WAVELETS E REDES NEURAIS APLICADAS À ESTIMAÇÃO DE VOLUME DE TRÁFEGO DE VEÍCULOS

Dissertação apresentada ao curso de Mestrado em Engenharia Elétrica da Universidade Federal da Bahia, como requisito parcial à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Prof. Antonio Cézar de Castro Lima
Universidade Federal da Bahia

Angelo Amâncio Duarte

Salvador
Departamento de Engenharia Elétrica da UFBa
Fevereiro de 2001

Agradecimentos

Uma dissertação de mestrado não pode ser construída sem a colaboração valiosa de familiares, amigos e mestres.

Registro meu agradecimento ao meu orientador, Prof. Antonio Cezar de Castro Lima, pela tranquilidade e ânimo constante com que me recebeu nos momentos de maior dificuldade durante a realização da pesquisa, além do companheirismo e sábios conselhos antes e durante a realização do mestrado.

Ao Prof. Luciano Rila pelas sugestões e atenção que dedicou ao estudo do meu trabalho.

À Fundação Escola Politécnica pela bolsa recebida durante o tempo de estudos, que ajudou a minimizar o peso dos investimentos em bibliografia. Estendo também meu agradecimento a todos os professores do mestrado e funcionários do Departamento de Engenharia Elétrica da UFBA, com os quais tive uma produtiva convivência nos últimos dois anos e sem os quais não seria possível a existência do curso.

Finalmente, porém com maior intensidade, agradeço a minha esposa e filha que abriram mão de valiosas horas de convivência para que esse trabalho fosse concluído. Sem elas eu não teria o equilíbrio emocional necessário para faze-lo. A elas cabe, em grande monta, o mérito pela conclusão do meu curso de mestrado.

Salvador, fevereiro de 2001

Angelo Amâncio Duarte

Índice

Lista de Figuras

Lista de Tabelas

Resumo

Esse trabalho associa as ferramentas de Transformada Wavelet Discreta, Redes Neurais e Tratamento de Imagens (melhoria de contraste e detecção de bordas) com o objetivo de identificar corretamente o número de carros em imagens obtidas a partir de um mesmo ponto fixo. A Transformada Wavelet Discreta é utilizada com grande eficiência para compactação de imagens nesse trabalho reduzindo a quantidade de pontos que representam uma imagem sem degradar a informação do número de carros que essa contem. A detecção de bordas converte a imagem original em uma imagem que apresenta apenas as bordas dos objetos nela contidos, já que experimentos realizados com seres humanos e outros animais mostraram que bordas são as mais importantes pistas para interpretar imagens. As redes neurais artificiais, por serem inspiradas no neurônio biológico, possuem características muito úteis no reconhecimento de padrões. Essa associação de ferramentas busca facilitar ou possibilitar a detecção do número de carros em imagens, informação relevante em diversas aplicações, tal como, por exemplo, a determinação do tempo de atuação de semáforos como resultado da comparação do fluxo de suas diversas vertentes de tráfego. Será demonstrado que a associação das três técnicas é bem sucedida e que a Transformada Wavelet Discreta viabiliza o treinamento mais eficiente da Rede Neural pela substancial redução do número de pontos necessários para representar a informação.

Abstract

This works associates Discret Wavelet Transform, Image Analysis and Artificial Neural Network to identify the number of cars in a image sequence taken from a fixed point. The Discret Wavelet Transform is used to compact the images with no degradation over the information of numbers cars in them. Edge detection is used to reduce the useless information by extracting only the edges of the scenes, once it has been demonstrated that edges is the main information used by animals and humans in image interpretation. Artificial Neural Networks, by inspiration in biological neuron, have very useful characteristics for pattern recognition. This association of tools aims to facilitate or to allow the correctly identification of the numbers of cars in a sequence of images, that is ver relevant in many applications like, for example, the specification of duty times in traffic lights. It will be demonstrated that the association of the three techniques is well succeed and that the Discret Wavelet Transform allows an improvement in the training of the Neural Net, as a consequence of a strong reduction in the number of points required to represent the information.

1

Introdução

A crescente preocupação com a qualidade de vida nas cidades tem fomentado o estudo dos meios de transporte e de seus impactos sobre o cotidiano. Nesse sentido, é notória a influência que o aumento do número de veículos nas cidades que, associado a um incipiente investimento na infra-estrutura e a uma insuficiente oferta de serviços de transporte público, vem se tornando uma das principais causas na deterioração do meio-ambiente e no aumento do tempo de deslocamento dentro das cidades.

Os sistemas de controle de tráfego têm sido alvo de inúmeras pesquisas e hoje já se valem de modernas técnicas de controle no sentido de automatizar e otimizar o controle do fluxo de veículos via semáforos. Partindo-se desse pressuposto, nota-se que a adaptabilidade dos tempos de operação dos semáforos a partir das características do tráfego torna-se um poderoso instrumento na implementação de estratégias de controle.

A acertada implementação desse controle passará por uma correta aquisição das características do tráfego, geralmente expressas através do número de veículos por unidade de tempo, também chamado de volume de tráfego.

Existem muitas maneiras de se conseguir medir o volume de tráfego, sendo que a grande maioria delas utiliza algum tipo de sensor que ou têm contato físico com o veículo (sensores colocados no asfalto) ou precisa ser instalado em uma posição que se torna inconveniente devido ao fluxo de pedestres, o que o deixa vulnerável a vandalismos.

Uma solução que possa fazer a medição do tráfego a partir da posição de um observador que esteja posicionado na altura do semáforo traria como vantagens sua maior invulnerabilidade contra vandalismos e a dispensa do contato físico com o veículo. Essa idéia já foi objeto de estudo anterior e se mostrou viável para um certo universo de aplicações.

A motivação desse estudo é desenvolver um método capaz de estimar o fluxo de veículos e que, quando comparado ao método anterior, possa atingir um dos, ou ambos os, objetivos listados a seguir:

a) diminuir o custo computacional;

b) diminuir o erro de estimação.

Para isso o presente trabalho é faz uma associação de três técnicas matemáticas para implementar um método de estimação do fluxo de veículos em uma rodovia a partir de fotogramas dessa rodovia, obtidos através de uma câmera filmadora posicionada de forma a simular um observador.

As imagens obtidas serão processadas visando adequa-las, sem deterioração da informação do número de carros presentes, a um sistema de Inteligência Artificial baseado numa Rede Neural Artificial que se incumbirá da estimação do número de veículos. Esse processamento se dará por métodos de equalização de contraste e detecção de bordas.

O grande avanço introduzido pelo método que será descrito reside na introdução do processamento da imagens através da Transformada Wavelet, que se mostra extremamente adequada para uma compactação das imagens sem perda da informação do número de veículos presentes na mesma.

Esse texto está dividido em sete capítulos.

O capítulo 1 faz uma introdução sobre o tema que será relatado.

O capítulo 2 trata da teoria do processamento de sinais, começando com uma breve descrição da Transforma de Fourier e da Transformada de Fourier de Tempo Curto, para em seguida introduzir a Transformada Wavelet em 1 e 2 dimensões.

O capítulo 3 fala das técnicas de equalização de imagens e dos métodos mais comuns para detecção de bordas.

O capítulo 4 fala das Redes Neurais Artificias enfocando o modelo que será usado nesse trabalho.

O capítulo 5 comenta os procedimentos e materiais usados durante a pesquisa.

O capítulo 6 apresenta os resultados obtidos comparando-os ao método anterior de referência.

O capítulo 7 tece considerações sobre os resultados e apresenta novos estudos que poderão evoluir a partir dos resultados obtidos.

Os apêndices apresentam listagens dos programas que foram utilizados para a obtenção dos resultados apresentados.

2

Wavelets

2.1 Análise de Frequência

Desde o início do século XIX, quando o matemático francês Joseph Fourier desenvolveu sua teoria de análise de frequências, sabe-se que uma função periódica pode ser representada pela soma infinita de funções periódicas exponenciais complexas.

$$X(\omega) = FT[x(t)] = \int_{-\infty}^{+\infty} x(t).e^{-jwt}\,dt \qquad \textbf{(Error! Style not defined.-1)}$$

$$x(t) = IFT[X(\omega)] = \int_{-\infty}^{+\infty} X(\omega).e^{jwt}\,d\omega \qquad \textbf{(Error! Style not defined.-2)}$$

A equação (2.1) é conhecida como equação de análise de Fourier, Transformada de Fourier ou simplesmente FT (*Fourier Transform*), e a equação (2-2) é conhecida como equação de síntese de Fourier, Transformada Inversa de Fourier ou simplesmente IFT (*Inverse Fourier Transform*). Ambas equações consideram a função $x(t)$ periódica e contínua no tempo.

O advento dos computadores e do cálculo numérico estimulou o estudo dos sinais discretos no tempo ou, por outro lado, dos sinais contínuos no tempo transformados em sinais discretos no tempo conforme o teorema da amostragem [Cand88]. Os sinais discretos no tempo devem ser tratados de acordo com suas peculiaridades, e muitas das ferramentas matemáticas utilizadas para sinais contínuos no tempo foram adaptadas para atender à essas exigências. Esse foi também o caso da Transformada de Fourier que provou-se válida também para o caso discreto, passando a se chamar na análise como Transformada Discreta de Fourier ou DFT (*Discrete Fourier Transform*) e, na síntese, Transformada Inversa Discreta de Fourier ou IDFT (*Inverse Discrete Fourier Transform*), conforme descrito nas seguintes equações:

$$X(\Omega_m) = DFT[x(t)] = \sum_{t=0}^{M-1} x(t).e^{-j\Omega_m t} \qquad \textbf{(Error! Style not defined.-3)}$$

$$x(t) = IDFT[\mathrm{X}(\Omega_m)] = \frac{1}{M}\sum_{t=0}^{M-1}\mathrm{X}(\Omega_m).e^{j\Omega_m t} \quad \textbf{(Error! Style not defined.-4)}$$

em que

$$x(t) = x(t+kM), \ \forall k \quad \textbf{(Error! Style not defined.-5)}$$

é um sinal discreto periódico de período M e

$$\Omega_m = \frac{2\pi m}{M} \quad \textbf{(Error! Style not defined.-6)}$$

é a frequência discreta, sendo m o números de amostras do sinal no período M.

A DFT passou a ser amplamente utilizada no processamento digital de sinais quando em 1965 os matemáticos Cooley e Tukey apresentaram um algoritmo rápido para o cálculo da DFT, que ficou conhecido como FFT (*Fast Fourier Transform*).

A análise de frequência tanto para sinais de tempo contínuo quanto de tempo discreto permite estudar quais as frequências que estão presentes em um sinal periódico. Entretanto, a maioria dos sinais naturais não segue esse padrão. Ao contrário, os sinais que são encontrados na natureza são, em sua grande maioria, sinais descontínuos e/ou aperiódicos ou mesmo sinais periódicos que apresentam alguma forma de descontinuidade ou perturbação. Nestes casos a informação do instante de tempo em que ocorre a singularidade é de tanta ou mais valia do que a informação sobre sua frequência, e é aqui que a Transformada de Fourier passa a apresentar sua principal desvantagem: a perda da informação de tempo dos eventos presentes no sinal.

Para ilustrar esse fato considere-se o seguinte sinal composto por quatro ondas senoidais puras com frequências de 20, 70, 120 e 200Hz, descrito pela equação

$$x_1(t) = sin(2\pi 20t) + sin(2\pi 70t) + sin(2\pi 120t) + sin(2\pi 200t) \quad \textbf{(Error! Style not defined.-7)}$$

Sendo que o sinal $x_1(t)$ foi amostrado com uma frequência de 1KHz e que um trecho com 512 amostras (aproximadamente 0,5s) foi armazenado. Esse trecho pode ser visualizado na Figura 2.1.

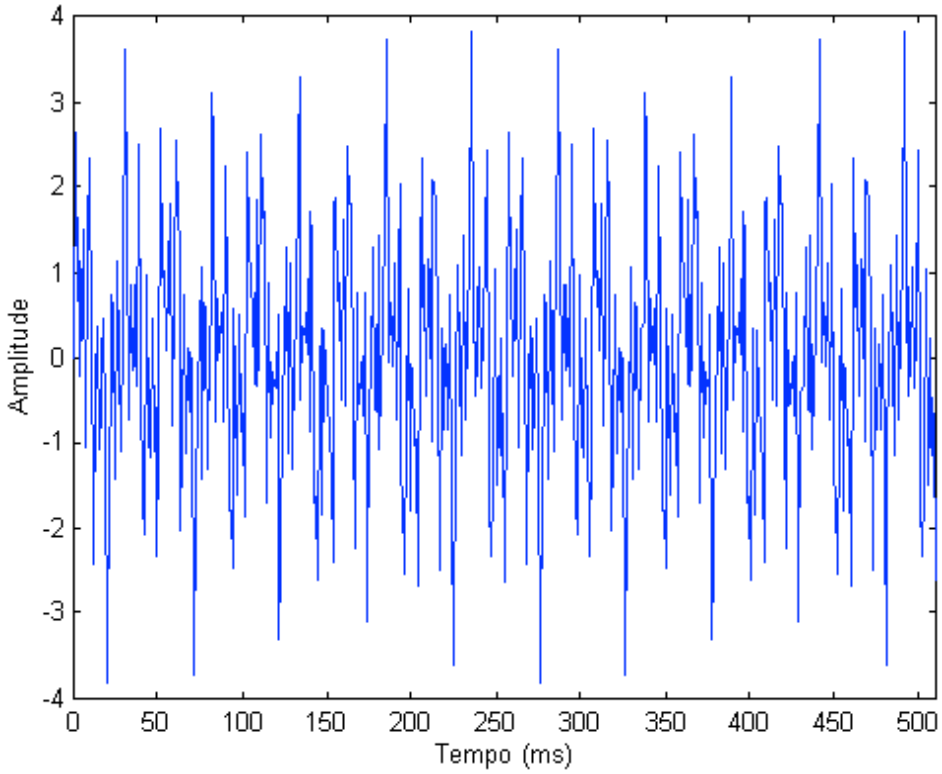


Figura **Error! Style not defined.**.1: Sinal $x_1(t)$ obtido pela superposição de 4 senóides com frequências de 20, 70, 120 e 200Hz

Aplicando-se a DFT sobre o sinal $x_1(t)$ obtemos a sequência $X_1(\Omega_m)$ e, a partir desta, pode-se analisar quais as frequências presentes no sinal $x_1(t)$. Essa análise é feita traçando um gráfico que relaciona a potência média do sinal por unidade de banda. Esse gráfico é conhecido como Densidade Espectral de Potência ou PSD (*Power Spectrum Density*) e é calculado através da seguinte expressão:

$$PSD(\Omega_m) = \frac{|X(\Omega_m)|^2}{N^2} \qquad \textbf{(Error! Style not defined.-8)}$$

A PSD para o sinal $x_1(t)$ pode ser vista na Figura 2.1-2.

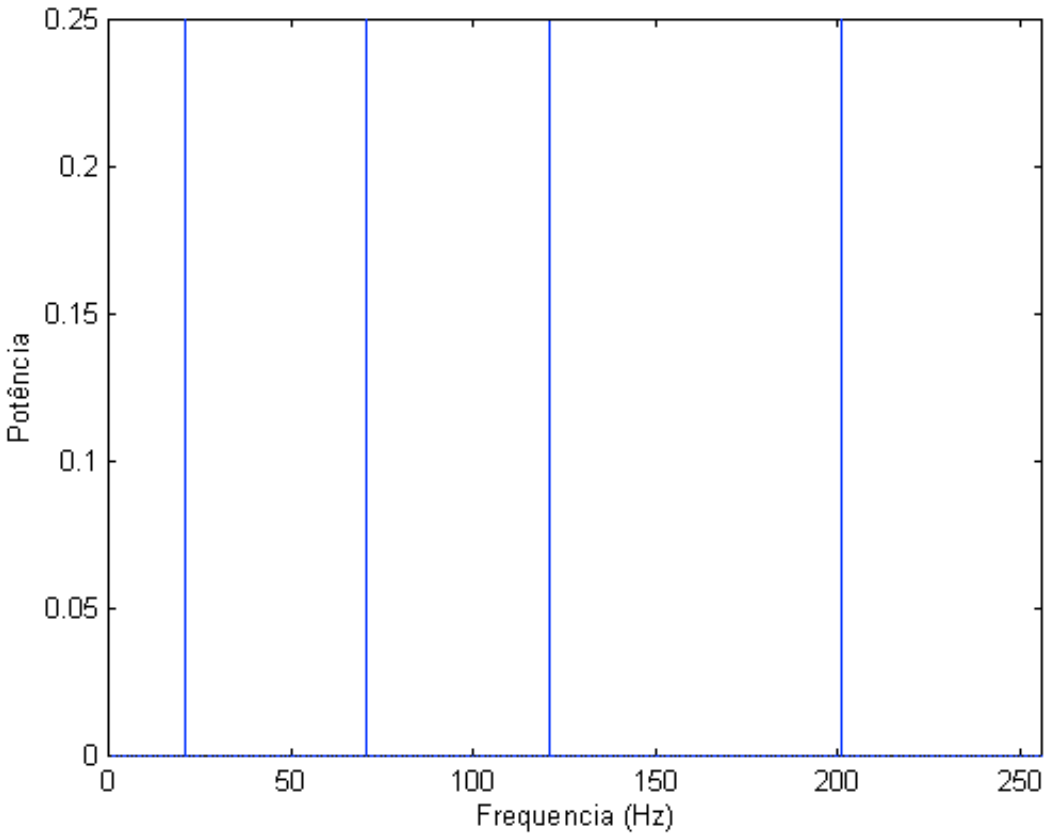


Figura **Error! Style not defined.**.2: PSD do sinal $x_1(t)$ obtido pela superposição de 4 senóides com frequências de 20, 70, 120 e 200Hz

Observa-se facilmente a presença das quatro frequências que compõem o sinal $x_1(t)$, bem como suas potências relativas que, nesse caso, são iguais.

Seja agora um outro sinal montado a partir das mesmas quatro componentes de 20, 70, 120 e 200Hz, só que não mais pela superposição das mesmas e sim pela sua concatenação, ou seja, outro sinal discreto $x_2(t)$ também com 512 amostras feitas a uma taxa de 1KHz e composto pela seguinte equação:

$$x_2(t) = \begin{cases} sin(2\pi 200t), 0 \le t < 128 \\ sin(2\pi 120t), 128 \le t < 256 \\ sin(2\pi 70t), 256 \le t < 384 \\ sin(2\pi 20t), t \ge 384 \end{cases} \quad \text{(Error! Style not defined.-9)}$$

Um trecho desse sinal está mostrado na Figura 2-3.

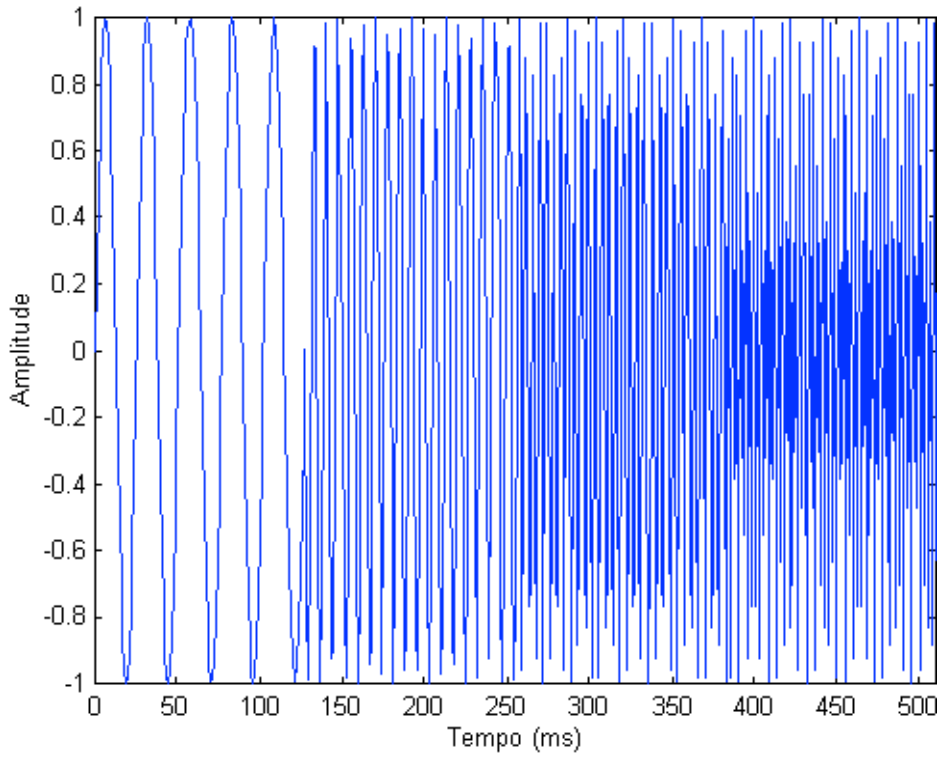


Figura **Error! Style not defined.**.3: Sinal $x_2(t)$ obtido pela concatenação de quatro senóides com frequências de 20, 70, 120 e 200Hz

Agora aplicando-se a DFT sobre esse sinal e calculando-se a sua PSD obtém-se o resultado mostrado na Figura 2-4.

