

**LUCIANO FRONTINO DE MEDEIROS**

**RECONSTRUÇÃO DE IMAGENS TOMOGRÁFICAS COM REDES  
NEURAS PARCIALMENTE CONECTADAS**

Dissertação apresentada como requisito parcial à  
obtenção do título de Mestre, Curso de Pós-  
Graduação em Informática, Setor de Ciências  
Exatas, Universidade Federal do Paraná.

Orientador: Prof. Dr. Eduardo Parente Ribeiro

CURITIBA  
2001

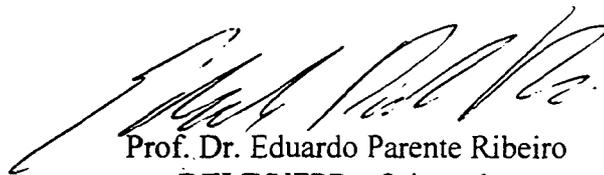


Ministério da Educação  
Universidade Federal do Paraná  
Mestrado em Informática

## PARECER

Nós, abaixo assinados, membros da Banca Examinadora da defesa de Dissertação de Mestrado em Informática do aluno **Luciano Frontino de Medeiros**, avaliamos o trabalho intitulado "**Reconstrução de Imagens Tomográficas com Redes Neurais Parcialmente Conectadas**", cuja defesa foi realizada no dia 30 de agosto de 2001, às quatorze horas, no anfiteatro B, do Setor de Ciências Exatas da Universidade Federal do Paraná. Após a avaliação, decidimos pela aprovação do candidato.

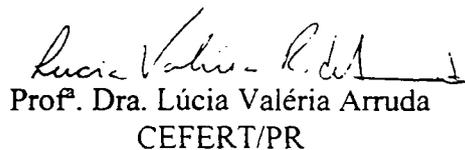
Curitiba, 30 de Agosto de 2001.



Prof. Dr. Eduardo Parente Ribeiro  
DELT/UFPR - Orientador



Prof.ª Dra. Olga Regina Pereira Bellon  
DINF/UFPR



Prof.ª Dra. Lúcia Valéria Arruda  
CEFERT/PR

LUCIANO FRONTINO DE MEDEIROS

**RECONSTRUÇÃO DE IMAGENS TOMOGRÁFICAS COM REDES  
NEURAI PARCIALMENTE CONECTADAS**

Dissertação aprovada como requisito parcial para obtenção do grau de Mestre no  
Curso de Pós-Graduação em Informática da Universidade Federal do Paraná,  
pela comissão formada pelos professores:

Orientador:

Prof. Dr. Eduardo Parente Ribeiro

Departamento de Engenharia Elétrica, UFPR

Prof. Dr. Olga Regina Bellon

Departamento de Ciências da Computação, UFPR.

Prof. Dr. Lúcia Valéria R. de Arruda

CPGEI / CEFET-PR

Curitiba, agosto de 2001

## AGRADECIMENTOS

Em primeiro lugar, agradeço a Deus por permitir chegar onde estou, todas as coisas boas pelas quais passei e pelas oportunidades que tive.

Agradeço a meus pais, Osvaldo e Maria Delourdes, por estar aqui, pela educação que recebi e pelo tempo que convivi e convivo, bem como a meus irmãos Leonardo, Carolina e Leandro por todos os momentos bons e não muito bons.

Agradeço e dedico este trabalho à minha esposa Paula pelo amor e paciência nesta fase de conclusão da dissertação, e também à luz da minha vida, Amanda.

Em especial agradeço ao amigo e irmão Hamilton pelas oportunidades de crescimento profissional e espiritual que temos passado, à sua esposa Tânia pela paciência e filhos.

Ao meu professor orientador Eduardo Parente Ribeiro, pelo esclarecimento de minhas dúvidas e seu apoio na orientação deste trabalho.

Aos professores que me auxiliaram na vida acadêmica dedicando uma parcela de seu precioso tempo, aos amigos presentes e distantes que me apoiaram em momentos difíceis e aos amigos sinceros que porventura terei, deixo meu agradecimento em especial.

## SUMÁRIO

<b>LISTA DE FIGURAS.....</b>	<b>VII</b>
<b>LISTA DE TABELAS.....</b>	<b>IX</b>
<b>RESUMO.....</b>	<b>X</b>
<b>ABSTRACT .....</b>	<b>XI</b>
<b>1 INTRODUÇÃO .....</b>	<b>1</b>
<b>2 HISTÓRICO .....</b>	<b>3</b>
<b>3 FUNDAMENTOS DE TOMOGRAFIA.....</b>	<b>7</b>
3.1 A TRANSFORMADA DE RADON.....	7
3.2 O OPERADOR RETROPROJEÇÃO .....	12
3.3 MÉTODOS DE RESOLUÇÃO.....	14
<b>4 REDES NEURAIS.....</b>	<b>21</b>
4.1 CONCEITOS BÁSICOS .....	21
4.2 O PERCEPTRON MULTICAMADAS .....	22
4.3 A FUNÇÃO DE ATIVAÇÃO.....	23
4.4 REGRA DO DELTA GENERALIZADO.....	25
4.5 PERCEPTRON USANDO EXPURGO DE PESOS.....	25
<b>5 REDE PARCIALMENTE CONECTADA PARA RECONSTRUÇÃO DE CT .....</b>	<b>28</b>
5.1 REDE DE RETROPROJEÇÃO (BACKPROJECTION) .....	28
5.2 REDE DE FILTRAGEM.....	47
5.3 RECONSTRUÇÃO EM 3D COM REDES PARCIALMENTE CONECTADAS .....	53
5.4 QUALIDADE DE IMAGEM.....	57
5.5 PROCESSAMENTO PARALELO .....	58
<b>6 RESULTADOS.....</b>	<b>60</b>
6.1 TEMPOS DE CONSTRUÇÃO, EXECUÇÃO E QUALIDADE DA IMAGEM.....	62
6.2 ANÁLISE DO NÚMERO DE COEFICIENTES NA REDE DE FILTRAGEM .....	63
6.3 COMPARAÇÃO DA REDE PARCIALMENTE CONECTADA COM PERCEPTRON DE DUAS CAMADAS .....	71
6.4 ALGORITMO UTILIZANDO PROCESSAMENTO PARALELO .....	73

<b>7 CONCLUSÃO .....</b>	<b>78</b>
<b>APÊNDICE A - ALGORITMO DE RETROPROJEÇÃO FILTRADA .....</b>	<b>80</b>
<b>APÊNDICE B - DESCRIÇÃO DOS RECURSOS COMPUTACIONAIS ...</b>	<b>82</b>
<b>APÊNDICE C - O TEOREMA DA FATIA DE FOURIER.....</b>	<b>83</b>
<b>ANEXO A - ARTIGO INTITULADO "RECONSTRUÇÃO DE IMAGENS TOMOGRÁFICAS UTILIZANDO REDES NEURAS PARCIALMENTE CONECTADAS" APRESENTADO NO V CONGRESSO BRASILEIRO DE REDES NEURAS .....</b>	<b>86</b>
<b>REFERÊNCIAS .....</b>	<b>93</b>

## LISTA DE FIGURAS

Figura 1 – Representação esquemática da atenuação da radiação atravessando um objeto.....	8
Figura 2 – Representação no plano cartesiano e plano de projeção .....	9
Figura 3 – Representação no plano cartesiano da relação entre as coordenadas das projeções e da imagem.....	10
Figura 4 – Representação básica de um neurônio .....	23
Figura 5 – Funções de ativação, em ordem: deslocamento, sinal e sigmóide.....	24
Figura 6 – Processo de treinamento de uma rede perceptron multicamada.....	27
Figura 7 – Visualização gráfica da equação (35) em forma de rede neural.....	29
Figura 8 – Senóides sobre o espaço de projeções filtradas.....	29
Figura 9 – Contribuição dos pontos $(s, \theta)$ sobre a senóide para o ponto $(r, \phi)$ da imagem reconstruída...	30
Figura 10 – Representação das redes totalmente conectada e parcialmente conectada.....	31
Figura 11 – Representação do vetor da estrutura dos pesos .....	33
Figura 12 – Representação da estrutura do nó (neurônio) da camada de de entrada.....	34
Figura 13 – Representação da estrutura do nó (neurônio) para a camada de saída .....	35
Figura 14 – Representação da estrutura da camada de entrada interligada.....	36
Figura 15 – Representação da estrutura da camada de saída interligada.....	38
Figura 16 – Representação da conexão dos pesos entre si e com os nós das camadas de entrada e de saída.....	39
Figura 17 – Eixo de coordenadas mostrando a região efetiva de imagem após a reconstrução .....	43
Figura 18 – Região aumentada da senóide sobre o espaço de projeções com os pontos $q(s, \theta)$ .....	45
Figura 19- Região aumentada da senóide sobre o espaço de projeções com os pontos $q(s, \theta)$ e os valores de $s_1$ (pixels de tom escuro) e $s_2$ .....	46
Figura 20- Diagrama esquemático mostrando o relacionamento das camadas das redes de filtragem e retroprojeção.....	47
Figura 21 - Resposta em frequência dos filtros .....	50
Figura 22 - Fluxograma com os passos necessários para a reconstrução de imagens 3D com listas ligadas.....	55
Figura 23 - Fluxograma dos passos necessários para a reconstrução 3D usando vetor de estruturas.....	56
Figura 24 - Head Phantom de Shepp-Logan utilizados para testes dos algoritmos.....	61
Figura 25 - Gráfico mostrando a relação do número de coeficientes com o tempo de construção de uma rede parcialmente conectada.....	64
Figura 26 - Gráfico do tempo de execução do algoritmo de redes com o número de coeficientes.....	65
Figura 27 - Gráfico mostrando a relação entre o número de coeficientes do filtro Ram-Lak e a qualidade da imagem medida em PSNR.....	66
Figura 28 - Gráfico mostrando a relação de PSNR com o número de coeficientes de filtro Ram-Lak para 256 pontos de 90 projeções .....	67
Figura 29 - Imagens reconstruídas da imagem com 100 pontos e 60 projeções.....	67

Figura 30 - Imagem reconstruída com redes a partir de 256 pontos e 90 projeções, utilizando 60 coeficientes de filtro .....	68
Figura 31 - Imagem reconstruída com redes a partir de 256 pontos e 90 projeções, utilizando 256 coeficientes de filtro .....	68
Figura 32 - Histograma mostrando os níveis de cinza das imagens reconstruídas para 100 pontos e 60 projeções sem interpolação.....	69
Figura 33 - Histograma mostrando os níveis de cinza das imagens reconstruídas para 100 pontos e 60 projeções com interpolação .....	70
Figura 34 - Histograma mostrando os níveis de cinza das imagens reconstruídas para 256 pontos e 90 projeções sem interpolação.....	70
Figura 35 - Histograma mostrando os níveis de cinza das imagens reconstruídas para 256 pontos e 90 projeções com interpolação .....	72
Figura 36 - Gráfico mostrando a evolução do PSNR da imagem reconstruída pelo perceptron .....	74
Figura 37 - Seção de um tubo de alumínio reconstruído com 100 pontos e 60 projeções utilizando algoritmo com processamento paralelo .....	75

## LISTA DE TABELAS

TABELA 1 – CORRESPONDÊNCIA ENTRE VARIÁVEIS DA EQUAÇÃO (36) e (41).....	49
TABELA 2 – NÚMERO DE COEFICIENTES POR VALOR DE LIMIAR .....	50
TABELA 3 - COMPARAÇÃO DE TEMPOS DE CONSTRUÇÃO DAS REDES .....	62
TABELA 4 - COMPARAÇÃO DE TEMPOS DE EXECUÇÃO ENTRE ALGORITMOS .....	63
TABELA 5 - COMPARAÇÃO ENTRE O PERCEPTRON E A REDE PARCIALMENTE CONECTADA .....	73

## RESUMO

Este trabalho trata do desenvolvimento de um algoritmo utilizando o conceito de redes neurais para reconstrução de imagens tomográficas. Mostramos que através de uma rede neural parcialmente conectada é possível reconstruir imagens com mais rapidez e mesma qualidade utilizando conceitos de tomografia em relação a algoritmos tradicionais. Senogramas referentes a várias seções de imagens de um determinado objeto podem ser reconstruídos por uma única rede parcialmente conectada. Esta estrutura não precisa ser treinada, sendo baseada na geometria da relação entre o espaço de projeção e o espaço da imagem final.

## **ABSTRACT**

This work presents a novel approach for tomographic image reconstruction by using neural networks. The reconstruction is performed by a partially connected neural network, using the same concepts from tomography. The processing time is smaller than traditional reconstruction yielding images with the same quality. Sinograms from different cross-sections of the object can be reconstructed by the same partially connected neural network. This structure doesn't need to be trained because it is based on the geometric relationship among the projection space and final image space.

## 1 INTRODUÇÃO

O foco desta dissertação está na apresentação de uma alternativa para reconstrução de imagens tomográficas através do paradigma de redes neurais. Uma vez que alguns trabalhos e artigos sobre esta interação já foram publicados na comunidade acadêmica, a idéia foi conciliar teoria e prática, construindo algoritmos e verificando a contrapartida em termos de vantagens e desvantagens para o processo de reconstrução e também aprofundando conceitos a nível de estruturas de dados.

Na atualidade existem uma variedade de algoritmos, e uma das metas para a reconstrução de imagens de tomografia computadorizada é a busca de velocidade e melhoria de qualidade de apresentação de imagens. Outro desafio à frente é a reconstrução 3D em tempo real, e os pesquisadores desta área dedicam boa parte de seus trabalhos nesta orientação. Uma vez que se consiga este objetivo, os benefícios são múltiplos para áreas tais como medicina e engenharia.

O objetivo deste trabalho é apresentar um algoritmo utilizando técnicas de redes neurais e processamento paralelo com a finalidade de reconstruir imagens de CT. Baseado no algoritmo de retroprojeção filtrada, uma rede parcialmente conectada sem necessidade de treinamento estará apta a realizar o processo de reconstrução, de maneira a reduzir o custo de processamento para projeções em seqüência (como em representação 3D de imagens tomográficas em pilha). Apresentar-se-á, então, as vantagens e desvantagens desta técnica. Como veremos na fundamentação teórica, a técnica de retroprojeção filtrada terá um isomorfismo com duas redes parcialmente conectadas, uma para a convolução (ou filtragem) dos dados e outra para a retroprojeção. Através desta abordagem, procuraremos explicar a diminuição da redundância inerente a um processo de reconstrução possibilitando assim uma alternativa para obtenção de imagens tomográficas em três dimensões.

A dissertação está estruturada de maneira a abordar os fundamentos de tomografia, partindo no capítulo 2 com o breve histórico sobre reconstrução de imagens de CT e redes neurais. No capítulo 3 aborda-se a expressão da Transformada de Radon e os operadores de retroprojeção, bem como a definição das alternativas para resolver a Transformada Inversa e obter a imagem a partir de suas projeções.

No capítulo 4 vem os conceitos básicos sobre Redes Neurais.

No capítulo 5 é explanado o processo de construção do algoritmo das redes a partir das definições discretas da teoria tomográfica, sendo abordado a rede de retroprojeção e após a rede de filtragem. Finalizando, explana-se a proposta de junção dos dois tipos de redes.

No capítulo 6 estão os resultados das medidas dos experimentos com a rede parcialmente conectada, a análise dos coeficientes dos filtros e a comparação com um perceptron modificado por expurgo de pesos, visto que existem trabalhos objetivando a reconstrução de imagens de CT com esta rede especificamente.

Desta forma, esta dissertação buscou contribuir para a área de tomografia computadorizada utilizando uma visão relativamente nova e direcionada para redes neurais e fundamentos de processamento paralelo, uma área de conhecimento e pesquisa crescente na comunidade acadêmica.

## 2 HISTÓRICO

A reconstrução de imagens tomográficas (também entendidas como CT, do inglês *Computerized Tomography*) foi fundamentada originalmente com os trabalhos matemáticos de Radon (RADON, 1917). Tais conceitos serviram de base para os trabalhos de A. Cormack (CORMACK, 1963) e para a construção do primeiro tomógrafo computadorizado por G. Housenfield (HOUSENFIELD, 1972). Os conceitos apresentados por Radon também foram utilizados em outras áreas como radioastronomia e microscopia eletrônica.

Com a evolução dos computadores, os tomógrafos proliferaram e técnicas mais eficientes foram desenvolvidas. Os algoritmos de reconstrução utilizando retroprojeção filtrada foram criados por Bracewell e Riddle (BRACEWELL ; RIDDLE, 1967) utilizados inicialmente em radioastronomia. Mas houve estudiosos (RAMACHANDRAN; LAKSHMINARAYANAN, 1971) que de forma independente desenvolveram uma implementação para reconstrução com dados de natureza discreta. Abordagens de filtragem com janelamento (windowing) foram aplicadas (HERMAN; ROWLAND, 1973), e outras opções introduzidas (SHEPP; LOGAN, 1974). Shepp e Logan provaram a eficiência do método de retroprojeção filtrada. Métodos de interpolação na retroprojeção também foram estudados (HERMAN *et al.*, 1978; OPPENHEIM, 1977; ROWLAND, 1979).

Trabalhos mais recentes abordam a questão do uso de filtros que trabalhem de forma otimizada em termos das restrições de tempo e frequência, utilizando a Transformada Curta de Fourier ou Transformadas Ortogonais Wavelet (SAHINER; YAGLE, 1998).

Outra abordagem diferenciada sobre reconstrução de imagens abrange métodos algébricos, ou ART (*Algebraic Reconstruction Techniques*), que são procedimentos iterativos que convergem para as imagens reconstruídas (KACZMARZ, 1937; GORDON, 1974). O método de resolução foi inicialmente proposto por Kaczmarz, com desenvolvimento posterior de Gordon. Tais

métodos no entanto ficaram sem foco devido à falta de eficácia e dificuldade de convergência com o aumento do tamanho da imagem.

Uma abordagem alternativa (MUNLEY, 1994; KERR; BARTLETT, 1995; RODRIGUES, 1995) reside na reconstrução de imagens tomográficas com o auxílio de *redes neurais artificiais* (RNA). As redes neurais artificiais (RNA) se constituem numa poderosa abstração computacional, capturada do paradigma biológico. Assim como um tecido nervoso contendo milhares de neurônios interconectados, células vivas desempenhando um determinado papel conforme o contexto onde se encontram, redes neurais computacionais são compostas para se chegar a um determinado objetivo, muitas vezes assemelhado a um processo biológico próprio (como um reconhecimento de imagem ou som). Porém, uma RNA apenas aproveita alguns conceitos subtraídos do processo biológico, visto que o entendimento completo das redes neurais reais ainda é um enigma. E o pouco conhecimento já foi o bastante para os cientistas formularem alguns conceitos básicos sobre o funcionamento e vislumbrar possíveis aplicações para as RNA. Podemos enumerar hoje em dia vários modelos de RNA. Muitos destes tipos estão relacionados à sua estrutura, à sua funcionalidade, e outras características.

Uma boa parte da abordagem histórica aqui descrita é atribuída a Simon Haykin (HAYKIN, 1994). O trabalho pioneiro que se aceita como tendo iniciado a era das redes neurais é atribuído a McCulloch e Pitts, de 1943, que descreveram um modelo simples de funcionamento de um neurônio. Em 1949, Hebb lançou a base para regras de aprendizado de redes, através de seu postulado de aprendizado, também baseado no comportamento de sistemas biológicos. Em meados dos anos 50, a idéia de filtros lineares adaptativos é trabalhada por Gabor (1954). O conceito de *memória associativa* iniciado por Taylor (1956) propôs novas idéias a respeito de aprendizado. Um dos modelos mais conhecidos (ROSENBLATT, 1958) é denominado *perceptron multicamada* (do original em inglês *Multilayer Perceptron*), proposto para reconhecimento de padrões. Widrow e Hoff introduziram o algoritmo da menor média quadrática (*least mean-square*) e utilizaram para o *Adaline*, um modelo de neurônio similar ao

perceptron. Nos anos 60 a idéia que se tinha das redes neurais é que poderiam executar qualquer tarefa. Minsky e Papert (1969) demonstraram matematicamente a limitação do perceptron em resolver certos problemas, o que desmotivou os estudos na área. Após poucos trabalhos publicados nos anos 70, vários modelos de redes surgem nos anos 80, sendo bastante importantes os trabalhos atribuídos a Grossberg (1980) com o *aprendizado competitivo*, Hopfield (1982) introduzindo a *função de energia* para as redes recorrentes; e Barto, Sutton e Anderson (1983) introduzindo o *aprendizado por reforço*. Em 1986 o desenvolvimento do algoritmo de retropropagação (*backpropagation*) por Rumelhart, Hinton e Williams emergiu como o mais popular método de treinamento para o perceptron multicamada. Tal algoritmo já tinha sido descrito anteriormente por Werbos em 1974, que descreveu um modo eficiente de computar o gradiente em modo reverso, sendo esta a base do algoritmo de retropropagação. Várias aplicações para o perceptron multicamada têm surgido desde então (HAYKIN, 1994, p.221), realizando tarefas em várias áreas como classificação de padrões, reconhecimento de fala, reconhecimento óptico de caracteres, entre outros. Outros tipos de treinamento da rede têm sido propostos. O método do gradiente conjugado, proposto por Fletcher e Reeves (1964) foi utilizado por Kramer (1989) para acelerar o treinamento do perceptron, bem como o método de Newton modificado por Saarinen (1991) e Battiti (1992).

Alguns trabalhos (MUNLEY, 1994; KERR; BARTLETT, 1995; RODRIGUES, 1995) propõem que a abordagem de redes neurais para a reconstrução de imagens de CT aumentaria a eficiência e qualidade de imagem, mostrando-se viável com redes do tipo *perceptron multicamada*, com algoritmos de treinamento *backpropagation*. Tal proposição está fundamentada no *teorema da aproximação universal* de Cybenko, onde funções contínuas podem ser aproximadas<sup>1</sup> por perceptrons multicamada (CYBENKO, 1988; HAYKIN, 1989). A reconstrução de imagens de CT tendo como base matemática a Transformada de Radon poderia ser incluída no escopo deste teorema. Rodrigues objetivou treinar o perceptron para executar a reconstrução. Apesar desta

metodologia ser viável para soluções locais, com um conjunto finito de imagens, pareceu pouco aplicável como solução generalizada (RODRIGUES, 2000).

Uma abordagem para busca de eficiência na reconstrução de imagens de CT é o uso de multiprocessamento ou processamento paralelo, fazendo com que o conjunto de instruções de um algoritmo seja distribuído por vários processadores de forma a executar a reconstrução tomográfica de forma rápida (RAO; KRIZ, 1995).

Alguns trabalhos acadêmicos (SILVA, 1998) referem-se à substituição de minicomputadores e sistemas antigos e obsoletos nos tomógrafos por microcomputadores pessoais (*personal computers*) nas fases de controle, aquisição de imagens e reconstrução, buscando aproveitar os subsistemas de fonte e detetores mecânicos, que ainda funcionam perfeitamente. Estes microcomputadores seriam mais eficientes na reconstrução de imagens de CT, com o objetivo também de prolongar a vida útil de equipamentos de gerações anteriores e adequado à realidade econômica de países menos favorecidos tecnologicamente.

Outras opções recentes no campo da teoria de reconstrução de CT é o uso em algoritmos de compressão de imagens (SANCHES, 2000), para o qual além dos métodos utilizados atualmente em compressão<sup>2</sup> a Transformada de Radon seria incorporada ao processo proporcionando maior compactação de imagens.

Para redes parcialmente conectadas um artigo foi publicado recentemente pelo autor (MEDEIROS *et al.*, 2001), sendo mostrado na íntegra no Anexo A.

---

<sup>1</sup> Desde que a função de ativação seja não constante, limitada e monotônica, como é o caso da função sigmoidal.

<sup>2</sup> Transformada, quantização, *delta pulse code modulation* (DPCM) e codificação aritmética.

### 3 FUNDAMENTOS DE TOMOGRAFIA

#### 3.1 A TRANSFORMADA DE RADON

Descrevemos a seguir uma fundação teórica sobre tomografia baseado em (JAIN, 1989, p.431). Para que se tenha uma projeção tomográfica de determinado objeto, é necessário fazer com que uma quantidade de radiação (como raios-X ou raios gama) atravesse este objeto e projete uma espécie de “sombra” em um anteparo devidamente colocado. Conforme as propriedades deste objeto (como massa ou densidade específica), o mesmo apresentará resistência à penetração da radiação de emissão  $I_0$ . O feixe será atenuado conforme a seguinte relação:

$$I = I_0 \exp \left[ - \int_L f(x, y) du \right] \quad (1)$$

Assim, o sinal observado será a seguinte relação logarítmica (figura 1)

$$p = \ln \left( \frac{I_0}{I} \right) \quad (2)$$

onde

$$p(s, \theta) = \int_L f(x, y) du, \quad -\infty < s < \infty, \quad 0 \leq \theta < \pi$$

Onde o par  $(s, \theta)$  representa as coordenadas do raio relativas ao objeto,  $s$  representando o deslocamento da projeção em relação ao eixo  $u$  e  $\theta$  o ângulo ao qual a projeção está sendo obtida. Deste modo,  $p(s, \theta)$  representa o valor assumido pela projeção nestas coordenadas.

De posse de várias projeções de um objeto, o problema da reconstrução tomográfica reside no fato de mapearmos  $p(s, \theta)$  para o espaço de imagem, de forma a obtermos novamente  $f(x, y)$ .

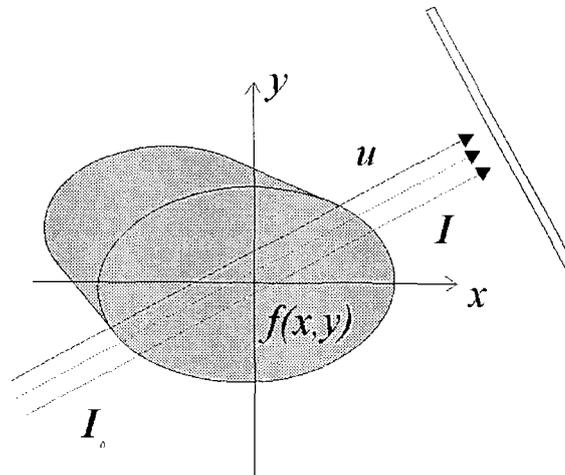


Figura 1 – Representação esquemática da atenuação da radiação atravessando um objeto.

