ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
DEPARTAMENTO DE ENGENHARIA MECÂNICA

Tomografia por Óptica Difusa – Protótipo de 16 canais

Trabalho de conclusão de curso apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do título de Graduação em Engenharia.

Elder Rubens Silveira Rampazzo Filho
Marcelo Idel Vasserman

Orientador: Raul Gonzalez Lima

Área de Concentração:
Engenharia Mecânica

São Paulo – 2010

FICHA CATALOGRÁFICA

Rampazzo Filho, Elder Rubens Silveira
Tomografia por óptica difusa: protótipo de 16 canais / E.R.S. Rampazzo Filho; M.I. Vasserman. -- São Paulo, 2010.
61 p.

Trabalho de Formatura - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecânica.

1. Bioengenharia 2.Tomografia 2. Método dos elementos finitos I. Vasserman, Marcelo Idel II. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecânica III. t.

SUMÁRIO

LISTA DE FIGURAS

Lista de Tabelas

1 INTRODUÇÃO

É evidente a utilidade dos aparelhos de tomografia nas áreas médicas e veterinárias. A obtenção de informações precisas sem causar dano ao paciente é de extremo valor para um diagnóstico preciso. Algumas informações só podem ser obtidas de forma confiável por tomografias, como, por exemplo, a existência de coágulos ou atelectasia de alvéolos pulmonares.

Atualmente existem diversos métodos de diagnóstico por imagem. Dentre eles é possível citar o Raio-X (RX), a ultrassonografia (US), a ressonância magnética (RM) e a tomografia. Todas estas técnicas fundamentam-se em determinados princípios físicos que determinam a sua aplicabilidade.

O Raio-X, por exemplo, é um equipamento que emite fótons energéticos o suficiente para transpassar a maioria dos tecidos corpóreos, com exceção dos ossos, permitindo visualizar com boa definição o complexo esquelético de um indivíduo. Já a ultrassonografia fundamenta-se em dois efeitos sonoros distintos, o eco e o Efeito Doppler. É utilizada para avaliar os órgãos que se localizam na cavidade abdominal e torácica de um paciente.

A tomografia e a ressonância magnética são técnicas que geram imagens semelhantes, no entanto diferem no princípio de funcionamento. A ressonância magnética fundamenta-se na absorção ressonante de energia eletromagnética, enquanto a tomografia se fundamenta em diversos outros princípios, tais como computadorizada (Raios-X), por impedância elétrica (diferença entre potenciais elétricos) e por óptica difusa.

A tomografia por óptica difusa (TOD) constrói imagens pela análise do comportamento de um feixe de fótons ao atravessar um corpo. Segundo NISSILÄ et al., 2005[1], sua aplicabilidade é variada, mas pode ser separada em três áreas: estudos musculares, mamografia óptica e geração de imagens cerebrais. O objetivo primário para estudos musculares é investigar o metabolismo e suas variações ao longo do tempo. Quando se considera a mamografia óptica, tem-se em vista substituir a técnica atual de RX, uma vez que os RX podem prejudicar o paciente. Por fim a varredura (*scanning*) cerebral pode ser usado para analisar a hemodinâmica de específicas áreas do cérebro, através de atividades que estimulem tais regiões.

Conquanto possibilite resolução espacial relativamente baixa quando comparada à tomografia computadorizada (TC) ou a uma chapa de RX, a TOD apresenta como grande diferencial a capacidade de fornecer informações **funcionais** da estrutura analisada, tal como a perfusão ou a oxigenação de um tecido.

NISSILÄ et al., 2005[1], afirma a existência de inúmeras vantagens, além da supracitada, para um ToOD:

- Modalidade de exame não invasiva, o que implica em uma baixa taxa de complicações;
- O fato do exame não ser influenciado por campos magnéticos externos, o que reduz gastos com isolamentos como os necessários para um aparelho de TC ou de RM;
- A possibilidade de possuir um equipamento portátil para evitar deslocamentos excessivos de pacientes; e
- A possibilidade de ser usado para exames neonatais.

Com vista às vantagens que um equipamento de TOD pode proporcionar aos diagnósticos clínicos, fica evidente a grande utilidade de um equipamento com tal princípio de funcionamento.

2 OBJETIVOS

O objetivo principal deste Trabalho de Conclusão de Curso é a elaboração do protótipo de um tomógrafo de óptica difusa (ToOD).

Como parâmetro de projeto, deseja-se que o mesmo tenha 16 canais emissor-receptor, o que capacita geração de imagens com boa resolução. Adicionalmente, intenta-se como aplicação a avaliação da perfusão em determinado tecido, o que torna as propriedades da hemoglobina norteadoras na seleção do equipamento adequado.

Como critério, o bom funcionamento do tomógrafo será definido pela sua qualidade de determinação da presença e geometria de uma perturbação inserida no interior do domínio de estudo. Esta será comparada com outros projetos já realizados, como os de NISSILÄ et al, 2005[1], e NOGUEIRA, 2007[2].

Serão realizados testes apenas *in vitro*, sem o contato direto com tecidos biológicos.

3 REVISÃO DE MATERIAL TÉCNICO

3.1 Introdução ao princípio de funcionamento

O princípio da óptica difusa é conduzir através de um corpo feixes de fótons para que, a partir dos fenômenos óticos por estes sofridos, seja possível obter informações a respeito da natureza do corpo estudado. Intenta-se determinar a distribuição de propriedades ópticas no interior do corpo de estudo baseando-se em medições no contorno. Luz adentra o corpo devido a fontes luminosas instaladas na fronteira, enquanto detectores medem a quantidade de fótons que atravessa a superfície.

Quando um objeto é iluminado por uma fonte luminosa, alguns fenômenos podem ocorrer com o feixe (reflexão, difração, absorção ou difusão) durante seu percurso, dependendo da natureza do material do corpo em estudo.

A tomografia por óptica difusa (TOD) gera imagens baseando-se no princípio de difusão da luz, a partir do qual se pode determinar a geometria e a estrutura interna do objeto iluminado. A Figura 1 a seguir ilustra o princípio da óptica difusa:

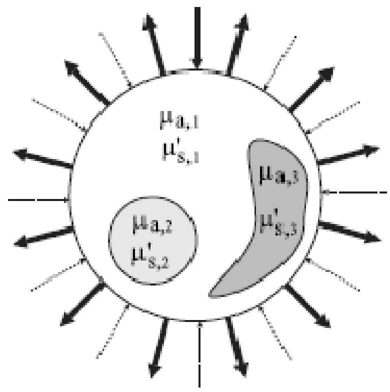


Figura 1 – Princípio da tomografia óptica – extraído de NISSILÄ et al., 2005[1].

As setas de direção radial e sentido para o centro do corpo representam fontes de luz, tais que as estreitas indicam fontes desligadas e a espessa representa uma fonte ligada. As setas que apontam no sentido exterior do corpo representam receptores.

Tendo em vista que a aplicação do tomógrafo é dirigida para o estudo das estruturas internas do corpo humano, os tecidos que o compõem serão os alvos de estudo.

Ondas eletromagnéticas, cujo comprimento se aproxima daquele de ondas infravermelhas ("Near-Infrared, NIR, light"), penetram facilmente nos tecidos corpóreos devido à chamada "janela óptica de tecido" que se encontra na faixa de comprimento de onda entre $650nm$ a $950nm$. Um componente que absorve esse tipo de radiação luminosa é a hemoglobina. NISSILÄ et al., 2005[1], prova que é possível monitorar mudanças no volume sanguíneo e na oxigenação do cérebro através de medidas da atenuação da luz infra-vermelha em diversas faixas de comprimento de ondas. Isso pode ser estendido aos outros tecidos do corpo humano.

3.2 Propriedades ópticas dos tecidos

A grandeza que é medida nos fotodiodos (receptores) do equipamento é a atenuação da luz que atravessou o corpo de prova. A atenuação do raio é dada, principalmente, pela sobreposição dos efeitos de absorção e espalhamento que ocorrem durante a propagação dos feixes.

3.2.1. Absorção

Segundo NISSILÄ et al, 2005[1], a absorção é normalmente devida a um cromóforo, a parte da cadeia molecular de determinada substância responsável pela coloração da mesma. Normalmente, a energia absorvida é convertida em energia térmica. As propriedades de absorção de um cromóforo variam em função do meio em que se encontra, como constituição da molécula hospedeira, pH e temperatura.

A abundante presença de água nos organismos torna essa substância um pólo de absorção significativo de ondas NIR. Por sorte, este composto tem uma depressão na taxa de absorção de luz na faixa compreendida entre $200nm$ e $950nm$.

Lipídios têm propriedades semelhantes às da água, mas como sua concentração em geral nos tecidos é baixa, estes não contribuem grandemente na atenuação do feixe. Essa verificação não cabe à tecidos adiposos, os quais são basicamente constituídos de lipídios.

Nos tecidos superficiais, a absorção de luz, é relativamente alta, especialmente para o espectro ultravioleta. Para o espectro NIR, a melanina é o cromóforo dominante na absorção dessa radiação. No entanto, como a pele é pouco espessa, a contribuição total para a atenuação da luz é baixa.

Há tecidos cuja absorção de luz local referente aos elementos citados acima (água, lipídios e tecidos superficiais) é pequena, enquanto que a concentração de hemoglobina pode ser bastante significativa. Estes tecidos terão propriedades absortivas dependentes da concentração local de oxi e desoxihemoglobinas, cujos coeficientes de absorção são bem distintos, na região de luz NIR. Tal fenômeno pode ser observado na Figura 2. Desta depreende-se que o coeficiente de absorção para a oxihemoglobina é relativamente baixo para a região abaixo de $650nm$. Esse fato reduz a faixa de ondas NIR que pode ser usada para a medição de estruturas espessas (da ordem de cm), para $650nm$ a $950nm$. Estudos da variação da concentração local de oxi e desoxihemoglobina podem ser conduzidos sabendo-se o percurso percorrido por fótons através de determinado tecido e aplicando-se pelo menos dois comprimentos de onda distintos.

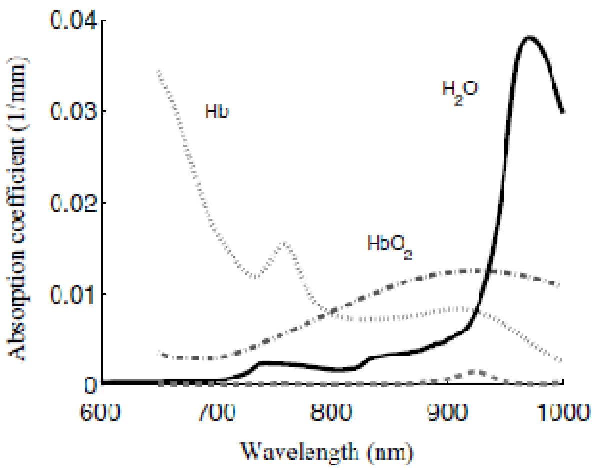


Figura 2 – Coeficientes de absorção para a água (H_2O), oxihemoglobina HbO_2, desoxihemoglobina (Hb) e lipídios (abaixo), em escala – extraído de NISSILÄ et al, 2005[1].

3.2.2. Espalhamento

Por afirmação de NISSILÄ et al, 2005[1], o espalhamento é o efeito predominante, quando se trata da interação entre o tecido e a luz que o percorre. Este efeito ocorre nas interfaces de tecido, onde há uma mudança no coeficiente de difração n. Estas interfaces podem ser, por exemplo, entre fluídos intra e extracelulares ou entre o citoplasma e os líquidos internos das organelas. Macroscopicamente, as diferenças para tais índices são insignificantes, com exceção da interface entre o ar e a pele onde há uma variação elevada deste. Logo, considera-se n como constante dentro dos tecidos.

O espalhamento é dado principalmente em função da dimensão das partículas presentes no meio de estudo, ou seja, quão próximas estas são do comprimento de onda do feixe emitido. Membranas celulares, por exemplo, representam uma porção grande do total de tecido irradiado e contribuem significativamente para o espalhamento da luz. Já as mitocôndrias, também como exemplo, têm dimensões adequadas para o espalhamento de luz NIR.

A estrutura macroscópica de alguns tecidos também pode promover uma propagação preferencial da luz. Em tecidos musculares a luz propaga-se no sentido de suas fibras, o mesmo ocorre para a região branca do cérebro no sentido das conexões neurais.

A Tabela 1 apresenta uma visão interessante a respeito de parâmetros de alguns tecidos:

Tabela 1 – Propriedades ópticas de alguns tecidos – adaptada de NISSILÄ et al, 2005[1].

Tecido	Coef. de absorção (mm^{-1})	Coef. de espalhamento (mm^{-1})	Comprimento de onda λ (nm)
Músculo (abdômen)	$0{,}0052 - 0{,}017$	$0{,}64 - 0{,}95$	$674 - 956$
Região Cinzenta (cérebro)	$0{,}0090 - 0{,}026$	$0{,}42 - 1{,}2$	$674 - 956$
Região Branca (cérebro)	$0{,}013 - 0{,}097$	$0{,}68 - 1{,}5$	$674 - 956$
Sangue	$0{,}13 - 0{,}49$	$2{,}5 - 4{,}0$	$665 - 960$
Osso (crânio humano)	$0{,}02 - 0{,}07$	$0{,}75 - 1{,}2$	$674 - 956$
Pele (branca)	$0{,}0053 - 0{,}049$	$1{,}3 - 3{,}4$	$618 - 950$
Gordura subcutânea	$0{,}0040 - 0{,}024$	$0{,}8 - 1{,}7$	$917 - 949$

Perceba que o coeficiente de espalhamento é, em geral, ao menos duas ordens de grandeza superior ao de absorção, o que torna aquele o fenômeno óptico dominante sofrido pelos fótons. Contudo, o mesmo não é útil para a geração de imagens já que, como afirmado acima, diferenças de coeficiente de espalhamento são insignificantes na maioria dos casos. Assim sendo, a TOD baseia-se na análise de diferenças nos coeficientes de absorção, enquanto os de espalhamento são considerados constantes em todo o domínio.

3.3 Geração de imagens

Como mencionado, a geração de imagens em TOD baseia-se na determinação da distribuição de propriedades ópticas no interior de um domínio a partir de medições de fluxos luminosos de entrada e saída que atravessam sua superfície limítrofe. Para definir tais, técnicas de reconstrução iterativas são aplicadas. Caracteriza-se, então, um típico caso de resolução de *Problemas Direto e Inverso*, temas clássicos de abordagem pelo Método dos Elementos Finitos.

3.3.1. Os fenômenos envolvidos e a equação que os rege

Segundo NISSILÄ et al, 2005[1], a propagação de ondas eletromagnéticas em um meio é tradicionalmente modelada mediante aplicação das Equações de Maxwell. Entretanto, estas são dispensáveis no estudo do comportamento dos fótons em tomografia por óptica difusa, tendo em vista que na maioria dos tecidos o espalhamento de fótons ocorre fortemente em comprimentos de onda NIR. Neste caso, fenômenos como polarização e interferência são desconsiderados já que a luz atravessa apenas poucos milímetros do meio. Adicionalmente, são consideradas desprezíveis as contribuições de fenômenos de espalhamento inelásticos, como a fosforescência, fluorescência e espalhamento de Raman.

Em suma, a modelagem aplicada baseia-se que nos fenômenos de espalhamento e absorção dos fótons. NOGUEIRA, 2007[2], e ARRIDGE, 1999[3], afirmam que a análise pode ocorrer sobre um domínio convexo $\Omega \in \Re^3$, que abrange o corpo ao qual se intenta a obtenção de imagens. Os conjuntos de emissores e receptores localizam-se na fronteira $\partial\Omega$ deste domínio.

Pela ação dos emissores, fótons penetram a região Ω. Estes serão descritos pela grandeza densidade de fótons $\phi(r,\theta,t)$ em um ponto qualquer $r \in \Omega$, em um instante de tempo t e velocidade. A grandeza $\phi(r,\theta,t)$ é expressa por m^{-3}. θ é uma grandeza polar, vetorialmente representada por:

$$\theta = \begin{bmatrix} \sin\alpha \,.\, \cos\beta \\ \sin\alpha \,.\, \sin\beta \\ \cos\alpha \end{bmatrix} \quad (1)$$

onde α é o ângulo de projeção de θ no plano xy e β é o ângulo de θ perpendicular a xy.

Pode-se determinar a densidade de fótons $\Phi(r,t)$ em um ponto r do domínio em um instante de tempo t pela integração de $\phi(r,\theta,t)$ em todas as direções possíveis de θ, representadas por uma região S^2:

$$\Phi(r,t) = \iint_{S^2} \phi(r,\theta,t)d\theta \quad (2)$$

Baseada nessa grandeza $\Phi(r,t)$ é enunciada a Equação de Transporte de Boltzmann que rege a propagação de fótons por um meio homogêneo. Sua aproximação é dada por:

$$-\nabla.\{D\nabla[\Phi(r,t)]\} + \mu_a \Phi(r,t) + \frac{1}{c}\frac{\partial\Phi(r,t)}{\partial t} = q_0 \quad (3)$$

As grandezas envolvidas nesta expressão representam:

- $D(r)$ é o coeficiente de difusão, expresso em m e definido por:

$$D(r) = \frac{1}{3.(\mu_a + \mu_s')} \quad (4)$$

- c é a velocidade da luz no meio, expressa em $^{m}/_{s}$;
- $\mu_a(r)$ é o coeficiente de absorção, expresso em m^{-1};
- $\mu_s^{'}(r)$ é o coeficiente de espalhamento reduzido, expresso em m^{-1} e definido por:

$$\mu_s^{'} = (1-g)\mu_s \quad (5)$$

- μ_s é o coeficiente de espalhamento, expresso em m^{-1};
- g é o cosseno esperado do ângulo de espalhamento;
- $q_0 = q_0(r,t)$ é a emissão de uma fonte de fótons, expressa em m^{-4}.

Segundo NISSILÄ et al, 2005[1], os coeficientes μ_a e μ_s supracitados representam a probabilidade, por unidade de comprimento, de um fóton ser absorvido ou espalhado, respectivamente.