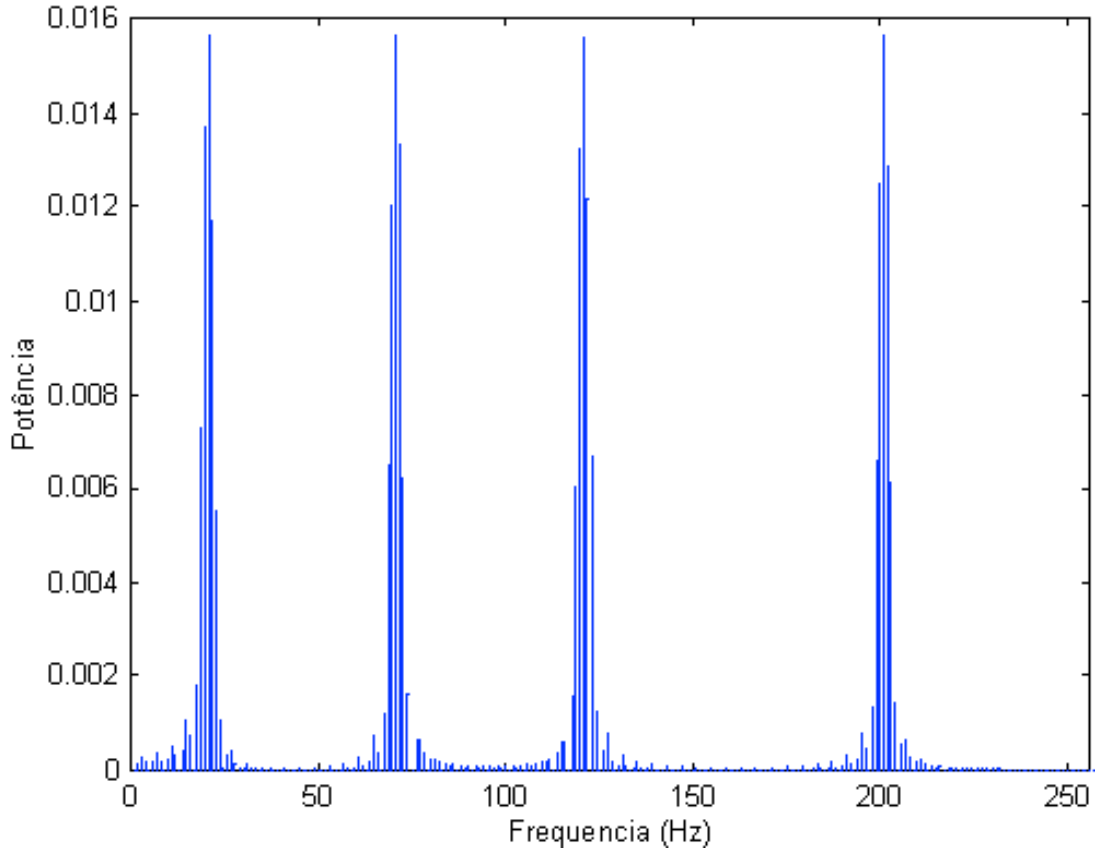


Figura **Error! Style not defined.**.4: PSD do sinal $x_2(t)$ obtido pela concatenação de quatro senóides com freqüências de 20, 70, 120 e 200Hz

Analisando o resultado obtido na Figura 2-4 pode-se perceber claramente que ainda existem quatro componentes principais centradas nas frequências de 20, 70, 120 e 200Hz. Porém, observa-se que agora há um certo espalhamento da energia ao longo de toda a faixa. Esse espalhamento é provocado pela subta mudança de uma frequência para outra no sinal original, o que causa uma descontinuidade nesses pontos. Essa descontinuidade provoca um espalhamento da energia ao longo do espectro de frequências, que aparece sob a forma de um "ruído" no gráfico.

Comparando-se de uma maneira mais grosseira os dois espectros apresentados nas Figuras 2-2 e 2-4, pode-se dizer que ambos apresentam a mesma aparência geral, isto é, ambos apontam para sinais que apresentam quatro componentes principais de frequências, as quais estão localizadas em 20, 70, 120 e 200Hz. Concluí-se com isso que a transformada de Fourier é incapaz de nos permitir uma clara diferenciação entre esses dois sinais bastante diferentes no domínio do tempo, uma vez que apresenta um resultado muito parecido para ambos no domínio da frequência. Resumindo, pode-se mesmo afirmar que a transformada de Fourier não é adequada para a análise de sinais com características não-estacionárias (sinais cujas componentes de frequência se alteram ao longo do tempo), a menos que não se esteja interessado na informação do tempo em que as frequências aparecem num sinal mas apenas nos valores dessas frequências.

2.2 Análise Tempo-Frequência

A perda da informação de tempo na transformada de Fourier gerou a necessidade de uma nova estratégia para a análise de sinais não-estacionários. Nessa estratégia assumisse que o sinal é estacionário em determinadas partes, e passasse a estudar essas partes individualmente.

Isso pode ser visto, por exemplo, na Figura 2-3, onde o sinal tem quatro porções de tempo onde apresenta um caráter estacionário. Aplicando-se uma janela sobre esse sinal, de forma a que sejam separadas cada uma de suas porções estacionárias, a DFT de cada porção obterá exatamente a frequência presente em cada uma. Essa estratégia provou-se adequada para diversas aplicações, dando origem a uma "revisão" da transformada de Fourier que passou a ser conhecida com Transformada de Fourier de Tempo Curto ou STFT (*Short Time Fourier Transform*).

A STFT corresponde à divisão do sinal em porções pequenas o suficiente para que dentro dessas porções o sinal possa ser tratado como um sinal estacionário. Essa divisão é

obtida pela aplicação de uma janela W ao sinal, com uma largura tal que dentro dela as porções do sinal sejam estacionárias. Matematicamente, essa operação pode ser descrita da seguinte forma:

$$\mathrm{X}(\tau,\Omega_m) = STFT[x(t)] = \sum_{t=0}^{M-1} x(t)\cdot w^*(t-\tau).e^{-j\Omega_m t} \quad \text{(Error! Style not defined.-10)}$$

em que $x(t)$ é o sinal original e $w^*(t)$ é o complexo conjugado da função janela escolhida. Essa operação pode ser também expressa no domínio da frequência pela convolução

$$\mathrm{X}(\tau,\Omega_m) = \mathrm{X}(\Omega_m) * W(\Omega_m) \quad \text{(Error! Style not defined.-11)}$$

Nota-se agora que a STFT é função de duas variáveis: tempo e frequência. Essa característica permite o estudo do sinal $x(t)$ levando em consideração não só as frequências que o compõem mas também o comportamento dessas frequências ao longo do tempo. Para ilustrar esse fato tome-se novamente o sinal $x_2(t)$ obtido pela equação (2-9) e aplique-se sobre ele a STFT usando uma janela gaussiana do tipo

$$w(t) = e^{\frac{-a.t^2}{2}} \quad \text{(Error! Style not defined.-12)}$$

em que a determina a largura da janela (quanto menor, mais larga) e t o tempo. O resultado obtido para a igual a 5000 pode ser visto na Figura 2-1.

Observa-se que o resultado apresenta claramente as quatro componentes de frequência do sinal. A diferença aqui, quando compara-se esse resultado com o resultado obtido pela DFT, é que tem-se também uma informação de tempo que mostra o instante em que as frequências ocorreram bem como sua duração. Vê-se agora que as frequências aparecem em instantes de tempo bem definidos. A título de comparação, analise-se agora o resultado da STFT quando aplicada sobre o sinal $x_1(t)$ obtido pela equação (2-7), utilizando-se os mesmos parâmetros para a janela $w(t)$.

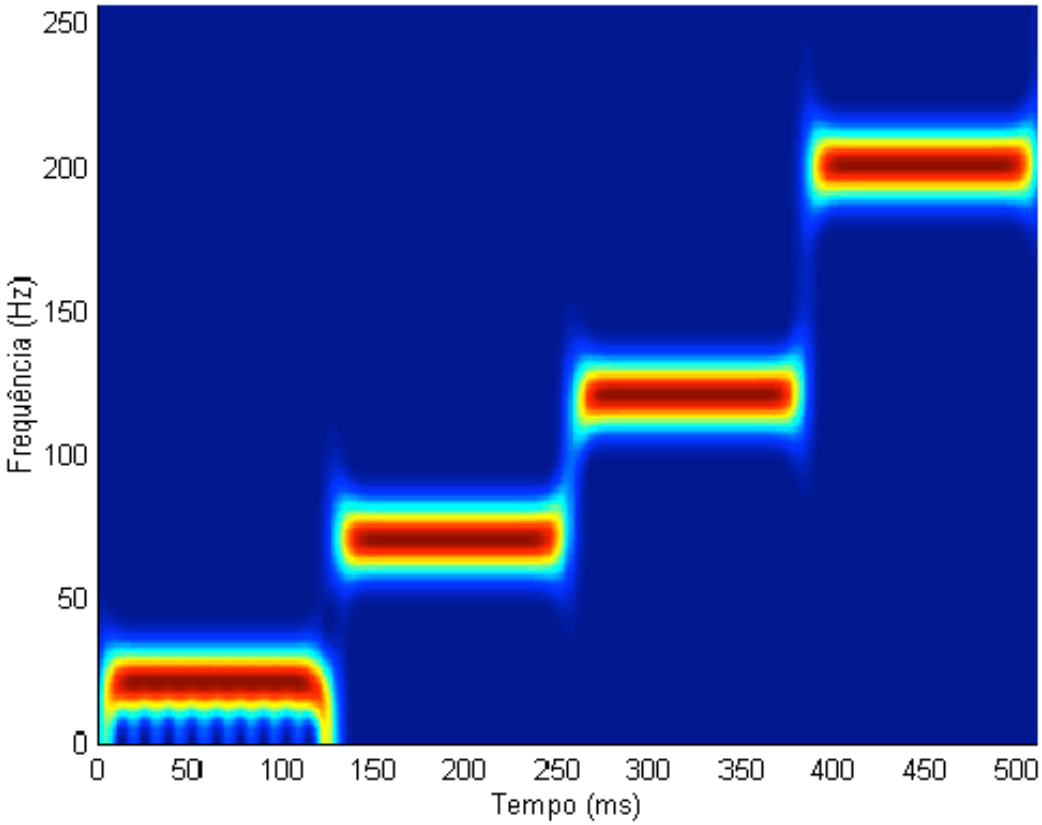


Figura **Error! Style not defined.**.5: PSD obtida pela STFT do sinal $x_2(t)$ obtido pela concatenação de quatro senóides com frequências de 20, 70, 120 e 200Hz (a = 5000)

O resultado, visto na Figura 2-5, mostra a existência novamente das quatro componentes de frequência e informa que as mesmas aparecem durante todo o tempo. Vê-se com isso que a STFT consegue diferenciar os dois sinais claramente, resolvendo o problema apresentado pela DFT.

Como toda nova abordagem, a nova análise tempo-frequência trouxe também novas questões ao universo da análise de sinais. O problema agora residia sobre qual a melhor configuração de janela que permitiria a melhor segmentação do sinal ou, mais precisamente, sobre qual a largura da janela que seria adequada para a obtenção dos melhores resultados. A prática mostrou que os resultados obtidos variavam significativamente com a mudança da largura da janela, que é chamada de suporte da janela. Parecia óbvio, entretanto, que janelas com suporte compacto, ou seja, janelas estreitas, conduziriam a resultados com melhor resolução já que separariam segmentos estreitos do sinal original, resultando em segmentos com fortes características estacionárias.

A obviedade dessa conclusão infelizmente não implicou em sua generalidade. De fato, como pode ser visto a seguir, janelas com alto suporte compacto provocam a perda da resolução temporal da STFT, que é a sua principal vantagem quando comparada à DFT.

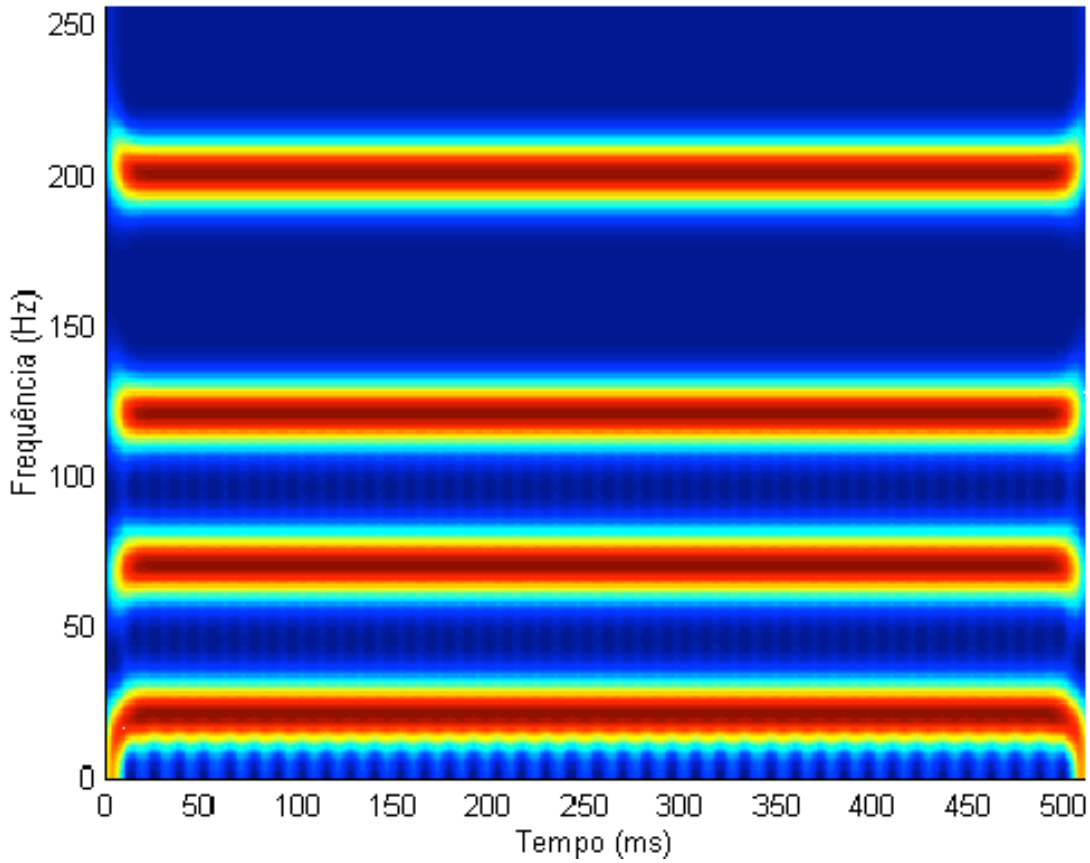


Figura **Error! Style not defined.**.6: PSD obtida pela STFT do sinal $x_1(t)$ obtido pela soma de quatro senóides com frequências de 20, 70, 120 e 200Hz (a = 5000)

Como exempo, considere-se mais uma vez o sinal $x_2(t)$ já definido na equação (2-9) e aplique-se sobre ele a STFT com janelas de largura diferentes (a = {10, 1000, 10000}). As novas PSDs obtidas, juntamente com a PSD já obtida para a igual a 5000 estão relacionadas na Figura 2.7. Atente-se nessa figura para uma significativa mudança no perfil das quatros densidades espectrais de potência encontradas.

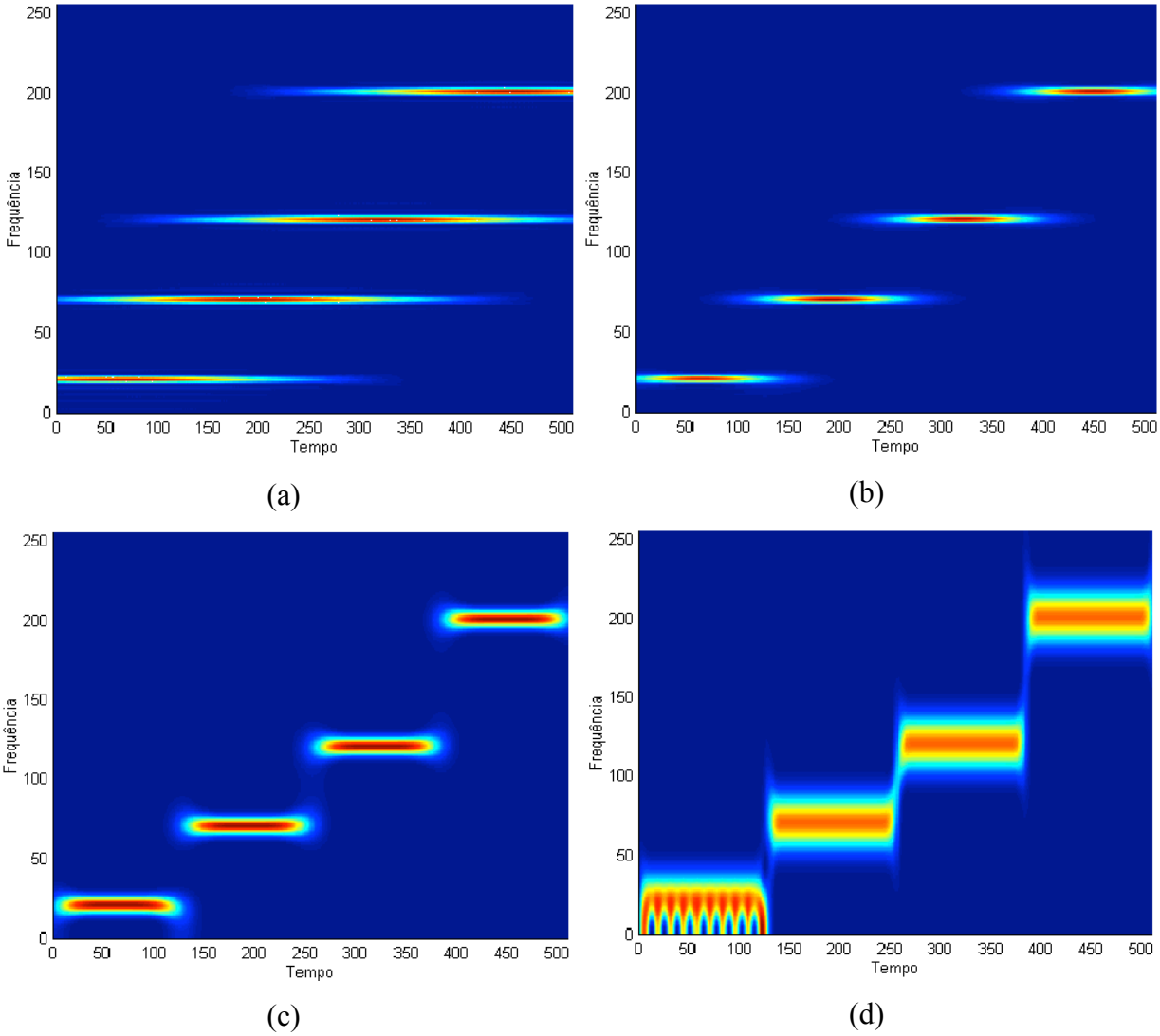


Figura **Error! Style not defined.**.7: PSDs com janelas de diferentes suportes, obtidas para o sinal $x_2(t)$. (a) $a = 10$; (b) $a = 100$; (c) $a = 1000$ e (d) $a = 10000$

É fácil perceber, comparando os quatro gráficos da Figura 2.7, que há um compromisso entre a resolução que obtem-se para o tempo e a frequência. Vê-se que nas Figuras 2.7.a e 2.7.b as raias são estreitas e bem separadas quando vistas no domínio da frequência, mas são largas e se justapõem quando vistas no domínio do tempo. De forma contrária, vê-se que nas Figuras 2.7.c e 2.7.d as raias são bem definidas no domínio do tempo mas se alargam no domínio da frequência.

A conclusão desse experimento é que há um compromisso entre a resolução no domínio da frequência e a resolução no domínio do tempo, não se pode alcançar ganho em uma sem perda na outra e vice-versa. Esse resultado é explicado por um princípio conhecido como Princípio da Incerteza de Heisenberg.

O Princípio da Incerteza foi originalmente desenvolvido por Heisenberg para justificar que não se pode saber simultaneamente o momento e a localização de uma partícula em movimento. Quando estendido para estudo dos sinais, esse princípio declara que não se pode saber a exata representação tempo-frequência de um sinal, ou seja, não se pode saber exatamente quais as componentes de frequência existem em um determinado instante de tempo. O que se pode saber são em quais intervalos de tempo certas bandas de frequências existem.

O Princípio da Incerteza no caso da análise de sinais via STFT está intimamente relacionado com a largura da janela $w(t)$. Se essa é muito estreita, visando obter-se uma boa resolução no tempo, perde-se resolução em frequência (Figuras 2.7.c e 2.7.d). Ao contrário, se a janela é muito larga ganha-se resolução em frequência mas perde-se resolução no tempo (Figuras 2.7.a e 2.7.b).

Analisando-se com mais cuidado a fórmula da STFT chega-se à interessante conclusão de que fazendo a janela $w(t)$ com largura infinita tem-se a máxima resolução em frequência possível e nenhuma resolução no tempo, ou seja, obtém-se $X(\Omega_m)$. Por outro lado, se fazendo-se a largura de $w(t)$ infinitesimalmente pequena, mas não nula, o resultado será simplesmente o máximo de resolução no tempo e nenhuma resolução na frequência, ou seja, obtém-se $x(t)$.

De todo o exposto acima vê-se que a escolha da função $w(t)$ mais adequada para a análise tempo-frequência de um sinal está longe de ser uma tarefa fácil ou, o que é pior, possível de ser auxiliada por alguma heurística. Devido a isso, foi necessário o desenvolvimento de uma outra forma de análise, a qual esta relatada a seguir.

2.3 Análise Multiresolucional

A alternativa encontrada para o problema da resolução na análise tempo-frequência foi o desenvolvimento de uma nova técnica em que as diferentes frequências contidas no sinal são analisadas em diferentes resoluções. A essa técnica deu-se o nome de análise multiresolucional ou MRA (Multiresolution Analysis).

A MRA foi desenvolvida para permitir uma análise com boa resolução no tempo e pouca resolução em frequência para altas frequências e, ao contrário, pouca resolução no tempo e alta resolução em frequência nas frequências baixas. Essa abordagem faz muito

sentido quando considera-se que a grande maioria dos sinais práticos apresentam componentes de alta frequência com curta duração e componentes de baixa frequência com longa duração. A questão principal na MRA é que as janelas agora não tem mais um tamanho fixo durante a análise mas sim um tamanho variável que se adapta conforme as necessidades de resolução de cada situação. Essas novas janelas passaram a ser representadas por uma classe de funções que receberam o nome de Wavelets.