A obtenção de  $p(s, \theta)$  é descrita matematicamente através da Transformada de Radon da função  $f(x, y)$ , sendo definida como a integral de linha ao longo de uma linha inclinada a um certo ângulo  $\theta$  do eixo  $y$  e à distância  $s$  da origem (ver figura 2). Na equação (3) temos a expressão da Transformada de Radon. A função delta multiplicando a função no integrando da equação especifica que apenas os pontos referentes à seção do objeto serão considerados na transformada.

$$p(s, \theta) = \mathfrak{R}f = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - s) dx dy \quad (3)$$

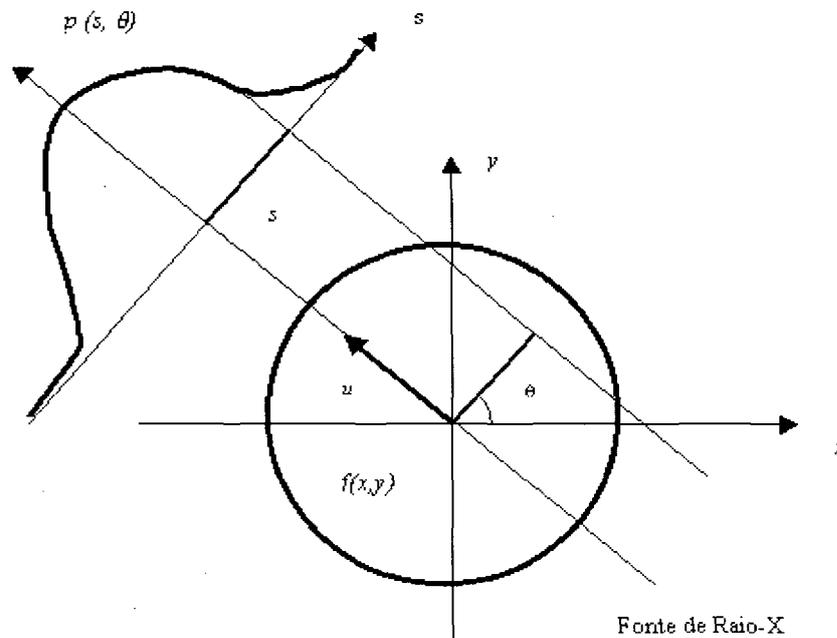


Figura 2 – Representação no plano cartesiano e plano de projeção.

A quantidade  $p(s, \theta)$  é descrita também como sendo o *raio-soma*, representando a soma dos  $f(x, y)$  ao longo do raio a uma distância  $s$  e ao ângulo  $\theta$ . Assim, a Transformada de Radon mapeia o domínio espacial para o domínio de projeções. Cada ponto em  $(s, \theta)$  irá corresponder a uma linha reta em  $f(x, y)$ .

Se tomarmos as coordenadas polares do domínio espacial como sendo  $(r, \phi)$ , de acordo com as seguintes definições

$$x = r \cos \phi, \quad y = r \sin \phi \quad (4)$$

concluimos que a distância em relação à origem  $s$  como sendo (figura 3)

$$s = r \cos(\theta - \phi) \quad (5)$$

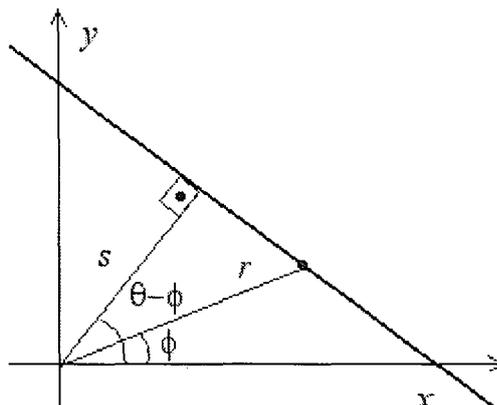


Figura 3 – Representação no plano cartesiano da relação entre as coordenadas das projeções e coordenadas polares da imagem.

Esta relação tem considerações fundamentais, visto que a relação entre o domínio de projeções e espacial é puramente geométrico, conceito que será amplamente explorado para o objetivo deste trabalho. Consideraremos também para fins práticos a suposição de que o feixe que atravessa o objeto a ser observado é paralelo<sup>3</sup>.

Para reconstruirmos a imagem  $f(x,y)$  necessitamos então fazer a *inversa* da Transformada de Radon. A definição contínua da inversa é expressa da seguinte forma:

$$f(x, y) = \left( \frac{1}{2\pi^2} \right) \int_0^\pi \int_{-\infty}^\infty \frac{[(\partial p / \partial s)(s, \theta)]}{x \cos \theta + y \sin \theta - s} ds d\theta \quad (6)$$

Ou, em coordenadas polares,

$$f(x, y) = \left( \frac{1}{2\pi^2} \right) \int_0^\pi \int_{-\infty}^\infty \frac{[(\partial p / \partial s)(s, \theta)]}{r \cos(\theta - \phi) - s} ds d\theta \quad (7)$$

<sup>3</sup> Existem sistemas que utilizam feixes divergentes (usualmente denominados *fan-beam*). Neste caso a Transformada de Radon sofre modificações (JAIN, 1989, p. 435).

Podemos identificar no integrando das equações acima que existe singularidade no cálculo da inversa. Não fosse esta singularidade o processo de cálculo da inversa seria único e bem definido. Outra limitação apresentada é que precisamos de infinitas projeções no intervalo  $[0, \pi)$  para reconstruirmos a imagem original. O que na prática acontece é a obtenção de um número finito de projeções suficiente para reconstruirmos uma estimativa da imagem.

### 3.2 O OPERADOR RETROPROJEÇÃO

Na Transformada de Radon vemos também o que é chamado de *retroprojeção*, e para entendermos o conceito da Transformada Inversa é necessário seu detalhamento com profundidade. Sua definição é a seguinte (JAIN, 1989, p.439; HERMAN, 1980, p.111)

$$b(x, y) = \int_0^{\pi} p(x \cos \theta + y \sin \theta, \theta) d\theta \quad (8)$$

O operador retroprojeção é definido no intervalo  $[0, \pi)$ . A quantidade  $b(x, y)$  é chamada de retroprojeção de  $p(s, \theta)$ . Este operador pode ser entendido como a acumulação dos raios-soma de todos aqueles raios que atravessam o ponto  $(x, y)$ . Portanto é feita a integração destes raios-soma, os quais estão sobre a senóide dada pela equação (5). Se expressarmos a equação (8) em coordenadas polares fica mais claro,

$$b(r, \phi) = \mathbf{B}[p] = \int_0^{\pi} p(r \cos(\theta - \phi), \theta) d\theta \quad (9)$$

Parece à primeira vista que o operador retroprojeção é naturalmente o processo inverso da Transformada de Radon, que detalha a imagem em projeções transversais, onde um mesmo ponto pode contribuir para várias projeções, e que pela retroprojeção busca-se resgatar a imagem original somando novamente estas contribuições para obtê-la. Porém, como será visto a seguir, somente o operador retroprojeção não restaura a imagem original, sendo necessário a filtragem das projeções para complementar o processo da inversa da Transformada de Radon.

Geralmente se divide uma avaliação numérica da integral das equações (8) e (9) em duas partes. Primeiro, uma aproximação da integral (ou soma de Riemann) expresso como (HERMAN, 1980, p.111)

$$b(r, \phi) = \sum_{\theta=0}^{N-1} p(r \cos(\theta - \phi), \theta) \quad (10)$$

Na equação (10), têm-se  $0 \leq \theta \leq N-1$ , sendo  $N$  o número de ângulos usados para obter a retroprojeção.

Na segunda parte, buscamos fazer uma estimativa, para cada ângulo  $\theta$ , do valor de  $p(s, \theta)$ ,  $s = r \cos(\theta - \phi)$  e  $s=0, \pm 1, \pm 2, \dots, \pm M$ , e  $2M$  sendo o número máximo de pontos da projeção (ver equação (5)). Por ser possível obter um valor fracionário para  $s$ , precisamos fazer uso de *interpolação*. Um exemplo de interpolação linear (HERMAN, 1980, p.112) é dado pela seguinte fórmula<sup>4</sup>:

$$[(s+1) - r \cos(\theta - \phi)]p(s, \theta) + [r \cos(\theta - \phi) - s]p(s+1, \theta) \quad (11)$$

onde se usa o valor fracionário obtido em  $r \cos(\theta - \phi)$ , calcula-se a sua diferença em relação aos valores inteiros abaixo ( $s$ ) e acima ( $s+1$ ) e pondera-se a contribuição de  $p(s, \theta)$  e  $p(s+1, \theta)$ .

---

<sup>4</sup> Na expressão original do autor é considerado um intervalo de amostragem que na fórmula acima é considerado unitário.

### 3.3 MÉTODOS DE RESOLUÇÃO

Ao encontro de soluções para a inversa da Transformada de Radon, foram desenvolvidos alguns métodos de estimação para obtenção da imagem original. Podemos enumerar (JAIN, 1989, p.445):

- Retroprojeção Filtrada
- Reconstrução por Transformada de Fourier
- Métodos Algébricos (ART)
- Transformada Inversa de Radon (propriamente dita)

O desenvolvimento de soluções para a Transformada Inversa de Radon segue uma linha deduzida a partir da definição da Transformada Inversa de Fourier (ver Apêndice C) para uma função bidimensional (JAIN, 1989, p.444).

$$f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u, v) e^{j2\pi(ux+vy)} dudv \quad (12)$$

Em coordenadas polares no plano de frequência  $w$ ,

$$f(x, y) = \int_0^{2\pi} \int_0^{\infty} F(w, \theta) e^{j2\pi w(x \cos \theta + y \sin \theta)} w dw d\theta \quad (13)$$

Como  $w$  está no intervalo  $[0, +\infty[$ , consideramos na fórmula  $|w|$  abrangendo a faixa  $]-\infty, +\infty[$  e limitamos também o intervalo de  $\theta$  em  $[0, \pi[$ . Conforme a teoria da fatia de Fourier, faz-se

$$\begin{aligned} f(x, y) &= \int_0^{\pi} \int_{-\infty}^{\infty} |w| F(w, \theta) e^{j2\pi w(x \cos \theta + y \sin \theta)} dw d\theta \\ &= \int_0^{\pi} \left\{ \int_{-\infty}^{\infty} |w| P(w, \theta) e^{j2\pi w(x \cos \theta + y \sin \theta)} dw \right\} d\theta \end{aligned} \quad (14)$$

o que permite chegar na fórmula da retroprojeção (ver equação (8)),

$$f(x, y) = \int_0^{\pi} q(s, \theta) d\theta = \int_0^{\pi} q(x \cos \theta + y \sin \theta, \theta) d\theta \quad (15)$$

onde

$$q(s, \theta) = \int_{-\infty}^{\infty} |w| P(w, \theta) e^{j2\pi ws} dw \quad (16)$$

Se escrevermos  $|w|P$  como  $wP\text{sgn}(w)$  e aplicando o teorema da convolução, obtêm-se

$$\begin{aligned} q(s, \theta) &= [\mathfrak{F}_1^{-1}\{wP(w, \theta)\}] * [\mathfrak{F}_1^{-1}\{\text{sgn}(w)\}] \\ &= \left[ \left( \frac{1}{j2\pi} \right) \frac{\partial p}{\partial s}(s, \theta) \right] * \left( \frac{-1}{j\pi s} \right) \end{aligned} \quad (17)$$

onde  $(1/j2\pi)[\partial p(s, \theta)/\partial s]$  e  $(-1/j\pi s)$  são as transformadas inversas de Fourier ( $\mathfrak{F}^{-1}$ ) de  $wP(w, \theta)$  e  $\text{sgn}(w)$ , respectivamente, e  $*$  denota a operação de convolução. Aplicando novamente o teorema da convolução, temos

$$q(s, \theta) = \left( \frac{1}{2\pi^2} \right) \int_{-\infty}^{\infty} \left[ \frac{\partial p(t, \theta)}{\partial t} \right] \frac{1}{s-t} dt \quad (18)$$

Substituindo a equação (18) na equação de retroprojeção obtida em (15), obtemos,

$$f(x, y) = \left( \frac{1}{2\pi^2} \right) \int_0^{\pi} \int_{-\infty}^{\infty} \left[ \frac{\partial p(t, \theta)}{\partial t} \right] \frac{1}{x \cos \theta + y \sin \theta - t} dt d\theta \quad (19)$$

a qual é a mesma equação expressa em (6), observando-se a troca da variável  $t$  por  $s$  necessária para expressarmos (18). Esta é a expressão, deduzida a partir da Transformada Inversa de Fourier, da Transformada Inversa de Radon.

### 3.3.1 Retroprojeção Filtrada (Eixo do Tempo ou Frequência)

Para o método de retroprojeção filtrada precisamos definir mais alguns operadores para podermos expressar a Transformada Inversa de Radon. O primeiro é o operador de derivada (JAIN, 1989, p.446; HERMAN, 1980, p.95), o qual descreve a diferenciação parcial com respeito à primeira variável

$$\mathbf{D}[p] = \frac{\partial p(s, \theta)}{\partial s} \quad (20)$$

Definimos também o operador da *Transformada de Hilbert*, também relativo à primeira variável (JAIN, 1989, p.445; HERMAN, 1980, p.96; ROSENFELD, 1982, p.375)

$$\mathbf{H}[\psi] = \psi(s) * \left( \frac{1}{\pi s} \right) = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{\psi(t)}{s-t} dt \quad (21)$$

Onde \* denota a operação de convolução. Deve-se notar que a integral acima é imprópria, sendo indefinida em  $s = t$ . Através desta definição podemos ver que na equação (18),  $q(s, \theta)$  é a Transformada de Hilbert de  $(1/2\pi)\partial p/\partial s$  para cada ângulo  $\theta$ .

Com operadores definidos em (9), (20) e (21), podemos escrever a Transformada Inversa de Radon da seguinte maneira

$$f(x, y) = \left( \frac{1}{2\pi} \right) \mathbf{B H D} p \quad (22)$$

Esta fórmula pode ser implementada fazendo-se a convolução (no eixo do tempo) das projeções diferenciadas com  $(1/2\pi s)$  e fazendo a retroprojeção do resultado.

Pode-se expressar a Transformada Inversa de Radon supondo as projeções diferenciadas da equação (18)

$$f(x, y) = \mathbf{B} \mathbf{H} q \quad (23)$$

Se considerarmos a Transformada de Hilbert  $\mathbf{H}$  como sendo um filtro unidimensional cuja resposta em frequência é  $|w|$ , da equação (18) podemos escrever

$$q = \mathbf{H} p = \int_{-\infty}^{\infty} |w| P(w, \theta) e^{j2\pi w s} ds = \mathfrak{S}_1^{-1} \{ |w| [\mathfrak{S}_1 p] \} \quad (24)$$

o que permite expressar

$$f(x, y) = \mathbf{B} \mathfrak{S}_1^{-1} [ |w| \mathfrak{S}_1 p ] \quad (25)$$

Esta equação pode ser implementada pela filtragem das projeções no domínio de frequência, seguida pela retroprojeção da Transformada Inversa de Fourier do resultado.

Ainda que na teoria, certos autores façam distinção entre os métodos de retroprojeção filtrada nos dois domínios (espaço e frequência), denominando "retroprojeção convoluída" para o domínio do espaço e "retroprojeção filtrada" para o domínio de frequência, esta distinção é bem visualizada quando utilizados para resolver a transformada inversa para funções contínuas (JAIN, 1989, p.446). Quando se fala de implementações práticas (digitais), os dois métodos podem ser explicados num único método de reconstrução, e a opção de filtragem pode ser considerada tanto no domínio do espaço quanto no domínio de frequência.

A Transformada Inversa de Radon é obtida então a partir de dois passos. Primeiro as projeções  $p(s, \theta)$  passam por um filtro unidimensional cuja resposta em frequência é igual a  $|w|$ . Obtemos o resultado  $q(s, \theta)$  que é retroprojetado para produzir  $f(x, y)$ . Como visto anteriormente, esta operação de filtragem pode ser feita tanto no domínio do espaço  $s$  como no domínio da frequência  $w$ . Uma técnica bastante praticada é a transformada para o domínio de frequência  $w$  através do algoritmo de FFT (*Fast Fourier Transform*, ou Transformada Rápida de Fourier), que tem menor complexidade em relação à DFT (*Discrete Fourier Transform*, ou Transformada Discreta de Fourier). O algoritmo implementado com DFT tem complexidade  $N^2$ , enquanto que o algoritmo FFT tem complexidade  $N \log_2 N$  (GONZALES; WOODS, 2000, p.84).

### 3.3.2 Reconstrução por Transformada de Fourier

Este método de reconstrução trata do preenchimento do espaço bidimensional de Fourier pelas transformadas unidimensionais das projeções, e após efetua-se a Transformada Inversa de Fourier. O que pode ser expresso pela fórmula (JAIN, 1989, p.462)

$$f(x, y) = \mathcal{S}_2^{-1}[\mathcal{S}_1 p] \quad (26)$$

O algoritmo deve ser composto por três estágios. Primeiro se obtém a Transformada de Fourier de  $p(s, \theta)$ , o que pode ser feito utilizando FFT. Logo a seguir, é preciso fazer a interpolação das coordenadas polares do espaço de Fourier para coordenadas retangulares. A seguir, faz-se a FFT inversa em duas dimensões. Este método pode originar efeitos de truncamento e *aliasing* (sobreposição), sendo aconselhável antes de proceder à FFT inversa multiplicar as projeções por uma janela apropriada (JAIN, 1989, p.462).

### 3.3.3 Métodos Algébricos (ART)

Através dos métodos algébricos busca-se resolver o processo de reconstrução da imagem final a partir de um sistema de equações e resolver por

processos iterativos este sistema onde as variáveis são os pontos de imagem. Cada raio-soma constitui-se em uma equação deste sistema. Se  $p_i$  for o raio-soma medido para o  $i$ -ésimo raio,  $f_j$  o ponto da imagem final e  $w_{ij}$  o fator de ponderação de cada ponto contribuindo para o raio-soma, o relacionamento entre estas variáveis pode ser feito da seguinte forma

$$\sum_{j=1}^N w_{ij} f_j = p_i, \quad i = 1, \dots, M \quad (27)$$

onde  $N$  é o número de pontos e  $M$  é o número total de raios em todas as projeções. Se  $M$  e  $N$  forem pequenos, são utilizados métodos de inversão de matrizes para o cálculo dos pontos  $f_j$ . Este método começa a ser bastante complexo com o aumento do tamanho da imagem, pois se temos imagens de 256 x 256, por exemplo,  $M$  e  $N$  serão 65536, e o número de equações será 65536 x 65536, tornando impraticável uma inversão de matrizes desta ordem. Métodos foram desenvolvidos por Kaczmarz e Tanabe (KAK; SLANEY, 1988, p.277) para resolver o sistema de equações, porém condicionados à existência de solução única para subgrupos de equações. Além disso é preciso lançar mão de outros artifícios algébricos para uma resolução mais eficaz. Porém, geralmente uma imagem reconstruída pelo método algébrico resulta em uma imagem ruidosa ("*salt and pepper noise*") devido às inconsistências introduzidas pelas aproximações no sistema de equações (KAK; SLANEY, 1988, p.284).

### 3.3.4 Transformada Inversa de Radon

A Transformada Inversa de Radon consiste no método matemático de resolução, obtido através da convolução no domínio do espaço das projeções com um filtro unidimensional. Após, executa-se a retroprojeção das projeções convoluídas. Ainda que para resolução de funções contínuas<sup>5</sup> possa ser uma

---

<sup>5</sup> Ou funções quadraticamente integráveis, conforme o conceito do espaço de Lebesgue (HERMAN, 1980, p.91).

método de resolução elencado com os métodos usuais, em implementações digitais não é utilizada devido à natureza discreta dos valores das projeções.

## 4 REDES NEURAIS

### 4.1 CONCEITOS BÁSICOS

Uma rede neural é um processador paralelo massivo distribuído que tem uma natural propensão para gravar conhecimento por experiência e disponibilizá-lo para utilização posterior (HAYKIN, 1994, p.2). Isto se resume em dois aspectos interessantes:

- 1) O conhecimento é adquirido pela rede através de um processo de aprendizado;
- 2) As conexões interneuronais conhecidas como pesos sinápticos são usadas para guardar o conhecimento.

Redes neurais artificiais têm aplicações das mais diversificadas. Alguns exemplo citados na literatura são (HAYKIN,1994,p.221):

- Detecção de padrões em sinais de radar / sonar
- Predição de séries temporais
- Reconhecimento de caracteres (OCR)
- Reconhecimento de fala
- Diagnóstico médico
- Aproximação de funções não-lineares
- Compressão de dados

Fazemos então um paralelo com a dinâmica biológica, para introduzirmos o funcionamento de uma rede a partir de suas unidades fundamentais. Sabemos que os *neurônios* são as unidades fundamentais dos tecidos do sistema nervoso, incluindo o cérebro. Cada neurônio consiste de um corpo celular, também designado como *soma*, o qual contém um *núcleo*. Partindo do corpo da célula existem um número de filamentos denominados *dendritos*, e um filamento mais longo que é denominado de *axônio*. Os dendritos ligam-se ao redor da célula a

outras células e o axônio faz uma conexão mais longa. A estas conexões dá-se o nome de *sinapses*. O sinal de uma célula a outra se faz mediante uma complicada reação eletroquímica. Substâncias químicas transmissoras são lançadas das sinapses e entram pelos dendritos, aumentando ou baixando o potencial elétrico da corpo da célula. Quando o potencial chega a um limiar, um pulso elétrico ou *potencial de ação* é mandado pelo axônio. O pulso espalha-se ao longo das conexões existentes pelo axônio, eventualmente chegando a outras sinapses e lançando transmissores ao corpo de outras células. Sinapses que incrementam o potencial de outras células são denominadas *excitatórias*, enquanto que as que decrementam são denominadas *inibitórias*. Os neurônios podem formar novas conexões com outros neurônios, e tais mecanismos são através do qual se forma a base para o aprendizado do cérebro (RUSSEL; NORVIG, 1995, p.564).

Uma rede neural artificial, tal como seu paralelo biológico, é composta de um certo número de nodos ou unidades (também denominados “neurônios”) conectados por ligações, conexões sinápticas ou *links*. Cada ligação possui uma quantidade associada como um peso. O conhecimento da rede é adquirido pela atualização destes pesos. Alguns dos neurônios são conectados ao ambiente externo, sendo designados como entradas (*inputs*) e saídas (*outputs*). (HAYKIN, 1994, p.13).

#### 4.2 O PERCEPTRON MULTICAMADAS

O perceptron é um dos modelos denominados clássicos de redes neurais. Proposto por Rosenblatt (1958), com uma única camada inicialmente foi utilizado para reconhecimento de caracteres. Um perceptron pode possuir várias camadas (*layers*) de neurônios além das camadas básicas de entrada e saída, sendo denominadas camadas escondidas (BISHOP, 1995, p.98). O perceptron como foi idealizado possui uma regra de aprendizado bastante simples, tendo apenas uma camada de pesos conectando as camadas de neurônios de entrada e saída. Porém, este modelo é inapto para classificar funções não linearmente

separáveis. Com mais camadas de pesos em um perceptron, é possível contornar esta limitação (AZEVEDO *et al*, 2000, p.73).

Num perceptron, os padrões são apresentados simultaneamente às camadas de entrada e saída. Os pesos conectando tais camadas são atualizados a partir de valores aleatórios. Os padrões são então apresentados à entrada, sendo calculadas as diferenças entre a saída obtida em relação à saída desejada. A diferença ou o erro total informa a próxima atualização de pesos, e assim acontece sucessivamente até que o erro seja zero ou o mínimo possível (BISHOP, 1995, p.100).

#### 4.3 A FUNÇÃO DE ATIVAÇÃO

Cada neurônio tem um conjunto de entradas ligadas de outros neurônios e outro conjunto de saídas conectadas a outros neurônios. O neurônio apresenta também um nível atual de ativação (figura 4). Conforme o nível atual das entradas do neurônio conjugadas com os pesos sinápticos das conexões, o valor resultante irá ativar ou não as saídas do neurônio. Esta conjugação pode ser representada como uma combinação linear das entradas ponderadas pelos pesos sinápticos, e o somatório destas irá ou não chegar ao nível de ativação. Temos assim,

$$u_k = \sum_{j=1}^p w_{kj} x_j \quad (28)$$

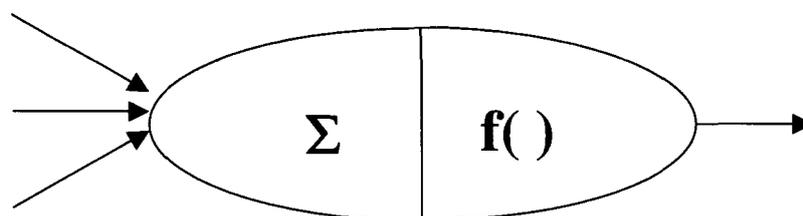


Figura 4 – Representação básica de um neurônio.

onde  $x$  é o valor das entradas para um neurônio qualquer,  $w$  são os pesos sinápticos e  $u$  é o resultado da soma das entradas ponderadas pelos pesos.

Um neurônio com duas entradas e uma saída teria como fórmula de representação:

$$u_1 = w_1x_1 + w_2x_2 \quad (29)$$

O nível de ativação (representado por uma função) determina o valor resultante para as saídas do neurônio. Isto determina o comportamento do neurônio dentro da rede, e este comportamento interfere no comportamento dos neurônios conectados à frente.