O espalhamento em tecidos normalmente é anisotrópico. Por este motivo é necessária a correção de μ_s para $\mu_s^{'}$ pela aplicação do parâmetro g. Este fornece uma noção qualitativa do grau de anisotropia do meio, sendo que $g = 0$ para meios isotrópicos, $g \to 1$ quando o espalhamento desaparece e $g \to -1$ para situações em que o espalhamento é completamente dirigido para trás. É freqüente denominar g por fator de anisotropia.

Conquanto a equação descreva o fenômeno físico evidenciado, ela é inadequada para a aplicação de resolução pelo método de elementos finitos. NOGUEIRA, 2007[2], ARRIDGE, 1999[3], e ZIENKIEWICZ; TAYLOR[4], 1989, recomendam a transformação desta em uma Equação de Helmholtz não-homogênea, já que esta é mais facilmente resolvida computacionalmente. Para tanto, NOGUEIRA, 2007[2], e ARRIDGE, 1999[3], recomendam alguns operações matemáticas. Primeiramente, o primeiro termo da Equação (3) é transformado usando-se um artifício e uma das propriedades do operador diferencial ∇.

$$-\nabla.\left[\sqrt{D}\sqrt{D}\ \nabla\Phi\right]+\mu_a\Phi+\frac{1}{c}\frac{\partial\Phi}{\partial t}=q_0 \tag{6}$$

$$-\nabla\left(\sqrt{D}\right).\sqrt{D}\nabla(\Phi)-\sqrt{D}\nabla\cdot\left[\sqrt{D}\ \nabla(\Phi)\right]+\mu_a\Phi+\frac{1}{c}\frac{\partial\Phi}{\partial t}=q_0 \tag{7}$$

$$-\nabla\left(\sqrt{D}\right).\sqrt{D}\nabla(\Phi)-\sqrt{D}\left[\nabla\left(\sqrt{D}\right).\nabla(\Phi)+\sqrt{D}\nabla^2(\Phi)\right]+\mu_a\Phi+\frac{1}{c}\frac{\partial\Phi}{\partial t}=q_0 \tag{8}$$

$$-D\nabla^2(\Phi)-2\sqrt{D}\nabla\left(\sqrt{D}\right)\cdot\nabla(\Phi)+\mu_a\Phi+\frac{1}{c}\frac{\partial\Phi}{\partial t}=q_0 \tag{9}$$

Em seguida, faz-se uma mudança de variável:

$$U=U(r,t)=\sqrt{D(r)}\ \Phi(r,t) \tag{10}$$

O laplaciano desta nova variável é dado por:

$$\nabla^2U=\nabla^2\left(\sqrt{D}\Phi\right)=\Phi\nabla^2\left(\sqrt{D}\right)+\sqrt{D}\nabla^2\Phi+2\nabla\left(\sqrt{D}\right)\cdot\nabla\Phi \tag{11}$$

Substituindo as Equações (10) e (11) na Equação (9) e colocando-se $\sqrt{D}$ em evidência:

$$-\nabla^2U+\frac{U}{\sqrt{D}}\nabla^2\left(\sqrt{D}\right)+\frac{\mu_a}{\sqrt{D}}\frac{U}{\sqrt{D}}+\frac{1}{c\sqrt{D}}\frac{1}{\sqrt{D}}\frac{\partial U}{\partial t}=\frac{q_0}{\sqrt{D}} \tag{12}$$

$$-\nabla^2U+\left(\frac{\nabla^2\left(\sqrt{D}\right)}{\sqrt{D}}+\frac{\mu_a}{\sqrt{D}}+\frac{1}{cD}\right)U=\frac{q_0}{\sqrt{D}} \tag{13}$$

Substituindo do termo em parênteses por uma nova variável η:

$$\eta=\frac{\nabla^2\left(\sqrt{D}\right)}{\sqrt{D}}+\frac{\mu_a}{\sqrt{D}}+\frac{1}{cD} \tag{14}$$

Obtém-se enfim a expressão para a Equação de Helmholtz:

$$-\nabla^2U+\eta U=\frac{q_0}{\sqrt{D}} \tag{15}$$

Em que os termos U e η são dados pelas Equações (10) e (14), respectivamente.

3.3.2. O Problema Direto

NISSILÄ et al, 2005[1], define o Problema Direto na geração de imagens por óptica difusa por: dadas as distribuições de fontes emissoras de fótons na fronteira do domínio e a do valor dos parâmetros ópticos relacionados, determina-se o fluxo de fótons resultantes na fronteira.

A solução desta equação é obtida através da aplicação de um princípio variacional. ZIENKIEWICS; TAYLOR, 1989[4], demonstram que uma equação pode ser escrita na forma vetorial como $\underline{Lu} + \underline{b} = 0$, em que $\underline{L}$ é um operador auto-adjunto que admite um funcional escalar do tipo:

$$\Pi = \int_{\Omega} \left(\frac{1}{2} \underline{u}^T \underline{Lu} + \underline{u}^T \underline{b} \right) d\Omega \qquad (16)$$

Vale ressaltar que o domínio $\Omega \in \Re^3$, ou seja, a integração por este domínio é equivalente a integração sobre todo o volume delimitado pela fronteira $\partial\Omega$.

Para a aplicação deste operador Π na Equação de Helmholtz anteriormente descrita tem-se que $\underline{L} = \left[\frac{\partial^2}{\partial x^2}; \frac{\partial^2}{\partial y^2}; \frac{\partial^2}{\partial z^2}; -\eta \right]$ e $\underline{b} = \frac{q_0}{\sqrt{D}}$. Sendo assim:

$$\Pi = \int_{\Omega} \left(\frac{1}{2} U \left(\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} + \frac{\partial^2 U}{\partial z^2} - \eta U \right) + U \frac{q_0}{\sqrt{D}} \right) d\Omega \qquad (17)$$

Como hipóteses, admite-se que a emissão de fótons para o interior do domínio não varia temporalmente (estado estacionário) e que o coeficiente de difusão $D = D(r)$ é constante no interior de um elemento finito. Estas constatações simplificam a expressão de η a $\eta = \frac{\mu_a}{D}$. Após integrar por partes dos três primeiros termos da Equação (17) e substituir de U por sua expressão dada pela Equação (10), a Equação (17) resulta em:

$$\Pi = -\int_{\Omega}\left\{\frac{D}{2}\left[\left(\frac{\partial\Phi}{\partial x}\right)^2 + \left(\frac{\partial\Phi}{\partial y}\right)^2 + \left(\frac{\partial\Phi}{\partial z}\right)^2\right] + \frac{\mu_a}{2}\Phi^2 - \mathrm{q}_0\Phi\right\} d\Omega \tag{18}$$

A solução do problema acima é definir Φ que faça Π estacionário com mudanças arbitrárias em $\delta\Phi$. Em outras palavras, Φ será a função que minimiza Π com respeito a ela mesma. Matematicamente:

$$\frac{\partial \Pi}{\partial \underline{\phi}} = \underline{0} \tag{19}$$

3.3.3. A discretização do domínio

Considera-se que o problema analisado apresente caráter bidimensional, já que as grandezas envolvidas são consideradas independentes da coordenada z. Assim, por recomendação de NOGUEIRA, 2007[2], ZIENKIEWICS; TAYLOR, 1989[4], e LOGAN, 1997[5], os elementos finitos são triangulares. A função será então dada por:

$$\Phi \approx \widehat{\Phi} = a_1 + a_2 x + a_3 y \tag{20}$$

Para que a aproximação seja condizente, os valores de $\widehat{\Phi}(x,y)$ devem ser iguais aos de $\Phi(x,y)$ ao menos nos nós (i,j,m) dos elementos finitos. Assim:

$$\begin{cases} \Phi(x_i,y_i) = \phi_i = \widehat{\Phi}(x_i,y_i) = a_1 + a_2 x_i + a_3 y_i \\ \Phi(x_j,y_j) = \phi_j = \widehat{\Phi}(x_j,y_j) = a_1 + a_2 x_j + a_3 y_j \\ \Phi(x_m,y_m) = \phi_m = \widehat{\Phi}(x_m,y_m) = a_1 + a_2 x_m + a_3 y_m \end{cases} \tag{21}$$

$$\therefore \begin{bmatrix} 1 & x_i & y_i \\ 1 & x_j & y_j \\ 1 & x_m & y_m \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \phi_i \\ \phi_j \\ \phi_m \end{bmatrix} \tag{22}$$

$$\therefore \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 1 & x_i & y_i \\ 1 & x_j & y_j \\ 1 & x_m & y_m \end{bmatrix}^{-1} \begin{bmatrix} \phi_i \\ \phi_j \\ \phi_m \end{bmatrix} \tag{23}$$

A aplicação do método dos cofatores para determinar a matriz inversa detalhada acima resulta em:

$$\begin{bmatrix} 1 & x_i & y_i \\ 1 & x_j & y_j \\ 1 & x_m & y_m \end{bmatrix}^{-1} = \frac{1}{2A}\begin{bmatrix} \alpha_i & \alpha_j & \alpha_m \\ \beta_i & \beta_j & \beta_m \\ \gamma_i & \gamma_j & \gamma_m \end{bmatrix} \tag{24}$$

Em que A representa a área do triângulo definido pelo elemento finito, sendo calculada por:

$$2A = \begin{vmatrix} 1 & x_i & y_i \\ 1 & x_j & y_j \\ 1 & x_m & y_m \end{vmatrix} \tag{25}$$

Os outros coeficientes da Equação (24) são expressos por:

$$\begin{matrix} \alpha_i = x_j y_m - y_j x_m & \alpha_j = x_m y_i - x_i y_m & \alpha_m = x_i y_j - x_j y_i \\ \beta_i = y_j - y_m & \beta_j = y_m - y_i & \beta_m = y_i - y_j \\ \gamma_i = x_m - x_j & \gamma_j = x_i - x_m & \gamma_m = x_j - x_i \end{matrix} \tag{26}$$

Dessa forma, $\widehat{\Phi}(x, y)$ pode ser escrita em notação matricial por:

$$\widehat{\Phi} = [1 \quad x \quad y]\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = [1 \quad x \quad y]\frac{1}{2A}\begin{bmatrix} \alpha_i & \alpha_j & \alpha_m \\ \beta_i & \beta_j & \beta_m \\ \gamma_i & \gamma_j & \gamma_m \end{bmatrix}\begin{bmatrix} \phi_i \\ \phi_j \\ \phi_m \end{bmatrix} \tag{27}$$

$$\therefore \widehat{\Phi} = [N_i \quad N_j \quad N_m]\begin{bmatrix} \phi_i \\ \phi_j \\ \phi_m \end{bmatrix} = \underline{\underline{N}}\,\underline{\phi} \tag{28}$$

Em que $N_k = \frac{1}{2A}(\alpha_k + \beta_k x + \gamma_k y), \forall k \in \{i, j, m\}$.

As derivadas parciais de Φ surgem na determinação de Π na Equação (18). Logo:

$$\begin{bmatrix} \frac{\partial \phi}{\partial x} \\ \frac{\partial \phi}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix}\phi \approx \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix}\hat{\phi} = \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix}[N_i \quad N_j \quad N_m]\begin{bmatrix} \phi_i \\ \phi_j \\ \phi_m \end{bmatrix} \tag{29}$$

$$\therefore \begin{bmatrix} \frac{\partial \phi}{\partial x} \\ \frac{\partial \phi}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial N_i}{\partial x} & \frac{\partial N_j}{\partial x} & \frac{\partial N_m}{\partial x} \\ \frac{\partial N_i}{\partial y} & \frac{\partial N_j}{\partial y} & \frac{\partial N_m}{\partial y} \end{bmatrix}\begin{bmatrix} \phi_i \\ \phi_j \\ \phi_m \end{bmatrix} \tag{30}$$

Aplicando as derivadas parciais na definição de $N_k, \forall k \in \{i,j,m\}$, obtém-se uma matriz B que correlaciona as derivadas parciais de ϕ e os valores da própria ϕ aplicado aos nós:

$$\begin{bmatrix} \frac{\partial \phi}{\partial x} \\ \frac{\partial \phi}{\partial y} \end{bmatrix} = \frac{1}{2A} \begin{bmatrix} \beta_i & \beta_j & \beta_m \\ \gamma_i & \gamma_j & \gamma_m \end{bmatrix} \begin{bmatrix} \phi_i \\ \phi_j \\ \phi_m \end{bmatrix} = \underline{\underline{B}}\underline{\phi} \tag{31}$$

3.3.4. Minimização do funcional Π

Como mencionado anteriormente, a determinação da função Φ está relacionada com a minimização do operador funcional Π. A aplicação da minimização deste funcional é análoga a minimização da "energia" envolvida na análise de alguns problemas. Como ilustração, este procedimento seria análogo a minimização de energia potencial em situações de equilíbrio estático de estruturas na mecânica clássica. Por NOGUEIRA, 2007[2], a substituição da aproximação por elementos finitos, Equação (20), com a consideração de análise bidimensional resulta em:

$$\Pi = -\iiint_\Omega \left\{ \frac{D}{2} \left[\left(\frac{\partial \Phi}{\partial x} \right)^2 + \left(\frac{\partial \Phi}{\partial y} \right)^2 \right] + \frac{\mu_a}{2} \Phi^2 - \mathrm{q}_0 \Phi \right\} d\Omega \tag{32}$$

$$\Pi = -\iiint_\Omega \left\{ \frac{D}{2} \begin{bmatrix} \frac{\partial \Phi}{\partial x} & \frac{\partial \Phi}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial \phi}{\partial x} \\ \frac{\partial \phi}{\partial y} \end{bmatrix} + \frac{\mu_a}{2} \Phi^T \Phi - \mathrm{q}_0 \Phi \right\} d\Omega \tag{33}$$

Aplicando a notação vetorial descrita pela discretização:

$$\Pi = -\iiint_\Omega \left\{ \frac{D}{2} \underline{\phi}^T \underline{\underline{B}}^T \underline{\underline{I}}\, \underline{\underline{B}} \underline{\phi} + \frac{\mu_a}{2} \underline{\phi}^T \underline{NN}^T \underline{\phi} - \mathrm{q}_0 \underline{N}^T \underline{\phi} \right\} d\Omega \tag{34}$$

$$\frac{\partial \Pi}{\partial \underline{\phi}} = -\iiint_\Omega \left\{ \frac{D}{2} \underline{\phi}^T \left(\underline{\underline{B}}^T \underline{\underline{I}}\, \underline{\underline{B}} + \left(\underline{\underline{B}}^T \underline{\underline{I}}\, \underline{\underline{B}} \right)^T \right) + \frac{\mu_a}{2} \underline{\phi}^T \left(\underline{NN}^T + \left(\underline{NN}^T \right)^T \right) - \mathrm{q}_0 \underline{N}^T \right\} d\Omega \tag{35}$$

Devido à simetria de $\underline{\underline{B}}^T \underline{\underline{I}}\, \underline{\underline{B}}$ e $\underline{NN}^T$:

$$\frac{\partial \Pi}{\partial \underline{\phi}} = -\iiint_\Omega \left\{ D \underline{\phi}^T \underline{\underline{B}}^T \underline{\underline{I}}\, \underline{\underline{B}} + \mu_a \underline{\phi}^T \underline{NN}^T - \mathrm{q}_0 \underline{N}^T \right\} d\Omega \tag{36}$$

A minimização consiste em aplicar a relação:

$$\frac{\partial \Pi}{\partial \underline{\phi}} = \underline{0} = -\iiint_{\Omega} \left\{ D\underline{\phi}^T \underline{\underline{B}}^T \underline{\underline{I}}\,\underline{\underline{B}} + \mu_a \underline{\phi}^T \underline{N}\underline{N}^T - \mathrm{q}_0 \underline{N}^T \right\} d\Omega \tag{37}$$

$$\left(\iiint_{\Omega} D\underline{\underline{B}}^T \underline{\underline{I}}\,\underline{\underline{B}} d\Omega + \iiint_{\Omega} \mu_a \underline{N}\underline{N}^T d\Omega \right) \underline{\phi} = \iiint_{\Omega} \mathrm{q}_0 \underline{N}^T d\Omega \tag{38}$$

Assim, a determinação de $\underline{\phi}$ está atrelada a determinação das integrais volumétricas sobre Ω evidenciadas acima. Não é necessário determinar a integral localizada à direita já que não há fontes emissoras de fóton no interior de Ω. As integrações serão realizadas de forma separada e mediante a substituição das relações na definição dos elementos finitos.

- $\iiint_{\Omega} D\underline{\underline{B}}^T \underline{\underline{I}}\,\underline{\underline{B}} d\Omega =$

$$\frac{D}{4A} \begin{bmatrix} \beta_i^2 + \gamma_i^2 & \beta_i\beta_j + \gamma_i\gamma_j & \beta_i\beta_m + \gamma_i\gamma_m \\ \beta_j\beta_i + \gamma_j\gamma_i & \beta_j^2 + \gamma_j^2 & \beta_j\beta_m + \gamma_j\gamma_m \\ \beta_m\beta_i + \gamma_m\gamma_i & \beta_m\beta_j + \gamma_m\gamma_j & \beta_m^2 + \gamma_m^2 \end{bmatrix} t = \frac{Dt}{4A} \underline{\underline{M}} \tag{39}$$

Em que t é a espessura do elemento finito e $\underline{\underline{M}} = M_{p,q} = \beta_p\beta_q + \gamma_p\gamma_q, \forall p,q \in \{i,j,m\}$.