2.3.1 Transformada Wavelet Contínua

De maneira geral as técnicas de análise de sinais consistem na decomposição de um sinal em uma combinação linear de funções base ou, em termos algébricos, na decomposição de um vetor do espaço vetorial do sinal em uma combinação linear de vetores base nesse ou em outro espaço vetorial que permita uma melhor compreensão das características do sinal. Essa operação consiste no somatório dos produtos dos vetores base do espaço vetorial por números escalares. A determinação desses escalares é chamada de análise do sinal e a reconstrução do sinal a partir dos escalares e dos vetores base é chamada de síntese.

Assim, seja o espaço de funções conhecido como $L^2(\mathbf{R})$, composto pelo conjunto de funções f(t) que atendem à seguinte condição:

$$\int |f(t)|^2 dt = E < \infty \qquad \text{(Error! Style not defined.-13)}$$

Esse espaço é denominado assim porque o L representa a integral de Lebesque, o número 2 indica o quadrado do módulo de $f(t)$ e o **R** indica que a variável independente t pode assumir qualquer valor no eixo real [Burr98].

O produto interno de duas funções $f(t)$ e $g(t)$ nesse espaço é o escalar a obtido pela expressão

$$a = \langle f(t), g(t) \rangle = \int f(t).g^*(t)dt \qquad \text{(Error! Style not defined.-14)}$$

e a norma de uma função $f(t)$ é definida por

$$\|f\| = \sqrt{|\langle f(t), f(t) \rangle|} \qquad \text{(Error! Style not defined.-15)}$$

As equações (2.14 e 2.15) são, na verdade, uma generalização das operações geométricas definidas no espaço Euclidiano tridimensional.

Duas funções de norma não nula são ditas ortogonais se seu produto interno for nulo, ou seja,

$$\langle \psi_k(t), \psi_l(t) \rangle = \int \psi_k(t) . \psi_l^*(t) dt = 0 \qquad k \neq l$$ (**Error! Style not defined.**-16)

e são ditas ortonormais se

$$\langle \psi_k(t), \psi_l(t) \rangle = \int \psi_k(t) . \psi_l^*(t) dt = \delta_{kl} \qquad k = l$$ (**Error! Style not defined.**-17)

em que δ_{kl} é a função delta de Kronecker definida por $\delta_{kl} = \begin{cases} 1 se k = l \\ 0 se k \neq l \end{cases}$

Um sinal ou função $f(t)$ pode ser escrito pela combinação linear

$$f(t) = \sum_l a_l \psi_l(t)$$ (**Error! Style not defined.**-18)

em que l é um índice inteiro para a soma (finita ou infinita), a_l são os coeficientes reais da expansão e $\psi_l(t)$ são um conjunto de funções reais chamado de conjunto de expansão.

Se a expansão é única diz-se que o conjunto de expansão é uma base para o espaço de funções. Se a base é ortonormal os coeficientes podem ser calculados pelo produto interno

$$a_k = \langle f(t), \psi_k(t) \rangle = \int f(t) \psi_k(t) dt$$ (**Error! Style not defined.**-19)

Por exemplo, na série de Fourier a base é formada pelo conjunto de expansão composto pelas funções $sen(k\omega_0 t)$ e $cos(k\omega_0 t)$ com freqüências $k\omega_0$. Na série de Taylor as funções são monômios t^k.

A Transformada Wavelet Contínua ou CWT (*Continuos Wavelet Transfom*), é definida por:

$$CWT\left[f, \psi^{a,b}\right] = \frac{1}{\sqrt{|a|}} \int f(t) . \psi^* \left(\frac{t-b}{a}\right) dt \quad \{a, b \in \Re; a \neq 0\}$$ (**Error! Style not defined.**-20)

$$f(t) = \frac{1}{C_\psi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{CWT\left[f, \psi^{a,b}\right]}{a^2} \psi^{a,b} da.db$$ (**Error! Style not defined.**-21)

A perfeita reconstrução de $f(t)$ a partir da equação de síntese 2.21 depende da constante $C\psi$ chamada constante de admissibilidade [Daub92], que é calculada pela seguinte fórmula:

$$C_\psi = 2\pi \int_{-\infty}^{\infty} \frac{\left|FT\left[\psi^{a,b}\right](\xi)\right|^2}{|\xi|} d\xi < \infty$$ (**Error! Style not defined.**-22)

Nota-se que a CWT é uma função de duas variáveis a e b conhecidas, respectivamente, como parâmetros de escala (largura da janela) e translação (posição da janela). A função $\psi(t)$ é a função de expansão chamada de wavelet mãe, ou seja, é a função modelo que será expandida e transladada.

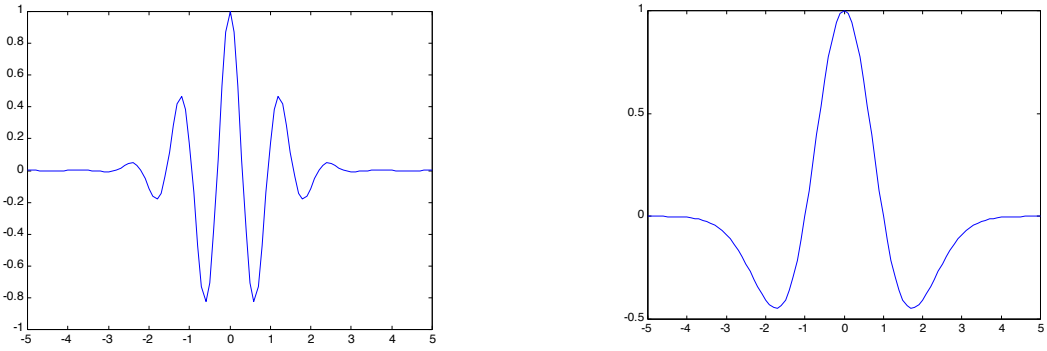
Existem diversas condições necessárias para que $\psi(t)$ seja considerada uma wavelet. Uma das mais importantes [Daub92] é que:

$$\int \psi(t)dt = 0 \qquad \textbf{(Error! Style not defined.-23)}$$

O termo wavelet é uma adaptação do termo francês ondelete usado pela primeira vez nos trabalhos de Yves Meyer, e quer dizer pequena onda. Esse termo surgiu do fato de que as funções $\psi(t)$ têm comprimento finito, ou seja, suporte compacto, e possuem caráter oscilatório (Figura 2.8.a e b).

As vantagens da utilização da CWT no lugar da STFT podem ser melhor avaliadas através de um exemplo. Analisando-se, através da CWT, o sinal $x_2(t)$ obtém-se o gráfico em duas dimensões mostrado na Figura 2.9. A análise foi feita utilizando uma wavelet Morlet (Figura 2.8.a), obtida pela expressão

$$\psi(t) = e^{\frac{-t^2}{2}} cos(5t) \qquad \textbf{(Error! Style not defined.-24)}$$



(a) (b)

Figura **Error! Style not defined.**.8: (a) Wavelet Morlet; (b) Wavelet Chapéu Mexicano

A análise multiresolucional trouxe um novo conjunto de informações que precisam ser analisados sob uma nova óptica. No exemplo citado (Figura 2.9) pode-se observar quatro regiões distintas que correspondem exatamente aos quatro segmentos de frequência do sinal $x_2(t)$. Cada uma dessas regiões tem a largura (tempo) equivalente à largura do segmento do sinal original. Simultaneamente, observa-se que essas quatro regiões apresentam alturas diferentes no gráfico quando observadas em relação ao eixo das escalas.

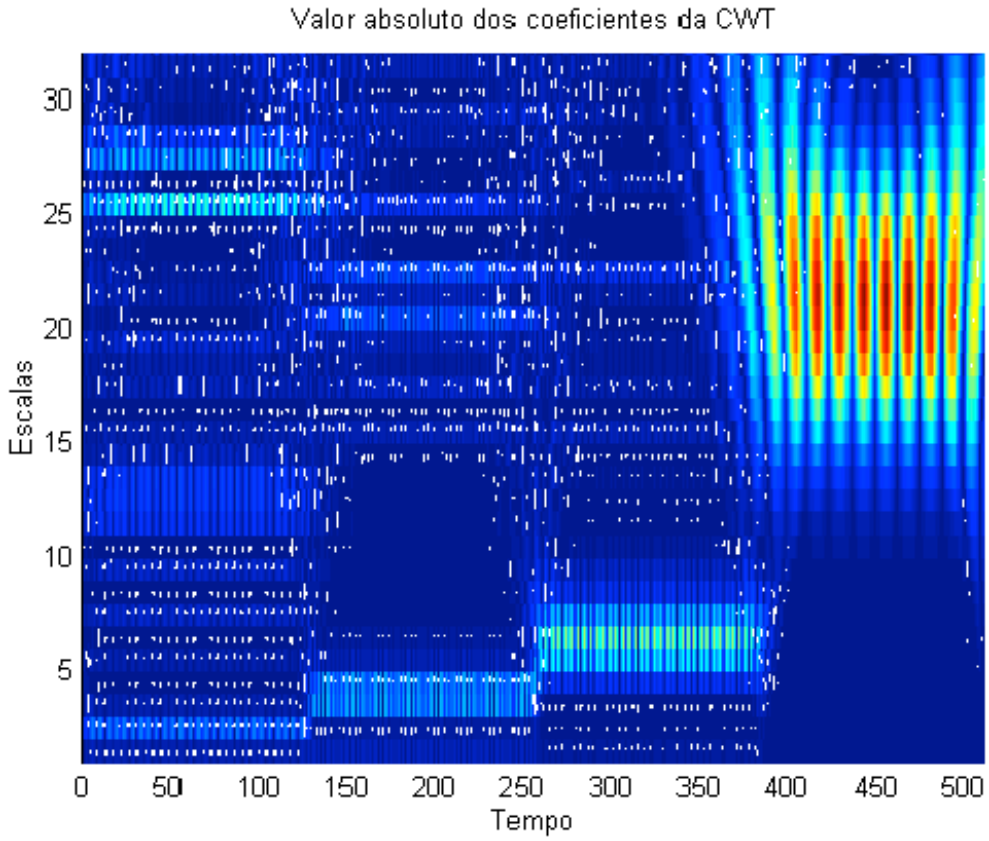


Figura **Error! Style not defined.**.9: Gráfico do valor absoluto dos coeficientes wavelet em 32 escalas, obtidos através da CWT do sinal $x_2(t)$ com a wavelet Morlet

O termo escala é usado na análise wavelet para designar o nível de detalhes com o qual se está observando o sinal ou, em outras palavras, qual a resolução de frequência que está sendo usada na análise. Pequenas escalas correspondem a altas frequências e grandes escalas a baixas frequências. Para facilitar o entendimento pode-se fazer uma analogia com o termo escala usado em topografia. Quando diz-se que um mapa está desenhado em grande escala significa que o mesmo apresenta uma visão global do terreno e, portanto, não apresenta seus detalhes, correspondendo apenas a informações de baixa frequência do relevo. Se, por outro lado, desenha-se um mapa em pequena escala apresenta-se as características do terreno com maior riqueza de detalhes, correspondendo às informações de alta frequência.

Embora exista uma relação entre a escala na análise Wavelet e a frequência na análise de Fourier ela está longe de ser uma regra geral já que a correspondência entre uma e outra é fortemente dependente da forma da wavelet escolhida. É comum a utilização do termo "duração" ($1/a$) no lugar de frequência, para evitar a confusão com os ciclos de repetição dos coeficentes wavelet [Lewa98].

Uma das relações frequentemente adotadas é a associação do pico do espectro dos coeficientes wavelet com a singularidade da FT de uma onda cossenoidal. Essa associação

pode ser alcançada computando a escala *a* do coeficiente wavelet máximo de uma onda cossenoidal conhecida e comparando-a com a frequência do espectro de Fourier dessa onda.

2.3.2 Transformada Wavelet Discreta

O cálculo dos coeficientes da CWT (ou da FT) não pode ser praticamente realizado em computadores por usar equações analíticas, integrais, etc. Assim, é necessário discretizar a CWT, o que é conseguido, da mesma forma que na FT, pela amostragem dos coeficientes. Esse procedimento da origem às séries wavelet, conhecidas como Transformada Wavelet Discreta, DWT - Discrete Waveet Transform. Entretanto, ao contrário da FT, a amostragem dos coeficientes da CWT não precisar ser realizada com uma taxa uniforme.

De fato, a mudança de resolução em cada escala permite utilizar taxas de amostragem variáveis de acordo com a escala em se está operando. Para escalas maiores (frequências baixas), a taxa de amostragem pode ser reduzida de acordo com o critério de Nyquist, o que permite a economia de quantidades razoáveis de memória.

Desse modo, o plano tempo-frequência pode ser amostrado a uma taxa de amostragem N_1 na escala a_1 e a uma taxa de amostragem N_2 na escala a_2. Se $a_1<a_2$ (correspondendo a frequências $f_1>f_2$) então $N_2<N_1$, atendendo à seguinte relação:

$$N_2 a_2 = N_1 a_1 \qquad \text{(Error! Style not defined.-25)}$$

ou ainda,

$$\frac{N_2}{f_2} = \frac{N_1}{f_1} \qquad \text{(Error! Style not defined.-26)}$$

A discretização é feita inicialmente sobre o parâmetro a (escala) numa grade logarítmica. Em seguida, o parâmetro *b* (deslocamento) é discretizado com respeito à escala *a*, ou seja, é usada uma taxa de amostragem diferente para cada escala.

A abordagem é muito semelhante à usada no tratamento de sinais pela FT onde os coeficientes são discretizados dando origem à série de Fourier ou FS (*Fourier Series*). A diferença é que agora a taxa de amostragem varia de acordo com a escala em que se está tratando o sinal.

Resumindo matematicamente esse procedimento, tem-se agora a equação da wavelet

$$\psi^{a,b}(t) = \frac{1}{\sqrt{|a|}}\psi\left(\frac{t-b}{a}\right) \qquad \text{(Error! Style not defined.-27)}$$

re-escrita em sua forma discreta [Daub92]

$$\psi^{j,k}(t) = a_0^{-j/2}\psi(a_0^{-j}t - kb_0) \qquad a_0 > 1; \quad b_0 > 0 \qquad \text{(Error! Style not defined.-28)}$$

A DWT viabiliza a computação da CWT, entretanto ela não pode ser considerada uma transformada discreta na total acepção da palavra. Isso porque a DWT é simplesmente a versão amostrada da CWT. Na verdade, a DWT é altamente redundante para viabilizar a reconstrução do sinal, e essa redundância gera um custo computacional bastante elevado.

Assim, uma nova abordagem foi desenvolvida considerando agora que o parâmetro tempo é de fato uma grandeza discreta. Cabe aqui uma observação quanto à nomenclatura usada para diferenciar a transformada wavelet discreta gerada a partir da amostragem da CWT e também conhecida como séries wavelet, da transformada wavelet de tempo discreto e que deveria se chamar DTWT (*Discrete Time Wavelet Transform*). Na literatura especializada convenciona-se chamar a DTWT de DWT, já que sua utilização é mais frequente [Burr98], enquanto que a DWT é geralmente conhecida como séries wavelet ou WS (*Wavelet Series*). Assumir-se-á doravante nesse trabalho o termo DWT para identificar a transformada wavelet discreta tal qual é largamente feito na literatura.

Os fundamentos da DWT remontam ao ano de 1976 com os trabalhos de Croiser, Esteban e Galand para o desenvolvimento de uma técnica para decomposição de sinais de tempo discreto. No mesmo ano foi desenvolvida uma técnica para codificação de sinais de voz chamada codificação de sub-bandas (*subband coding*), que foi aprimorada em 1983 dando lugar à codificação piramidal (*pyramidal coding*), que é também conhecida como análise multiresolucional. Mais tarde, Vetterli [Vett92] otimizou o esquema de codificação sub-banda, removendo as redundâncias no esquema de codificação piramidal.

2.3.3 Codificação Sub-Banda

A idéia por trás da codificação sub-banda é essencialmente a mesma da CWT. A CWT é a medida da correlação entre a wavelet em diferentes escalas e o sinal, com essa escala servindo de medida de similaridade, e é computada pela alteração da escala (suporte) de uma função janela, pelo deslocamento dessa janela no tempo, pela multiplicação dessa janela pelo sinal e, finalmente, pela integração do resultado em todo o intervalo de tempo.

No caso discreto (WS), o sinal é analisado em diferentes escalas por filtros com diferentes frequências de corte. Para as pequenas escalas o sinal é tratado por filtros passa-alta e para grande escalas por filtros passa-baixa. A quantidade de informação do sinal em cada

escala é alterada pelas operações de filtragem e a escala é alterada por operações de sub-amostragem ou sobre-amostragem. A sub-amostragem corresponde à remoção de amostras do sinal, correspondendo a uma redução da taxa de amostragem. A sobre-amostragem corresponde à inserção de novas amostras ao sinal, geralmente iguais a zero ou a um valor interpolado, correspondendo a um aumento da taxa de amostragem.

A escolha mais usual na obtenção dos coeficientes da WS a partir da CWT é fazer uma amostragem diádica dos coeficientes. Em outras palavras, escolhe-se a_0=2 e b_0=1 na equação (2.28) gerando a equação

$$\psi^{j,k}(t) = 2^{-j/2}\psi(2^{-j}t - k) \qquad \textbf{(Error! Style not defined.-29)}$$

que é uma base ortonormal para ao espaço $L^2(\Re)$ [Daub92].

Todo o esquema pode ser graficamente representado pela Figura 2.10.

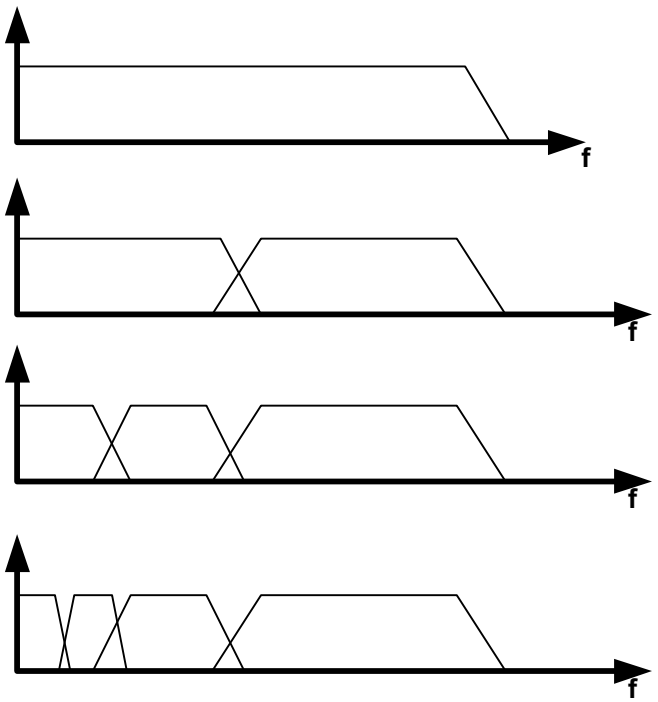


Figura **Error! Style not defined.**.10: Divisão do espectro de frequências através de um banco de filtros

Considere-se um sinal discreto no tempo representado por uma sequência $x[n]$, (n = 0,1,2,3,...). A filtragem desse sinal [Cand88] corresponde a operação matemática de convolução do mesmo com a função de resposta ao impulso $h[n]$ do filtro. Essa operação pode ser expressa pela seguinte equação

$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k].h[n-k] \qquad \textbf{(Error! Style not defined.-30)}$$

Considerando-se agora que o sinal discreto no tempo $x[n]$ foi obtido pela amostragem de um sinal contínuo $x(t)$ a uma frequencia Fs hertz., tem-se que a máxima frequência

possível no sinal contínuo terá que ser *Fs/2* hertz para atender ao teorema de Nyquist. Em termos de frequência radial, temos que *Fs* é definida como 2π radianos, resultando que a máxima frequência contida em $x[n]$ é π radianos.

Após fazer-se atravessar o sinal por um filtro passa-baixa com metade da banda do sinal original a maior frequência presente nesse sinal será de $\pi/2$ radianos. Sendo assim, pode-se excluir metade das amostras do sinal e ainda continuar-se-á a atender ao critério de Nyquist. Isso corresponde a uma sub-amostragem por um fator de 2, o que, na prática, significa uma redução da frequência de amostragem pela metade. Esse processo pode ser descrito pela seguinte equação:

$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k].h[2n-k] \quad \text{(Error! Style not defined.-31)}$$

Resumindo, vê-se que filtrando-se metade da banda do sinal através de um filtro passa-baixa reduz-se metade de suas frequências, o que pode ser interpretado como uma redução da quantidade de informações do sinal pela metade ou ainda, em outras palavras, uma redução pela metade em sua resolução. Ora, uma redução da resolução pela metade permite dobrar a escala em que se está analisando o sinal sem que haja perda de informação na análise, ou seja, pode-se extrair metade das amostras do sinal sem que isso afete a quantidade de informação presente no mesmo, o que corresponde a reduzir a frequência de amostragem pela metade.

Essa é a estratégia adotada no cálculo da DWT. A DWT analisa o sinal em diferentes bandas de frequência com diferentes resoluções pela decomposição do sinal em dois conjuntos: aproximação (baixas frequências) e detalhe (altas frequências). As aproximações são obtidas pela operação com filtros passa-baixa e os detalhes com filtros passa-alta, cada um com metade da banda do sinal original. A saída de cada filtro é, então, sub-amostrada por um fator de dois, que corresponde a eliminação de metade das amostras. Esse processo continua até que sobrem apenas duas amostras do sinal, e pode ser matematicamente descrito pelas seguintes equações

$$d[n] = x[n] * g[n] = \sum_{k=-\infty}^{\infty} x[k].g[2n-k] \quad \text{(Error! Style not defined.-32)}$$

$$a[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k].h[2n-k] \quad \text{(Error! Style not defined.-33)}$$

em que $d[n]$ e $a[n]$ são, respectivamente, as saídas do filtro passa-alta e passa-baixa sub-amostradas por um fator de 2. As sequências $d[n]$ e $a[n]$, respectivamente, são também conhecidos como detalhes e aproximações do sinal $x[n]$.

