



Sérgio Alexandre Alves Mendes

Reconstrução de imagem médica de mamografia por emissão de positrões (PEM) com GPU

Dissertação para obtenção do Grau de Mestre em
Engenharia Biomédica

Orientador: Professor Doutor Pedro Almeida, FCUL
Co-orientador: Professor Doutor Nuno Matela, FCUL

Júri:

Presidente: Prof. Doutora Maria Adelaide de Almeida Pedro de Jesus
Arguente: Prof. Doutor Pedro Manuel Cardoso Vieira
Vogais: Prof. Doutor Pedro Miguel Dinis de Almeida
Prof. Doutor Nuno Miguel de Pinto Lobo e Matela



FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro de 2011

Copyright

Copyright©2011 - Todos os direitos reservados. Sérgio Alexandre Alves Mendes.
Faculdade de Ciências e Tecnologia. Universidade Nova de Lisboa.

A Faculdade de Ciências e Tecnologia e a Universidade Nova de Lisboa têm o direito, perpétuo e sem limites geográficos, de arquivar e publicar esta dissertação através de exemplares impressos reproduzidos em papel ou de forma digital, ou por qualquer outro meio conhecido ou que venha a ser inventado, e de a divulgar através de repositórios científicos e de admitir a sua cópia e distribuição com objectivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

Agradecimentos

Em primeiro lugar quero agradecer à minha família por me terem apoiado em todos os momentos na minha vida, sendo esta jornada académica apenas mais um deles. Por tudo o que significam para mim e porque quem sou a vocês muito o devo, muito obrigado Mãe, Pai, Avó e Mana. Um obrigado especial ao meu Avô que, onde quer que esteja, deverá estar orgulhoso.

De seguida, quero agradecer à Mafalda por tudo aquilo que passámos juntos ao longo destes cinco anos, toda a paciência que sempre tiveste nos momentos mais complicados do curso. Quero também que saibas que contigo foi tudo muito mais fácil de ultrapassar, graças ao teu amor e amizade, que resultaram num apoio constante e numa motivação ímpar para mim.

Quero também agradecer aos meus orientadores, ao Professor Pedro Almeida, pela possibilidade de realizar a tese de mestrado no IBEB e pela sua contagiante motivação e ao Professor Nuno Matela, pelo acompanhamento constante e disponibilidade na resolução de todos os problemas relacionados com a tese, bem como todas as explicações que envolviam a reconstrução de imagem.

Por fim, quero agradecer aos meus amigos destes cinco anos, mas que ficarão para a vida. A todos vocês agradeço por todos os momentos fantásticos que passámos, com a certeza que preencheram a minha vida de felicidade e que tornaram tudo inesquecível. Um obrigado a todos, Mafalda, Leo, Gémeo, Heavy, Nuno, Guida, Pedro, Ritinha, Sara, Cascalho, Hugo, Fernando, Frazão, Luís, Su e Su, Luís Ribeiro, Cátia, Pedro Mendes, Alentejano, etc.

Resumo

O cancro da mama é uma das principais causas de mortes entre as mulheres, sendo a detecção precoce desta patologia uma das áreas de maior interesse e desenvolvimento nos últimos anos.

O projecto Clear-PEM consistiu no desenvolvimento de um *scanner* baseado numa técnica tomográfica de medicina nuclear, designada por mamografia por emissão de positrões (PEM). Este *scanner* é constituído por duas cabeças detectoras que rodam em torno da mama, permitindo a detecção de radiação emitida do interior do corpo da paciente.

Este trabalho consistiu no desenvolvimento de dois algoritmos iterativos de reconstrução de imagem: Máxima Verosimilhança – Maximização da Expectativa (MLEM) e Subconjuntos Ordenados – Maximização da Expectativa (OSEM). O objectivo era recorrer às vantagens da computação em paralelo, através da programação em placas gráficas (GPU - *Graphics Processing Unit*), em oposição à programação mais tradicional que utiliza o processador do computador (CPU - *Central Processing Unit*). Deste modo, pretende-se minimizar a principal desvantagem dos algoritmos iterativos de reconstrução de imagem, quando comparados com soluções analíticas, que é o seu elevado tempo de computação.

Neste trabalho recorreu-se a uma placa gráfica (GPU) da NVIDIA®, tendo sido utilizada a CUDA™ para desenvolver os dois algoritmos de reconstrução.

Os dois algoritmos desenvolvidos (MLEM e OSEM) apresentam uma melhoria significativa em termos do tempo de computação recorrendo à programação em placas gráficas (GPU), aproximadamente 29 e 27 vezes inferior, respectivamente, em relação ao tempo necessário utilizando o processador do computador (CPU), sendo os resultados obtidos em termos de qualidade de imagem semelhantes.

Palavras-chave: Mamografia por Emissão de Positrões (PEM), Máxima Verosimilhança – Maximização da Expectativa (MLEM), Subconjuntos Ordenados – Maximização da Expectativa (OSEM), *Graphics Processing Unit* (GPU), *Compute Unified Device Architecture* (CUDA™)

Abstract

Breast cancer is one of the main causes of deaths among women, and early detection of this disease is one of the major areas of interest and development in recent years.

The Clear-PEM project was the development of a scanner based on a nuclear medicine tomographic technique, called positron emission mammography (PEM). This scanner consists of two detector heads that rotate around the breast, allowing the detection of radiation emitted from inside the body of the patient.

This work consisted in the development of two iterative algorithms for image reconstruction: Maximum Likelihood - Expectation Maximization (MLEM) and Ordered Subsets - Expectation Maximization (OSEM). The aim of this work was to use the advantages of parallel computing through the programming on graphics cards (GPU - Graphics Processing Unit), as opposed to more traditional programming that uses the computer's processor (CPU - Central Processing Unit). This is intended to minimize the main disadvantage of iterative algorithms for image reconstruction compared with analytical solutions, which is the high computation time.

In this work it was used a graphics card (GPU) from NVIDIA ® and CUDA™ to develop the two reconstruction algorithms.

The two developed algorithms (MLEM and OSEM) show a significant improvement in terms of computing time, using the programming on graphics cards (GPU), approximately 29 and 27 times lower, respectively, compared to the time required using the computer processor (CPU). The results obtained, in terms of image quality, are similar.

Keywords: Positron Emission Mammography (PEM), Maximum Likelihood - Expectation Maximization (MLEM), Ordered Subsets - Expectation Maximization (OSEM), Graphics Processing Unit (GPU), Compute Unified Device Architecture (CUDA™)

Índice

Agradecimentos.....	iii
Resumo.....	v
Abstract.....	vii
Índice.....	ix
Índice de Figuras.....	xi
Índice de Tabelas.....	xiii
Siglas e Acrónimos	xi
1 Introdução.....	1
2 Considerações Teóricas.....	5
2.1 Medicina Nuclear	5
2.2 Tomografia por emissão de positrões (PET).....	7
2.2.1 Princípios físicos da detecção e emissão de radiação em PET	7
2.3 Mamografia por emissão de positrões (PEM).....	10
2.3.1 Estado da arte em PEM.....	10
2.3.2 Projecto Clear-PEM.....	11
3 Reconstrução de imagem em PEM	13
3.1 Organização dos dados adquiridos	13
3.2 Reconstrução de imagem analítica	17
3.3 Reconstrução de imagem iterativa.....	18
3.3.1 Máxima Verosimilhança – Maximização da Expectativa (MLEM)	19
3.3.2 Subconjuntos Ordenados – Maximização da Expectativa (OSEM).....	20
4 Programação em paralelo.....	21
4.1 GPU (<i>Graphics Processing Unit</i>)	22
4.2 CUDA (<i>Compute Unified Device Architecture</i>)	24
5 Materiais e Método.....	27
5.1 <i>Hardware e Software</i>	27
5.2 Matriz de Sistema.....	28

5.3	Programação em CUDA	29
5.4	Biblioteca <i>Thrust</i>	31
5.5	Implementação do programa para OSEM.....	33
5.6	Planos ortogonais da mama	35
6	Resultados e Discussão	37
6.1	Fonte Pontual.....	38
6.1.1	Resolução espacial	38
6.1.2	Imagem reconstruída.....	42
6.2	Fantoma da mama.....	44
6.2.1	Homogeneidade	44
6.2.2	Razão sinal-ruído (SNR).....	45
6.2.3	Erro médio absoluto normalizado	47
6.2.4	Imagem reconstruída.....	47
6.3	Pré-clínico de um rato	49
6.3.1	Erro médio absoluto normalizado	49
6.3.2	Imagem reconstruída.....	49
6.4	Tempos de computação	51
7	Discussão Final e Conclusões	53
	Bibliografia	55

Índice de Figuras

<i>Figura 1.1 - Taxas anuais de incidência de cancro entre as mulheres e ajustada à idade[1].....</i>	<i>2</i>
<i>Figura 1.2 - Taxas anuais de morte devido a cancro entre as mulheres e ajustada à idade[1].....</i>	<i>2</i>
<i>Figura 2.1 - Decaimento por emissão de positrões[7]</i>	<i>8</i>
<i>Figura 2.2 - Aniquilação entre um positrão e um electrão, ocorrendo a emissão de dois fótons com a mesma direcção e sentidos opostos[7]</i>	<i>8</i>
<i>Figura 2.3 - LOR com e sem informação DOI.....</i>	<i>9</i>
<i>Figura 2.4 - Representação esquemática da aquisição Clear-PEM da mama[21].....</i>	<i>12</i>
<i>Figura 2.5 - Representação esquemática da aquisição Clear-PEM da zona axilar[21]</i>	<i>12</i>
<i>Figura 3.1 - Definição das coordenadas do linograma[23].....</i>	<i>14</i>
<i>Figura 3.2 - (A) Objecto formado por cinco fontes de actividade; (B) Linograma correspondente.....</i>	<i>16</i>
<i>Figura 4.1 - Performance de dois processadores GPU e um CPU[38]</i>	<i>23</i>
<i>Figura 4.2 - Esquema de execução de um programa em CUDA[39].....</i>	<i>25</i>
<i>Figura 4.3 - Esquema da hierarquia da memória que uma thread tem disponível em CUDA[39].....</i>	<i>26</i>
<i>Figura 5.1 - TOR a unir os dois detectores[23]</i>	<i>28</i>
<i>Figura 5.2 - Grelha de blocos de threads[39].....</i>	<i>29</i>
<i>Figura 5.6 - Planos sagital, coronal e transaxial numa imagem de corpo inteiro[47].</i>	<i>35</i>
<i>Figura 5.7 - Planos transaxial, sagital e coronal numa imagem de PEM[48]</i>	<i>36</i>
<i>Figura 6.1 - Perfis dos planos sagital, coronal e transaxial (da esquerda para a direita)</i>	<i>38</i>
<i>Figura 6.2 - Curva gaussiana ajustada ao perfil do plano sagital.....</i>	<i>39</i>
<i>Figura 6.3 - Resolução espacial sagital do algoritmo MLEM</i>	<i>39</i>
<i>Figura 6.4 - Resolução espacial coronal do algoritmo MLEM.....</i>	<i>40</i>
<i>Figura 6.5 - Resolução espacial transaxial do algoritmo MLEM</i>	<i>40</i>

<i>Figura 6.6 - Resolução espacial coronal do algoritmo OSEM.....</i>	<i>41</i>
<i>Figura 6.7 - Resolução espacial sagital do algoritmo OSEM</i>	<i>41</i>
<i>Figura 6.8 - Resolução espacial transaxial do algoritmo OSEM</i>	<i>42</i>
<i>Figura 6.9 - Plano coronal central da fonte pontual após 10 iterações (MLEM) com CPU (esquerda) e GPU (direita)</i>	<i>43</i>
<i>Figura 6.10 - Plano coronal central da fonte pontual após 4 iterações (OSEM) com CPU (esquerda) e GPU (direita)</i>	<i>43</i>
<i>Figura 6.11 – ROI escolhida para determinar o índice de homogeneidade</i>	<i>44</i>
<i>Figura 6.12 - Índice de homogeneidade ao longo do processo iterativo de MLEM.....</i>	<i>45</i>
<i>Figura 6.13 - Índice de homogeneidade ao longo do processo iterativo de OSEM</i>	<i>45</i>
<i>Figura 6.14 – ROI da lesão a azul e ROIs do fundo a vermelho</i>	<i>46</i>
<i>Figura 6.15 - Plano coronal central do fantoma de mama após 10 iterações (MLEM) com CPU esquerda) e GPU (direita)</i>	<i>47</i>
<i>Figura 6.16 - Plano coronal central do fantoma de mama após 4 iterações (OSEM) com CPU (esquerda) e GPU (direita).....</i>	<i>48</i>
<i>6.17 - Plano coronal central do pré-clínico de um rato após 10 iterações (MLEM) com CPU (esquerda) e GPU (direita)</i>	<i>49</i>
<i>Figura 6.18 - Plano coronal central do pré-clínico de um rato após 4 iterações (OSEM) com CPU (esquerda) e GPU (direita).....</i>	<i>50</i>

Índice de Tabelas

<i>Tabela 5.1 - Função <code>order_by_key</code> (ordenação por chave) da biblioteca Thrust</i>	<i>32</i>
<i>Tabela 5.2 - Função <code>reduction_by_key</code> (redução por chave) da biblioteca Thrust.....</i>	<i>32</i>
<i>Tabela 6.1 - Tempos de computação para os algoritmos MLEM e OSEM com CPU e GPU</i>	<i>51</i>

Siglas e Acrónimos

APD	Fotodíodo de Avalanche
ART	Técnica de Reconstrução Algébrica
CT	Tomografia Computorizada
CPU	Do inglês <i>Central Processing Unit</i>
CUDA	Do inglês <i>Compute Unified Device Architecture</i>
DOI	Profundidade de Interação
FBP	Retro projecção Filtrada
GPGPU	Do inglês <i>General-purpose computing on graphics processing units</i>
GPU	Do inglês <i>Graphics Processing Unit</i>
IH	Índice de Homogeneização
LOR	Linha de Resposta
MLEM	Máxima Verosimilhança – Maximização da Expectativa
NME	Erro Médio Absoluto Normalizado
OSEM	Subconjuntos Ordenados – Maximização da Expectativa
PEM	Mamografia por Emissão de Positrões
PET	Tomografia por Emissão de Positrões
ROI	Região de Interesse
SNR	Razão Sinal-Ruído
SPECT	Tomografia Computorizada por Emissão de Fóton Único
TOR	Tubo de Resposta

1 Introdução

O cancro da mama é a neoplasia maligna com maior taxa de incidência nas mulheres (Figura 1.1), sendo a segunda maior causa de morte por cancro (Figura 1.2), nos Estados Unidos da América, sendo apenas suplantado pelo cancro do pulmão[1]. A dimensão deste problema para a saúde pode ser facilmente entendida quando se espera que, nos Estados Unidos da América, uma em cada oito mulheres desenvolva cancro da mama durante a sua vida.

Nos últimos anos têm sido realizados esforços no sentido de se desenvolverem novas técnicas imagiológicas que permitam uma detecção cada vez mais precoce do cancro da mama e a possibilidade de se saber caracterizar o tipo de cancro cada vez mais cedo, de modo a que se possa proceder à melhor terapêutica possível para cada caso.

A detecção precoce do cancro da mama é uma das razões para a diminuição das taxas de mortalidade desta doença, juntamente com a descoberta de novas técnicas e o aperfeiçoamento das já existentes. Deste modo, as novas técnicas centram os seus esforços em detectar lesões de dimensões cada vez mais reduzidas, possibilitando o tratamento da patologia num estágio de desenvolvimento mais precoce, aumentando a possibilidade de sucesso deste. As novas técnicas também visam ultrapassar as dificuldades sentidas pelas técnicas mais antigas em analisar, por exemplo, mamas densas, ou até detectar lesões em zonas anteriormente impossíveis, como a zona da axila. Desta forma, uma técnica que pode vir a desempenhar um

papel fulcral é a mamografia por emissão de positrões (PEM), uma técnica imagiológica funcional e não-invasiva, que se dedica à imagem da mama, partilhando os seus princípios físicos com a tomografia por emissão de positrões (PET).

