UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE CIÊNCIAS E TECNOLOGIA
COORDENAÇÃO DE PÓS-GRADUAÇÃO EM INFORMÁTICA

Um Sistema de Visão Inteligente para Detecção e Reconhecimento de Peças em um Tabuleiro de Xadrez em Tempo Real

Sérgio Faustino Ribeiro

Campina Grande – PB
Julho de 2001

Um Sistema de Visão Inteligente para Detecção e Reconhecimento de Peças em um Tabuleiro de Xadrez em Tempo Real

Sérgio Faustino Ribeiro

Dissertação de Mestrado submetida à Coordenação de Pós-graduação em Informática – COPIN – da Universidade Federal da Paraíba – Campus II, como parte dos requisitos necessários à obtenção do grau de Mestre em Informática.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Visão Computacional e Inteligência Artificial

José Homero Feitosa Cavalcanti – Ph. D.
Orientador

Campina Grande, Paraíba, Brasil
Julho de 2001

RIBEIRO, Sérgio Faustino

R484S

Um Sistema de Visão Inteligente para Detecção e Reconhecimento de Peças em um Tabuleiro de Xadrez em Tempo Real

Dissertação (Mestrado) - Universidade Federal da Paraíba, Centro de Ciências e Tecnologia, Coordenação de Pós-Graduação em Informática, Campina Grande, Pb, Julho de 2001.

110 p. Il.

Orientador: José Homero Feitosa Cavalcanti

Palavras Chaves:

1. Inteligência Artificial
2. Visão Computacional
3. Redes Neurais Artificiais

CDU - 007.52

UM SISTEMA DE VISÃO INTELIGENTE PARA DETECÇÃO E RECONHECIMENTO DE PEÇAS EM UM TABULEIRO DE XADREZ EM TEMPO REAL

SÉRGIO FAUSTINO RIBEIRO

DISSERTAÇÃO APROVADA EM 20.07.2001

Homero Cavalcanti

PROF. JOSÉ HOMERO FEITOSA CAVALCANTI, D.Sc
Orientador

J. Felício da S.

PROF. JOSÉ FELÍCIO DA SILVA, Dr.
Examinador

Guilherme

PROF. GUILHERME VILAR , Dr.
Examinador

CAMPINA GRANDE – PB

Agradecimentos

Agradeço, primeiramente, a Deus por ter concluído este trabalho e por gozar de plena saúde necessária à luta diária diante dos obstáculos e dos desafios da vida.

Ao Prof. José Homero Feitosa Cavalcanti pela orientação e por acreditar em minha capacidade de contribuir no desenvolvimento deste projeto.

À Prof[a]. Fernanda Cecília C. L. Loureiro do DEE/UFPB por aceitar a realização do estágio-docência em sua disciplina, o que propiciou obter conhecimentos mais específicos em Processamento de Sinais.

Aos colegas de laboratório, principalmente Tarig, Juracy e Alexandre, pelo companheirismo e conversas esclarecedoras.

Agradeço principalmente aos meus pais Faustino e Isabel pelo apoio incondicional desde o princípio, sem o qual este trabalho não teria início. Agradeço também aos meus irmãos Fernando e Malan pelo companheirismo diário, e a Paulo que mesmo de longe demonstrou seu apoio.

À COPIN/UFPB pela confiança depositada e à CAPES que proporcionou o suporte financeiro deste trabalho.

Resumo

Esta dissertação de mestrado apresenta o desenvolvimento de um sistema de visão inteligente, baseado em redes neurais artificiais, desenvolvido para detectar e reconhecer peças de um tabuleiro de xadrez em tempo real. Para isso foi implementado um software, denominado NeuroMorfo, capaz de realizar diversas tarefas como aquisição de imagens em tempo real, processamento de imagens, detecção *on-line* e *off-line* de peças de xadrez, e reconhecimento *off-line* destas peças. No caso do sistema de detecção, foram feitas modificações no sistema original, resultando em um novo algoritmo de detecção com conceitos do Sistema de Previsão Inteligente (SPI). Com relação ao sistema de reconhecimento, foram realizados estudos referentes às etapas necessárias ao processo de reconhecimento de padrões que possibilitaram seu desenvolvimento. O desempenho dos sistemas de detecção e reconhecimento é avaliado a partir de uma série de testes realizados.

Abstract

This dissertation presents the development of a intelligent vision system, based on artificial neural networks, developed to detect and to recognize pieces of a chessboard in real time. A software, denominated NeuroMorfo, was implemented to accomplish several tasks as acquisition of images in real time, image processing, detection *on-line* and *off-line* of chess pieces, and *off-line* recognition of these pieces. Regarding the detection system, modifications were made in the original system, resulting in a new detection algorithm with concepts of the Intelligent Forecast System (SPI). Regarding the recognition system, studies were accomplished related to the necessary stages to the process of pattern recognition that facilitated its development. The performance of the detection and recognition systems is evaluated starting from a series of accomplished tests.

Índice

Lista de Figuras

Lista de Tabelas

Lista de Símbolos e Siglas

Símbolos

β	-	Representa a declividade da sigmóide
τ	-	Representa o deslocamento da sigmóide
θ	-	Valor máximo da sigmóide
ε	-	Limite de erro final
$\varnothing$	-	Conjunto vazio
2D	-	Bidimensional
3D	-	Tridimensional
8TD	-	Casa do tabuleiro situada na oitava linha e primeira coluna
8CD	-	Casa do tabuleiro situada na oitava linha e segunda coluna
8BD	-	Casa do tabuleiro situada na oitava linha e terceira coluna
$f(x,y)$	-	Função bidimensional para descrição de imagens
G	-	Garra do AutoXad
L	-	Parte constituinte do braço robótico
M	-	Motor de passo
X_i	-	Entradas da rede
W_i	-	Conjunto de pesos
Z^2	-	Espaço inteiro bidimensional
Z^3	-	Espaço inteiro tridimensional