A Figura 2.11 ilustra o procedimento. Nela, $x[n]$ representa o sinal original e $g[n]$ e $h[n]$ as funções de transferência dos filtros passa-alta e passa-baixa respectivamente. Cada nível inferior apresenta metade das amostras do nível imediatamente superior a ele. As funções de transferência $h[n]$ e $g[n]$ são relacionadas através da expressão (2-34) e correspondem a filtros conhecidos como QMF (Quadrature Mirror Filter).

$$g[L-1-n] = (-1)^n h[n] \qquad \text{(Error! Style not defined.-34)}$$

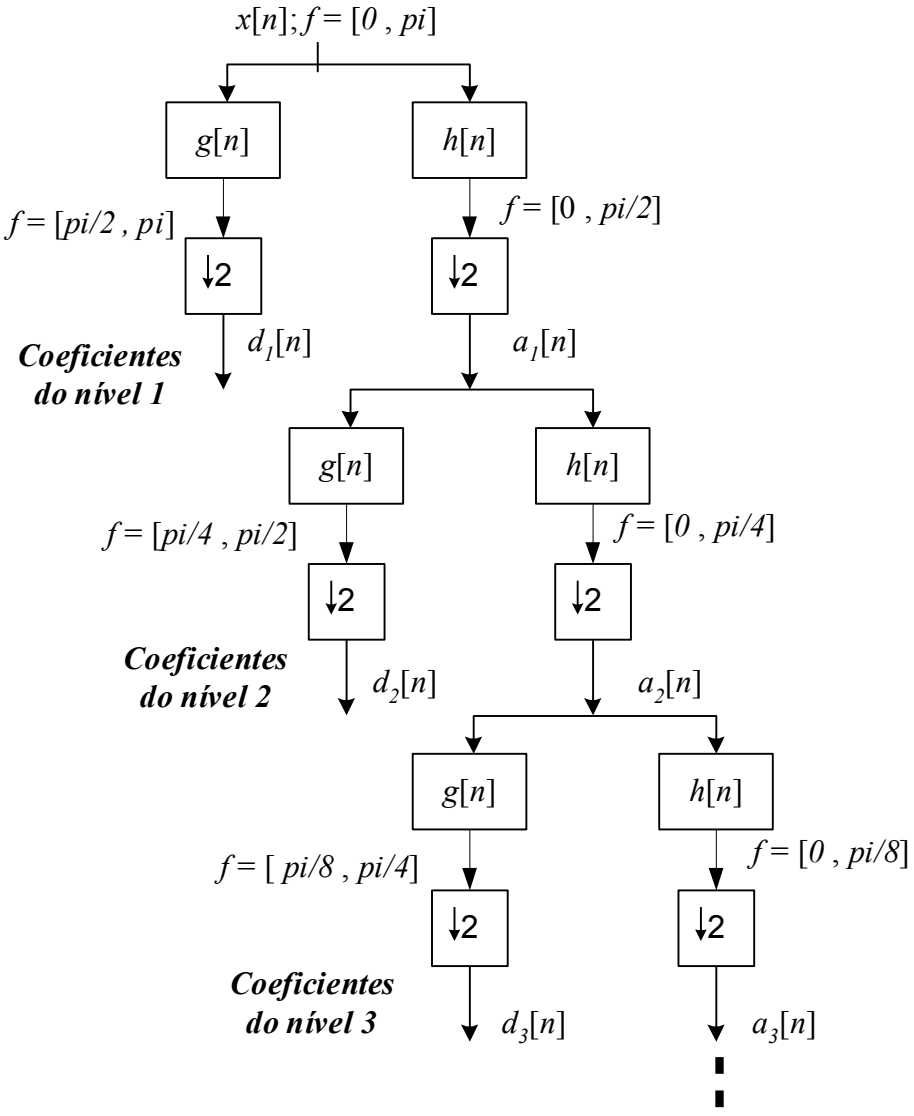


Figura **Error! Style not defined.**.11: Descrição do algoritmo de codificação sub-banda

Pode-se demonstrar [Burr98] que a síntese do sinal $x[n]$ pode ser obtida a partir dos coeficientes $d[n]$ e $a[n]$ utilizando um processo inverso ao da análise. Nesse caso, os sinais em cada nível sofrem uma superamostragem por um fator de 2 (inserção de mais amostras à sequência original), passam pelos filtros pass-alta e passa-baixa e então são somados. É interessante notar-se que as funções de transferencia $h^*[n]$ e $g^*[n]$ usadas na síntese são exatamente as mesmas dos filtros usados na análise, exceto por uma inversão das sequências no tempo. Dessa forma, a equação de síntese pode ser escrita por

$$x[n] = \sum_{n=-\infty}^{\infty} \left(a[n].h[2k-n] + d[n].g[2k-n] \right) \qquad \textbf{(Error! Style not defined.-35)}$$

A perfeita reconstrução do sinal x[n] não pode ser conseguida se os filtros não possuírem uma banda ideal. Entretanto, é possível obter-se filtros que, sob certas condições, permitam uma reconstrução perfeita. Os mais famosos foram obtidos por Ingrid Daubechies [Daub92] e deram origem às waveletes de Daubechies.

Um outro fator importante a observar é que a operação de sub-amostragem é feita em múltiplos de 2 e isso gera a necessidade de que o comprimento da sequência x[n] seja uma potência inteira de 2. Simultaneamente, o comprimento do sinal determina o máximo número de níveis de decomposição possível, conforme a fórmula

$$\textit{número de níveis de decomposição} = \mathbf{log_2}(\textit{número de amostras}) \qquad \textbf{(Error! Style not defined.-36)}$$

A interpretação dos coeficientes resultantes da DWT requer uma nova abordagem já que sua apresentação não segue o mesmo padrão apresentado na análise de frequências. Uma das formas mais usuais é a apresentação dos coeficientes obtidos em cada nível da análise em um gráfico sequencial em que as escalas decrescem a partir da origem. A Figura 2.12 apresenta um exemplo. Note-se como não há uma interpretação elementar para cada caso, principalmente pelo fato de que a DWT não destrói a informação de tempo tal qual o faz a DFT. A habilidade em extrair melhores informações da DWT é dependente da experiência do operador não só na transformada wavelet propriamente dita mas também no sistema que está sendo analisado.

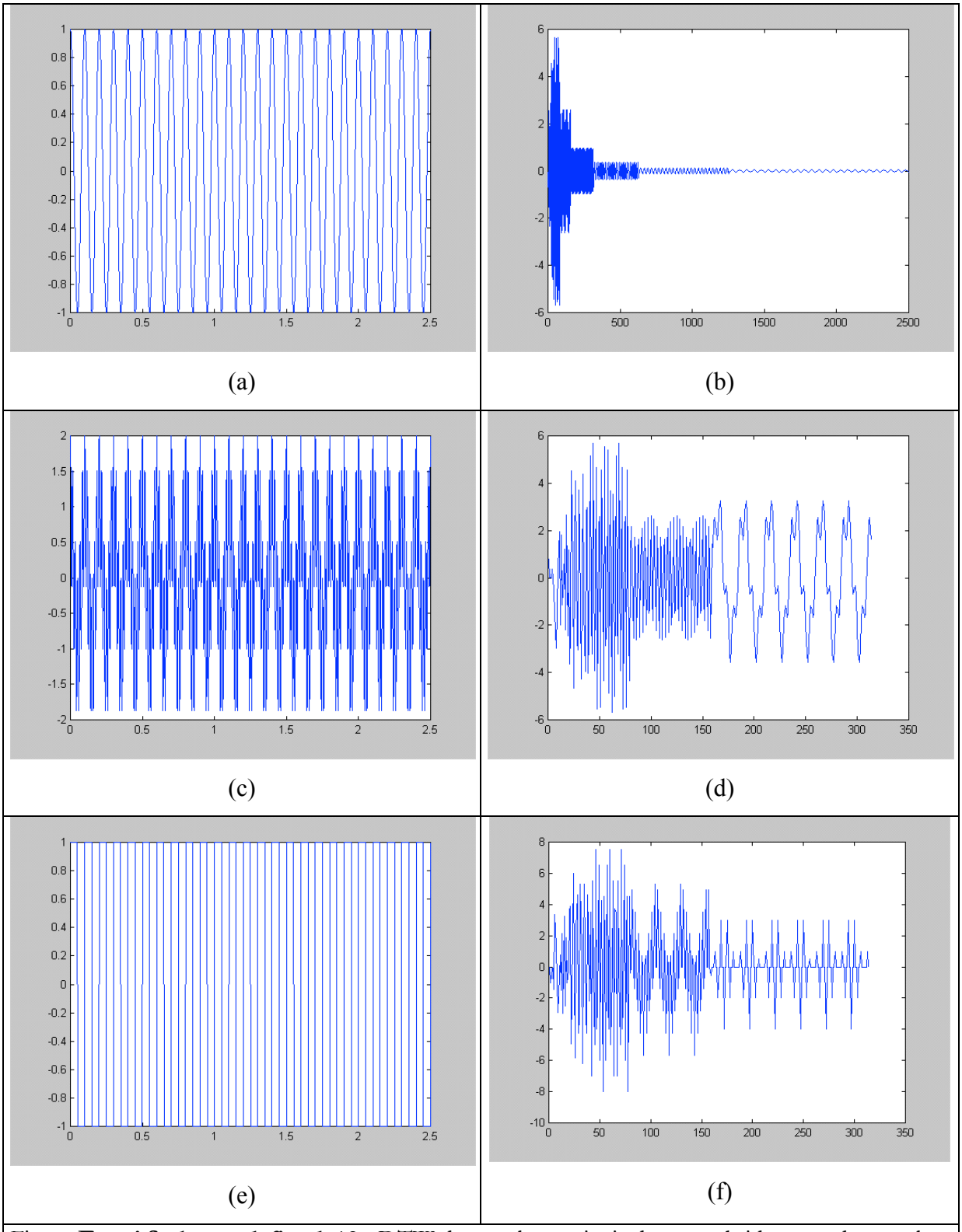


Figura **Error! Style not defined.**.12: DTW de um alguns sinais de teste obtidos usando wavelets Haar. À esquerda o sinal original e à direita os coeficientes *d*[*n*]. (a) *f* = cos(*2.pi.10.t*); (c) *f* = cos(*2.pi.10.t*) + cos(*2.pi.60.t*); (e) onda quadrada de 10Hz

2.3.4 Transformada Wavelet Discreta em 2 dimensões

O conceito da DWT estudado em uma dimensão pode ser facilmente estendido para duas dimensões. Nesse caso, se considerando-se uma matriz bidimensional M qualquer pode-se aplicar a DWT sobre as linhas, as colunas ou ainda as diagonais de M. Com isso, serão computados os coeficientes de aproximações e detalhes da mesma forma que para uma sequência unidimensional.

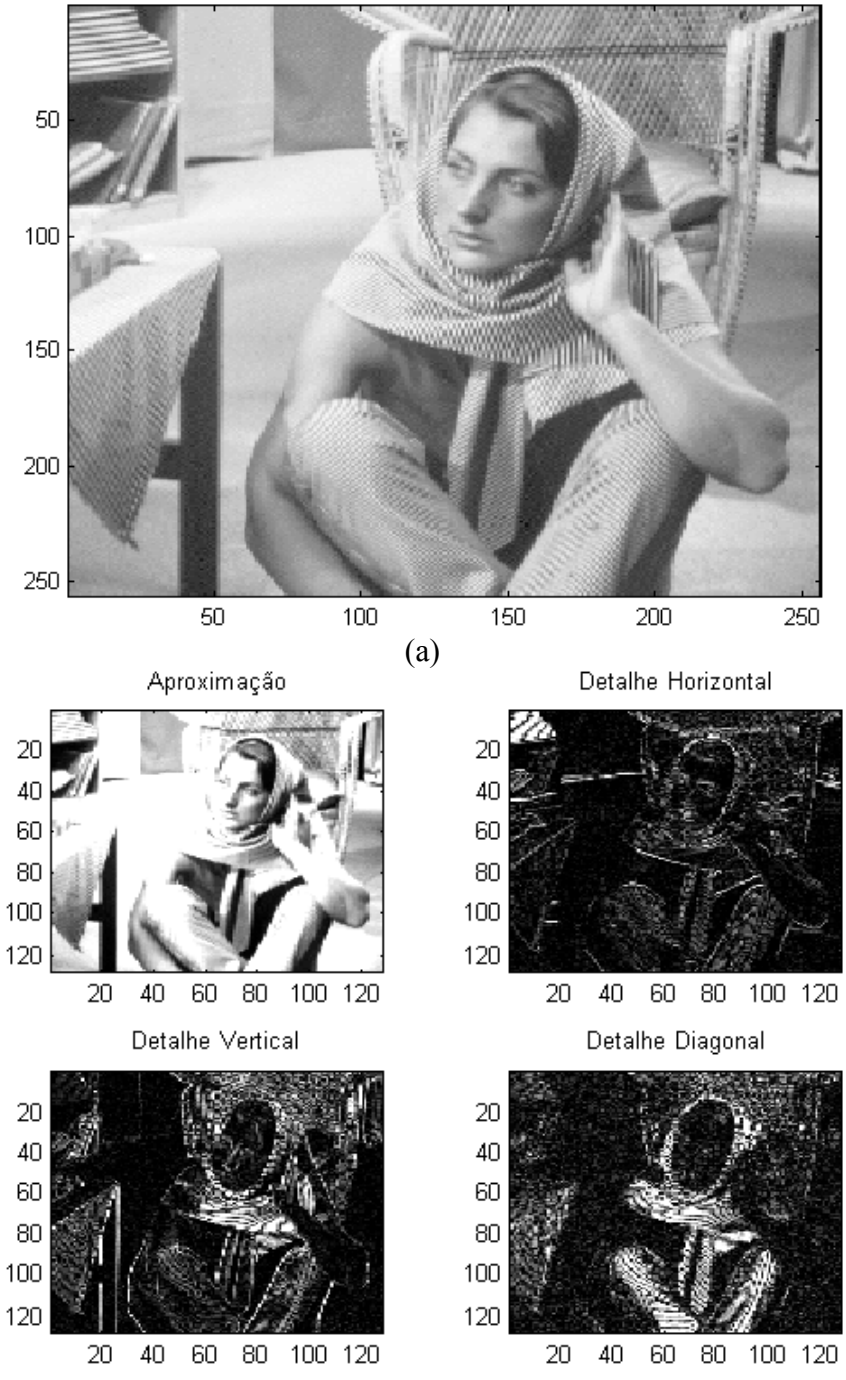


Figura **Error! Style not defined.**.13: DTW de uma imagem apresentando os coeficientes obtidos no primeiro nível. (a) Imagem original; (b) Coeficientes da DWT de 1 nível com wavelet de Haar

Uma abordagem usualmente feita é extrair os coeficientes da DWT e compara-los lado a lado de forma se ter uma visão geral de todos os coeficientes simultaneamente. Um exemplo disso pode ser visto nas imagens apresentadas na Figura 2.13. Note-se que as imagens obtidas pelos coeficientes possuem metade do tamanho da imagem original, como consequência da filtragem de metade da banda causada pela DWT.

3

Análise de Imagens

O sistema visual humano é uma ferramenta altamente especializada em sua tarefa de codificar informações para o cérebro e a sua simulação em um ambiente computacional tem demandado muitos esforços de diversos pesquisadores nas mais diversas áreas [Bron00].

A primeira dificuldade na implementação de um sistema artificial de visão é a generalidade das funções realizadas pelo sistema visual: reconhecimento de padrões, interpretação de cores, acompanhamento de movimentos, compensação automática de variações na luminosidade da cena em foco, etc. Na verdade, a complexidade e especialização dessas funções levou os pesquisadores a se dedicarem a cada uma delas isoladamente, gerando diversas teorias e sistemas que procuram repetir com o sucesso do sistema visual e a análise de imagens é uma dessas área .

A análise de imagens lida com o processamento das imagens e a análise de suas características [Gose96]. Frequentemente, uma imagem crua precisa ser processada para que se diminua a quantidade de informação irrelevante ou para que se reduza o ruído, melhorando as propriedade da imagem visando facilitar e tornar mais confiável a extração das características que se quer estudar na mesma.

Genericamente falando, as operações sobre imagens, ou transformações, podem assumir três naturezas [Gose96]: a) transformação imagem/imagem, onde a imagem original é substituída por uma nova imagem, resultante do processamento da primeira; b) transformação imagem/característica, onde são extraídas informações de caracterização da imagem e c) transformação característica/decisão, na qual uma ou mais características da imagem é usada para algum tipo de classificação e decisão.

No processamento em computadores as imagens são representadas por matrizes obtidas pela digitalização de um sinal analógico recebido de algum elemento de captação (câmera de vídeo, câmera de raios-X, sensor de prótons, etc) e convertido para digital por um conversor A/D. A taxa de amostragem define a quantidades de pontos presentes na matriz, que é chamada de *resolução geométrica* e define o nível máximo de detalhes da imagem.

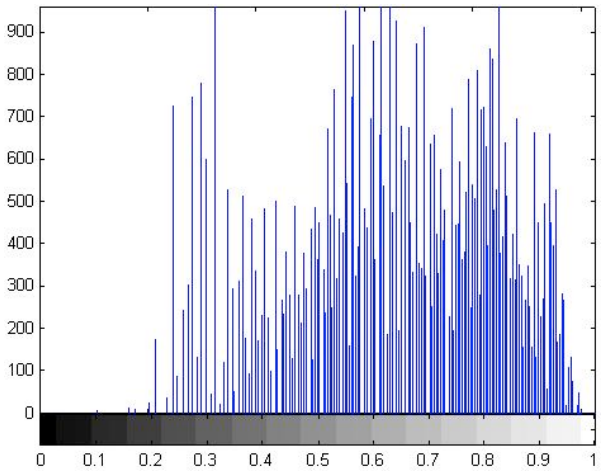
Cada ponto da matriz é chamado de *pixel*, uma abreviação do termo inglês *picture element*. O valor do pixel está relacionado com a intensidade luminosa do respectivo ponto da imagem e sua faixa de valores geralmente é uma potência 2. Estudos mostraram que o olho humano não pode distinguir mais que 256 níveis de energia em uma cor. Por isso, tipicamente usa-se 8 bits para representar o valor do pixel, sendo que para imagens em preto e branco essa faixa é chamada de resolução da escala de cinza, profundidade de cinza ou *escala de cinza*, com a cor preta representada por 0 e a branca por 255.

3.1 Histogramas

Uma informação bastante significativa da imagem é a sua distribuição de níveis de cinza, ou seja, a frequência com que cada nível ocorre na imagem. Essa distribuição é apresentada através de um histograma, que nada mais é do que um gráfico relacionando o nível de cinza (abscissa) com a frequência de sua ocorrência na imagem (ordenada).

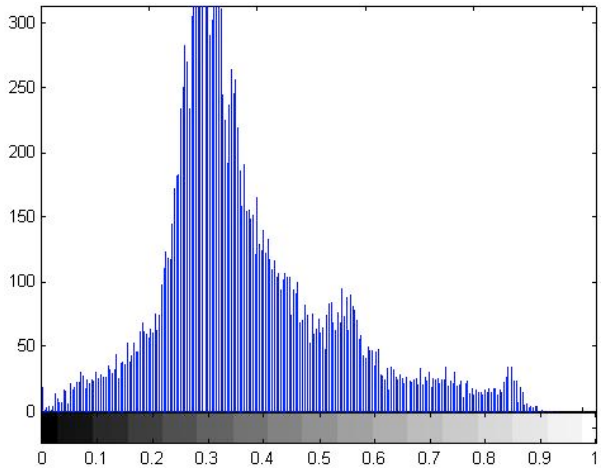
Um histograma é montado dividindo-se a faixa de níveis que se quer estudar em intervalos que são chamados de *bins* ou células. Após isso, conta-se o número de pixels cujos níveis estão presentes dentro desse intervalo e plota-se esse resultado, em função dos bins, na forma de um gráfico de barras. A Figura 3.1-1 apresenta algumas imagens e seus respectivos histogramas.

(a)


900
800
700
600
500
400
300
200
100
0
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

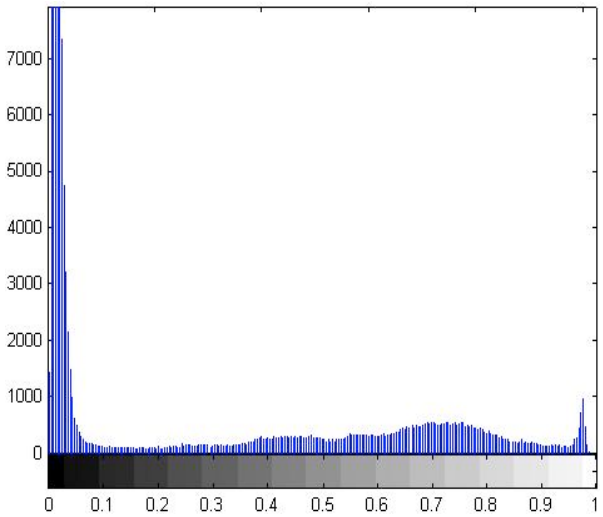
(b)

(c)


300
250
200
150
100
50
0
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

(d)

(e)


7000
6000
5000
4000
3000
2000
1000
0
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

(f)

Figura **Error! Style not defined.**.1: Exemplos de histogramas de imagens com 256 níveis de cinza

Os histogramas apresentados na Figura 3.1-1 foram construídos para imagens fonte com 256 níveis de cinza representados dentro de uma faixa de valores entre 0 e 1, sendo 1 o valor para o nível de branco e 0 para o nível de preto.

3.2 Equalização

Frequentemente o histograma de uma imagem apresenta uma concentração de pontos em alguma região. Isso pode acontecer por consequência da própria natureza da imagem ou como resultado de transformações aplicadas sobre a mesma durante o processamento.