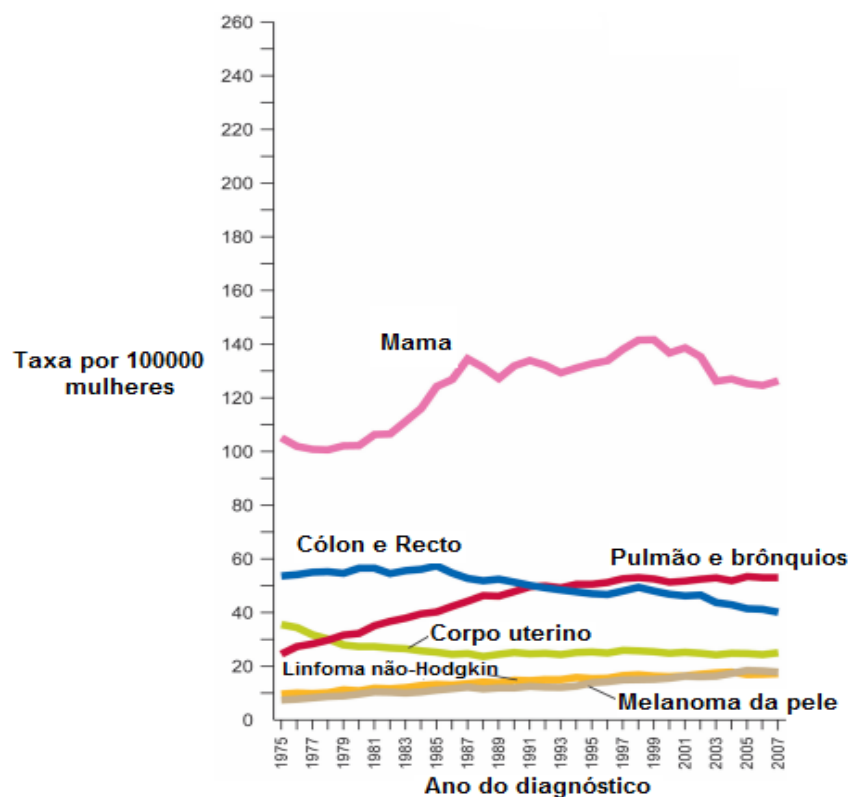


Figura 1.1 - Taxas anuais de incidência de cancro entre as mulheres e ajustada à idade[1]

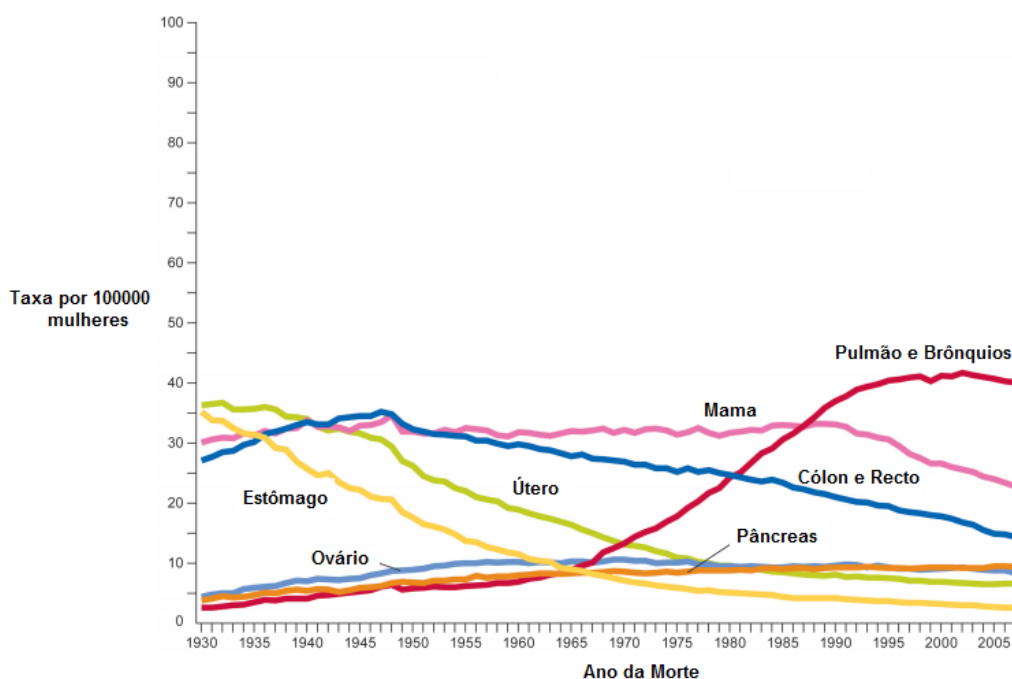


Figura 1.2 - Taxas anuais de morte devido a cancro entre as mulheres e ajustada à idade[1]

As técnicas imagiológicas que estudam o cancro da mama estão divididas em duas categorias principais: anatómicas e funcionais. As técnicas predominantemente anatómicas são, por exemplo, a mamografia por raio-X, a imagem de ressonância magnética ou a ultrasonografia, que obtêm imagens da estrutura ou de mudanças anatómicas que se encontram associadas a uma patologia por descobrir. Na área da imagiologia funcional, para além da PET e PEM referidas anteriormente, temos a tomografia computadorizada por emissão de fóton único (SPECT), técnicas onde a imagiologia captura a fisiologia do corpo ou mudanças funcionais e metabólicas associadas a uma patologia[2].

O projecto Clear-PEM é um consórcio de vários grupos de investigação que desenvolveu um *scanner* de PEM. Este scanner é constituído por duas cabeças detectoras que rodam em torno da mama, permitindo a detecção de radiação emitida do interior do corpo da paciente, na zona da mama ou axila consoante o exame que está a ser realizado.

O objectivo deste trabalho era acelerar dois algoritmos iterativos de reconstrução de imagem: Máxima Verosimilhança – Maximização da Expectativa (MLEM) e Subconjuntos Ordenados – Maximização da Expectativa (OSEM). Para que o tempo de computação fosse mais baixo do que aquele já praticado era necessário recorrer às vantagens da computação em paralelo, através da programação em placas gráficas (GPU), em oposição à programação mais tradicional que utiliza o processador do computador (CPU). Deste modo, seria possível ultrapassar o principal problema dos algoritmos iterativos de reconstrução de imagem, quando comparados com algoritmos analíticos, que é o seu elevado tempo de computação. No final da reconstrução para além do tempo de computação, era necessário garantir que as imagens reconstruídas neste trabalho eram similares às reconstruídas pelo *software* que já se encontrava desenvolvido em linguagem C.

Esta tese encontra-se organizada em sete capítulos principais, sendo esta introdução o primeiro deles. No capítulo 2, serão apresentados os conceitos teóricos que suportam este trabalho. Primeiramente, serão apresentadas as técnicas de imagem de medicina nuclear. De seguida, focar-nos-emos na técnica de PET e nos seus princípios físicos de detecção e emissão da radiação. No fim do capítulo, será apresentada a técnica de PEM e uma apresentação breve de alguns grupos de investigação que desenvolvem sistemas de PEM. Depois disto, será apresentado o Projecto Clear-PEM, salientando as especificidades do scanner desenvolvido e os objectivos do projecto.

No capítulo 3, será abordada a reconstrução de imagem em PEM, começando pelo modo como se organiza os dados adquiridos. Seguidamente, serão explicadas as diferenças entre os algoritmos de reconstrução diferenciando os analíticos dos iterativos, ilustrando as suas características e as vantagens e desvantagens.

No capítulo 4 será desenvolvido o tema da programação em paralelo. Inicialmente, mostrando a evolução dos GPUs (*Graphics Processing Unit*) até ao dia de hoje e de, de seguida, dando a conhecer a CUDA. No capítulo 5 serão apresentados os materiais utilizados neste trabalho como o hardware ou software. Voltamos, logo de seguida, à programação em CUDA, onde será apresentado um código onde são se executa a soma de dois vectores. Depois, será apresentada a biblioteca *Thrust* que desempenhou um papel fundamental neste trabalho. Ainda no capítulo 5, será apresentada uma rotina de cada um dos algoritmos de iteração MLEM e OSEM: Por fim serão apresentados os planos ortogonais da mama e os métodos de avaliação e comparação das imagens. No capítulo 6 serão apresentados todos os resultados obtidos com os dois algoritmos, tanto para a programação em GPU como para a programação recorrendo somente ao CPU, *software* desenvolvido em linguagem C. Os resultados foram obtidos com dados adquiridos com uma fonte pontual, obtidos por simulação com um fantoma de mama e com dados de um ensaio pré-clínico, onde foi analisado um rato. Também nesse capítulo serão feitas análises qualitativas das imagens, bem como aos tempos de computação.

Para finalizar, no capítulo 7 serão apresentadas as conclusões e será feita uma discussão global de todo o trabalho.

2 Considerações Teóricas

2.1 Medicina Nuclear

As técnicas de imagem de medicina nuclear permitem obter informação clínica que resulta da distribuição de radiofármaco administrado ao paciente. Este radiofármaco possui um radionuclido e um fármaco constituído por moléculas orgânicas que apresentam uma fixação preferencial num determinado tecido ou órgão, em função de uma determinada função química.

As propriedades físicas de um isótopo são fundamentais para o seu potencial de utilização em medicina nuclear. Um isótopo deve apresentar um tempo de decaimento que não se prolongue excessivamente para lá da duração do exame, mas que possa permitir a realização do mesmo sem qualquer falha. A energia da radiação emitida deve também situar-se entre um intervalo de valores que permita a sua detecção sem que exista o risco de se confundir com outra radiação secundária, mas também não deve ser excessiva para uma utilização diagnóstica, uma vez que a radiação ionizante não é inócua. Quanto mais fácil a produção do isótopo, mais baixo será o seu preço.

A medicina nuclear permite conhecer as funções bioquímicas do corpo humano, contrariamente a outras técnicas de imagem, cujo principal objectivo é a observação anatómica, por exemplo, a mamografia de raio-X.

As técnicas de medicina nuclear mais utilizadas são a cintigrafia planar convencional, e as técnicas tomográficas, a tomografia por emissão de fóton simples (SPECT) e a tomografia por emissão de positrões (PET). Nos últimos anos, as técnicas de imagem tomográfica têm vindo a superiorizar-se à aproximação convencional de imagem planar, em relação à qualidade das imagens obtidas[3]. Relativamente às técnicas tomográficas, a SPECT apresenta contraste e resoluções espacial e temporal, inferiores à PET[4].

2.2 Tomografia por emissão de positrões (PET)

A PET surgiu como uma técnica de imagem com um futuro muito promissor com aplicação nas áreas de oncologia, cardiologia e neurologia. Esta técnica baseia-se na relação entre a composição química dos radioisótopos emissores de positrões e os princípios físicos que estão ligados ao decaimento radioactivo.

A PET é uma técnica com custos elevados ao nível do equipamento e da sua operacionalidade, exigindo uma infra-estrutura sofisticada para a produção do radiofármaco. Esta é a principal razão para que não exista um aproveitamento ainda maior desta técnica em toda a prática clínica.

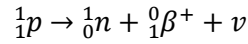
Os isótopos emissores de positrões existem em vários elementos químicos presentes nas moléculas orgânicas do corpo humano (^{11}C , ^{13}N , ^{15}O , ^{18}F). Destes o ^{18}F é o mais utilizado e apresenta um tempo de semi-vida de, aproximadamente, 110 minutos. O tempo de semi-vida dos outros isótopos é de 20, 10 e 2 minutos, respectivamente. O tempo de semi-vida de um radioisótopo, é o tempo necessário para que este diminua para metade da sua quantidade original[5]. Por isso é necessário um certo tempo de semi-vida que permita a realização do exame sem qualquer problema de desintegração do radioisótopo.

Um tumor é um crescimento anormal de células, quer seja um tumor maligno ou benigno. Nos dois casos ocorrem taxas elevadas de metabolismo, quando comparadas com as das células vizinhas saudáveis. Este facto cria excelentes condições para a imagiologia funcional. Assim, sabe-se que o consumo de glicose será elevado nas células tumorais. Deste modo, substitui-se a glicose por um análogo, a deoxiglicose, que não é consumido pelo organismo, ficando aprisionado no interior das células. Por fim, podemos referir que o radiomarcador oncológico mais comum em PET é $[^{18}\text{F}]2\text{-deoxi-2-fluoro-d-glucose}$ ($[^{18}\text{F}]\text{FDG}$).

2.2.1 Princípios físicos da detecção e emissão de radiação em PET

O princípio básico em que se apoia a tomografia por emissão de positrões é o decaimento por emissão de positrões (decaimento β^+). Este fenómeno ocorre quando um radionuclido é rico em protões, existindo duas formas possíveis de decaimento: decaimento por emissão de positrões ou por captura electrónica, podendo esta ser desprezada por apresentar uma probabilidade de ocorrência baixa para os radiofármacos mais utilizados[6]. A captura electrónica não permite construir uma imagem PET. Um isótopo só poderá decair por emissão de positrões caso possua mais de 1.022 MeV de energia, do que o isótopo para o qual decai[6].

A reacção de decaimento por emissão de positrões é apresentada na Eq. 2.1, onde se verifica que o protão é convertido num neutrão, com emissão de um positrão e um neutrino. A reacção encontra-se esquematizada na figura Figura 2.1.



Eq. 2.1

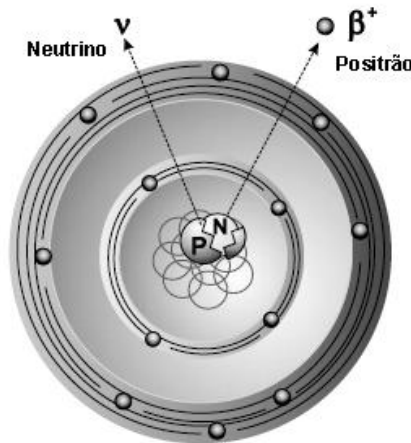


Figura 2.1 - Decaimento por emissão de positrões[7]

Depois de ocorrer a emissão de positrões, estes tendem a interagir quase instantaneamente com a matéria ocorrendo reacções de aniquilação positrão-electrão, sendo emitidos dois fótons de 511 keV de energia (fótons γ), formando um ângulo de, aproximadamente, 180° entre si (Figura 2.2). Num exame de PET, o paciente está rodeado por um anel ou uma matriz de cristais de cintilação, a cadeia de detectores. No caso dos fótons não serem absorvidos pelos tecidos do corpo, estes podem ser absorvidos pelos cristais de cintilação. Assim, cada sinal luminoso que se forme nos cristais de cintilação será transformado em sinal eléctrico. No momento em que dois sinais eléctricos activem, dentro de uma janela bastante curta de tempo, um circuito electrónico de coincidências, será registado um evento de coincidência.

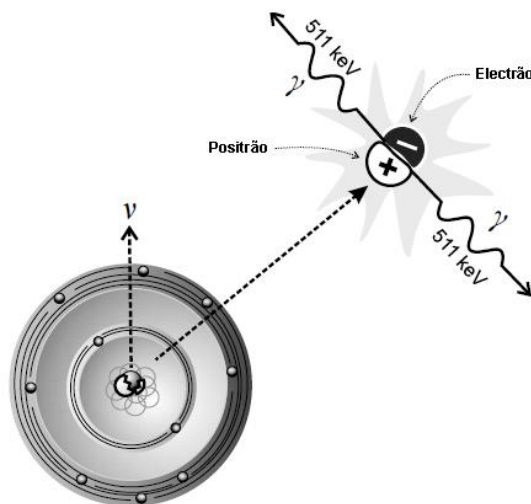


Figura 2.2 - Aniquilação entre um positrão e um electrão, ocorrendo a emissão de dois fótons com a mesma direcção e sentidos opostos[7]

A linha de resposta (LOR) é a linha que une os dois cristais onde a interacção ocorreu. Esta linha representa o conjunto de localizações possíveis para a ocorrência da aniquilação.

Como o alcance do positrão é bastante curto, pode considerar-se que o ponto de aniquilação corresponde, aproximadamente, ao ponto de emissão do positrão, sendo este o local da ocorrência da função fisiológica que se pretende ilustrar [6, 8]. Recorrendo à reconstrução de imagem será possível inferir o ponto em que a aniquilação teve lugar (ver secção 3.3).

Os cristais de cintilação têm um comprimento na ordem dos 10-20 mm, de modo a maximizarem a probabilidade de absorção dos fótons de 511 keV. A interacção da radiação com o cristal pode ocorrer em qualquer ponto deste. Se a profundidade de interacção (DOI) não puder ser calculada, a LOR deve ser considerada a linha que une os centros das superfícies dos cristais que se encontram direccionadas para dentro da câmara, mas será impossível garantir que a aniquilação se deu ao longo da LOR (Figura 2.3). Esta característica do sistema (detecção da DOI) é mais importante no caso de câmaras de tamanhos reduzidos, onde o objectivo fundamental é uma boa resolução espacial.

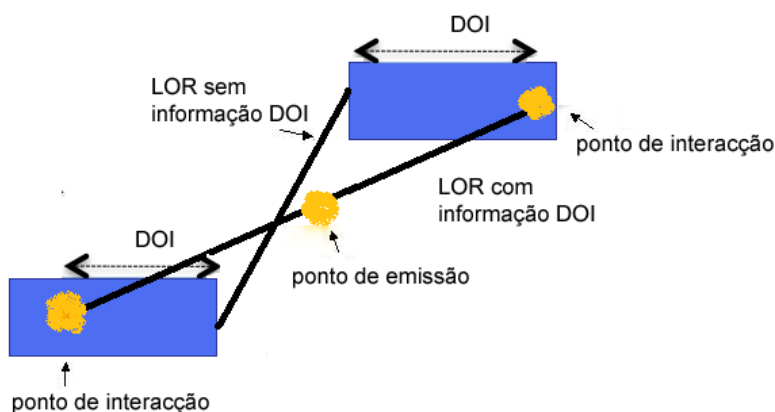


Figura 2.3 - LOR com e sem informação DOI

2.3 Mamografia por emissão de positrões (PEM)

A mamografia por emissão de positrões (PEM) partilha os princípios básicos de PET e é uma técnica de imagem funcional especificamente concebida para a detecção de cancro da mama. Esta técnica surge da necessidade de melhorar a sensibilidade e resolução espacial em relação aos exames de PET de corpo inteiro. Com o aumento da sensibilidade seria possível uma diminuição da dose fornecida ao paciente e um período de exame mais curto[9]. A solução para aumentar a capacidade de detecção teria de passar pela utilização de pequenos detectores muito próximos da mama, sendo possível limitar a área em estudo, utilizando cristais detectores de menores dimensões e mantendo uma produção de baixo custo. Como as matrizes detectoras poderiam estar dispostas de um modo planar ou circular, a geometria das câmaras poderiam variar bastante. Caso a opção fosse uma geometria planar, o sistema poderia ser constituído por duas ou mais placas detectoras. No caso de serem apenas duas, ainda havia a possibilidade de serem estáticas ou se poderem movimentar em torno da mama; se fossem mais do que duas, poderia ter uma geometria rectangular ou poligonal, semelhante a uma configuração em anel.

A escolha de todos estes parâmetros deveria e deve ser efectuada tendo em conta um compromisso entre a flexibilidade de adaptação do detector à estatura da mama e região axilar, a cobertura angular da câmara PEM e o próprio custo associado.

2.3.1 Estado da arte em PEM

Actualmente existem diversos grupos de pesquisa que se encontram a desenvolver projectos inteiramente dedicados a equipamentos de PEM. Podem observar-se diferenças essencialmente na geometria das câmaras, na possibilidade de rodar em torno da mama, de analisar a zona da axila, nos cristais de cintilação utilizados, na capacidade de detectarem a DOI, na forma de conversão da radiação electromagnética cintilante em sinais eléctricos ou nos algoritmos de reconstrução de imagem utilizados.