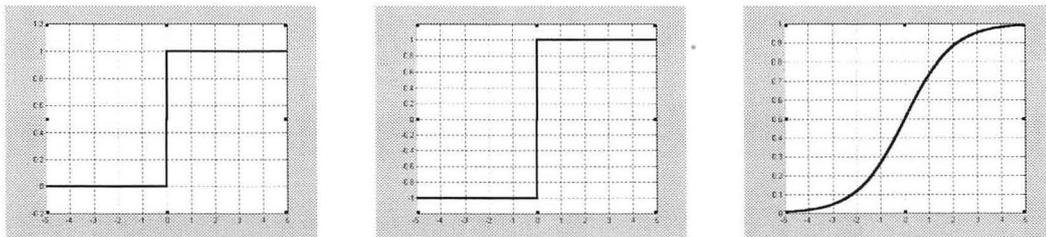


Figura 5 – Funções de ativação, em ordem: deslocamento, sinal e sigmóide.

As funções de ativação básicas (figura 5) são: função deslocamento (*step*), sinal (*signal*) e função sigmóide (*sigmoid*).

Assim,

$$y_k = \varphi(u_k) \quad (30)$$

onde  $\varphi$  é a função de ativação. Se como exemplo tomarmos a função sinal (caso do perceptron básico), sempre que o valor de  $u_k$  seja negativo,  $y_k$ , que é o neurônio de saída da rede, será  $-1$ , senão será  $+1$ .

$$\varphi(v) = \begin{cases} 1 & , v \geq 0 \\ -1 & , v < 0 \end{cases} \quad (31)$$

A função de ativação geralmente utilizada para perceptrons que devam trabalhar com funções não lineares é a sigmóide, expressa pela equação

$$\varphi(v) = \frac{1}{1 + \exp(-v)} \quad (32)$$

#### 4.4 REGRA DO DELTA GENERALIZADO

A regra de aprendizado mais conhecida foi introduzida por Paul Werbos conhecida como regra do delta generalizado (AZEVEDO *et al.*, 2000, p.83). Diferente do treinamento para um perceptron básico, esta regra tem como base a determinação do erro total do conjunto de treinamento em relação às saídas desejadas. A partir do gradiente do erro em relação aos pesos, tais pesos são atualizados, a uma determinada taxa. O processo de aprendizado da rede se resume então à busca de um ponto mínimo global na superfície do erro em relação aos pesos. Este processo também é denominado *descida de gradiente* (BISHOP, 1995, p.95). Quando este ponto é encontrado, a rede está treinada com os padrões de saída apresentando mínima diferença com a saída desejada.

#### 4.5 PERCEPTRON USANDO EXPURGO DE PESOS

Um perceptron pode ser orientado na fase de treinamento a direcionar a atualização dos pesos de forma a tornar zero os pesos próximos de zero. Após o treinamento, os pesos assumirão diversos valores. Aqueles próximos de zero praticamente não contribuirão em nada no somatório da equação (28). Uma técnica utilizada é a *eliminação de pesos* (HAYKIN, 1995, p.207-208). A função erro global  $E$  é modificada tal como segue

$$E = E_t + \lambda E_c \quad (33)$$

onde  $E_t$  é o erro global utilizado pela regra do delta generalizado e  $E_c$  é o termo adicionado denominado *penalidade*, o qual interfere na computação do erro global  $E$  mediante o termo  $\lambda$ , denominado *parâmetro de regularização*. O termo penalidade é expresso da seguinte forma

$$E_c = \sum_{i \in C} \frac{(w_i / w_0)^2}{1 + (w_i / w_0)^2} \quad (34)$$

na qual  $w_0$  é um parâmetro livre de ajuste do decaimento e  $w_i$  um peso qualquer dentro do conjunto total de pesos  $C$  do perceptron modificado.

#### 4.6 IMPLEMENTAÇÃO

Neste trabalho, o desempenho do algoritmo de redes parcialmente conectadas foi comparado também com um algoritmo utilizando um perceptron de duas camadas totalmente conectado, no qual foi executado o algoritmo de retropropagação baseado na regra do delta generalizado, porém com o uso da técnica de expurgo de pesos. Ao final do treinamento o perceptron está apto a reconstruir a imagem original com alta qualidade a partir das projeções na entrada do perceptron. Ressalta-se que a rede foi treinada apenas com uma imagem e seu conjunto de projeções, estes sendo aplicados à entrada e a imagem original à saída. Com a finalidade de posterior comparação foi feito uso da imagem de teste (definida no capítulo 6) com 28 pontos e 10 projeções (produzindo imagens de 28 x 28 pixels). Ver figura 6.

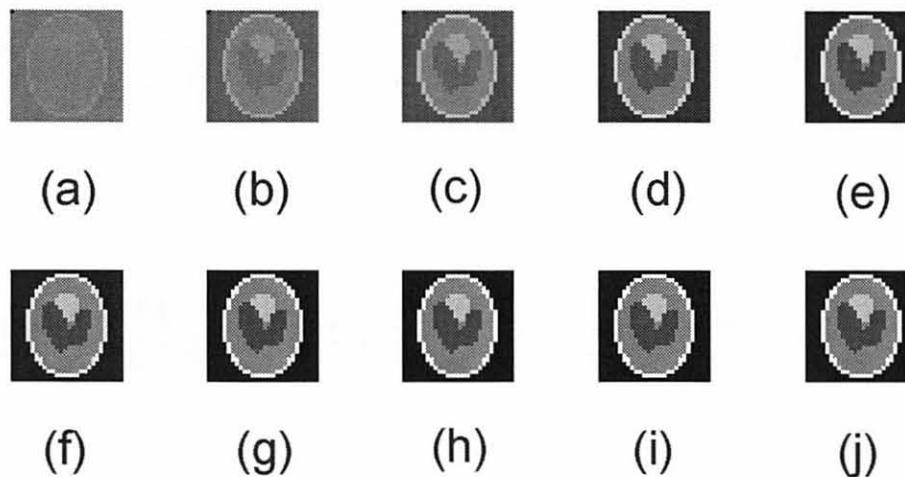


Figura 6 - Processo de treinamento de uma rede perceptron multicamada mostrando a evolução da imagem final com o aumento do número de épocas de treinamento: (a) 5; (b) 10; (c) 20; (d) 50; (e) 100; (f) 200; (g) 500; (h) 1000; (i) 2000; e (j) 5000.

## 5 REDE PARCIALMENTE CONECTADA PARA RECONSTRUÇÃO DE CT

Neste capítulo é explicado o ponto central deste trabalho, a construção da rede parcialmente conectada para reconstrução de CT. Com base nos conceitos vistos anteriormente, buscaremos avançar no algoritmo de reconstrução de imagens tomográficas abordando o conceito de redes neurais. Baseando-se no algoritmo de retroprojeção filtrada, divide-se o processo em duas redes:

- Retroprojeção
- Filtragem.

Em determinados momentos desta explanação sobre redes parcialmente conectadas, são utilizados conceitos de programação para descrever os algoritmos envolvendo a linguagem Pascal.

### 5.1 REDE DE RETROPROJEÇÃO (BACKPROJECTION)

Na fase da retroprojeção toma-se as projeções da imagem de CT e opera-se a transformação para a imagem final, expressa pela equação (15) na forma contínua e para a forma discreta na equação (35).

$$\begin{aligned}
 f(x, y) &= \frac{\pi}{N} \sum_{i=1}^N q(s, \theta_i) = \frac{\pi}{N} \sum_{i=1}^N q(x \cos \theta_i + y \sin \theta_i, \theta_i) = \\
 &= \frac{\pi}{N} \sum_{i=1}^N q(r \cos(\theta_i - \phi), \theta_i)
 \end{aligned}
 \tag{35}$$

Na análise desta equação verifica-se que as projeções filtradas  $q(s, \theta_i)$  são referenciadas pelo valor paramétrico dado pelo pixel da imagem  $f(x, y)$  (ou em coordenadas polares  $f(r, \phi)$ ) em relação aos ângulos  $\theta_i$ , e assim somados para todos os ângulos das projeções. Ou seja, os pontos de cada projeção filtrada  $q(s, \theta_i)$  segundo a relação contribuem para o valor do pixel no ponto  $(x, y)$ . Ver figura 7, onde um paralelo com o conceito de rede neural é feito.

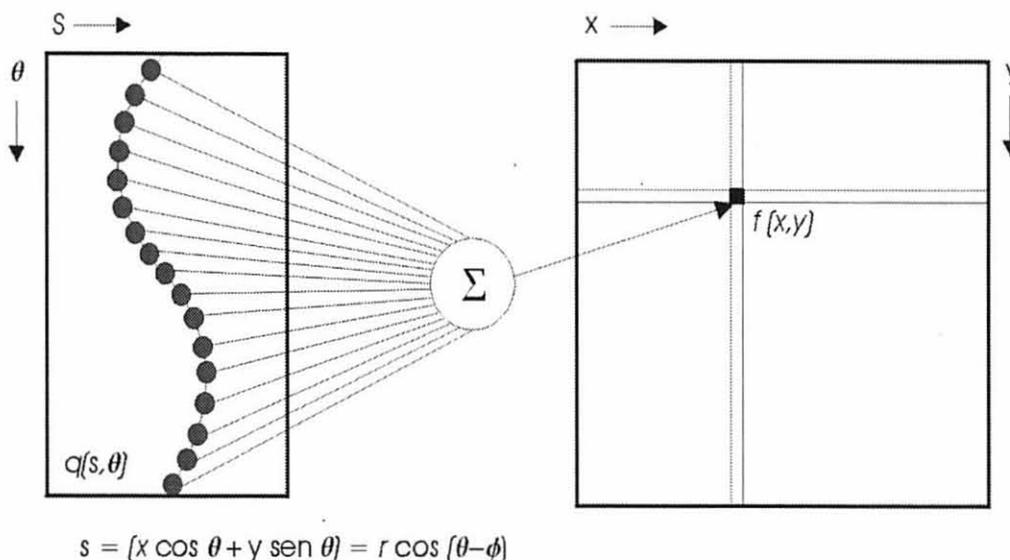


Figura 7 - Visualização gráfica da equação (35) em forma de rede neural, mostrando os valores  $q(s, \theta)$  referenciados pelos "pesos" dados por  $(x \cos \theta + y \text{sen } \theta)$  e somados entre si resultando no valor do ponto  $f(x, y)$ .

Se ao invés de considerarmos coordenadas cartesianas para a imagem final, utilizarmos coordenadas polares  $(r, \phi)$ , e considerando que no espaço de projeções cada ponto pode ser representado pelo par  $(s, \theta)$ , sendo  $s$  a distância do ponto em relação à origem para cada ângulo de projeção  $\theta$ , a relação existente entre o espaço de projeções e o espaço de imagem pode ser representada pela equação (5),  $s = r \cos(\theta - \phi)$ . Na figura 8 pode-se ver as senóides expressas por esta equação sobre as projeções.

Através desta equação vê-se que existem valores  $q(s, \theta)$ , para todas as projeções, que contribuem para um determinado ponto de imagem  $(r, \phi)$ , e que

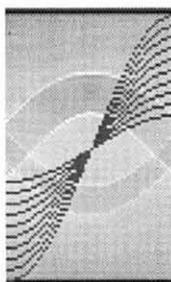


Figura 8 – Senóides sobre o espaço de projeções filtradas.

estão localizados ao longo de uma senóide (figura 9) sobre o espaço de projeções. Apenas estes valores  $q(s, \theta)$  irão gerar conexões na rede.

Uma rede neural com uma camada totalmente conectada é descrita pela combinação linear expressa da seguinte forma:

$$y_j = \sum_{i=0}^{M-1} w_{ij} x_i \quad (36)$$

Onde ligando cada entrada  $x_i$ , à saída  $y_j$  temos um peso equivalente  $w_{ij}$ . Se o índice  $j$  estiver no intervalo  $[0, N-1]$ , esta rede terá  $M \times N$  conexões, e tais conexões transferem deste modo os valores de entrada para a saída.

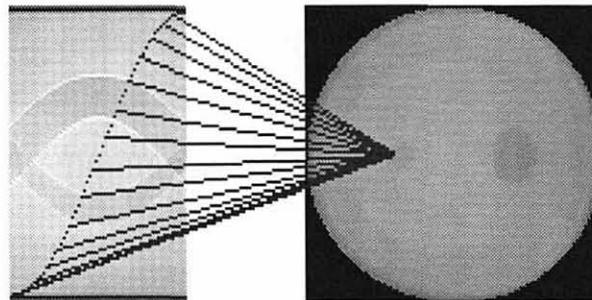


Figura 9 – Contribuição dos pontos  $(s, \theta)$  sobre a senóide para o ponto  $(r, \phi)$  da imagem reconstruída.

A equação acima expressa que a rede é totalmente conectada. Porém vê-se que não há necessidade de conexão total porque os pontos sobre o espaço de projeções que interessam para o ponto  $(x, y)$  da imagem final estão sobre a senóide dada pela equação (5). Deste modo uma rede parcialmente conectada pode executar a tarefa de retroprojeção (figura 10).

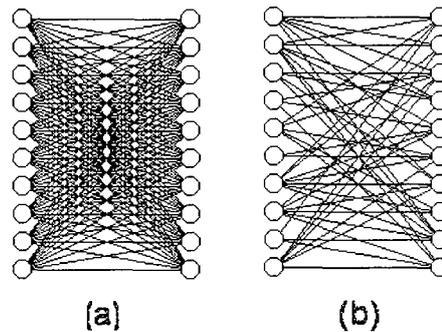


Figura 10 - Em (a) vemos um exemplo de rede totalmente conectada. Em (b) a rede parcialmente conectada, que pode ser vista como uma rede totalmente conectada com alguns pesos sendo zero.

### 5.1.1 Aspectos Computacionais

Se tomarmos como exemplo uma imagem de tamanho  $100 \times 100$ , reconstruída a partir de 60 projeções, necessita-se compor uma rede da seguinte maneira:

- 1) A primeira camada será composta de neurônios de entrada representando os pontos  $(s, \theta)$ . Este neurônios conterão o valor  $q(s, \theta)$ . Têm-se então 60 projeções de 100 pontos, totalizando 6000 neurônios
- 2) A segunda camada será composta de neurônios de saída representando os pixels da imagem  $(x, y)$ , contendo o valor  $f(x, y)$ . Têm-se assim uma imagem  $100 \times 100$ , totalizando 10000 neurônios.
- 3) Conectando cada neurônio de entrada com o de saída, temos as sinapses ou pesos. Estas sinapses a princípio terão valor unitário, pois irão transferir os valores contidos nos neurônios de entrada para os neurônios de saída.

Se conectássemos totalmente, teríamos 60 milhões de neurônios. Caso tenhamos  $M$  projeções e  $N$  pontos para produzir uma imagem  $N \times N$ , o algoritmo passa a ter complexidade  $O(MN^3)$ . O número de conexões levando-se em conta a conexão parcial pela equação (5) obtido experimentalmente foi de apenas 451260. Ou seja, a conexão total seria aproximadamente 133 vezes maior que a parcial.

Pode-se pensar na utilização de um perceptron com duas camadas totalmente conectado, e utilizar uma abordagem de expurgo de pesos (ver capítulo 4) onde, a cada época de treinamento, o algoritmo de retropropagação direciona gradativamente para zero os pesos sem significância (HAYKIN,1994,p.207). Com um alto nível de redundância, a rede após o treinamento pode vir a assumir um número de conexões sinápticas bem reduzido ao final. Porém, é necessário começar o processo com a rede totalmente conectada. Se fosse utilizada uma estrutura de dados para estas conexões (detalhado posteriormente) com tamanho de 16 bytes, seria necessário um hardware com memória RAM próximo a 960 MB (megabytes) para processar uma rede para imagem de tamanho 100x100. Para uma rede para imagens de tamanho 256x256, devido à complexidade  $O(MN^3)$ , a necessidade explodiria para 68 GB (gigabytes) de memória RAM. O uso de redes neurais para reconstrução de imagens totalmente conectadas, como o perceptron, demanda uma quantidade enorme de pesos sinápticos. Aliado a isto, o processo de treinamento de uma rede para chegar a uma aproximação da imagem final com tamanhos cada vez maiores, pode não convergir.

Este é o motivo central que pesa na escolha da rede com conexão parcial. É preciso então definir para a implementação de software, a estrutura dos dados que serão manipulados, de forma a representar a abstração da rede tanto dos neurônios quanto das sinapses. Duas abordagens aqui utilizadas foram a lista ligada e o vetor de estruturas.

### 5.1.2 Estruturas de Dados - Vetores de Estruturas

Uma alternativa para trabalhar com uma rede é considerar a estrutura de pesos como sendo um *vetor de estruturas*. Um vetor de estruturas não necessita de ponteiros, pois todos os nós estão contíguos na memória (figura 11). Assim, bastaria incrementar o índice do vetor para se passar ao próximo peso ou conexão (ZIVIANI, 1993, p.37). Outra vantagem é que as projeções filtradas podem estar



A criação da estrutura poderia ser da seguinte forma:

```
BP: array[0..M*N*N-1] of Estrutura;
```

Onde  $N$  é o tamanho da imagem e  $M$  é o número de projeções.

### 5.1.3 Estruturas Sequenciais de Dados – Listas Ligadas

Outra alternativa de construção do algoritmo está baseada no conceito de *listas ligadas*. Listas ligadas (TENENBAUM *et al.*, 1995, p.224) são estruturas dinâmicas de dados nas quais tem-se elementos denominados *nós*, possuindo um campo para armazenar *informação* e outro campo ou campos para armazenar *endereçamento*. Este endereçamento refere-se a outros nós seguintes ou anteriores que contenham informação.

A estrutura do nó da lista que seria o neurônio de entrada da rede teria a seguinte definição, utilizando a expressão da linguagem Pascal (figura 12):

```
NI = record
  ID : integer;
  valor : real;
  s, theta : integer;
  P, N: pointer;
end;
```

Utilizamos então a cláusula *record*, que nos permite formar um novo tipo ou estrutura a partir de tipos de variáveis básicos. Na estrutura *NI*, *ID* é a identificação do neurônio. Com  $s$  variando no intervalo  $[0, N-1]$  e  $\theta$  em  $[0, M-1]$ , optou-se por identificar o neurônio no intervalo  $[0, MN-1]$ . Se  $N=100$  e  $M=60$  por exemplo, *ID* assume no mínimo 0 e no máximo 5999.

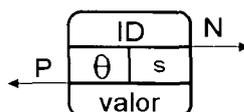


Figura 12 – Representação da estrutura do nó(neurônio) da camada de de entrada.

A variável *valor* irá assumir o valor de  $q(s, \theta)$ . As variáveis  $P$  e  $N$  serão o que chamamos de ponteiros. Estes ponteiros situam o neurônio, sendo que  $P$

aponta para o neurônio anterior e  $N$  aponta para o neurônio posterior. Desta forma tem-se condições de definir um algoritmo que faça o percurso desde o primeiro neurônio até o último ou o inverso. Precisa-se deste algoritmo para armazenar os valores de  $p(s, \theta)$  na rede. Além disso, é necessário outro algoritmo para efetuar a busca de um neurônio específico, também baseado nos ponteiros  $P$  e  $N$ .

Da mesma forma, a estrutura do nó que representaria o neurônio de saída da rede seria (figura 13)

```
NO = record
  ID : integer;
  valor : real;
  x, y : integer;
  P, N: pointer;
end;
```

Na estrutura  $NO$ ,  $ID$  é a identificação do neurônio de saída. Com  $x$  e  $y$  variando no intervalo  $[0, N-1]$ , optou-se por identificar o neurônio no intervalo  $[0, N^2-1]$ . Para o exemplo anterior, onde  $N=100$ ,  $ID$  assume no mínimo 0 e no máximo 9999. A variável  $valor$  irá assumir o valor de  $f(x,y)$ .

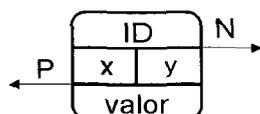


Figura 13 – Representação da estrutura do nó (neurônio) para a camada de saída.

Teríamos um conjunto de nós representando a entrada da rede como pixels da imagem formada pelas projeções, e um conjunto de saída representando os dados retrojetados também considerados como pixels de imagem. A informação dos nós seria o valor de  $q(n, \theta)$  do conjunto de entrada, e a saída seria o valor de  $f(x,y)$  referente ao conjunto de saída. Os nós dentro dos conjuntos seriam interconectados (figura 14), sendo guardados em cada nó o endereço ou posição do nó anterior e do nó posterior do intervalo discreto  $n=[0, N-1]$ , para cada projeção discreta  $\theta=[0, 2\pi]$ .

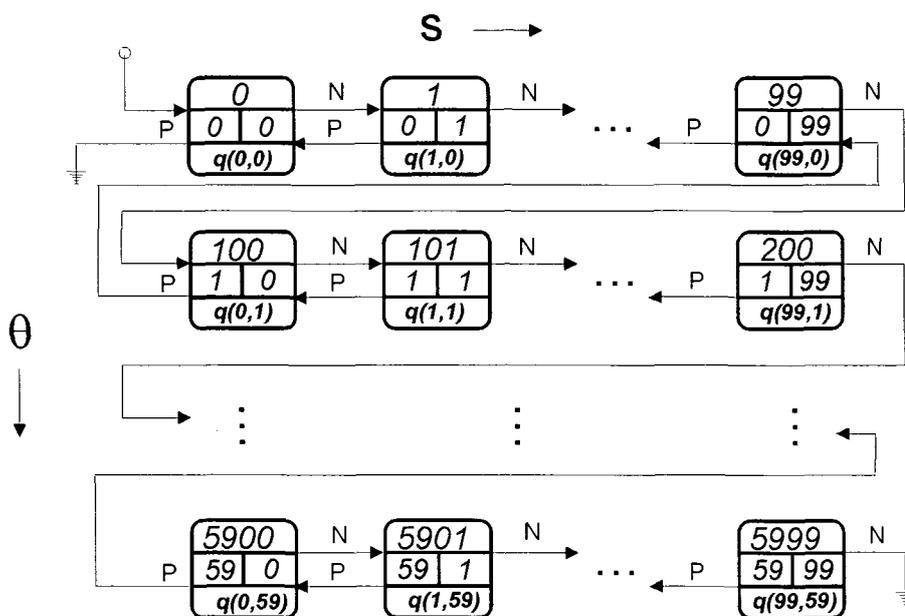


Figura 14 – Representação da estrutura da camada de entrada interligada para o exemplo de uma rede de 60 projeções de 100 pontos (o valor representando  $\theta$  na verdade é o índice de projeções, devendo ser multiplicado por  $\pi/60$  para se obter o resultado em radianos).

A conexão entre os nós se faz de maneira ligar os ponteiros P e N aos nós anterior e posterior, respectivamente. O primeiro nó tem o ponteiro para o início da rede, sendo que o ponteiro anterior do primeiro nó e o ponteiro posterior do último nó são nulos.

Esta conexão em seqüência permite desenvolver algoritmos de percurso para inserirmos os valores na rede. Um ponteiro temporário começa tendo como valor o ponteiro da rede, e assim que armazena o conteúdo em *valor*, assume o endereço do próximo nó. Quando chegasse ao último nó, a rede conteria os valores das projeções. Um exemplo de algoritmo para este processo de armazenamento de valores seria (o exemplo ilustra a forma utilizando uma matriz para as projeções, porém na prática se faria a leitura dos dados diretamente do arquivo):

```

var
  Ponteiro: ^NI; { estrutura do nó temporário }
  Rede: ^NI; { ponteiro de entrada da rede }
  s,theta: integer;
{ Presume-se que Rede já aponte para o início da rede
  já construída e os valores das projeções em uma matriz
  q[0..99,0..59] }
Ponteiro := Rede;
for theta := 0 to 59 do
begin
  for s := 0 to 99 do
  begin
    Ponteiro.Valor := q[s,theta];
    Ponteiro := Ponteiro.N; { vai para o próximo nó }
  end;
end;
end;

```

Porém apenas um algoritmo de visitação em seqüência para a rede seria insuficiente. É preciso conectar os pesos para um nó final  $(x,y)$  aos nós calculados  $(s,\theta)$ . Para se efetuar uma busca mais eficiente dos nós  $(s,\theta)$  uma alternativa é acrescentar à rede de entrada uma estrutura de árvore binária permitindo o acesso aos nós  $(s,\theta)$  através de um algoritmo de pesquisa binária (TENENBAUM *et al.*,1995, p.322; ZIVIANI, 1996, p.133) Assim, não seria preciso visitar todos os nós anteriores ao nó desejado  $(s,\theta)$  no momento da construção da rede.

A rede de saída representando a imagem seria também conectada em seqüência (figura 15). Isto se faz útil em três algoritmos: construção da rede, preenchimento de zeros em *valor*, e recuperação da imagem calculada.

As conexões entre os conjuntos de entrada e de saída (ou pesos) também estariam estruturadas como listas ligadas. Assim, a partir do nó (representando o peso) inicial visitaríamos os outros nós por meio dos endereços contidos na estrutura do nó, até o último nó apontando para endereço nulo. Nesta visita de cada nó a informação seria transferida da estrutura de nós de entrada para os respectivos nós de saída. Como vimos anteriormente que não há necessidade de total conexão entre os nós de saída e entrada, os nós-pesos visitados seriam somente aqueles que iriam contribuir para o valor final, diminuindo o tempo de execução necessário.