Nessas situações, pode-se decidir pela construção de um histograma assimétrico, ou seja, um histogramas onde a largura de cada *bin* diferente. Nos histogramas assimétricos usa-se *bins* mais estreitos nas regiões onde há uma alta concentração de níveis, e *bins* mais largos nas regiões onde a baixa concentração. Dessa maneira pode-se melhorar a resolução do histograma nas regiões com maior população de níveis.

Uma maneira de se alcançar esse objetivo é fazer com que a largura dos *bins* seja calculada para que todos eles contenham o mesmo número de pixels. Outra é aplicar uma transformação de forma a que os níveis de cinza da imagem sejam igualmente distribuídos ao longo de toda a faixa possível da escala de cinza, de tal forma que cada nível contenha aproximadamente a mesma quantidade de pixels. Essa transformação recebe o nome de equalização, ou também transformação de alisamento do histograma [Gose96].

A equalização tem o efeito de melhorar o contraste das regiões da imagem que apresentam alta concentração de pixels com níveis de cinza similares, como consequência do espalhamento que provoca na escala de cinza.

Um algoritmo frequentemente usado para implementar a equalização é listar os pixels da imagem original em ordem crescente do nível de cinza. Após isso, divide-se a lista em N partes iguais, onde N é o número de níveis de cinza para a nova imagem que se quer obter, e atribui-se nível 0 para os pixels da primeira parte, 1 para os da segunda e assim sucessivamente.

Uma outra abordagem corrige o valor de cada pixel utilizando uma função de transformação ax^{γ}, vista na Figura 3.2-a, também chamada função de mapeamento. Essa função faz o mapeamento dos valores dos pixels dentro de uma faixa níveis baixo e alto de forma a alargar ou encurtar a escala de cinzas de uma imagem, alterando ou não o valor dos pixels dentro da faixa (se $\gamma=1$ o valor do pixel na saída da função é o mesmo da entrada).

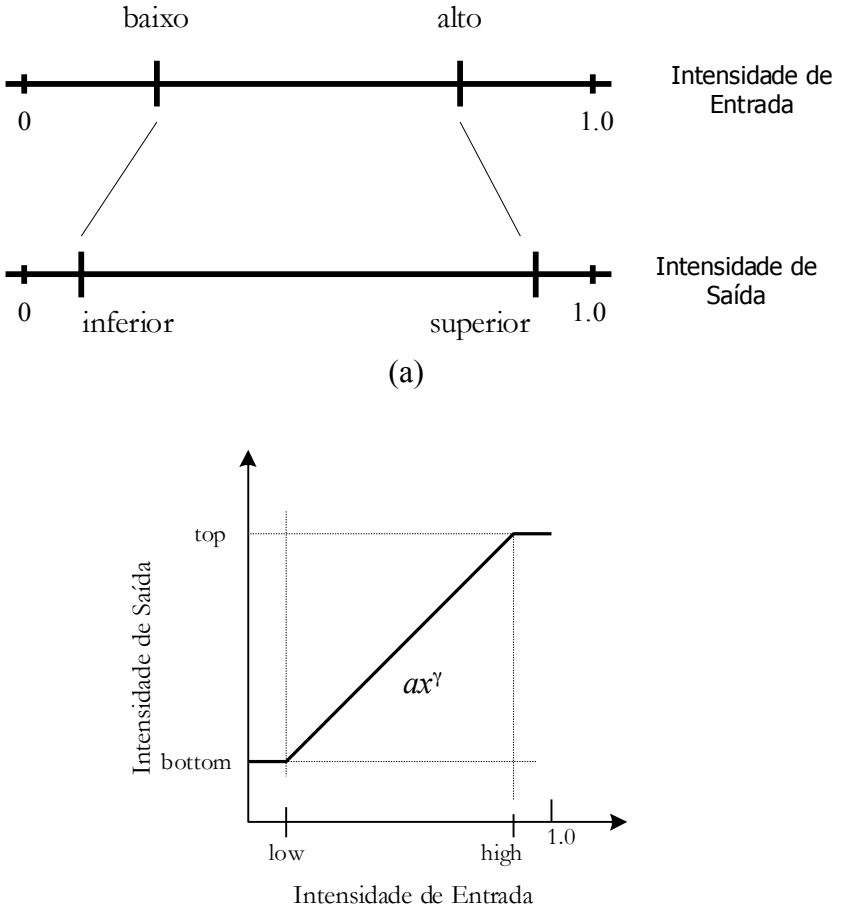


Figura **Error! Style not defined.**.2: Função de mapeamento para correção dos níveis de cinza.. (a) Correspondência entre os níveis de cinza da entrada e saída; (b) Função de mapeamento

Note-se que a função de mapeamento tem valores para limitar a faixa de operação. Os valores low e high são usados para definir a faixa de intensidade dos pixels que será usada na transformação. Valores fora dessa faixa serão truncados para os extremos da faixa de níveis da saída.

A título de ilustração, pode-se encontrar um exemplo de equalização pelo método de ajuste na Figura 3.2-b.

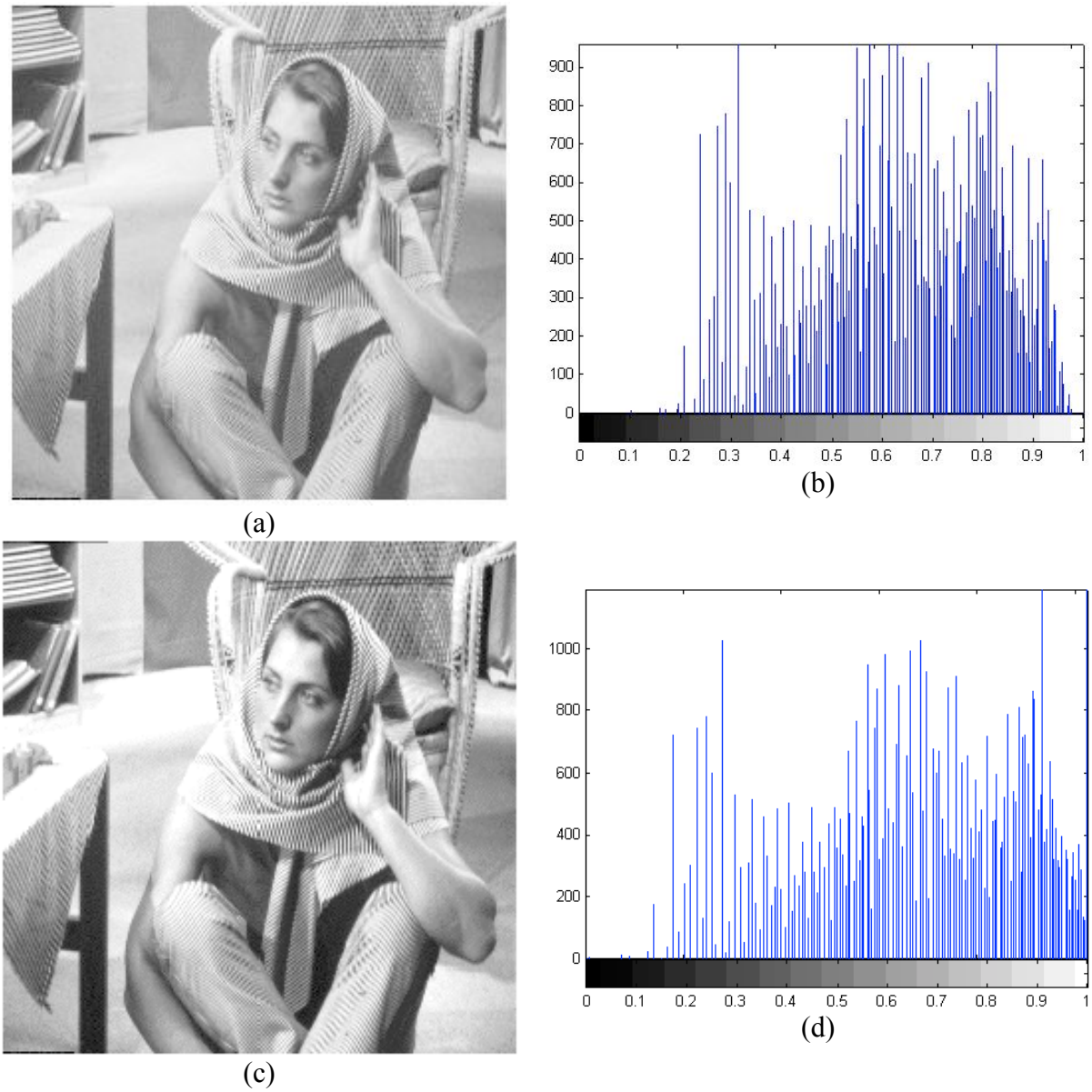


Figura **Error! Style not defined.**.3: Exemplo de equalização de imagens. (a) Imagem original; (b) Histograma da imagem (a); (c) Imagem após equalização (low=0.1, high=0.9, botton=0, top=1); (d) Histograma da imagem (c)

3.3 Detecção de bordas

Sabe-se que as bordas são uma das características mais importantes na interpretação de imagens. O que se chama de borda é região de transição entre um objeto na imagem e o fundo sobre o qual ele esta posicionado. Quanto maior a variação de intensidade entre os pixels do objeto e do fundo mais definida é a borda, sendo essa definição chamada de intensidade da borda.

A taxa de variação do nível de cinza em uma direção da imagem é igual à derivada parcial, ou seja,

$$\frac{\partial g(x,y)}{\partial x} = \lim_{\Delta x \to 0} \frac{g(x+\Delta x, y) - g(x,y)}{\Delta x} \quad \text{(Error! Style not defined.-1)}$$

para a taxa de variação horizontal e

$$\frac{\partial g(x,y)}{\partial y} = \lim_{\Delta y \to 0} \frac{g(x, y+\Delta y) - g(x,y)}{\Delta y} \quad \text{(Error! Style not defined.-2)}$$

para a taxa de variação vertical.

Partindo-se da equação (3-1) e considerando-se que o menor valor possível de Δx é 1, tem-se que a equação (3-1) torna-se

$$g'_x(x,y) = g(x+1,y) - g(x,y) \quad \text{(Error! Style not defined.-3)}$$

que é o diferença de primeira ordem de g em relação a x. Raciocínio análogo pode ser feito em relação à variável y chegando-se a diferença de primeira ordem de g em relação a y, ou seja,

$$g'_y(x,y) = g(x,y+1) - g(x,y) \quad \text{(Error! Style not defined.-4)}$$

Em processamento de imagens é comum representar expressões dessa natureza definindo operadores que representam os pesos de cada termo da equação. Assim, reescrevendo-se as equações (3-3) e (3-4) de uma nova maneira, tem-se que:

$$g'_x(x,y) = -1.g(x,y) + 1.g(x+1,y) \quad \text{(Error! Style not defined.-5)}$$

e

$$g'_y(x,y) = -1.g(x,y) + 1.g(x,y+1) \quad \text{(Error! Style not defined.-6)}$$

Considerando-se que, por definição, a variável x cresce da esquerda para a direita e a variável y cresce de baixo para cima ao longo da imagem, as equações (3-5) e (3-6) geram dois operadores,

a) $\begin{array}{|c|c|}\hline -1 & 1 \\ \hline \end{array}$ e b) $\begin{array}{|c|}\hline 1 \\ \hline -1 \\ \hline \end{array}$. (Error! Style not defined.-7)

que são chamados de detetores de borda, já que são usados para enfatizar ou detectar mudanças no nível de cinza entre dois pixels adjacentes.

Esses detetores indicam a velocidade com que o nível de cinza varia de um pixel para outro nas direções x e y. Valores positivos de g'_x indicam que o nível de cinza aumenta

quando se caminha da esquerda para a direita ao longo da imagem. Valores positivos de g'_y indicam que o nível de cinza aumenta quando se caminha de baixo para cima ao longo da imagem.

Muitas vezes é necessário detectar bordas diagonais nos objetos presentes na cena. Nesses casos usam-se operadores baseados em diferenças diagonais, tais como

$$\text{a)}\ \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad \text{e b)}\ \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}. \qquad \text{(Error! Style not defined.-8)}$$

ou

$$\text{a)}\ \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{e b)}\ \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}. \qquad \text{(Error! Style not defined.-9)}$$

Esses operadores são conhecidos como detetores de borda de Robert e são sensíveis a bordas diagonais da esquerda para a direita no sentido ascendente (equações 3-8a e 3-9a) ou descendente (equações 3-8b e 3-9b).

Outros detetores combinam algum tipo de suavização da imagem juntamente com a detecção de borda. Dentre estes, alguns dos mais comuns são [Gose96]:

a) o operador que combina suavização uniforme da imagem em uma direção e detecção de borda na direção perpendicular, conhecido como detetor de borda de Prewitt, representado por

$$\text{a)}\ \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{e b)}\ \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}; \qquad \text{(Error! Style not defined.-10)}$$

b) o operador que combina suavização binomial e detecção de borda, conhecido como detetor de borda de Sobel, representado por

$$\text{a)}\ \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{e b)}\ \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}; \qquad \text{(Error! Style not defined.-11)}$$

A Figura 3.4 apresenta alguns exemplos dos operadores uma mesma imagem. Note-se a significativa diferença do resultado obtido pelo método de Robert com os resultados obtido pelos métodos de Sobel e Prewitt.

(a)

(b)

(c)

(d)

Figura **Error! Style not defined.**.4: Resultados de diversos métodos de detecção de borda sobre uma mesma imagem. (a) Imagem original; (b) Método de Robert; (c) Método de Prewitt; (d) Método de Sobel

4

Redes Neurais Artificiais (RNA)

4.1 Modelo de McCulloch-Pitts

As Redes Neurais Artificiais (RNA) são inspiradas nos sistemas neuronais biológicos e nos sistemas nervosos dos seres vivos, também denominados de Redes Neurais Naturais (RNN).

Atualmente, as pesquisas em RNA não se preocupam primordialmente imitar os sistemas naturais e sim em atender à duas novas motivações [Azev00]:

- Modelar o sistema nervoso com precisão tal que permita a observação de um comportamento que, equivalendo ao comportamento de um ser vivo modelado, possa servir como apoio às hipóteses usadas na modelagem;
- Construir computadores com alto grau de paralelismo

A teoria de RNA surge a partir do final século XIX com a identificação do neurônio biológico pelo neurologista espanhol Ramón y Cajal. A partir daí muitos pesquisadores e cientistas agregaram novos conhecimentos ao estudo do sistema nervoso até que em 1943 Warren McCulloch e Walter Pitts publicaram na revista *Bulletin of Mathematical Biophysics* o artigo "*A Logical Calculus of the Ideas Immanent in Nervous Activity*", criando a base para a teoria das RNA tal qual é conhecida atualmente.

O modelo de McCulloch-Pitts ou MCP, como é também denominado, define o neurônio como um sistema binários cujas entradas binárias e a saída binária são relacionadas por um discriminador linear que pode ser genericamente definido pela seguinte equação:

$$y = H\left(\sum_{i=1}^{n} w_i x_i - \Theta\right) = h(\mathbf{w}^t \mathbf{x} - \Theta) \qquad \text{(Error! Style not defined.-1)}$$

em que o vetor $\mathbf{w} = [w_1, w_2, ..., w_n]$ é o vetor de ganhos associadas às entradas do vetor **x**, Θ é o valor do limiar, $h(.)$ é a função degrau unitário e $y \in [0;1]$ é a saída binária. A figura 4.1 ilustra o diagrama de blocos do discriminador linear.

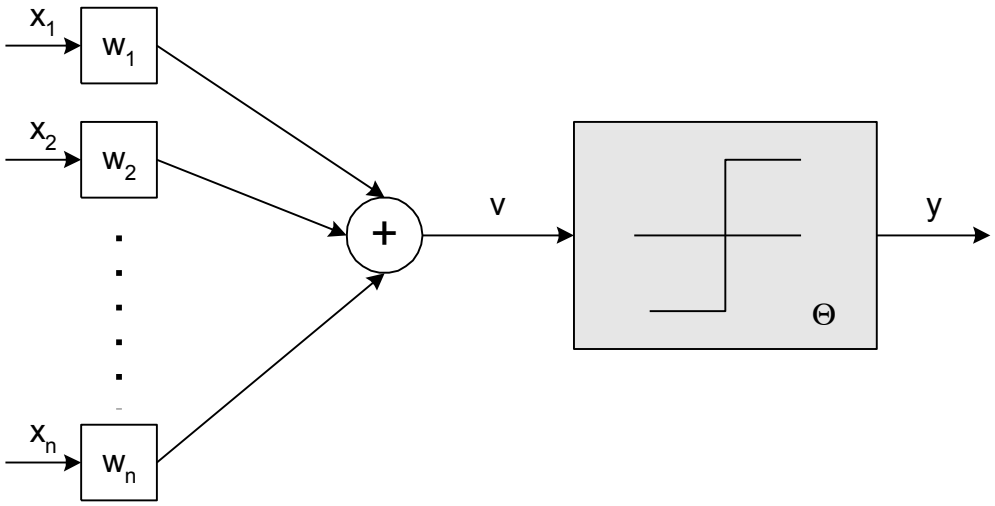


Figura **Error! Style not defined.**.1: Diagrama de blocos de um discriminador linear representando o modelo de neurônio McCulloch-Pitts

Considerando-se um vetor **x** no espaço euclidiano $\Re^n$, vê-se que, matematicamente, o modelo de McCulloch-Pitts é capaz de separar esse espaço em duas regiões *A* e *B*, conforme o seguinte comportamento:

$$\begin{cases} \mathbf{w}^t\mathbf{x} - \Theta > 0 \Rightarrow \mathbf{x} \in A \Rightarrow y = 1 \\ \mathbf{w}^t\mathbf{x} - \Theta < 0 \Rightarrow \mathbf{x} \in B \Rightarrow y = 0 \end{cases} \quad \text{(Error! Style not defined.-2)}$$

Esse comportamento pode ser usado como classificador de padrões ou separador de aglomerados, desde que os padrões que se quer separar possam ser representados por vetores linearmente separáveis. Essa situação pode ser melhor descrita considerando-se dois conjuntos de vetores **U** e $\mathbf{V} \in \Re^n$, compostos, respectivamente, por uma quantidade *k* e *m* vetores de *n*-dimensionais. Se **U** e **V** formarem aglomerados no espaço $\Re^n$ tal que exista um hiperplano Θ que atenda

$$y = H\left(\sum_{i=1}^{n} w_i x_i - \Theta\right) = h(\mathbf{w}^t\mathbf{x} - \Theta) \quad \text{(Error! Style not defined.-3)}$$

então diz-se que os conjuntos **U** e **V** são linearmente separáveis, caso contrário não haverá um discriminador linear porque o conjuntos não são linearmente separáveis.

4.2 Perceptrons

Frank Rossenblatt, em 1958, dando prosseguimento ao trabalho de McCulloch, desenvolveu uma rede de múltiplos neurônios e a chamou de Perceptron (Figura 4.2), que também é conhecida pela sigla MLP, do inglês *Multi Layer Perceptron* (Perceptron Multi Camadas). Conforme pode ser visto na Figura 4.2, a MLP é uma rede composta por camadas de neurônios, as quais podem ser divididas basicamente em três classes [Kova96]:

c) Camada de entrada, ou primeira camada, composta pelos neurônios que recebem os sinais de excitação;
d) Camada de saída, composta pelos neurônios cujas saídas são as saídas da rede;
e) Camadas intermediárias, ou camadas ocultas, que interligam as camadas de entrada e saída.

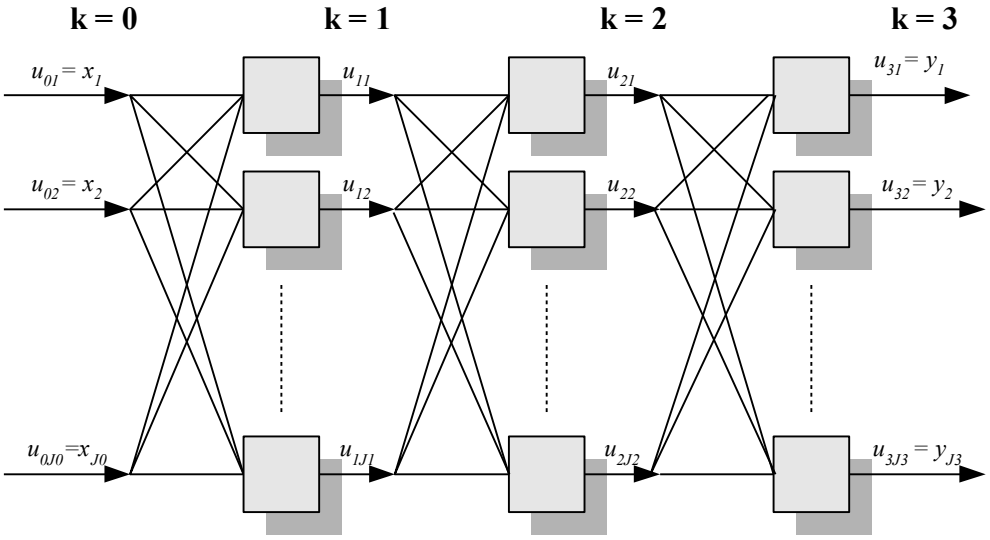


Figura **Error! Style not defined.**.2: Diagrama de uma MLP (perceptron) de 4 camadas

As camadas da MLP da Figura 4.2 têm J0, J1, J2 e J3 dimensões, que correspondem ao número de neurônios em cada uma.

Rosenblatt procurava desenvolver um método para ajuste dos ganhos e limiares dos nós do Perceptron de forma a poder adequa-lo a situações onde a função discriminatória não é simples de ser determinada [Kova96]. Para isso Rossenblat inspirou-se nos sistemas nervosos biológicos e imaginou que seria possível "ensinar" uma MLP qual a função que ela deveria modelar através de algum método de ajuste dos pesos e dos limiares dos MCPs em cada nó. Além disso, Rossenblatt apoiou-se num estudo feito pelo biólogo D. Hebb em 1949, que propôs um princípio pelo qual o aprendizado de um sistema nervoso complexo pode ser

reduzido a um processo puramente local, em que os ajustes de ganhos sinápticos de cada célula nervosa é dependente apenas dos erros detectados nessa célula.

