9x7.

Nome	Taxa de bit (bpp)	
	Níveis de decomposição	
	(4,4,0)	(4,4,1)
CT1	5.53	5.34
CT2	3.59	3.16
CT3	4.63	3.32
CT4	5.11	4.12
CT5	5.10	4.08
CT6	3.35	3.20
CT7	3.22	3.08
CT8	6.13	5.93
CT9	4.83	4.41
PET1	6.28	5.92

se constatou que não existe melhoria na taxa de bit com a aplicação de decomposição na direção axial para imagens MRI.

No caso da aplicação do *kernel* 9x7 na codificação das imagens, tabela 4.4, e sem aplicação de qualquer nível de decomposição axial constata-se que em média a taxa de bit obtida para as imagens CT é 4.61 bpp e para a imagem PET é de 6.28 bpp. Na utilização de um nível de decomposição na direção axial a taxa de bit obtida para as imagens CT e PET são 4.07111 bpp e 5.92 bpp, respetivamente. Em consequência disto, nota-se uma redução na taxa de bit para os casos em que a codificação recorre à decomposição axial. Para as imagens CT a redução é na ordem de 0.53889 bpp, enquanto para a imagem PET é de 0.36 bpp.

Na tabela 4.5 consta os valores de PSNR para as imagens CT e PET com profundidade de bit igual a oito bits. Estes valores foram obtidos com a compressão de cada imagem utilizando o *kernel* 9x7. Em cada compressão definiu-se que o valor da taxa de bit seja igual ao que foi obtido na compressão das imagens utilizando o *kernel* 5x3 (tabela 4.3). No anexo B apresenta-se para cada imagem um determinado *slice* obtido após a compressão utilizando o *kernel* 5x3 e o mesmo *slice* mas agora obtido após a compressão utilizando o *kernel* 9x7. Permitindo assim avaliar a possibilidade de existência de perda de qualidade na compressão da imagem utilizando o *kernel* 9x7.

Uma vez que os valores de PSNR para cada imagem são elevados e também com a ob-

Tabela 4.5: Valores de PSNR na aplicação do *kernel* 9x7 com bit-rate igual ao obtido com o *kernel* 5x3 para os níveis de decomposição (4,4,2) para CT e (4,4,1) para PET.

Nome	PSNR (dB)	
	Níveis de decomposição	
	(4,4,0)	(4,4,2)
CT1	63.01660	63.95134
CT2	64.17920	66.27982
CT3	61.81065	66.30914
CT4	60.46588	64.39902
CT5	60.75396	64.64463
CT6	63.11642	64.16345
CT7	63.46583	64.51441
CT8	61.93701	63.40538
CT9	63.93626	64.59617
Nome	Níveis de decomposição	
	(4,4,0)	(4,4,1)
PET1	66.20039	67.65847

servação das imagens que constam no anexo B, pode-se afirmar que não existe perda de qualidade detetável relativamente ao sistema visual humano.

A figura 4.8 discrimina a diminuição obtida em termos de taxa de bit com a utilização de decomposição na direção axial na codificação utilizando o *kernel* 9x7 e 5x3 para as imagens do tipo CT e PET.

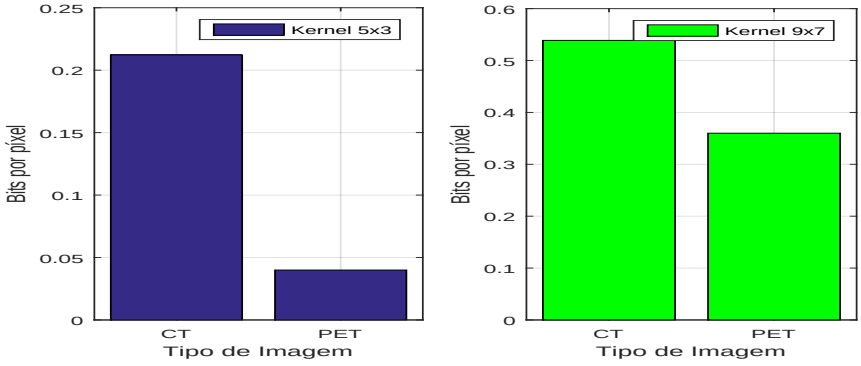


Figura 4.8: Diminuição obtida relativamente à taxa de bit com a codificação utilizando os kernels 5x3 e 9x7, e aplicando decomposição na direção axial para as imagens CT e PET.

As figuras 4.2, 4.3, 4.4, 4.5, 4.6 e 4.7 têm os seus valores apresentados na forma de tabelas no anexo A.

5

Conclusões

O objetivo principal desta tese consistiu no estudo da compressão de imagens médicas volumétricas, uma vez que atualmente ainda não está bem definido qual o método, assim como a forma mais eficiente das configurações do codificador para atingir a melhor compressão de imagem possível. Com este objetivo, começou-se por escolher o sistema de codificação de imagens, JPEG 2000. No entanto, deste sistema de codificação apenas foi abordado de uma forma mais detalhada a etapa dois do compressor de imagens. Esta etapa diz respeito à implementação da transforma discreta *wavelet*, também conhecida como DWT.