- $$\iiint_{\Omega} \mu_a \underline{N}\underline{N}^T d\Omega \underline{\phi} = \mu_a \iiint_{\Omega} \begin{bmatrix} N_i^2 & N_iN_j & N_iN_m \\ N_jN_i & N_j^2 & N_jN_m \\ N_mN_i & N_mN_j & N_m^2 \end{bmatrix} d\Omega = \frac{\mu_a t}{4A} \underline{\underline{H}} \tag{40}$$

Para $\underline{\underline{H}}$ uma matriz obtida pela integração de um elemento genérico $N_pN_q, \forall p,q \in \{i,j,m\}$ no domínio Ω. Se adotado como origem das coordenadas o centróide do elemento finito:

$$\underline{\underline{H}} = H_{p,q} = \alpha_p\alpha_q + \beta_p\beta_q \frac{x_i^2 + x_j^2 + x_m^2}{12} + \gamma_p\gamma_q \frac{y_i^2 + y_j^2 + y_m^2}{12} + \cdots$$

$$\cdots + \left(\beta_p\gamma_q + \beta_q\gamma_p\right) \frac{x_iy_i + x_jy_j + x_my_m}{12} \tag{41}$$

Logo, a solução do Problema Direto expressa resumidamente é:

$$\left[\frac{Dt}{4A}\underline{\underline{M}} + \frac{\mu_a t}{4A}\underline{\underline{H}}\right]\underline{\phi} = \iiint_{\Omega} q_0 \underline{N}^T d\Omega \quad (42)$$

3.3.5. O Problema Inverso

NISSILÄ et al, 2005[1], enuncia o Problema Inverso para tomográfica por óptica difusa por: dadas as medições dos fluxos de fótons que atravessam a fronteira do domínio e a distribuição de fontes emissoras pela mesma superfície, determina-se a distribuição interna dos parâmetros ópticos envolvidos. Matematicamente, deseja-se determinar a matriz B, que abrange a distribuição dos coeficientes de absorção e espalhamento interiores ao domínio, que relaciona a seguinte equação, descrita por NOGUEIRA, 2007[2], e ZIENKIEWICS; TAYLOR, 1989[4]:

$$\Theta = B\Lambda \quad (42)$$

- Θ é a matriz de perturbações na difusão $D(r)$ de fótons, conseqüentemente nos coeficientes de absorção $\mu_a(r)$. A existência de n elementos finitos na malha implicará uma matriz Θ de grandeza $n\,x\,n$. O elemento $\Theta_{i,j}$ será uma perturbação arbitrária, não-nula, na difusão dos fótons no i-ésimo elemento finito se $i = j$, e nulo se $i \neq j$. Portanto, a matriz Θ é diagonal.
- Λ é uma matriz de perturbações na distribuição de densidade de fótons Φ, captada pelos fotorreceptores. Seja p o número de emissores e q o de receptores, para uma alteração nas propriedades ópticas de em um elemento finito k qualquer (mudança no parâmetro não-nulo da k-ésima coluna de Θ, ou seja, Θ_k) é possível a realização de p distintos testes que geram q leituras. Assim sendo, a matriz Λ é de ordem $pq\ x\ n$.

A solução do Problema Inverso será obtida mediante aplicação do algoritmo Caixa-Preta, descrito por AYA et al, 2006[6]. Este método assume linearidade entre a variação na densidade de fótons Φ e a variação nos coeficientes de absorção μ_a.

Este algoritmo necessita assumir uma distribuição inicial conhecida de μ_a para ser usada como referência. Este é adotado como o estado homogêneo, em que a grandeza $\mu_a(r) = \mu_0$ é constante em todo o volume Ω. A aplicação do Problema Direto, para os casos de ativação individual de cada um dos p emissores e registrando q medidas com os receptores, gera um vetor Λ_0. A resolução do Problema Direto com a perturbação de absorção no k-ésimo elemento finito determina um vetor Λ_k. A k-ésima coluna da matriz Λ será a diferença $\Lambda_k - \Lambda_0$.

A solução da matriz B é dada isolando-se a mesma na expressão inicial. No entanto, dadas as dimensões da matriz Λ, não é possível invertê-la através de algoritmos convencionais. Portanto, para resolver este problema, serão utilizados alguns métodos de regularização que possibilitem a determinação de B com o menor nível de ruído possível.

3.3.6. Métodos de Regularização

AYA et al, 2006[6] sugere alguns métodos de regularização, dos quais três serão ser testados na resolução do problema inverso.

O primeiro, conhecido como *regularização de Tikhonov*, pode ser visto na fórmula a baixo, onde a matriz I é a matriz identidade e α é um escalar:

$$B = \Theta \Lambda^T (\Lambda \Lambda^T + \alpha I)^{-1} \tag{43}$$

O segundo é o *truncamento de valores singulares* que resolverá a seguinte equação, sendo que a matriz truncada é resultado do produto $\Lambda \Lambda^T$:

$$B = \Theta \Lambda^T (\Lambda \Lambda^T)^{-1} \tag{44}$$

A metodologia consiste em transformar essa matriz em um produto de três outras matrizes (U, S e V), sendo a primeira e a ultima ortonormais e S diagonal. Ou seja:

$$\Lambda \Lambda^T = U\, S\, V^T \tag{45}$$

Invertendo o produto $\Lambda\Lambda^T$:

$$(\Lambda\Lambda^T)^{-1} = VS^{-1}U^T \tag{46}$$

Sendo que:

$$S^{-1}(i,i) = \begin{cases} \frac{1}{S(i,i)}, & S(i,i) > \varepsilon \\ 0, & S(i,i) < \varepsilon \end{cases} \tag{47}$$

O terceiro é uma evolução do método de Tikhonov, nomeado de *método heurístico* e é dado como:

$$B = (\Lambda^T\Lambda + \alpha F^T F + \beta M^T M)^{-1}\Theta\Lambda^T \tag{48}$$

Onde a matriz F representa um filtro passa alta que considera cada coluna de B como uma imagem e a matriz Mé uma matriz que melhora a uniformidade da sensibilidade. Os parâmetros α e β são escalares e auxiliam na regularização da imagem.

Um quarto método que não está proposto na literatura é a composição entre o Método da Inversão por Valores Singulares ao Método.

A teoria expressa acima possibilita a solução de B pela aplicação sucessiva do Problema Direto. Determinada a matriz B, pode-se utilizá-la para a geração de imagens pela simples associação de diferentes "*pixels*" a respectivos valores de coeficientes de difusão em B, sendo este o objetivo da TOD. Obviamente, a resolução das imagens está intimamente ligada às quantidades p e q de emissores e receptores, ao meio de análise e à perturbação introduzida.

4 METODOLOGIA

A construção do ToOD divide-se em dois principais grupos de atividades as quais foram desenvolvidas paralelamente: o ***hardware*** e o ***software***. O primeiro é composto pelo protótipo físico em si, além do sistema de aquisição e o procedimento de testes. Já o segundo é responsável pelo controle do hardware e pelo processamento de dados. Os dados serão provenientes dos testes do protótipo, os quais serão descritos nessa seção.

4.1 Hardware

O protótipo é constituído do corpo de estudo, meio que o preenche, a eventual perturbação a ser detectada pelo tomógrafo e as condições externas que melhoram a eficácia dos testes, os quais seguem uma lógica definida e comum em TOD.

O sistema de aquisição de dados do *hardware* é composto por duas partes: o circuito elétrico e o elemento de controle. O circuito elétrico abrange todos os componentes eletrônicos do tomógrafo, que vão desde os pares, por sua vez denominados canais, emissor-receptor até a ligação dos mesmos com o sistema de controle. Este, por sua vez, é o responsável pela interface do circuito elétrico com o computador.

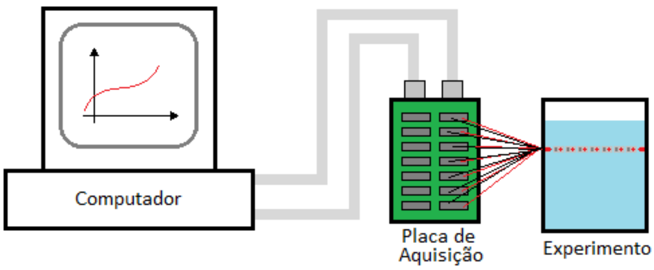


Figura 3 – Diagrama ilustrativo do experimento.

4.1.1. <u>As condições de experimento e os testes</u>

Abrange todos os elementos que definem o corpo de estudo. Este será um objeto oco ao qual serão fixados em seu contorno os elementos óptico-eletrônicos e definirá o domínio do teste. Interiormente, o corpo será preenchido com uma solução fluídica que atua como meio homogêneo de estudo. Um pequeno corpo será inserido no meio, tornando-se uma perturbação a qual se intenta detectar com o uso do aparelho. Adicionalmente, há outros objetos que proporcionam as condições de teste desejadas, como o isolamento luminoso.

Em TOD, a geração de imagens baseia-se na sobreposição de informações obtidas pelos receptores seguindo uma ativação seqüencial de cada um dos emissores de forma isolada. Assim, a lógica desempenhada nos testes é:

1. Com todos os emissores desligados, registrar os valores de voltagem nos receptores;
2. Ativar apenas um LED e registrar os valores obtidos por todos os receptores;
3. Repetir o passo 2 para os outros LEDs isoladamente e de forma seqüencial; e
4. Salvar as informações obtidas em um arquivo.

4.1.2. <u>O circuito elétrico</u>

Os emissores e os receptores são responsáveis, respectivamente, pela inserção de luz e pela medição dos potenciais luminosos. Os emissores são basicamente LEDs (*Light Emitting Diodes*), enquanto os receptores são conversores luz-voltagem (fotodiodos).

A seleção dos emissores e receptores é definida pelo comprimento de onda (λ) característico. Este depende do tipo de aplicação desejada ao tomógrafo. Escolhidos os emissores e receptores, o complemento do circuito elétrico é elaborado com base nas voltagens e correntes de operação dos mesmos (obtido por seus *datasheets*), bem como as mesmas características para os elementos de controle.

4.1.2. O elemento de controle

O sistema de controle consiste em uma placa I/O digital e analógica que liga o circuito elétrico a um computador que contenha o *software*. O controle e calibração da mesma serão feitos com o auxílio do computador.

A placa I/O tem basicamente duas funções. A primeira é a ativação dos emissores, ela converte os comandos digitais do computador em voltagens de +5V que supre o sistema acionador dos LEDs. A segunda é a conversão de voltagens dadas pelos receptores em dados digitais, processáveis pelo computador. Estima-se que a corrente máxima sobre os terminais de ambas as placas não deva exceder 5mA, tornando esta um parâmetro de projeto para o circuito elétrico.

4.2 Software

O conjunto de programas computacionais aplicados na operação do tomógrafo é denominado *software*. Este será o responsável pela ativação do aparelho, bem como por todo o tratamento matemático dos dados obtidos e, por fim, pela geração de imagens. Os arquivos elaborados serão, em sua maioria, escritos em Linguagem C. A plataforma operacional será o *Linux*, servidor de domínio público, bem como todos os programas aplicados.

4.2.1. Coleta de Dados

Como mencionado, a placa de interface computador/circuito elétrico necessita de arquivos próprios de calibração e controle. Estes determinarão como o circuito elétrico se comportará, segundo uma lógica pré-definida, para aumentar a confiabilidade dos dados obtidos. Estes arquivos também serão responsáveis pela correta coleta e armazenamento efetivo dos dados obtidos.

4.2.2. Processamento das informações

Nesta seção, encontram-se todos os arquivos necessários para que, a partir das medições na fronteira, seja possível determinar qual a distribuição de difusão da luz no interior do domínio. Processar-se-á toda a aplicação do

Método dos Elementos Finitos (MEF), bem como as soluções dos Problemas Direto e Inverso, como descrito na literatura que embasa este projeto.

O MEF inicia-se com a discretização do domínio, feita mediante aplicação do programa *Gmsh*. Este fornecerá dois arquivos distintos: o primeiro conterá as posições x e y de cada um dos nós; o segundo explicitará toda a topologia dos elementos, indicando quais nós os compõem, bem como sua orientação.

Posteriormente, segue a elaboração das matrizes locais e global do problema. Esta será a base para resolução do sistema e, segundo LOGAN, 1997[5], é dada pela somatória das matrizes locais, escritas em coordenadas globais.

Em seguida, encontram-se os arquivos para a resolução do Problema Direto, mediante aplicação do Princípio Variacional, como descrito no embasamento teórico. O Problema Direto é necessário para a solução do Problema Inverso, já que para resolver o mesmo são necessárias informações *a priori* do interior do domínio.

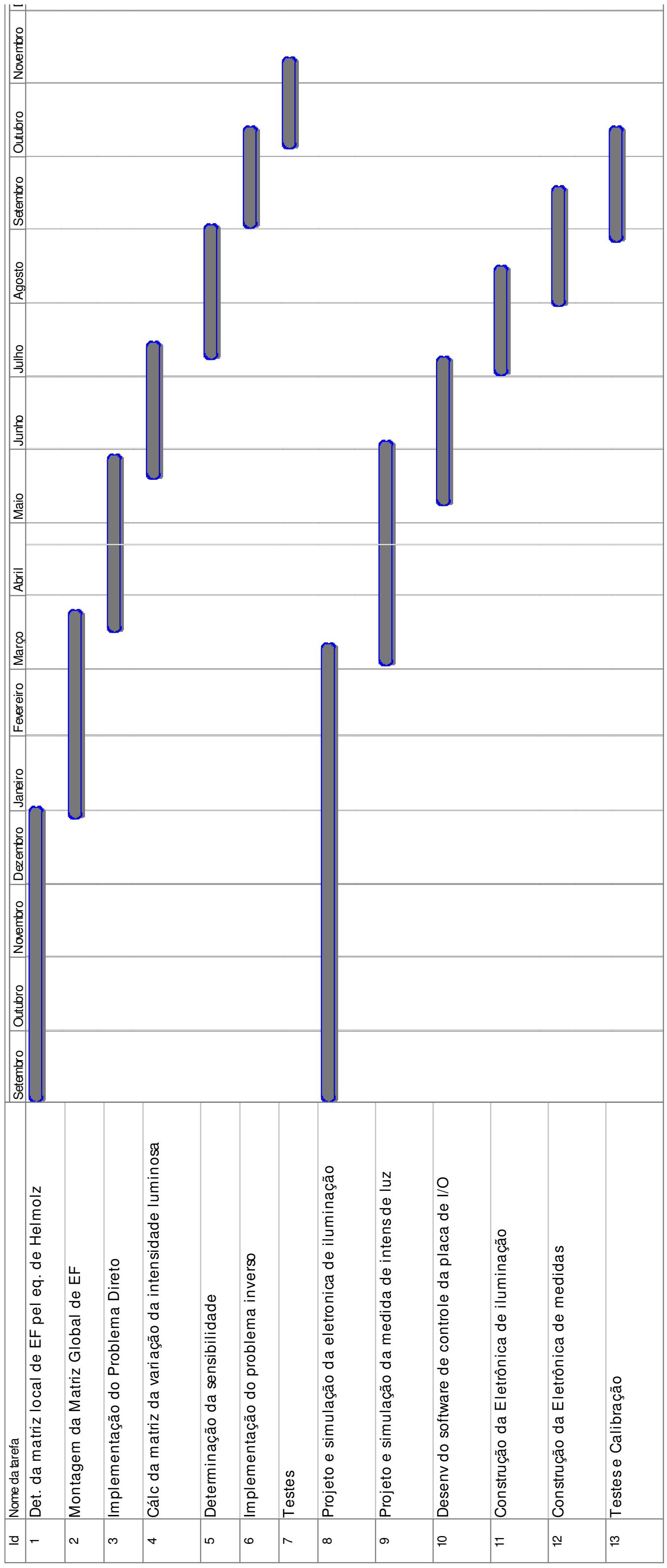
Por fim, encontram-se os arquivos para solucionar o Problema Inverso e assim obter a informação desejada. A saída final de todos estes arquivos será uma matriz com a distribuição de coeficientes de difusividade dos elementos que compõem a malha de finitos. Ressalta-se a importância da aplicação do método Caixa-Preta, descrito por AYA et al, 2006[6].

1.2.2. Geração de imagens

Encontram-se aqui os arquivos para a geração de imagens de fato. Pretende-se construir gráficos que contenham curvas de difusividade em função da localização espacial no domínio. Intenta-se detectar alguma acentuada variação em uma região, inferindo-se então a presença e geometria da perturbação.

5 Descrição das etapas a serem desenvolvidas

5.1 Cronograma



5.2 Custos

Os custos para elaboração do protótipo estão relacionados com o circuito elétrico a ser construído, bem como as placas que fazem a interface deste com um computador. Os seguintes valores são apresentados

Tabela 2 - Custos do protótipo.

Componente	Valor (R$)	Quantidade
Placa I/O digital-analógica	3.050,00*	1
Receptor TSL-257	3,54	16
Optoacoplador 4N33	1,50	16
LED emissor	1,30	16
Resistências (geral)	0,05	48

* Valor estimado

Assim, o custo total do protótipo resulta em aproximadamente R$3.160,00, adicionando outros custos como transporte ou importação. Estes serão custeados pelos alunos e professor.

6 RESULTADOS

6.1 Hardware

6.1.1. Seleção do emissor e receptor

Como realce, deseja-se com o protótipo a possibilidade de identificar características de perfusão em tecidos, o que torna as características ópticas da hemoglobina norteadoras. A análise da Figura 2 possibilitou a definição do comprimento de onda (λ) de operação como *700nm*, tal este pertencente na sobreposição de espectros NIR e Vermelho. Para este λ, o coeficiente de difusão da luz é demasiadamente maior para a hemoglobina em comparação com a oxihemoglobina e a água, o que torna aquela apta a detecção.

Para tal λ, o emissor escolhido é um LED vermelho comum. Especificamente, determinou-se para tal o componente L-513HURC, um LED vermelho de 5mm de diâmetro produzido por *PARA Light*, com alimentação de 5V e corrente de operação de 20mA.

O receptor escolhido foi o TSL-257, um conversor luz-voltagem produzido pela TAOS (*Texas Advanced Optoeletronic Solutions ®*). Seu comprimento de onda nominal é 470nm. Contudo, pela análise de seu *datasheet*, infere-se que o TSL-257 apresenta melhor resposta para 700nm, tornando-o apropriado. O TSL-257 apresenta ainda a interessante característica de possuir um amplificador operacional interno, o que dispensa a associação de um externo e torna as medições mais fáceis e sensíveis. Adicionalmente, o tempo de resposta é diminuto e possibilita que a placa de aquisição seja mais modesta.