De uma maneira geral, o processo de aprendizagem, também chamado de adaptação, consiste em uma sequência de ajustes nos pesos de cada nó de forma a aproximar a saída da rede ao valor da solução desejada. Na forma algébrica, pode-se dizer que, sendo **w** o vetor de pesos de um MCP da rede, o aprendizado consistirá em definir-se um $\Delta\mathbf{w}$ tal que $\mathbf{w}(t+1) = \mathbf{w}(t) + \Delta\mathbf{w}$ gere uma saída melhor que $\mathbf{w}(t)$. Hebb teorizou que o $\Delta\mathbf{w}$ de cada MCP da rede pode ser calculado unicamente com base em seu erro de saída, não importando os demais MCPs da rede. Esse princípio ficou conhecido como P*rincípio de Aprendizado de Hebb* e pode ser escrito pela seguinte equação:

$$w_i^{new} = w_i^{old} + \Delta w_i \qquad \text{(Error! Style not defined.-4)}$$

com

$$\Delta w_i = \eta \cdot e \cdot x^d \qquad \text{(Error! Style not defined.-5)}$$

e

$$e = (y^d - y) \qquad \text{(Error! Style not defined.-6)}$$

em que y^d é a saída desejada quando o MCP é excitado com a entrada x^d, y é a saída calculada do MCP e η é a taxa de aprendizagem. Raciocínio semelhante pode ser empregado para o cálculo do novo limiar Θ.

Rosenblatt definiu um algoritmo que baseava-se na apresentação de diversos exemplos à rede e no ajuste sucessivo dos pesos e limiares de cada nó até que fosse atingido um determinado patamar de erro. O algoritmo de Rosenblatt, que ficou conhecido como *lei de aprendizado do Perceptron* sempre converge em um número finito de operações [Hayk94], desde que o problema consista na classificação de dois conjuntos linearmente separáveis.

4.3 ADALINE

O termo ADALINE é um acrônimo da denominação ADAptive LInear Neuron e representa um modelo desenvolvido por Widrow e Hoff em 1960, cujo algoritmo de treinamento é conhecido como Regra Delta, que foi a precursora do algoritmo *backpropagation* para treinamento de MLPs (ver item 4.5).

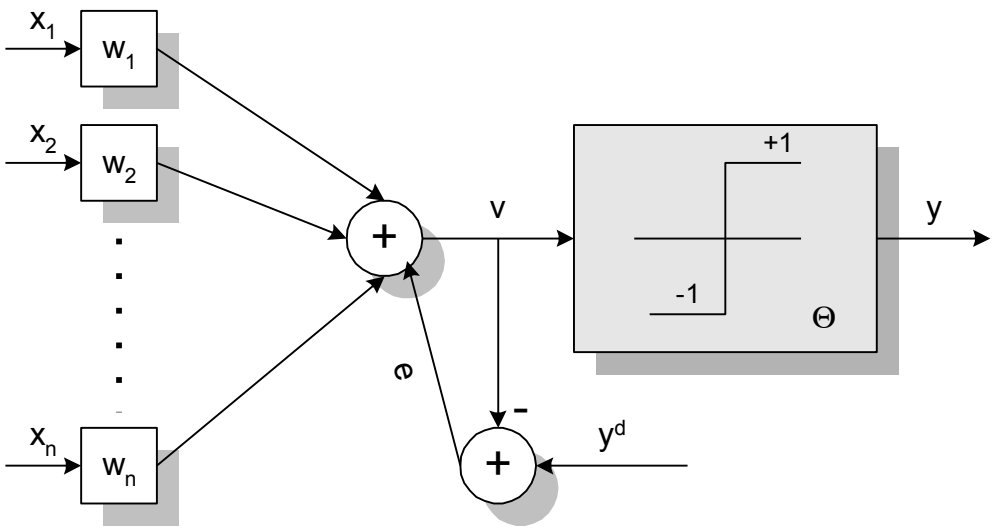


Figura **Error! Style not defined.**.3: Diagrama de blocos do modelo ADALINE

A principal característica do modelo ADALINE é que a função erro é calculada a partir da saída linear v (Figura 4.3) o que leva a uma conveniente minimização de uma função erro quadrática. Isso pode ser expresso da seguinte maneira:

Seja a função erro expressa pela equação

$$e = \frac{1}{2}\sum_{i=1}^{n}(y_i^d - v_i)^2 \qquad \textbf{(Error! Style not defined.-7)}$$

em que $\Gamma=\{(x_i\ ,\ y^d{}_i)\}$, i é o i-ésimo padrão de treinamento ($i = 1,2,...n$), é um conjunto de treinamento e $v_i = w_i.x_i$ é a saída calculada para uma determinada entrada x_i.

Utilizando-se o método do gradiente e partindo-se um ponto arbitrário **w**(0), objetiva-se encontrar qual o vetor de pesos que minimiza e. Isso pode ser conseguido ajustando-se **w** de forma a caminhar-se na direção oposta ao gradiente, ou seja, $\Delta\mathbf{w} \propto -\nabla e$. Através de algumas manipulações algébricas, chega-se à fórmula

$$\nabla e = \frac{\partial e}{\partial w_i} = -x_i \cdot e \qquad \textbf{(Error! Style not defined.-8)}$$

Com isso conclui-se que $\Delta\mathbf{w} \propto e.x_i$, ou

$$\Delta\mathbf{w} = \eta.e.x_i \qquad \textbf{(Error! Style not defined.-9)}$$

4.4 Escolha da topologia de uma rede MLP

O trabalho de definição da melhor topologia de uma rede MLP é basicamente empírico e não há uma heurística para resolver esse problema. Entretanto, os trabalhos de Cybenko, em 1988 e 1989, determinaram que uma MLP de três camadas pode aproximar qualquer função contínua e uma de quatro pode aproximar qualquer função matemática, ajudando a restringir o universo de configurações possíveis.

A definição do número adequado de nós em cada camada da rede depende de vários fatores, como: número de exemplos usados no treinamento, quantidade de ruído presente nos exemplos, complexidade da função a ser aprendida, distribuição estatística dos dados de treinamento, etc. Também aqui não uma forma determinística para cálculo do número de elementos, ficando o método sujeito à experiência do projetista em relação ao problema que está sendo tratado.

4.5 Treinamento de redes MLP (Algoritmo *backpropagation*)

Existe atualmente um grande número de algoritmos para treinamento de redes MLP, sendo que estes podem ser divididos basicamente em estáticos e dinâmicos. Os algoritmos estáticos não alteram a estrutura da rede durante o treinamento, modificando apenas os pesos e limiares. Os algoritmos dinâmicos alteram também a topologia da rede, aumentando ou diminuindo o número de nós e conexões.

O algoritmo de aprendizado mais utilizado em redes MLP é o *backpropagation*, desenvolvido em 1986 por Rummelhart, Hinton e Williams. Esse é um algoritmo do tipo estático onde o aprendizado ocorre em duas fases, chamadas de *forward* e *backward*. Na fase *forward* a saída da rede é calculada com base num padrão de entrada. Na fase *backward* calcula-se o erro entre a saída desejada e a calculada para que seja feito o ajuste dos pesos das conexões.

O algoritmo *backpropagation* é montado com base na generalização da Regra Delta de Widrow e por isso é também conhecido como Regra Delta Generalizada. Nesse algoritmo é proposta uma forma de definir os erros das camadas intermediárias, possibilitando o ajuste dos pesos destas. Dessa forma, a função erro a ser minimizada passa a contemplar as saídas de todos os neurônios de todas camadas, ou seja,

$$E = \frac{1}{2}\sum_{n}\sum_{i=1}^{k}(y_i^d - y_i)^2 \quad \text{(Error! Style not defined.-10)}$$

em que E é o erro total, n o número de padrões de treinamento, k o número de unidades de saída, y_i a i-ésima saída calculada e y_i^d a i-esima saída desejada.

A regra delta generalizada requer que as funções de ativação sejam semi-lineares, ou seja, sejam contínuas, diferenciáveis e não-decrescentes [Kova96]. Para esse algoritmo, a correção Δw_{ji} que deve ser aplicada aos pesos do i-ésimo neurônio da j-ésima camada será

$$\Delta w_{ji} = \eta . \delta_j . x_i \quad \text{(Error! Style not defined.-11)}$$

em que

$$\delta_j = \frac{\partial E}{\partial v_j} = (y_j^d - y_j)\Theta'(v) \quad \text{(Error! Style not defined.-12)}$$

se j for a camada de saída, ou

$$\delta_j = \frac{\partial E}{\partial v_j} = \Theta'(v)\sum_{l}\delta_l . w_{jl} \quad \text{(Error! Style not defined.-13)}$$

se j for uma camada intermediária.

Um problema enfrentado pelos algoritmos de treinamento de redes MLP é a dificuldade na definição dos parâmetros de treinamento. Um outro problema diz respeito a quando deve ser suspenso o treinamento, os assim chamados critérios de parada. Alguns dos critérios de parada a mais usados são: o encerramento do treinamento ao se atingir um número máximo de ciclos; o encerramento quando o erro quadrático médio ficar abaixo de valor; o encerramento quando o percentual de classificações corretas estiver acima de um valor; uma combinação dos métodos anteriores.

Uma outra questão é a frequência de atualização dos pesos. Nesse sentido existem duas abordagens em uso. A abordagem denominada *on-line* prevê que os pesos são ajustados após a apresentação de cada padrão de treinamento. Essa abordagem, embora mais rápida, pode ser instável de acordo com a magnitude do ator de aprendizado escolhido. Já a abordagem, denominada *batch*, prevê que os pesos só serão ajustados após todos os padrões terem sido apresentados. Embora mais estável, essa última tem um custo computacional elevado em função da grande massa de dados do conjunto de treinamento.

5

Metodologia

O estudo atual se aproveitou das imagens já captadas e catalogadas no trabalho apresentado por Fernando Ribeiro [Ribe99].

O trabalho citado catalogou um total de 8402 imagens obtidas a partir da filmagem do tráfego da Av. Garibaldi (próximo à Vasco da Gama), do ponto de um viaduto que liga a Rua Caetano Moura à Av. Cardeal da Silva. A partir desse conjunto, foi utilizado um subconjunto de 3200 imagens escolhidas aleatoriamente mas que atendessem ao pré-requisito de apenas possuir um número inteiro de carros e não apresentar ônibus, caminhões ou motos, já que não permitiam uma equivalência consistente com um número de carros.

Cada imagem, após digitalizada e processada, resultou num arquivo em formato Bitmap, com resolução de 160x120 e 256 níveis na escala de cinza.

Neste estudo foram usados os seguintes equipamentos e programas:

a) **Equipamento:** Microcomputador padrão IBM-PC, marca COMPAQ, modelo Deskpro 4000, com processador Kingston TurboChip AMD-K6-2 de 366MHz, 96 MB de memória RAM e disco rígido de 3 GB.

b) Programas:

- Sistema Operacional Microsoft Windows 98SE;
- MATLAB® v5.0.0;
- MATLAB® Neural Network Toolbox v2.0.3;
- MATLAB® Image Processing Toolbox v2.0-Beta;
- MATLAB® Wavelet Toolbox v1.0.2.

A estratégia de trabalho foi implementada em quatro etapas. Na primeira fez-se uma equalização nas imagens para melhorar o contraste dos carros de cor escura em relação ao piso asfaltico. Em seguida aplicou-se a Transformada Wavelet Discreta (DWT) com Wavelet de Haar para reduzir o número de pontos das imagens. Na sequência foi realizado o processamento para a detecção de bordas dos carros presentes nas imagens e, finalmente, as imagens resultantes foram aplicadas à rede neural. Cada uma dessas etapas será descrita nos próximos tópicos.

5.1 Equalização das imagens

Durante o curso da pesquisa uma das principais dificuldades encontradas foi a de detectar carros cuja intensidade de brilho fosse baixa a ponto de dificultar a sua comparação com o piso asfáltico. Essa dificuldade foi acentuada quando as imagens passaram pelo processamento com wavelets (item 5.2), pois o resultado desse processamento terminava por reduzir o número de detalhes das imagens.

Um outro fator que se apresentou também como uma dificuldade foi a perspectiva apresentada pelas imagens captadas, que fez com que os carros mais distantes da câmera de filmagem tivessem seu tamanho relativo reduzido, resultando também numa redução da quantidade de informação que o representava e dificultando ainda mais a distinção de carros de cor escura (pouco brilho) em relação ao asfalto.

A solução para eliminar esse problema veio através da equalização das imagens (Capítulo 3), ou seja, da compensação seletiva do nível de cinza com o objetivo de melhorar o contraste dos carros de cor escura em relação ao piso asfáltico.

Pela observação do histograma das imagens originais foi possível estabelecer um critério que levou a um resultado satisfatório para o processo. Esse critério considera que os

pixels dentro da faixa [0.05,0.5], numa escala de cinza onde o valor 0 significa cor preta e 1 significa cor branca, têm seus valores corrigidos de forma a assumirem a faixa máxima, ou seja [0,1]. Todo trabalho foi realizado considerando uma curva de ajuste com $\gamma=1$, ou seja, uma reta.

O MATLAB® , através do Toolbox de Processamento de Imagens, permite a rápida equalização das imagens através da função **imadjust**. Essa função recebe como argumentos a imagem que se quer manipular na forma de uma matriz, a faixa de valores de entrada que serão computados na equalização e a faixa de valores de saída nos quais os valores de entrada serão computados. Dessa maneira foi possível melhorar significativamente o contraste das áreas escuras das imagens, como pode ser visto em alguns exemplos apresentados na Figura 3.1.

Note-se que a função de equalização corrige apenas os valores que estão dentro da faixa de entrada transformando-os em valores correspondentes na faixa de saída. Os valores que estão fora da faixa de entrada, ou seja, valores menores que 0,05 ou maiores que 0,5, são ajustados para o mínimo e o máximo da faixa de saída, respectivamente.

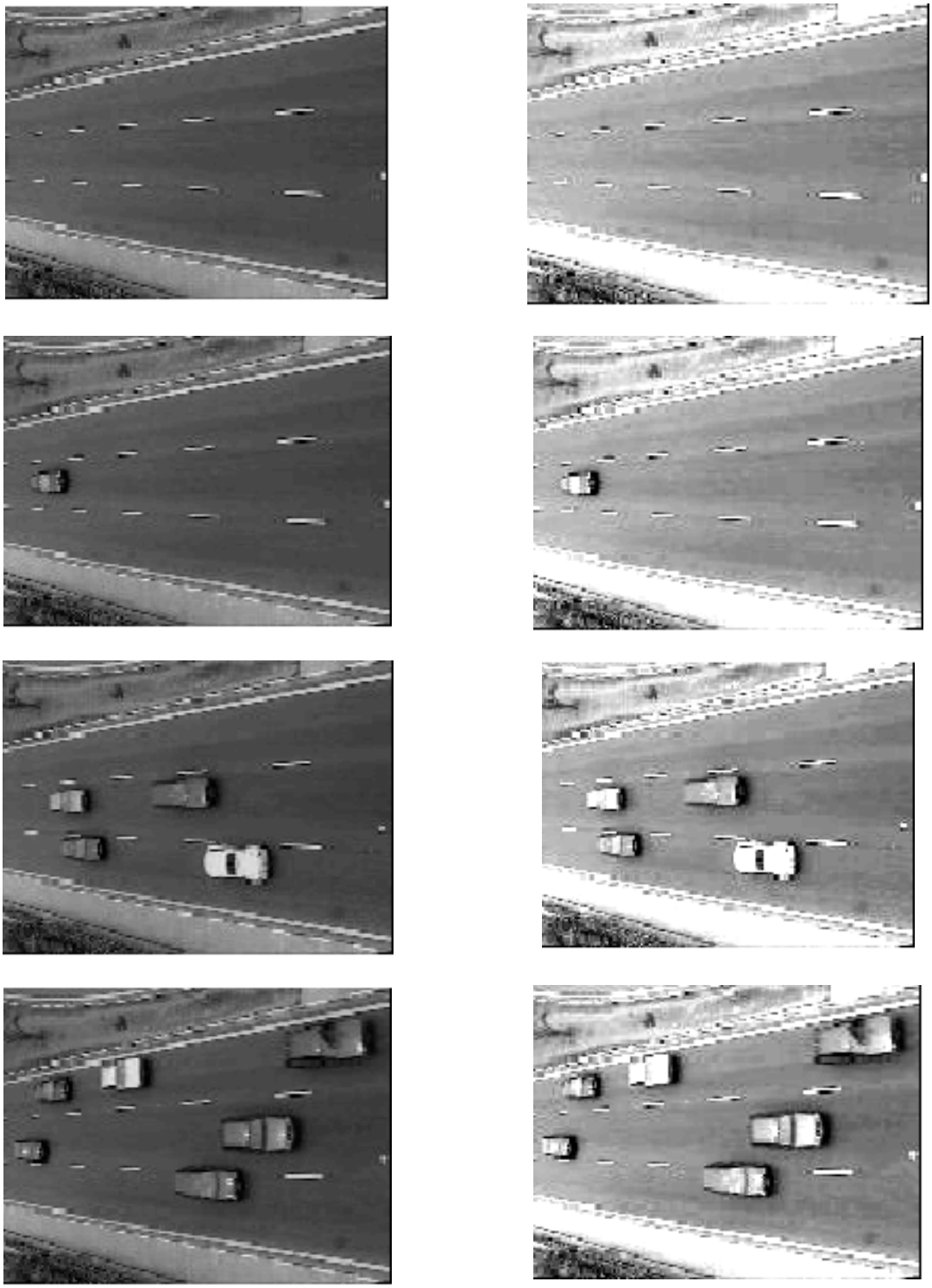


Figura **Error! Style not defined.**.1: Exemplos de imagens com e sem equalização. Do lado esquerdo as imagens originais e do lado direito as imagens equalizadas via função **imadjust** do MATLAB com faixa de entrada [0,05 0,5] e faixa de saúda [0 1].

5.2 Processamento com Wavelets

Nessa técnica reside o grande avanço conseguido no presente estudo. O resultado da aplicação da DWT sobre as imagens permitiu uma significativa redução no tamanho das mesmas sem que houvesse perda da informação do número de carros que se queria registrar.

A aplicação da DWT foi conseguida através das rotinas presentes no Toolbox Wavelet do MATLAB®. Nesse toolbox encontram-se as funções necessárias para utilização da Transformada Wavelet tanto contínua quanto discreta. Para a DWT, especificamente, existem as funções tanto para análise quanto para síntese de sinais de uma ou duas dimensões. Também é possível a escolha de uma das diversas wavelets mãe disponíveis.

No caso em questão, o objetivo foi obter os coeficientes de aproximação da DWT de segundo nível de cada imagem e, para isso, optou-se pelo uso das wavelets de Haar. A escolha dessa classe de wavelets mãe se deu por dois fatores: o primeiro foi a relação inteira entre o número de pontos da matriz das imagens original e o número de pontos da matriz dos coeficientes de aproximação obtidos; o segundo foi a decisão de observar o comportamento de uma wavelet clássica e que permiti-se uma rápida fácil implementação da DWT em um algoritmo externo ao MATLAB® para uso em estudos futuros. Durante o trabalho outras wavelets mãe foram testadas sem que se observa-se um ganho significativo em relação ao processo.

Feita a escolha da wavelet mãe (Haar), partiu-se das imagens originais na forma de uma matriz de 120 linhas por 160 colunas. Em seguida, utilizando-se a função **wavedec2** do MATLAB®, fez-se a aplicação da DWT obtendo-se os coeficientes da decomposição de segundo nível. No passo seguinte, utilizando a função **appcoef2,** gerou-se a matriz com 30 linhas por 40 colunas que corresponde a aproximação de segundo nível da matriz original.

Um exemplo dos resultados obtidos pode ser observado na Figura 5.2 onde são apresentadas as imagens da Figura 5.1 já devidamente equalizadas (lado direito da Figura 5.1) e ao lado dessas o resultado obtido pela DWT.