Siglas

APR	-	Algoritmo de Propagação Retroativa (do erro)
ASCII	-	American Standard Code for Information Interchange
BFH	-	Bitmap File header
BGI	-	Borland Graphics Interface
BIH	-	Bitmap Info Header
BIOS	-	Basic Input/Output System
BMP	-	Extensão para arquivos Bitmap
bpp	-	bits por polegada
CAD	-	Computer Aided Design
CCD	-	Charge Coupled Device
cdf	-	Cumulative Distribution Function
CEP	-	Código de Endereçamento Postal
CJT	-	Conjunto Janela de Treinamento
CPU	-	Central Processing Unit
DARPA	-	Defense Advanced Research Projects Agency
DOS	-	Disk Operating System
DSC	-	Departamento de Sistemas e Computação
ECP	-	Extended Capability Port
ENIAC	-	Eletronic Numerical Integrator and Calculator
EPP	-	Extended Parallel Port
FPN	-	Filtro Previsor Neural
IA	-	Inteligência Artificial
IBM	-	International Business Machine
IDE	-	Interface Development Environment
IEEE	-	Institute of Electrical and Electronic Engineers
INNS	-	International Neural Networks Society
I/O	-	Input/Output
LPT	-	Line Printer (porta de impressora)
Madaline	-	Multiple Adaptive Element
MS-DOS	-	Microsoft Disk Operating System
NEUROLAB	-	Laboratório de Redes Neurais e Automação Inteligente
NTSC	-	National Television System Committee
OCR	-	Optical Character Recognition
P/B	-	Preto e Branco

PC	-	Personal Computer
p.u.	-	por unidade (normalização)
RAD	-	Rapid Application Development
RAM	-	Random Access Memory
RGB	-	Componentes de cor (Red, Green, Blue)
RLE	-	Run Length Encoded
RNA	-	Rede Neural Artificial
RNMC	-	Rede Neural Multicamada
ROM	-	Read Only Memory
SCV	-	Sinal Composto de Vídeo
SPI	-	Sistema de Previsão Inteligente
SPP	-	Standard Parallel Port
XOR	-	Operação lógica Ou Exclusivo
TRC	-	Tubo de Raios Catódicos
VLSI	-	Very Large Scale Integration

Capítulo 1

Introdução

1.1 Objetivos do trabalho

Esta dissertação tem dois objetivos principais:

a) desenvolver um sistema de visão computacional que utilize redes neurais artificiais para detecção da posição de cada peça no tabuleiro de xadrez em tempo real, mesmo em um ambiente pouco iluminado, demonstrando, conseqüentemente, a robustez do sistema, e que possa ser capaz de enviar sinais de controle para um sistema robótico a partir da percepção de seu ambiente de trabalho;
b) e, por último, desenvolver um sistema de reconhecimento que seja capaz de reconhecer cada uma das seis peças de xadrez distintas, utilizando como classificador uma rede neural multicamada.

1.2 Objetivos específicos

O projeto a ser desenvolvido pretende atingir alguns objetivos específicos, tais como:

1. desenvolver o software NeuroMorfo para que possa capacitar o sistema de visão a atingir seu objetivo final. Isto envolve projetar um sistema de previsão inteligente adequado e eficiente para o sistema de visão.

2. fazer o sistema de visão detectar as peças do tabuleiro em tempo real.
3. implementar um sistema de reconhecimento das peças de xadrez de forma simples e interativa com o usuário. Isto envolve implementar adequados algoritmos de segmentação e extração de característica, além de uma apropriada configuração da rede neural artificial para o reconhecimento das peças.
4. testar exaustivamente o sistema de visão tanto para o sistema de detecção quanto para o sistema de reconhecimento das peças.
5. elaborar relatórios técnicos com a finalidade de documentar progressos efetivos em cada etapa de desenvolvimento do sistema de visão.

1.3 Justificativa

Várias pesquisas em robótica e inteligência artificial já foram desenvolvidas no Laboratório de Redes Neurais e Automação Inteligente (NEUROLAB), dando origem a uma ampla teoria no controle de estruturas robóticas. Por isto, justifica-se, atualmente, o desenvolvimento de procedimentos que viabilizem a implementação de sistemas de visão computacional inteligentes, os quais têm por objetivo auxiliar um sistema robótico, enviando sinais de controle gerados a partir da percepção do ambiente.

1.4 Contribuição

Diversos robôs desenvolvidos tanto em laboratórios universitários como na indústria utilizam somente sensores ópticos de posição dos eixos dos motores elétricos de corrente contínua. Obviamente, a não existência de sensores de visão limita a operação destes robôs. A maior contribuição esperada neste projeto é apresentar um sistema de visão que sirva como passo inicial para auxiliar os sensores ópticos. O resultado esperado consiste na simplicidade e na ampliação do potencial dos robôs desenvolvidos com o amparo de um sistema de visão computacional.

1.5 Material utilizado

Para desenvolver este projeto de dissertação foi necessário utilizar alguns materiais e equipamentos específicos:

- tabuleiro de xadrez com 8x8 casas (27mm^2 por casa);
- 32 peças de xadrez;
- câmera digital QuickCam P/B;
- microcomputador PENTIUM 350 MHz, com 64 MB de RAM.

1.6 Organização da dissertação

Esta dissertação é constituída por seis capítulos e dois apêndices. O primeiro capítulo descreve o hardware e o software utilizados para desenvolver o projeto, os objetivos da dissertação assim como a justificativa, a contribuição do trabalho e o material utilizado.

O capítulo 2 realiza um estudo a respeito de processamento digital de imagens incluindo definições de histograma, equalização, binarização, extração de contorno, filtragem e morfologia matemática. O capítulo é finalizado com um pequeno resumo da importância que o processamento digital de imagens possui por fornecer um conjunto de dados significativo para o correto reconhecimento do objeto via redes neurais artificiais.

Apresenta-se no capítulo 3 um breve estudo sobre redes neurais artificiais, descrevendo os seus fundamentos, a sua capacidade de aprender e suas aplicações. Encerra-se o capítulo destacando-se a importância que a ferramenta possui para o perfeito funcionamento do sistema de detecção e reconhecimento desenvolvidos.

No capítulo 4 comenta-se em detalhes o desenvolvimento do sistema de detecção das peças sobre o tabuleiro de xadrez utilizando o sistema de previsão inteligente como ferramenta principal para detectar as peças. Realiza-se uma comparação com o sistema de detecção original desenvolvido anteriormente ao início do trabalho de dissertação. O capítulo é concluído com uma breve análise do desempenho do sistema desenvolvido e do sistema original.

O capítulo 5 aborda o desenvolvimento do sistema de reconhecimento, destacando-se o uso de um conjunto de treinamento específico para treinar a rede neural e o desenvolvimento de

algoritmos necessários para a segmentação da peça de xadrez e para a geração do vetor característica. O capítulo é finalizado com uma análise das dificuldades encontradas e de possíveis melhorias ao sistema.

Finalmente conclui-se a dissertação com o capítulo 6, no qual é feita uma análise de todo o trabalho realizado para implementar o sistema de detecção *on-line* e *off-line* e o sistema de reconhecimento *off-line*. Encerra-se a conclusão apresentando algumas contribuições realizadas e sugestões de trabalhos futuros.