O conceito de Mamografia por Emissão de Positrões foi primeiramente introduzido, em 1994, por C.J. Thomson *et al*, da Universidade McGill de Montreal[10]. A câmara PEM desenvolvida, foi desenhada com o objectivo da sua integração numa unidade de mamografia convencional, podendo obter-se imagens radiográficas da mama, para além de imagens por emissão de positrões. Os primeiros resultados obtidos confirmaram as expectativas de que a PEM seria uma tecnologia promissora para a detecção do cancro da mama, devido a uma eficiência e uma resolução espacial elevada[11].

A primeira câmara de PEM a ser comercializada foi desenvolvida pela *PEM Technologies Inc*, actualmente *Naviscan PET Systems*. A pesquisa que levou à criação desta

câmara começou em *National Institute of Health* em Bethesda e estava protegida por uma patente¹. Só quando a tecnologia passou para a *Naviscan PET Systems* é que começaram a publicar, pois até então só haviam tido algumas comunicações em conferências. A ideia deste projecto era montar um detector de raios gama numa unidade convencional de mamografia de raio-X e biópsia[12]. Em 2003, apresentaram a sua primeira versão comercial de uma máquina de PEM denominada PEM-2400.

Murthy e seus colaboradores demonstraram as vantagens de uma PET combinada com um sistema de mamografia em 14 pacientes [13]. Outros grupos também desenvolveram a ideia de uma plataforma com a dupla modalidade anatómica/funcional para a imagiologia do cancro da mama através da incorporação de geometrias alternativas para a cadeia de detectores de PET[14, 15] ou propondo até a incorporação de mesas para a realização de biópsias[16].

Outros equipamentos PEM foram sendo desenvolvidos, como o sistema *maxPET* [17] e o sistema *LBNL-PEM* [18, 19] que apresenta uma geometria rectangular, formado por quatro placas detectoras em torno da mama e possuindo a capacidade de medir a DOI.

2.3.2 Projecto Clear-PEM

Foi criado em 2002 o consórcio PET – Mammography com o objectivo de desenvolver uma câmara de Mamografia por Emissão de Positrões. Este consórcio é formado por sete instituições que trabalham em áreas diversas como física de partículas, biofísica, medicina, engenharia médica, computação e engenharia mecânica. O Consórcio Clear-PEM procura desenvolver um sistema baseado na tecnologia PET que possibilite a detecção de cancro da mama nos seus estágios iniciais, fazendo uma análise da mama e axila, recorrendo ao radiomarcador [¹⁸F]FDG. A câmara que está a ser desenvolvida tem como principal objectivo a detecção de tumores da mama com dimensões menores do que 2 mm[20].

A câmara desenvolvida tem o nome de Clear-PEM, consistindo em duas cabeças planares de detecção, formadas por cristais LYSO:Ce acoplados com matrizes de APDs, ligadas a um braço rotatório, permitindo efectuar exames rodando em torno da mama (Figura 2.3) e na zona axilar (Figura 2.4). As duas placas detectoras têm aproximadamente 16,5x14,5 cm² [21] e a distância entre ambas é ajustada ao tamanho da mama, não havendo compressão do peito. Cada cristal tem as dimensões de 2x2x20 mm³ e apresenta dois APDs (Fotodíodos de Avalanche) acoplados, um em cada extremidade. São estes dois APDs que permitem retirar informação sobre a DOI.

¹ U.S. Patent 5252830

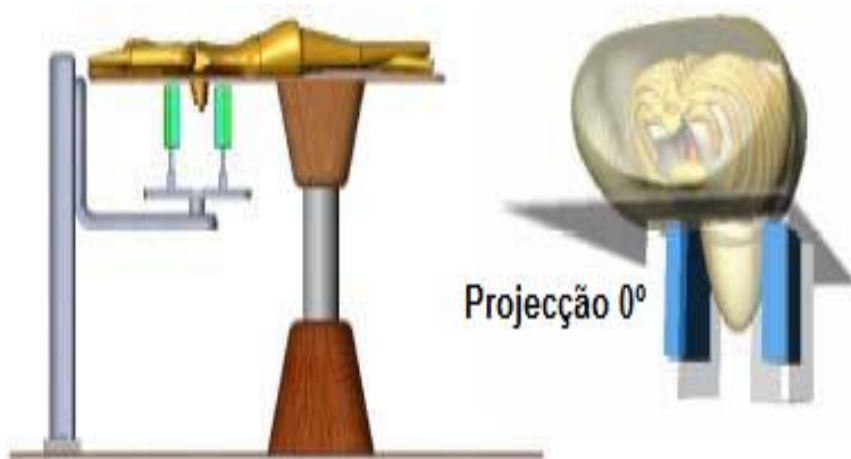


Figura 2.4 - Representação esquemática da aquisição Clear-PEM da mama[21]

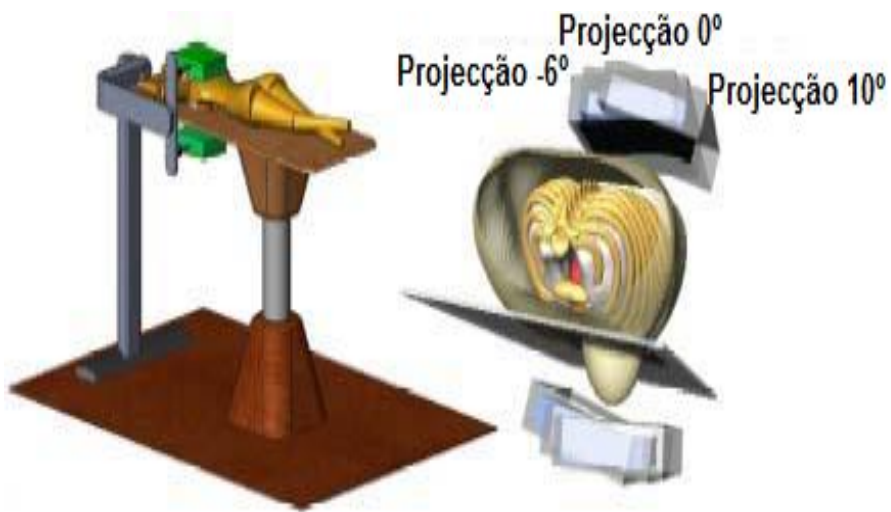


Figura 2.5 - Representação esquemática da aquisição Clear-PEM da zona axilar[21]

3 Reconstrução de imagem em PEM

Até recentemente a reconstrução de imagem em PEM recorria, normalmente, a aproximações analíticas provenientes da Tomografia Computorizada (CT). A reconstrução permite adquirir imagens tomográficas a partir de diferentes projecções adquiridas da distribuição de actividade. Nos últimos anos, devido ao enorme desenvolvimento nos processadores e até a optimizações nos próprios algoritmos, a reconstrução recorrendo a métodos iterativos tem ganho bastante importância e são cada vez mais utilizadas, substituindo os métodos analíticos[22].

Neste capítulo será apresentado o modo como os dados adquiridos são organizados, seguido de possíveis métodos de reconstrução de imagem em PEM. Dos tipos de algoritmos de reconstrução será dado um maior ênfase aos algoritmos estatísticos iterativos, MLEM e OSEM, que foram implementados neste trabalho.

3.1 Organização dos dados adquiridos

As primeiras etapas no processo de reconstrução de imagem são a organização e parametrização do conjunto de dados adquiridos num exame de PEM. Esta organização consiste em agrupar LORs similares de acordo com as suas características geométricas. Apesar de, com

este processo, se perder alguma informação, a reconstrução de imagem será acelerada de forma significativa, sem relevante degradação da imagem, especialmente quando o número de coincidências detectadas é muito superior ao de possíveis LORs.

A parametrização normalmente utilizada resulta no agrupamento dos dados num sinograma, sendo este um histograma obtido recorrendo a coordenadas polares. Outra possível histogramização dos dados é recorrendo a linogramas, que é explorada por N.Matela, na reconstrução de imagem com os dados adquiridos com o *scanner* Clear-PEM[23]. Os resultados apresentados neste trabalho tiveram origem em dados organizados em linogramas, que para geometrias de aquisição planares apresentam vantagens em relação aos sinogramas[24].

Os métodos de reconstrução de imagem incluem sempre dois processos principais de cálculo: a projecção e a retro projecção (ver secção 3.3). No passo de retro projecção é calculado o valor do integral ao longo das LORs que atravessam um certo voxel da imagem. Caso os dados tenham sido parametrizados com coordenadas polares, o integral é obtido integrando os valores do sinograma ao longo de uma sinusóide. Edholm e Herman[25] chegaram à conclusão que este integral seria calculado mais facilmente se fosse realizado ao longo de uma linha recta, ao invés de uma sinusóide. Para ser possível efectuar este integral, seria necessária uma nova organização e parametrização dos dados. Foi dado o nome de linograma a esta nova representação, que se baseia no facto que o conjunto de LORs que atravessam um certo pixel é representado por uma linha recta.

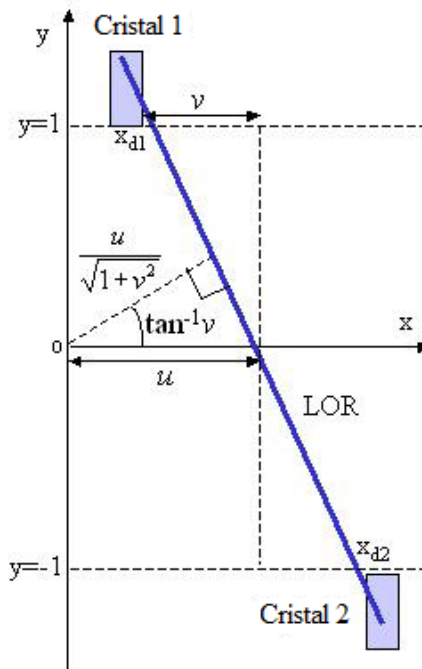


Figura 3.1 - Definição das coordenadas do linograma[23]

De acordo com a Figura 3.1 é possível parametrizar cada uma das LORs com as coordenadas (u, v) , definidas pelas Eq. 3.1 e Eq. 3.2.

$$u = \frac{x_{d2} + x_{d1}}{2} \quad \text{Eq. 3.1}$$

$$v = \frac{x_{d2} - x_{d1}}{2} \quad \text{Eq. 3.2}$$

Como o conjunto de LORs que atravessam um certo pixel (x, y) é representado por uma linha recta, podemos representar como na Eq. 3.3.

$$y = mx + b \quad \text{Eq. 3.3}$$

Sendo m o declive da LOR, podemos relacioná-lo com a coordenada do linograma v na Eq. 3.4.

$$m = \frac{y_{d2} - y_{d1}}{x_{d2} - x_{d1}} = \frac{-2}{2v} = -\frac{1}{v} \quad \text{Eq. 3.4}$$

Substituindo a Eq. 3.4 na Eq. 3.3, obtém-se a Eq. 3.5.

$$y = -\frac{x}{v} + b \quad \text{Eq. 3.5}$$

Considerando a definição de u , quando a Eq. 3.6 é verificada, b é dado pela Eq. 3.7.

$$y = 0 \Rightarrow x = u \quad \text{Eq. 3.6}$$

$$b = \frac{u}{v} \quad \text{Eq. 3.7}$$

Por fim, cada conjunto de LORs que atravessa um certo pixel (x, y) deve obedecer à Eq. 3.8.

$$y = -\frac{x}{v} + \frac{u}{v} \Leftrightarrow u = x + yv \quad \text{Eq. 3.8}$$

Esta característica pode ser observada no linograma apresentado na Figura 3.2 - (A) Objecto formado por cinco fontes de actividade; (B) Linograma correspondente, onde se observa um segmento de recta para cada fonte de actividade presente no objecto representado na Figura 3.2. Cada segmento de recta corresponde ao conjunto de LORs que atravessa cada fonte de actividade.

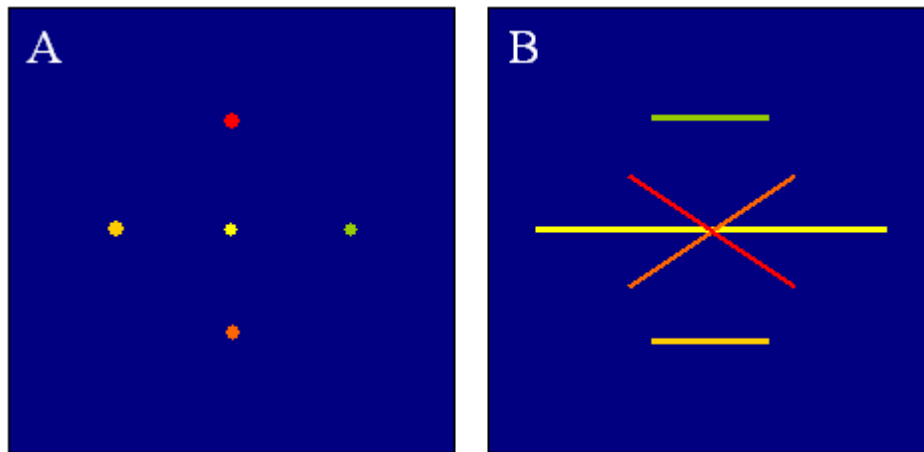


Figura 3.2 - (A) Objecto formado por cinco fontes de actividade; (B) Linograma correspondente

A única limitação na utilização desta parametrização é a impossibilidade de representar uma LOR paralela ao eixo-xx, onde v é indeterminado, que pode ocorrer quando os detectores têm a capacidade de rodar à volta do objecto. No caso do *scanner* Clear-PEM associa-se sempre o eixo-xx aos detectores, criando um linograma para cada posição de aquisição.

3.2 Reconstrução de imagem analítica

Os processos de reconstrução analítica assumem modelos de emissão e detecção bastante simples, que à medida que se tenta torná-los rigorosos, os algoritmos tornam-se demasiado complexos para que seja exequível a sua implementação [26]. Por outro lado, é difícil adicionar conhecimento prévio das estruturas em estudo, nomeadamente informação anatómica.

A retro projecção filtrada (FBP) é um método de reconstrução analítico e aquele que mais utilizado tem sido em prática clínica. Este facto deve-se, essencialmente, ao seu reduzido tempo de computação, principalmente quando comparado com processos iterativos.

A FBP apresenta várias limitações devido à presença de artefactos (riscas), que se tentam ultrapassar recorrendo a filtros que vão efectuar uma valorização relativa das altas frequências, potenciando um aumento do ruído estatístico, que é mais significativo nas frequências mais elevadas. Deste modo, é necessário um compromisso entre a maximização da resolução espacial e a minimização do ruído na imagem final reconstruída. Mais detalhes sobre reconstrução analítica de imagem podem ser encontrados em [27].

3.3 Reconstrução de imagem iterativa

Os algoritmos iterativos de reconstrução de imagem têm sido cada vez mais utilizados nos últimos anos devido, essencialmente, ao desenvolvimento de processadores cada vez mais potentes e, mais recentemente, à utilização cada vez mais global do GPGPU (*General-purpose computing on graphics processing units*). Estes progressos permitiram a criação de algoritmos com tempos de execução que conseguem competir com os tempos de algoritmos analíticos[28-30].

Um algoritmo iterativo de reconstrução de imagem é um procedimento gradual que determina a estimativa possível mais próxima da distribuição de actividade real, produzida pelos dados adquiridos. Este algoritmo deve realizar estimativas sucessivas que deverão convergir assintótica e monotonamente para a solução do problema, ou seja, cada estimativa produzida numa determinada iteração, deverá ser uma estimativa melhorada da distribuição de actividade real.

De modo a obter resultados realistas é fundamental que os algoritmos apresentem os seguintes componentes[23, 31, 32]:

1. Um modelo para os dados adquiridos;
2. Um modelo para a imagem de distribuição de actividade;
3. Um modelo físico para calcular as projecções esperadas da estimativa de distribuição de actividade (*matriz de sistema*);
4. Um método para avaliar as diferenças entre as medidas reais e esperadas, através de uma *função objectivo*;
5. Um método para minimizar essas diferenças, através da optimização da *função objectivo*.

A *matriz de sistema*, A , é um modelo dos processos de emissão e detecção e os seus elementos são dados por a_{ij} , em que cada a_{ij} define a probabilidade dos fotões emitidos num certo voxel j ter originado a LOR i .

O problema matemático da reconstrução iterativa de imagem é dado pela Eq. 3.9, sendo (Y) os dados adquiridos, (f) a distribuição de actividade e (a_{ij}) a matriz de sistema, em que i corresponde às LORs e j aos voxels.

$$Y_i = \sum_j f_j a_{ij} \quad \text{Eq. 3.9}$$

A reconstrução de imagem consiste em resolver a Eq. 3-9, de forma a determinar os valores de f , baseado nos valores adquiridos de Y . Este processo não pode ser resolvido

linearmente devido ao elevado número de valores de f para serem calculados e de Y a serem tidos em consideração.

A optimização da *função objectivo* permite determinar qual a estimativa da imagem, de entre todas as possíveis, que mais se assemelha à imagem real. Um algoritmo estatístico apresenta uma *função objectivo* que é uma função estatística. Existem algoritmos iterativos que utilizam a distância algébrica como *função objectivo* e são denominados por algoritmos algébricos, sendo o seu algoritmo mais popular conhecido por Técnica de Reconstrução Algébrica (ART)[33].

Os algoritmos iterativos estatísticos podem ser distinguidos consoante a natureza da assumção que fazem do ruído. Os algoritmos podem assumir uma distribuição de Poisson para o ruído, como são o caso dos implementados neste trabalho (MLEM e OSEM), ou então assumir uma distribuição de Gauss para o ruído.