6.1.2. Dimensionamento do circuito elétrico

O circuito elétrico é dividido em duas partes: o circuito emissor e o circuito receptor. Intenta-se alimentar ambos os circuitos com o auxílio de uma fonte comum de computador, a qual dispõem duas voltagens distintas, 5V e 12V. Serão 16 canais emissor-receptor, cada qual com seus respectivos circuitos.

O circuito emissor é constituído pelos elementos eletrônicos que possibilitam a ativação do LED emissor a partir de incentivos – pulsos de 5V – provindos da placa I/O digital-analógica de emissão. Esta admite corrente máxima entre seus terminais de 5mA, sendo esta a corrente que atravessará uma resistência denominada R_1. O LED emissor, acoplado em série com uma resistência R_2, será alimentado por uma tensão de 12V. A corrente máxima que atravessa o LED emissor não pode ultrapassar 15mA. Para evitar possíveis sobrecargas na placa digital-analógica, utiliza-se um optoacoplador, o 4N33 da *Siemens*, para isolar a placa I/O do LED emissor. Considerando os parâmetros apresentados, obteve-se o seguinte circuito emissor:

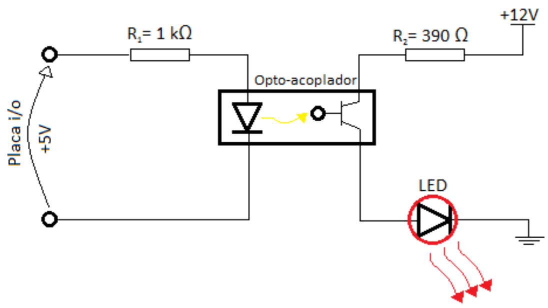


Figura 4 – Esboço do circuito elétrico de um canal de emissão.

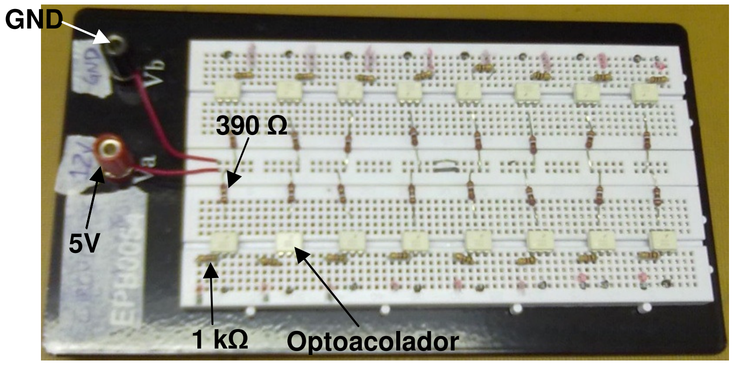


Figura 5 – Placa de circuito elétrico de emissão.

O circuito receptor, analogamente ao emissor, é composto pelos componentes que possibilitam o registro das voltagens obtidas nos receptores (conversores luz-voltagem) pela placa I/O analógico-digital. A ligação entre o TSL-257 e a placa de aquisição é feita por uma resistência R_3, a qual estará com uma diferença de potencial fornecida pelo emissor e sendo atravessada por uma corrente que não pode ultrapassar 5mA para manter o bom funcionamento da placa. Assim, sobre estas restrições, surge o seguinte circuito receptor:

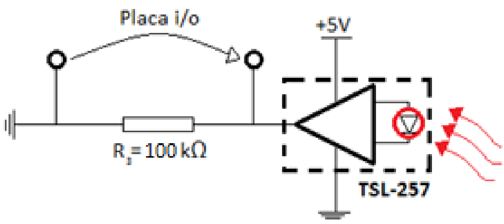


Figura 6 – Esboço do circuito elétrico de um canal de recepção.

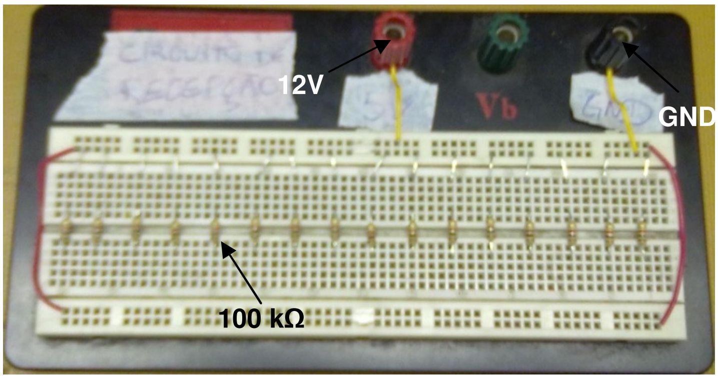


Figura 7 – Placa de circuito elétrico de recepção.

6.1.3. <u>Configuração da placa de aquisição</u>

A placa de aquisição utilizada é o modelo KPCI-3110, da *Keithley Instruments Inc*. Esta é uma placa I/O que apresenta 16 canais de entrada analógicos diferenciais e 16 canais digitais de saída.

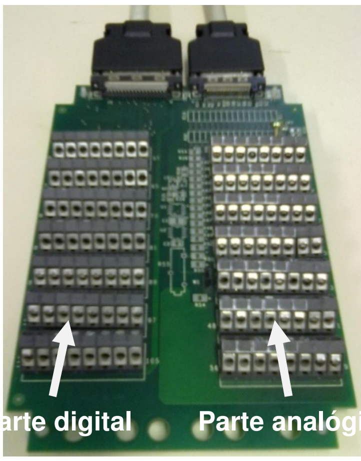


Figura 8 – Placa de controle KPCI-3110.

Esta placa apresenta a complicação de operar preferencialmente com a plataforma *Windows* e com o uso de Linguagem C++ ou VBA, o que demonstrou ser um empecilho já que todos os outros arquivos de *software* foram feitos na plataforma *Linux* e em Linguagem C. O arquivo de controle foi desenvolvido como pequenas alterações de alguns exemplos disponibilizados gratuitamente pela própria fabricante. A saída destes são arquivos *.txt* que contêm os valores das leituras dos sensores.

6.1.4. <u>Condições de experimento</u>

Os emissores e sensores foram instalados em 16 pequenas placas acrílicas, formando os canais, para que facilitar a realização dos experimentos. A fixação foi feita com silicone adesivo.

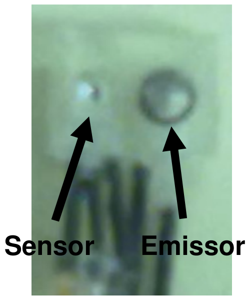


Figura 9 – Sensor e emissor fixados à placa de suporte.

Os experimentos foram realizados utilizando-se um vaso de polimérico como corpo de prova. Este apresenta diâmetro inferior 8cm, superior de 14cm e altura de 12cm. Os canais foram posicionados na altura de tal forma que formassem uma circunferência de aproximadamente 10cm. Os canais foram fixados ao meio com o uso de cola quente.

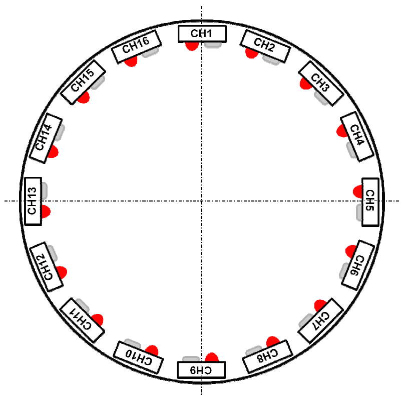


Figura 10 – Diagrama de posicionamento dos canais.

A distribuição dos canais é feita de ordem controlada. Intuitivamente, os sensores mais próximos à fonte emissora devem apresentar maiores leituras e deve ocorrer um decaimento suave conforme a distância em relação à mesma aumenta, devido à influência dos efeitos ópticos.

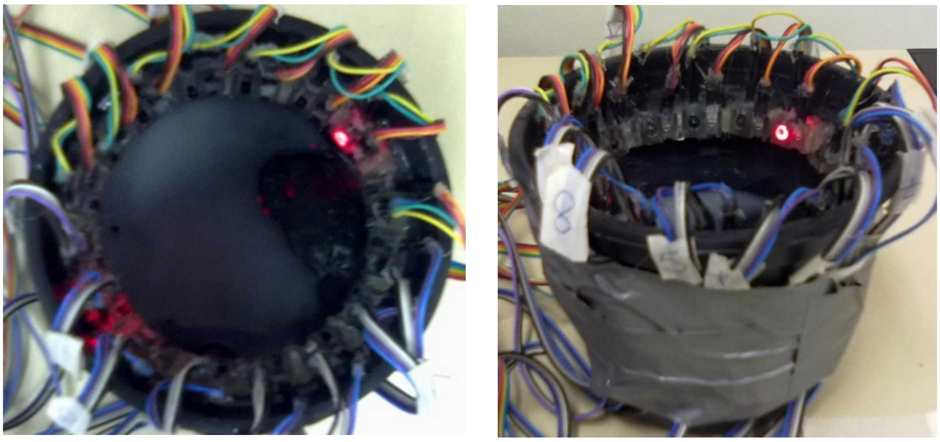


Figura 11 – Fixação dos canais no meio de estudo.

6.2 Testes

Após toda a montagem do *hardware*, os testes foram executados. Devido à característica da placa de aquisição de operar pela plataforma *Windows*, os testes puderam ser efetuados em um computador o qual não apresentava os programas desenvolvidos para tratamento de dados. Assim sendo, os testes realizados geraram arquivos de texto, os quais contêm as leituras de voltagem dos sensores registradas pela placa, sendo estes então submetidos ao conjunto de operações do *software*.

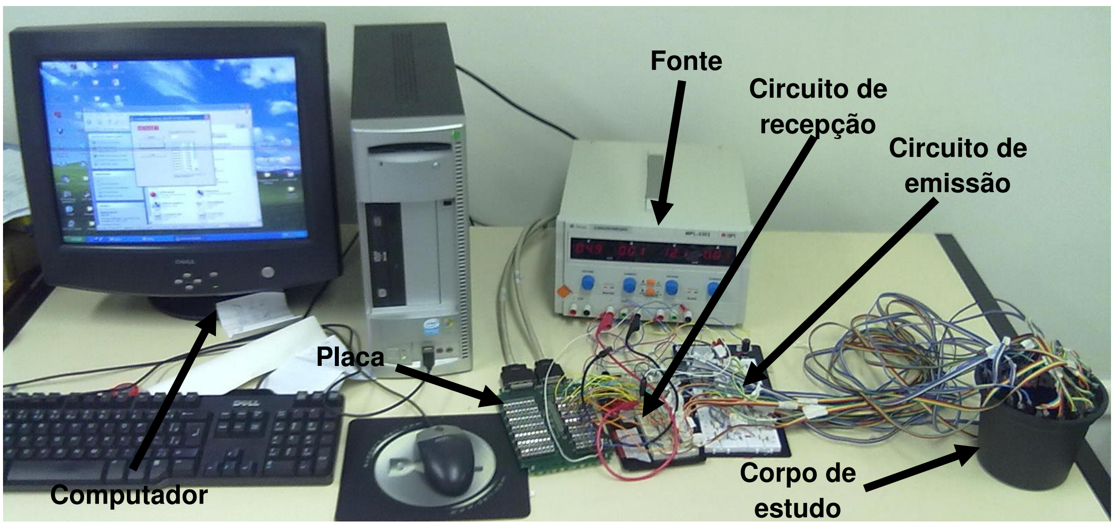


Figura 12 – Protótipo montado.

O meio homogêneo utilizado foi uma mistura de água e leite. Este meio foi assim escolhido por apresentar propriedades ópticas com comportamento semelhante às dos tecidos (alto espalhamento e baixa absorção) para o comprimento de onda de operação.

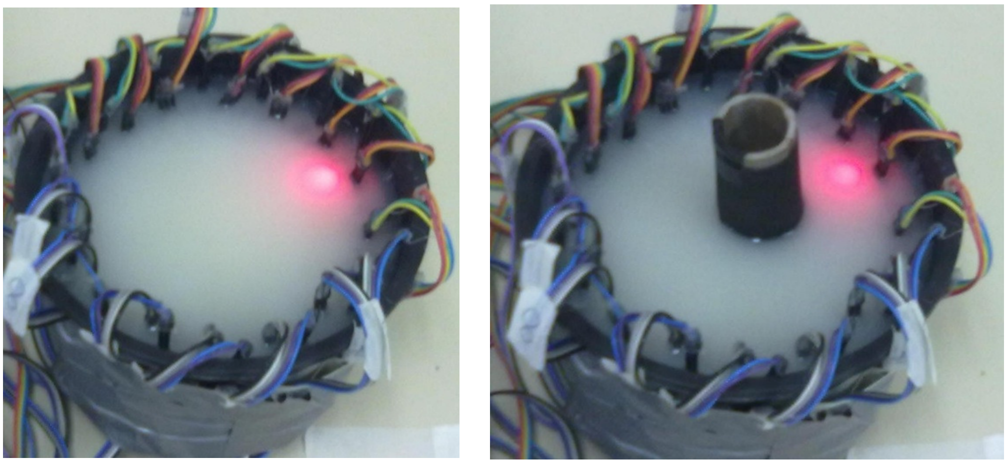


Figura 13 – Corpo de estudo preenchido pelo meio. À esquerda, sem perturbação. À direita, com a presença de uma perturbação.

O corpo que provocará a perturbação óptica no meio foi feito com o uso de um cano de PVC de diâmetro 25mm, envolto por uma lixa preta 600. Esta foi aplicada por sua coloração negra e por ser satisfatoriamente opaca, diminuindo a influencia da reflexão.

Realizaram-se dois distintos experimentos: meio sem perturbação e perturbação próxima a alguns canais. Pela diferença dos casos, intenta-se que o protótipo seja capaz de identificar a perturbação. O procedimento dos testes obedece ao da TOD, já descrito anteriormente. Sendo assim, para cada experimento, foram realizadas 17 leituras dos canais, 16 úteis (com somente um dos emissores ativados) e uma de calibração (todos os emissores desativados). Os dados obtidos são apresentados no Apêndice A. A leitura zero corresponde à calibração, enquanto os subseqüentes são nomeados conforme o canal emissor ativado.

6.3 Software

6.3.1 Elaboração da Malha de Finitos

A malha é desenvolvida a partir da determinação do tamanho do domínio e do número de nós na fronteira que o compõe. O número de nós na fronteira do domínio é o dobro do número de canais que o TOD possuirá, ou seja, 32 nós. Estes são dispostos circularmente num domínio de $10cm$ de diâmetro, de forma que a distância angular entre os mesmo fosse constante. Se fossem tomados apenas 16 pontos para delimitar a fronteira do domínio poderia haver imprecisão na mesma, prejudicando a solução do problema inverso uma vez que os potenciais de iluminação são calculados em função dos valores obtidos para essa fronteira. Além disso, em problemas de elementos finitos, a resolução também é aumentada com o aumento do número de pontos no domínio.

O programa utilizado para a geração da malha foi o *GMSH* para a plataforma Linux. A malha gerada pode ser vista abaixo.

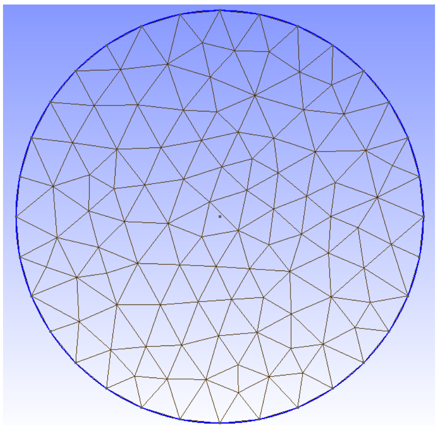


Figura 14 – Malha de Elementos Finitos.

A malha é gerada seguindo algumas etapas:

- Definem-se os nós que irão compor a fronteira do domínio, bem como suas coordenadas.
- A partir desses nós são geradas as linhas características: dezesseis arcos, centrados em (0,0,0), unem os sucessivos pontos.

- Em seguida, é definido o *loop* que caracteriza a fronteira do domínio e é constituído pela fronteira do domínio.
- Por último, cria-se o domínio a ser discretizado que é a área interior ao *loop* gerado

O programa *Gmsh* tem como saída dois arquivos que serão fundamentais para a determinação das matrizes locais e global dos elementos. O primeiro arquivo contém as coordenadas espaciais para cada nó gerado na malha, enquanto o segundo contém os nós que formam cada um dos elementos triangulares.

6.3.2 Resolução do Problema Direto

Como já afirmado no embasamento teórico, o Problema Direto para TOD é enunciado por: dadas as distribuições de fontes emissoras de fótons na fronteira do domínio e a do valor dos parâmetros ópticos relacionados, determina-se o fluxo de fótons resultantes na fronteira.

Definidas as propriedades geométricas como explicitado acima, é necessário adicionar informações sobre das propriedades ópticas do meio para a aplicação do Problema Direto. Isto significa a escolha de coeficientes de absorção μ_a e espalhamento reduzido μ'_s. Como o comprimento de onda $\lambda = 700nm$ é sensível a presença de hemoglobina, escolheu-se como parâmetros as propriedades do sangue, $\mu_a = 0{,}173\, mm^{-1}$ e $\mu'_s = 2{,}678\, mm^{-1}$, mesmo sabendo que as mesmas implicarão alguma dificuldade para a realização dos experimentos.

O programa desenvolvido deve desempenhar a seguinte função: determinar a distribuição da densidade de fótons $\Phi(x, y)$ para o domínio Ω após conhecida excitação, entendida como a ativação de um ou um grupo de LED's. Prevê-se o seguinte comportamento: a distribuição de $\Phi(x, y)$ deve apresentar altos valores nas regiões próximas a excitação e decrescer conforme afasta-se da mesma.