Visto que este sistema de compressão permite uma abrangente gama de configurações para a compressão de imagens, tal como, a escolha entre a compressão com perdas em oposição à compressão sem perdas, ou por exemplo a escolha do número de níveis de decomposição a aplicar nas direções X, Y e Z da imagem. Existem outras configurações tal como o controlo da taxa de bit para a compressão com perdas. Na obtenção dos resultados apresentados nesta dissertação apenas foram utilizadas as configurações acima descritas.

Com os resultados exibidos no capítulo 4, pode-se retirar como conclusão que tanto para o caso da compressão com e sem perdas, o número aconselhável de níveis de decomposição a serem realizados nas direções X e Y é quatro, para as imagens de tipo CT, MRI e PET. Neste seguimento, constatou-se que o número de decomposições desejável na direção Z também no caso da compressão sem perdas é duas para as imagens CT e para a imagem PET é necessário apenas uma, enquanto na compressão com perdas é apenas necessário uma decomposição. Estes valores foram obtidos ao ter em conta o compromisso existente entre o número de decomposições a realizar e o aumento do tempo de processamento e de memória que uma nova decomposição acarreta. Outra conclusão que é possível apontar consiste que as imagens obtidas pela técnica MRI não obtém qualquer ganho de compressão com a aplicação de decomposição na direção axial.

5.1 Trabalho Futuro

Os sistemas de compressão de imagens estão em constante desenvolvimento, através da implementação de novas metodologias ou melhoria das já existentes. Portanto, pode-se afirmar que ainda há muito trabalho a ser desenvolvido nesta área. Por este motivo considera-se que no seguimento do trabalho desenvolvido nesta dissertação, pode-se, como trabalho futuro proceder com a continuação da implementação da DADWT ou implementar uma outra alternativa a esta designada por SD-DADWT. Uma outra proposta consiste na avaliação subjetiva da qualidade das imagens médicas volumétricas após utilização do sistema de compressão.

Bibliografia

[1] a. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG 2000 still image compression standard," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 36–58, 2001.

[2] P. Schelkens, A. Skodras, and T. Ebrahimi, *The JPEG 2000 Suite*, 1st ed. Wiley Publishing, 2009.

[3] T. Bruylants, A. Munteanu, and A. Alecu, "Volumetric image compression with JPEG2000," *SPIE The International ...*, pp. 2–3, 2007. [Online]. Available: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Volumetric+image+compression+with+JPEG2000#0

[4] ISO/IEC JTC 1/SC29/WG Programme of Work. (2015) JPEG2000 image coding system. [Online]. Available: http://jpeg.org/jpeg2000/

[5] DICOM. (2015) Digital Imaging and Communications in Medicine. [Online]. Available: http://dicom.nema.org/

[6] International Telecommunications Union, "ITU-T Recomendation T.800. JPEG 2000 image coding system: Core coding system," 2002.

[7] M. Rabbani and R. Joshi, *An overview of the JPEG 2000 still image compression standard*, 2002, vol. 17.

[8] International Telecommunications Union , "ITU-T Recomendation T.809. JPEG 2000 image coding system: Extensions for three-dimensional data," 2011.

[9] D. S. Taubman and M. W. Marcellin, *JPEG2000 Image Compression Fundamentals, Standards and Practice*, 3rd ed. Springer Science+Business Media, LLC, 2004.

[10] D. Le Gall and A. Tabatabai, "Sub-band coding of digital images using symmetric short kernel filters and arithmetic coding techniques," pp. 761–764 vol.2, Apr 1988.

[11] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *Image Processing, IEEE Transactions on*, vol. 1, no. 2, pp. 205–220, Apr 1992.

[12] W. Daubechies Ingrid; Sweldens, "Factoring wavelet transforms into lifting steps," *Journal of Fourier Analysis and Applications*, vol. 4, pp. 247–269, 1998. [Online]. Available: http://dx.doi.org/10.1007/BF02476026

[13] W. Sweldens, "Lifting scheme: a new philosophy in biorthogonal wavelet constructions," *Proc. SPIE*, vol. 2569, pp. 68–79, 1995. [Online]. Available: http://dx.doi.org/10.1117/12.217619

[14] ——, "The lifting scheme: A custom-design construction of biorthogonal wavelets," *Applied and Computational Harmonic Analysis*, vol. 3, no. 2, pp. 186–200, 1996. [Online]. Available: https://lirias.kuleuven.be/handle/123456789/134869

[15] ——, "The lifting scheme: A construction of second generation wavelets," *SIAM Journal on Mathematical Analysis*, vol. 29, no. 2, pp. 511–546, 1998. [Online]. Available: http://dx.doi.org/10.1137/S0036141095289051

[16] A. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo, "Wavelet transforms that map integers to integers," *Applied and Computational Harmonic Analysis*, vol. 5, no. 3, pp. 332 – 369, 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1063520397902384

[17] T. Bruylants, A. Munteanu, and P. Schelkens, "Wavelet based volumetric medical image compression," *Signal Processing: Image Communication*, vol. 31, pp. 112–133, 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0923596514001854

[18] C.-L. Chang and B. Girod, "Direction-Adaptive Discrete Wavelet Transform via Directional Lifting and Bandeletization," pp. 3–5, 2006.

[19] Chuo-Ling Chang and B. Girod, "Direction-adaptive discrete wavelet transform for image compression." *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, vol. 16, no. 5, pp. 1289–1302, 2007.