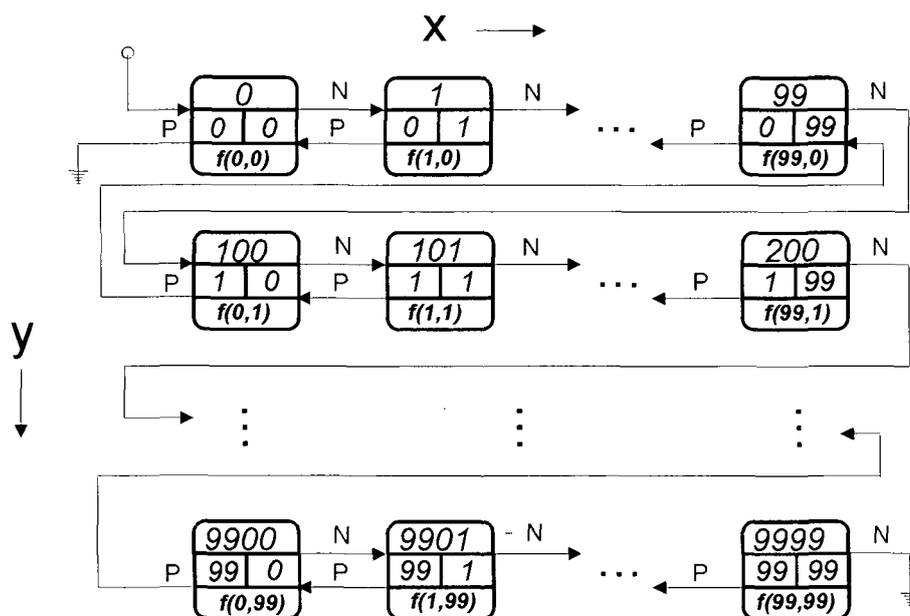


Figura 15 – Representação da estrutura da camada de saída interligada para o exemplo de uma rede de 60 projeções de 100 pontos. A imagem final produzida teria dimensões 100x100.

A princípio a estrutura do nó representando o peso ou sinapse seria simples, apenas contendo a sua identificação e os ponteiros para o próximo peso na lista, o neurônio na camada de entrada e o neurônio na camada de saída (figura 16). Para uma rede com interpolação, é necessário prever na estrutura uma variável para armazenar o valor do peso. Assim, uma declaração para esta estrutura é

```

W = record
  ID : integer;
  N : pointer;
  N1 : pointer;
  N2 : pointer;
end;
```

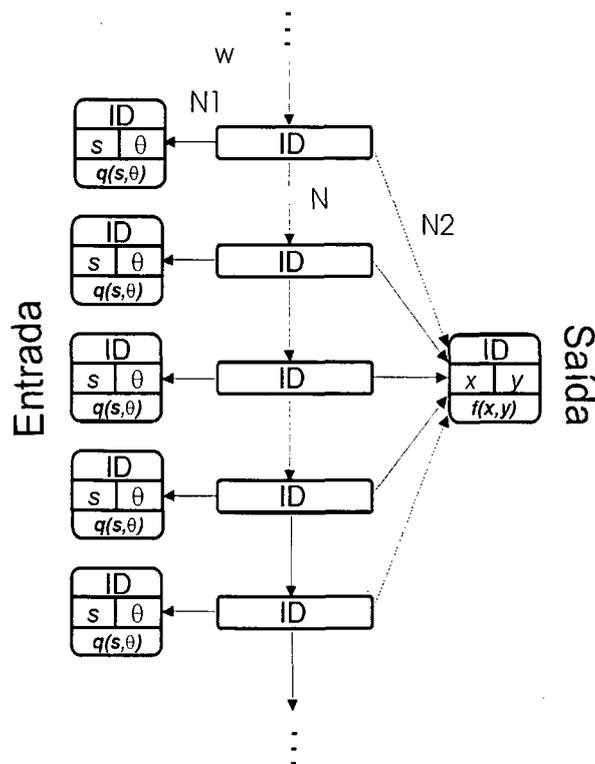


Figura 16 – Representação da conexão dos pesos entre si e com os nós das camadas de entrada e saída, segundo a relação da equação (5).

Temos então a variável  $ID$  que identifica o nó ou peso, o ponteiro  $N$  que indica o próximo peso na lista, o ponteiro  $N1$  que indica o nó ou neurônio da camada de entrada e  $N2$  que aponta para o nó ou neurônio da camada de saída.

Uma das desvantagens do uso de listas ligadas é a necessidade de embutir na estrutura das sinapses o ponteiro para o nó posterior, o ponteiro para o nó do ponto da projeção e para o nó do pixel de imagem. Como no hardware utilizado têm-se endereçamento de 32 bits (4 bytes), seriam utilizados 12 bytes por conexão, o que aumentaria substancialmente a necessidade de memória.

O uso de vetor de estruturas é vantajoso em relação a listas ligadas em virtude da economia de memória e necessidade de menos estruturas de dados para a construção da rede.

#### 5.1.4 Fases da Rede de Retroprojeção

Pode-se dividir, a partir do que foi exposto, os processos da rede de retroprojeção em duas fases: criação ou montagem e processamento. Em resumo, para a criação da rede de retroprojeção são necessários os seguintes passos:

- A criação das camadas de entrada e saída a partir das dimensões especificadas. Os neurônios das camadas são criados e conectados como uma lista encadeada dupla. Para o início do percurso temos os ponteiros que contém o endereço inicial das redes. Cria-se a estrutura de árvore binária para a camada de entrada.
- A criação da estrutura de pesos, sendo uma lista encadeada simples, onde cada peso une o neurônio da camada de saída ao neurônio de entrada regido pela equação (5). Os neurônios na camada de entrada são encontrados utilizando a pesquisa na árvore binária.

Para o processamento da rede de retroprojeção, temos então:

- Armazenamento de zeros nos neurônios das camadas de saída.
- Alimentação dos valores das projeções aos neurônios da camada de entrada de forma seqüencial.
- Percurso na lista de pesos do primeiro peso ao último, acumulando o valor do neurônio da camada de entrada no neurônio da camada de saída.

#### 5.1.5 Vantagens e Desvantagens

Talvez o aspecto mais importante a considerar e que emana da equação (5) é que a relação entre os espaços de projeções e de imagem é puramente geométrico, não dependendo do valor da projeção no ponto específico ( $s$ ,  $\theta$ ). Assim sendo, uma mesma rede de retroprojeção pode reconstruir *qualquer*

*imagem* que seja apresentada à entrada da rede, para tamanho da imagem e número de projeções invariantes.

Comparando-se com a rede totalmente conectada, onde vimos que a complexidade é expressa por  $O(MN^3)$ , na rede parcialmente conectada a coordenada  $s$  torna-se uma variável *dependente* ao invés de ser livre e, sendo obtida a partir das coordenadas  $(x,y)$  da imagem final<sup>6</sup> e do ângulo  $\theta$ , e  $0 \leq s < N$ , reduz-se a complexidade  $N$  vezes, ficando então  $O(MN^2)$ . Traduzindo em codificação de linguagem Pascal a rede totalmente conectada conforme a equação (36),

```

...
for x:= 0 to N-1 do
  for y := 0 to N-1 do
    for theta := 0 to M-1 do
      for s := 0 to N-1 do
        f[x,y] := f[x,y] + w[x,y,s,theta]*q[s,theta];
      ...

```

onde os  $w[x,y,s,theta]$  representam os valores dos pesos (obtidos após treinamento) a ser multiplicados pelos valores das projeções  $q[s,theta]$ , e acumulados para resultar  $f[x,y]$ . Para a rede parcialmente conectada<sup>7</sup>, considerando-se um vetor para a estrutura de pesos (aproveitando as definições anteriores) e de conformidade com a equação (35),

```

...
Estrutura = record
  s,theta: byte;
  x,y: byte;
end;
BP: array[0..M*N*N-1] of Estrutura;
...
{ Construção Prévia do Vetor }
k := 0;
for x:= 0 to N-1 do
  for y := 0 to N-1 do
    for theta := 0 to M-1 do
      begin
        BP[k].x := x;
        BP[k].y := y;
        BP[k].theta := theta;
        BP.s[k] := x*cos(theta) + y*sin(theta);
        Inc(k);
      end;

```

<sup>6</sup> Ou em coordenadas polares  $(r,\phi)$ . Ver equação (4).

<sup>7</sup> Considerando-se sem interpolação e supondo os valores de  $q$  normalizados.

```

...
{ Realização / Obtenção da Imagem Final }
for k:= 0 to M*N*N-1 do
    f[BP[k].x, BP[k].y] := f[BP[k].x, BP[k].y] +
q[BP[k].s, BP[k].theta];
...

```

onde  $BP$  guardará para cada ponto da imagem final as coordenadas das projeções de entrada. O algoritmo é dividido em duas fases distintas: *construção* e *realização* (ou *execução*). A construção é executada uma única vez, e várias realizações podem ser efetuadas para diferentes conjuntos de projeções apresentados<sup>8</sup>. Na realização, o vetor  $BP$  é varrido através do índice  $k$  indo de zero a  $M*N*N-1$ . Ainda que seja preciso por meio desta implementação, duas fases, a complexidade permanece  $O(MN^2)$ .

Caso se utilize coordenadas polares  $(r, \phi)$  para a imagem final, conforme a equação (5), numa imagem de tamanho  $N \times N$  os pixels que estivessem posicionados de forma que  $r \geq N$  não seriam mostrados. Em tais valores de  $r$  não se teria pesos conectando o espaço de projeções com o espaço de imagem. Levando em conta que a imagem final limita-se numa circunferência de raio  $N/2$ , considera-se a totalidade dos pontos internos como a área da circunferência (figura 17). Assim a expressão  $\pi N^2/4$  reflete a quantidade de pontos representativos da imagem  $f(x, y)$ .

---

<sup>8</sup> Desde que  $M$  e  $N$  não sejam modificados e que a matriz da imagem final  $f$  seja preenchida com zeros antes de cada realização.

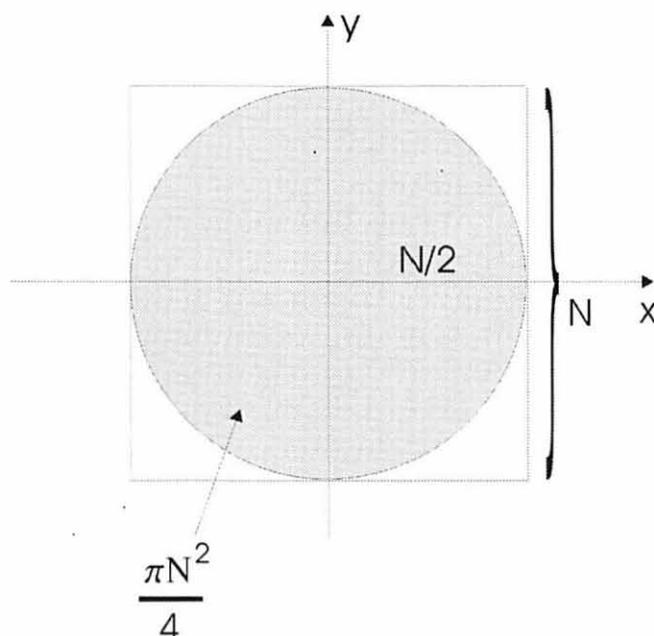


Figura 17 - Eixo de coordenadas mostrando a região efetiva de imagem após a reconstrução, e o número de pesos necessários sendo aproximadamente igual à área da circunferência de raio  $N/2$  vezes o número de pontos das projeções.

Como a cada ponto  $(x,y)$  se conecta uma senóide sobre o espaço das projeções filtradas e têm-se  $M$  projeções, a quantidade de conexões é aproximadamente  $\pi MN^2/4$ . Assim, na realização do algoritmo o cálculo para a área externa da circunferência é economizado, reduzindo-se o processamento a um fator de  $\pi/4$ .

Resumindo, a economia representada pelo uso da rede parcialmente conectada para a retroprojeção poderia ser descrita em 3 (três) pontos fortes:

- 1) Cálculos trigonométricos realizados durante a construção e na realização são usadas apenas somas, evitando chamadas a funções de seno e cosseno
- 2) Valores de imagem fora do espaço ( $r \geq N$ ) desconsiderados para procedimento de cálculo da retroprojeção, *a priori*.

- 3) Após a primeira criação da rede de retroprojeção em memória, não seria necessário reconstruir uma segunda vez, sendo assim útil para seqüências de imagens ou projeções 3D de mesmo tamanho.

Para uma lista ligada, os nós de entrada e saída seriam então estruturas alocadas em memória dinâmica, ligados por ponteiros de memória. Se ao invés de estruturas dinâmicas fossem utilizados vetores bidimensionais (percorrendo os elementos do vetor via *loops for-next* ou semelhantes) é obrigatório visitar todos os índices do intervalo considerado e deve-se testar se o valor é zero ou não. Em contrapartida, a lista ligada dinamicamente já teria em si a construção de conexões entre os nodos de entrada e saída relevantes, e onde os  $w_{ij}$  assumissem 0 (zero) não haveria percurso na rede.

A desvantagem mais restritiva para o uso da rede de retroprojeção é a exigência de memória para processamento. Para  $N$  pontos em  $M$  projeções temos requerimento de memória na ordem  $O(MN^2)$ . Caso o número de pontos nas projeções fosse duplicado, para  $2N$  pontos teríamos  $4MN^2$ , ou seja, um aumento de quatro vezes no número necessário de conexões.

### 5.1.6 Interpolação

Num segundo momento, poderia se fazer uso da interpolação para obter uma imagem final com maior resolução. No cálculo da equação (5), o valor de  $s$  pode não resultar em uma posição de valor inteiro (figura 18). Assim, para um mesmo ponto  $(x,y)$  podemos ter contribuição de dois pontos no espaço de projeções  $(s, \theta)$  ao invés de um ponto apenas. Estas contribuições precisariam ser ponderadas para resultar no valor final. Uma fórmula para interpolação das projeções foi apresentada anteriormente na equação (11).

Se optarmos por interpolação, podemos utilizar os pesos  $w_{ij}$  (que anteriormente eram unitários) para guardar o valor de ponderação específico para cada ponto em  $(s, \theta)$ . Em contrapartida dobraríamos o número de pesos necessários, além de precisarmos multiplicar os valor do peso pelo valor do

neurônio da camada de entrada antes de acumular no neurônio de saída. A declaração para a estrutura do peso ficaria

```
W = record
  ID : integer;
  valor: real;
  N: pointer;
  N1 : pointer;
  N2 : pointer;
end;
```

onde *valor* armazenaria o fator de ponderação da projeção do peso indicado.

Como exemplo, se no cálculo para o ponto da projeção encontrássemos o valor  $s = 5.3$ , a contribuição para o pixel de saída seria a partir dos dois neurônios com posições  $s_1 = 5$  e  $s_2 = 6$  (figura 19). O valor total de contribuição, supondo a interpolação linear da equação (11), teria uma maior representação para a posição  $s_1 = 5$  por estar mais próximo. Sem interpolação, seria este o valor escolhido devido ao truncamento.

O resultado seria a soma  $q(s, \theta) = (1 - f)q(s_1, \theta) + fq(s_2, \theta)$ , onde  $f$  é a parte fracionária próxima do ponto anterior. No exemplo acima,  $f = 0.3$ . Assim, a ponderação para o exemplo acima seria  $q(s, \theta) = (0,7)q(s_1, \theta) + (0,3)q(s_2, \theta)$ .

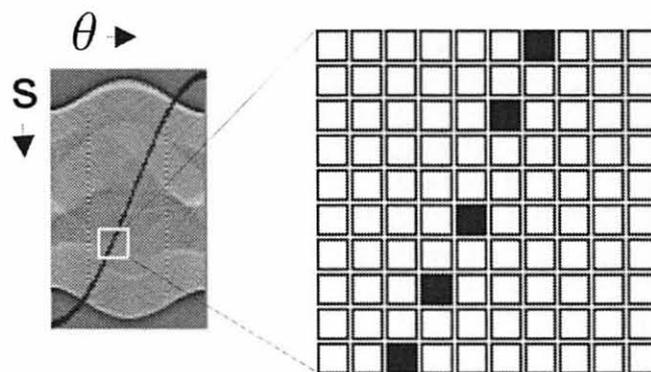


Figura 18- Região aumentada da senóide sobre o espaço de projeções com os pontos  $q(s, \theta)$  e os valores de  $s$  (pixels de tom escuro) obtidos pela equação (5) truncados (cálculo sem interpolação).

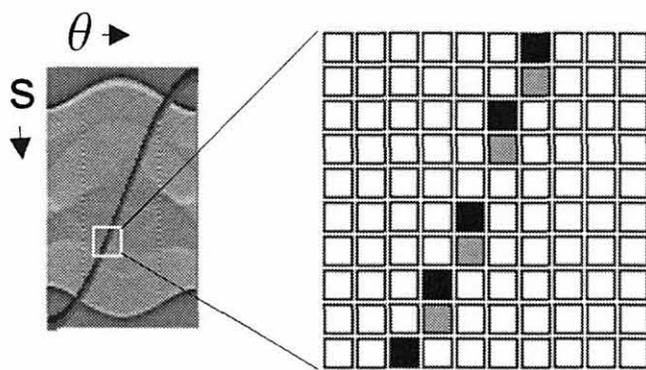


Figura 19 - Região aumentada da senóide sobre o espaço de projeções com os pontos  $q(s, \theta)$  e os valores de  $s_l$  (pixels de tom escuro) obtidos pela equação (5). Juntamente com os pontos na posição  $s_2$  (tom claro), serão multiplicados pelos pesos conforme equação (11) e somados para obter o valor do pixel de imagem  $f(x, y)$  (cálculo com interpolação).

### 5.1.7 Normalização

Analisando a equação de retroprojeção (35), outro ponto a considerar é que o resultado do somatório de  $N$  projeções filtradas é multiplicado por um fator  $\pi/N$ . Como no procedimento explicado para a rede de retroprojeção acontece apenas o somatório, é preciso ainda multiplicar o resultado deste somatório por  $\pi/N$ . Caso se utilize 256 tonalidades de cinza, a fórmula para normalização dos dados seria

$$\hat{f}(x, y) = \frac{256 (f(x, y) - f_{\min}(x, y))}{f_{\max}(x, y) - f_{\min}(x, y)} \quad (37)$$

onde  $f(x, y)$  seria qualquer ponto da imagem,  $f_{\min}(x, y)$  e  $f_{\max}(x, y)$  seriam o menor e maior valor da imagem respectivamente. A equação de normalização dos dados precisa então ser aplicada para visualização da imagem reconstruída.

## 5.2 REDE DE FILTRAGEM

Assim como na fase de retroprojeção, para a fase de filtragem pode-se ter outra rede neural parcialmente conectada. Na camada de entrada da rede de retroprojeção temos que inserir os valores das projeções filtradas. Montando-se uma rede de filtragem, a camada de saída respectiva pode ser a própria camada de entrada da rede de retroprojeção (figura 20). Precisamos então definir alguns conceitos para que possamos construir esta rede de filtragem, que nada mais vem a ser do que o processo matemático de convolução expresso em uma rede parcialmente conectada.

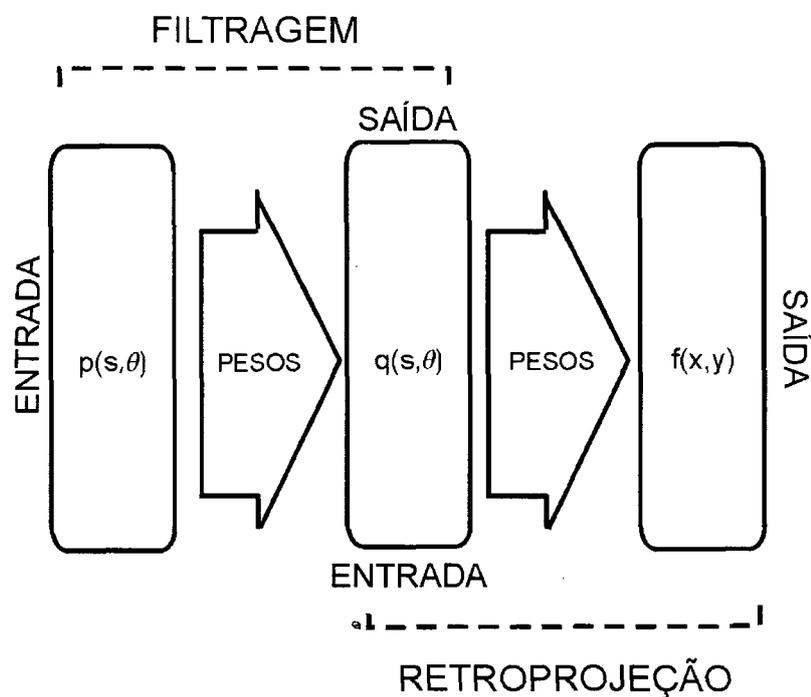


Figura 20 - Diagrama esquemático mostrando o relacionamento das camadas das redes de filtragem e retroprojeção. A camada de saída da rede de filtragem coincide com a camada de entrada da rede de retroprojeção.

### 5.2.1 Filtragem das Projeções

Na construção da rede de filtragem, as projeções serão filtradas (ROSENFELD e KAK, 1982, p.378) pela seguinte função de transferência  $H(w)$ , no espaço de frequência (também conhecida como filtro *Ram-Lak*):

$$H(w) = |w| b_w(w) \quad (38)$$

$$b_w(w) = 1, |w| \leq W$$

$$b_w(w) = 0, \text{ senão}$$

Transformando para o espaço de tempo, temos:

$$h(n) = \begin{cases} 1/4, & n = 0 \\ 0, & n \text{ par} \\ -\frac{1}{n^2\pi^2}, & n \text{ ímpar} \end{cases} \quad (39)$$

Na equação (39)  $h(n)$  são os coeficientes do filtro<sup>9</sup> para cada  $n$ . Analisando esta equação vemos que a amplitude função da resposta impulsional diminui com  $n$  tendendo ao infinito, e os valores pares são zerados. A convolução seria expressa da seguinte forma, em termos discretos:

$$q(n, \theta) = \sum_{k=-\infty}^{+\infty} h(n-k) p(k, \theta), \quad n = 0, 1, \dots, N-1 \quad (40)$$

Na prática considera-se que cada projeção têm extensão finita, sendo  $p(k, \theta)$  igual a zero fora do intervalo  $k=0, 1, \dots, N-1$ . A equação toma a seguinte forma:

$$q(n, \theta) = \sum_{k=0}^{N-1} h(n-k) p(k, \theta), \quad n = 0, 1, \dots, N-1 \quad (41)$$

---

<sup>9</sup> Para efeito de simplificação, o termo representando o intervalo de amostragem no texto original foi considerado unitário.

Fazendo um paralelo com a equação (36), vemos a correspondência conforme a tabela 1.

TABELA 1 – CORRESPONDÊNCIA ENTRE VARIÁVEIS DA EQUAÇÃO (36) E (41)

Eq. de Convolução	Rede Neural
$p(k, \theta)$	$x_i$
$h(n-k)$	$w_{ij}$
$q(n, \theta)$	$y_j$

A equação (36) expressa que a rede é totalmente conectada. Porém analisando a equação de filtragem, vemos que não há necessidade de total conexão devido ao fato de que  $h(n)$  assume 0 (zero) para valores pares. Outro ponto que também pode ser explorado é o de que a função de transferência assume valores ínfimos conforme aumenta o valor de  $n$ , e podemos definir um limiar  $\varepsilon$  para, abaixo do qual, truncarmos os valores (assim como em alguns algoritmos de transformada para aplicações em compressão de dados). Deste modo, na construção da rede, uma rede parcialmente conectada pode executar a tarefa de filtragem das projeções.

Se não utilizamos todos os valores para  $h(n)$  na convolução, significa que o processo de filtragem pode ser entendido como o processamento das projeções por um filtro FIR (*Finite-Duration Impulse Response*, ou Resposta ao Impulso Finita em Duração). O filtro FIR é aquele que possui um número finito de coeficientes diferentes de zero (OPPENHEIM & SCHAFFER, 1989, p.31) sendo sempre estável.

O truncamento de valores de  $h(n)$  equivale então a utilizar um número de coeficientes do filtro FIR menor que o tamanho total da projeção. A Tabela 2 a seguir mostra para uma projeção de 100 pontos, o número de coeficientes acima do valor de limiar  $\varepsilon$  utilizado.

TABELA 2 – NÚMERO DE COEFICIENTES POR VALOR DE LIMIAR

Limiar( $\epsilon$ )	Número de coeficientes
0,1	2
0,05	3
0,01	6
0,005	8
0,001	16
0,0005	23
0,0001	52
0,00005	68
0,00001	100

Como o filtro  $h(n)$  é genérico, pode-se utilizar outros tipos além do filtro *Ram-Lak*, como os filtros *Shepp-Logan*, *Cosseno Passa-baixa* e *Hamming Generalizado* (JAIN, 1989,p.448). Para o caso de imagens com ruído, tais filtros produzem nas projeções filtradas uma suavização, diminuindo o efeito das altas frequências na imagem (figura 21).

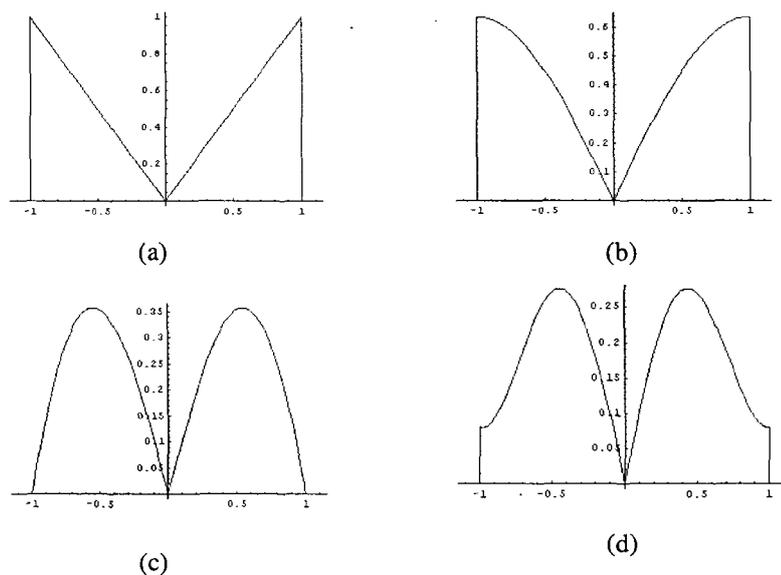


Figura 21 - Resposta em frequência dos filtros: (a) *Ram-Lak*, (b) *Shepp-Logan*, (c) *Cosseno Passa-baixa* e (d) *Hamming Generalizado*. Ao contrário do filtro *Ram-Lak*, os outros tipos de filtros apresentam uma resposta menor às altas frequências.