3.3.1 Máxima Verosimilhança – Maximização da Expectativa (MLEM)

O algoritmo MLEM foi apresentado em 1977 por Dempster *et.al* [34]. Mais tarde, o conceito foi desenvolvido para a reconstrução de imagem em tomografia por emissão por Shepp, L.A. e Y. Vardi [35] e independentemente por Lange, K. e R. Carson[36].

O algoritmo de Máxima Verosimilhança (ML) permite calcular estimativas a partir de dados desconhecidos, sendo em cada iteração caracterizada por duas etapas, a de Expectativa e a de Maximização. Na etapa de Expectativa o algoritmo procura uma estimativa para a imagem desconhecida. Na etapa de Maximização, o algoritmo procura maximizar a função de Verosimilhança, derivada da estatística de Poisson, que é dada pela Eq. 3.10.

$$L(f_j) = \prod_i e^{-\sum_j a_{ij}f_j} \frac{(\sum_j a_{ij}f_j)^{Y_i}}{Y_i!} \quad \text{Eq. 3.10}$$

Para facilitar o processo de Maximização maximiza-se o logaritmo da função dada pela Eq. 3-10. Como a função logarítmica é monótona crescente, os resultados de maximização da função serão idênticos, que neste caso serão os valores de f_j que maximizam $L(f_j)$. A fórmula geral do MLEM será então dada pela Eq. 3.11.

$$f_j^{(k+1)} = \frac{f_j^{(k)}}{\sum_i a_{ij}} \sum_i \frac{a_{ij}Y_i}{\sum_w a_{iw}f_w^{(k)}} \quad \text{Eq. 3.11}$$

O objectivo do algoritmo é determinar a distribuição de actividade f_j , em que j corresponde a cada voxel, que tenha maior probabilidade de ter originado as projecções adquiridas, Y_i , em cada LOR i , utilizando uma matriz de sistema a_{ij} .

A etapa de projecção corresponde a $\sum_w a_{iw} f_w^{(k)}$, em que se obtêm as projecções que seriam medidas caso $f_w^{(k)}$ estivesse correcto. De seguida, as projecções que foram efectivamente medidas Y_i são divididas pelas obtidas na etapa de projecção. Daí resulta um conjunto de valores que será ponderado e somado com os elementos da matriz de sistema a_{ij} , etapa de retro projecção.

O algoritmo MLEM apresenta duas grandes desvantagens: a baixa velocidade de convergência e a sua instabilidade quando se utilizam dados que apresentam ruído[23].

3.3.2 Subconjuntos Ordenados – Maximização da Expectativa (OSEM)

Devido à baixa velocidade de convergência do algoritmo MLEM, houve necessidade de se encontrar uma solução para este problema. A alternativa surgiu em 1994 por Hudson e Larkin[37] onde o conjunto de LORs foi dividido em subconjuntos. As projecções agrupadas em cada subconjunto devem ser seleccionadas de modo a maximizar a distância angular entre elas, possibilitando que cada subconjunto contribua com o máximo de nova informação possível.

A única diferença na fórmula geral do MLEM e do OSEM, é que na última os somatórios ao longo das LORs são aplicados a cada subconjunto S_m e não ao conjunto total. A fórmula que descreve o algoritmo OSEM é dada pela Eq. 3.12.

$$f_j^{(k+1)} = \frac{f_j^{(k)}}{\sum_{i \in S_m} a_{ij}} \sum_{i \in S_m} \frac{a_{ij} Y_i}{\sum_w a_{iw} f_w^{(k)}} \quad \text{Eq. 3.12}$$

Observando a Eq. 3.12 verifica-se que a actualização da estimativa da distribuição de actividade é feita para cada subconjunto. Deste modo, por cada iteração do algoritmo terão ocorrido m actualizações da estimativa da distribuição de actividade. Daqui se entende que, caso consideremos apenas um subconjunto, estamos a produzir o algoritmo MLEM.

A desvantagem apresentada pelo MLEM em relação ao ruído aqui acentua-se pois cada iteração corresponde a m actualizações e quanto maior o número de subconjuntos maior será o ruído. Por outro lado, como são feitas mais actualizações, serão necessárias menos iterações em relação ao MLEM para se atingir a “convergência”.

A convergência do algoritmo OSEM ainda não foi provada matematicamente. Contudo, os seus bons resultados, similares aos do algoritmo MLEM, encorajam a sua utilização, inclusivamente em prática clínica[23].

4 Programação em paralelo

Actualmente os processadores de computadores (CPU) são a pedra basilar de praticamente toda a indústria informática e de todas as áreas que dependem de forma indispensável desta. No entanto, nem sempre os computadores tiveram uma unidade central de processamento. Os actuais processadores resultam de décadas de pesquisa e consequente evolução.

Nos primórdios da computação, tudo era construído a uma escala bastante elevada e ainda se pretendia compreender como manipular dados através da electrónica, pelo que o tamanho ainda não era um factor de preocupação. O primeiro processador moderno foi construído pela Intel®, o Intel 4004, em 1971. As disputas entre as companhias mais poderosas da altura em muito contribuíram para a rápida evolução dos processadores.

Nos anos 90 houve uma imensa evolução com aumentos significativos da frequência de *clock* e dispositivos de acesso rápido (*cache*) cada vez mais extensos. A partir de 2005, esta tendência começa a mudar, devido a problemas de dissipação de calor, devido ao facto dos componentes estarem reduzidos a dimensões dificilmente expectáveis há uns anos atrás. Deste modo, foi necessário encontrar soluções para este problema e a Intel® e a AMD® decidem lançar no mercado processadores de dois núcleos (*dual-core*). Deste modo, a aposta passa a ser em coordenar e aperfeiçoar a possibilidade de se executarem tarefas simultaneamente nos vários

núcleos, ao invés de se tentar melhorar o desempenho de um núcleo sozinho. Um programa sequencial irá correr em apenas um dos núcleos do processador e o desempenho deste não irá melhorar mais do que hoje em dia, pelo que, assim, se compreende a importância que a programação em paralelo poderá vir a ter nos próximos anos no desenvolvimento de todo o tipo de aplicações[38].

4.1 GPU (*Graphics Processing Unit*)

A indústria dos semicondutores aposta em duas correntes distintas na criação de processadores com vários núcleos: a *multicore* e a *many-core*. A primeira corrente (*multicore*) procura manter a velocidade de execução dos programas sequenciais enquanto se aumenta o número de núcleos. A segunda corrente (*many-core*) procura um aumento na velocidade da taxa de transferência de dados das aplicações paralelas.

Os GPU (*Graphics Processing Unit*) são um tipo de microprocessador especializado em processar gráficos em computadores pessoais, estações de trabalho e profundamente enraizados na indústria dos videojogos.

O GPU utilizado neste trabalho (ver secção 5.1) não é um topo de gama e apresenta 192 núcleos (*many-core*), enquanto os processadores *multicore* topo de gama apresentam 6 núcleos. Deste modo, percebe-se que os objectivos que se pretendem atingir com cada um destes processadores será diferente. Enquanto o CPU foi concebido para realizar qualquer tipo de tarefas como transferência de dados, processamento, entre outros, o GPU foi desenvolvido para realizar uma quantidade enorme de cálculos aritméticos, onde toda a área do *chip* é aproveitada para realizar operações de vírgula flutuante. Grande parte da área do *chip* de CPU é utilizada para memória de acesso rápido e para lógica de controlo, que não contribuem para a velocidade de cálculo, e daí se confirma que a arquitectura de cada *chip* é a responsável pela diferença que é apresentada na Figura 4.1, em relação às operações de vírgula flutuante por segundo em cada processador[38].

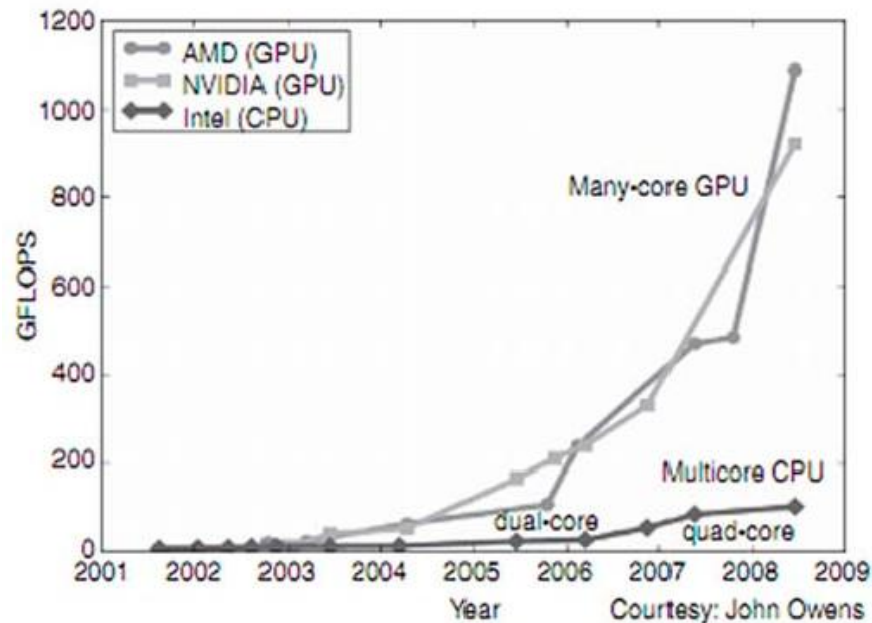


Figura 4.1 - Performance de dois processadores GPU e um CPU[38]

Como se verifica na Figura 4.1, em 2009, um GPU conseguia realizar 1200 Gigas de operações de vírgula flutuante por segundo, 10 vezes mais do que um CPU de quatro núcleos. Esta diferença enorme demonstra que os algoritmos iterativos de reconstrução de imagem, onde existe elevada quantidade de cálculos numéricos, ao serem adaptados à programação em GPU deverão sofrer uma aceleração nos seus tempos de computação. O custo reduzido destes processadores e o seu consumo energético baixo são mais um aliciente para a implementação de algoritmos que recorram ao GPU. No entanto, convém que o utilizador tenha presente que nem todos os algoritmos são passíveis de serem paralelizáveis.

Existem projectos que apresentam resultados bastante satisfatórios na utilização de GPU para a reconstrução de imagem utilizando os algoritmos desenvolvidos neste trabalho (MLEM e OSEM), como é o caso de Kim e Ye[28], em que o algoritmo OSEM implementado para PET com GPU é 125 vezes mais rápido do que com um CPU de um núcleo. Como é óbvio estas comparações são difíceis de realizar porque estão directamente dependentes do CPU e do GPU utilizados.

4.2 CUDA (*Compute Unified Device Architecture*)

A CUDA (*Compute Unified Device Architecture*) surgiu da necessidade de abrir ao mundo a programação em paralelo, que antes era de acesso bastante difícil, devido à extrema complexidade que era programar recorrendo a interfaces para programação de aplicativos (API) gráficas.

A NVIDIA desenvolveu a CUDA, permitindo uma evolução muito grande na programação em paralelo, visto que facilitou a implementação de código em paralelo aos utilizadores. Isto deve-se ao facto da CUDA apresentar um modelo de programação que interliga o código sequencial e o código em paralelo e também permitir o uso de extensões de C/C++, que são linguagens bastante populares e de fácil acesso.

Quando se desenvolve um código em CUDA é necessário desenvolver um código sequencial, que é executado no *host* (CPU), que poderá chamar uma função denominada *kernel*, que é uma função que vai ser executada em paralelo, no *device* (GPU). De referir que apenas um *kernel* é executado de cada vez, ou seja, o *kernel* é executado, a acção volta ao *host* e aí um o *kernel* pode ser chamado. Por outro lado, a execução de um *kernel* é realizada por várias *threads*, que são a menor unidade de processamento.

As *threads* de CUDA são extremamente leves computacionalmente quando comparadas com as de CPU e enquanto a CUDA utiliza milhares para atingir a máxima eficiência possível, os CPUs de vários núcleos apenas conseguem utilizar algumas unidades. Um *kernel* de CUDA é executado por uma série ordenada de *threads*, em que todas as *threads* correm o mesmo código e em que cada *thread* tem uma identificação (*index*) que utiliza para tomar decisões de controlo.

A chamada de um *kernel* a partir do código sequencial resulta na inicialização de uma grelha (*grid*) de blocos (*blocks*) de *threads*, como está ilustrado na Figura 4.2.

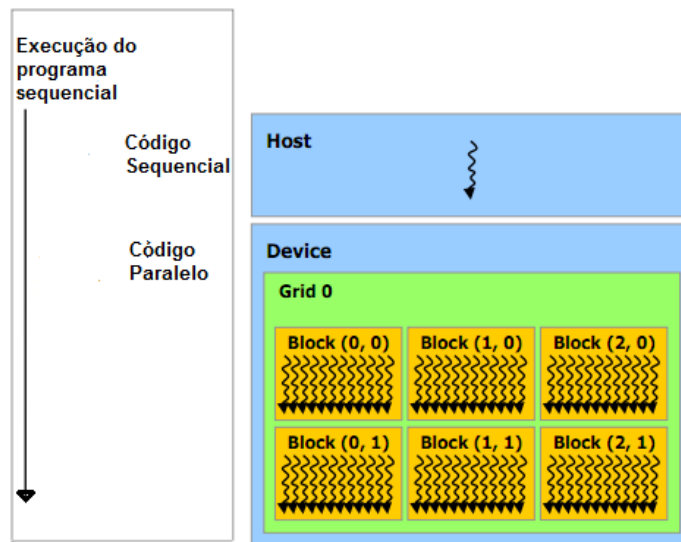


Figura 4.2 - Esquema de execução de um programa em CUDA[39]

As *threads* de um mesmo bloco podem cooperar compartilhando uma certa memória (*shared memory*) e podem também sincronizar-se, isto é, só se poderá voltar ao código sequencial quando todas as ações desse bloco estiverem realizadas. A cooperação entre as várias *threads* ocorre e é extremamente valiosa, visto que a partilha de resultados evita computação redundante. As *threads* em blocos diferentes não podem cooperar entre si. Na Figura 4.3 é esquematizada a hierarquia de diferentes tipos de memória disponíveis para as *threads*. Todas as *threads* têm acesso à memória global (*global memory*) e cada uma tem acesso a memória privada que não é partilhada com mais nenhuma *thread* (*local memory*). Existem ainda outros tipos de memória, como a memória constante, a de textura e a de superfície cuja explicação não entra no âmbito deste trabalho devido à sua complexidade e especificidade. No entanto explicações mais pormenorizadas deste assunto podem ser obtidas através de [39].

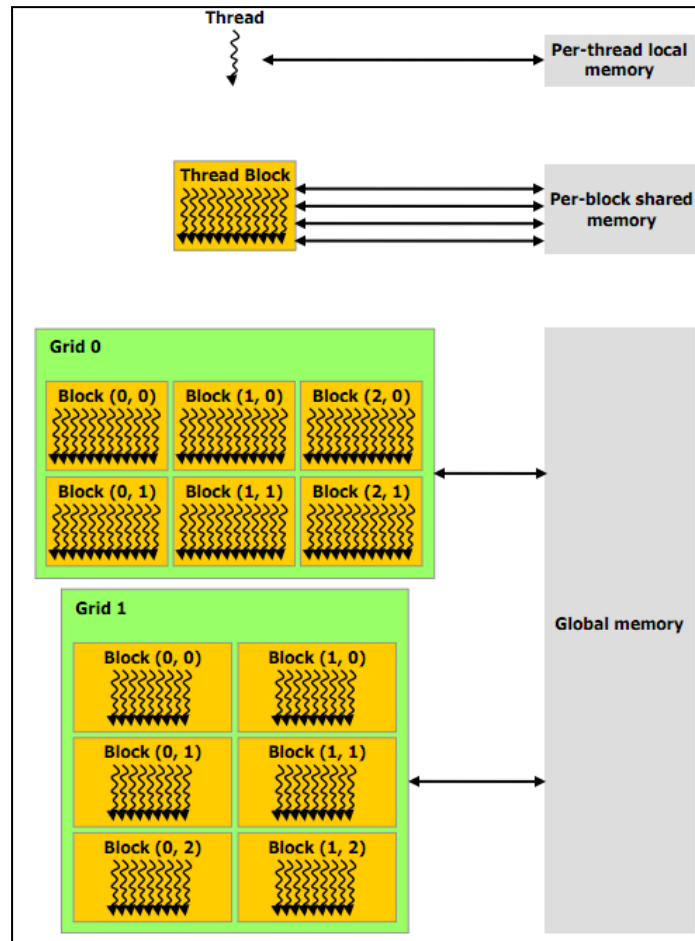


Figura 4.3 - Esquema da hierarquia da memória que uma *thread* tem disponível em CUDA[39]

5 Materiais e Método

5.1 *Hardware e Software*

Os algoritmos MLEM e OSEM foram implementados e testados num Intel® Xeon® E5530 2,4 GHz e 11,7 Gb de memória RAM. O sistema operativo é Red Hat Enterprise Linux 5.3. O GPU utilizado foi um NVIDIA® Quadro® FX 3800 com 1 GB de memória DDR3 e 192 núcleos de CUDA.

Para este trabalho utilizou-se a versão 3.2 do CUDA *Toolkit* [40] e a versão 1.3.0 da biblioteca *Thrust*[41].

A visualização das imagens obtidas foi possível devido ao *software* QuasiManager, desenvolvido por N. Oliveira para o projecto Clear-PEM[42].

5.2 Matriz de Sistema

As matrizes de sistema utilizadas neste trabalho já se encontravam previamente calculadas, recorrendo ao método *Tube-Driven*. Este método caracteriza-se por considerar que a coincidência detectada pode estar num tubo que liga os dois cristais onde se ocorreu a detecção. Como os cristais não são suficientemente pequenos para se poder saber com certeza o ponto onde o fóton entrou no cristal, este método não une simplesmente os centros dos dois cristais como noutro tipo de método que é denominado por *Pixel-Driven*. Deste modo não se obtém uma LOR, mas sim um tubo de resposta (TOR). Basicamente este TOR é a junção de todas as possíveis LORs entre os dois detectores (Figura 5.1).