Os apêndices A e B fazem um estudo sobre dois sistemas de apoio ao desenvolvimento do projeto: o sistema de processamento em tempo real, e o sistema de programação, ou seja, a plataforma de desenvolvimento do software.

Capí tulo 2

Processamento Digital de Imagens

O principal objetivo deste capítulo é realizar um estudo sobre processamento de imagens descrevendo as diversas etapas envolvidas para melhorar uma imagem utilizando técnicas de processamento digital de imagens. Técnicas como digitalização, detecção de bordas, equalização, e histograma são abordadas. No final faz-se uma breve introdução ao uso de redes neurais artificiais como classificador de padrões.

2.1 Introdução

Duas das principais áreas de aplicação foram as responsáveis pelo interesse em métodos de processamento digital de imagens: correções e aprimoramento de imagens para interpretação humana, e análise automática por computador de informações extraídas de uma cena. Uma das primeiras aplicações de técnicas de processamento de imagens foi em melhorar figuras digitalizadas de jornal enviadas por cabo submarino entre Londres e Nova Iorque [FN99].

O interesse e as pesquisas na área de processamento de imagens cresceram expressivamente da década de 60 aos dias atuais. Além de aplicações em programas espaciais, técnicas de processamento digital de imagens agora são utilizadas para resolver uma variedade de problemas. Estes problemas comumente requerem métodos capazes de obter informações de imagem para

interpretação e análise humana. Em medicina, por exemplo, procedimentos computacionais aumentam o contraste ou codificam a intensidade de níveis em cor para uma interpretação mais fácil de radiografias e outras imagens biomédicas. Geógrafos usam técnicas semelhantes a esta para estudar padrões de poluição através de imagens aéreas ou por satélite. Procedimentos de realce e restauração de imagens são utilizados para processar imagens degradadas de objetos irrecuperáveis. Técnicas de restauração de imagens auxiliam arqueologistas a recuperar fotos borradas de artefatos raros, algumas vezes já destruídos. Em física, técnicas computacionais habitualmente realçam imagens de experiências em áreas como plasmas de alta energia e microscópio eletrônico.

A segunda maior área de aplicação de técnicas de processamento digital de imagens está em solução de problemas relacionados com percepção de máquina. Neste caso, o interesse se focaliza em procedimentos para extração de uma informação de imagem para processamento computacional. Freqüentemente, esta informação pouco se assemelha às características visuais que os seres humanos usam para interpretar o conteúdo de uma imagem. Exemplos deste tipo de informação usada em percepção de máquina são momentos estatísticos, coeficientes da transformada de Fourier, e medidas de distância.

Problemas típicos em percepção de máquina que geralmente utilizam técnicas de processamento de imagens são encontrados em reconhecimento automático de caracteres, visão de máquina industrial para montagem e inspeção de produtos, processamento automático de impressão digital, filtragem radiográfica e amostras de sangue.

2.2 Representação da imagem digital

A palavra *imagem* ou *imagem monocromática* pode ser descrita matematicamente por uma função bidimensional $f(x,y)$ que representa a intensidade luminosa. As variáveis x e y representam as coordenadas espaciais e o valor de f em um ponto qualquer (x,y) é proporcional ao brilho (ou nível de cinza) da imagem naquele ponto.

A imagem digital é uma imagem $f(x,y)$ que foi amostrada em coordenadas espaciais e brilho. Uma imagem digital pode ser considerada uma matriz cujos índices de linha e de coluna identificam um ponto na imagem e o valor do elemento na matriz identifica o nível de cinza naquele ponto. Os elementos de tal formação digital são chamados elementos de imagem, elementos de figura, pixels, ou pels, com os dois últimos sendo comumente abreviações usadas de *elementos de figura*.

2.3 Etapas do processamento de imagens

Como os computadores somente podem processar imagens digitais, e a natureza fornece imagens analógicas, um pré-requisito para processamento digital de imagens é a conversão de imagens na forma digital. Esta seção aborda as diversas etapas para se ter uma imagem digital de fácil análise humana.

2.3.1 Aquisição da imagem

O primeiro passo no processo de reconhecimento é a aquisição de imagens. Para tanto, são necessários um sensor de luz (câmera, por exemplo) e um digitalizador. A aquisição de uma imagem é a conversão de informação óptica em sinal elétrico seguido de uma transformação da imagem analógica em imagem digital.

A conseqüência direta da conversão de uma cena real tridimensional em uma imagem eletrônica é a redução de dimensionalidade. Portanto uma câmera fotográfica ou câmera de vídeo converterá uma cena tridimensional em uma imagem bidimensional adequada.

O dispositivo de aquisição de imagens mais utilizado atualmente é a câmera CCD (*Charge Coupled Device*). Ela consiste de uma matriz de células fotossensíveis que atuam como capacitores, armazenando carga elétrica proporcional à energia luminosa incidente. O sinal elétrico produzido é condicionado por circuitos eletrônicos especializados, produzindo na saída um Sinal Composto de Vídeo (SCV) analógico e monocromático. Para a aquisição de imagens coloridas utilizando CCDs, necessita-se de um conjunto de prismas e filtros de cor encarregado de decompor a imagem colorida em suas componentes RGB (Red, Green, Blue), cada qual capturado por um CCD independente [Gro89].

2.3.2 Digitalização da imagem

O sinal analógico de vídeo, obtido à saída do dispositivo de aquisição, deve ser submetido a processos de discretização. Isto porque uma função contínua não pode ser representada com exatidão em um computador digital. A imagem deve ser amostrada em um número finito de pontos e cada amostra deve ser representada dentro do tamanho de palavra finito definido em um sistema computacional. Isto é o processo de amostragem e quantização. Amostragem é o processo de discretização espacial e quantização é o processo de discretização em amplitude. Cada amostra de

imagem é denominada pixel. Cada pixel é representado no computador como um número inteiro. Freqüentemente, o pixel é representado como um inteiro de 8 bits na faixa [0, 255], com 0 correspondendo ao preto, 255 ao branco, e níveis de cinza distribuídas sobre os valores médios.

Muitas câmeras adquirem uma imagem analógica, que é então amostrada e quantizada para convertê-la em uma imagem digital. A taxa de amostragem determina quantos pixels terá a imagem digital (a resolução de imagem), e a quantização determina quantos níveis de intensidade serão usados para representar o valor de intensidade em cada ponto da amostra. Na maioria das aplicações de visão computacional, as taxas de amostragem e de quantização são predeterminadas devido à limitada disponibilidade de câmeras e hardware de aquisição de imagem; mas em muitas aplicações pode ser importante saber os efeitos da amostragem e quantização.