Se simbolizarmos a resposta em frequência do filtro *Ram-Lak* da equação (38) como  $H_{RL}$ , a expressão para o filtro Shepp-Logan fica

$$H_{SL} = H_{RL} \frac{\text{sen}(\pi w)}{\pi w} \quad (42)$$

Para o Cosseno Passa-baixa

$$H_{CP} = H_{RL} \cos(\pi w) \quad (43)$$

E para o *Hamming* Generalizado

$$H_{HG} = H_{RL} (\alpha + (1-\alpha) \cos 2\pi w), \quad 0 \leq \alpha \leq 1 \quad (44)$$

Portanto, os outros tipos de filtros podem ser expressos em termos do filtro básico *Ram-Lak*.

### 5.2.2 Estrutura de Dados – Vetor de Estruturas

Assim como na rede de retroprojeção, pode-se trabalhar no algoritmo com um vetor de estruturas que represente a rede de filtragem. Ressalta-se novamente a economia de memória que pode ser conseguida através desta alternativa, por não ser necessário ponteiros para os nós das camadas de entrada e saída, bem como para o próximo peso.

Além disto, os dados das projeções não necessitam estar em uma estrutura de nós como listas ligadas, bastando um vetor bidimensional para os pontos das projeções anteriores e outro para as projeções filtradas. Este último representa próprio vetor de entrada para a rede de retroprojeção vista no capítulo 5.

### 5.2.3 Estrutura de Dados – Listas Ligadas

Semelhante à rede de retroprojeção, a estrutura do nó da lista que seria o neurônio de entrada da rede teria a seguinte definição, utilizando a expressão da linguagem Pascal:

```
NF = record
    ID: integer;
    valor : real;
    s, theta : integer;
    P, N: pointer;
end;
```

Na estrutura *NF*, *ID* é a identificação do neurônio, com *s* variando no intervalo  $[0, N-1]$  e *theta* em  $[0, M-1]$ , o neurônio é identificado no intervalo  $[0, MN-1]$ . Se  $N=100$  e  $M=60$  por exemplo, *ID* assume no mínimo 0 e no máximo 5999. A variável *valor* guardará o valor da projeção no ponto especificado.

A estrutura da camada de entrada *NF* é, portanto, idêntica à da camada de saída que, por sua vez, é a estrutura da camada de entrada *NI* da rede de retroprojeção. Da mesma forma, também teremos a camada de entrada conectada seqüencialmente como uma lista encadeada dupla.

Para conectarmos as camadas de entrada e saída da rede de filtragem, o procedimento agora difere do exposto para a rede de retroprojeção. Ali, o processo de montagem da lista de pesos era orientado da camada de saída para a de entrada.

Na rede de filtragem, sabe-se que para a conexão total, todos os neurônios da camada de entrada têm um peso conectado a um neurônio de saída, assim acontecendo em consequência para toda a camada de saída. Pode-se proceder à montagem da lista de pesos conectando para cada neurônio de saída todos os neurônios de entrada e restringindo a existência do peso ao limiar  $\varepsilon$  definido.

A estrutura de dados para o peso difere da estrutura para a rede de retroprojeção. Aqui se precisa guardar o valor correspondente aos coeficientes da função  $h(n)$ . Assim,

```

W = record
  ID : integer;
  valor: real;
  N: pointer;
  N1 : pointer;
  N2 : pointer;
end;

```

Temos nesta estrutura a identificação do peso  $ID$ , o *valor* a ser armazenado,  $N$  como ponteiro para o próximo peso,  $N1$  o ponteiro para o neurônio na camada de entrada e  $N2$  o ponteiro para o neurônio da camada de saída.

Para calcularmos os valores do filtro, utilizamos a equação (39) que expressa o filtro *Ram-Lak*. Pela equação (42) relaciona-se a diferença  $h(n-k)$  com o peso que conecta o neurônio na camada de entrada  $p(k, \theta)$  com o neurônio na camada de saída  $q(n, \theta)$ , sendo que se  $h(n-k) < \varepsilon$ , faz-se  $h(n-k) = 0$ , ou seja, não se cria peso ou conexão.

### 5.3 RECONSTRUÇÃO EM 3D COM REDES PARCIALMENTE CONECTADAS

Após o estudo das redes de retroprojeção e rede de filtragem, pode-se estruturar um modelo conjunto de algoritmo a partir da união dos dois tipos. Este modelo visa a reconstrução de várias seções de um objeto a partir das projeções destas seções. Uma proposta de algoritmo está baseada no modelo de listas ligadas e outra em vetores de estruturas (figuras 22 e 23 respectivamente). Nestes algoritmos entram-se com  $z$  seções a ser reconstruídas, sendo que as redes são construídas uma única vez, e a cada leitura das projeções seguintes as imagens são geradas apenas pela alimentação dos valores das projeções nas redes interconectadas.

Para que seja vantajosa a reconstrução usando redes parcialmente conectadas, o tempo de construção somado ao tempo de alimentação das redes deve ser menor que o tempo de reconstrução utilizando um algoritmo comum.

Considerando as seguintes definições no quadro 1, onde os passos referem-se às etapas no fluxograma das figuras.

QUADRO 1 - DEFINIÇÃO DE VARIÁVEIS DE INTERVALOS DE TEMPOS

Variável	Descrição	Passos (Fig.22)	Passos (Fig.23)
$t_b$	Intervalo de tempo de construção das redes (retroprojeção e filtragem)	2,3,4,5,6	2,3
$t_f$	Intervalo de tempo de alimentação de uma projeção nas redes	9,10	6,7
$T_a$	Intervalo de tempo de reconstrução de uma projeção utilizando algoritmo normal.	-	-
$Z$	Número de seções a serem reconstruídas	-	-

O tempo total de reconstrução  $t_r$  pelas redes deverá ser

$$t_r = t_b + z t_f \quad (45)$$

E o tempo de reconstrução pelo algoritmo normal  $t_n$  deverá ser

$$t_n = z t_a \quad (46)$$

Conforme o exposto sobre a vantagem da reconstrução por redes, a seguinte relação deverá valer

$$t_r < t_n \quad (47)$$

Substituindo (45) e (46) em (47) e isolando para  $z$ , têm-se

$$z > \frac{t_b}{t_a - t_f} \quad (48)$$

Desta forma, existirá um valor de  $z$  a partir do qual é vantajosa a reconstrução por redes parcialmente conectadas contra a reconstrução por algoritmo normal. Estas relações serão úteis para comparações na fase de resultados.

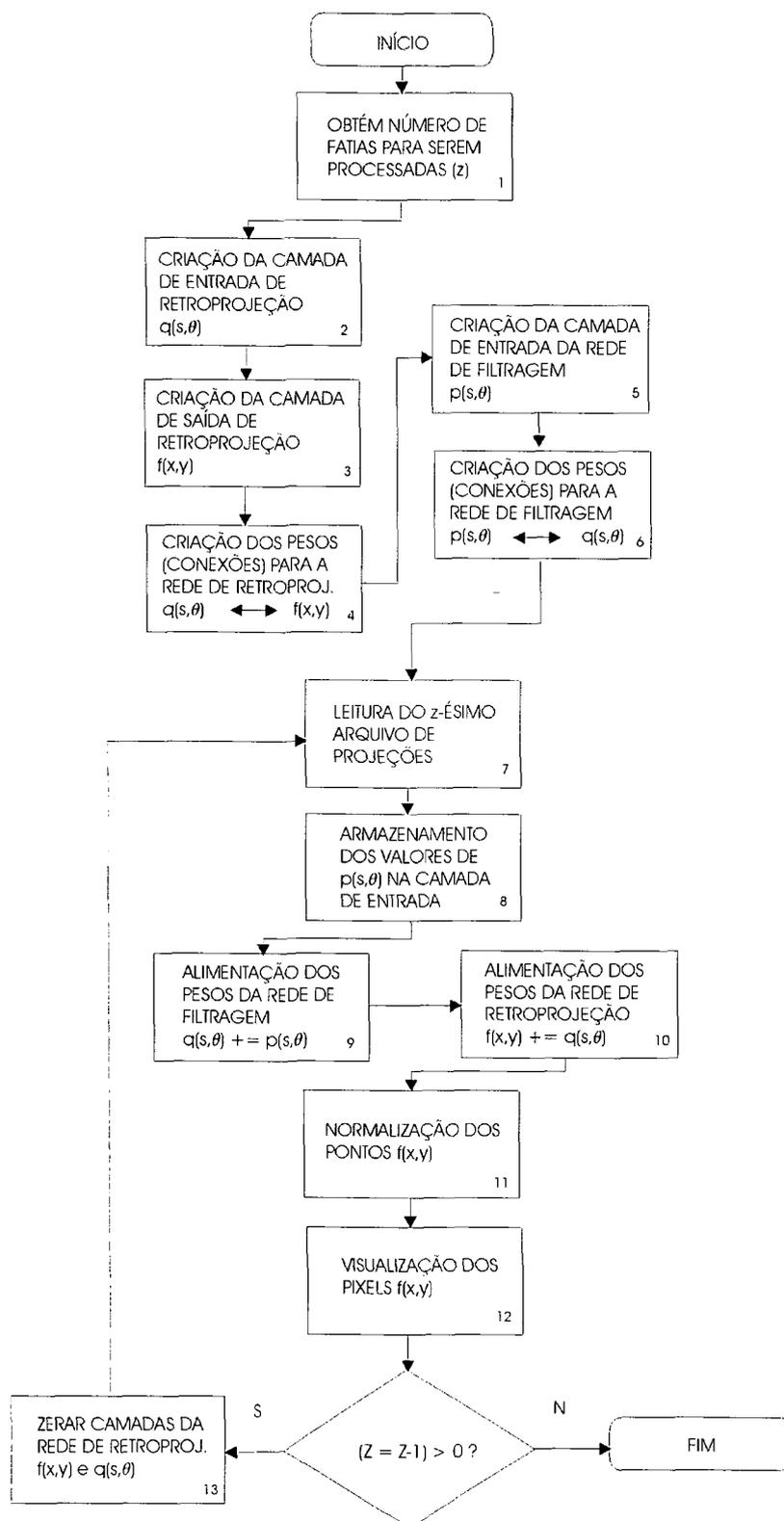


Figura 22 - Fluxograma com os passos necessários para a reconstrução de imagens 3D com listas ligadas.

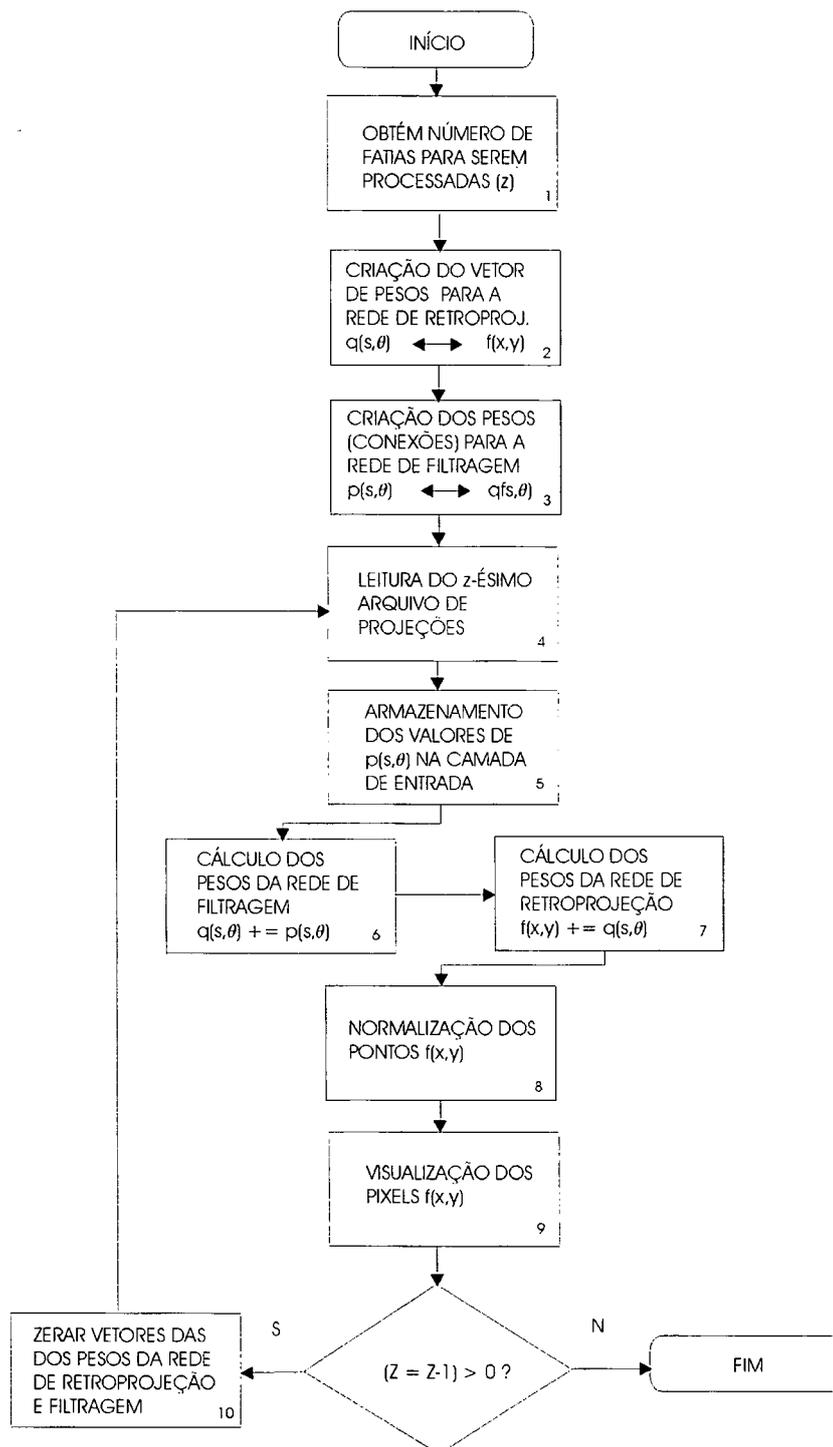


Figura 23 - Fluxograma dos passos necessários para a reconstrução 3D usando vetor de estruturas. Vê-se que a diferença principal em relação às listas ligadas está na construção das redes (passos 2 e 3, enquanto que na figura anterior 2,3,4,5 e 6).

## 5.4 QUALIDADE DE IMAGEM

Para que se tenha um meio de avaliar a imagem reconstruída por um determinado algoritmo, é preciso comparar o resultado com a representação da imagem original. O aspecto da qualidade de imagem pode ser abordado de duas formas: quantitativa e subjetiva. Pela forma subjetiva a comparação é feita através de avaliação subjetiva de um observador utilizando um escala crescente de qualidade de imagem. Através da forma quantitativa é preciso definir uma métrica para comparação entre a imagem original e a reconstruída. (JAIN, 1989, p.57). Apesar da forma quantitativa permitir a medição da qualidade de imagem por uma grandeza, nem sempre irá coincidir com a avaliação subjetiva de um observador.

O critério de qualidade utilizado para a fase de resultados foi a razão de pico de sinal-ruído (PSNR). Esta grandeza é relativa e mede a diferença de uma imagem reconstruída em relação à imagem original em uma escala de quantização de cores. A fórmula (49) define o cálculo do PSNR (LEPLEY; FORKERT, 1997, p. 64; JAIN, 1989, p.59), onde  $o_{ij}$  é a imagem original,  $r_{ij}$  a imagem reconstruída, com  $1 \leq i \leq M$  e  $1 \leq j \leq N$ ; e  $p$  é o número de bits significativos que define a faixa de valores que um pixel pode assumir.

$$PSNR = 10 \log_{10} \left[ \frac{(2^p)^2}{\frac{1}{MN} \sum_{i,j} (o_{ij} - r_{ij})^2} \right] \quad (49)$$

Caso se trabalhe com imagens de 8 bits, ou seja, 256 níveis de cinza, a fórmula torna-se

$$PSNR = 10 \log_{10} \left[ \frac{65536}{\frac{1}{MN} \sum_{i,j} (o_{ij} - r_{ij})^2} \right] \quad (50)$$

Têm-se desta forma um critério de classificação da qualidade da imagem com a métrica sendo definida pelo PSNR da imagem reconstruída através dos algoritmos em relação à imagem original.

Uma alternativa de análise mais aprofundada de uma imagem pode ser dada pelo histograma das imagens. No histograma representamos os níveis de cinza possíveis de uma imagem e a quantidade de ocorrências para determinado nível. Estas quantidades podem ser interpretadas como estimativas de probabilidades de determinado nível de cinza ocorrerem (GONZALES; WOODS, 1992, p.122). Mesmo se duas imagens possuírem o mesmo valor de PSNR, pode acontecer das mesmas apresentarem perfis de histograma bastante diferenciados.

## 5.5 PROCESSAMENTO PARALELO

As redes neurais podem ter a sua arquitetura de pesos modificada de forma a tratar cada alimentação de valor da camada de entrada para a camada de saída como um processo individual e autônomo<sup>10</sup>. Pode-se ter então vários processos de transferência de valores entre as camadas concorrendo paralelamente. Em sistemas que tenham arquitetura de multiprocessamento, isto pode ser uma vantagem. Cada processador pode se encarregar de rodar um processo independente de outro durante a execução do algoritmo, compartilhando

---

<sup>10</sup> O processo individual é conhecido como *thread* (carretel).

o mesmo espaço de memória onde os vetores dos dados dos senogramas e da imagem final estarão dispostos.

A implementação paralela da rede neural pode ser bem explorada se o hardware utilizar processadores paralelos.

## 6 RESULTADOS

Para avaliar o uso de redes parcialmente conectadas para a reconstrução de imagens de CT, precisou-se comparar o desempenho frente ao algoritmo de retroprojeção filtrada, sendo este o mais empregado (ver Apêndice A). Pelo uso de determinados critérios de avaliação, mensurou-se tal desempenho de forma relativa.

Os seguintes critérios de avaliação foram utilizados:

- Tempo de execução
- Qualidade de imagem do algoritmo normal
- Número de pesos ou conexões necessárias

Como foi preciso uma avaliação comparativa, para o teste dos algoritmos usou-se o *head phantom* de *Shepp-Logan* (KAK; SLANEY, 1988, p.53-55; JAIN, 1989, p.438-439). O modelo foi construído, com base nas especificações das elipses em posições fixas, em imagens de tamanho 100x100 e 256x256 e após foram efetuadas as respectivas transformadas de Radon, com a imagem 100x100 resultando em 60 projeções de 100 pontos; e para a imagem de 256x256 em 90 projeções de 256 pontos (figura 24).

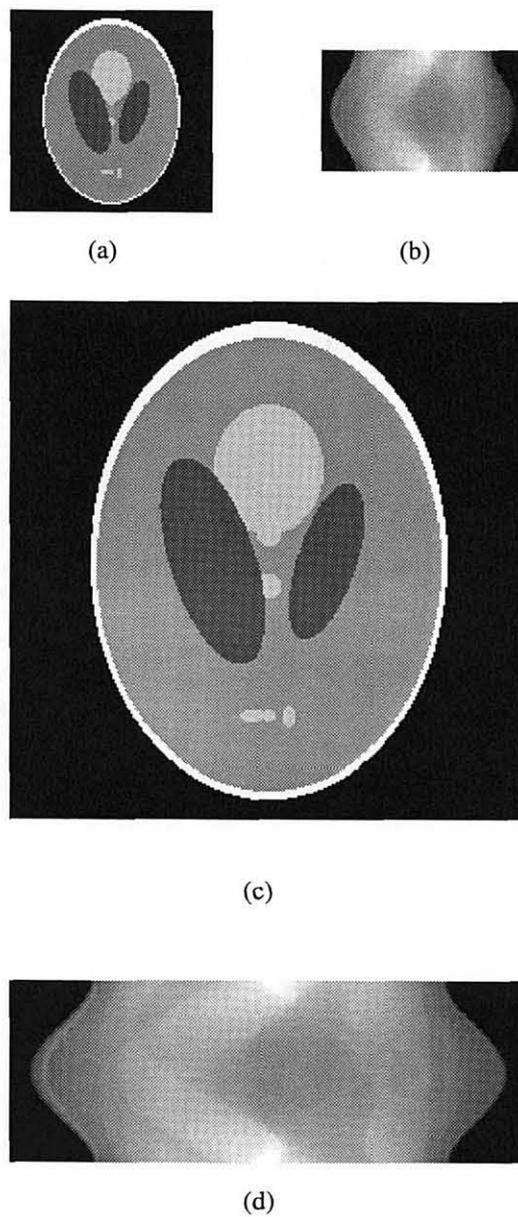


Figura 24 - Head Phantom de Shepp-Logan utilizados para testes dos algoritmos para redes parcialmente conectadas: (a) Imagem 100x100; (b) Imagem da Transformada de Radon de (a) com 60 projeções de 100 pontos; (c) Imagem 256x256; (d) Imagem da Transformada de Radon (c) com 90 projeções de 256 pontos.

## 6.1 TEMPOS DE CONSTRUÇÃO, EXECUÇÃO E QUALIDADE DA IMAGEM

Em termos de tempo de execução, foi medido o tempo de execução do algoritmo de retroprojeção filtrada e das redes parcialmente conectadas (filtragem e retroprojeção), em um microcomputador Pentium 800 MHz com 256 Mb de memória (características em detalhes no Apêndice B). No método de reconstrução usando redes, é necessário medir ainda o tempo de construção da rede, que é separado do tempo de realização ou execução. O tempo de construção foi calculado através da média de 1000 execuções, enquanto que o tempo de execução foi calculado a partir de uma média de 10.000 execuções dos algoritmos. O *head phantom* utilizado para medição de tempos foi o de 60 projeções de 100 pontos, com interpolação e sem interpolação. O filtro utilizado foi o Ram-Lak sem janelamento da equação (38) com 100 coeficientes.

TABELA 3 - COMPARAÇÃO DE TEMPOS DE CONSTRUÇÃO DAS REDES

Algoritmo	Interpolação	Tempo (ms)
Listas Ligadas	Não	930
	Sim	1553
Vetor de Estruturas	Não	191
	Sim	384

Na Tabela 3 pode ser comparado o tempo de construção das redes (referentes aos passos 2,3,4,5,6 para listas ligadas da figura 22) e passos 2 e 3 para vetor de estruturas na figura 23). Os tempos de construção estão expressos em milisegundos. Fica claro que o tempo de construção através de vetor de estruturas é cerca de 5 (cinco) vezes menor que o tempo dispendido para listas ligadas.

TABELA 4 - COMPARAÇÃO DE TEMPOS DE EXECUÇÃO ENTRE ALGORITMOS

Algoritmo		Interpolação	Tempo(ms)	z	PSNR(dB)
Retroprojeção Filtrada	Convolução	Não	346	1,7	12,94
		Sim	406	3,8	14,72
	FFT	Não	285	3,7	14,86
		Sim	374	7,1	14,08
Rede Parcialmente Conectada		Não	63		13,88
		Sim	103		18,32

Na tabela 4 temos os algoritmos de retroprojeção filtrada (utilizando convolução e filtro no domínio de frequência utilizando FFT) e redes parcialmente conectadas. Para cada algoritmo foi considerada a reconstrução de imagens com interpolação ou não, sendo medidos os tempos de execução dos passos 9 e 10 para as listas ligadas e passos 6 e 7 para o vetor de estruturas (ver fluxogramas nas figuras 22 e 23). Foi colocado também o número de imagens necessárias para haver vantagem no uso das redes conforme a equação (48), na qual o tempo de construção da rede levado em conta foi com o algoritmo usando vetor de estruturas. Pode-se concluir que com uma pilha de mais de 7 (sete) imagens o algoritmo de rede supera o algoritmo mais rápido (com FFT). Ainda com relação à tabela 4 a última coluna expressa o PSNR em relação à imagem original, sendo que os algoritmos com redes interpoladas têm maior PSNR em relação aos algoritmos normais.

## 6.2 ANÁLISE DO NÚMERO DE COEFICIENTES NA REDE DE FILTRAGEM

Um aspecto que emergiu dos resultados a ser apontado é que o número de coeficientes utilizado para a rede de filtragem também interfere no tempo de construção e execução do algoritmo e na qualidade da imagem. Assim, para diferentes números de coeficientes foi medido o tempo de construção e de execução bem como o PSNR.

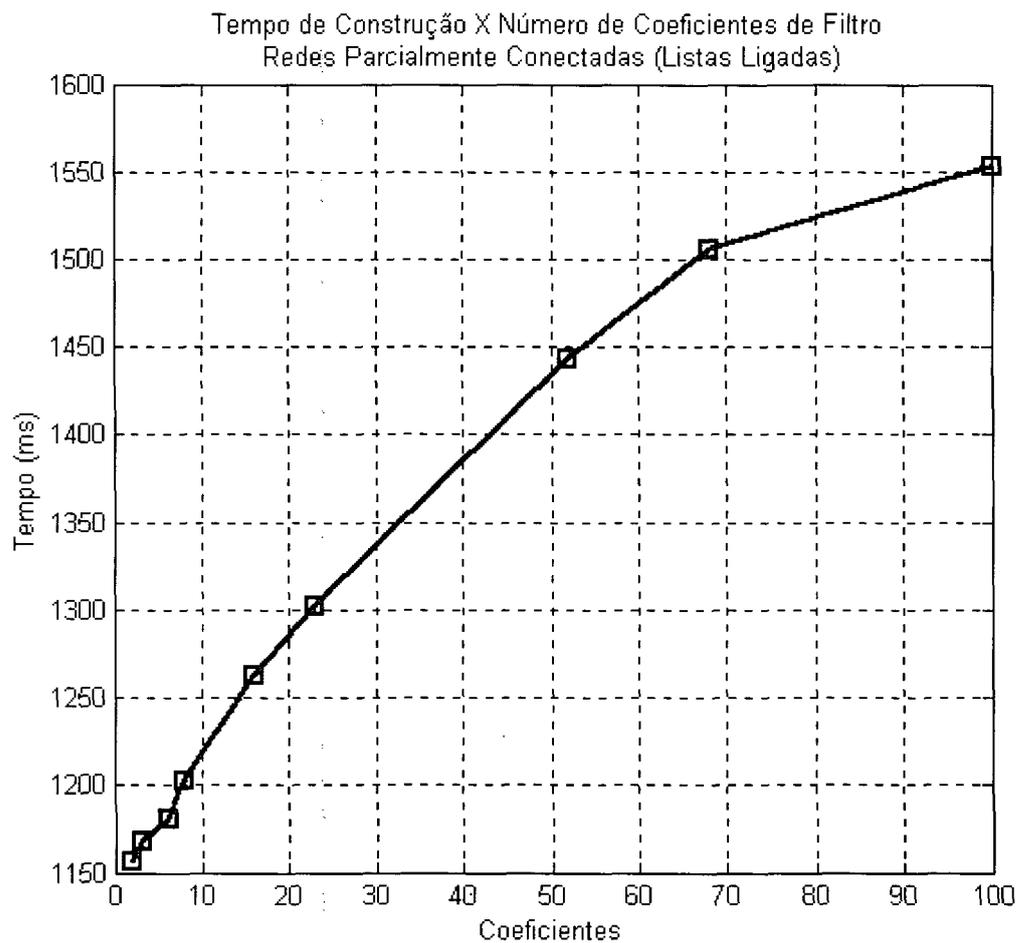


Figura 25 - Gráfico mostrando a relação do número de coeficientes com o tempo de construção de uma rede parcialmente conectada, mostrando uma relação aproximadamente linear.