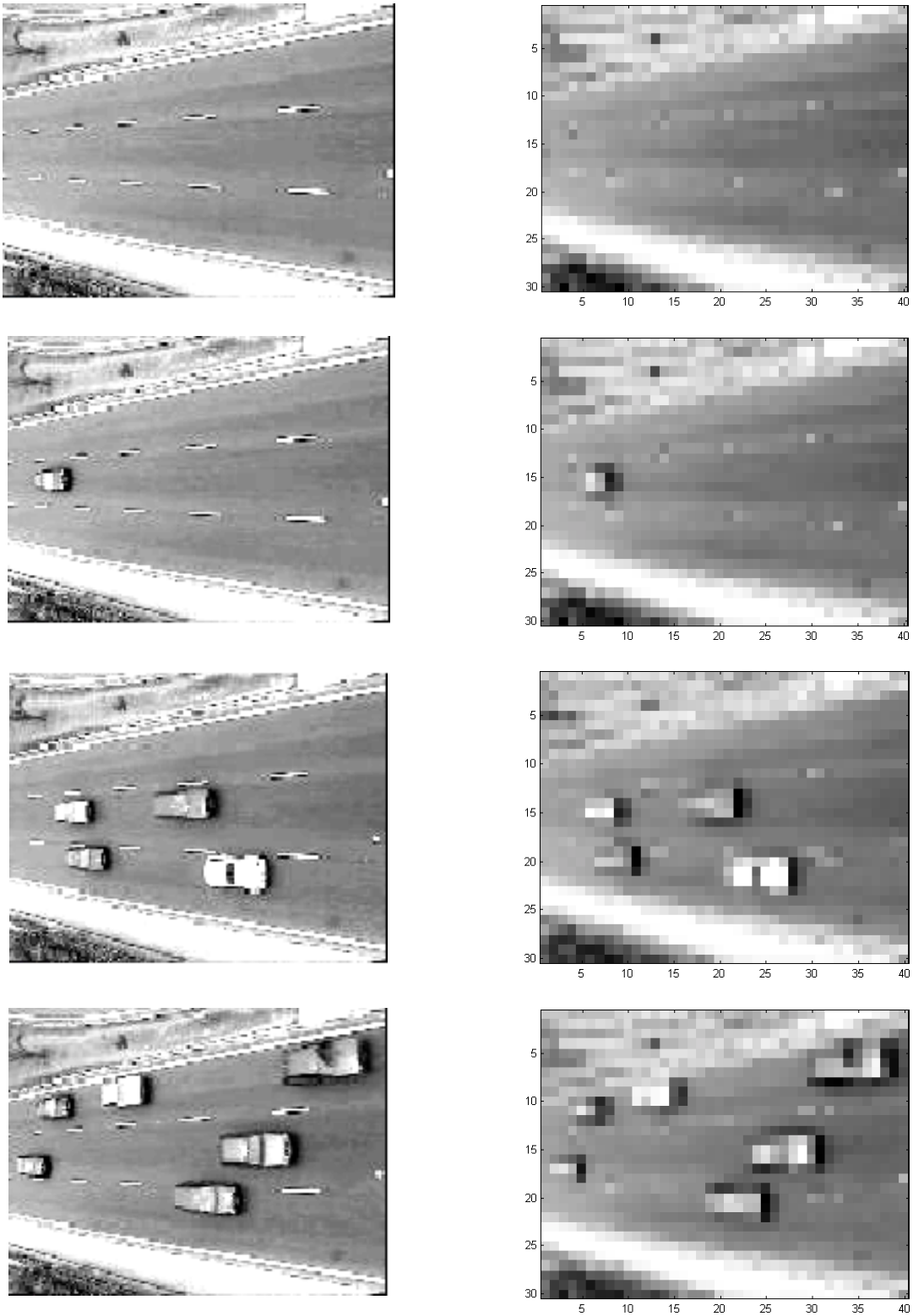


Figura **Error! Style not defined.**.2: Exemplos dos resultados obtidos para os coeficientes de aproximação de segundo nível após a aplicação da DWT com wavelets de Haar. As imagens originais tem resolução de 160x120 pontos e as aproximações de 40x30 pontos

5.3 Detecção de Bordas

A última etapa no tratamento das imagens foi realizar uma detecção de bordas sobre as imagens resultantes da DWT. Mais uma vez lançou-se mão do MATLAB através do Toolbox Processamento de Imagens que possui a rotina **edge** específica para esse fim.

No presente estudo optou-se pela utilização do detetor de bordas de Sobel. Essa escolha baseou-se no fato de que esse detetor apresenta uma boa resposta para bordas horizontais e verticais da imagem, além de realizar uma ponderação no cálculo do valor do pixel considerando o valor dos pixels imediatamente vizinhos.

O resultado da detecção de bordas foram matrizes binárias de 40x30 pontos onde as bordas eram representadas por bits 1. Essas matrizes sofreram ainda uma última manipulação para se tornarem vetores adequados para a apresentação à rede neural. Essa manipulação consistiu em extrair da imagem apenas a região de interesse, ou seja, a região que contem os carros, descartando-se os pontos fora dessa região.

Observando-se as Figuras 5.1 ou 5.2 pode-se notar que a área de interesse correspondo a um trapézio na horizontal, com o lado menor na borda esquerda da imagem. Por observação consegue-se facilmente definir para cada uma das 40 colunas que compõem cada imagem quais os pixels que podem ser descartados. Assim, dos 1200 (30x40) pontos da matriz original consegue-se formar um vetor coluna de 717 pontos pela concatenação dos segmentos de interesse em cada coluna da matriz.

Uma melhor compreensão do resultado pode ser obtida observando-se na Figura 5.3 o produto da função de detecção de bordas sobre as imagens obtidas da DWT. Note-se que as imagens apresentadas na Figura 5.3 já apresentam as regiões descartáveis devidamente anuladas.

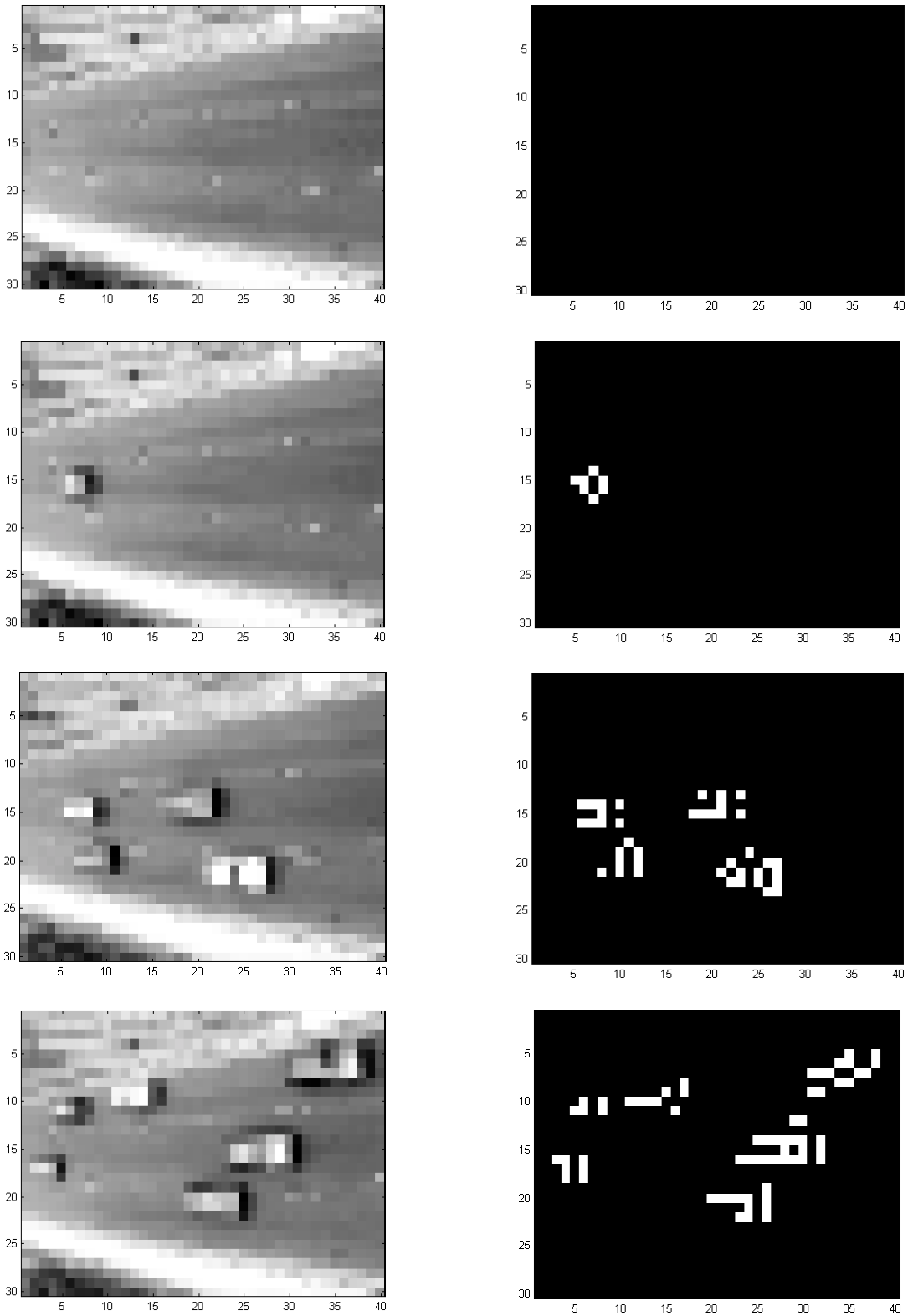


Figura **Error! Style not defined.**.3: Resultados da detecção de bordas sobre as imagens formadas pelos coeficientes de aproximação da DWT de segundo nível com wavelet de Haar.

5.4 Treinamento da Rede Neural

A topologia da rede Neural inicialmente utilizada foi a mesma já definida no trabalho de Fernando Ribeiro [Ribe99], para que fosse possível estabelecer uma comparação entre o novo método e o anterior.

Sendo assim repetiu-se o modelo de uma rede do tipo feed-forward composta por 20 neurônios na camada inicial, 20 na intermediária e 1 neurônio na camada de saída.

Para o treinamento foram usadas as mesmas 800 imagens originalmente selecionadas. Essas imagens passaram pelo processo relatado nos itens 5.1 a 5.3 e geraram ao final uma matriz de 717x800 pontos, onde cada uma das 800 colunas representou o resultado do processamento de uma imagem. A tabela 5.1 apresenta a distribuição da quantidade de carros por imagem no grupo de treinamento.

Total de carros por imagem do grupo treinamento								
0	1	2	3	4	5	6	7	8
209	297	85	75	68	45	11	0	10

Tabela **Error! Style not defined.**.1: Relação de carros por imagem no grupo de 800 imagens usadas no treinamento

Devido à significativa redução da massa de dados em relação ao método anterior pode-se optar pela apresentação simultânea das 800 imagens à rede, no lugar da divisão das 800 imagens em um grupo de 400 e dois de 200 imagens que foi feita anteriormente.

Nessa parte do trabalho foram usadas as funções do Toolbox Rede Neural do MATLAB para redes do tipo feed-forward. A primeira função utilizada foi a **initff** para inicialização da rede no MATLAB.

O treinamento propriamente dito foi feito com a função **trainbpx** que utiliza o algoritmo backpropagation. A limitação de memória do equipamento onde foram realizados os testes impediu a utilização da função **trainlm** que utiliza o algoritmo Levenberg-Marquardt, muito mais eficiente que o algoritmo backpropagation. Cada passo do treinamento, ou seja, a apresentação dos dados à rede (fase *forward*) seguida do cálculo dos erros e ajuste dos pesos (fase *backward*) recebe o nome de época. Assim, o número de épocas ao final do treinamento define quantos ciclos são necessários para que a rede atinja a precisão necessária na modelagem.

5.5 Validação da Rede Neural

Realizado o treinamento da rede partiu-se para a simulação da mesma utilizando-se as 2400 imagens restantes do lote de 3200 originalmente selecionado. Essas 2400 imagens passarram pelo mesmo processo de tratamento das 800 imagens usadas no treinamento da rede gerando uma matriz de 717 linhas por 2400 colunas, onde cada coluna representou uma imagem.

Total de carros por imagem do grupo de simulação								
0	1	2	3	4	5	6	7	8
634	1058	384	173	108	37	5	1	0

Tabela **Error! Style not defined.**.2: Relação de carros por imagem no grupo de 2400 imagens usadas na simulação

Para a simulação foi utilizada a função **simuff** do Toolbox Redes Neurais do MATLAB. Essa função recebe os pesos e constantes encontrados com a função **trainbpx** e calcula o vetor objetivo a partir de uma matriz de entrada. Para interpretar os resultados foram calculados diversos tipos de erros absolutos e relativos, através da comparação entre as saídas desejadas e as saídas calculadas pela rede. As fórmulas usadas para os erros foram as seguintes:

$$erro\ absoluto = |no.\ de\ carros\ estimado - no.\ real\ de\ carros| \quad \text{(Error! Style not defined.-1)}$$

$$erro\ absoluto\ médio = \frac{\sum_{N} erro\ absoluto}{N}, \text{em que } N \text{ é o número de imagens} \quad \text{(Error! Style not defined.-2)}$$

$$erro\ relativo = \frac{erro\ absoluto}{número\ de\ carros\ na\ imagem} \quad \text{(Error! Style not defined.-3)}$$

$$erro\ relativo\ médio = \frac{\sum_{N} erro\ relativo}{N}, \text{em que } N \text{ é o número de imagens} \quad \text{(Error! Style not defined.-4)}$$

5.6 Procedimentos gerais

Todas as funções descritas nos itens anteriores foram agrupadas de forma a automatizar as tarefas. As imagens utilizadas foram as mesmas obtidas e catalogadas no trabalho descrito em [Ribe99]

Inicialmente optou-se pela manipulação das imagens através da criação de dois arquivos texto com uma estrutura simples, onde uma linha continha o nome do arquivo da imagem e a outra o número de carros presentes na mesma. Um exemplo desses arquivos pode ser encontrado no Apêndice A.

O tratamento das imagens foi implementado através de um programa do MATLAB condensando todas as técnicas descritas nos itens 5.1 a 5.3. Esse arquivo está listado no Apêndice B. Ao final da execução esse programa gera um arquivo de dados no formato MATLAB contendo as matrizes resultantes do tratamento das imagens, bem como vetores *target* que serão usados no treinamento e simulação da rede neural.

O treinamento da rede neural foi implementado pelo programa descrito no Apêndice C. Ao final da execução esse programa gerou um arquivo de dados no formato MATLAB contendo os pesos e as constantes da rede bem como a curva de aprendizagem da rede durante o treinamento.

A validação da rede neural foi realizada pelo programa descrito no Apêndice D. Esse programa obtém a resposta da rede neural para um conjunto novo de imagens e calcula os erros dessas respostas em relação às respostas esperadas.

6

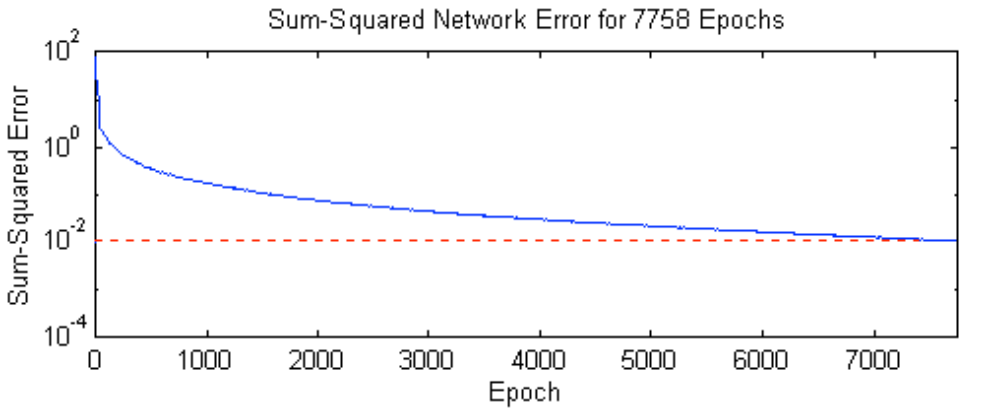
Resultados obtidos

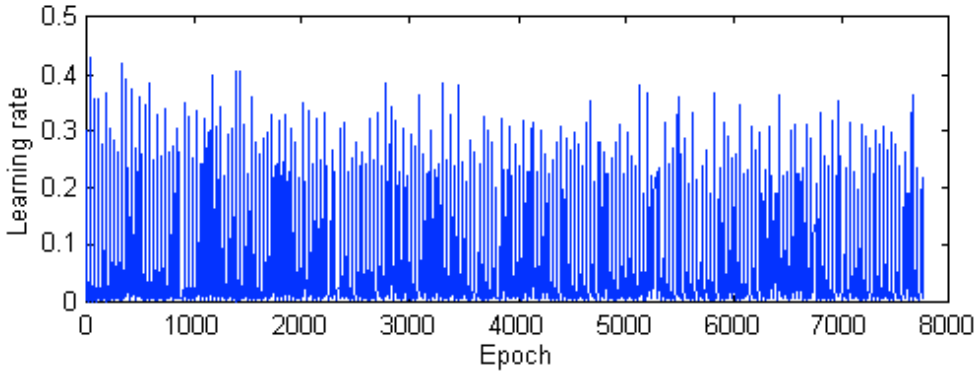
Nesse capítulo serão apresesentados os resultados em duas etapas. Na primeira será feita uma descrição dos resultados obtidos no estudo atual, focalizando os erros e tempos de execução do método desenvolvido. Na segunda será feita uma comparação com o método desenvolvido anteriormente no trabalho de Fernando Ribeiro [Ribe99] no intuito de mostrar a evolução obtida com o uso do método atual. Durante o capítulo usar-se-á a nomenclatura *método atual* com referência ao método desenvolvido nesse estudo e *método anterior* com referência ao método desenvolvido por Ribeiro.

6.1 Desempenho do método atual

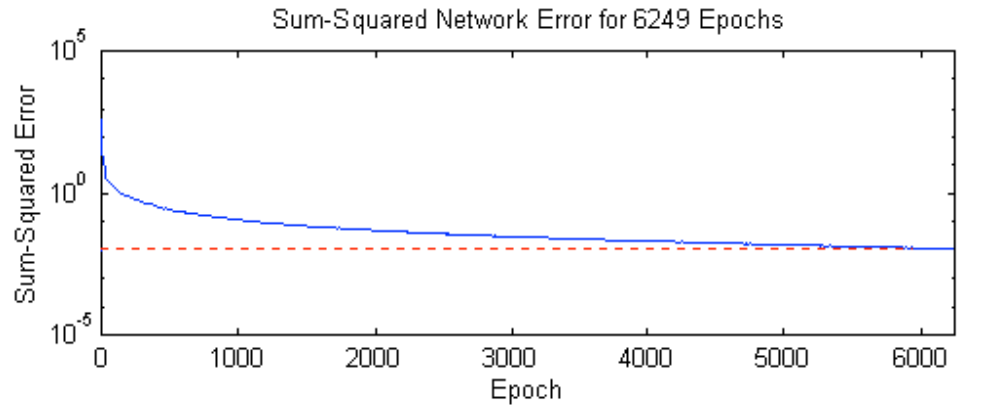
Utilizando-se a estrutura descrita no capítulo 5, foi possível realizar um processo de treinamento da rede neural com um grupo de 800 imagens. A rede utilizada tinha três camadas com 20x20x1 neurônios. Esse treinamento demorou 7 horas, com 7758 épocas, e apresentou o comportamento demonstrado na Figura 6.1.a. Os parâmetros usados no treinamento podem ser observados no programa listado Apêndice C. O treinamento foi executado até que o erro quadrático médio na saída da rede atingisse um valor menor ou igual a 0,01.

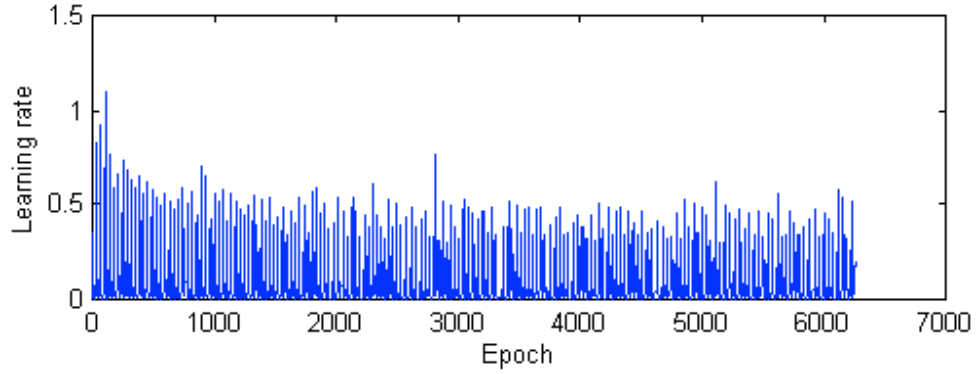
A título de comparação, também foram feitas experiências com uma rede menor, de 10x10x1 neurônios, cujo o resultado do treinamento está apresentado na Figura 6.1.b. Essa rede foi treinada após 6249 épocas, num total de 3 horas e 10 minutos .





(a)





(b)

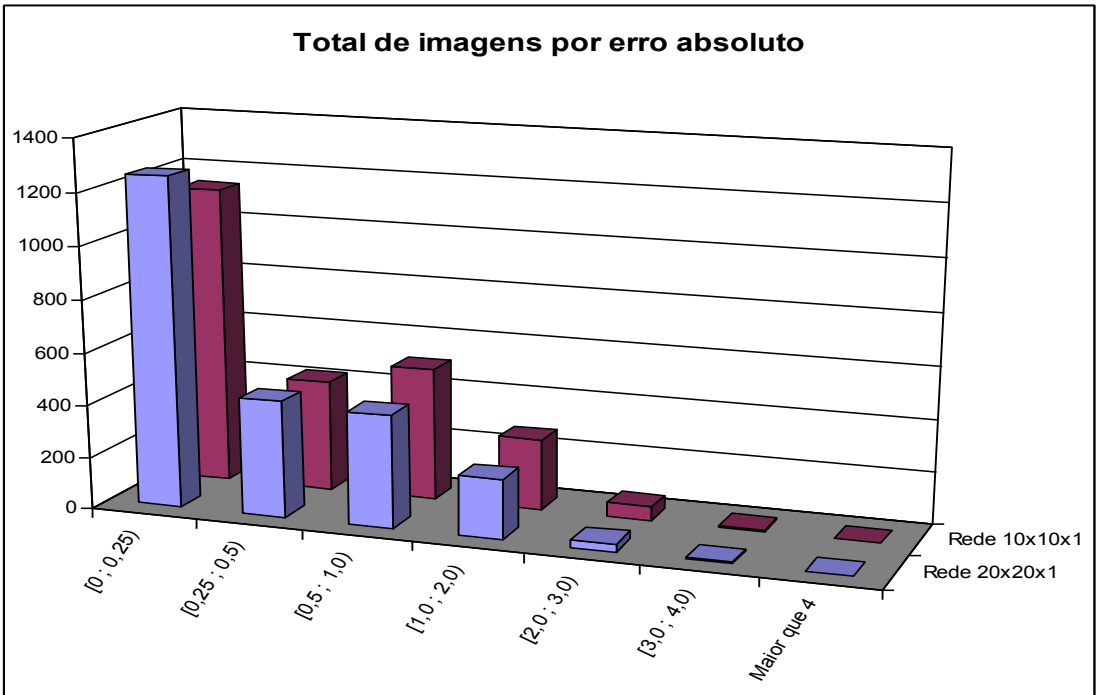
Figura **Error! Style not defined.**.1: (a)Resultado do treinamento da rede neural feed-forward com 3 camadas 20x20x1; (b) Idem para uma rede 10x10x1

Após o treinamento foram realizadas simulações para averiguar a capacidade das redes em calcular resultados a partir de novas entradas. Os resultados, obtidos pela metodologia explícita no capítulo 5, demonstraram que ambas as redes são capazes de estimar o volume de carros numa sequência de fotogramas. A tabela 6.1 apresenta, os resultados do erro quadrático médio absoluto e normalizado (erro dividido pelo número de carros na imagem) obtido nas simulações. Esses resultados foram obtidos pelo programa descrito no Apêndice D.