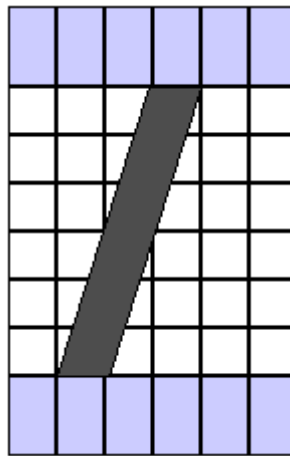


Figura 5.1 - TOR a unir os dois detectores[23]

Este método foi desenvolvido por N.Matela e uma explicação mais detalhada pode ser encontrada em [23, 43].

O desenvolvimento deste método em GPU foi ponderado, mas devido à reduzida quantidade de dados resultante da matriz de sistema (10000 elementos) e à dificuldade em paralelizar certos conceitos do método, este tornava-se mais lento em relação ao código sequencial, mas não foi levado por diante. Caso os elementos resultantes da matriz de sistema fossem em número mais elevado, provavelmente compensaria desenvolver um algoritmo recorrendo ao GPU. Assim, a quantidade de processos necessários para implementar um código que paralelizasse a criação desta matriz de sistema não compensa em termos de tempo, pelo que se poderá considerar recorrer a GPU caso se venha a pretender desenvolver um algoritmo de reconstrução 3D. Neste caso, o tempo de computação em CPU seria bastante elevado e compensaria recorrer à programação em GPU.

Ainda assim foi criado um modelo que juntava a programação em série e a programação em paralelo, mas a aceleração do tempo de computação era mínima. O tempo de computação aproximado deste método utilizando código sequencial é de, aproximadamente, um minuto.

5.3 Programação em CUDA

Na secção 4.2 foi referido que a chamada de um *kernel* resulta na inicialização de uma grelha de blocos de *threads*. Estas *threads* podem encontrar-se em blocos unidimensionais, bidimensionais ou tridimensionais. Estes blocos fazem parte de uma grelha que pode ser unidimensional ou bidimensional. A cada bloco e *thread* é atribuído um *índice* único, como foi referido anteriormente, através de uma variável bidimensional, *blockIdx*, no caso do bloco e uma tridimensional, *threadIdx*, no caso da *thread*. Esta hierarquia das *threads* pode ser compreendida mais facilmente através da Figura 5.2.

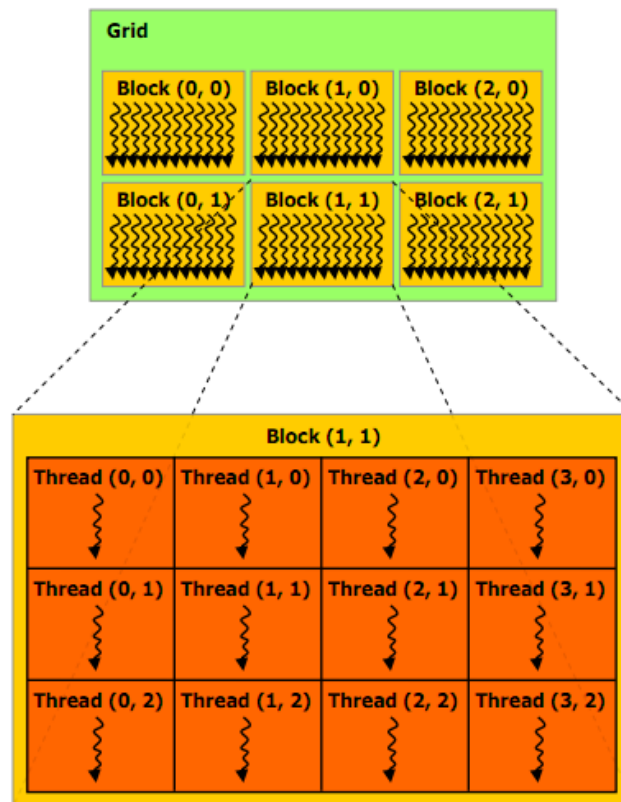


Figura 5.2 - Grelha de blocos de *threads*[39]

Existe um limite para o número de *threads* por bloco (512 *threads*), pois todas as *threads* de um bloco devem residir no mesmo núcleo de processador e devem partilhar os recursos disponíveis de memória do núcleo de processador.

A invocação do *kernel* indica tanto o número de blocos como o número de *threads*. A transferência de dados entre o *host* e o *device* é feita recorrendo à função *cudaMemcpy()*, enquanto a alocação de memória utiliza a função *cudaMalloc()* e a libertação de memória é realizada através da função *cudaFree()*. De seguida é apresentado um código muito simples, onde se encontram as funções atrás descritas e onde se pretende adicionar dois vectores:

```

__global__ void add_vec (float* v1, float* v2, float* v3, int

    int i=threadIdx.x+blockIdx.x*blockDim.x;
    if (i<dim)
        v3[i]=v1[i]+v2[i];
}

int main(){

int dim=8;

//Alocação de memória no CPU
float* v1=(float*) malloc(dim*sizeof(float));
float* v2=(float*) malloc(dim*sizeof(float));
float* v3=(float*) malloc(dim*sizeof(float));

for (int=0;i<dim;i++){
    v1[i]=i;
    v2[i]=i;
    v3[i]=0;
}

//Alocação de memória no GPU
float *v1_d, *v2_d, *v3_d;
cudaMalloc((void**)&v1_d, dim*sizeof(float));
cudaMalloc((void**)&v2_d, dim*sizeof(float));
cudaMalloc((void**)&v3_d, dim*sizeof(float));

//Transferência para o GPU
cudaMemcpy(v1_d, v1, dim*sizeof(float), cudaMemcpyHostToDevice);
cudaMemcpy(v2_d, v2, dim*sizeof(float), cudaMemcpyHostToDevice);
cudaMemcpy(v3_d, v3, dim*sizeof(float), cudaMemcpyHostToDevice);

//Invocação do kernel para adicionar dois vectores, usando 2
blocos de 4 threads (<<<2,4>>>)
add_vec<<<2,4>>>(v1_d, v2_d, v3_d, dim);

//Transferência para o CPU
cudaMemcpy(v3, v3_d, dim*sizeof(float), cudaMemcpyDeviceToHost);

//Libertação da memória alocada no GPU
cudaFree(v1_d);
cudaFree(v2_d);
cudaFree(v3_d);

return 0;
}

```

5.4 Biblioteca *Thrust*²

A biblioteca *Thrust* surge após alguns meses de desenvolvimento de código de CUDA, como o apresentado anteriormente, onde era necessário especificar tudo em relação a grelhas, blocos, *threads*. O desenvolvimento de um simples algoritmo que multiplicasse duas matrizes de forma sempre correcta resulta num código de elevada complexidade que pode ter centenas de linhas de código. Com o desenvolvimento das bibliotecas de CUDA todo este trabalho para o utilizador deixa de ser necessário, sendo o próprio algoritmo a decidir quais as melhores escolhas em termos de número de grelhas, blocos e *threads*. Para além disso, estas bibliotecas encontram-se optimizadas e vão permitir resultados muito próximos, mas com uma implementação muito mais rápida para o utilizador, dos que se obtêm caso seja o próprio utilizador a implementar um código como o que foi apresentado na secção 5.3.

Simultaneamente com a primeira experiência com a biblioteca *Thrust*, foram testados e analisados algoritmos na biblioteca *CUBLAS*. No entanto, esta biblioteca não apresentava certas funções fundamentais para a implementação dos algoritmos, como a ordenação de vectores, que a biblioteca *Thrust* apresentava. Por esta razão o código foi implementado recorrendo à biblioteca *Thrust*. Esta biblioteca aproxima a programação em paralelo à programação sequencial, o que torna mais simples para o utilizador compreender o que tem de ser implementado e de que forma[44]. Além disso, com estas bibliotecas a alocação e libertação da memória do GPU são automaticamente realizadas[45].

As funções mais utilizadas na implementação dos algoritmos de reconstrução foram de redução, redução por chave e ordenação por chave. Por outro lado, existe uma operação denominada *transform* que permite ao utilizador criar uma estrutura onde poderá realizar as operações que pretender, tudo isto em paralelo.

A função ordenação por chave da biblioteca *Thrust* recebe dois vectores de entrada, em que um deles é a chave que irá estabelecer a ordem dos elementos do outro vector de entrada. Como se pode observar na Tabela 5.1, o vector CHAVE é ordenado de forma crescente, passando a ser o vector chave de saída. O vector de entrada vai ser ordenado pela ordem com que o vector CHAVE também foi, isto é, se o quarto elemento do vector CHAVE era o terceiro menor de todo o vector, então o quarto elemento do vector de entrada será colocado na terceira posição do vector de saída. Como se verifica no vector de saída, a terceira posição é ocupada pelo valor 3, que, no vector de entrada, se encontrava na quarta posição.

² <http://code.google.com/p/thrust/>

Tabela 5.1 - Função *order_by_key* (ordenação por chave) da biblioteca *Thrust*

CHAVE	0	1	3	2	6	4	5	7
Vector de entrada	0	1	2	3	4	5	6	7
Ordenação								
Chave de saída	0	1	2	3	4	5	6	7
Vector de saída	0	1	3	2	5	6	4	7

A função redução por chave da biblioteca *Thrust* recebe dois vectores de entrada, em que um deles é a chave que irá estabelecer quais os elementos do vector de entrada se irão somar entre si. Como se pode observar na Tabela 5.2, o primeiro elemento do vector de saída é dado pela soma dos dois primeiros elementos do vector de entrada. Isto deve-se ao facto de as duas primeiras posições do vector CHAVE apresentarem o mesmo valor (ou a mesma “chave”). Caso, por acaso, o terceiro elemento de CHAVE fosse também 0, o primeiro elemento do vector de saída seria 3 (1+2).

Tabela 5.2 - Função *reduction_by_key* (redução por chave) da biblioteca *Thrust*

CHAVE	0	0	1	2	2	2	3	3
Vector de entrada	0	1	2	3	4	5	6	7
Ordenação								
Chave de saída	0	1	2	3				
Vector de saída	1	2	12	13				

5.5 Implementação do programa para OSEM

A equação geral do OSEM é:

$$f_j^{(k+1)} = \frac{f_j^{(k)}}{\sum_{i \in S_m} a_{ij}} \sum_{i \in S_m} \frac{a_{ij} Y_i}{\sum_w a_{iw} f_w^{(k)}} \quad \text{Eq. 5.1}$$

Antes de se começar o processo de reconstrução propriamente dito, existem uma série de processos que vão ser realizados para que, assim que o código entre na 1ª iteração comece a programar em paralelo (só com vectores de *device*) e não necessite de trabalhar com vectores de *host*, a não ser quando se pretende gravar uma imagem ou acabar o processo iterativo. Para esta explicação assume-se que a reconstrução de imagem vai ser realizada a partir das projecções obtidas a partir de quatro posições de aquisição.

Inicialmente, procede-se à leitura da matriz de sistema e dos quatro linogramas para vectores de *host*. De seguida devido ao tamanho da matriz de sistema (10000 x 10000) e ao facto de, normalmente, mais de 90% dos elementos serem desprezáveis pelos algoritmos por serem muito baixos, elimina-se todos esses elementos desprezáveis e cria-se dois vectores do tamanho do novo vector de sistema, que guardarão os índices antigos dos valores não da matriz de sistema que não foram eliminados. Neste processo os valores são guardados por uma ordem diferente da inicial devido ao facto de existirem subconjuntos. Por exemplo, com quatro subconjuntos, a 100ª posição da matriz de sistema não seria a 100ª a entrar neste somatório $\sum_{i \in S_m} a_{ij}$, mas a 25000ª. Este tipo de ordenação efectua-se antes do processo iterativo começar, para otimizar o tempo de computação.

De seguida é necessário rodar o vector de sistema, em quatro variações angulares diferentes, que variam com a posição de aquisição. Depois deste passo fica-se com cinco vectores de sistema e dez vectores com os índices dos valores da antiga matriz de sistema.

Como é necessário rodar o objecto em ângulos não rectos, é necessário transformar um quadrado 100x100 num círculo, que permita qualquer rotação. A estimativa inicial é criada, mas só tens valores não nulos no círculo referido anteriormente. Assim, garante-se que a imagem nunca fará rotações que não sejam permitidas. O último passo antes de se entrar no processo iterativo é calcular este somatório que nunca se altera, por isso não é lógico ser calculado mais que uma vez.

Agora segue a explicação de uma iteração do algoritmo OSEM. Com quatro subconjuntos, vão ocorrer quatro sub-iterações. Primeiro, calcula-se este somatório $\sum_w a_{iw} f_w^{(k)}$ e obtêm-se as projecções que seriam medidas caso a estimativa inicial estivesse certa (Etapa de Projecção). Depois procede-se à divisão $\frac{Y_i}{\sum_w a_{iw} f_w^{(k)}}$ em que Y_i são as projecções efectivamente

medidas. De seguida, estes valores são somados e ponderados com a matriz de sistema. Por fim, estes valores são divididos por $\sum_{i \in S_m} a_{ij}$ previamente calculados e depois são multiplicados pela estimativa anterior. Chega-se, assim, ao fim da 1ª sub-iteração e este processo repete-se, só variando o subconjunto de dados com que se está a trabalhar.

O caso do algoritmo MLEM é um caso particular deste, já que o subconjunto é o próprio conjunto todo, pelo que a explicação de uma iteração de MLEM é igual à explicação de uma sub-iteração de OSEM.

5.6 Planos ortogonais da mama

Até à data, tanto quanto se sabe, não existe uma convenção aceita para as classificações dos diferentes planos de uma imagem de PEM: Deste modo, neste trabalho, foi definida uma classificação para estes planos tendo em conta os planos das imagens de corpo inteiro. A diferença mais significativa entre estes dois tipos de imagem reside na posição do eixo de rotação da câmara, como se pode observar na Figura 5.3 e Figura 5.4.

Nos estudos anatómicos, o corpo geralmente é dividido em planos. Aqueles que se utilizam com mais frequência são os planos transaxial, sagital e coronal (Figura 5.3), que são perpendiculares entre si. O primeiro é um plano horizontal, que divide o corpo em parte superior e inferior. O segundo é um plano vertical, que divide o corpo em parte direita e esquerda. E, em último lugar, o plano coronal, que também é um plano vertical e divide o corpo em parte anterior e posterior[46].

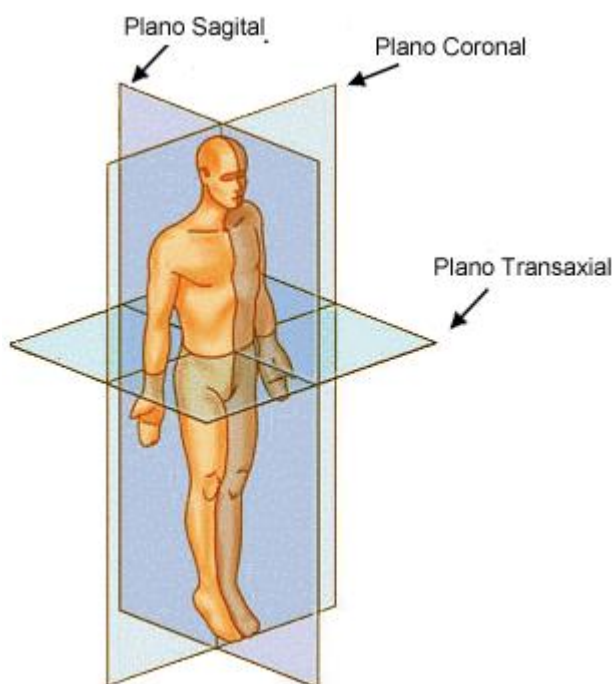


Figura 5.3 - Planos sagital, coronal e transaxial numa imagem de corpo inteiro[47]

Como a definição do plano transaxial se baseia na direcção do eixo de rotação (são perpendiculares) e numa câmara de PEM esse eixo é perpendicular ao torso, o plano transaxial será o plano paralelo ao torso (divide a mama em parte anterior ou posterior). Tendo em conta os planos definidos para as imagens de corpo inteiro, o plano sagital será o plano que divide a mama em região central e lateral e o coronal divide a mama em região superior e inferior (Figura 5.4).

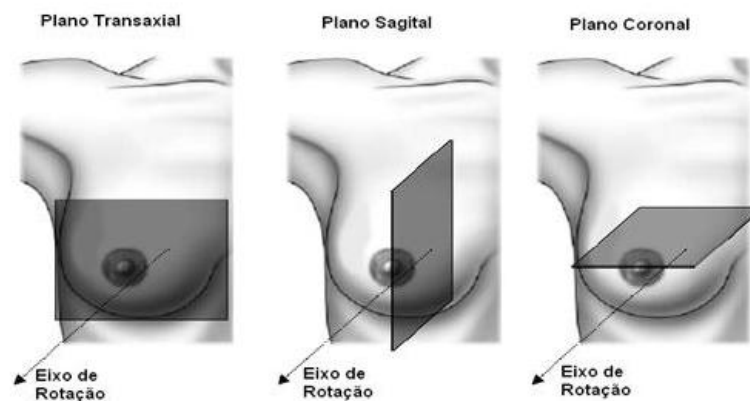


Figura 5.4 - Planos transaxial, sagital e coronal numa imagem de PEM[48]

6 Resultados e Discussão

Neste trabalho implementou-se dois algoritmos de reconstrução iterativa de imagem médica, o MLEM e o OSEM. Os resultados obtidos do algoritmo OSEM que vão ser apresentados foram obtidos sempre com 4 subconjuntos ordenados, apesar de também estar feita a implementação para 10 subconjuntos ordenados.

O objectivo deste trabalho era diminuir o tempo de computação dos algoritmos, mas garantindo que a qualidade das imagens era similar à obtida com a implementação já testada em linguagem C, utilizando exclusivamente a programação sequencial. Neste capítulo proceder-se-á à avaliação da qualidade das imagens obtidas, comparando-as com as imagens obtidas utilizando o código de linguagem C.