Para a solução do Problema Direto, será aplicada a metodologia de Princípio Variacional (minimização do funcional Π), como descrito no embasamento teórico. Este artifício é necessário para estabelecer a densidade de fótons para ϕ_i para cada um dos nós da malha. A multiplicação da matriz global $\underline{\underline{N_G}}$ por este vetor $\underline{\phi}$ fornece a forma distribuída da função $\Phi(x, y)$ para todo o domínio:

$$\Phi(x, y) \approx \widehat{\Phi} = \underline{\underline{N_G}\phi} \tag{49}$$

Matematicamente, resolve-se o seguinte sistema linear descrito pela Equação (42). A rotina escrita deve desempenhar as seguintes funções:

1. Determinação das matrizes $\underline{\underline{M}}$ e $\underline{\underline{H}}$. É importante realçar que no cálculo de $\underline{\underline{H}}$ a origem das coordenadas deve ser deslocada para o centróide de cada um dos elementos finitos triangulares.
2. Criação da matriz que multiplica $\underline{\phi}$ na expressão resultante da minimização do funcional Π:

$$\left[\frac{Dt}{4A}\underline{\underline{M}} + \frac{\mu_a t}{4A}\underline{\underline{H}}\right] \tag{50}$$

3. Elaboração do vetor independente, localizado à direita na equação:

$$\iiint_{\Omega} \mathrm{q}_0 \underline{N}^T d\Omega \tag{51}$$

4. Resolução do sistema linear:

$$\underline{\phi} = \left[\frac{Dt}{4A}\underline{\underline{M}} + \frac{\mu_a t}{4A}\underline{\underline{H}}\right]^{-1} \left(\iiint_{\Omega} \mathrm{q}_0 \underline{N}^T d\Omega\right) \tag{52}$$

   Sendo esta de fato a característica da resolução do Problema Direto. Como artifício de resolução, utiliza-se a biblioteca *gsl_linalg_LU*, a qual resolve o sistema linear pela decomposição em duas matrizes triangulares, *L* (*Lower*) e *U* (*Upper*).
5. Plotagem dos valores de $\Phi(x, y)$ com a aplicação do *software gnuplot*.

As Figuras 15 e 16 são geradas para a condição de meio homogêneo, ou seja, com coeficiente de absorção μ_a, e conseqüentemente o de difusão D, constante para todo o domínio. O sistema de coordenadas tem como origem o centro da circunferência de 10cm de diâmetro a qual delimita o espaço de estudo. Contudo, as imagens são geradas para um quadrado de 10x10cm.

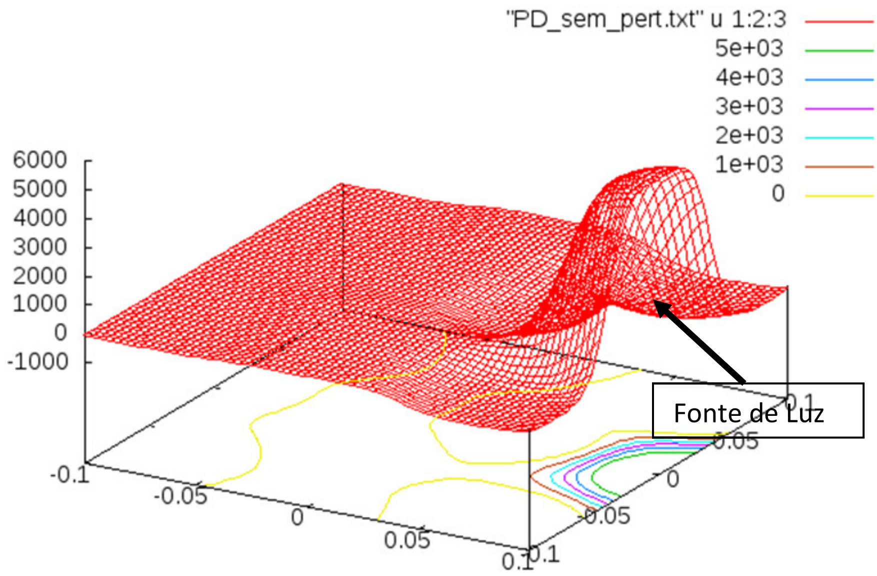


Figura 15 - Resolução do PD para meio homogêneo não perturbado 3D.

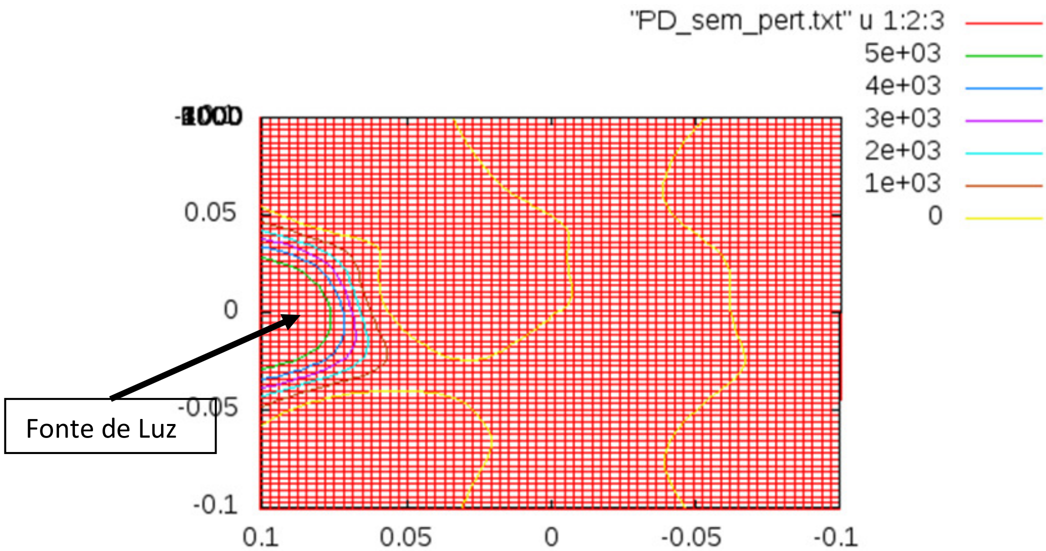


Figura 16 - Resolução do PD para meio homogêneo não perturbado 2D.

A fonte de luz (LED emissor) encontra-se no ponto em destaque, o qual apresenta então a maior concentração de fótons. Perceba que quanto mais distante o ponto de análise com relação ao emissor, a função densidade de fótons decai, resultado condizente com o esperado.

Para análise comportamental, foi simulada uma perturbação no meio. Essa perturbação interfere no comportamento da luz dentro do domínio, realizando-se novamente a resolução do Problema Direto com a perturbação, obteve-se as Figuras 17 e 18, a seguir.

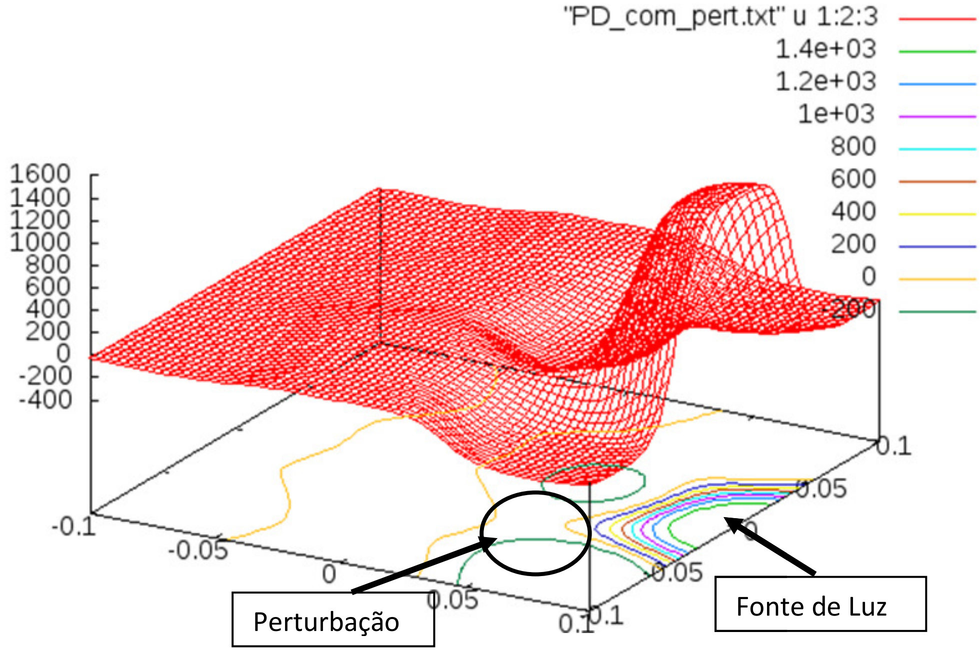


Figura 17 - Resolução do PD para meio homogêneo perturbado 3D.

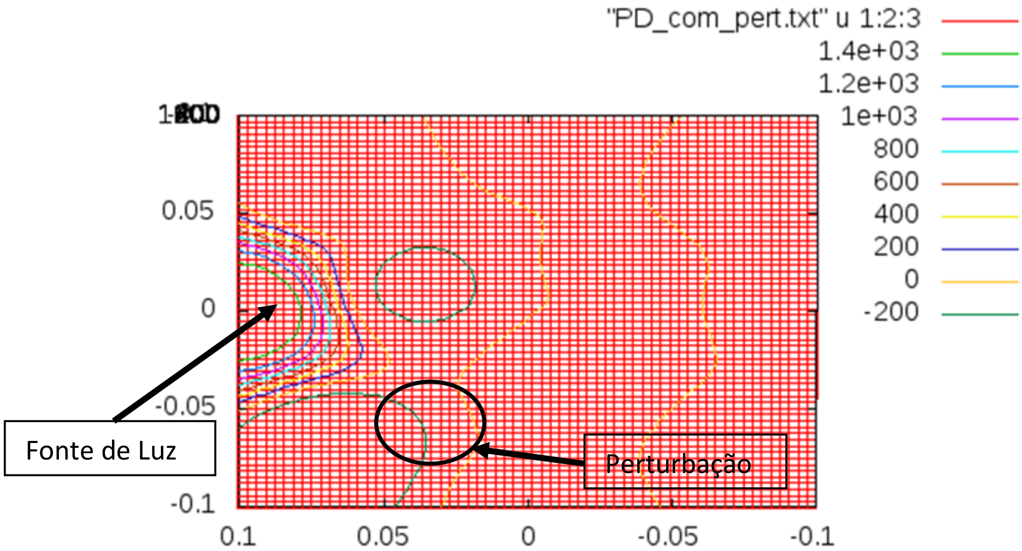


Figura 18 – Resolução do PD para meio homogêneo perturbado 2D.

O aumento do coeficiente de absorção μ_a para um valor 5 vezes maior na região do domínio (perturbação virtual da resolução apresentada acima), para mesma posição do emissor, resulta numa alteração significativa na distribuição dos potenciais de fótons. A intenção deste aumento é simular a presença de um elemento com propriedades ópticas diferentes do meio.

6.3.3 Problema Inverso - Os métodos de regularização

Nesta seção, os métodos de regularização descritos nas seções teóricas serão comparados e o método que gerar a imagem mais próxima da real será utilizado para a execução dos testes.

6.3.3.1 Regularização de Tikhonov

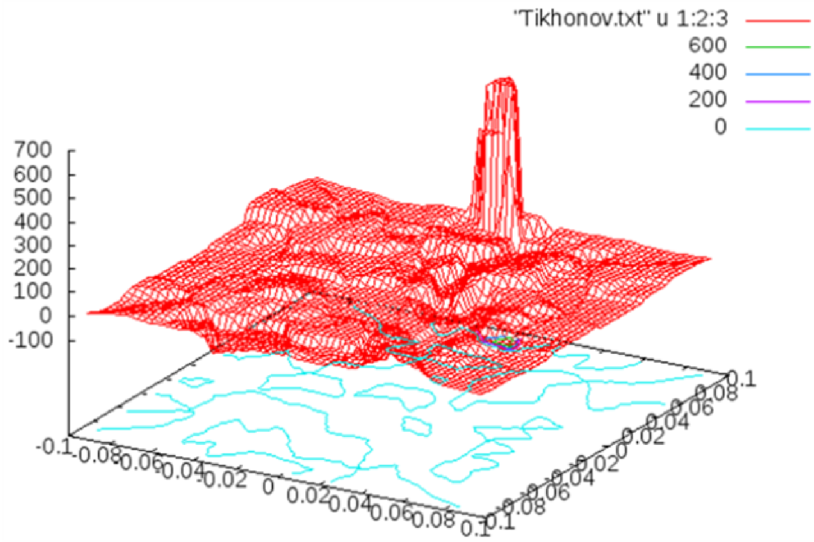


Figura 19 – Aplicação da regularização de Tikhonov para solução do PI, em 3D.

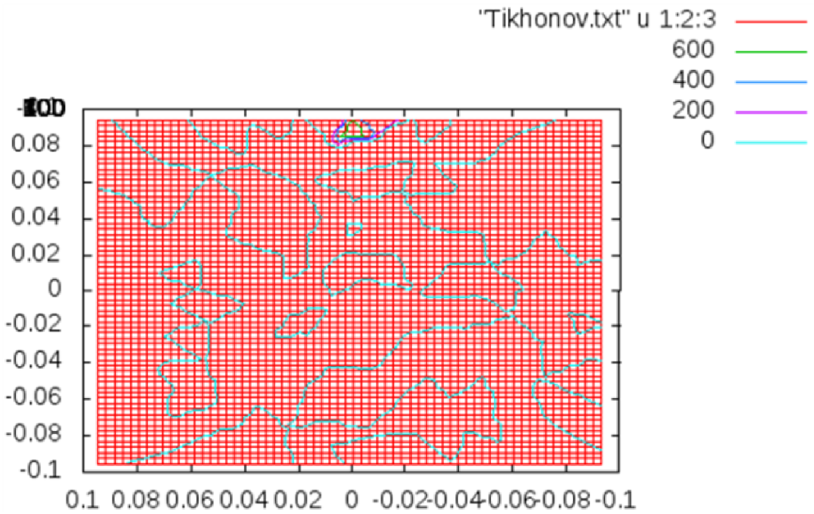


Figura 20 – Aplicação da regularização de Tikhonov para solução do PI, em 2D.

6.3.3.2 Truncamento de Valores Singulares

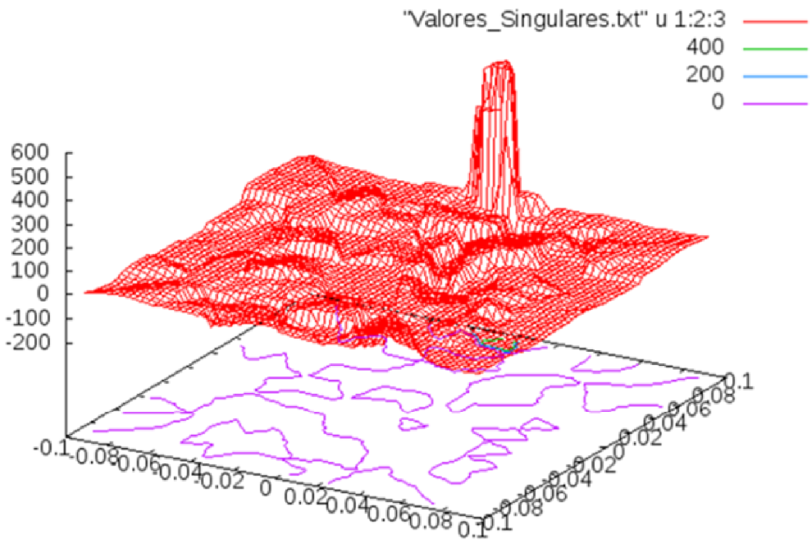


Figura 21 – Aplicação do Truncamento de Valores Singulares para solução do PI, em 3D.

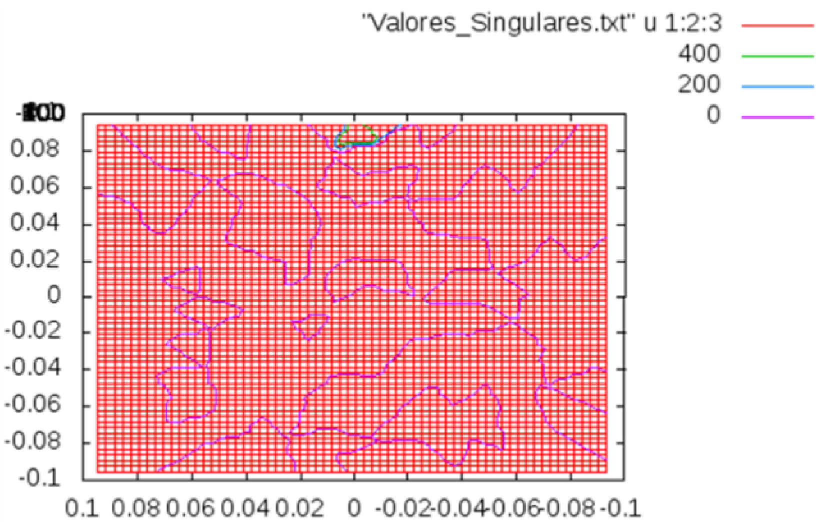


Figura 22 – Aplicação do Truncamento de Valores Singulares para solução do PI, em 2D.

6.3.3.3 Método Heurístico

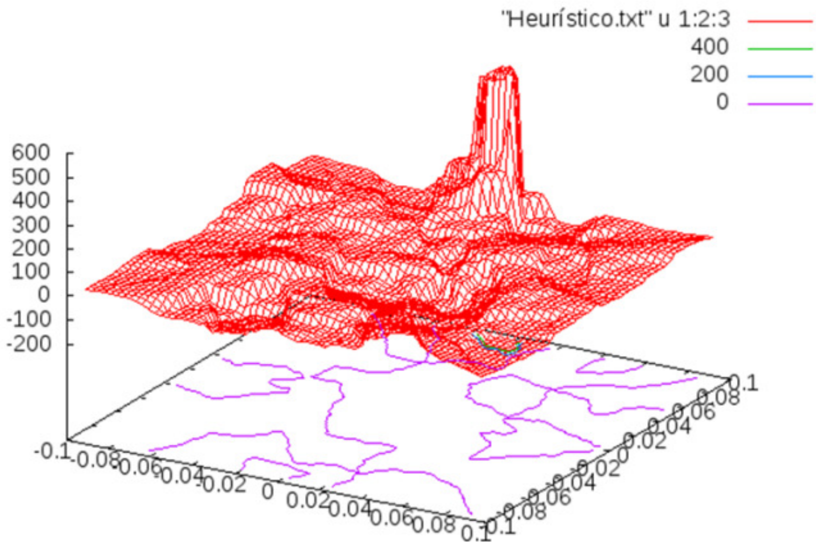


Figura 23 – Aplicação do Método Heurístico para solução do PI, em 3D.