[20] A. Munteanu, O. M. Surdu, J. Cornelis, and P. Schelkens, "Segmentation-driven direction-adaptive discrete wavelet transform," vol. 0, no. 1, pp. 437–440, 2007.

[21] E. A. Silva, K. Panetta, and S. S. Agaian, "Quantifying image similarity using measure of enhancement by entropy," *Proc. SPIE*, vol. 6579, pp. 65 790U–65 790U–12, 2007. [Online]. Available: http://dx.doi.org/10.1117/12.720087

[22] E. W. M. S. G. Taubman David; Ordentlich, "Embedded Block Coding in JPEG2000," *HP Labs Technical Reports*, p. 36, 2001.

A

Anexo A

No anexo A consta os valores obtidos através da execução do sistema de codificação JPEG 2000 no conjunto de imagens referidas na subsecção 4.3.2 e utilizando um vasto conjunto de níveis de decomposição. Estes resultados são apresentados na forma de tabelas.

Tabela A.1: Valores de taxa de bit apresentados na figura 4.2 obtidos após compressão da conjunto de imagens com aplicação de um determinado número de decomposições na direção X e Y, utilizando o *kernel* 5x3.

Nome	Níveis de decomposição					
	(6,6,0)	(5,5,0)	(4,4,0)	(3,3,0)	(2,2,0)	(1,1,0)
CT1	1.32	1.32	1.33	1.34	1.37	1.49
CT2	0.84	0.84	0.84	0.85	0.88	0.95
CT3	1.50	1.50	1.51	1.51	1.54	1.64
CT4	1.79	1.79	1.79	1.80	1.82	1.91
CT5	1.71	1.71	1.71	1.72	1.74	1.85
CT6	1.00	1.00	1.01	1.02	1.03	1.09
CT7	0.94	0.94	0.95	0.95	0.96	1.02
CT8	1.66	1.66	1.66	1.67	1.71	1.80
CT9	1.37	1.37	1.37	1.39	1.41	1.50
MRI1	2.00	2.00	2.00	2.00	2.03	2.23
MRI2	2.66	2.66	2.66	2.67	2.72	2.93
MRI3	2.70	2.70	2.70	2.72	2.77	2.78
MRI4	3.64	3.64	3.64	3.65	3.69	3.87
MRI5	3.54	3.54	3.54	3.54	3.55	3.61
MRI6	3.25	3.25	3.25	3.25	3.29	3.49
MRI7	1.21	1.21	1.21	1.22	1.25	1.35
MRI8	3.30	3.30	3.30	3.31	3.36	3.38
MRI9	3.18	3.18	3.18	3.19	3.23	3.45
PET1	0.61	0.61	0.61	0.63	0.69	0.88

Tabela A.2: Valores de taxa de bit apresentados na figura 4.3 obtidos após compressão da conjunto de imagens com aplicação de um determinado número de decomposições na direção X e Y, utilizando o *kernel* 9x7.

Nome	Níveis de decomposição					
	(6,6,0)	(5,5,0)	(4,4,0)	(3,3,0)	(2,2,0)	(1,1,0)
CT1	5.52	5.52	5.53	5.55	5.61	5.83
CT2	3.58	3.58	3.59	3.60	3.62	3.76
CT3	4.63	4.63	4.63	4.65	4.71	4.91
CT4	5.11	5.11	5.11	5.12	5.19	5.33
CT5	5.09	5.09	5.10	5.10	5.18	5.32
CT6	3.35	3.35	3.35	3.36	3.37	3.47
CT7	3.22	3.22	3.22	3.23	3.23	3.34
CT8	6.13	6.13	6.13	6.14	6.16	6.27
CT9	4.83	4.83	4.83	4.84	4.88	5.09
MRI1	6.11	6.11	6.11	6.12	6.12	6.34
MRI2	8.07	8.07	8.08	8.09	8.11	8.85
MRI3	5.31	5.31	5.31	5.31	5.32	5.33
MRI4	5.45	5.45	5.46	5.46	5.49	5.56
MRI5	8.38	8.38	8.38	8.39	8.41	8.62
MRI6	7.28	7.28	7.28	7.29	7.32	7.54
MRI7	4.07	4.07	4.08	4.09	4.11	4.19
MRI8	7.07	7.07	7.07	7.07	7.10	7.19
MRI9	4.79	4.80	4.80	4.80	4.80	4.82
PET1	6.28	6.28	6.28	6.28	6.28	6.42

Tabela A.3: Valores de PSNR apresentados na figura 4.6 obtidos após compressão da conjunto de imagens com aplicação de um determinado número de decomposições na direção X e Y, utilizando o *kernel* 9x7.