Na medida em que se utiliza menos coeficientes para construir uma rede, leva-se menos tempo pois menos conexões serão necessárias, como se pode ver no gráfico da figura 25 . Esta variação de tempo também ocorre na execução do algoritmo (figura 26).

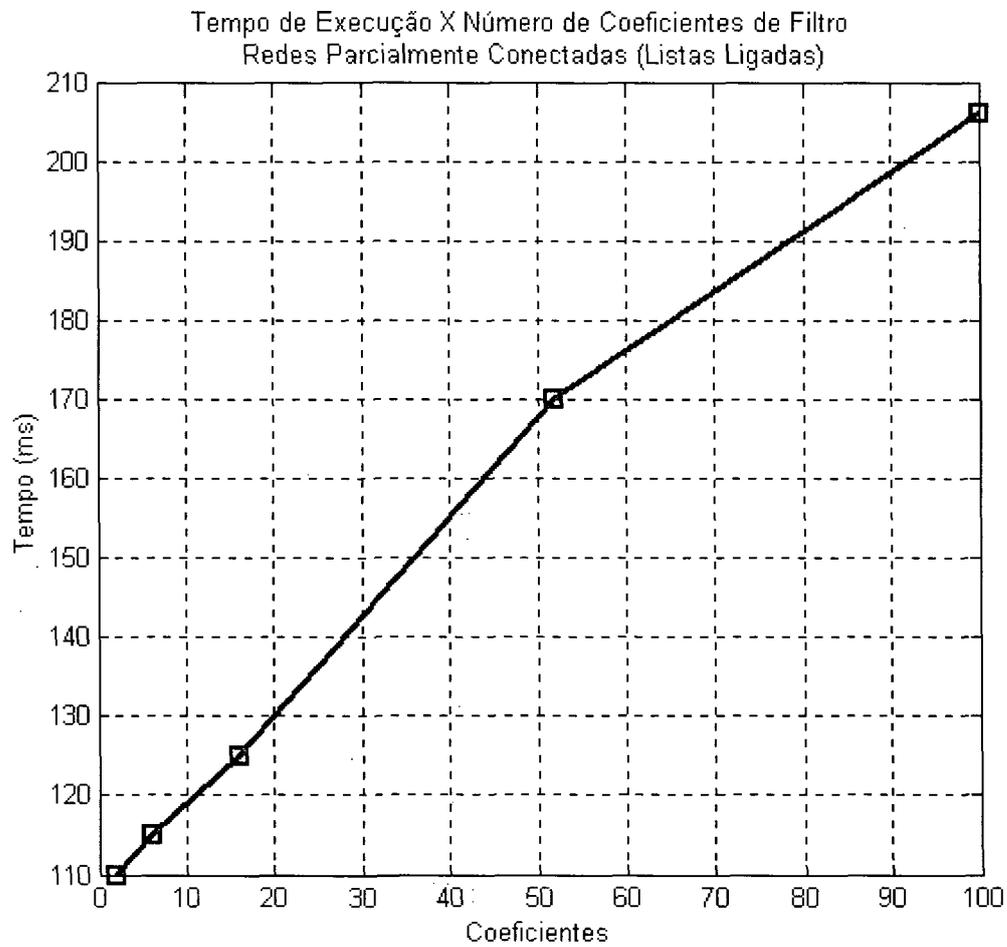


Figura 26 - Gráfico do tempo de execução do algoritmo de redes com o número de coeficientes utilizado para filtro, mostrando uma relação aproximadamente linear.

Com o impacto do número de coeficientes no tempo de construção e execução, pode-se comparar também a relação com o PSNR da imagem. No gráfico da figura 27 vemos que o PSNR da imagem aumenta com o aumento do número de coeficientes atingindo um pico, e depois decaindo até chegar ao número máximo de coeficientes. Se o critério de qualidade levado em conta for apenas o PSNR, então é possível construir uma rede com um número ótimo de coeficientes no qual teremos um tempo de construção e execução menor que com o uso de todos os coeficientes de filtragem. O uso de interpolação sempre resulta em um PSNR maior após certo número de coeficientes. O comportamento desta relação foi medido para a imagem de 100 pontos com 60 projeções, mas o padrão se repete de forma aproximada com diferentes tamanhos de imagem e número de projeções (figura 28).

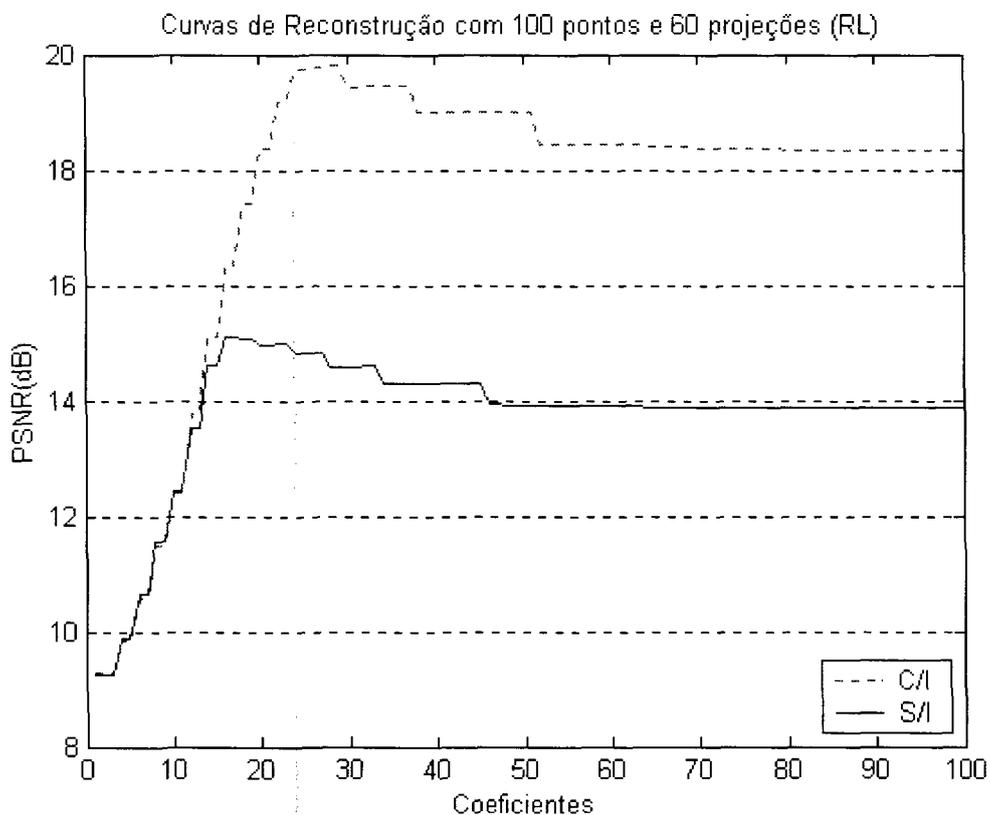


Figura 27 - Gráfico mostrando a relação entre o número de coeficientes do filtro Ram-Lak e a qualidade da imagem medida em PSNR com imagem de 100 pontos e 60 projeções. Em vermelho temos a rede com interpolação e em azul sem interpolação.

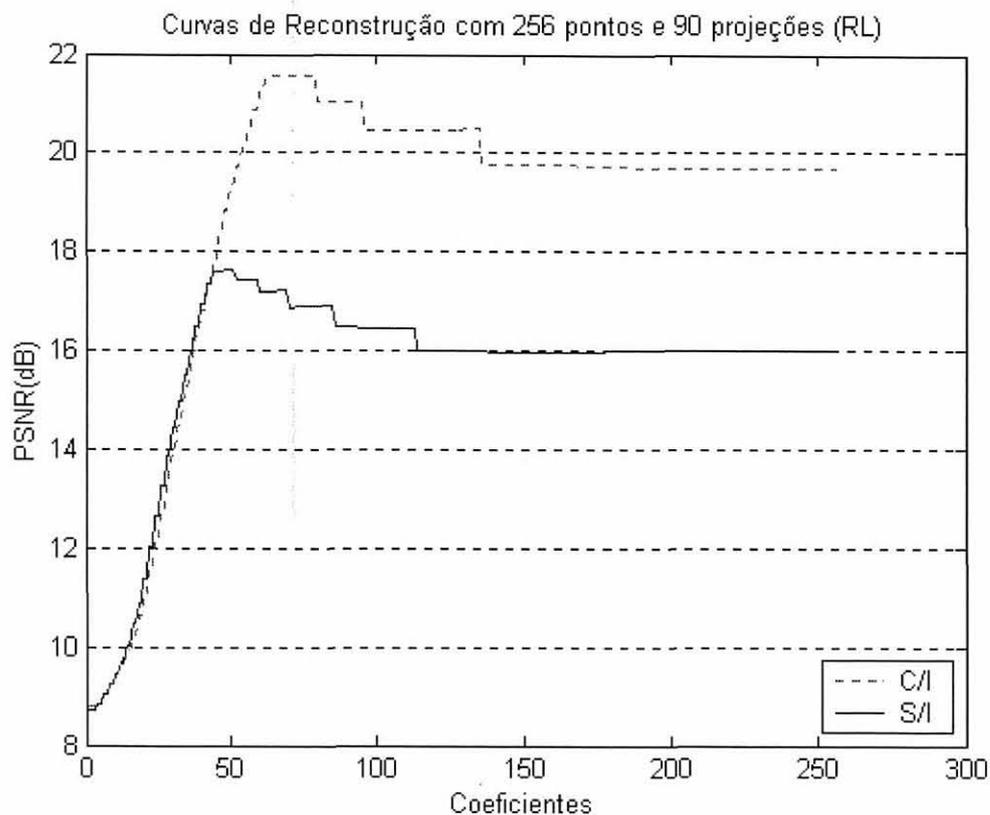


Figura 28 - Gráfico mostrando a relação de PSNR da imagem com o número de coeficientes de filtro Ram-Lak para 256 pontos de 90 projeções.

O número ótimo de coeficientes para a rede com interpolação do gráfico da figura 26 é de 29 coeficientes, e da rede sem interpolação é de 17 coeficientes para o *phantom* utilizado.

Nas figuras 29b e 29c pode se ver, respectivamente, a comparação entre

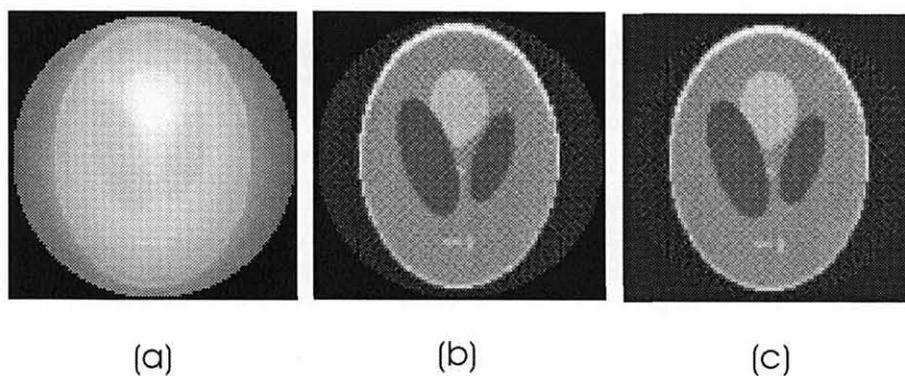


Figura 29 - Imagens reconstruídas da imagem com 100 pontos e 60 projeções com redes interpoladas usando filtros Ram-Lak com (a) 1 coeficiente; (b) 29 coeficientes; (c) 100 coeficientes.

as imagens reconstruídas com 29 coeficientes e 100 coeficientes, praticamente não havendo diferenças entre as mesmas. Nas figuras 30 e 31 podemos fazer a mesma comparação para uma imagem reconstruída a partir de 256 pontos e 90 projeções.

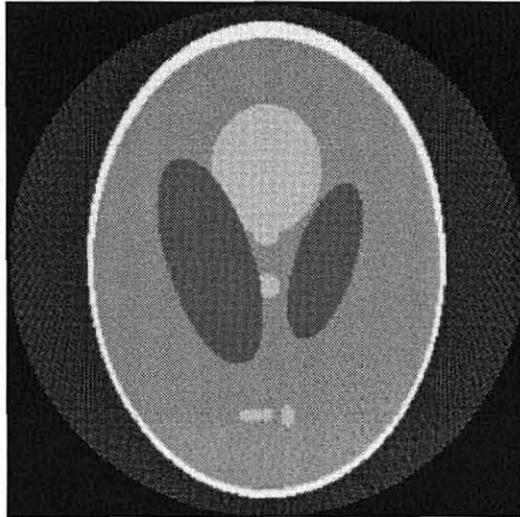


Figura 30 - Imagem reconstruída com redes a partir de 256 pontos e 90 projeções, utilizando 60 coeficientes de filtro.

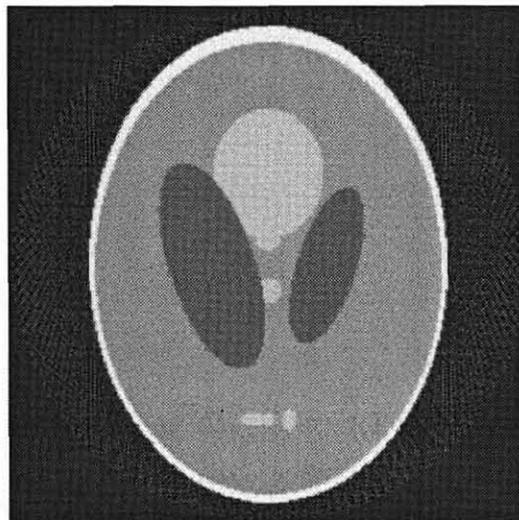


Figura 31 - Imagem reconstruída com redes a partir de 256 pontos e 90 projeções, utilizando 256 coeficientes de filtro.

Para verificar esta semelhança entre as imagens reconstruídas com menos coeficientes que o total, uma análise de histograma mostrando o perfil de níveis de cinza das imagens pode trazer mais luz ao entendimento.

Podemos verificar que no histograma da figura 32, a imagem reconstruída com um filtro de 1 coeficiente apresenta maior quantidade de níveis de cinza próximo do nível máximo (ver figura 29a). O perfil de uma imagem reconstruída com 17 coeficientes está muito aproximado do perfil com 100 coeficientes, mostrando diferença irrelevante. É válido salientar que mesmo utilizando 100 coeficientes, o perfil do histograma ainda está muito distante daquele da imagem original.

Semelhante análise pode ser feita para imagens com interpolação (figura 33), podendo-se notar que as diferenças entre a imagem reconstruída com um filtro de 29 coeficientes e com 100 coeficientes são um pouco mais significativas.

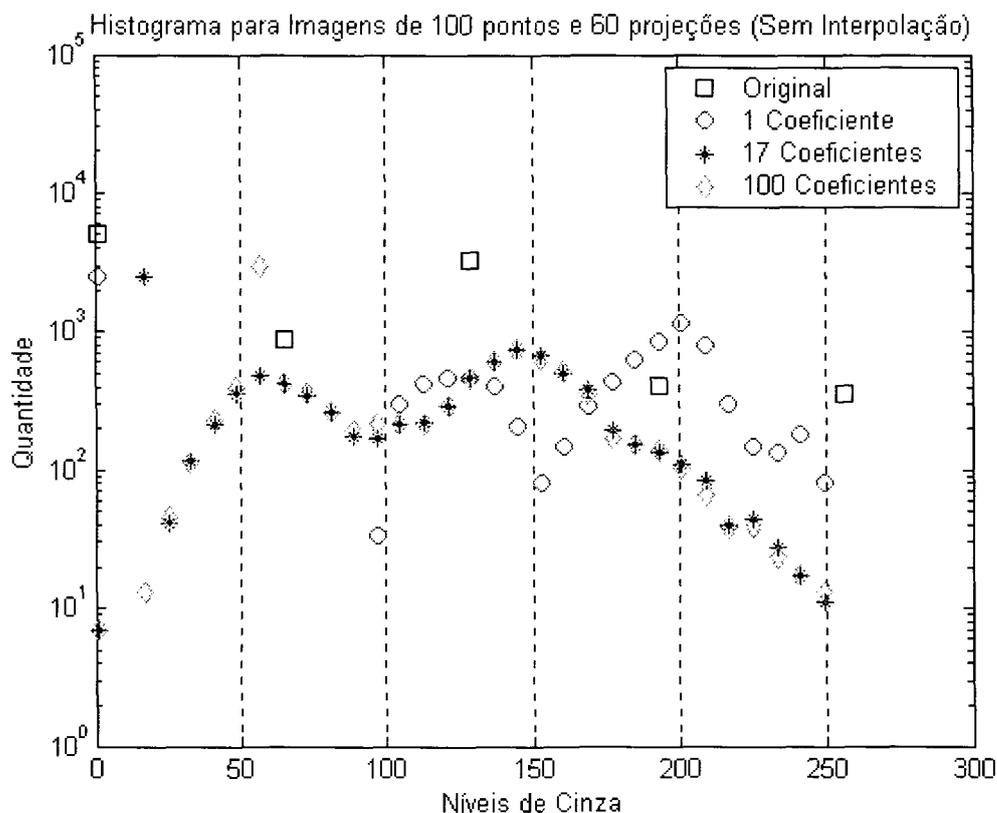


Figura 32 - Histograma mostrando os níveis de cinza das imagens reconstruídas para 100 pontos e 60 projeções sem interpolação. Na legenda, "original" representa o perfil de histograma para a imagem original.

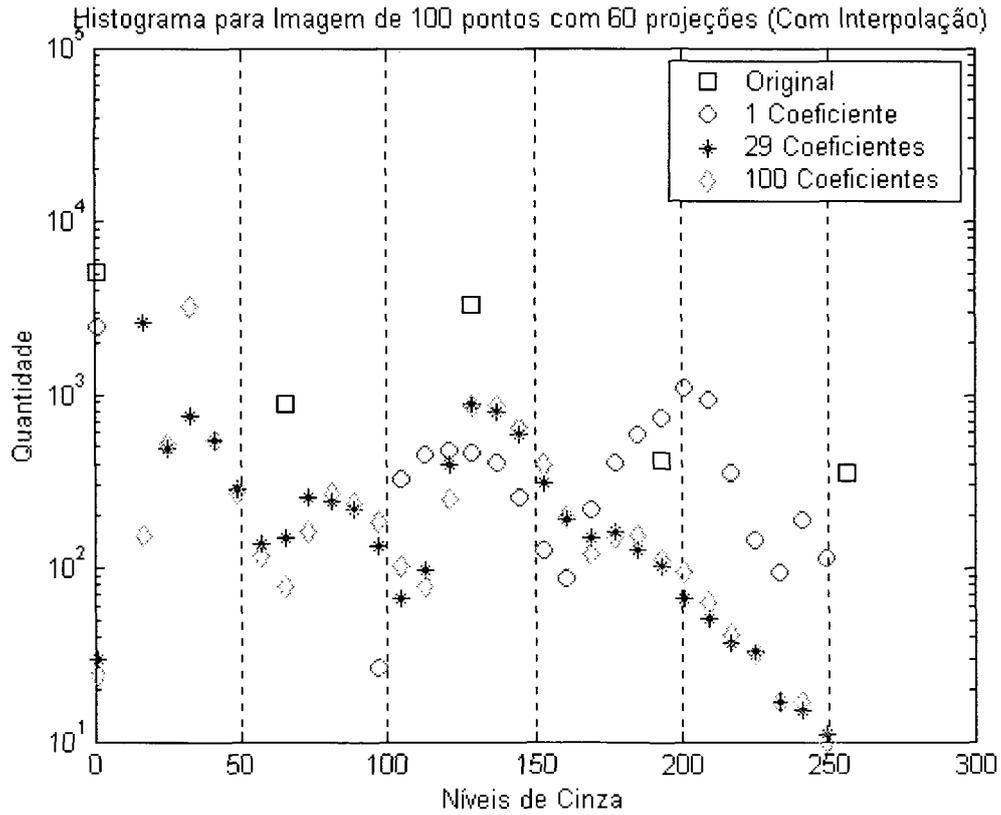


Figura 33 - Histograma mostrando os níveis de cinza das imagens reconstruídas para 100 pontos e 60 projeções com interpolação.

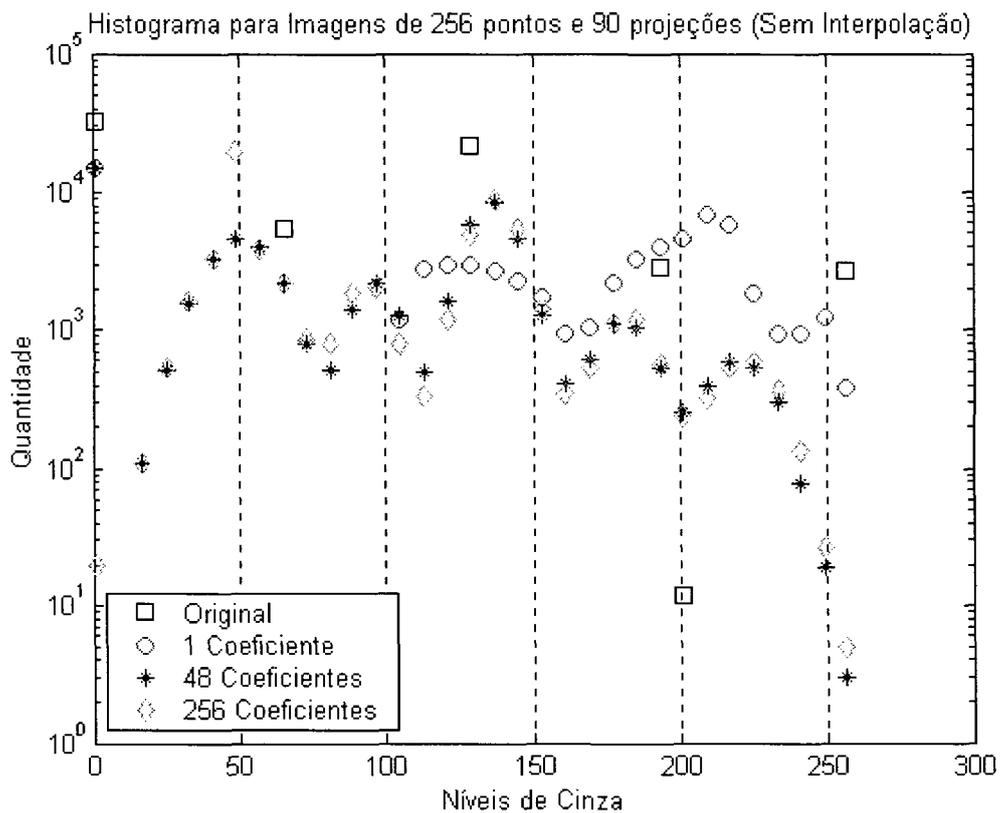


Figura 34 - Histograma mostrando os níveis de cinza das imagens reconstruídas para 256 pontos e 90 projeções sem interpolação.

A análise pode ser estendida para imagens com 256 pontos e 90 projeções (figuras 34 e 35), onde temos um detalhamento maior das imagens. Ainda assim, pode-se notar que a linha de análise para imagens com 100 pontos e 60 projeções se repete, notando-se uma diferença maior entre perfis de imagens sem interpolação para 48 e 256 coeficientes. Novamente ressalta-se que a aproximação com o perfil da imagem original ainda está muito distante. Para imagens com interpolação, o nível máximo de PSNR acontece com 60 coeficientes na rede de filtragem (figura 30). O nível de aproximação dos perfis é menor que a reconstrução sem interpolação.

### 6.3 COMPARAÇÃO DA REDE PARCIALMENTE CONECTADA COM PERCEPTRON DE DUAS CAMADAS

Esta comparação teve como objetivo aproximar a implementação do perceptron com expurgo de pesos definido no capítulo 4, em termos de número de conexões, da rede parcialmente conectada. Enquanto que a arquitetura da rede parcialmente conectada é definida *a priori*, no perceptron com expurgo de pesos a rede começa o treinamento totalmente conectada, a seguir diminuindo o número de conexões até alcançar um determinado valor.

O gráfico (figura 36) mostra que o PSNR da imagem reconstruída alcança valores bastante altos, praticamente igualando-se à imagem original. Mas este processo só é válido para esta imagem em particular, pois qualquer outra imagem de mesmo tamanho e número de projeções irá produzir imagens espúrias na saída do perceptron. A convergência na fase de treinamento para apenas esta imagem é garantida, acontecendo o treinamento de forma rápida.