Há algumas décadas atrás, um equipamento digitalizador era tão dispendioso e complexo que apenas poucos centros de pesquisa eram capazes de utilizá-los. Avanços em tecnologia, entretanto, tornaram os digitalizadores de imagens mais baratos e seu uso mais difundido.

2.3.3 Filtragem e suavização de imagens

O principal objetivo das técnicas de realce de imagens é processar uma certa imagem de modo que a imagem resultante seja mais apropriada que a imagem original, para uma aplicação específica. Conseqüentemente:

- a interpretação de que o resultado é mais adequado, ou não, normalmente é subjetiva e depende do conhecimento prévio do observador a respeito das imagens analisadas.
- As técnicas de realce de imagens são por natureza orientadas a um problema que se deseja resolver. Logo, não existem técnicas capazes de resolver 100% dos problemas que uma imagem digital possa apresentar, como também nem sempre uma técnica que produz bons resultados para imagens biomédicas, adquiridas através de um tomógrafo computadorizado, apresentará desempenho satisfatório, se aplicada, por exemplo, a uma imagem contendo uma impressão digital.

Os métodos de filtragem de imagens são normalmente classificados em duas classes: as técnicas de filtragem espacial e as técnicas de filtragem no domínio da freqüência. Os métodos que trabalham no domínio espacial operam diretamente sobre a matriz de pixels que é a imagem digitalizada, normalmente utilizando operações de convolução com máscaras. Neste caso, duas

imagens X e Y podem ser processadas pixel a pixel, utilizando operadores aritméticos (soma, multiplicação), produzindo uma terceira imagem Z, cujos pixels correspondem ao resultado de X convoluído com Y. Os métodos que atuam no domínio das freqüências se baseiam na modificação da transformada de Fourier da imagem. Existem técnicas de filtragem que combinam ambas as abordagens [FN99].

2.3.4 Detecção de bordas

Há muitos anos o tema "detecção de bordas" (*edge detection*) vem desafiando os pesquisadores da área de Processamento de Imagens e sobre ele continuam sendo experimentadas novas técnicas. Os resultados são publicados ainda hoje nos mais conceituados periódicos científicos mundiais. Trata-se de um tema em aberto principalmente em cenas consideradas "difíceis".

Define-se borda (*edge*) como a fronteira entre duas regiões, cujos níveis de cinza predominantes são razoavelmente diferentes. Pratt [Pra91] define uma borda de luminosidade como uma descontinuidade na luminosidade de uma imagem. Analogamente, pode-se definir borda de textura ou borda de cor, em imagens onde as informações de textura ou cor, respectivamente, são as mais importantes.

Para a detecção e realce de bordas, aplicam-se habitualmente filtros espaciais lineares que costumam ser implementados a partir de máscaras de convolução ou operadores 3 x 3. Exemplos destas máscaras são os operadores de Roberts, Sobel, Prewitt e Frei-Chen.

2.3.5 Conceito de histograma

O histograma de uma imagem (Figura 2.1) é um conjunto de números indicando o percentual de pixels, em uma imagem, que apresentam um determinado nível de cinza. Estes valores são normalmente representados por um gráfico de barras que fornece para cada nível de cinza o número (ou o percentual) de pixels correspondentes na imagem. Através da utilização do histograma de uma imagem obtém-se uma indicação de sua qualidade quanto ao nível de contraste e quanto ao seu brilho médio (se a imagem é predominantemente clara ou escura).

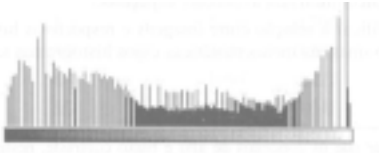


Figura 2.1 – Exemplo de histrograma

Um histograma nada mais é que uma função de distribuição de probabilidade e como tal deve obedecer aos axiomas e teoremas da teoria da probabilidade.

O conceito de histograma também é aplicável a imagens coloridas. Neste caso, a imagem é decomposta de alguma forma (por exemplo, em seus componentes RGB) e para cada componente é calculado o histograma correspondente.

2.3.6 Equalização de histograma

A equalização de histograma (Figura 2.2) é uma técnica a partir da qual se procura redistribuir os valores de tons de cinza dos pixels em uma imagem, de modo a obter um histograma uniforme, no qual o número (percentual) de pixels de qualquer nível de cinza é praticamente o mesmo. Para tanto, utiliza-se uma função auxiliar, denominada função de transformação.

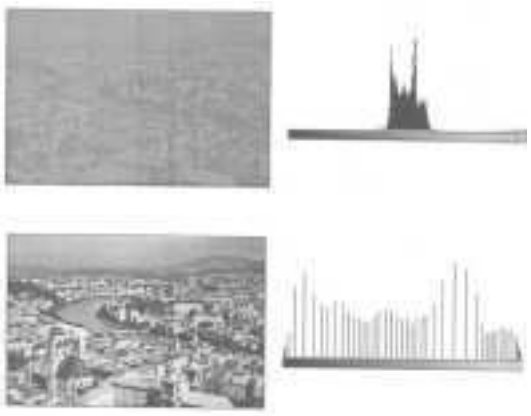


Figura 2.2 – Aplicação da equalização de histograma

A forma mais usual de se equalizar um histograma é utilizar a função de distribuição acumulada (*cdf – cumulative distribution function*) da distribuição de probabilidades original [FN99], que pode ser expressa por:

$$s_k = T(r_k) = \sum_{j=0}^{k} \frac{n_j}{n} = \sum_{j=0}^{k} p_r(r_j) \qquad (2.1)$$

onde:
$0 \le r_k \le 1$

$k = 0,1,...,255$

$T(\ )$ = função de transformação;

n = número total de pixels na imagem;

n_j = número de pixels cujo nível de cinza corresponde a j.

$p_r(r_j)$ = probabilidade do j-ésimo nível de cinza.

O histograma equalizado apresenta melhor distribuição de pixels ao longo da escala de cinza em relação ao original, realçando conseqüentemente, o contraste da imagem. Esta técnica é algumas vezes utilizada para analisar imagens tais como raios X ou fotografias de satélite que podem ter pouco contraste em algumas regiões de interesse.