Os resultados deste trabalho foram obtidos com dados adquiridos com uma fonte pontual, dados obtidos por simulação com um fantoma de mama e, por fim, com dados de um ensaio pré-clínico, onde foi analisado um rato.

Após a avaliação da qualidade das imagens reconstruídas serão apresentados os tempos de computação, tanto das implementações deste trabalho, como do código em linguagem C.

Por fim, serão discutidos os diversos pontos desde capítulo, de modo a sintetizar toda a informação disponível.

6.1 Fonte Pontual

A reconstrução de imagem da fonte pontual foi realizada a partir das projecções obtidas a partir de quatro posições de aquisição: 90°, 135°, 180° e 225°.

Nesta secção iremos apresentar a resolução espacial nos três planos (sagital, coronal e transaxial) ao longo de 10 iterações para os algoritmos MLEM e OSEM implementados em GPU e em CPU. Com a informação da resolução espacial será decidido quantas iterações serão feitas em cada algoritmo para se reconstruírem as imagens.

6.1.1 Resolução espacial

A definição física de resolução espacial é a distância mínima entre dois pontos objecto observáveis como imagens separadas. A resolução espacial está associada a qualidades, por vezes invocadas na apreciação das imagens, como definição, pormenor, detalhe.

A partir dos perfis medidos dos planos sagital, coronal e transaxial da fonte pontual (Figura 6.1), ajustou-se uma curva *gaussiana* (Figura 6.2) e mediu-se a largura a meia altura (FWHM). Estes passos foram efectuados no *software* QuasiManager. De seguida multiplicou-se pela dimensão dos detectores e dividiu-se pelo número de pixels.

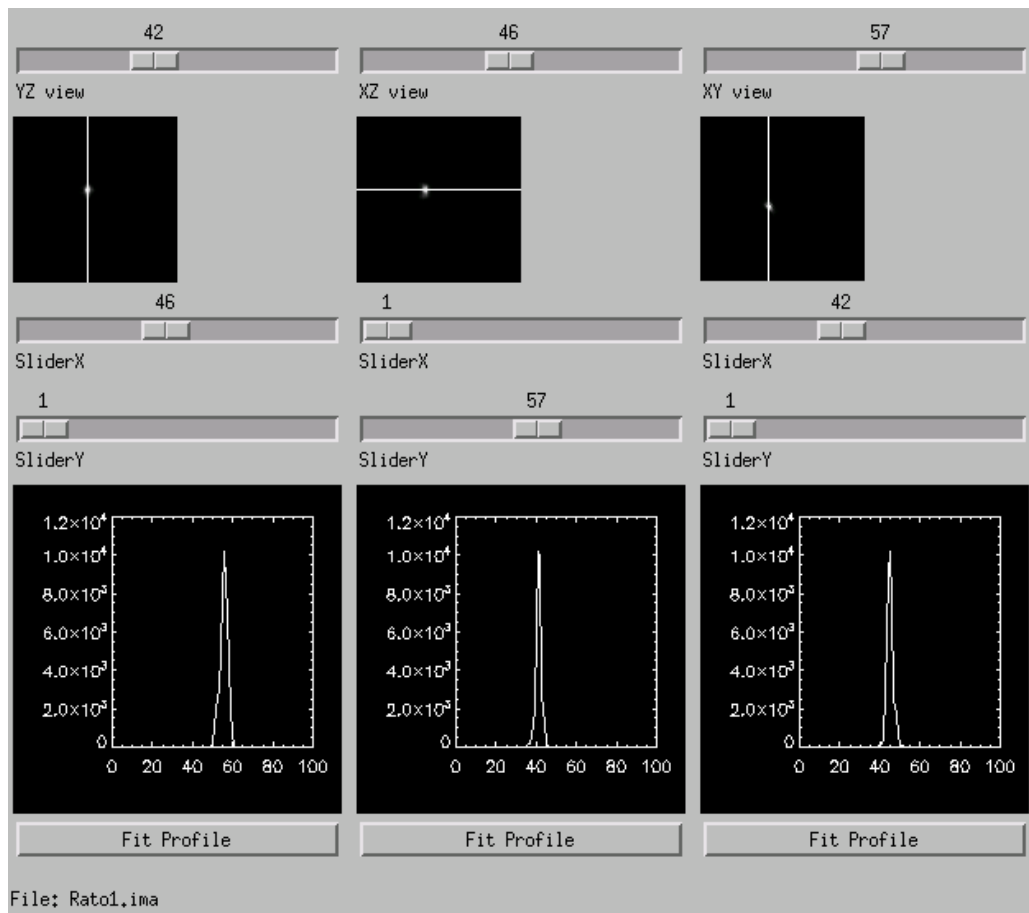


Figura 6.1 - Perfis dos planos sagital, coronal e transaxial (da esquerda para a direita)

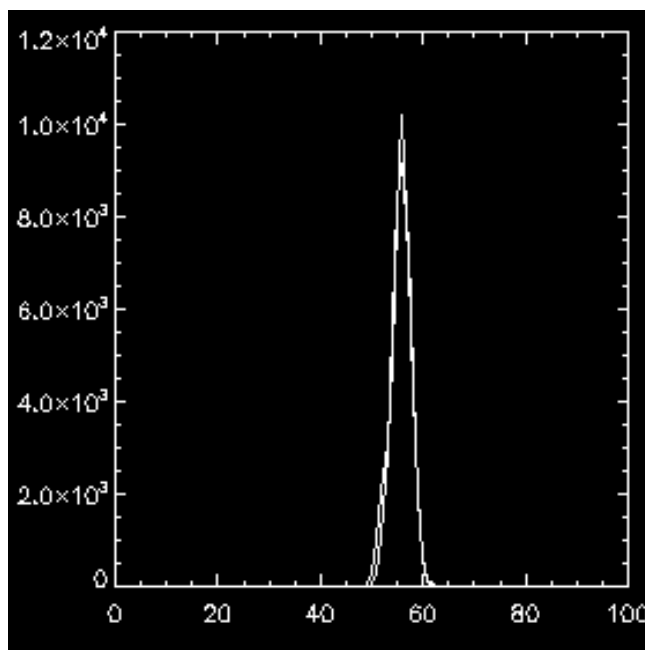


Figura 6.2 - Curva *gaussiana* ajustada ao perfil do plano sagital

De seguida são apresentadas as resoluções espaciais dos 3 planos ao longo do processo iterativo do algoritmo MLEM (Figura 6.3, Figura 6.4, Figura 6.5) e do algoritmo OSEM (Figura 6.6, Figura 6.7, Figura 6.8), com o objectivo de verificar se as implementações em GPU e CPU são idênticas e também tentar seleccionar uma iteração onde os algoritmos já tivessem convergido, para apresentar os resultados dessa iteração.

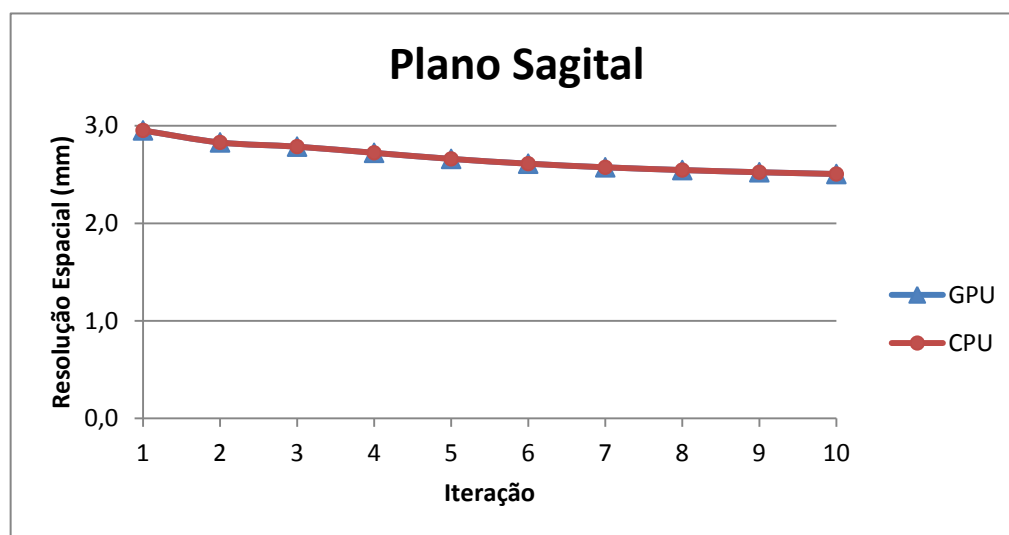


Figura 6.3 - Resolução espacial sagital do algoritmo MLEM

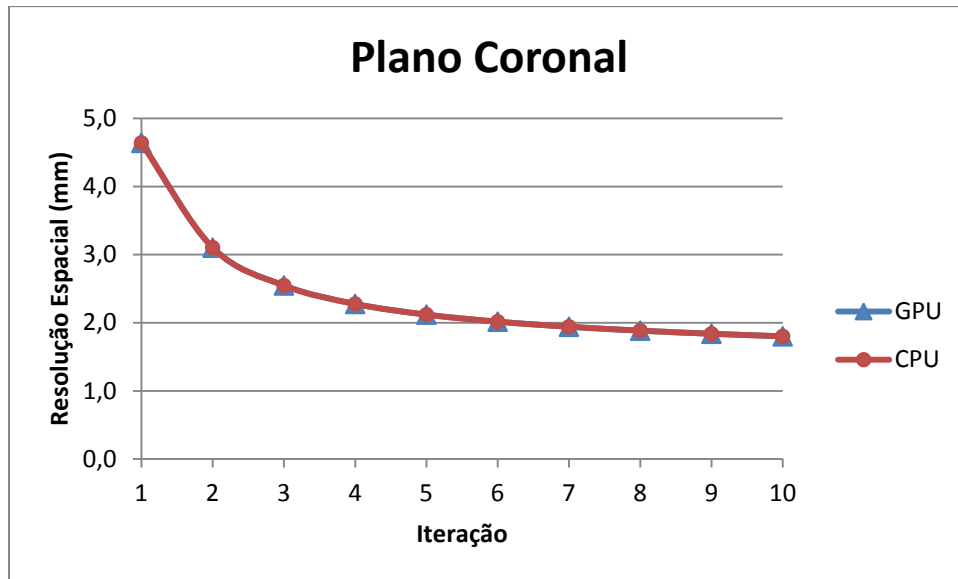


Figura 6.4 - Resolução espacial coronal do algoritmo MLEM

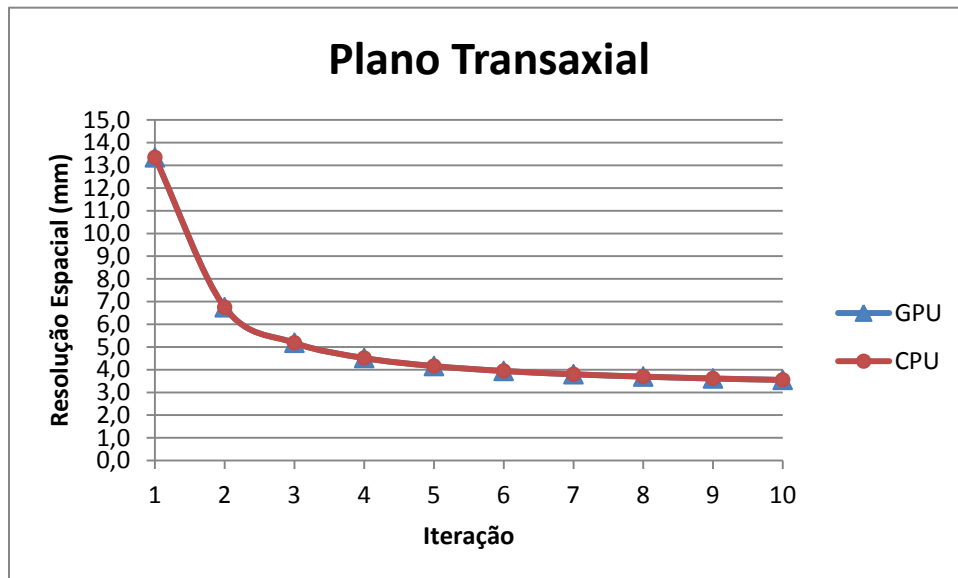


Figura 6.5 - Resolução espacial transaxial do algoritmo MLEM

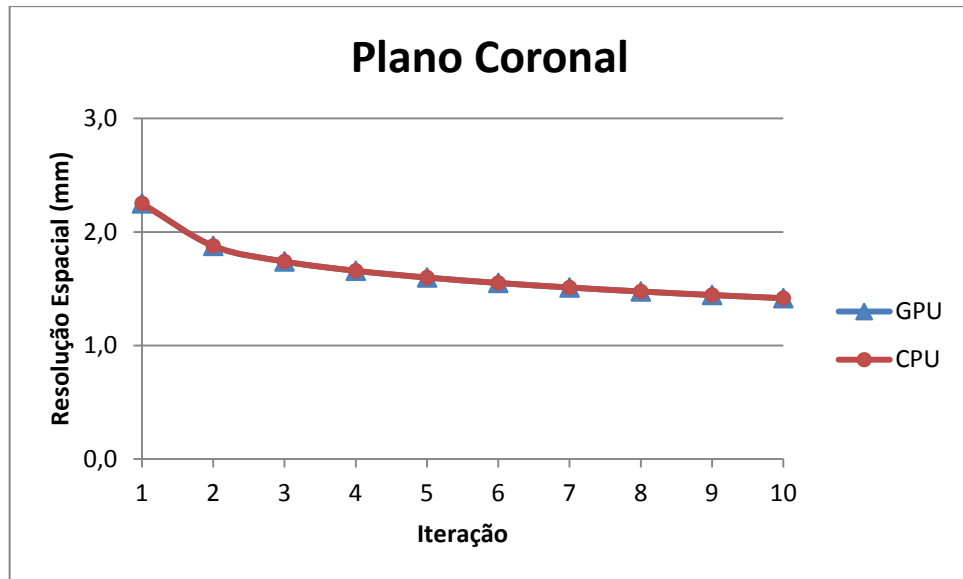


Figura 6.6 - Resolução espacial coronal do algoritmo OSEM

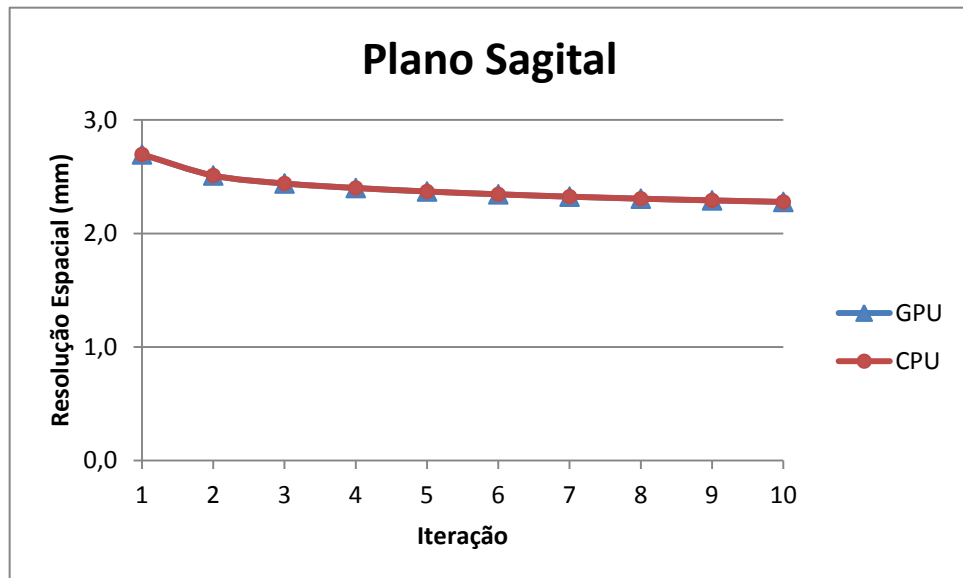


Figura 6.7 - Resolução espacial sagital do algoritmo OSEM

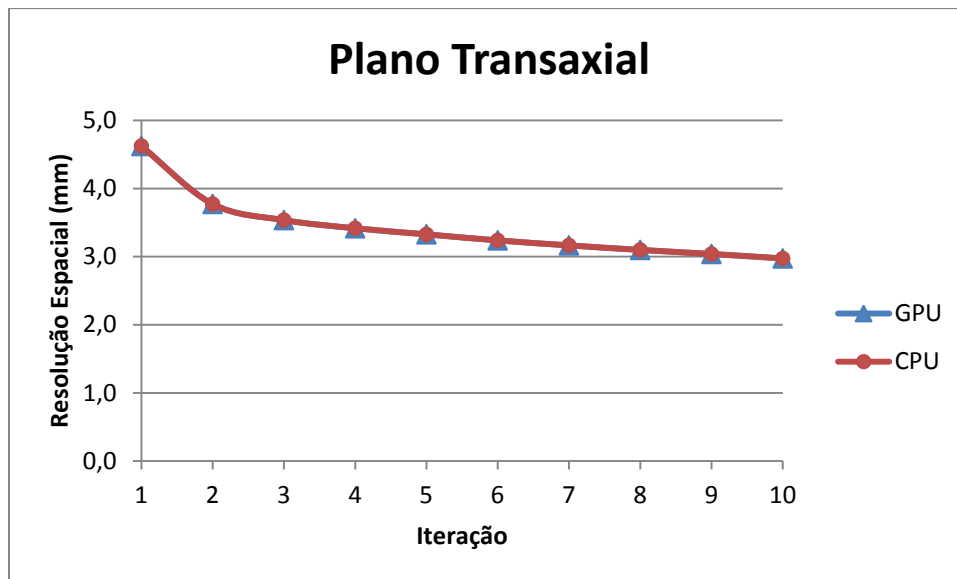


Figura 6.8 - Resolução espacial transaxial do algoritmo OSEM

A partir dos gráficos de resolução espacial verifica-se que em todos eles os valores são idênticos para a implementação em CPU e em GPU, pelo que é uma indicação de que as imagens estarão idênticas também. Verifica-se que o plano coronal apresenta uma melhor resolução espacial do que os outros dois planos, tanto para o algoritmo MLEM como OSEM.