Nome	Níveis de decomposição					
	(6,6,0)	(5,5,0)	(4,4,0)	(3,3,0)	(2,2,0)	(1,1,0)
CT1	62.72891	62.91029	63.01660	62.84055	61.60153	59.11457
CT2	64.18843	64.23063	64.17920	64.22002	63.72354	61.44884
CT3	61.41373	61.71433	61.81065	61.48623	60.70238	56.69806
CT4	60.42934	60.47593	60.46588	60.45818	59.79314	58.06283
CT5	60.80610	60.83757	60.75396	60.68817	60.02262	57.85298
CT6	63.04210	63.02089	63.11642	62.96383	61.77625	51.90514
CT7	63.37146	63.34563	63.46583	63.10449	62.14473	47.70906
CT8	61.96046	61.98470	61.93701	61.65538	61.01697	57.74501
CT9	63.68557	64.01184	63.93626	63.98513	63.14502	59.07531
MRI1	61.33253	61.74626	61.61337	61.23735	60.26836	57.57957
MRI2	60.48968	60.68767	60.62054	60.32119	59.35507	56.89350
MRI3	58.23260	58.98415	59.41513	59.40211	59.14572	56.82662
MRI4	58.63303	58.72000	58.70919	58.60956	57.94708	56.53995
MRI5	57.13081	57.25045	57.34671	57.21206	56.67753	54.32665
MRI6	59.62944	59.94950	60.00096	59.70933	58.73142	55.83487
MRI7	62.47588	62.93164	62.88492	62.95351	62.74051	60.65451
MRI8	57.51254	57.77370	58.01849	58.41003	57.54444	57.74320
MRI9	58.18841	58.79076	59.14482	59.28986	59.66861	59.50792
PET1	66.18001	66.18641	66.20039	66.52588	66.42356	67.09351

Tabela A.4: Valores de taxa de bit apresentados na figura 4.4 obtidos após compressão da conjunto de imagens com aplicação de um determinado número de decomposições na direção X, Y e Z, utilizando o *kernel* 5x3.

Nome	Níveis de decomposição				
	(4,4,4)	(4,4,3)	(4,4,2)	(4,4,1)	(4,4,0)
CT1	1.29	1.29	1.29	1.31	1.33
CT2	0.74	0.75	0.75	0.76	0.84
CT3	1.06	1.06	1.07	1.11	1.51
CT4	1.34	1.34	1.35	1.39	1.79
CT5	1.28	1.28	1.29	1.33	1.71
CT6	0.92	0.92	0.93	0.94	1.01
CT7	0.89	0.89	0.89	0.90	0.95
CT8	1.42	1.42	1.42	1.44	1.66
CT9	1.26	1.26	1.27	1.28	1.37
MRI1	2.11	2.11	2.11	2.10	2.00
MRI2	2.87	2.87	2.86	2.84	2.66
MRI3	2.81	2.81	2.80	2.79	2.70
MRI4	3.65	3.65	3.65	3.64	3.64
MRI5	3.44	3.44	3.44	3.45	3.54
MRI6	3.32	3.32	3.32	3.31	3.25
MRI7	1.34	1.34	1.33	1.32	1.21
MRI8	3.35	3.35	3.35	3.34	3.30
MRI9	3.06	3.06	3.06	3.10	3.18
PET1	0.56	0.56	0.57	0.57	0.61

Tabela A.5: Valores de taxa de bit apresentados na figura 4.5 obtidos após compressão da conjunto de imagens com aplicação de um determinado número de decomposições na direção X, Y e Z, utilizando o *kernel* 9x7.

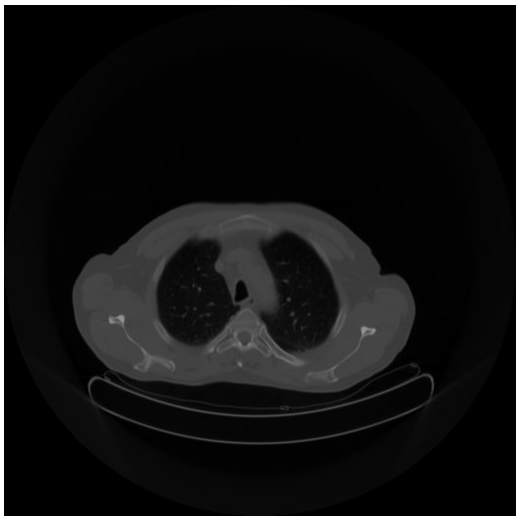
Nome	Níveis de decomposição				
	(4,4,4)	(4,4,3)	(4,4,2)	(4,4,1)	(4,4,0)
CT1	6.57	6.14	5.71	5.34	5.53
CT2	3.89	3.63	3.37	3.16	3.59
CT3	4.19	3.85	3.52	3.32	4.63
CT4	4.96	4.65	4.34	4.12	5.11
CT5	4.92	4.61	4.31	4.08	5.10
CT6	3.83	3.60	3.38	3.20	3.35
CT7	3.72	3.49	3.26	3.08	3.22
CT8	7.15	6.72	6.30	5.93	6.13
CT9	5.71	5.26	4.81	4.41	4.83
MRI1	7.35	6.95	6.56	6.18	6.11
MRI2	9.80	9.26	8.72	8.21	8.08
MRI3	6.69	6.28	5.87	5.47	5.31
MRI4	6.95	6.44	5.91	5.42	5.46
MRI5	9.81	9.28	8.75	8.26	8.38
MRI6	8.87	8.34	7.80	7.29	7.28
MRI7	5.13	4.79	4.44	4.11	4.08
MRI8	8.46	8.04	7.62	7.24	7.06
MRI9	6.04	5.58	5.12	4.73	4.80
PET1	7.32	6.84	6.36	5.92	6.28

Tabela A.6: Valores de PSNR apresentados na figura 4.7 obtidos após compressão da conjunto de imagens com aplicação de um determinado número de decomposições na direção X, Y e Z, utilizando o *kernel* 9x7.