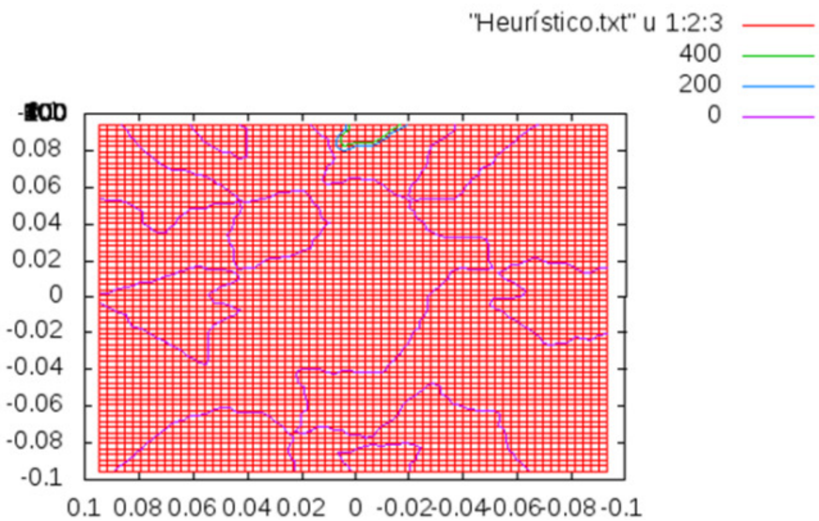


Figura 24 – Aplicação do Método Heurístico para solução do PI, em 2D.

6.3.3.4 Composição do Método Heurístico com Truncamento de Valores Singulares

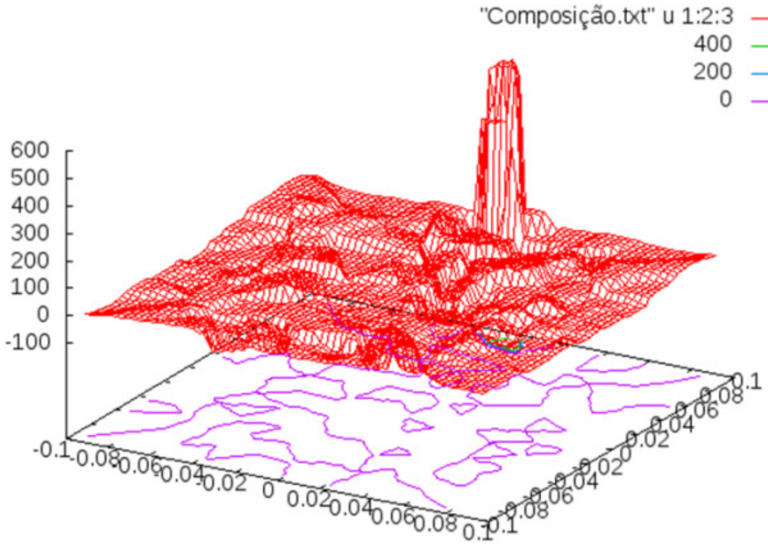


Figura 25 – Aplicação da composição do Método Heurístico com Truncamento dos Valores Singulares para solução do PI, em 3D.

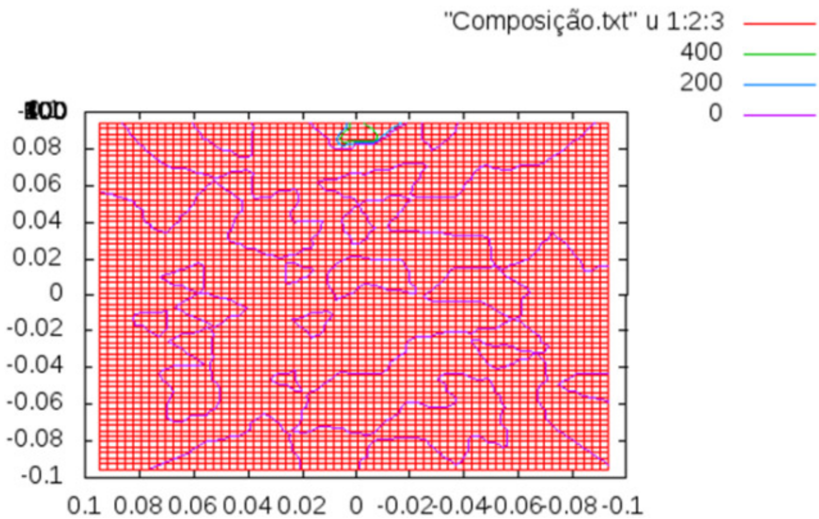


Figura 26 – Aplicação da composição do Método Heurístico com Truncamento dos Valores Singulares para solução do PI, em 2D.

Todos os métodos apresentaram resultados coerentes com a perturbação que foi introduzida no domínio. No entanto o método que mais se adéqua ao problema é o do Método Heurístico (também conhecido como Filtro de Gauss), pois além da imagem apresentar o perfil mais suave esse método é o que produz menos artefatos. Nos diagnósticos médicos é fundamental que não se tenham artefatos na imagem, para que estes não sejam equivocados.

6.3.4 Resolução do Problema Inverso – Solução dos testes

Os dados gerados na realização dos testes são tratados matematicamente para a geração de imagens. Objetiva-se que estas imagens apresentem a informações quanto à localização e geometria da perturbação interior ao meio de estudo e não necessariamente definir os coeficientes de absorção envolvidos no experimento.

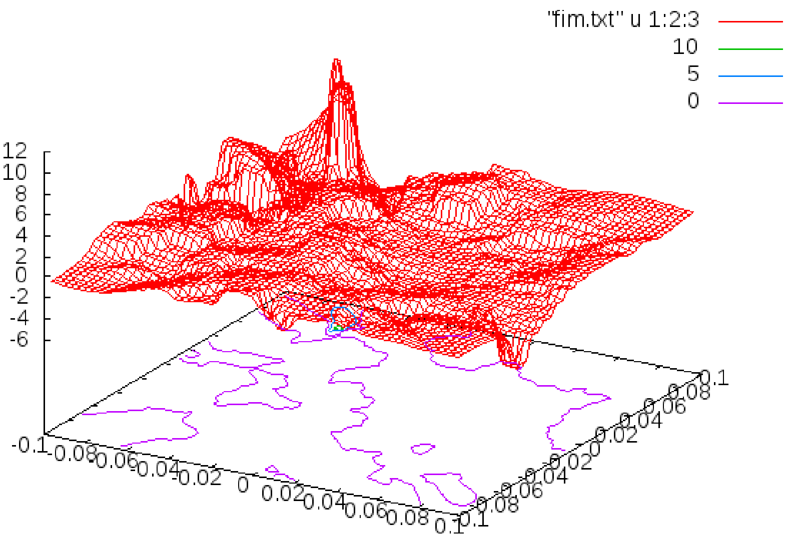


Figura 27 – Solução do PI para teste, em 3D.

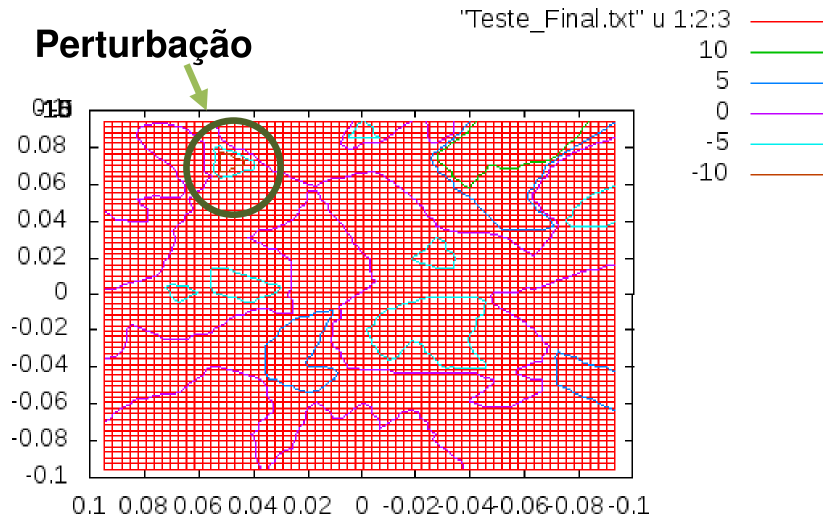


Figura 28 – Solução do PI para teste, em 2D.

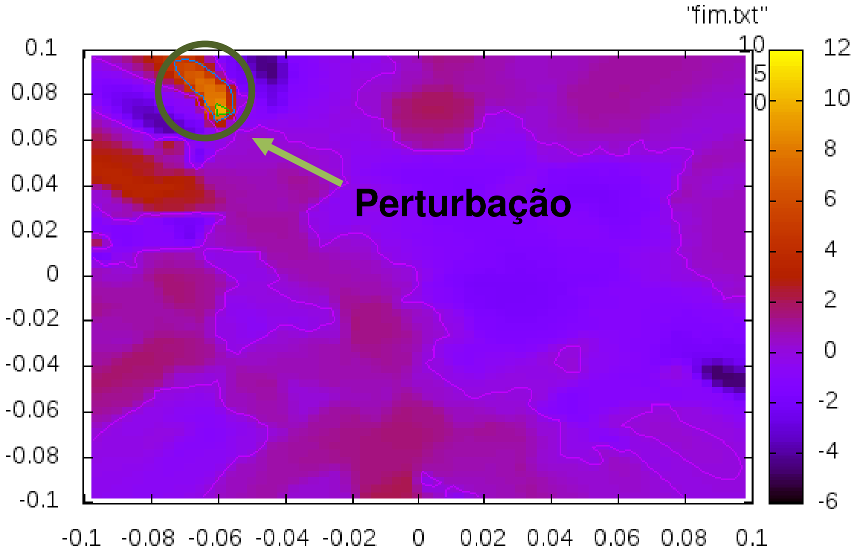


Figura 29 - Solução do PI para teste, imagem 2D colorida.

No conjunto de imagens acima, apesar da presença de ruído, é possível observar uma região onde há uma significativa mudança com relação ao restante. Esta mudança indica a presença da perturbação. A Figura 29 torna esta conclusão mais óbvia.

É importante ressaltar que devido à diferença da geometria circular do domínio da retangular de plotagem há uma distorção inerente, principalmente nos extremos das figuras.

7 ANÁLISE

O principal objetivo do presente trabalho era a construção do protótipo de um tomógrafo de óptica difusa com 16 canais, bem como a arquitetura de um software que aplicasse o Método dos Elementos Finitos e solucionasse os Problemas Direto e Inverso, além de operar o hardware e ser capaz de gerar imagens. Este foi atingido satisfatoriamente.

Pretendia-se que o protótipo fosse apto a identificar a presença e inferir a geometria de uma perturbação. Os resultados obtidos mostram que o equipamento foi capaz de detectá-las, contudo o mesmo não é capaz de determinar com boa precisão a geometria, sendo este um critério a ser aprimorado.

Em comparação com a bibliografia consultada, principalmente com NISSILÄ et al, 2005[1] e NOGUEIRA, 2007[2], o trabalho desenvolvido enquadra-se em um meio-termo. Os resultados apresentados demonstram um grande avanço com relação aos obtidos por NOGUEIRA, 2007[2], os quais foram desenvolvidos em mesmo ambiente profissional e sob a tutela do mesmo mestre, mas ainda estão muito aquém dos obtidos por NISSILÄ et al, 2005[1], de referência norteadora internacionalmente verificável.

Esta constatação demonstra que houve avanços na tecnologia local, mas aprimoramentos são necessários para que os resultados apresentem-se de forma mais apreciável. Algumas propostas de melhoria podem ser estudadas nas próximas gerações do equipamento, tais como: incremento da resolução do aparelho, por meio de refino da malha finita; aumento da potência do emissor; substituição sensores e emissores por cabos de fibra ótica, os quais permitem a operação em simultânea em diferentes comprimentos de onda; aprimoramento do *software*; tratamento tridimensional do problema; realização de simulações com propriedades mais próximas da realidade dos tecidos biológicos; estudo do espalhamento refinado e absorção de luz NIR nos tecidos corpóreos.

Por fim, destaca-se como avanços atingidos: a preocupação com a aplicação do protótipo; a seleção de equipamento mais apropriado; resolução obtida demasiadamente melhor; refino da malha de elementos finitos, tornando-a mais refinada; aumento no número de canais; teste de diferentes métodos de regularização da imagem e seleção do mais apropriado; geração de imagens coloridas, as quais transmitem melhor intuição sobre o resultado obtido.

8 REFERÊNCIAS

[1] NISSILÄ, ILKA.; NOPONEN, TOMMI.; HEINO,JENNI.; KAJAVA,TIMO.; KATILA, TOIVO. **Advances in Electromagnetic Fields in Living Systems, Volume 4, Chapter 3.** Springer Science+Business Media, New York, 2005.

[2] NOGUEIRA, MARCELO. **Tomografia Óptica Difusa: Hardware e Testes *in vitro.*** Departamento de engenharia mecânica da EPUSP,2007.

[3] ARRIDGE, S. R. **Optical tomography in medical imaging.** Inverse Problems 13, R41, 1999.

[4] ZIENKIEWICZ, O. C. & TAYLOR, R. L. **The finite element method.** McGraw-Hill, 1989.

[5] LOGAN, Daryl L. **A first course in the finite element method using Algor™**, PWS publishing company, 1997.

[6] AYA, J. C. C.; MOURA, F. S. DE.; NAN, P. C.; SHWEDER, R. K.; LIMA, R. G. **Regularizations for a Black Box Back-projection EIT algorithm.** ABCM Symposium Series in Mechatronics, 2006.

9 APÊNDICE A

Meio sem perturbação (valores médios de 12 amostras)

Leituras	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Canal 1	4,620	3,655	0,188	0,125	0,190	0,039	0,086	0,031	0,031	0,062	0,134	0,167	0,347	1,128	4,644	4,633
Canal 2	4,754	4,751	0,945	0,391	0,457	0,064	0,132	0,040	0,033	0,055	0,071	0,102	0,165	0,367	1,054	3,828
Canal 3	3,057	4,779	4,779	2,035	1,558	0,157	0,264	0,061	0,045	0,063	0,068	0,084	0,110	0,188	0,390	0,924
Canal 4	0,554	2,556	4,745	4,717	4,712	0,419	0,515	0,097	0,058	0,068	0,062	0,064	0,070	0,097	0,157	0,272
Canal 5	0,175	0,541	0,738	4,775	4,776	2,165	1,615	0,205	0,093	0,091	0,069	0,060	0,057	0,064	0,088	0,119
Canal 6	0,082	0,195	0,170	1,629	4,760	4,747	4,737	0,621	0,198	0,147	0,095	0,067	0,057	0,054	0,062	0,071
Canal 7	0,058	0,110	0,071	0,408	4,766	4,766	4,766	3,422	0,684	0,362	0,180	0,108	0,076	0,063	0,064	0,062
Canal 8	0,032	0,051	0,028	0,115	1,122	1,447	4,734	4,714	3,191	0,976	0,337	0,153	0,087	0,061	0,051	0,042
Canal 9	0,032	0,044	0,022	0,061	0,420	0,347	4,751	4,729	4,728	4,726	1,108	0,371	0,163	0,096	0,068	0,049
Canal 10	0,031	0,035	0,017	0,033	0,158	0,099	1,038	1,778	4,764	4,761	4,757	1,067	0,327	0,145	0,090	0,053
Canal 11	0,059	0,060	0,045	0,073	0,111	0,059	0,420	0,460	2,018	4,757	4,758	4,757	1,101	0,361	0,173	0,085
Canal 12	0,063	0,056	0,018	0,029	0,095	0,041	0,222	0,178	0,477	4,338	4,769	4,770	4,770	1,437	0,497	0,186
Canal 13	0,096	0,067	0,020	0,027	0,067	0,029	0,110	0,072	0,142	0,794	4,256	4,744	4,743	4,735	1,341	0,354
Canal 14	0,205	0,112	0,020	0,025	0,065	0,021	0,082	0,044	0,073	0,297	1,015	4,773	4,773	4,774	4,773	1,200
Canal 15	0,644	0,257	0,032	0,060	0,085	0,081	0,098	0,078	0,076	0,151	0,378	1,210	4,757	4,759	4,759	4,758
Canal 16	2,924	0,733	0,061	0,058	0,106	0,026	0,067	0,030	0,034	0,084	0,156	0,364	1,089	4,771	4,771	4,768