2.3.7 Limiarização (thresholding)

O princípio da *limiarização* consiste em separar as regiões de uma imagem quando esta apresenta duas classes (o fundo e o objeto). Devido ao fato da *limiarização* produzir uma imagem binária na saída, o processo também é denominado, muitas vezes, *binarização*. A forma mais simples de *limiarização* consiste na bipartição do histograma, convertendo os pixels cujo tom de cinza é maior ou igual a um certo valor limiar (T) em brancos e os demais em pretos, como ilustra a Figura 2.3.

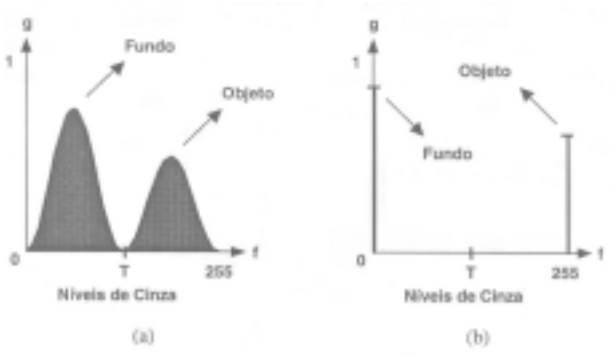


Figura 2.3 – *Limiarização* de uma imagem monocromática utilizando limiar T:
(a) histograma original, (b) histograma da imagem *binarizada*.

No caso de níveis de cinza divididos basicamente em duas classes, onde o histograma apresenta dois picos e um vale, a limiarização é trivial (Figura 2.3).

2.3.8 Morfologia matemática

Assim como na biologia, onde a expressão morfologia se refere ao estudo da forma dos animais e plantas, a morfologia matemática, elaborada inicialmente por Georges Matheron e Jean Serra [Ser82], concentra seus esforços no estudo da estrutura geométrica das entidades presentes em uma imagem. A morfologia matemática pode ser aplicada em várias áreas de processamento e análise de imagens, com objetivos tão distintos como realce, filtragem, segmentação, detecção de bordas, afinamento, dentre outras.

O princípio básico da morfologia matemática consiste em extrair as informações relativas à geometria e à topologia de um conjunto desconhecido (uma imagem), pela transformação através de

outro conjunto completamente definido, denominado elemento estruturante. Portanto, a base da morfologia matemática é a teoria dos conjuntos. Por exemplo, o conjunto de todos os pixels pretos em uma imagem binária descreve completamente a imagem (uma vez que os demais pontos só podem ser brancos). Em imagens binárias, os conjuntos em questão são membros do espaço inteiro bidimensional Z^2, onde cada elemento do conjunto é um vetor bidimensional cujas coordenadas são as coordenadas (x,y) do pixel preto (por convenção) na imagem. Imagens com mais níveis de cinza podem ser representadas por conjuntos cujos elementos estão no espaço Z^3. Neste caso, os vetores têm três elementos, sendo os dois primeiros as coordenadas do pixel e o terceiro seu nível de cinza.

Em morfologia matemática existem duas operações básicas: dilatação e erosão [FN99]. A dilatação expande uma imagem, enquanto a erosão a encolhe. Há duas outras importantes operações morfológicas: a abertura e o fechamento.

Dilatação

Sejam A e B conjuntos no espaço Z^2 e seja $\varnothing$ o conjunto vazio. A dilatação de A por B, denotada $A \oplus B$, é definida como:

$$A \oplus B = \{x \mid (\hat{B})_x \cap A \neq \varnothing\} \tag{2.2}$$

Portanto, o processo de dilatação consiste em obter a reflexão de B (representada por $\hat{B}$) sobre sua origem e depois deslocar esta reflexão de *x*. A dilatação de A por B é, então, o conjunto de todos os *x* deslocamentos para os quais a interseção de $(\hat{B})_x$ e A inclui pelo menos um elemento diferente de zero. Com base nesta interpretação, a equação anterior pode ser escrita como:

$$A \oplus B = \{x \mid [(\hat{B})_x \cap A] \subseteq A\} \tag{2.3}$$

O conjunto B é normalmente denominado elemento estruturante.

Erosão

Sejam A e B conjuntos no espaço Z^2. A erosão de A por B, denotada $A \Theta B$, é definida como:

$$A \Theta B = \{x \mid (B)_x \subseteq A\} \tag{2.4}$$

o que, em outras palavras, significa dizer que a erosão de A por B resulta no conjunto de pontos *x* tais que B, transladado de *x*, está contido em A.

Abertura e fechamento

A abertura, em geral, suaviza o contorno de uma imagem, quebra istmos estreitos e elimina proeminências delgadas. O fechamento, por sua vez, funde pequenas quebras e alarga golfos estreitos, elimina pequenos orifícios e preenche *gaps* no contorno.

A abertura de um conjunto A por um elemento estruturante B, denotada A o B, é definida como:

$$A \circ B = (A \ominus B) \oplus B \tag{2.5}$$

o que equivale a dizer que a abertura de A por B é simplesmente a erosão de A por B seguida de uma dilatação do resultado por B.

O fechamento do conjunto A pelo elemento estruturante B, denotado A • B, é definido como:

$$A \bullet B = (A \oplus B) \ominus B \tag{2.6}$$

o que nada mais é que a dilatação de A por B seguida da erosão do resultado pelo mesmo elemento estruturante B.

Dougherty e Giardina [DG87] aborda detalhadamente a Morfologia Matemática. Para quem deseja estudar algoritmos morfológicos, Facon [Fac96] apresenta vários algoritmos.

2.4 Visão computacional

Visão computacional é a área da ciência que se dedica a desenvolver teorias e métodos voltados à extração automática de informações úteis contidas em imagens. Tais imagens são capturadas por sensores como câmera de vídeo, scanner, etc.

Uma interessante abordagem para a construção de máquinas inteligentes é expandir os sentidos através dos quais o computador pode comunicar-se com o mundo exterior. A utilização da visão de máquina amplia as aplicações em computadores, como por exemplo, navegação móvel por robô, tarefas complexas de manufatura, análise de imagens de satélites e processamento de imagens médicas.

A visão computacional procura oferecer de modo mais eficiente possível uma vasta quantidade de informações ao sistema computacional para que este possa agir como esperado. O reconhecimento de padrões está engajado no campo da visão computacional com atuações e perspectivas importantíssimas para alcançar e realizar a "máquina inteligente".

2.4.1 Reconhecimento de padrões

O primeiro passo no processo de tratamento da imagem é a aquisição de imagem – isto é, adquirir uma imagem digital. Para isso são necessários um sensor de imagem e a capacidade para digitalizar o sinal produzido pelo sensor. Esta etapa foi abordada com mais detalhes em seções anteriores.