	Erro absoluto médio por lote		Erro normalizado médio por lote	
Rede	**Treinamento**	**Simulação**	**Treinamento**	**Simulação**
20x20x1	0,0182	0,4067	0,0035	0,1972
10x10x1	0,0154	0,4623	0,0032	0,2185

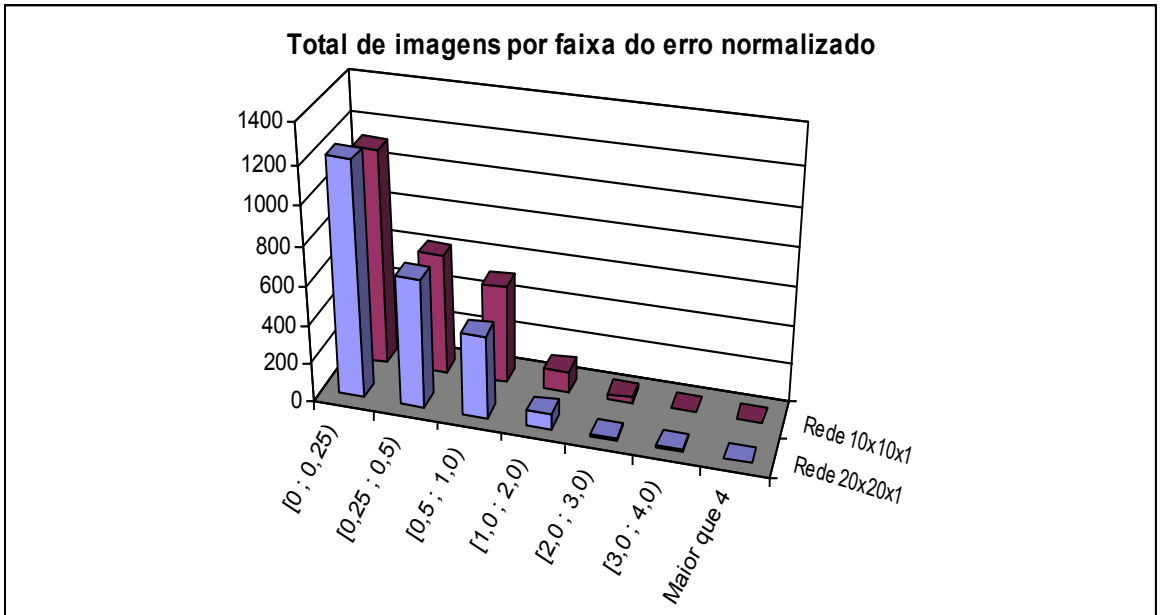
Tabela **Error! Style not defined.**.1: Comparação dos erros obtidos pelos resultados dos testes com lotes de treinamento (800 imagens) e simulação (2400 imagens)

Uma outra avaliação dos resultados pode ser apreciada nas figuras a seguir, onde vêem-se uma classificação dos erros absolutos (Figura 6.2) e normalizados (Figura 6.3) por faixas. Essa classificação mostra qual o total de imagens do lote de simulação em que erro do resultado calculado pela rede é menor que um determinado valor.



	[0 ; 0,25)	[0,25 ; 0,5)	[0,5 ; 1,0)	[1,0 ; 2,0)	[2,0 ; 3,0)	[3,0 ; 4,0)	Maior que 4
Rede 20x20x1	1256	446	431	229	30	7	1
Rede 10x10x1	1139	425	506	270	52	6	2

Figura **Error! Style not defined.**.2: Distribuição do número de imagens por faixa de erro absoluto no lote de simulação



	[0 ; 0,25)	[0,25 ; 0,5)	[0,5 ; 1,0)	[1,0 ; 2,0)	[2,0 ; 3,0)	[3,0 ; 4,0)	Maior que 4
Rede 20x20x1	1215	663	427	76	12	6	1
Rede 10x10x1	1120	622	508	111	27	10	2

Figura **Error! Style not defined.**.3: Distribuição do número de imagens por faixa de erro normalizado no lote de simulação

Uma outra análise pode também ser realizada se catalogarmos os erros nas imagens com o mesmo número de carros. Os resultados obtidos dessa análise foram obtidos pelo programa listado no apêndice E e estão representados nas tabelas a seguir.

	Faixa de erro absoluto								
# carros	<0,25	[0,25;0,5)	[0,5;1)	[1,0;2,0)	[2,0;3,0)	[3,0;4,0)	> 4,0	# imagens	%
0	634	0	0	0	0	0	0	634	26,42%
1	470	318	216	50	3	0	1	1058	44,08%
2	96	79	133	67	6	3	0	384	16,00%
3	30	30	46	61	5	1	0	173	7,21%
4	16	15	28	36	11	2	0	108	4,50%
5	10	4	7	11	4	1	0	37	1,54%
6	0	0	1	3	1	0	0	5	0,21%
7	0	0	0	1	0	0	0	1	0,04%
8	0	0	0	0	0	0	0	0	0,00%
Totais	1256	446	431	229	30	7	1	2400	100,00%
%	52,33%	18,58%	17,96%	9,54%	1,25%	0,29%	0,04%	100,00%	

Tabela **Error! Style not defined.**.2: Classificação dos erros por imagens com as mesmas quantidades de carros (simulação com a rede 20x20x1)

6.2 Comparação com o método anterior

Os resultados obtidos no método anterior [Ribe99] precisam ser manipulados para que possam ser comparados aos resultados do método atual.

A primeira questão diz respeito ao tempo de treinamento da rede. No método anterior não foi possível efetuar o treinamento da rede em uma só etapa devido à limitação de performance do computador consequente da grande massa de dados que foi utilizada. Dessa maneira, foram realizados três treinamentos em sequência, utilizando um lote de 400 imagens no primeiro treinamento (duração de 3 horas), acrescentando 200 imagens no segundo (total de 5 horas) e mais 200 imagens no último (total de 9,5 horas). Desse modo vê-se que o tempo total gasto no processo foi de 17,5 horas. A tabela a seguir compara os resultados obtidos nos dois métodos

Método	Anterior	Atual 20x20x1	Atual 10x10x1
Tempo de treinamento (hs)	17:30'	7	3hs e 10min

Tabela **Error! Style not defined.**.3: Comparação dos tempo de treinamento dos métodos anterior e atual

A segunda questão refere-se ao erro de modelagem da rede obtida. Pela mesma razão relatada no parágrafo anterior, a simulação da rede no método de Ribeiro foi realizada dividindo-se o lote de 2400 imagens de teste em 6 lotes com 400 imagens cada. Os valores de erro obtidos são dispostos na tabela abaixo, juntamente com os valores do método atual relatados no item 6.1.

	Erro absoluto médio por lote		Erro normalizado médio por lote	
Método	**Treinamento**	**Simulação**	**Treinamento**	**Simulação**
anterior	0,0353	0,8000	0,0028	0,2496
atual 20x20x1	0,0182	0,4067	0,0035	0,1972
atual 10x10x1	0,0154	0,4623	0,0032	0,2185

Tabela **Error! Style not defined.**.4: Comparação dos erros obtidos pelos resultados dos testes usando os métodos anterior e atual

7

Conclusões

A partir dos resultados obtidos e demonstrados no capítulo anterior pode-se concluir que a combinação das ferramentas Wavelet, Tratamento de Imagens e Redes Neurais configura uma solução para a estimação do volume de tráfego de veículos a partir de uma sequência de fotogramas. Os erros obtidos pelo método mostraram-se aceitáveis quando o interesse é a comparação do volume de tráfego entre duas ou mais vertentes de tráfego.

A nova combinação de ferramentas também apresentou uma significativa redução do custo computacional quando comparada ao método anterior que combinava apenas Análise de Imagens e Redes Neurais. Essa redução se apresentou na forma de uma significativa economia na quantidade de memória gasta para armazenar as imagens tratadas e também no menor tempo de processamento do algoritmo de treinamento da rede.

Cabe ressalvar que o método não se mostrou adequado para um estimação precisa do número de veículos em uma imagem, já que eventualmente acontecem erros de estimação muito significativos. O que se pôde observar foi a sua utilidade na estimação do fluxo de veículos a partir de uma sequência de imagens estáticas. Nesse sentido, pode-se afirmar que os erros obtidos são perfeitamente aceitáveis para um grande número de aplicações prática, tal como, por exemplo, a captação de dados para um sistema inteligente de temporização de semáforos.

Muitos estudos podem ser feitos a partir dos resultados conseguido té aqui. Na ferramenta wavelet pode-se avaliar a utilização de outras wavelets mãe para averiguar a alteração na performance do método. Também pode-se averiguar se uma aproximação maior, nível 3 ou 4, ainda manterá uma quantidade de informação suficiente para ser detectada pela rede.

Ainda com relação à ferramenta wavelet é possível implementar a detecção de bordas diretamente a partir dos coeficientes obtidos pela DWT, eliminando-se o passo da detecção de bordas e melhorando a performance do algoritmo.

No tratamento de imagens é possível estudar outras formas de equalização que possam melhorar ainda mais o contraste das imagens. Também pode-se alterar os parâmetros de sensibilidade do método detecção de bordas, ou mesmo alterar esse método, visando uma melhoria no resultado do tempo de treinamento da rede.

Por fim, a rede neural utilizada pode ser melhor treinada apresentando-se um conjunto maior e mais representativo de imagens. Além do mais, os parâmetros de treinamento podem ser afinados para otimizar o tempo de treinamento.

Uma vez definido o algoritmo final, todas as ferramentas utilizadas podem ser ainda desenvolvidas em uma linguagem de programação mais otimizada (C ou FORTRAN) evitando-se com isso as desvantagens de performance e consumo de memória do ambiente de desenvolvimento do MATLAB.

Apêndice A

Arquivos de listas de imagens

Arquivos com listagem das figuras e respectivos número de carros. No trabalho foram montados dois arquivos dessa natureza. O primeiro, denominado **`list123_800.txt`**, contem a listagem com nome das 800 imagens usadas no treinamento da rede neural. O segundo, denominado **`list4-9_2400.txt`**, contem a listagem com nome das 2400 imagens usadas na simulação da rede neural.

A listagem a seguir apresenta as linhas iniciais do arquivo **`list123_800.txt`**.

```
000000t1.bmp
0
000000t2.bmp
0
000000t3.bmp
0
000000t4.bmp
0
000000t5.bmp
0
000000t6.bmp
0
000000t7.bmp
0
:
:
:
```

Apêndice B

Programa para criação das matrizes de dados para a RNA

Listagem do programa **`makept.m`** responsável pela criação das matrizes que serão aplicadas à rede neural.

```
% MAKEPT - Create input matrix and target vector for neural net
% based on DWT, image equalization and edge detection
%
% imglist - Text file with the list of images
% imgdir  - Directory of images
% imgpt   - MATLAB file with results
% p       - input matrix for neural net
% t       - target vector from neural net

%   Angelo A. Duarte, 01-20-2000
%   Copyright © 2000 by Angelo Duarte
clc
% Initialization
p = [];
t = [];
i1 = round((53-(1:40))/5.55)+1;  % Upper part that will be discarded
i2 = round((117+(1:40))/5.55)-1; % Lower part that will be discarded
wtlevel = 2;                     % Wavelet Transform Level
wavelet = 'haar';                % Wavelet type
fid = fopen(imglist);            % File with list of images
fimg = fgetl(fid);               % Get first image file name
cimg = fgetl(fid);               % Get number of cars in the image
image = 0;                       % Image coounter
while cimg ~= -1                 % Process until EOF
  image = image + 1;             % ith input
  [X,map] = bmpread(strcat(imgdir,'\',fimg));   % Read image
  t(:,image) = str2num(cimg);
  home
  [fimg '   ' imglist '   ' imgpt '   ' num2str(image)] % Print message
  X = ind2gray(X,map);           % Transform to gray level matrix
  X = imadjust(X,[0.05 0.5]);    % Image equalization
  [C,S] = wavedec2(X,wtlevel,wavelet);  % Wavelet Transform
  X  = appcoef2(C,S,wavelet,wtlevel);   % Calculates de approximation
coeficients
  X = edge(X,0.5,'sobel');       % Edge extraction
  P = [];                        % Create na empty column vector
  for j = 1:40
    P = [P ; X(i1(j):i2(j),j)];  % Fill column vector based on image's
useful part
  end
  p(:,image) = P;                % Build the input matrix for neural net
  fimg = fgetl(fid);             % Get next image file name
  cimg = fgetl(fid);             % Get number of cars in the image
end
t = (t+1)/10;                    % t values in [0,1] range
```

```
fclose(fid);                        % Close file
clear ans X map i1 i2 image j fid fimg cimg C S P wavelet wtlevel
save(imgpt,'p','t')                 % Save results
```

Apêndice C

Programa de treinamento da RNA

Programa **`train3lay.m`** usado no treinamento da rede neural.

```
% TRAIN3LAY - Train neural net using backpropagation
% p - Input matrix (717xN), N = total of images
% t - Target vector

%   Angelo A. Duarte, 01-20-2000
%   Copyright (c) 2000 by Angelo Duarte

clc
begintime = datestr(datenum(now),0)  % Store begin time

% Neural functions
f1 = 'logsig';        % 1st layer
f2 = 'logsig';        % 2nd layer
f3 = 'logsig';        % 3rd layer

% Net initialization
mm = repmat([0 1],size(p,1),1);
[w1,b1,w2,b2,w3,b3] = initff(mm,10,f1,10,f2,1,f3);
clear mm;

% Net training
tp = [1 10000 0.01 0.3 1.2 0.3 0.9 1.01];
[w1,b1,w2,b2,w3,b3,te,tr] = trainbpx(w1,b1,f1,w2,b2,f2,w3,b3,f3,p,t,tp);

endtime = datestr(datenum(now),0)     % Store end time

save train3laylt1    % Save results
```

Apêndice D

Programa para simulação e contagem de erros

Programa **sim3lay.m** usado na simulação da rede neural e cálculo dos erros.

```
% SIM3LAY - Validate the neural net obtained by TRAIN3LAY program and
% calculate the errors
% ptrain - input matrix used on training
% psim   - input matrix for simulation
% ttrain - target vector used on training
% tsim   - correct target vector used for compare the result of the
simulation

%   Angelo A. Duarte, 01-20-2000
%   Copyright (c) 2000 by Angelo Duarte

% Simultion with the same images used on training
atrain = simuff(ptrain,w1,b1,f1,w2,b2,f2,w3,b3,f3);
etrain = (atrain - ttrain)*10;
absetrain = abs(etrain);
metrain = sum(abs(etrain))/800  % Mean error for this group of images

% Simulation with new images
assim = simuff(psim,w1,b1,f1,w2,b2,f2,w3,b3,f3);
esim = (assim - tsim)*10;
absesim = abs(esim);
mesim = sum(absesim)/2400  % Mean error for this group of images

% Ranges of absolute errors
mEd = [0 0 0 0 0 0 0];
for j = 1:2400,
  if absesim(j) < 0.25;,
    mEd(1) = mEd(1)+1;,
  elseif absesim(j) < 0.5;,
    mEd(2) = mEd(2)+1;,
  elseif absesim(j) < 1;,
    mEd(3) = mEd(3)+1;,
  elseif absesim(j) < 2;,
    mEd(4) = mEd(4)+1;,
  elseif absesim(j) < 3;,
    mEd(5) = mEd(5)+1;,
  elseif absesim(j) < 4;,
    mEd(6) = mEd(6)+1;,
  else
    mEd(7) = mEd(7)+1;,
  end
end
mEd

% Ranges of normalized errors
Nmetrain = absetrain/(ttrain*10)
```

```
Nmesim = absesim/(tsim*10)
NmEd=[0 0 0 0 0 0 0];
for j = 1:2400,
  Nabsesim(j) = absesim(j)/(tsim(j)*10);
  if Nabsesim(j)<0.10;,
    NmEd(1)=NmEd(1)+1;,
  elseif Nabsesim(j)<0.25;,
    NmEd(2)=NmEd(2)+1;,
  elseif Nabsesim(j)<0.5;,
    NmEd(3)=NmEd(3)+1;,
  elseif Nabsesim(j)<0.75;,
    NmEd(4)=NmEd(4)+1;,
  elseif Nabsesim(j)<1;,
    NmEd(5)=NmEd(5)+1;,
  elseif Nabsesim(j)<1.5;,
    NmEd(6)=NmEd(6)+1;,
  else
    NmEd(7)=NmEd(7)+1;,
  end
end
NmEd
```

Apêndice E

Programa para classificação dos erros

Programa **sim3lay1.m** usado na classificação dos erros nas imagens com os mesmos números de carros.

```
% SIM3LAY1 - Calculate the errors in images with the same number of cars
% psim   - input matrix for simulation
% tsim   - correct target vector used for compare the result of the
simulation

%   Angelo A. Duarte, 01-20-2000
%   Copyright (c) 2000 by Angelo Duarte

% Simulation with new images
asim = simuff(psim,w1,b1,f1,w2,b2,f2,w3,b3,f3);
esim = (asim - tsim)*10;
absesim = abs(esim);

% Ranges of absolute errors
mEd = zeros(9,7);
t = tsim*10;
for j = 1:2400,
  if absesim(j) < 0.25,
    mEd(t(j),1) = mEd(t(j),1)+1;,
  elseif absesim(j) < 0.5,
    mEd(t(j),2) = mEd(t(j),2)+1;,
  elseif absesim(j) < 1,
    mEd(t(j),3) = mEd(t(j),3)+1;,
  elseif absesim(j) < 2,
    mEd(t(j),4) = mEd(t(j),4)+1;,
  elseif absesim(j) < 3,
    mEd(t(j),5) = mEd(t(j),5)+1;,
  elseif absesim(j) < 4,
    mEd(t(j),6) = mEd(t(j),6)+1;,
  else
    mEd(t(j),7) = mEd(t(j),7)+1;,
  end
end
mEd
```

Bibliografia

[Akay98] AKAY, Metin, *Time frequency and wavelets in biomedical signal processing*, IEEE Press Series in Biomedical Engineering, IEEE , New York, NY, 1998

[Azev00] AZEVEDO, Fernando M. de, BRASIL, Lourdes M., OLIVEIRA, Roberto C. L. de, *Redes neurais com aplicações em controle e sistemas especialistas*, Bookstore, Forianópolis, Santa Catarina, 2000

[Bron00] BRONZINO, Joseph D., *The Biomedical Engineering Handbook*, 2 ed., CRC Press LLC, Boca Raton, Florida, 2000

[Burr98] BURRUS, C. Sidney, GOPINATH, Ramesh [a], GUO, Haitao, *Introduction to wavelets and wavelet transforms:* A primer, Prentice Hall, Inc., Upper Saddle River, New Jersey, 1998

[Cand88] CANDY, James V., *Signal processing: the modern approach*, International Edition 1988, McGraw-Hill Book Co., Singapore

[Daub92] DAUBECHIES, Ingrid, *Tem lectures on wavelets*, SIAM – Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania, 1992

[Galv99] GALVÃO, Roberto K. H., *Wavelet-based techniques for adaptive feature and pattern recognition*, Tese de Doutorado, ITA, São José dos Campos, São Paulo, 1999

[Gose96] GOSE, Earl, JOHNSONBAUGH, Richard, JOST, Steve, *Pattern recognition & image analysis*, Prentice Hall, Upper Saddle River, New Jersey, 1996

[Grap95] GRAPS, Amara, *An introduction to wavelets*, IEEE Computational Science and Engineering, vol. 2, num. 2, Summer 1995, IEEE Computer Society, Los Alamitos, CA

[Hans96] HANSELMAN, Duane C., LITTLEFIELD, Bruce, *Mastering MATLAB®:*A comprehensive tutorial and reference, Prentice Hall, Inc., Upper Saddle River, New Jersey, 1996

[Hayk94] HAYKIN, S., *Neural Networks: A comprehensive foundation*, New York, Macmillan Publishing, 1994

[Kova96] KOVÁCS, Zsolt L., Redes neurais artificiais: fundamentos e aplicações, 2. ed., Edição Acadêmica, São Paulo, SP, 1996

[Kuma99] KUMAR, V. Ravi, et. al., *Simple denoising algorithm using wavelet transform,* AIChE Journal, Vol.45, n. 11, p. 2461-2466, 1999

[Lewa98] LEWALLE, Jacques, *Wavelet without lemmas*, Lecture Series on Applications of Continuos Wavelets to Data Analysis, Syracuse University, 6-8 April 1998

[Misi96] MISITI, Michel, et. al., *Wavelet toolbox for use with MATLAB*®, The MathWorks, Inc., Natick, Massachusets, March 1996

[Ribe99] RIBEIRO, Fernando F. S., *Detecção de volume de tráfego de veículos proporcionada por visão computacional via redes neurais*, Dissertação de Mestrado, Departamento de Engenharia Elétrica da UFBA, Salvador, Bahia, Janeiro de 1999

[Thom95] THOMPSON, Clay M., SHURE, Loren, *Image processing toolbox for use with MATLAB*®, The MathWorks, Inc., Natick, Massachusets, 1995

[Vett92] VETTERLI, M., HERLEY, C., *Wavelets and filter banks: theory and design*, IEEE transactions on Signal Processing, Vol. 40, 1992, pp. 2207-2232