Meio com perturbação (valores médios de 12 amostras)																
Leituras	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Canal 1	4,603	3,544	0,176	0,094	0,071	0,008	0,011	0,011	0,019	0,044	0,075	0,144	0,313	1,055	4,708	4,712
Canal 2	4,757	4,748	0,902	0,315	0,209	0,011	0,012	0,011	0,016	0,034	0,051	0,082	0,138	0,328	0,979	3,664
Canal 3	2,944	4,769	4,767	1,776	0,897	0,023	0,014	0,010	0,018	0,031	0,042	0,057	0,082	0,152	0,339	0,842
Canal 4	0,490	2,367	4,736	4,734	4,729	0,106	0,032	0,011	0,014	0,021	0,027	0,034	0,042	0,064	0,115	0,222
Canal 5	0,108	0,379	0,620	4,772	4,773	1,097	0,238	0,012	0,011	0,014	0,016	0,018	0,020	0,023	0,040	0,061
Canal 6	0,019	0,049	0,064	0,936	4,747	4,745	3,225	0,080	0,021	0,013	0,009	0,009	0,009	0,011	0,014	0,017
Canal 7	0,010	0,013	0,012	0,065	2,201	4,765	4,767	1,527	0,285	0,100	0,038	0,022	0,015	0,012	0,011	0,010
Canal 8	0,006	0,006	0,002	0,006	0,133	0,613	4,747	3,306	2,804	0,724	0,207	0,077	0,036	0,021	0,017	0,011
Canal 9	0,017	0,017	0,007	0,008	0,024	0,077	4,345	2,849	4,741	4,737	0,970	0,298	0,119	0,062	0,042	0,028
Canal 10	0,021	0,019	0,007	0,010	0,012	0,016	0,551	0,952	4,758	4,761	4,758	0,975	0,283	0,122	0,068	0,039
Canal 11	0,029	0,026	0,011	0,011	0,014	0,011	0,171	0,221	1,897	4,757	4,756	4,756	1,015	0,322	0,147	0,066
Canal 12	0,057	0,041	0,014	0,014	0,012	0,009	0,068	0,117	0,421	4,115	4,767	4,764	4,763	1,342	0,449	0,265
Canal 13	0,083	0,054	0,080	0,017	0,016	0,010	0,032	0,043	0,118	0,724	4,075	4,750	4,747	4,740	1,259	0,322
Canal 14	0,186	0,097	0,014	0,014	0,015	0,001	0,018	0,020	0,055	0,260	0,938	4,693	4,773	4,772	4,771	1,120
Canal 15	0,602	0,227	0,030	0,024	0,021	0,019	0,016	0,017	0,036	0,128	0,338	1,127	4,755	4,759	4,759	4,761
Canal 16	2,828	0,676	0,054	0,038	0,033	0,002	0,012	0,011	0,020	0,064	0,134	0,327	1,015	4,768	4,766	4,762

10 APÊNDICE B

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <gsl/gsl_vector.h>
#include <gsl/gsl_matrix.h>
#include <gsl/gsl_math.h>
#include <gsl/gsl_permutation.h>
#include <gsl/gsl_linalg.h>
#include <gsl/gsl_blas.h>

#define numel  198
#define numnos  116
#define numled  16
#define mua_cte 5
#define mus  700
#define t 0.04
#define var 5.0
#define alfa 0.05
#define beta 0.01
#define fonte 1.0
#define std_devi 0.01
#define raio 0.05
#define pot  0.1
#define epsilon 0.0000000000001
void matrizglobal(gsl_matrix *NG,
gsl_matrix *N, gsl_vector_int *nos);
void pd1(gsl_vector *x, gsl_vector *y,
gsl_vector_int *nos, gsl_matrix *M,
gsl_matrix *H, gsl_vector *xg, gsl_vector
*yg);
void pd2(gsl_matrix *M, gsl_matrix *Npd,
gsl_matrix *H, gsl_vector *mua);
void pd3(gsl_matrix *Ngb, gsl_vector *I,
gsl_vector *x,int LED);
void centroides (gsl_vector *x, gsl_vector
*y,gsl_vector *xg, gsl_vector *yg,
gsl_vector_int *nos);
int coordenadas (gsl_vector *x, gsl_vector
*y);
int topologia(gsl_vector_int *nos);
void print_phi_to_file(gsl_vector *x,
gsl_vector *y, gsl_vector *phi, int
imprimir);
void print_lambda_to_file(gsl_vector *x,
gsl_vector *y, gsl_vector *lambda0, int
imprimir);
void gera_coluna_lambda_k (gsl_vector *I,
gsl_vector *phi, gsl_matrix *NGpd,
gsl_vector *lambda0, gsl_vector *x,
gsl_vector *y);
void gera_matriz_lambda (gsl_vector
*lambda0, gsl_vector *lambdak, gsl_matrix
*lambda, int elem);
void calc_F(gsl_vector *xg,gsl_vector
*yg,gsl_matrix *f,double std_dev);
void calc_M(gsl_vector *xg,gsl_vector
*yg,gsl_matrix *m,double potencia, double
raio_dominio);

int main () {

/*--Início da Declaração de Variáveis--*/
        gsl_vector *x = gsl_vector_alloc
(numnos);
        gsl_vector *y = gsl_vector_alloc
(numnos);
        gsl_vector *xg = gsl_vector_alloc
(numel);
        gsl_vector *yg = gsl_vector_alloc
(numel);
        gsl_vector_int *nos =
gsl_vector_int_alloc (3*numel);
        gsl_matrix *N = gsl_matrix_alloc
(numel, 9);
        gsl_matrix *NG = gsl_matrix_alloc
(numnos, numnos);
        gsl_matrix *M = gsl_matrix_alloc
(numel, 9);
        gsl_matrix *H = gsl_matrix_alloc
(numel, 9);
        gsl_matrix *Npd = gsl_matrix_alloc
(numel, 9);
        gsl_matrix *NGpd =
gsl_matrix_alloc (numnos, numnos);
        gsl_vector *I = gsl_vector_alloc
(numnos);
        gsl_vector *phi = gsl_vector_alloc
(numnos);
        gsl_vector *lambda0 =
gsl_vector_alloc (numled*numled);
        gsl_vector *testepi =
gsl_vector_alloc (numled*numled);
        gsl_vector *lambdak =
gsl_vector_alloc (numled*numled);
        gsl_vector *lambdak1 =
gsl_vector_alloc (numled*numled);
        gsl_vector *mua = gsl_vector_alloc
(numel);
        gsl_matrix *lambda =
gsl_matrix_alloc (numled*numled, numel);
        gsl_matrix *teta =
gsl_matrix_alloc (numel, numel);
        gsl_matrix *f = gsl_matrix_alloc
(numel, numel);
        gsl_matrix *f_inv =
gsl_matrix_alloc (numel, numel);

        gsl_matrix *m = gsl_matrix_alloc
(numel, numel);
        gsl_matrix *b = gsl_matrix_alloc
(numel, numel);
        gsl_matrix *B_irreg =
gsl_matrix_alloc (numel, numled*numled);
        gsl_matrix *B_reg =
gsl_matrix_alloc (numel, numled*numled);

        int imprimir=0;

        gsl_vector_set_zero(x);
        gsl_vector_set_zero(y);
        gsl_vector_set_zero(xg);
        gsl_vector_set_zero(yg);
        gsl_vector_int_set_zero(nos);

        gsl_matrix_set_zero(N);
        gsl_matrix_set_zero(NG);

        gsl_matrix_set_zero(M);
        gsl_matrix_set_zero(H);
        gsl_matrix_set_zero(Npd);
        gsl_matrix_set_zero(NGpd);

        gsl_vector_set_zero(phi);
        gsl_vector_set_zero(I);
        gsl_vector_set_zero(lambda0);
        gsl_vector_set_zero(lambdak);
        gsl_vector_set_zero(lambdak1);
        gsl_vector_set_all(mua, mua_cte);

        gsl_matrix_set_zero(f);
```

```
/*--Fim da Declaração de Variáveis--*/

/*--Início do Problema Direto--- */

        coordenadas(x,y); /* registra as
coordenadas dos nós nos vetores x e y */

        topologia(nos); /* topologia -
pega os nós que compõem cada elemento. */

        centroides (x,y,xg,yg,nos);

        pd1(x, y, nos, M, H, xg, yg);
/* Calcula as matrizes M e H do problema
direto */

        pd2(M, Npd, H,mua); /*Calcula os
coeficientes de absorção e distribuição e
soma as matrizes M e H para gerar a matriz
Npd*/

        matrizglobal(NGpd,Npd,nos);
/*Globaliza a matriz Npd na matriz NGpd*/

        gera_coluna_lambda_k (I, phi,
NGpd, lambda0,x,y);

        print_phi_to_file(x,y,phi,
imprimir);

        print_lambda_to_file(x, y,
lambda0, imprimir);
/*---Fim do Problema Direto------- */

/*--Início do Teste do PD--*/
imprimir=1;
gsl_vector *mu=gsl_vector_alloc(numel);
gsl_vector_set_all(mu, mua_cte);
gsl_vector_set(mu,36,(1+var)*mua_cte);
gsl_vector_set(mu,37,(1+var)*mua_cte);
gsl_vector_set(mu,99,(1+var)*mua_cte);
gsl_vector_set(mu,100,(1+var)*mua_cte);
gsl_vector_set(mu,101,(1+var)*mua_cte);

int cont;
double aux1,aux2;
aux1=aux2=0;
FILE *fpp = fopen ("Sombra.txt","w");
for (cont=0;cont<numel;cont++) {
aux1= 1/(3*(gsl_vector_get(mu,cont) +
mus));
aux2= 1/(3*(gsl_vector_get(mua,cont) +
mus));
fprintf(fpp," %lf  %lf  %lf \n",
gsl_vector_get(xg,cont), gsl_vector_get
(yg,cont), aux2-aux1);
}
fclose (fpp);


pd2(M, Npd, H,mu); /* Recalcula a matriz
Npd com o valor novo de mua*/
matrizglobal(NGpd,Npd,nos); /*Rearranja a
matriz global*/
gera_coluna_lambda_k (I, phi, NGpd,
lambdak,x,y); /*Calcula uma coluna k da
matriz lambda*/
gsl_vector_memcpy (testepi,lambdak);
print_phi_to_file(x,y,phi, imprimir);
print_lambda_to_file(x, y, lambdak,
imprimir);

/*---Fim do Teste do PD----------*/
```

```
/*--Início do Problema Inverso--*/

/* Criação da Matriz Lambda */
int elem;
for (elem=0; elem < numel; elem++){
gsl_vector_set_all (mua, mua_cte);
gsl_vector_set (mua, elem,
(1+var)*mua_cte);
gsl_matrix_set_zero (Npd);
pd2(M, Npd, H,mua);
matrizglobal(NGpd,Npd,nos);
gera_coluna_lambda_k (I, phi, NGpd,
lambdak,x,y);
gera_matriz_lambda (lambda0, lambdak,
lambda, elem);
}

int lin,col;
FILE *fp;
fp = fopen("matriz_lambda.txt", "w");
for (lin=0; lin< numled*numled; lin++){
        fprintf(fp, "\n");
  for (col=0; col<numel; col++)
        fprintf(fp, " %lf
",gsl_matrix_get(lambda, lin, col));
}
fclose(fp);

/*---Problema Inverso---*/
gsl_matrix_set_zero (b);
gsl_matrix_set_zero (B_irreg);
gsl_matrix_set_zero (B_reg);
gsl_matrix_set_identity (teta);
gsl_matrix_scale (teta, var);

calc_F(xg,yg,f, std_devi);
calc_M(xg,yg,m,pot,raio);

gsl_blas_dgemm(CblasTrans,CblasNoTrans,1.0
,lambda,lambda,alfa,f);
gsl_matrix_scale(m,beta);
gsl_matrix_add(f,m);
gsl_permutation *p = gsl_permutation_alloc
(numel);
gsl_permutation_init (p);
int s;
gsl_linalg_LU_decomp(f,p,&s);
gsl_linalg_LU_invert(f,p,f_inv);
gsl_blas_dgemm(CblasNoTrans,CblasNoTrans,1
.0,f_inv,teta,0.0,b);
gsl_blas_dgemm(CblasNoTrans,CblasTrans,1.0
,b,lambda,0.0,B_reg);

/*--Fim do Problema Inverso---*/

/*--Teste do Problema Inverso--*/
aux1=0;
FILE *bh =
fopen("Vetor_sombra_leite_11_10_12v2.txt",
"r");
for (cont=0;cont<numled*numled;cont++){
        fscanf(bh, "%lf ", &aux1);
        gsl_vector_set(testepi,cont,aux1);
}

gsl_vector *fim= gsl_vector_alloc(numel);
gsl_vector_set_zero(fim);
gsl_blas_dgemv(CblasNoTrans,1.0,B_reg,test
epi,0.0,fim);
fp = fopen ("fim.txt","w");
for (cont=0;cont<numel;cont++)
```

```
fprintf(fp," %lf  %lf  %lf \n",
gsl_vector_get(xg,cont), gsl_vector_get
(yg,cont), gsl_vector_get(fim,cont));
fclose (fp);

system ("gnuplot fim.gnu");

/*--Fim Teste do Problema Inverso---*/
gsl_vector_free(mua);
gsl_vector_free(x);
gsl_vector_free(y);
gsl_vector_free(xg);
gsl_vector_free(yg);
gsl_vector_free(lambda0);
gsl_vector_free(lambdak);
gsl_vector_free(lambdak1);
gsl_vector_free(phi);

gsl_matrix_free(Npd);
gsl_matrix_free(NGpd);
gsl_matrix_free(M);
gsl_matrix_free(H);
gsl_matrix_free(teta);
gsl_matrix_free(f);
gsl_matrix_free(b);
gsl_matrix_free(lambda);
return 0;       }

/* DECLARAÇÃO DE FUNÇÕES */

int coordenadas(gsl_vector *x, gsl_vector
*y){
int cont1,num;
double aux1, aux2;
cont1 = num =0;
aux1 = aux2 = 0;

num=x->size;

FILE *arq1 = fopen ("nos.txt","r");

if (arq1 == NULL) {
printf("ERRO: arquivo de entrada nao
encontrado\n");
return -1;      }
while (cont1<num) {
fscanf(arq1, "%lf %lf", &aux1, &aux2);
gsl_vector_set(x,cont1,aux1);
gsl_vector_set(y,cont1,aux2);
cont1 = cont1 + 1;       }
fclose(arq1);
return 0;       }

int topologia (gsl_vector_int *nos)
{
/* Seta no gsl_vector*nos os nós i,j,m que
compõem cada um dos elementos */

int cont1;
int i, j, m;
int num;
num=nos->size;
cont1 = i = j= m = 0;

FILE*arq2 = fopen ("elem.txt","r");

if (arq2 == NULL) {
printf("ERRO: arquivo de entrada nao
encontrado\n");
return -1;         }

while (cont1<num) {
fscanf(arq2, "%d %d %d", &i, &j, &m);
gsl_vector_int_set(nos, cont1, i);
cont1 = cont1 + 1;
gsl_vector_int_set(nos, cont1, j);
cont1 = cont1 + 1;
gsl_vector_int_set(nos, cont1, m);
cont1 = cont1 + 1;              }
fclose(arq2);
return 0;       }

void matrizglobal(gsl_matrix *NG,
gsl_matrix *N, gsl_vector_int *nos){
int cont1, aux1, aux2, aux3;
double help1, help2;

cont1 = aux1 = aux2 = aux3 = 0;
help1 = help2 = 0;
for(cont1 = 0; cont1 < numel; cont1++)  {

aux1 = cont1*3;
aux2 = gsl_vector_int_get(nos, aux1) - 2;
aux3 = gsl_vector_int_get(nos, aux1) - 2;
help1 = gsl_matrix_get(NG, aux2, aux3);
help2 = gsl_matrix_get(N, cont1, 0);
help1 = help1+help2;
gsl_matrix_set(NG, aux2, aux3, help1);

aux2 = gsl_vector_int_get(nos, aux1) - 2;
aux3 = gsl_vector_int_get(nos, aux1+1) -
2;
help1 = gsl_matrix_get(NG, aux2, aux3);
help2 = gsl_matrix_get(N, cont1, 1);
help1 = help1+help2;
gsl_matrix_set(NG, aux2, aux3, help1);

aux2 = gsl_vector_int_get(nos, aux1) - 2;
aux3 = gsl_vector_int_get(nos, aux1+2) -
2;
help1 = gsl_matrix_get(NG, aux2, aux3);
help2 = gsl_matrix_get(N, cont1, 2);
help1 = help1+help2;
gsl_matrix_set(NG, aux2, aux3, help1);

aux2 = gsl_vector_int_get(nos, aux1+1) -
2;
aux3 = gsl_vector_int_get(nos, aux1) - 2;
help1 = gsl_matrix_get(NG, aux2, aux3);
help2 = gsl_matrix_get(N, cont1, 3);
help1 = help1+help2;
gsl_matrix_set(NG, aux2, aux3, help1);

aux2 = gsl_vector_int_get(nos, aux1+1) -
2;
aux3 = gsl_vector_int_get(nos, aux1+1) -
2;
help1 = gsl_matrix_get(NG, aux2, aux3);
help2 = gsl_matrix_get(N, cont1, 4);
help1 = help1+help2;
gsl_matrix_set(NG, aux2, aux3, help1);

aux2 = gsl_vector_int_get(nos, aux1+1) -
2;
aux3 = gsl_vector_int_get(nos, aux1+2) -
2;
help1 = gsl_matrix_get(NG, aux2, aux3);
help2 = gsl_matrix_get(N, cont1, 5);
help1 = help1+help2;
gsl_matrix_set(NG, aux2, aux3, help1);

aux2 = gsl_vector_int_get(nos, aux1+2) -
2;
aux3 = gsl_vector_int_get(nos, aux1) - 2;
help1 = gsl_matrix_get(NG, aux2, aux3);
help2 = gsl_matrix_get(N, cont1, 6);
help1 = help1+help2;
gsl_matrix_set(NG, aux2, aux3, help1);
```