Depois que uma imagem digital é obtida, o próximo passo lida com o pré-processamento desta imagem [GW92]. A função chave do pré-processamento é melhorar a imagem de forma que aumente as chances para o sucesso dos outros processos. O pré-processamento tipicamente lida com técnicas para aumentar o contraste, remover ruído, e isolar regiões cuja textura indica uma probabilidade de informação alfanumérica. Algumas técnicas foram apresentadas em seções anteriores, como, por exemplo, filtragem e detecção de bordas.

As próximas fases lidam com segmentação. A segmentação particiona uma imagem de entrada em suas partes constituintes ou objetos. Em geral, segmentação autônoma é uma das tarefas mais difíceis em processamento digital de imagens. Por um lado, um procedimento de segmentação traz para o processo um modo para solução viável de um problema de reconhecimento. Por outro lado, algoritmos de segmentação pobres, ou irregulares, quase sempre geram falhas eventuais. Em termos de reconhecimento de caracter, o papel chave da segmentação é extrair caracteres e palavras individuais do fundo.

As saídas da fase de segmentação normalmente são dados de pixel puro, constituindo ou o limite de uma região ou todos os pontos na própria região. Em qualquer caso, a conversão dos dados para uma forma satisfatória de processamento computacional é necessária. A primeira decisão que deve ser tomada é se os dados deveriam ser representados como um limite ou como uma região completa. A representação por limite é apropriada quando o enfoque está em características externas da forma, como cantos e inflexões. A representação por região é apropriada quando o enfoque está em propriedades internas, como textura ou forma de esqueleto. Em algumas aplicações, porém, estas representações coexistem. Esta situação ocorre em aplicações de reconhecimento de caracter, que

freqüentemente requerem algoritmos baseados em forma de limite como também esqueletos e outras propriedades internas [GW92].

A escolha de uma representação é só parte da solução para transformar dados puros em uma forma satisfatória para o processamento computacional subseqüente. Um método também deve ser especificado para descrever os dados de forma que as características de interesse são destacadas. Descrição, também denominada *seleção de característica*, lida com extração de características que resultam em alguma informação quantitativa de interesse ou características que são básicas para diferenciar uma classe de objetos de outra.

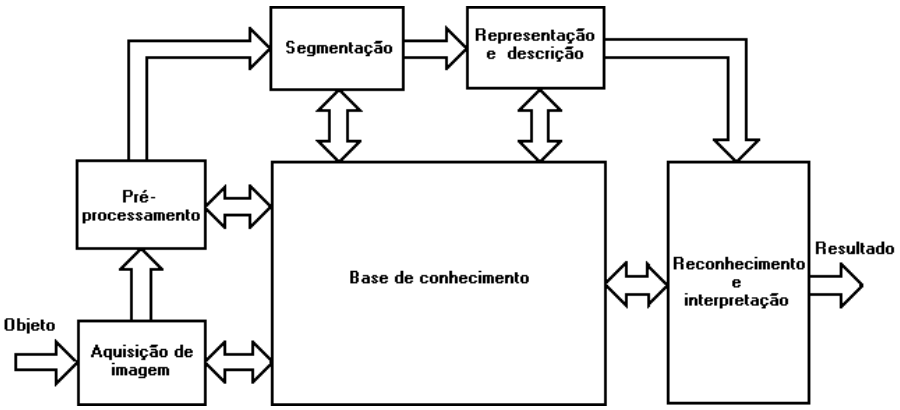


Figura 2.4 – Passos fundamentais em processamento digital de imagens.

A última fase na Figura 2.4 envolve reconhecimento e interpretação. O reconhecimento é o processo que fixa um rótulo a um objeto baseado na informação fornecida por seus descritores. A interpretação envolve fixação de significados a um conjunto de objetos reconhecidos. Logo, para identificar um caracter como um *c* necessita-se associar os descritores para aquele caracter com o rótulo *c*. A interpretação tenta fixar significados a um conjunto de entidades rotuladas. Por exemplo, uma string de cinco números - ou de cinco números seguidos por um hífen e mais três números - pode ser interpretado como sendo um CEP (Código de Endereçamento Postal).

O conhecimento sobre um domínio de problema é codificado em um sistema de processamento de imagens na forma de um banco de dados de conhecimento. Este conhecimento pode ser tão simples como as regiões detalhadas de uma imagem na qual sabe-se onde a informação de interesse está localizada, limitando conseqüentemente a procura a ser conduzida para obter aquela informação. A base de conhecimento também pode ser bastante complexa, tal como uma lista inter-relacionada de todos os principais defeitos possíveis em um problema de inspeção de materiais ou um banco de dados de imagens contendo imagens de satélite de alta resolução de uma região com relação a aplicações de detecção de alterações. Além de guiar a operação de cada módulo de processamento, a base de conhecimento também controla a interação entre módulos. Esta distinção é

feita na Figura 2.4 pelo uso de setas bidirecionais entre os módulos de processamento e a base de conhecimento, ao invés de setas com único sentido que unem os módulos de processamento. Esta representação indica que a comunicação entre módulos de processamento geralmente é baseada no conhecimento anterior [GW92].

2.4.2 Aplicações da visão computacional

A visão computacional, sob a forma de reconhecimento de padrões, está presente hoje nos mais diversos campos de atuação, manipulando estruturas de dados simples ou complexas.

Para os reconhecimentos mais simples, do tipo presença ou ausência de objetos, ou elementos de imagem simples que sofrem pouca variação na sua forma, os sensores podem atuar no papel de olheiros. Como exemplo, pode ser citado o reconhecimento/leitura dos códigos de barras impressos nos produtos consumidos atualmente.

A visão computacional através do reconhecimento de padrões encontrou variadas formas de aplicações industriais, que vão desde a inspeção de peças até a montagem automatizada. Certamente é na aplicação industrial o grande enfoque atual dos sistemas de visão computacionais, pelo poder econômico dos grandes empreendedores e pelo retorno notável que a visão artificial pode trazer a esse segmento.

A visão artificial também está atuando no entretenimento e em áreas esportivas fazendo análise de posições e movimentos de atletas e das outras entidades participantes do esporte em questão, como a bola, o dardo e outros.

No segmento médico há um grande interesse para que a visão computacional auxilie no diagnóstico de moléstias analisando imagens radiográficas e tomográficas. O reconhecimento de padrões atua também na contagem e identificação de cromossomos e como interpretador de curvas e sinais (contínuos ou digitais) vindos de equipamentos médicos, especialmente os aplicados em fisiologia [www01].

A segurança também recebe auxílio da visão computacional, no reconhecimento de assinaturas, impressões digitais e no casamento dos traços do rosto para identificação de pessoas.