A partir dos gráficos era complicado estabelecer um critério para definir a iteração onde se consideraria que o algoritmo já teria convergido. Assim, analisou-se os dados do OSEM e decidiu-se que se consideraria que o algoritmo teria convergido quando a diminuição da resolução espacial em cada um dos planos fosse inferior a 5%. Para o algoritmo OSEM, essa diminuição ocorreu da 3ª para a 4ª iteração. De uma maneira geral, com 4 subconjuntos no algoritmo OSEM, o algoritmo MLEM demoraria 4 vezes mais a convergir do que o OSEM, pelo que seriam 16 iterações. Como só se analisou a resolução espacial até 10 iterações, vamos apresentar resultados do algoritmo OSEM ao fim de 4 iterações e do algoritmo MLEM ao fim de 10 iterações.

6.1.2 Imagem reconstruída

Na Figura 6.9 encontram-se as imagens reconstruídas da fonte pontual após 10 iterações com o algoritmo MLEM, implementado em CPU e GPU. Tendo como base a observação das imagens as imagens parecem similares. Na Figura 6.9 encontram-se as imagens reconstruídas da fonte pontual após 4 iterações com o algoritmo OSEM, implementado em CPU e GPU. Tendo como base a observação das imagens, estas parecem similares.

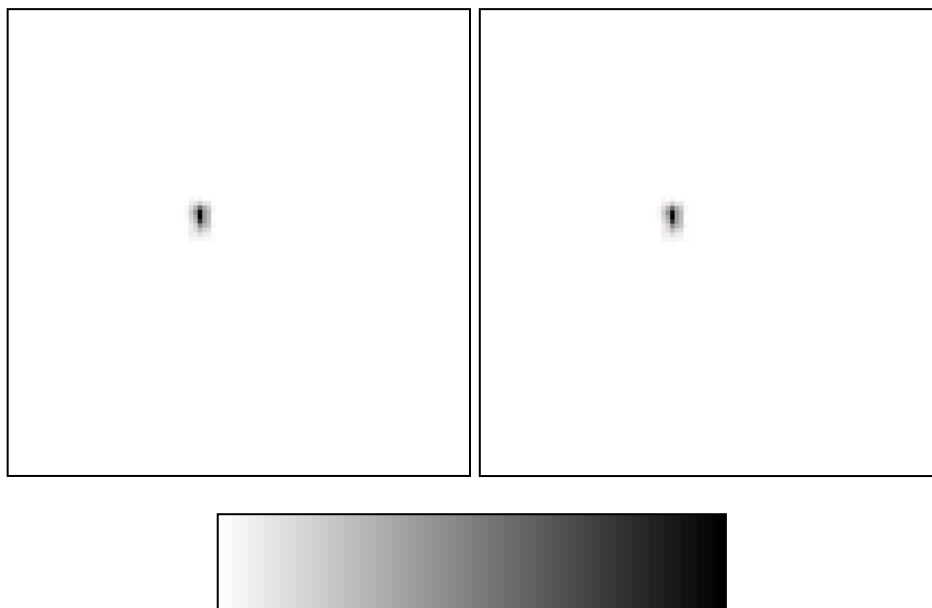


Figura 6.9 - Plano coronal central da fonte pontual após 10 iterações (MLEM) com CPU (esquerda) e GPU (direita)

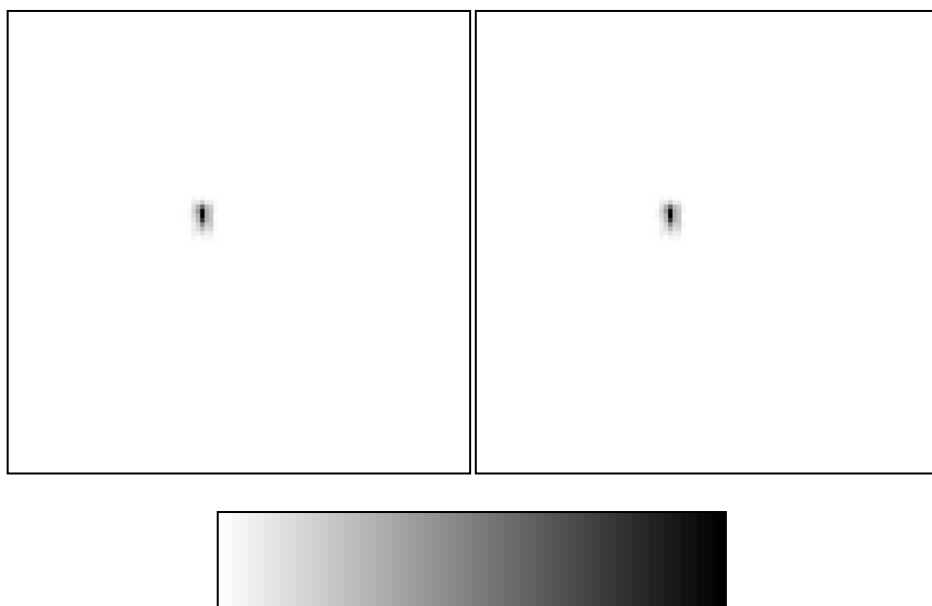


Figura 6.10 - Plano coronal central da fonte pontual após 4 iterações (OSEM) com CPU (esquerda) e GPU (direita)

6.2 Fantoma da mama

A reconstrução de imagem do fantoma de mama foi realizada a partir das projecções obtidas por simulação de MonteCarlo[49] a partir de duas posições de aquisição: 0° e 90°.

Nesta secção iremos apresentar o índice de homogeneidade ao longo de 10 iterações para os algoritmos MLEM e OSEM implementados em GPU e em CPU.

De seguida, irá calcular-se a razão sinal-ruído e o erro médio absoluto normalizado, de modo a comparar a imagem de CPU com a de GPU.

6.2.1 Homogeneidade

O índice de homogeneidade (IH) foi calculado a partir da Eq. 6.1:

$$IH = \frac{\mu}{\sigma} \quad \text{Eq. 6.1}$$

Para se determinar a média (μ) e o desvio-padrão (σ), no *software* QuasiManager seleccionou-se uma ROI (Região de Interesse) numa zona do fundo da imagem e foi-se calculando os parâmetros sempre nessa zona ao longo das iterações. Na Figura 6.11 é apresentada a ROI escolhida delimitada a vermelho.

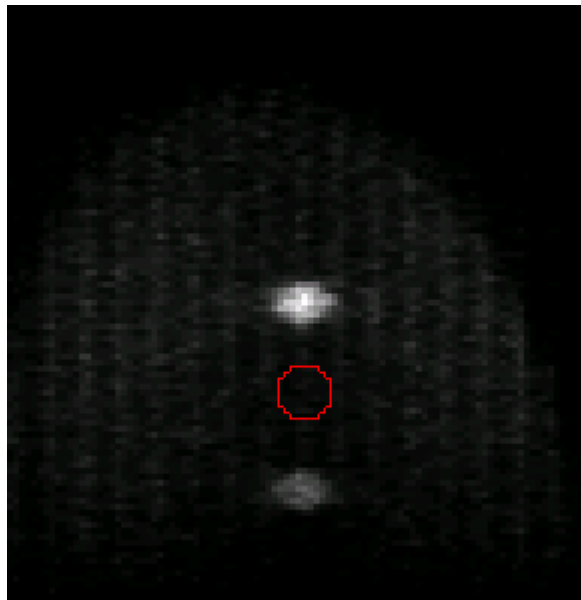


Figura 6.11 – ROI escolhida para determinar o índice de homogeneidade

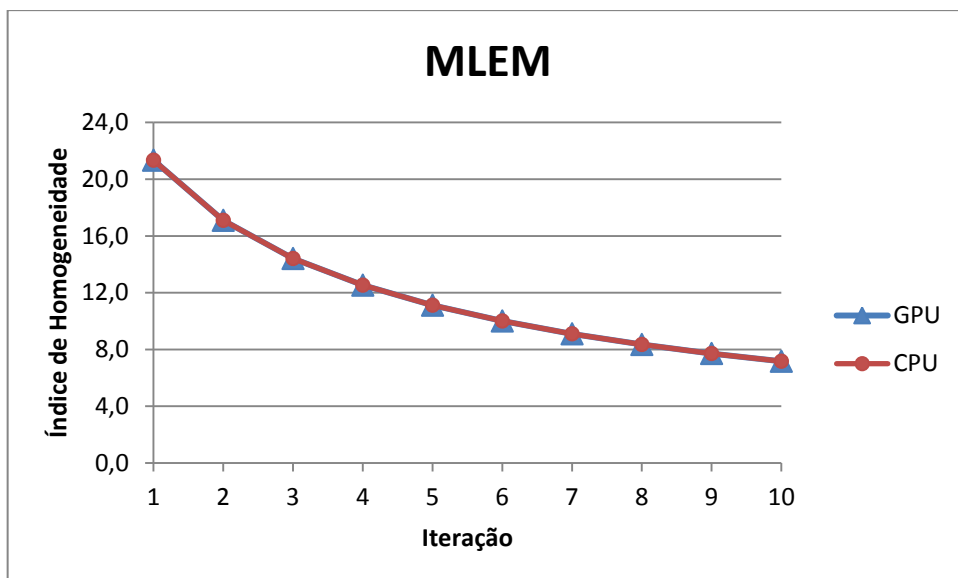


Figura 6.12 - Índice de homogeneidade ao longo do processo iterativo de MLEM

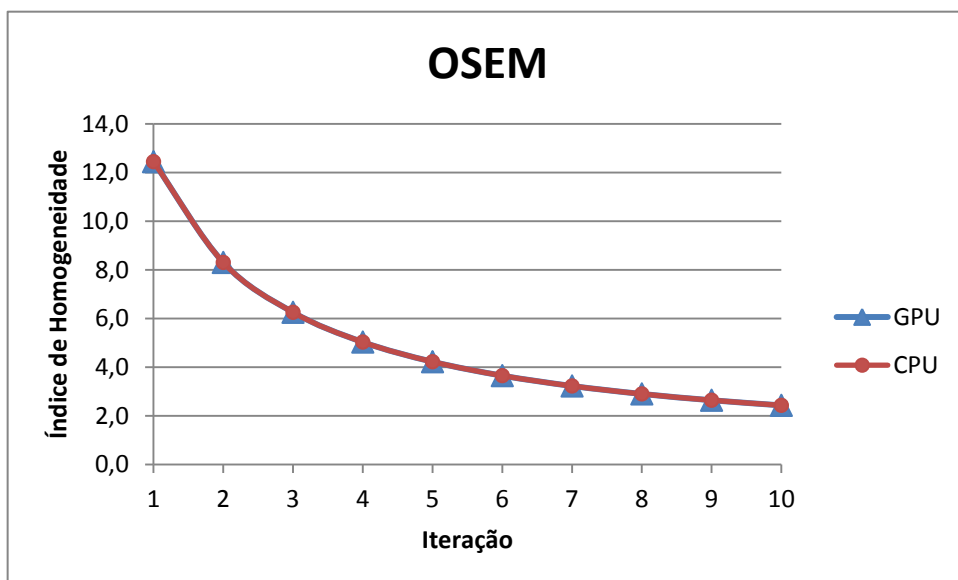


Figura 6.13 - Índice de homogeneidade ao longo do processo iterativo de OSEM

O índice de homogeneidade diminuiu sempre ao longo das iterações tanto para MLEM (Figura 6.12) como para OSEM (Figura 6.13). Desta forma, pode concluir-se que o ruído está a sobrepor-se ao sinal, pelo que se este fosse o único critério de convergência que tivéssemos, decidiríamos só realizar uma iteração, de modo a não tornar a nossa imagem ruidosa. Os valores mantiveram-se similares para as implementações em CPU e GPU, pelo que é um sinal de que a reconstrução em GPU deve estar correcta.

6.2.2 Razão sinal-ruído (SNR)

O SNR é um parâmetro que informa sobre o nível de ruído de uma imagem. Para um objecto ser identificável, o seu SNR tem de ser superior a 5, valor convencionalmente usado em mamografia de raio-X.

Calculou-se a razão sinal-ruído para a 10ª iteração do MLEM e para a 4ª iteração do OSEM, de modo a comparar os resultados das implementações em CPU e GPU.

A razão sinal-ruído (SNR) foi calculada a partir da Eq. 6.1:

$$SNR = \frac{\mu_{lesão} - \mu_{fundo}}{\sigma_{fundo}} \quad \text{Eq. 6.2}$$

Para se determinar a média (μ) e o desvio-padrão (σ), no *software* QuasiManager seleccionou-se uma ROI na zona da lesão que se encontra na zona central da imagem e seleccionaram-se várias ROIs no fundo da imagem, que funcionavam como se de uma só se tratasse.

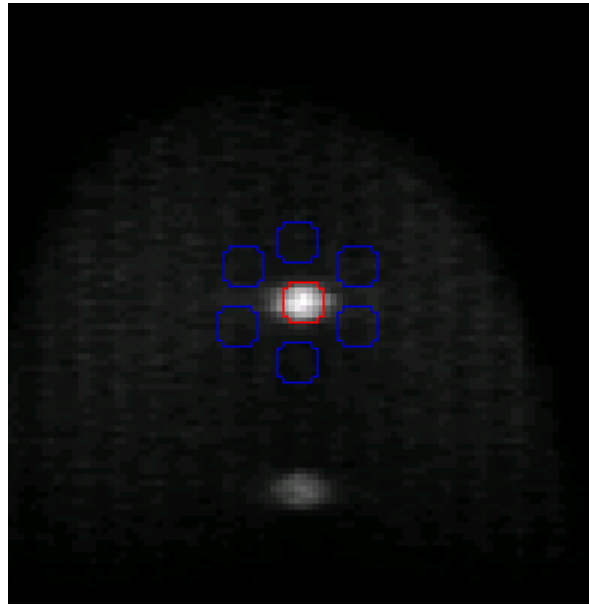


Figura 6.14 – ROI da lesão a azul e ROIs do fundo a vermelho

$$SNR = \frac{\mu_{lesão} - \mu_{fundo}}{\sigma_{fundo}} \quad \text{Eq. 6.3}$$

Para a 10ª iteração do MLEM em CPU:

$$SNR = \frac{179,9704 - 28,03169}{\sqrt{39,5658}} = 24,1551 \quad \text{Eq. 6.4}$$

Para a 10ª iteração do MLEM em GPU:

$$SNR = \frac{179,9704 - 28,03169}{\sqrt{39,5658}} = 24,1551 \quad \text{Eq. 6.5}$$

Para a 4ª iteração do OSEM em CPU:

$$SNR = \frac{202,2527 - 27,26488}{\sqrt{56,39872}} = 23,3009 \quad \text{Eq. 6.6}$$

Para a 4ª iteração do OSEM em GPU:

$$SNR = \frac{202,2527 - 27,26488}{\sqrt{56,39872}} = 23,3009 \quad \text{Eq. 6.7}$$

Como se pode verificar os valores da razão sinal-ruído são iguais com o mesmo algoritmo em CPU e em GPU e isso é uma evidência muito forte de que a implementação do código em GPU está feita de forma correcta, pelo menos para o caso em que há duas posições de aquisição.

6.2.3 Erro médio absoluto normalizado

O erro médio absoluto normalizado (*NME*) é dado pela Eq. 6.8:

$$NME = \frac{\sum_i |P_i - O_i|}{\sum_i O_i} \quad \text{Eq. 6.8}$$

P_i são os elementos da imagem de GPU e O_i são os elementos da imagem de CPU.

Para o fantoma de mama o resultado foi da ordem dos 10^{-9} tanto para o MLEM como para o OSEM, o que indica que as diferenças entre a imagem de CPU e GPU é desprezável e que poderá vir dos arredondamentos que o processador tem de fazer.

6.2.4 Imagem reconstruída

Na Figura 6.15 encontram-se as imagens reconstruídas do fantoma de mama após 10 iterações com o algoritmo MLEM, implementado em CPU e GPU. Tendo como base a observação das imagens, mais uma vez as imagens parecem similares. Na Figura 6.16 encontram-se as imagens reconstruídas da fonte pontual após 4 iterações com o algoritmo OSEM, implementado em CPU e GPU. Tendo como base a observação das imagens, estas parecem similares.

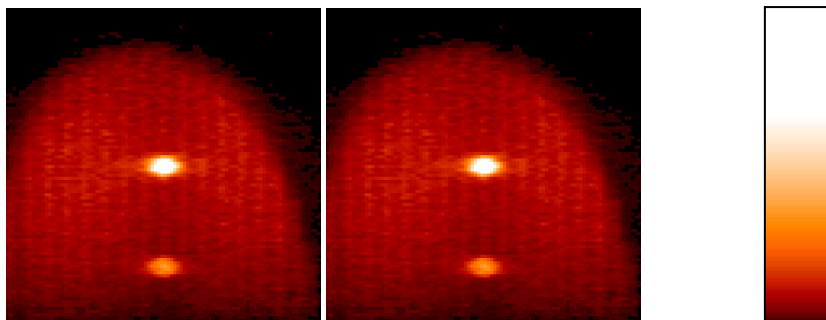


Figura 6.15 - Plano coronal central do fantoma de mama após 10 iterações (MLEM) com CPU esquerda) e GPU (direita)

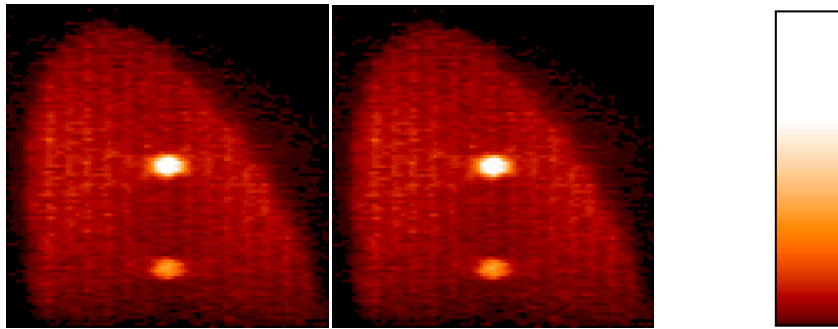


Figura 6.16 - Plano coronal central do fantoma de mama após 4 iterações (OSEM) com CPU (esquerda) e GPU (direita)

6.3 Pré-clínico de um rato

Nesta secção pretende mostrar-se que, também com casos reais, a implementação dos algoritmos MLEM e OSEM em GPU produz resultados similares aos produzidos anteriormente em CPU. Como não temos informação prévia sobre a fisiologia do rato, apenas será calculado o *NME* e serão apresentadas imagens do rato com cada uma das implementações.