```
aux2 = gsl_vector_int_get(nos, aux1+2) -
2;
aux3 = gsl_vector_int_get(nos, aux1+1) -
2;
help1 = gsl_matrix_get(NG, aux2, aux3);
help2 = gsl_matrix_get(N, cont1, 7);
help1 = help1+help2;
gsl_matrix_set(NG, aux2, aux3, help1);
aux2 = gsl_vector_int_get(nos, aux1+2) -
2;
aux3 = gsl_vector_int_get(nos, aux1+2) -
2;
help1 = gsl_matrix_get(NG, aux2, aux3);
help2 = gsl_matrix_get(N, cont1, 8);
help1 = help1+help2;
gsl_matrix_set(NG, aux2, aux3, help1);
}}

void pd1 (gsl_vector *x, gsl_vector *y,
gsl_vector_int *nos, gsl_matrix *M,
gsl_matrix *H,gsl_vector *xg, gsl_vector
*yg) {

int aux1, aux2, cont1;
double Ai, Aj, Am, Bi, Bj, Bm, Gi, Gj, Gm;
double xi, xj, xm, yi, yj, ym, xgg, ygg,
a;
double XIS,YIS,XYIS, aux;

gsl_vector *A = gsl_vector_alloc (numel);
gsl_vector *alfai = gsl_vector_alloc
(numel);
gsl_vector *alfaj = gsl_vector_alloc
(numel);
gsl_vector *alfak = gsl_vector_alloc
(numel);
gsl_vector *betai = gsl_vector_alloc
(numel);
gsl_vector *betaj = gsl_vector_alloc
(numel);
gsl_vector *betak = gsl_vector_alloc
(numel);
gsl_vector *gamai=  gsl_vector_alloc
(numel);
gsl_vector *gamaj = gsl_vector_alloc
(numel);
gsl_vector *gamak = gsl_vector_alloc
(numel);
gsl_vector *xis = gsl_vector_alloc
(numel);
gsl_vector *yis = gsl_vector_alloc
(numel);
gsl_vector *xyis = gsl_vector_alloc
(numel);

gsl_vector_set_zero (A);
gsl_vector_set_zero (alfai);
gsl_vector_set_zero (alfaj);
gsl_vector_set_zero (alfaj);
gsl_vector_set_zero (betai);
gsl_vector_set_zero (betaj);
gsl_vector_set_zero (betak);
gsl_vector_set_zero (gamai);
gsl_vector_set_zero (gamaj);
gsl_vector_set_zero (gamak);
gsl_vector_set_zero (xis);
gsl_vector_set_zero (yis);
gsl_vector_set_zero (xyis);


Ai= Aj= Am= Bi= Bj= Bm= Gi= Gj= Gm= 0;
xi= xj= xm= yi= yj= ym= xgg= ygg= a= 0;
XIS= YIS= XYIS= aux=0;
```

```
for(cont1 = 0; cont1 < numel; cont1++) {

aux1 = cont1*3;
aux2 = gsl_vector_int_get(nos, aux1) -2;
xi = gsl_vector_get(x,aux2);
yi = gsl_vector_get(y,aux2);
aux1 = aux1+1;
aux2 = gsl_vector_int_get(nos, aux1) - 2;
xj = gsl_vector_get(x,aux2);
yj = gsl_vector_get(y,aux2);
aux1 = aux1+1;
aux2 = gsl_vector_int_get(nos, aux1) - 2;
xm = gsl_vector_get(x,aux2);
ym = gsl_vector_get(y,aux2);
a = (xj*ym+xm*yi+xi*yj-xj*yi-xm*yj-
xi*ym)/2;
gsl_vector_set (A,cont1,a);
aux = (xj*ym-xm*yj);
gsl_vector_set(alfai, cont1, aux);
aux = -(xi*ym-xm*yi);
gsl_vector_set(alfaj, cont1, aux);
aux = (xi*yj-xj*yi);
gsl_vector_set(alfak, cont1, aux);

aux = -(ym-yj);
gsl_vector_set(betai, cont1, aux);
aux = (ym-yi);
gsl_vector_set(betaj, cont1, aux);

aux = -(yj-yi);
gsl_vector_set(betak, cont1, aux);

aux = (xm-xj);
gsl_vector_set(gamai, cont1, aux);

aux = -(xm-xi);
gsl_vector_set(gamaj, cont1, aux);

aux = (xj-xi);
gsl_vector_set(gamak, cont1, aux);}

for(cont1 = 0; cont1 < numel; cont1++) {
Ai = gsl_vector_get(alfai, cont1);
Aj = gsl_vector_get(alfaj, cont1);
Am = gsl_vector_get(alfak, cont1);
Bi = gsl_vector_get(betai, cont1);
Bj = gsl_vector_get(betaj, cont1);
Bm = gsl_vector_get(betak, cont1);
Gi = gsl_vector_get(gamai, cont1);
Gj = gsl_vector_get(gamaj, cont1);
Gm = gsl_vector_get(gamak, cont1);
/*------Calculando a Matriz M---------*/
aux = Bi*Bi + Gi*Gi;
aux = aux/gsl_vector_get(A,cont1);
gsl_matrix_set(M, cont1, 0, aux);
aux = Bi*Bj + Gi*Gj;
aux = aux/gsl_vector_get(A,cont1);
gsl_matrix_set(M, cont1, 1, aux);
gsl_matrix_set(M, cont1, 3, aux);
aux = Bi*Bm + Gi*Gm;
aux = aux/gsl_vector_get(A,cont1);
gsl_matrix_set(M, cont1, 2, aux);
gsl_matrix_set(M, cont1, 6, aux);
aux = Bj*Bj + Gj*Gj;
aux = aux/gsl_vector_get(A,cont1);
gsl_matrix_set(M, cont1, 4, aux);
aux = Bm*Bj + Gm*Gj;
aux = aux/gsl_vector_get(A,cont1);
gsl_matrix_set(M, cont1, 5, aux);
gsl_matrix_set(M, cont1, 7, aux);
aux = Bm*Bm + Gm*Gm;
aux = aux/gsl_vector_get(A,cont1);
gsl_matrix_set(M, cont1, 8, aux); }
```

```
/*--Mudando as Coordenadas--*/

for(cont1 = 0; cont1 < numel; cont1++) {

aux1 = cont1*3;
aux2 = gsl_vector_int_get(nos, aux1) -2;
xi = gsl_vector_get(x,aux2)-
gsl_vector_get(xg,cont1);
yi = gsl_vector_get(y,aux2)-
gsl_vector_get(yg,cont1);
aux1 = aux1+1;
aux2 = gsl_vector_int_get(nos, aux1) - 2;
xj = gsl_vector_get(x,aux2)-
gsl_vector_get(xg,cont1);
yj = gsl_vector_get(y,aux2)-
gsl_vector_get(yg,cont1);
aux1 = aux1+1;
aux2 = gsl_vector_int_get(nos, aux1) - 2;
xm = gsl_vector_get(x,aux2)-
gsl_vector_get(xg,cont1);
ym = gsl_vector_get(y,aux2)-
gsl_vector_get(yg,cont1);

a = (xj*ym+xm*yi+xi*yj-xj*yi-xm*yj-
xi*ym)/2;
gsl_vector_set (A,cont1,a);

aux = (xj*ym-xm*yj);
gsl_vector_set(alfai, cont1, aux);
aux = -(xi*ym-xm*yi);
gsl_vector_set(alfaj, cont1, aux);
aux = (xi*yj-xj*yi);
gsl_vector_set(alfak, cont1, aux);

aux = -(ym-yj);
gsl_vector_set(betai, cont1, aux);

aux = (ym-yi);
gsl_vector_set(betaj, cont1, aux);

aux = -(yj-yi);
gsl_vector_set(betak, cont1, aux);

aux = (xm-xj);
gsl_vector_set(gamai, cont1, aux);

aux = -(xm-xi);
gsl_vector_set(gamaj, cont1, aux);

aux = (xj-xi);
gsl_vector_set(gamak, cont1, aux);

XIS= (xi*xi+xj*xj+xm*xm)/12;
YIS= (yi*yi+yj*yj+ym*ym)/12;
XYIS= (xi*yi+xj*yj+xm*ym)/12;

gsl_vector_set (xis,cont1,XIS);
gsl_vector_set (yis,cont1,YIS);
gsl_vector_set (xyis,cont1,XYIS); }
/*----Calculando a Matriz H----*/
for(cont1 = 0; cont1 < numel; cont1++) {
Ai = gsl_vector_get(alfai, cont1);
Aj = gsl_vector_get(alfaj, cont1);
Am = gsl_vector_get(alfak, cont1);
Bi = gsl_vector_get(betai, cont1);
Bj = gsl_vector_get(betaj, cont1);
Bm = gsl_vector_get(betak, cont1);
Gi = gsl_vector_get(gamai, cont1);
Gj = gsl_vector_get(gamaj, cont1);
Gm = gsl_vector_get(gamak, cont1);

XIS=gsl_vector_get(xis,cont1);
YIS=gsl_vector_get(yis,cont1);
XYIS=gsl_vector_get(xyis,cont1);
aux =
Ai*Ai+(Bi*Bi*XIS)+(Gi*Gi*YIS)+((Bi*Gi+Bi*G
i)*XYIS);
aux = aux/gsl_vector_get(A,cont1);
gsl_matrix_set(H, cont1, 0, aux);
aux =
Ai*Aj+(Bi*Bj*XIS)+(Gi*Gj*YIS)+((Bi*Gj+Bj*G
i)*XYIS);
aux = aux/gsl_vector_get(A,cont1);
gsl_matrix_set(H, cont1, 1, aux);
gsl_matrix_set(H, cont1, 3, aux);
aux =
Ai*Am+(Bi*Bm*XIS)+(Gi*Gm*YIS)+((Bi*Gm+Bm*G
i)*XYIS);
aux = aux/gsl_vector_get(A,cont1);
gsl_matrix_set(H, cont1, 2, aux);
gsl_matrix_set(H, cont1, 6, aux);
aux =
Aj*Aj+(Bj*Bj*XIS)+(Gj*Gj*YIS)+((Bj*Gj+Bj*G
j)*XYIS);
aux = aux/gsl_vector_get(A,cont1);
gsl_matrix_set(H, cont1, 4, aux);
aux =
Aj*Am+(Bj*Bm*XIS)+(Gj*Gm*YIS)+((Bj*Gm+Bm*G
j)*XYIS);
aux = aux/gsl_vector_get(A,cont1);
gsl_matrix_set(H, cont1, 5, aux);
gsl_matrix_set(H, cont1, 7, aux);
aux =
Am*Am+(Bm*Bm*XIS)+(Gm*Gm*YIS)+((Bm*Gm+Bm*G
m)*XYIS);
aux = aux/gsl_vector_get(A,cont1);
gsl_matrix_set(H, cont1, 8, aux);

}}


void pd2(gsl_matrix *M, gsl_matrix *Npd,
gsl_matrix *H, gsl_vector *mua){
int lin, col;
double aux, aux1, aux2, D;

aux= aux1= aux2= D= 0;

for(lin = 0; lin < numel; lin++){
for(col = 0; col < 9; col++){
D= 1/(3*((gsl_vector_get(mua,lin))+mus));
aux1 = gsl_matrix_get(M, lin, col);
aux2 = gsl_matrix_get(H, lin, col);

aux=(aux1*D+aux2*(gsl_vector_get(mua,lin))
)*t/4;

gsl_matrix_set(Npd, lin, col, aux);
}}}

void pd3(gsl_matrix *Ngb, gsl_vector *I,
gsl_vector *phi,int LED)

{
  /* DECLARANDO VARIAVEIS */
  int s,col;
  int size_Ngb;

size_Ngb = Ngb->size1;
gsl_matrix *Ngb_copia =
gsl_matrix_alloc(size_Ngb,size_Ngb);
gsl_matrix *Ngb_inv =
gsl_matrix_alloc(size_Ngb,size_Ngb);
gsl_matrix_memcpy(Ngb_copia,Ngb);
gsl_matrix_set_zero (Ngb_inv);

gsl_vector_set(I,2*LED,fonte);
```

```
  /* ALOCANDO MEMORIA */
  /*Resolvendo pela forma tradicional*/
gsl_permutation * z =
gsl_permutation_alloc(size_Ngb);
gsl_permutation_init (z);
gsl_linalg_LU_decomp(Ngb_copia,z,&s);

gsl_linalg_LU_invert(Ngb_copia,z,Ngb_inv);
gsl_blas_dgemv
(CblasNoTrans,1.0,Ngb_inv,I,0.0,phi);

/* LIBERANDO MEMORIA */
gsl_matrix_free(Ngb_copia);
gsl_matrix_free(Ngb_inv);
gsl_permutation_free(z); }

void gera_coluna_lambda_k (gsl_vector *I,
gsl_vector *phi, gsl_matrix *NGpd,
gsl_vector *lambda0, gsl_vector *x,
gsl_vector *y){

int cont,LED=0;
double aux2=0;
int aux=2;
for (LED=0; LED< numled; LED++){
gsl_vector_set_zero(phi);
gsl_vector_set_zero(I);
pd3(NGpd, I, phi,LED); aux = numled*LED;
for (cont=0; cont< numled; cont++){
aux2 = gsl_vector_get(phi,2*cont);
gsl_vector_set(lambda0,aux+cont,aux2);
}}}

void print_phi_to_file(gsl_vector *x,
gsl_vector *y, gsl_vector *phi, int
imprimir){
int aux = 0;
double a,b,p=0;
int size_phi;
size_phi = phi->size;
FILE *fp;

if (imprimir==0)
fp = fopen("phi_0.txt", "w");
else {
if (imprimir==1)
fp = fopen("phi_k.txt", "w");

else
fp = fopen("phi_k1.txt", "w");        }

for (aux=0; aux < size_phi; aux++){
a=gsl_vector_get(x,aux);
b=gsl_vector_get(y,aux);
p=gsl_vector_get(phi,aux);
fprintf(fp, "%lf %lf %lf \n",a,b,p);
}
fclose(fp);
}

void print_lambda_to_file(gsl_vector *x,
gsl_vector *y, gsl_vector *lambda0, int
imprimir){
int size_lambda,aux;
double p=0;
size_lambda = lambda0->size;
FILE *fp;

if (imprimir==0)
fp = fopen("Lambda_Elemento_1.txt", "w");
else {
if (imprimir==1)
fp = fopen("Lambda_Elemento_2.txt", "w");
else {
if (imprimir==2)
fp = fopen("Lambda_Elemento_3.txt", "w");
else {
if (imprimir==3)
fp = fopen("Lambda_Elemento_4.txt", "w");
else {
if (imprimir==4)
fp = fopen("Lambda_Elemento_5.txt", "w");
else {
if (imprimir==5)
fp = fopen("Lambda_Elemento_6.txt", "w");
else {
if (imprimir==6)
fp = fopen("Lambda_Elemento_7.txt", "w");
else {
if (imprimir==7)
fp = fopen("Lambda_Elemento_8.txt", "w");
else {
if (imprimir==8)
fp = fopen("Lambda_Elemento_9.txt", "w");
else {
if (imprimir==9)
fp = fopen("Lambda_Elemento_10.txt", "w");
else {
if (imprimir==10)
fp = fopen("Lambda_Elemento_11.txt", "w");
else
fp = fopen("Lambda_Elemento_12.txt", "w");
}}}}}}}}}

for (aux=0; aux < size_lambda ; aux++){
p=gsl_vector_get(lambda0,aux);
fprintf(fp, " %lf \n",p);        }
fclose(fp);      }

void gera_matriz_lambda (gsl_vector
*lambda0, gsl_vector *lambdak, gsl_matrix
*lambda, int elem){

int cont=0;
double aux1, aux2;

aux1=aux2=0;

for (cont=0; cont < numled*numled;
cont++){

aux1= gsl_vector_get (lambda0,cont);
aux2= gsl_vector_get (lambdak,cont);
gsl_matrix_set (lambda, cont, elem, aux2-
aux1);          }}

void centroides (gsl_vector *x, gsl_vector
*y,gsl_vector *xg, gsl_vector *yg,
gsl_vector_int *nos){
int cont1=0;
int cont2=0;
int no=0;
double aux1,aux2;

for (cont1=0;cont1<numel;cont1++){

aux1=aux2=0;

for (cont2=0;cont2<3;cont2++){

no=gsl_vector_int_get(nos,3*cont1+cont2)-
2;
aux1+=gsl_vector_get(x,no);
aux2+=gsl_vector_get(y,no);
}
gsl_vector_set(xg,cont1,aux1/3);
gsl_vector_set(yg,cont1,aux2/3);
```

```
}}

void calc_F(gsl_vector *xg,gsl_vector
*yg,gsl_matrix *f,double std_dev) {
/* DECLARANDO VARIAVEIS */
int i;
int j;
int nelements;
double dist;
double dist_x;
double dist_y;
double soma;
double fator;

nelements=f->size1;

/* ALOCANDO MEMORIA */
gsl_matrix
*F=gsl_matrix_alloc(nelements,nelements);

for (i=0; i<nelements; i++) {
soma = 0.0;
for (j=0; j<nelements; j++) {

dist_x =  fabs(gsl_vector_get(xg,j) -
gsl_vector_get(xg,i));

dist_y =  fabs(gsl_vector_get(yg,j) -
gsl_vector_get(yg,i));

dist=sqrt(dist_x*dist_x+dist_y*dist_y);

if (i==j){
fator=1.0;      }
else {
fator=exp(-dist*dist/(2*std_dev*std_dev));
soma+=fator;

gsl_matrix_set(F,i,j,fator);    }}

for (j=0; j<nelements; j++){
if (i==j){
gsl_matrix_set(F,i,j,(1.0-
gsl_matrix_get(F,i,j)/soma));    }
else {
gsl_matrix_set(F,i,j,(-
gsl_matrix_get(F,i,j)/soma));     }}}

gsl_blas_dgemm(CblasTrans,CblasNoTrans,1.0
,F,F,0.0,f);
gsl_matrix_free(F);      }

void calc_M(gsl_vector *xg,gsl_vector
*yg,gsl_matrix *m,double potencia, double
raio_dominio){

int nelements, cont;
double aux1,aux2,distancia;

nelements=m->size1;

gsl_matrix *M =
gsl_matrix_alloc(nelements,nelements);

gsl_matrix_set_zero(M);

for(cont=0;cont<nelements;cont++){

aux1 =
gsl_vector_get(xg,cont)*gsl_vector_get(xg,
cont);
aux2 =
gsl_vector_get(yg,cont)*gsl_vector_get(yg,
cont);
distancia =
pow(((sqrt(aux1+aux2))/raio_dominio),poten
cia);
gsl_matrix_set(M,cont,cont,distancia);   }

'gsl_blas_dgemm(CblasTrans,CblasNoTrans,
1.0,M,M,0.0,m);

}
```