A visão computacional, portanto, está envolvida nas mais variadas formas de aplicação e sendo exigida cada vez mais a realizar tarefas tão eficientes quanto a visão humana.

2.4.3 Fundamentos e desafios

O reconhecimento de padrões apresenta algumas dificuldades. É possível classificar objetos bidimensionais, como peças mecânicas em uma correia transportadora, mas é difícil classificar objetos tridimensionais, por causa do grande número de possíveis orientações de cada objeto.

O reconhecimento de imagens é a tarefa visual mais complexa [Ric88]. Embora alguns aspectos do reconhecimento de imagens reduzam-se à análise de medição e ao reconhecimento de padrões, todo problema continua sem solução devido aos seguintes fatores:

- Uma imagem é bidimensional, ao passo que o mundo é tridimensional. Portanto há perda de informações quando a imagem é criada.
- Uma imagem pode conter vários objetos, e alguns objetos podem ocultar parcialmente outros.
- O valor de um único pixel é afetado por fenômenos diferentes, incluindo cor do objeto, fonte de luz, ângulo e distância da câmera, poluição do ar, qualidade do sensor de luz, etc.

Como resultado, as imagens bidimensionais podem ser bastante ambíguas. Dada uma única imagem, pode-se construir qualquer número de mundos tridimensionais que dariam origem à imagem.

Nas primeiras páginas de seu livro, Schalkoff [Sch89] formula uma sugestiva pergunta quando questiona: “O que estamos tentando fazer e por que isto é tão difícil?”.

Uma resposta simplista à primeira parte da pergunta acima é: “Estamos tentando ensinar robôs a enxergar.” Ao relacionar as dificuldades inerentes ao processo de dotar o computador de uma capacidade visual semelhante à dos seres humanos, depara-se com três admiráveis características do processo de percepção visual humano:

- uma base de dados muito rica;
- altíssima velocidade de processamento; e
- a capacidade de trabalhar sob condições muito variadas.

Os avanços na tecnologia de dispositivos de armazenamento de massa e o surgimento de novas CPUs e arquiteturas computacionais cada vez mais rápidas, com alto grau de paralelismo, induz a crer que há condições cada vez melhores de modelar as duas primeiras características relacionadas anteriormente. O grande desafio permanece sendo o de fazer com que os sistemas de visão artificial trabalhem em diferentes condições de luminosidade, contraste, posicionamento relativo dos objetos em uma cena, sem perder a capacidade de interpretar a cena, de forma análoga à

capacidade do ser humano de reconhecer um amigo ou parente com relativa facilidade, independentemente de ele estar usando óculos ou não, ter deixado crescer a barba ou estar no carro ao lado de outro em uma esquina num final de tarde, onde uma pessoa não dispõe de outra imagem senão a vista de perfil e onde as condições de luminosidade são bastante inferiores as que seriam obtidas ao meio-dia.

2.5 Redes neurais artificiais

Uma abordagem diferente para reconhecimento de padrão que atraiu considerável interesse há alguns anos provém do campo da tecnologia de redes neurais artificiais. Inicialmente inspirado em sistemas nervosos biológicos, o desenvolvimento de redes neurais artificiais foi recentemente motivado por sua aplicabilidade em certos tipos de problemas e seu potencial para implementações em processamento paralelo. Conseqüentemente, vem surgindo um grande número de projetos de redes que são capazes de resolver, através de treinamento supervisionado e não-supervisionado, diversos problemas de reconhecimento de padrão. Uma rede neural artificial (RNA) é composta por neurônios artificiais que possuem características básicas como entradas (X_i), pesos (W_i), função soma (Σ), função de transferência (f(y)) e saída (Z).

Atualmente, redes neurais artificiais estão sendo intensivamente aplicadas para resolver problemas relacionados à visão computacional. Porém, tais redes estão sendo simuladas em máquinas uniprocessadoras. Logo, atualmente, a velocidade de processamento para reconhecimento e aprendizagem não é suficientemente rápida para uso prático – especialmente em aplicações em tempo real. Recentemente, arquiteturas paralelas estão sendo desenvolvidas com velocidades suficientemente altas para permitir processamento em tempo real para solução de alguns problemas, mas construir redes neurais artificiais com o tamanho e a complexidade dos cérebros humanos está ainda longe de ocorrer. O próximo capítulo aborda algumas características fundamentais sobre redes neurais artificiais com mais detalhes.

2.6 Considerações finais

Muitos softwares de processamento de imagens básicos podem ser encontrados comercialmente. Contudo, o processamento de imagens é caracterizado por soluções específicas. Daí, técnicas que trabalham bem em uma área podem ser totalmente inadequadas em outra. Tudo o que a

disponibilidade de muitos hardwares e softwares básicos faz é prover um ponto de partida por um custo menor do que era uma década atrás.

A comunicação através de grandes distâncias apresenta um desafio mais sério se a intenção é comunicar dados de imagem em lugar de resultados abstratos. Como deve ser até agora evidente, imagens digitais contêm uma quantia significante de dados. Atualmente uma linha telefônica de voz comum pode transmitir a uma taxa de no máximo de 56 kbits/s. Assim, uma imagem de 512 x 512 pixels de 8 bits exigiria, a esta taxa, aproximadamente um minuto para transmissão. Links sem fio que usam estações intermediárias, como satélites, é muito mais rápido, mas também custam consideravelmente mais. O ponto é que a transmissão de imagens inteiras a longas distâncias está longe do trivial.

Capí tulo 3

Redes Neurais Artificiais

Este capítulo realiza um breve estudo sobre redes neurais artificiais abordando seus fundamentos e aplicações. Apresenta-se um breve histórico das redes neurais, os conceitos e a descrição das redes neurais, os principais métodos de treinamento, e a classificação das redes neurais. No final do capítulo comenta-se a aplicação das redes neurais em reconhecimento de padrões.

3.1 Introdução

Há cerca de duas décadas atrás o termo Rede Neural Artificial (RNA) era raro na literatura científica. Hoje em dia representa uma vigorosa área de pesquisa multidisciplinar. Freqüentemente a RNA identifica-se como uma sub especialidade de Inteligência Artificial, outras vezes como uma classe de modelos matemáticos para problemas de classificação e reconhecimento de padrões. Atualmente, as RNA's constituem genuinamente uma teoria para o estudo de fenômenos complexos, sendo consideradas sistemas distribuídos, não lineares e robustos, dotadas da capacidade de aprender complexos mapeamentos não lineares [Kov96].