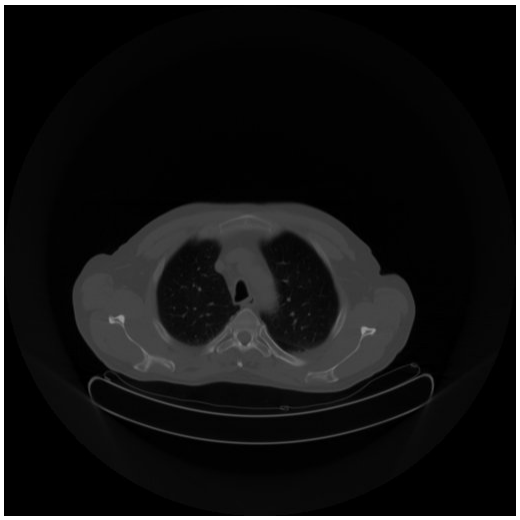
Nome	Níveis de decomposição				
	(4,4,4)	(4,4,3)	(4,4,2)	(4,4,1)	(4,4,0)
CT1	64.30068	64.95783	63.95134	64.62750	63.01660
CT2	66.29323	67.29323	66.27982	66.57670	64.17920
CT3	66.71037	66.08499	66.30914	64.73585	61.81065
CT4	64.61569	63.69833	64.39902	62.74851	60.46588
CT5	65.00002	63.94989	64.64463	63.23599	60.75396
CT6	64.43759	64.85212	64.16345	63.88591	63.11642
CT7	64.96003	65.59897	64.51441	64.53770	63.46583
CT8	63.74793	64.31334	63.40538	63.95312	61.93701
CT9	64.90933	65.97527	64.59617	65.31355	63.93626
MRI1	59.59593	60.91600	60.41972	61.72996	61.61337
MRI2	60.06971	60.53839	59.87983	60.53435	60.62054
MRI3	58.24857	58.99770	58.12717	59.59876	59.41513
MRI4	57.30417	59.06673	56.78473	59.00264	58.70919
MRI5	57.37545	57.63976	56.95430	57.44528	57.34671
MRI6	58.44553	60.15686	58.64433	60.26579	60.00096
MRI7	62.02096	63.35811	63.91447	63.65459	62.88492
MRI8	57.65364	57.96557	57.13291	57.85836	58.01849
MRI9	57.88365	58.74400	57.47039	58.60700	59.14482
PET1	67.78487	68.16501	67.64464	67.65847	66.20039

B

Anexo B

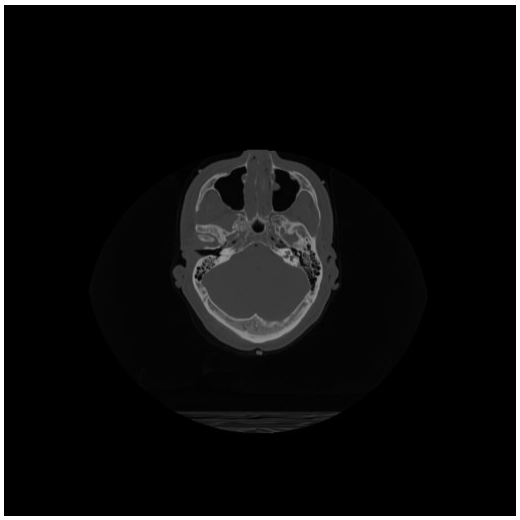


(a) Kernel 5x3 com aplicação de níveis de decomposição (4,4,2).

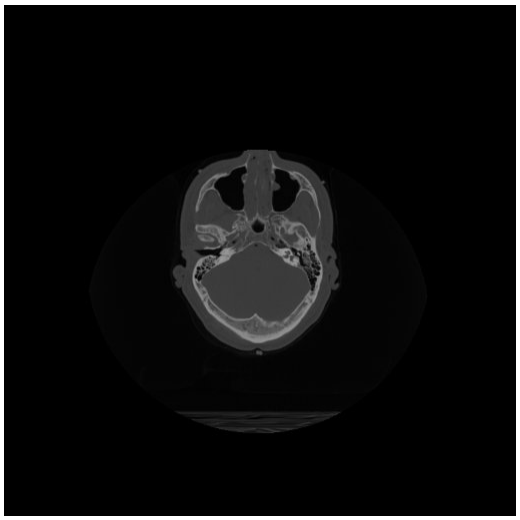


(b) Kernel 9x7 com aplicação de níveis de decomposição (4,4,2).

Figura B.1: Imagem volumétrica CT1 (*slice* 70).

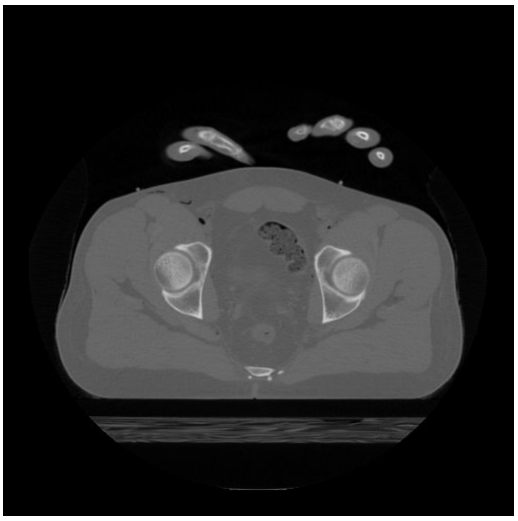


(a) Kernel 5x3 com aplicação de níveis de decomposição (4,4,2).