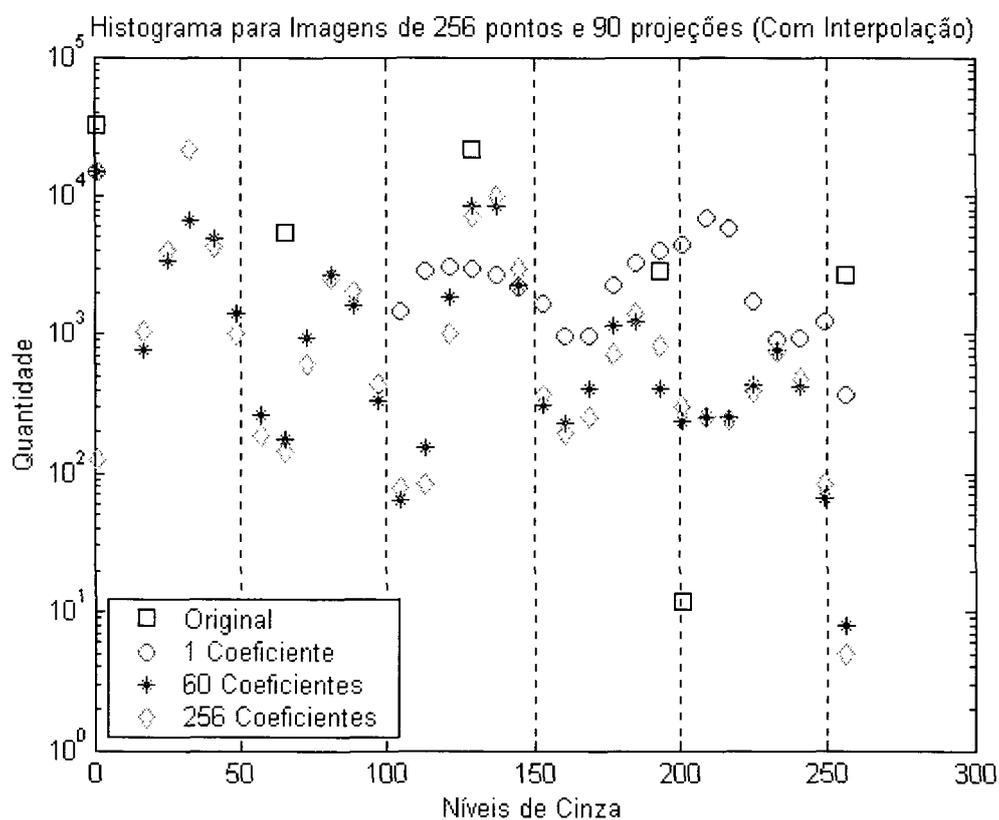


Figura 35 - Histograma mostrando os níveis de cinza das imagens reconstruídas para 256 pontos e 90 projeções com interpolação.

A tabela 5 mostra o número de conexões expurgadas a partir do treinamento do perceptron. E mesmo com 10.000 épocas de treinamento da rede, o nível de expurgo reduziu o número de conexões em 38% com relação à conexão total. Com a rede parcialmente conectada, esta redução chega a 95% em redes com interpolação e 98% sem interpolação.

TABELA 5 - COMPARAÇÃO ENTRE O PERCEPTRON E A REDE PARCIALMENTE CONECTADA

Algoritmo	Total de conexões	Conexões expurgadas	Conexões necessárias	% de Redução
Perceptron com expurgo de pesos	<b>219520</b>	<b>84080</b>	<b>135440</b>	<b>38%</b>
Rede parcialmente conectada	Com interpolação		<b>10340</b>	<b>95%</b>
	Sem interpolação		<b>5170</b>	<b>98%</b>

OBS.: Perceptron multicamada treinado com 10000 épocas.

Com referência à equação (33) utilizada para modificar o perceptron multicamada, o valor adotado para o parâmetro de regularização  $\lambda$  foi 0.9, e o parâmetro livre da equação (34)  $w_0$  variando de 0.1 até 0.000001, no total das 10000 épocas, decaindo exponencialmente.

Talvez uma alternativa para o uso de um perceptron multicamada seja o de aumentar o conjunto de treinamento, inserindo mais imagens para que o mesmo possa servir a um universo maior de imagens. O que se pode inferir a partir disto é que o algoritmo poderá levar mais tempo para chegar a um mínimo global (ou mesmo este nem poderá existir). Se no processo for efetuada a técnica de expurgo de pesos, pode ser também que não se tenha um número mais expressivo de pesos expurgados devido ao fato de que a rede de sinapses do perceptron multicamada precisará guardar mais informações. Concluindo, um perceptron pode não ser a melhor forma de reconstruir uma imagem tomográfica se comparados seu desempenho com a arquitetura da rede de retroprojeção e filtragem.

#### 6.4 ALGORITMO UTILIZANDO PROCESSAMENTO PARALELO

Foi desenvolvido um algoritmo no qual o código referente à alimentação de pesos para a camada de saída foi transformado de forma a permitir processamento paralelo. Ou seja, durante o somatório referente aos passos 9 e 10 para as listas ligadas do fluxograma da figura 22 os pesos teriam condições de ser

calculados paralelamente. Na fase de construção da rede referente aos passos 2 a 6, os processos (*threads*) eram criados individualmente e ficavam no aguardo da instrução de cálculo dos pesos para rodarem simultaneamente. Foi observado que o tempo de execução praticamente não mudou em relação ao algoritmo da rede parcialmente conectada normal, quando rodando em uma máquina de apenas um processador. Porém, em um computador com arquitetura dual (dois processadores) de mesma característica, o tempo das reconstruções foi reduzido pela metade. A figura 37 mostra uma imagem de um tubo de alumínio com dois orifícios reconstruída com o algoritmo modificado para processamento paralelo.

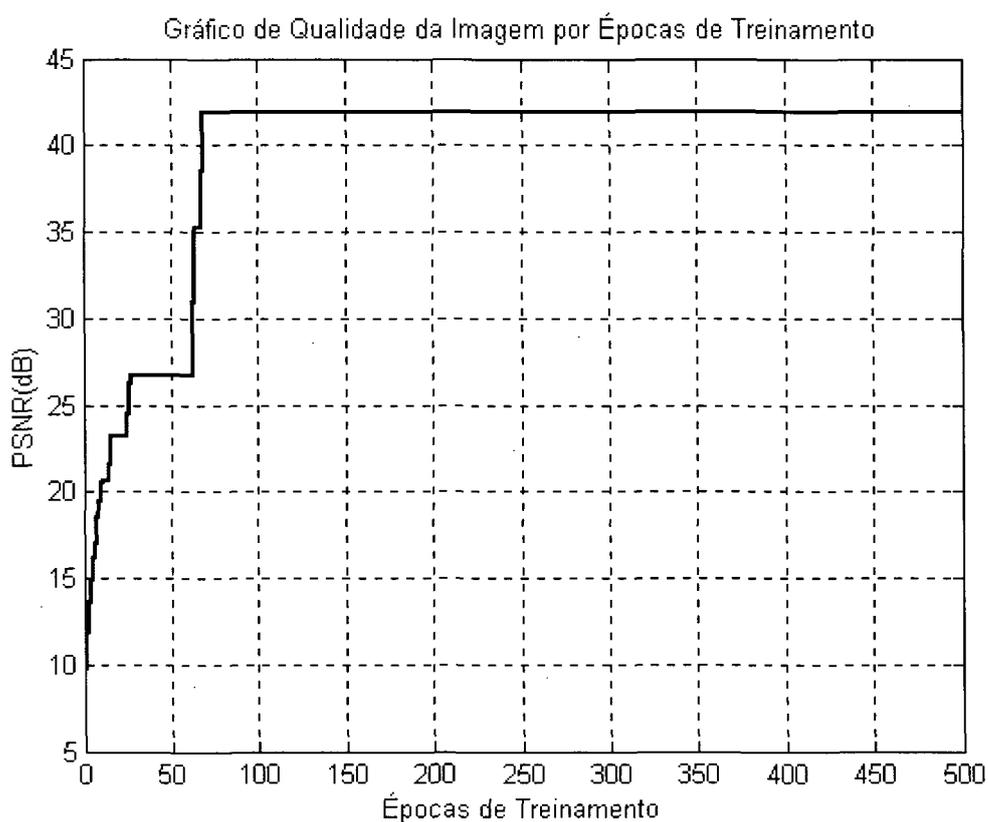


Figura 36 - Gráfico mostrando a evolução do PSNR da imagem reconstruída pelo perceptron durante as épocas de treinamento.

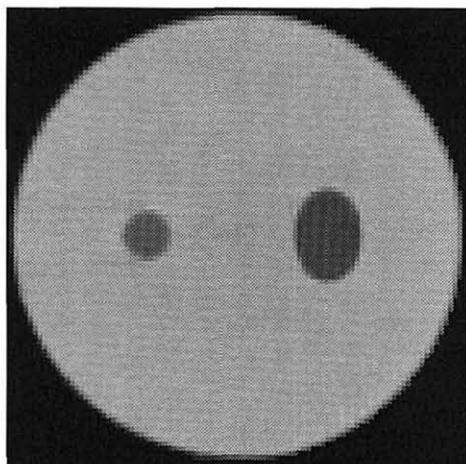


Figura 37 - Seção de um tubo de alumínio reconstruído com 100 pontos e 60 projeções utilizando algoritmo com processamento paralelo.

O algoritmo foi desenvolvido em linguagem Object Pascal compilado através da ferramenta Delphi™. É necessário declarar anteriormente uma classe do tipo *TThread* (classe base para todas as *threads* a serem rodadas, da qual herdamos a nova classe denominada *TSynapsisThread*):

```

TSynapsisThread = class(TThread)
private
    FST: integer;
protected
    procedure Execute; override;
public
    constructor Create(ST: integer);
end;

...
var
    SynapsisThread: array[0..N-1] of TSynapsisThread;

```

Na fase de construção da rede é preciso criar um vetor que irá guardar todas as *threads* (com o vetor *SynapsisThread* previamente declarado):

```

for i := 0 to N-1 do
    SynapsisThread[i] := TSynapsisThread.Create(i);

```

A variável  $i$  é o índice da *thread*. No algoritmo foi convencionado que para cada valor de  $s$ , possuindo o conjunto de todos os ângulos  $\theta$ , teria-se uma *thread*.

A declaração do método *Execute*, próprio de cada *thread*, que roda o algoritmo de alimentação da rede parcialmente conectada, seria

```

procedure TSynapsisThread.Execute;
var T : pointer;
begin
  T := Start[FST];
  while T <> nil do
    begin
      T.NeuronNext.Value := T.NeuronNext.Value +
(T.NeuronPrior.Value * T.Value);
      T := T.Next;
    end;
  end;

```

Para este algoritmo, o *loop while* executa para toda a lista de pesos existente e conectada entre si (começando pelo peso apontado por *Start*), o valor dos neurônios de saída (*NeuronNext.Value*), ou seja, a imagem final, é igual à soma do valor acumulado ali mais o valor do neurônio de entrada (*NeuronPrior.Value*), ou seja, o valor das projeções ponderadas pelo valor do peso (*T.Value*). Vale ressaltar aqui que os valores dos neurônios de saída devem iniciar com valor zero.

E finalmente o algoritmo de alimentação (*Feed*) que dá a partida no processo através do método *Resume* (quando da sua criação a *thread* fica em estado de latência, e aguarda pela execução, disparada por este método):

```

procedure TRede.Feed;
var T : pointer;
    i : integer;
begin
  for i := 0 to 99 do
    SynapsisThread[i].Resume;
  end;

```

As *threads* desta forma são executadas em paralelo. Ao final da execução de cada *thread*, a imagem final terá acumulado todos os valores necessários à reconstrução.

Neste algoritmo foi explicado apenas a rede de retroprojeção. Da mesma forma o conceito pode ser estendido para a rede de filtragem.

Como mencionado anteriormente, as vantagens deste algoritmo aumentam com o aumento do número de processadores existentes no hardware. Processadores com maior capacidade de *pipeline* podem também desempenhar o algoritmo com melhor performance.

## 7 CONCLUSÃO

A partir dos resultados obtidos pode-se concluir que a rede neural parcialmente conectada é uma boa alternativa de reconstrução tomográfica de imagens, proporcionando menor tempo de obtenção das imagens. Este método não requer o treinamento da rede ao contrário das redes perceptron multicamadas. A retroprojeção realizada pela rede neural constitui uma abordagem eficiente, diminuindo a complexidade de ordem  $O(N^4)$  para  $O(N^3)$ , em troca de um aumento de memória para o armazenamento das conexões entre os planos de projeção e imagem. Uma rede neural também foi empregada para filtragem das projeções, fornecendo uma abordagem uniforme mas outros métodos podem ser utilizados.

Além da retroprojeção truncada foi implementada uma retroprojeção com interpolação linear possibilitando uma melhora sensível na qualidade das imagens. Isto foi adicionado ao método na forma de pesos não unitários, aumentando-se a necessidade de memória para armazená-los. Porém, a complexidade computacional ainda se manteve  $O(N^3)$ .

A reconstrução de imagens de CT em três dimensões pode ser explorada a partir da rede parcialmente conectada para processar várias seções de um determinado objeto. Se forem conservados o tamanho da imagem e número de projeções, a rede não precisará ser reconstruída, aumentando a velocidade de reconstrução. Talvez esta seja a maior vantagem apresentada pela rede parcialmente conectada. Outro aspecto a ser explorado a partir disto é a reconstrução de imagens 3D a partir da teoria de reconstrução tomográfica 3D (Transformada de Radon com três variáveis). Pode-se imaginar uma rede que reconstrua a imagem em três dimensões diretamente, sem processar o objeto fatia a fatia. A ser considerado em trabalhos futuros, outra possibilidade refere-se ao uso de projeções radiais (*fan-beam*), geradas por fonte de raios divergente atravessando o objeto, o que exige correções para a Transformada Inversa de

Radon. Aqui se trabalhou com a suposição do feixe de raios paralelos e desta forma simplificou-se o processo.

Uma possibilidade que se abre também, é o uso das redes parcialmente conectadas para a obtenção de imagens *on-line* da seção de um objeto, caso o equipamento de tomografia possua vários conjuntos de fonte-detector que permitam obter todas as projeções ao mesmo tempo em intervalos de tempo curtos.

O algoritmo de reconstrução utilizado para as redes pode ser modificado (de maneira semelhante ao explanado no capítulo 6) de forma a permitir o processamento paralelo e ser executado em sistemas constituídos de mais de um processador e também em máquinas virtuais (PVM). Pode-se ir mais longe e também especular a construção de chips que tenham a arquitetura da rede parcialmente conectada, viabilizando a reconstrução tomográfica por hardware, tais como os chips neurais, equipando aparelhos de reconstrução portáteis.

Um tópico que merece ser objeto de estudo é a relação entre o número de coeficientes de filtro com a qualidade de imagem reconstruída obtida, a qual apresentou um decaimento com o aumento do número de coeficientes a partir de certo valor. Sem mencionarmos o aspecto subjetivo da apreciação da qualidade de imagem, o padrão para avaliação objetiva parece se repetir para vários tamanhos de imagem e número de projeções.

Outra aplicação para uso das redes está em viabilizar o armazenamento de imagens comprimidas. O processo da Transformada de Radon efetua a diminuição de tamanho da imagem original e isto tem condições de diminuir ainda mais o tamanho da imagem comprimida por métodos de transformada. O algoritmo com redes parcialmente conectadas pode efetuar de maneira rápida o processo inverso necessário para a etapa de descompressão de imagens.

## APÊNDICE A - ALGORITMO DE RETROPROJEÇÃO FILTRADA

A seguir está o algoritmo em linguagem Pascal utilizado para comparação com as redes parcialmente conectadas (RAO; KRIZ, 1995).

```

{ Fase de Filtragem das Projeções }

for i:= 1 to PROJ do { Numero de Projecoes }
begin
  for j:= 1 to TAMANHO do { Numero de Pontos da Projeção }
  begin
    { Gerando a Função h[k] }
    for k:= 1 to TAMANHO do
      h[k]:= (-2/((4.0*Abs(j-k)*Abs(j-k)*PI*PI)-1));

    { Multiplicando h[k] pela projeção }
    Q[i,j] := 0.0;
    for k:= 1 to TAMANHO do
      Q[i,j] := Q[i,j] + h[k]*P[i,k];
    end;
  end;
end;

{ Fase de Retroprojeção - Com Interpolação }

Fator := PI/PROJ;
TAMBY2 := TAMANHO div 2;

for Ang := 1 to PROJ do
begin
  { Convertendo para radianos }
  Theta := (Ang*Fator);

  for y := 1 to TAMANHO do
  begin
    for x := 1 to TAMANHO do
    begin
      { Calcula a posição S para a projeção }
      s := TAMBY2 - (x-TAMBY2)*Cos(Theta) - (y-TAMBY2)*Sin(Theta);

      { Calcula o fator de ponderação para interpolação linear }
      s1 := Floor(s);
      s2 := Ceil(s);
      W := s - s1;

      { Soma a contribuição para o ponto f(x,y) }
      F[x,y] := F[x,y] + Q[Ang,s1]*W + Q[Ang,s2]*(1-W);
    end;
  end;
end;
end;

```

No algoritmo acima, as constantes *PROJ* e *TAMANHO* indicam, respectivamente, o número de projeções e o número de pontos em cada projeção. As projeções estão na matriz  $P[PROJ, TAMANHO]$ , as projeções filtradas em  $Q[PROJ, TAMANHO]$  e a imagem reconstruída em  $F[TAMANHO, TAMANHO]$ .

A retroprojeção sem interpolação tem o seguinte algoritmo:

```
{ Fase de Retroprojeção - Sem Interpolação }

Fator := PI/PROJ;
TAMBY2 := TAMANHO div 2;

for Ang := 1 to PROJ do
begin
    { Convertendo para radianos }
    Theta := (Ang*Fator);

    for y := 1 to TAMANHO do
    begin
        for x := 1 to TAMANHO do
        begin
            { Calcula a posição S para a projeção }
            s := TAMBY2 - (x-TAMBY2)*Cos(Theta) - (y-TAMBY2)*Sin(Theta);

            { Soma a contribuição para o ponto f(x,y) }
            F[x,y] := F[x,y] + Q[Ang,s];
        end;
    end;
end;
```

## **APÊNDICE B - DESCRIÇÃO DOS RECURSOS COMPUTACIONAIS**

O equipamento utilizado na fase de resultados tem a seguinte descrição: um computador com processador Pentium 800 MHZ, 256 MB de memória RAM, 512 KB de memória cache e disco rígido de 10 GB. O sistema operacional utilizado foi o Windows™ 98 SE®, sendo utilizado para desenvolvimento dos sistemas de reconstrução a ferramenta Delphi™ 5 e C++ Builder® 5, além de outros softwares para apoio como Matlab® 6.0 R 12, Mathematica® 4 e Corel Draw™ 8.0. Nos testes para determinar tempos de execução, o sistema operacional estava apenas com elementos básicos de operação, não tendo programas residentes nem processos paralelos. A medida de tempo foi obtida usando marcadores de tempo dentro do próprio software de reconstrução.

## APÊNDICE C - O TEOREMA DA FATIA DE FOURIER

Este teorema (ROSENFELD; KAK, 1982, p.365), também conhecido como *teorema da projeção*, trata do relacionamento entre a Transformada de Fourier bidimensional da imagem  $f(x,y)$  e a transformada unidimensional das projeções  $p(s,\theta)$ , utilizado em algumas técnicas de reconstrução de imagens.

Seja  $F(u,v)$  a Transformada de Fourier da imagem  $f(x,y)$ . Pela definição,

$$F(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) e^{-j2\pi(ux+vy)} dx dy \quad (C.1)$$

Seja  $P(w,\theta)$  a transformada unidimensional de Fourier da projeção  $p(s,\theta)$  com relação à primeira variável. Assim,

$$P(w,\theta) = \int_{-\infty}^{\infty} p(s,\theta) e^{-j2\pi ws} ds \quad (C.2)$$

Vamos considerar os valores de  $F(u,v)$  usando como base  $v = 0$  no plano  $uv$ . Partindo da equação (8) temos

$$\begin{aligned} F(u,0) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) e^{-j2\pi ux} dx dy \\ &= \int_{-\infty}^{\infty} \left[ \int_{-\infty}^{\infty} f(x,y) dy \right] e^{-j2\pi ux} dx \\ &= \int_{-\infty}^{\infty} p(s,\theta) e^{-j2\pi us} ds = P(w,0) \end{aligned} \quad (C.3)$$

pois no ângulo  $\theta = 0$  os eixos  $x$  e  $s$  são coincidentes, e a integral entre colchetes representa a projeção da imagem neste ângulo.

Pode-se então obter os valores da transformada bidimensional de Fourier  $F(u,v)$  na linha  $v = 0$  a partir da transformada unidimensional da projeção vertical da imagem.

Este resultado pode ser generalizado considerando que, se  $F(w,\theta)$  representar a transformada bidimensional de Fourier em coordenadas polares, e  $P(w,\theta)$  a Transformada de Fourier da projeção  $p(s,\theta)$ , têm-se

$$F(w,\theta) = P(w,\theta) \quad (\text{C.4})$$

Para a prova do teorema em (C.4), faz-se com que  $f(s,u)$  seja a função  $f(x,y)$  rotacionada no sistema de coordenadas  $(s,u)$ . As coordenadas são dadas então por

$$\begin{bmatrix} s \\ t \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (\text{C.5})$$

As projeções então podem ser expressas por

$$p(s,\theta) = \int_{-\infty}^{\infty} f(s,t) dt \quad (\text{C.6})$$

Portanto,

$$P(w,\theta) = \int_{-\infty}^{\infty} p(s,\theta) e^{-j2\pi ws} ds = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(s,t) dt e^{-j2\pi ws} ds \quad (\text{C.7})$$

Fazendo a transformação em coordenadas  $(x,y)$  o lado direito da equação (C.7) temos

$$\begin{aligned} P(w,\theta) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) e^{-j2\pi w(x \cos\theta + y \sin\theta)} dx dy \\ &= F(u,v) \quad \text{para } u = w \cos\theta, v = w \sin\theta \\ &= F_p(w,\theta) \end{aligned} \quad (\text{C.8})$$

Assim, prova-se o teorema, que é conhecido como teorema da fatia de Fourier. Ou seja, obtendo-se as projeções de uma imagem para os ângulos  $\theta = 1, \dots, k$ , e fazendo a Transformada de Fourier de cada uma destas projeções, pode-se determinar os valores de  $F(u, v)$ . Se existem infinitas projeções, então a imagem final pode ser obtida pela Transformada Inversa de Fourier

$$f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u, v) e^{j2\pi(ux+vy)} du dv \quad (\text{C.9})$$

O teorema da fatia de Fourier permite assim outra alternativa de reconstrução de imagens de CT. Como na prática temos projeções finitas, é utilizado a Transformada Discreta de Fourier (DFT) ou a Transformada Rápida de Fourier (FFT), já mencionada no capítulo 3.

**ANEXO A - ARTIGO INTITULADO "RECONSTRUÇÃO DE IMAGENS  
TOMOGRAFICAS UTILIZANDO REDES NEURAS PARCIALMENTE  
CONECTADAS" APRESENTADO NO V CONGRESSO BRASILEIRO DE  
REDES NEURAS**

## Reconstrução de Imagens Tomográficas utilizando Redes Neurais Parcialmente Conectadas

Luciano F. de Medeiros<sup>1</sup>, Hamilton P. da Silva<sup>2</sup>, Eduardo Parente Ribeiro<sup>3</sup>

<sup>1</sup>Mestrando em Informática, UFPR

<sup>2</sup>Doutorando em Engenharia dos Materiais, MSc., UFPR/LACTEC

<sup>3</sup>Doutor, Prof. Adjunto, Depto. Engenharia Elétrica, UFPR

E-mails: lfm@netpar.com.br, hps@lactec.com.br, edu@eletr.ufpr.br

### Abstract

*We present a neural network approach for the tomographic imaging problem known as reconstruction from projections. Two feed-forward neural networks are used to perform the tasks of filtering (convolution) and back-projection. They are partially connected networks that are previously assembled based on problem geometry and do not need to be trained. We believe this approach should enable faster reconstruction by exploring the high degree of parallelism present on neural networks being suitable for the processing of sequential images (stacked slices) for 3D visualization.*

### 1. Introdução

Neste artigo apresentamos uma alternativa para obtenção de imagens tomográficas (*computerized tomography*, ou CT) através do paradigma de redes neurais. Na literatura, existe uma variedade de algoritmos. Duas das metas para a reconstrução de imagens de tomografia computadorizada é a busca de velocidade e melhoria de qualidade de apresentação de imagens.

Com base no conhecido algoritmo de retroprojeção filtrada, propomos aqui uma rede neural parcialmente conectada e sem necessidade de treinamento apta a realizar o processo de reconstrução de CT, de maneira a reduzir o tempo de processamento para projeções em seqüência (representação 3D de imagens de CT como seções de um objeto). Apresentamos também as vantagens e desvantagens desta técnica, pela diminuição da redundância inerente ao processo de reconstrução de CT.

Alguns autores [1,2,3] propuseram para o processo de reconstrução o modelo de rede neural *perceptron* multicamada, para aumentar a eficiência e qualidade de imagem, com o uso de

algoritmos de treinamento conforme o algoritmo *backpropagation*. Pode-se começar com uma rede totalmente conectada, e após utilizar métodos de expurgo de pesos para diminuir a redundância da reconstrução.