A reconstrução de imagem foi realizada a partir das projecções obtidas a partir de quatro posições de aquisição: 90°, 135°, 180° e 225°.

6.3.1 Erro médio absoluto normalizado

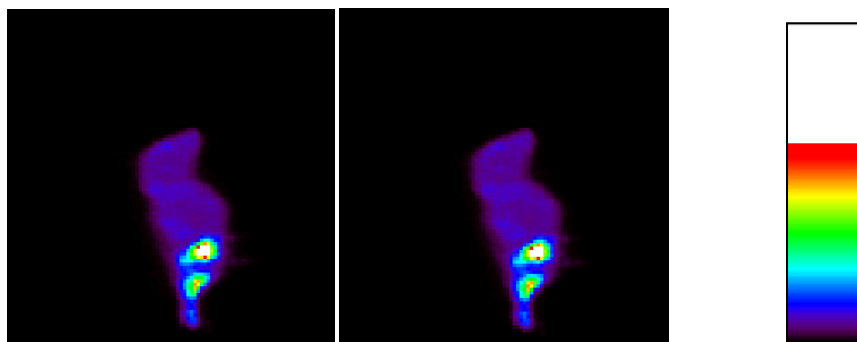
Para as 10 iterações do MLEM o erro médio absoluto normalizado foi de 1,11%.

Para as 4 iterações do OSEM o erro médio absoluto normalizado foi de 0,98%.

Pela primeira vez neste trabalho há informação que indica que, com quatro posições de aquisição, há diferenças, ainda que mínimas, entre imagens de CPU e de GPU.

6.3.2 Imagem reconstruída

Na Figura 6.16 encontram-se as imagens reconstruídas do pré-clínico de um rato após 10 iterações com o algoritmo MLEM, implementado em CPU e GPU. Tendo como base a observação das imagens, as imagens não apresentam diferenças entre si. Na Figura 6.16 encontram-se as imagens reconstruídas do pré-clínico de um rato após 10 iterações com o algoritmo MLEM, implementado em CPU e GPU. Tendo como base a observação das imagens, estas parecem similares.



6.17 - Plano coronal central do pré-clínico de um rato após 10 iterações (MLEM) com CPU (esquerda) e GPU (direita)

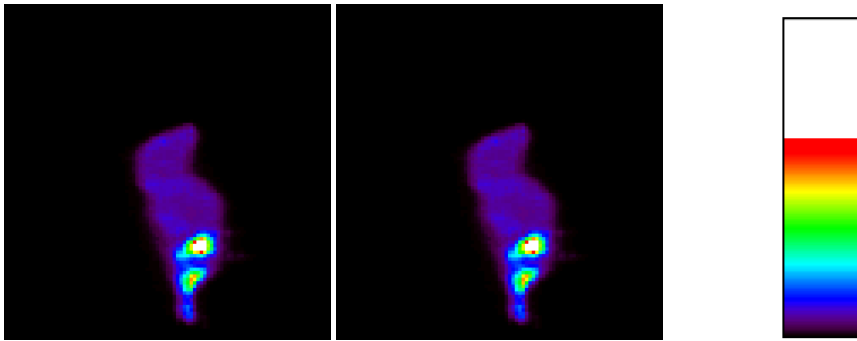


Figura 6.18 - Plano coronal central do pré-clínico de um rato após 4 iterações (OSEM) com CPU (esquerda) e GPU (direita)

Apesar do erro médio absoluto normalizado mostrar que há diferenças entre as duas imagens, isso não é visível para um observador. Mais testes às imagens terão de ser feitos para detectar o erro existente.

6.4 Tempos de computação

O objectivo principal deste trabalho era acelerar a implementação dos algoritmos OSEM e MLEM. Na Tabela 6.1 são apresentados os tempos de computação para todos os testes realizados neste trabalho ao fim de 10 iterações. O factor de aceleração é dado pela divisão entre o tempo gasto em CPU e o tempo gasto em GPU.

Tabela 6.1 - Tempos de computação para os algoritmos MLEM e OSEM com CPU e GPU

	MLEM			OSEM		
	$t_{CPU}(s)$	$t_{GPU}(s)$	Factor de aceleração	$t_{CPU}(s)$	$t_{GPU}(s)$	Factor de aceleração
Fonte pontual	2002	68	29,44	3755	141	26,63
Fantoma de mama	1296	45	28,80	2310	85	27,18
Pré-clínico de um rato	2211	76	29,09	4097	150	27,31

Da observação da concluímos que as reconstruções do fantoma de mama são mais rápidas que as da fonte pontual e do pré-clínico. Este facto deve-se ao fantoma de mama só ter duas posições de aquisição, pelo que os cálculos efectuados passam para quase metade. O aspecto a reter é que no algoritmo MLEM, o GPU apresenta um factor de aceleração, aproximadamente, 29. Isto significa que o código em GPU para o algoritmo MLEM é 29 vezes mais rápido que o do CPU, nas condições em que decorreu este trabalho. Para o OSEM, o factor de aceleração é de, aproximadamente, 27. Logo, o código implementado em GPU é 27 vezes mais rápido do que aquele implementado em CPU, mais propriamente em linguagem C.

7 Discussão Final e Conclusões

Os resultados obtidos neste trabalho devem ser considerados muito positivos, visto que se atingiu um factor de aceleração máximo de 29 vezes, sem que se notem à vista desarmada erros na reconstrução da imagem ou até diferenças para as imagens obtidas a partir do código desenvolvido em CPU, mais concretamente em linguagem C. Em relação à comparação das imagens penso que foram efectuados testes que permitem garantir que a reconstrução está a ser feita como se pretendia, isto é, a mais próxima ou igual possível à realizada pelo código de CPU. Em relação à reconstrução com projecções obtidas a partir de duas posições de aquisição penso que não devem restar dúvidas que as imagens são o mais similar possível, visto que com processos estatísticos, não devemos pensar e assumir que as imagens são iguais.

Por outro lado, na reconstrução com projecções obtidas a partir de quatro posições de aquisição foi detectada uma anomalia com o cálculo do erro médio absoluto normalizado e, apesar de não ser um erro que tenha sido detectado visualmente até ao momento (ainda só foi detectado através de cálculos), será alvo da maior atenção para que possa ser corrigido o quanto antes. As hipóteses mais prováveis para este erro são na rotação dos objectos e na própria matriz de sistema, mas, por enquanto, são apenas suposições.

Falando um pouco sobre os testes efectuados, eles não foram feitos de forma exaustiva e muitos mais se poderiam ter feito, no entanto, o objectivo destes testes não era avaliar a

imagem por si, mas sim comparar esses parâmetros testados com os obtidos pela implementação em CPU.

Em relação ao factor de aceleração ser superior no algoritmo MLEM, é perfeitamente justificável, porque são efectuados processos de ordenação no algoritmo OSEM antes do processo iterativo se iniciar, que não são realizados no algoritmo MLEM. Em relação ao facto da reconstrução do fantoma de mama ter sido mais rápido que os outros, isso deve-se a só ter duas posições de aquisição, que fará com que o número de cálculos efectuados se reduza para, aproximadamente, metade.

Esta aceleração no tempo de computação destes algoritmos iterativos permite que se mantenha a perspectiva de que os algoritmos iterativos venham a ter um papel cada vez mais importante na Imagiologia em geral e na Medicina Nuclear em particular.

Por fim, é importante referir que, caso existisse um aglomerado de GPUs ou GPUs com maior capacidade poderia ser possível paralelizar, por exemplo uma sub-iteração inteira, ou pelo menos, conjuntos de dados cada vez maiores, isso faria com que o factor de aceleração fosse ainda maior e se atingissem resultados extraordinários.

Por fim, penso que esta é uma área onde se deve apostar forte, porque existe ainda um grande caminho a percorrer na programação em paralelo e existem ainda bastantes algoritmos que poderiam ser paralelizados e tornariam o dia-a-dia das pessoas mais simples, mais rápido e mais eficiente.

Bibliografia

1. Siegel, R., et al., *Cancer statistics, 2011*. CA: A Cancer Journal for Clinicians, 2011. **61**(4): p. 212-236.
2. Townsend, D.W. and Cherry, S.R., *Combining anatomy and function: the path to true image fusion*. European Radiology, 2001. **11**(10): p. 1968-74.
3. Pogrel, M.A., et al., *A comparison of single-photon emission computed tomography and planar imaging for quantitative skeletal scintigraphy of the mandibular condyle*. Oral Surgery, Oral Medicine, Oral Pathology, Oral Radiology, and Endodontology, 1995. **80**(2): p. 226-231.
4. Alavi, A. and Basu, S., *Planar and SPECT imaging in the era of PET and PET-CT: can it survive the test of time?* European Journal of Nuclear Medicine and Molecular Imaging, 2008. **35**(8): p. 1554-1559.
5. Grissom, M.P., *Radiation protection and dosimetry: an introduction to health physics*. Radiation Protection Dosimetry, 2010. **138**(4): p. 407-409.
6. Jadvar, H. and Parker, J.A., *Clinical PET and PET/CT*. 1st Ed. Springer ed. 2005.
7. Powsner, R.A., Powsner, E.R., and *Essential Nuclear Medicine Physics*. 2nd Edition Wiley-Blackwell ed. 2006.
8. Levin, C.S. and Hoffman, E.J., *Calculation of positron range and its effect on the fundamental limit of positron emission tomography system spatial resolution*. Physics in Medicine and Biology, 1999. **44**(3): p. 781.
9. Lecoq, P. and Varela, J., *Clear-PEM, a dedicated PET camera for mammography*. Nuclear Instruments & Methods in Physics Research Section A - Accelerators Spectrometers Detectors and Associated Equipment, 2002. **486**(1-2): p. 1-6.
10. Thompson, C.J., et al., *Feasibility study for positron emission mammography*. Medical Physics, 1994. **21**(4): p. 529-38.
11. Thompson, C.J., et al., *Positron Emission Mammography (Pem) - a Promising Technique for Detecting Breast-Cancer*. IEEE Transactions on Nuclear Science, 1995. **42**(4): p. 1012-1017.
12. Weinberg, I., et al., *Preliminary results for positron emission mammography: Real-time functional breast imaging in a conventional mammography gantry*. European Journal of Nuclear Medicine and Molecular Imaging, 1996. **23**(7): p. 804-806.
13. Murthy, K., et al., *Results of preliminary clinical trials of the positron emission mammography system PEM-I: a dedicated breast imaging system producing glucose metabolic images using FDG*. Journal of Nuclear Medicine, 2000. **41**(11): p. 1851-8.
14. Doshi, N.K., et al., *Design and evaluation of an LSO PET detector for breast cancer imaging*. Medical Physics, 2000. **27**(7): p. 1535-43.
15. Freifelder, R. and Karp, J.S., *Dedicated PET scanners for breast imaging*. Physics in Medicine and Biology, 1997. **42**(12): p. 2463-80.
16. Raylman, R.R., et al., *Positron emission mammography-guided breast biopsy*. Journal of Nuclear Medicine, 2001. **42**(6): p. 960-6.
17. Doshi, N.K., et al., *maxPET: A dedicated mammary and axillary region PET imaging system for breast cancer*. IEEE Transactions on Nuclear Science, 2001. **48**(3): p. 811-815.

18. Huber, J.S., et al., *Development of the LBNL positron emission mammography camera*. IEEE Transactions on Nuclear Science, 2003. **50**(5): p. 1650-1653.
19. Wang, G.C., et al., *Characterization of the LBNL PEM camera*. IEEE Transactions on Nuclear Science, 2006. **53**(3): p. 1129-1135.
20. Abreu, M.C., et al., *Design and Evaluation of the Clear-PEM Scanner for Positron Emission Mammography*. IEEE Transactions on Nuclear Science, 2006. **53**(1): p. 71-77.
21. Santos, A.I., et al., *Design and evaluation of the clear-PEM detector for positron emission mammography*, in *2004 IEEE Nuclear Science Symposium Conference Record, Vols 1-7*. 2004. p. 3805-3809.
22. Vandenberghe, S., et al., *Iterative reconstruction algorithms in nuclear medicine*. Computerized Medical Imaging and Graphics, 2001. **25**(2): p. 105-11.
23. Matela, N., *2D Iterative Image Reconstruction for a Dual Planar Detector for Positron Emission Mammography*. 2008, Universidade de Lisboa - Faculdade de Ciências: Lisboa. p. 251.
24. Edholm, P., Herman, G.T., and Roberts, D.A., *Image-Reconstruction from Linograms - Implementation and Evaluation*. IEEE Transactions on Medical Imaging, 1988. **7**(3): p. 239-246.
25. Edholm, P.R. and Herman, G.T., *Linograms in Image-Reconstruction from Projections*. IEEE Transactions on Medical Imaging, 1987. **6**(4): p. 301-307.
26. Hutton, B.F., Nuyts, J., and Zaidi, H., *Iterative Reconstruction Methods Quantitative Analysis in Nuclear Medicine Imaging*, H. Zaidi, Editor. 2006, Springer US. p. 107-140.
27. Wernick, M.N. and Aarsvold, J.N., *Emission Tomography - The Fundamentals of PET and SPECT*. Elsevier Academic Press, 2004.
28. Kim, K.S. and Ye, J.C., *Fully 3D iterative scatter-corrected OSEM for HRRT PET using a GPU*. Physics in Medicine and Biology, 2011. **56**(15): p. 4991-5009.
29. Xu, F. and Mueller, K., *Accelerating popular tomographic reconstruction algorithms on commodity PC graphics hardware*. Ieee Transactions on Nuclear Science, 2005. **52**(3): p. 654-663.
30. Nguyen, V.G., Lee, S.J., and Lee, M.N., *GPU-accelerated 3D Bayesian image reconstruction from Compton scattered data*. Physics in Medicine and Biology, 2011. **56**(9): p. 2817-2836.
31. Lewitt, R.M. and Matej, S., *Overview of methods for image reconstruction from projections in emission computed tomography*. Proceedings of the Ieee, 2003. **91**(10): p. 1588-1611.
32. Fessler, J.A., *Penalized weighted least-squares image reconstruction for positron emission tomography*. Medical Imaging, IEEE Transactions on, 1994. **13**(2): p. 290-300.
33. Gordon, R., *Tutorial on Art - Algebraic Reconstruction Techniques*. IEEE Transactions on Nuclear Science, 1974. **21**(3): p. 78-93.
34. Dempster, A.P., Laird, N.M., and Rubin, D.B., *Maximum Likelihood from Incomplete Data Via Em Algorithm*. Journal of the Royal Statistical Society Series B-Methodological, 1977. **39**(1): p. 1-38.
35. Shepp, L.A. and Vardi, Y., *Maximum Likelihood Reconstruction for Emission Tomography*. IEEE Transactions on Medical Imaging, 1982. **MI-1**(2): p. 113-122.
36. Lange, K. and Carson, R., *EM reconstruction algorithms for emission and transmission tomography*. Journal of Computer Assisted Tomography, 1984. **8**(2): p. 306-16.

37. Hudson, H.M. and Larkin, R.S., *Accelerated Image-Reconstruction Using Ordered Subsets of Projection Data*. IEEE Transactions on Medical Imaging, 1994. **13**(4): p. 601-609.
38. Kirk, D. and Hwu, W., *Programming massively parallel processors: a hands-on approach*. 2010: Morgan Kaufmann Publishers.
39. NVIDIA, *NVIDIA CUDA C Programming Guide Version 4.0*. 2011.
40. NVIDIA, *CUDA Toolkit 3.2 Downloads*. 2011(acedido em 27 de Fevereiro de 2011).
41. Thrust, *Thrust*. 2011(acedido em 29 de Março de 2011).
42. Oliveira, N., *Visualização e Análise de Imagens em Mamografia por Emissão de Positrões*. 2004, Universidade de Lisboa, Faculdade de Ciências: Lisboa.
43. Matela, N., et al., *System matrix calculation for Clear-PEM using ART and linograms*, in *2004 IEEE Nuclear Science Symposium Conference Record, Vols 1-7*. 2004. p. 2601-2604.
44. Hwu, W.M.W., *Gpu Computing Gems: Jade Edition*. 2011: Elsevier Science & Technology.
45. Hoberock, J. and Bell, N., *Thrust*. 2011, NVIDIA.
46. Marieb, E.N. and Hoehn, K., eds. *Human anatomy & physiology*. 2007, Pearson Education.
47. Valk, P.E., Delbeke, D., and Bailey, D.L., *Positron emission tomography: clinical practice*. 2006: Springer.
48. Jatoi, I., Kaufmann, M., and Petit, J.Y., *Atlas of breast surgery*. 2006: Springer.
49. Trindade, A., *Design and Evaluation of a Positron Emission Tomograph for Breast Cancer Imaging*. 2007, Universidade Técnica de Lisboa - Instituto Superior Técnico: Lisboa. p. 236.