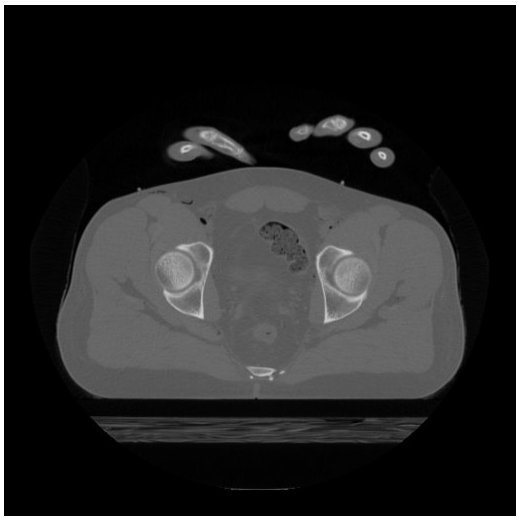


(b) Kernel 9x7 com aplicação de níveis de decomposição (4,4,2).

Figura B.2: Imagem volumétrica CT2 (*slice* 131).

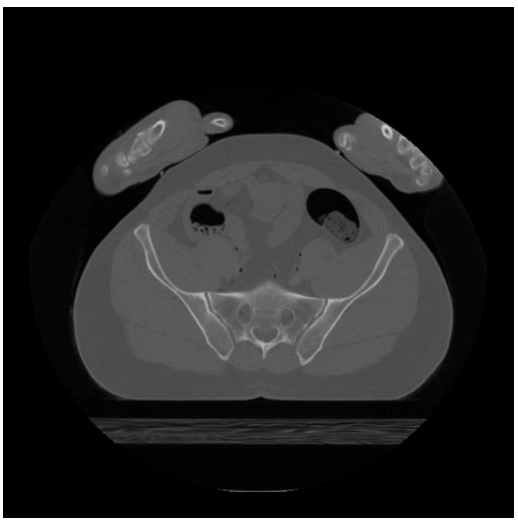


(a) Kernel 5x3 com aplicação de níveis de decomposição (4,4,2).

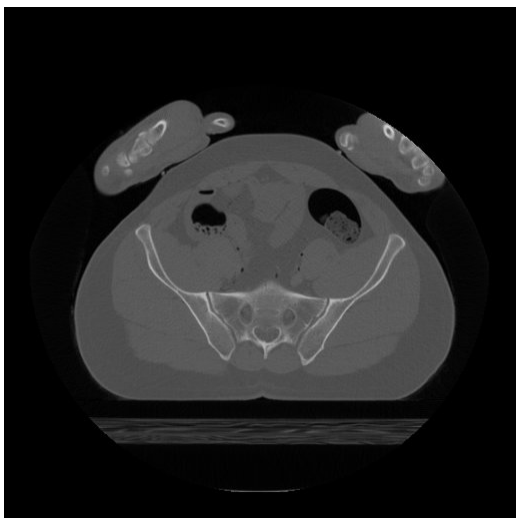


(b) Kernel 9x7 com aplicação de níveis de decomposição (4,4,2).

Figura B.3: Imagem volumétrica CT3 (*slice* 400).

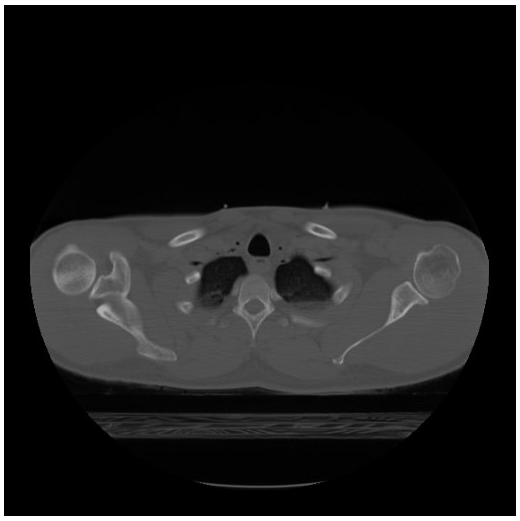


(a) Kernel 5x3 com aplicação de níveis de decomposição (4,4,2).

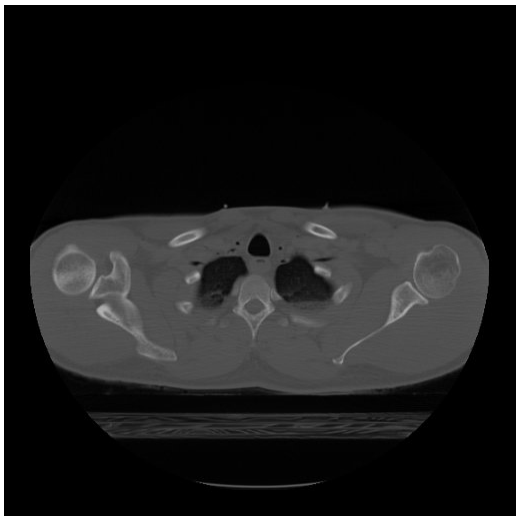


(b) Kernel 9x7 com aplicação de níveis de decomposição (4,4,2).