Apesar de o método ser viável para soluções locais, com um conjunto finito de imagens, o método pareceu pouco aplicável como solução generalizada, como demonstrado nos trabalhos de Rodrigues [4]. Em nosso algoritmo, foi efetuada a reconstrução com redes *feedforward* de uma camada parcialmente conectada sem a necessidade de treinamento pois os pesos são determinados previamente a partir da geometria do problema

### 2. Métodos

Para que tenhamos uma projeção tomográfica de determinado objeto, é necessário fazer com que uma certa quantidade de radiação (como raios-X ou raios gama) atravesse este objeto e projete uma espécie de "sombra" em um anteparo devidamente colocado. Conforme as propriedades deste objeto (como massa ou densidade específica), o mesmo apresentará resistência à penetração da radiação de emissão, atenuando conforme a seguinte relação [5, 6]:

$$I = I_0 \exp \left[ \int_L f(x, y) du \right] \quad (1)$$

onde  $I$  representa a intensidade do feixe atenuado,  $I_0$  representa a intensidade do feixe original,  $f(x, y)$  os pontos sobre a integral de linha na direção  $u$ . Assim, o sinal observado será a seguinte relação logarítmica

$$p = \ln \left( \frac{I_0}{I} \right) \quad (2)$$

onde

$$p(s, \theta) = \int_L f(x, y) du, \quad -\infty < s < \infty, \quad 0 \leq \theta < \pi$$

O par  $(s, \theta)$  representa as coordenadas do raio relativas ao objeto,  $s$  representando o deslocamento da projeção e  $\theta$  o ângulo ao qual a projeção está sendo obtida. Deste modo,  $p(s, \theta)$  representa o valor assumido pela projeção nestas coordenadas.

De posse de várias projeções de um objeto, o problema da reconstrução tomográfica reside no fato de mapearmos  $p(s, \theta)$  para o espaço de imagem, de forma a obtermos novamente  $f(x, y)$ .

A obtenção de  $p(s, \theta)$  é descrita matematicamente através da transformada de Radon da função  $f(x, y)$ , sendo definida como a integral de linha ao longo de uma linha inclinada a um certo ângulo  $\theta$  do eixo  $y$  e à distância  $s$  da origem (Equação (3))

$$p(s, \theta) = \mathfrak{R}f = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - s) dx dy \quad (3)$$

A quantidade  $p(s, \theta)$  é descrita também como sendo *raio-soma*, representando a soma dos  $f(x, y)$  ao longo do raio a uma distância  $s$  e ângulo  $\theta$ . Assim, a transformada de Radon mapeia o domínio espacial para o domínio de projeções. Cada ponto em  $(s, \theta)$  irá corresponder a uma linha reta em  $f(x, y)$ .

Se tomarmos as coordenadas polares do domínio espacial como sendo  $(r, \phi)$ , de acordo com as seguintes definições

$$x = r \cos \phi, \quad y = r \sin \phi \quad (4)$$

concluimos que a distância em relação à origem  $s$  como sendo (figura 1)

$$s = r \cos(\theta - \phi) \quad (5)$$

A equação (5) têm considerações fundamentais, mostrando que a relação entre o

domínio de projeções e domínio espacial é puramente geométrico, conceito que foi devidamente explorado para a construção da pseudo-rede neural.

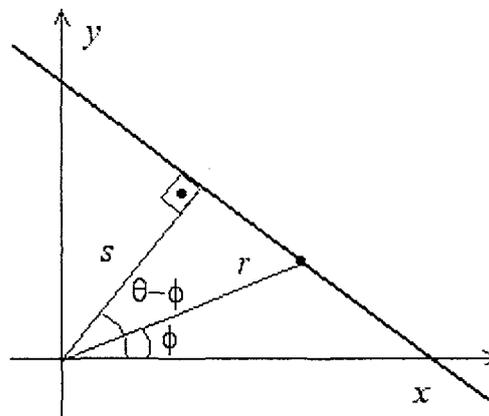


Figura 1: Relação entre o espaço de projeções  $(s, \theta)$  e o de imagem  $(x, y)$ .

A imagem tomográfica pode ser obtida com a reconstrução a partir das projeções. Existem vários métodos que permitem esta reconstrução. Baseando-se no algoritmo de retroprojeção filtrada desenvolvido por Ramachandran e Lakshminarayanan [5,6], a intenção é dividir o processo em duas redes:

1. Rede parcialmente conectada para filtragem.
2. Rede parcialmente conectada de retroprojeção (*backprojection*)

## 2.1 Rede de Retroprojeção

A seguir temos a fórmula de retroprojeção, na qual utilizamos como entrada as projeções filtradas da imagem de CT e processamos a transformação para a imagem final, expressa pela equação (6), na forma discreta [5].

$$f(x, y) = \sum_{i=1}^N q(i, \theta) (x \cos \theta + y \sin \theta) \quad (6)$$

onde  $q(i, \theta)$  é a imagem correspondente à projeção filtrada.

Na análise da equação (6), verifica-se que a projeção filtrada  $q(i, \theta)$  é multiplicada pelo valor paramétrico dado pelo *pixel* da imagem  $(x, y)$  em relação ao ângulo  $\theta$ , e assim somados para todos os ângulos das projeções. Ou seja, os

pontos de cada projeção filtrada  $q(i, \theta)$  segundo uma determinada relação contribuem para o valor do *pixel* no ponto  $(x, y)$ .

Uma rede neural [7] do tipo *feed-forward* totalmente conectada será construída conforme uma combinação linear expressa como na equação (7).

$$y_j = \sum_{i=0}^{N-1} w_{ij} x_i \quad (7)$$

Ligando cada saída  $y_j$  à entrada  $x_i$ , temos um peso equivalente  $w_{ij}$ . Esta rede terá  $M \times N$  conexões, e tais conexões transferem deste modo os valores de entrada para a saída.

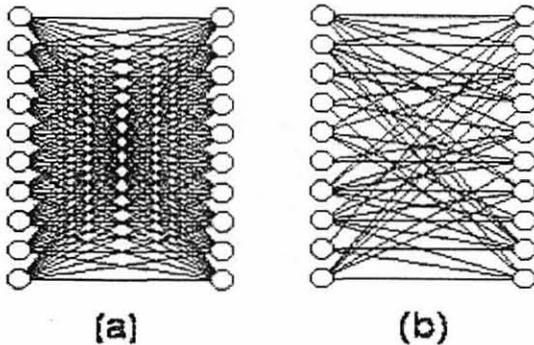


Figura 2: Ilustração de uma rede de duas camadas totalmente conectada (a) e parcialmente conectada (b).

Podemos fazer um paralelo entre as equações (6) e (7), para que possamos expressar o processo de retroprojeção como uma rede perceptron multicamada, ou seja,  $x_i$  para  $(x \cos \theta + y \sin \theta)$ ,  $w_{ij}$  para  $q(i, \theta)$  e  $y_j$  para  $f(x, y)$ .

Se ao invés de considerarmos coordenadas cartesianas para a imagem final e utilizarmos coordenadas polares  $(r, \phi)$ , e considerando que no espaço de projeções cada ponto pode ser representado pelo par  $(s, \theta)$ , sendo  $s$  a distância do ponto em relação à origem para cada ângulo de projeção  $\theta$ . A relação existente entre o espaço de projeções e o espaço de imagem pode ser representada pela equação (5) vista anteriormente.

Pela equação (5) vê-se que existem valores  $(s, \theta)$ , para todas as projeções, que contribuem para um determinado ponto de

imagem  $(r, \phi)$ , e que estão localizados ao longo de uma senóide sobre o espaço de projeções (ver figuras 3 e 4).



Figura 3: Senóides sobre o espaço de projeções filtradas.

Ainda pela equação (5), vemos que o processo pode ser bastante simplificado. A rede terá apenas pesos unitários e será parcialmente conectada, visto que somente os valores ao longo de uma senóide específica irão contribuir para um valor de *pixel* da imagem final. O processo de cálculo da rede torna-se então, após a montagem da mesma, um simples processo de somatório de valores de projeções.

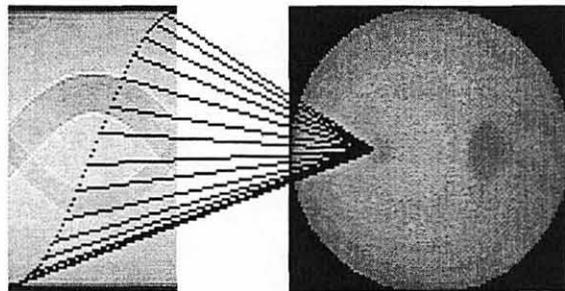


Figura 4: Contribuição dos pontos  $(s, \theta)$  sobre a senóide para o ponto  $(r, \phi)$  da imagem reconstruída.

Em termos de estrutura dos dados, o algoritmo para a rede de retroprojeção deve ser construído baseado no conceito de listas ligadas. Listas ligadas [8] são estruturas de dados nas quais temos elementos denominados *nós*, possuindo um campo para armazenar *informação* e outro campo (ou campos) para armazenar *endereçamento*. Este endereçamento refere-se a outros nós seguintes ou anteriores que contenham informação. Os nós de entrada e saída seriam estruturas alocadas em memória dinâmica, ligados por ponteiros de memória. Se

fôssemos utilizar *loops* do tipo *for-next*, teríamos que visitar todos os índices para efetuar o cálculo, sendo que com a lista ligada, apenas os pesos significativos seriam visitados.

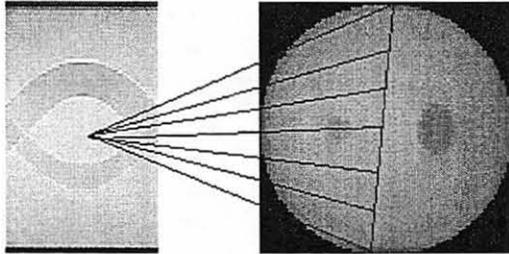


Figura 5: Contribuição dos pontos  $(r, \phi)$  da imagem final para o ponto  $(s, \theta)$  de uma certa projeção (relação inversa da eq.(5)).

Podemos considerar também como vantagem o fato de se trabalhar com coordenadas polares para a imagem final. Numa imagem de tamanho  $N \times N$ , aqueles pixels que estivessem com valor  $r > N$  não precisariam ser mostrados, visto que não teríamos conexões do espaço de projeções para o espaço de imagem.

Resumindo, a economia representada por este processo poderia ser descrita em 3 (três) pontos fortes:

- 1 - Os cálculos trigonométricos seriam substituídos por apenas somas
- 2 - Os valores dos pixels fora do domínio da imagem ( $r > N$ ) seriam desconsiderados para o cálculo
- 3 - Após a primeira construção em memória, não seria necessário a reconstrução, sendo assim vantajoso para seqüências de imagens ou projeções 3D.

Em termos de construção do algoritmo a nível de software, o mesmo deve ser baseado no conceito de listas ligadas, pois aproveitaríamos a saída da rede de filtragem para a entrada da rede de retroprojeção. Teríamos um conjunto de nós representando a entrada da rede como *pixels* da imagem formada pelas projeções, e um conjunto de saída representando os dados convoluídos também considerados como *pixels* de imagem. A informação dos nós seria o valor de  $p(k, \theta)$  do conjunto de entrada, e a saída seria o valor de  $q(n, \theta)$  referente ao conjunto de saída. Os nós dentro dos conjuntos seriam interconectados,

sendo guardados em cada nó o endereço ou posição do nó anterior e do nó posterior do intervalo discreto  $0 \leq k < N$ , para cada projeção discreta no intervalo  $0 \leq \theta < 2\pi$ .

## 2.2 Rede de Filtragem

Na construção da rede de filtragem, as projeções devem ser filtradas com o filtro Ram-Lak que possui a seguinte resposta em frequência  $H(w)$ :

$$H(w) = |w| \text{rect}(w) \quad (8)$$

Transformando para o domínio do espaço, temos sua resposta impulsional ou PSF (figura 6):

$$h(n) = \begin{cases} 1/4, & n = 0 \\ 0, & n \text{ par} \\ -\frac{1}{n^2 \pi^2}, & n \text{ ímpar} \end{cases} \quad (9)$$

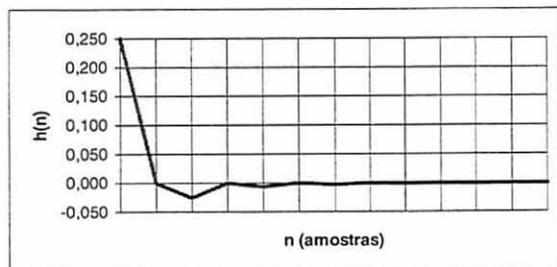


Figura 6: Gráfico da função de transferência  $h(n)$  (equação (9)), tendendo a zero para valores altos de  $n$ .

Analisando esta equação vemos que a função de transferência  $h(n)$ , diminui à medida que  $n$  aumenta, e os valores pares são zerados. Considera-se então, apenas um número finito de pontos, os  $N$  primeiros, como é o procedimento realizado num filtro FIR (*Finite Impulse Response*) por janelamento [9]. O sinal filtrado  $q(n, \theta)$  será dado pela convolução da entrada (projeção) com a resposta impulsional truncada:

$$q(n, \theta) = \sum_{k=0}^{N-1} h(n-k) p(k, \theta), \quad n=0, 1, \dots, N-1 \quad (10)$$

Podemos verificar que há uma correspondência de variáveis entre a equação (7) e (10) ( $x_i$  para  $p(k, \theta)$ ,  $w_{ij}$  para  $h(n-k)$ ,  $y_j$  para  $q(n, \theta)$ ).

Pela equação (10) podemos presumir que a rede é totalmente conectada. Porém analisando a equação de convolução em conjunto com a definição da função de transferência  $h(n)$ , vemos que não há necessidade de total conexão devido ao fato de que  $h(n)$  assume 0 (zero) para valores pares. Assim, uma rede parcialmente conectada com metade do número de pesos pode executar a tarefa de filtragem da matriz de projeções.

As conexões entre os conjuntos de entrada e de saída (ou pesos) também podem ser estruturadas como listas ligadas. Assim, a partir do nó (peso) inicial visitaríamos os outros nós por meio dos endereços contidos na estrutura do nó, até o último nó apontando para endereço nulo. Nesta visita a cada nó a informação seria transferida da estrutura de nós de entrada para os respectivos nós de saída. Como vimos anteriormente que não há necessidade de total conexão entre os nós de saída e entrada, os nós-pesos visitados seriam somente aqueles que iriam contribuir para o valor final, diminuindo o tempo de execução necessário.

Os nós de entrada e saída seriam então estruturas alocadas em memória dinâmica, ligados por ponteiros de memória. Se ao invés de estruturas dinâmicas fossem utilizados *loops* do tipo *for-next*, não haveria vantagens, pois seria obrigatório visitar todos os índices do intervalo considerado e testar se o valor está zerado ou não. Em contrapartida, a lista ligada dinamicamente já teria em si a construção de conexões entre os nodos de entrada e saída relevantes, e onde os  $w_{ij}$  assumissem 0 (zero) não haveria percurso na rede. Desta forma, podemos também na fase de filtragem das projeções utilizar o mesmo princípio explanado para a retroprojeção. Um aspecto a considerar é que, mesmo existindo no mercado algoritmos eficientes [10] para efetuar convolução que rodam baseados em DSP's (*Digital Signal Processor*), a grande vantagem em se implementar a filtragem com a rede neural vem da arquitetura paralela que esta estrutura proporciona, possibilitando implementações bastante rápidas a nível de *hardware* em chips neurais.

### 3. Resultados

Em testes feitos com algoritmos construídos com base em redes parcialmente conectadas, o aumento de performance foi satisfatório. O tempo de alimentação da rede de retroprojeção levou em média (em um microcomputador Pentium 450 MHz Dual) 95 milisegundos para uma imagem de CT de 60 projeções com 100 pontos, enquanto que o algoritmo tradicional no mesmo computador levou em média 440 milisegundos. O objeto na figura 7 representa, em níveis de cinza, a seção transversal de um tubo de alumínio com duas cavidades, reconstruído com as redes de retroprojeção filtrada, utilizando interpolação.

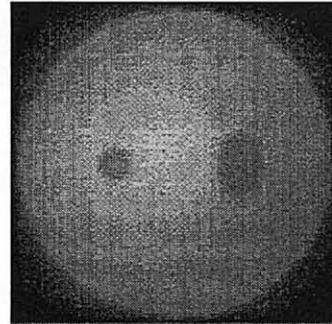


Figura 7: Exemplo de imagem reconstruída de um tubo de alumínio utilizando redes de filtragem e retroprojeção, com 60 projeções de 100 pontos, gerando uma imagem 100x100. (projeções gentilmente cedidas pela COPPE-RJ)

A partir do estudo de novos tipos de implementação, tais como algoritmos de processamento paralelo [11], podemos esperar que a performance de reconstrução das imagens seja aumentada, de forma a comportar a visualização de imagens tomográficas em 3D, em tempo real.

### 4. Conclusão

Apresentamos neste artigo a possibilidade de uso de redes neurais parcialmente conectadas, sem necessidade de treinamento, na reconstrução de imagens de CT. Tal implementação em hardware paralelo deve permitir a obtenção de imagens tomográficas com muito mais rapidez. Uma primeira avaliação feita num microcomputador com

apenas dois processadores já mostrou um ganho de 4.6 vezes no tempo de reconstrução.

Outro aspecto importante é que a relação entre os espaços de projeções e de imagem é puramente geométrico. Portanto uma mesma rede de retroprojeção pode reconstruir qualquer imagem que seja apresentada à entrada da rede, desde que sejam invariantes o tamanho da imagem e número de projeções utilizadas.

Em um segundo momento, planejamos fazer uso da interpolação para obter uma imagem final com maior resolução.. Neste caso, o número de pesos da rede será maior e eles não serão unitarios mas terão valores fracionários o que deve implicar num aumento do tempo de processamento.

## Referências

- [1] Rodrigues, R.G.S., Pelá, C.A. & Silva, A.C.R. *Reconstrução de imagens tomográficas utilizando redes neurais*. Anais do II Congresso Brasileiro de Redes Neurais. Curitiba-PR. (1995).
- [2] Munley, M.T. et al. *An artificial neural network approach to quantitative single photon emission computed tomographic reconstruction with collimator, attenuation and scatter compensation*. Med. Phys. 21 (12), 1889-1899. (1994).
- [3] Kerr, J. P. & Bartlett, E.B. *A statistical tailored neural network approach to tomographic image reconstruction*. Med. Phys. 22(5). (1995).
- [4] Rodrigues, R.G.S. *Desenvolvimento e Aplicação de um Algoritmo de Reconstrução Tomográfica com Base em Redes Neurais Artificiais*. Tese de doutorado, Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto-USP, Ribeirão Preto-SP. (2000).
- [5] Jain, A.K. *Fundamentals of Digital Image Processing*. Prentice-Hall. pp.431-475.(1989).
- [6] Herman, G.T. *Image Reconstruction from Projections*. Academic Press. pp.90-160. Orlando, Florida. (1980).
- [7] Haykin, S. *Neural Networks-A Comprehensive Foundation*. Macmillan Coll. Pub. Com. Inc. pp. 1-41. (1994).
- [8] Tenenbaum, A. M., Langsam, Y. & Augenstein, M. J. *Estruturas de dados usando C*. Makron Books. pp 223-302. (1995).
- [9] Oppenheim, A.V. & Schafer, R. W. *Discrete-Time Signal Processing*. Prentice-Hall. pp-31-79. (1989).
- [10] *TMS320C3x, User Guide*. Texas Instruments, Inc., Dallas, Texas. pp-11-58,11-61.(1996).
- [11] Rao, R. & Kriz, R. *Parallel Implementation of the Filtered Back Projection Algorithm for Tomographic Imaging*. Dep.of Ele.Eng. Virginia Polytechnic Inst. And State University. (1995).

## REFERÊNCIAS

- AZEVEDO, F. M., BRASIL, L. M. & OLIVEIRA, R. C. *Redes Neurais com Aplicações em Controle e em Sistemas Especialistas*. Florianópolis: Visual Books, 2000.
- BISHOP, C. *Neural Network for Pattern Recognition*. Oxford: Oxford University Press, 1995.
- BRACEWELL, R. N.; RIDDLE, A. C. Inversion of fanbeam scans in radioastronomy. *Astrophysics Journal*, v. 150, p. 427-434, 1967.
- CORMACK, A. M. Representation of a function by its line integrals, with some logical applications. *J. Appl. Physics*, v. 34, p. 2722-2727, 1963.
- CYBENKO, G. *Approximation by Superpositions of a Sigmoidal Function*. Urbana: University of Illinois, 1988.
- GONZALES, R. C. ; WOODS, R. E. *Processamento de Imagens Digitais*. São Paulo: Blucher, 2000.
- GORDON, R. A tutorial in ART (Algebraic Reconstruction Techniques). *IEEE Trans. Nuc. Sci.* NS-21, 78-93, 1974.
- HAYKIN, S. *Neural Networks-A Comprehensive Foundation*. Macmillan Coll. Pub. Com, 1994. .
- HERMANN, G. T. *Image reconstruction from projections*. London. Academic Press, Inc. 1980.
- \_\_\_\_\_, \_\_\_\_\_.; ROWLAND, S. W. Three methods for reconstructing objects from x-rays: a comparative study. *Comput. Graphics Image Processing*, v. 2, p. 151-178, 1973.
- \_\_\_\_\_, \_\_\_\_\_.; YAU, M. M. A comparative study of the use of linear and modified cubic spline interpolation for image reconstruction. *IEEE Trans. Nucl. Sci.* NS-36, pp 2879-2894, 1973.
- HOUSENFELD, G. N. A method and apparatus for examination of a body by radiation such a X or gamma radiation. *Patent Specification* 1283915. The Patent Office, London, 1972.
- JAIN, A.K. *Fundamentals of Digital Image Processing*. New York: Prentice-Hall, 1989

KACZMARZ, S. Angenährte Auflösung von Systemen linearer Gleichungen. *Bull. Int. Acad. Pol. Sci. Lett. A.*, p.355-357, 1937.

KAK, A.; SLANEY, M. *Principles of Computerized Tomographic Imaging*. New York: IEEE Press, 1988.

KERR, J. P.; BARTLETT, E.B. A statistical tailored neural network approach to tomographic image reconstruction. *Med. Phys.*, v. 22, n. 5, 1995.

LEPLEY, M.A.; FORKERT, R. DAWIC: Adaptive Wavelet Image Compression. *Mitre Technical Report*. MTR97B0000040, 1997.

MEDEIROS, L.F., SILVA, H.P.; PARENTE, E.R. (2001) Reconstrução de Imagens Tomográficas utilizando Redes Neurais Parcialmente Conectadas. In: CONGRESSO BRASILEIRO DE REDES NEURAI, 5., 2001, Rio de Janeiro, Anais... Rio de Janeiro, 2001. p.289-294.

MUNLEY, M.T. *et al.* An artificial neural network approach to quantitative single photon emission computed tomographic reconstruction with collimator, attenuation and scatter compensation. *Med. Phys.*, v. 21, n. 12, p. 1889-1899, 1994.

OPPENHEIM, B. E. (1977) Reconstruction tomography from incomplete projections In: POGOSSIAN, M. M. Ter *et al.* (Eds.). *Reconstruction Tomography in Diagnostic Radiology and Nuclear Medicine*. Baltimore, Maryland: Univ.Park Press, p. 155-183, 1977.

OPPENHEIM, A, V., SCHAFER, R, W. *Discrete-Time Signal Processing*, Prentice-Hall Inc., Englewood Cliffs, NJ, 1989.

PRESS, W. H. *et al.* *Numerical Recipes in C – The Art of Scientific Computing*. 2. ed. Cambridge.

RADON, J. Über die Bestimmung von Funktionen durch ihre Integralwerte langs gewisser Mannigfaltigkeiten. *Berichte Saechsische Akademie des Wissenschaften*, Leipzig, Math. Phys.Kl., v 69, p.262-277, 1917.

RAMACHANDRAN, G. N. & LAKSHIMINARAYANAN, A. V. Three-Dimensional reconstruction from radiographs and electron micrographs: application of convolutions instead of Fourier transforms. *Proc.Nat. Acad.Sci. U.S.*, v. 68, p. 2236-2240, 1971.

RAO, R.; KRIZ, R. *Parallel Implementation of the Filtered Back Projection Algorithm for Tomographic Imaging*. Dep.of Eletr. Eng. Virginia Polytechnic Inst. And State University, 1995.

RODRIGUES, R.G.S. *Desenvolvimento e Aplicação de um Algoritmo de Reconstrução Tomográfica com Base em Redes Neurais Artificiais*. Ribeirão Preto, 2000. Tese (Doutorado) - Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto-USP.

RODRIGUES, R.G.S., PELÁ, C.A.; SILVA, A.C.R. Reconstrução de imagens tomográficas utilizando redes neurais. In: CONGRESSO BRASILEIRO DE REDES NEURAS, 2., 1995, Curitiba. *Anais...* Curitiba-PR, 1995.

ROSENBLATT, F. The Perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, v. 65, n.1, p.386-408, 1958.

ROWLAND, S. W. Computer Implementation of image reconstruction formulas. In: HERMAN, G. T. *Image Reconstruction from Projections: Implementation and Applications*. Berlin: Springer, 1979.

ROSENFELD, A.; KAK, A. C. *Digital Picture Processing*. 2. ed. San Diego: Academic Press, San Diego, Califórnia, 1982. v.1.

RUSSEL, S.J.; NORVIG, P. *Artificial Intelligence: a Modern Approach*. New York: Prentice-Hall, 1995.

SAHINER, B.; YAGLE, A. E. Reconstruction from projections under time-frequency constraints. *IEEE Trans. On Med. Imag.*, v. 14, p. 193-204, 1998.

SANCHES, I. J. *Compressão sem perdas de projeções de tomografia computadorizada usando a transformada wavelet*. Curitiba, 2001. Dissertação de Mestrado. Universidade Federal do Paraná.

SHEPP, A. P.; LOGAN, B. F. The Fourier reconstruction of a head section. *IEEE Trans. Nucl. Sci.*, v. NS-21, p. 21-43, 1974.

SILVA, H. P. *Adaptação do Sistema Tomográfico Delta Scan-50 para Controle, Aquisição de Dados e Reconstrução de Imagens por PC Compatível*. Curitiba-PR Dissertação de Mestrado. Centro Federal de Educação Tecnológica do Paraná.

TENEMBAUM, A. M., LANGSAM, Y.; AUGENSTEIN, M. J. *Estruturas de dados usando C*. São Paulo: Makron Books, 1995.

ZIVIANI, N. *Projeto de Algoritmos com Implementações em Pascal e C*. 3. ed. São Paulo: Pioneira, 1996.