Figura B.4: Imagem volumétrica CT4 (*slice* 75).

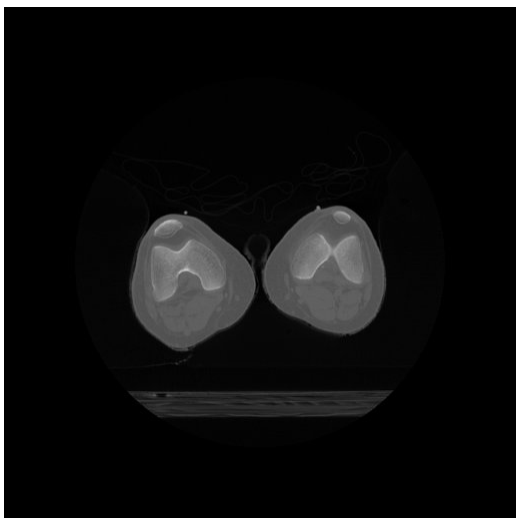


(a) Kernel 5x3 com aplicação de níveis de decomposição (4,4,2).

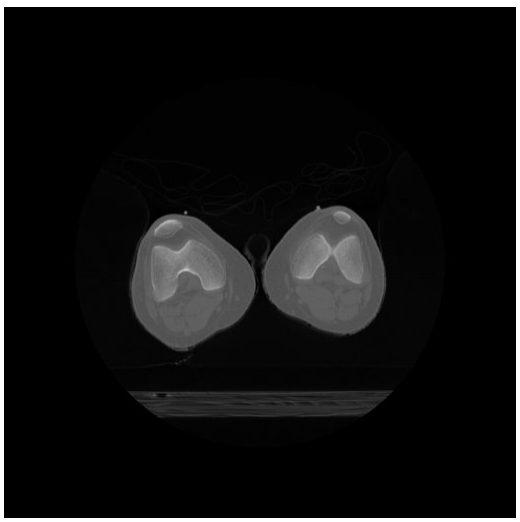


(b) Kernel 9x7 com aplicação de níveis de decomposição (4,4,2).

Figura B.5: Imagem volumétrica CT5 (*slice* 416).

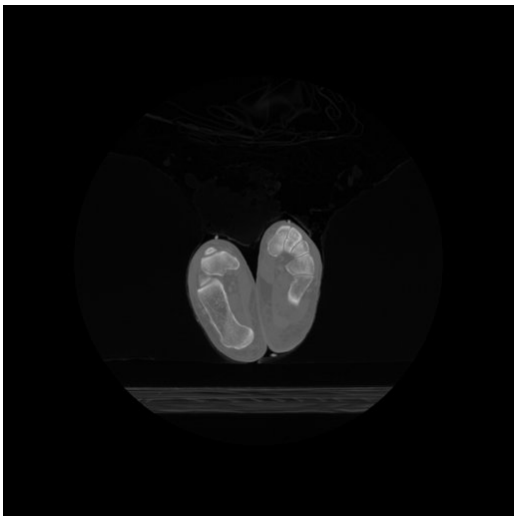


(a) Kernel 5x3 com aplicação de níveis de decomposição (4,4,2).

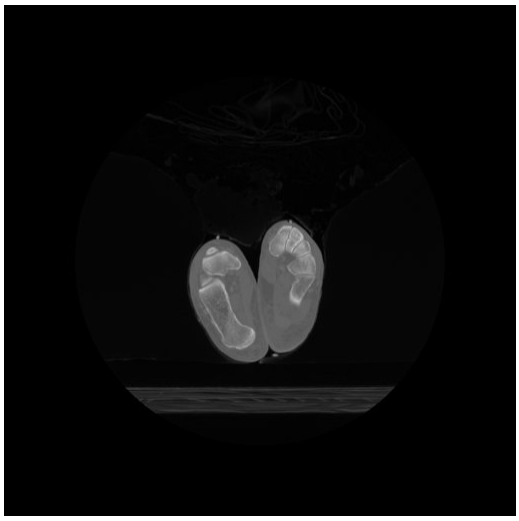


(b) Kernel 9x7 com aplicação de níveis de decomposição (4,4,2).

Figura B.6: Imagem volumétrica CT6 (*slice* 350).

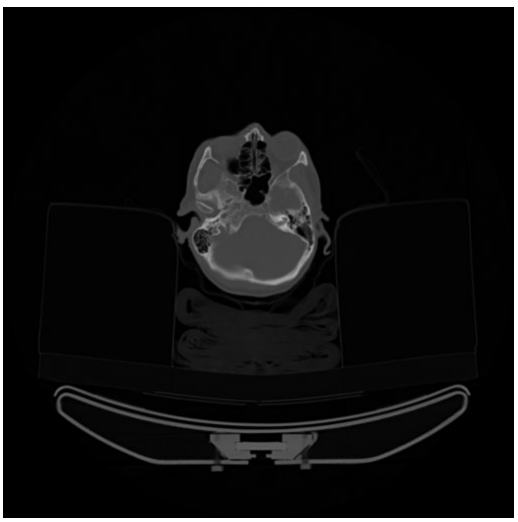


(a) Kernel 5x3 com aplicação de níveis de decomposição (4,4,2).

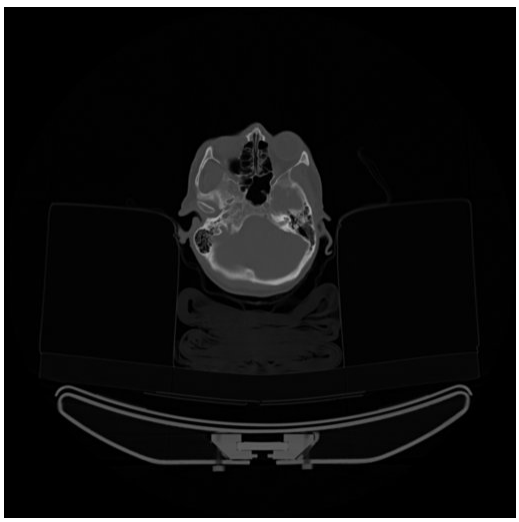


(b) Kernel 9x7 com aplicação de níveis de decomposição (4,4,2).

Figura B.7: Imagem volumétrica CT7 (*slice* 109).

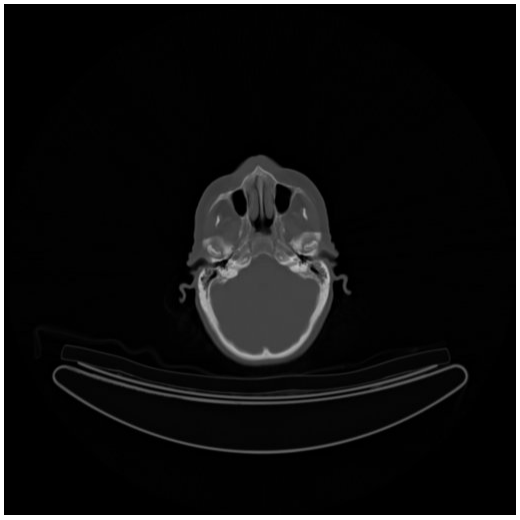


(a) Kernel 5x3 com aplicação de níveis de decomposição (4,4,2).

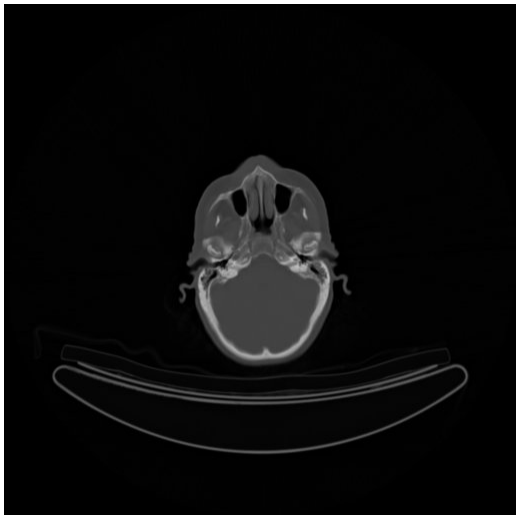


(b) Kernel 9x7 com aplicação de níveis de decomposição (4,4,2).

Figura B.8: Imagem volumétrica CT8 (*slice* 29).

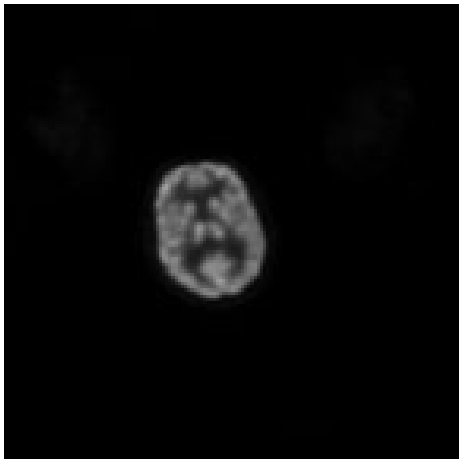


(a) Kernel 5x3 com aplicação de níveis de decomposição (4,4,2).

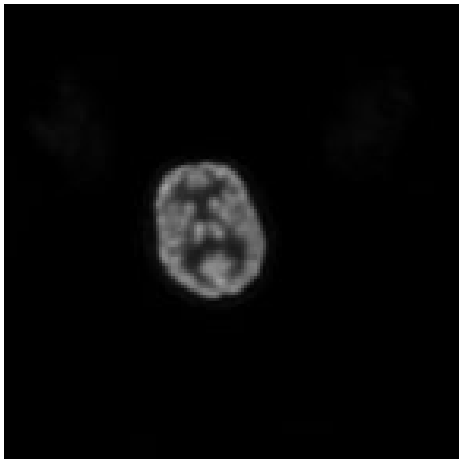


(b) Kernel 9x7 com aplicação de níveis de decomposição (4,4,2).

Figura B.9: Imagem volumétrica CT9 (*slice* 86).



(a) Kernel 5x3 com aplicação de níveis de decomposição (4,4,1).



(b) Kernel 9x7 com aplicação de níveis de decomposição (4,4,1).

Figura B.10: Imagem volumétrica PET1 (*slice* 235).