Pode-se considerar que a utilização de RNA é uma técnica que pode ser utilizada na solução de problemas de inteligência artificial. Neste caso, em lugar de tentar programar um computador digital de modo a fazê-lo imitar um comportamento inteligente – saber jogar xadrez, compreender e manter um diálogo, traduzir línguas estrangeiras, resolver problemas de matemática tais como se

encontram nos primeiros anos dos cursos de engenharia, etc. – procura-se construir um computador que tenha seus circuitos (emulando uma RNA) modelando os circuitos cerebrais. Desse computador espera-se ver um comportamento inteligente emergindo, aprendendo novas tarefas, errando, fazendo generalizações e descobertas, e freqüentemente ultrapassando seu professor. Estes circuitos neurais artificiais poderão se auto-organizar, quando apresentados a ambientes diversos, criando suas próprias representações internas e apresentar comportamentos imprevisíveis. E, melhor ainda, ter um comportamento que nem sempre se pode prever e compreender, tal como ainda hoje não é possível compreender certos mecanismos do próprio cérebro humano [TXF96].

3.2 Histórico

O primeiro modelamento matemático sobre a neurocomputação data de 1943, em artigos de McCulloch e Pitts [MP43], em que sugeriam a construção de uma máquina baseada ou inspirada no funcionamento de um neurônio biológico. Muitos outros artigos e livros surgiram desde então, porém, por um longo período de tempo, pouco resultado foi obtido. Até que em 1950 Donald Hebb [Heb50] escreveu um livro entitulado "The Organization of Behavior" (A Organização do Comportamento) que perseguia a idéia de que o condicionamento psicológico clássico está presente em qualquer parte dos animais pelo fato de que esta é uma propriedade de neurônios individuais. Suas idéias não eram completamente novas, mas Hebb foi o primeiro a propor uma lei de aprendizagem específica para as sinapses dos neurônios. Este primeiro passo serviu de inspiração para que muitos outros pesquisadores perseguissem a mesma idéia.

Também ocorreu nesta época a construção do primeiro neurocomputador, denominado *Snark*, por Mavin Minsky, em 1951. O Snark operava com sucesso a partir de um ponto de partida técnico, ajustando seus pesos automaticamente, entretanto, ele nunca executou qualquer função de processamento de informação interessante, mas serviu de inspiração para as idéias de estruturas de RNA's que o sucederam.

Em 1956 no "Darthmouth College" nasceram os dois paradigmas da Inteligência Artificial, o simbólico e o conexionista [TXF96]. A Inteligência Artificial Simbólica tenta simular o comportamento inteligente humano desconsiderando os mecanismos responsáveis por tal. Já a Inteligência Artificial Conexionista acredita que se for construído um sistema que simule a estrutura do cérebro, este sistema apresentará inteligência, ou seja, será capaz de aprender, assimilar, errar e aprender com seus erros.

O primeiro neurocomputador a obter sucesso (Mark I Perceptron) surgiu em 1957 e 1958, criado por Frank Rosenblatt, Charles Wightman e outros [Ros58]. Devido à profundidade de seus estudos, suas contribuições técnicas e de sua maneira moderna de pensar, muitos o vêem como o fundador da neurocomputação na forma em que há hoje. Seu interesse inicial para a criação do Perceptron era o reconhecimento de padrões.

Após Rosenblatt, Bernard Widrow [WH60], com a ajuda de alguns estudantes, desenvolveu um novo tipo de elemento de processamento de redes neurais denominado de Adaline, equipado com uma poderosa lei de aprendizado, que diferente do Perceptron ainda permanece em uso. Widrow também fundou a primeira companhia de hardware de neurocomputadores e componentes.

Infelizmente, os anos seguintes foram marcados por um entusiasmo exagerado de muitos pesquisadores, que passaram a publicar mais e mais artigos e livros que faziam uma previsão pouco confiável para a época, sobre máquinas tão poderosas quanto o cérebro humano que surgiriam em um curto espaço de tempo. Isto tirou quase toda a credibilidade dos estudos desta área e causou grandes aborrecimentos aos técnicos de outras áreas.

Progressivamente as duas correntes da IA, a simbólica e a conexionista, separaram-se, e as pesquisas em redes neurais (corrente conexionista) andaram lentamente enquanto a corrente da manipulação simbólica se acelerou. É interessante notar que um motivo para esta separação foi o livro de Minsky & Papert [MP69]. Este livro, entretanto, constitui um dos primeiros estudos sobre a complexidade do problema e a correspondente capacidade das redes neurais para resolvê-lo: “um perceptron de uma única camada é incapaz de resolver os problemas linearmente não separáveis”. Administradores do governo dos Estados Unidos, responsáveis por distribuir fundos de pesquisa, concluíram que o assunto não era interessante e cortaram os investimentos em redes neurais.

Um período de pesquisa silenciosa seguiu-se entre 1967 e 1982, quando poucas pesquisas foram publicadas devido aos fatos ocorridos anteriormente. Entretanto, aqueles que pesquisavam nesta época, e todos os que se seguiram no decorrer de treze anos conseguiram novamente estabelecer um campo concreto para o renascimento da área.

Nos anos 80, muitos dos pesquisadores foram bastante corajosos e passaram a publicar diversas propostas para a exploração de desenvolvimento de redes neurais bem como suas aplicações. Porém talvez o fato mais importante deste período tenha ocorrido quando Ira Skurnick, um administrador de programas da DARPA (Defense Advanced Research Projects Agency) decidiu ouvir os argumentos da neurocomputação e seus projetistas, e divergindo dos caminhos tradicionais dos conhecimentos convencionais, fundou em 1983 pesquisas em neurocomputação. Este ato não só

abriu as portas para a neurocomputação, como também deu à DARPA o status de uma das líderes mundiais em se tratando de "moda" tecnológica.

Outra celebridade que emergiu neste período foi John Hopfield. Renomado físico de reputação mundial, se interessou pela neurocomputação e escreveu artigos que percorreram o mundo todo persuadindo centenas de cientistas, matemáticos e tecnólogos altamente qualificados a se unirem nesta nova área emergente [Hop82].

Apesar de um terço dos pesquisadores da área terem aderido à mesma pela influência de Hopfield, foi em 1986 que este campo de pesquisa "explodiu" com a publicação do livro "Parallel Distributed Processing" (Processamento Distribuído Paralelo) editado por David Rumelhart e James McClelland [RM86].

Em 1987 ocorreu em São Francisco a primeira conferência de redes neurais em tempos modernos, a IEEE (International Conference on Neural Networks), e também foi formada a International Neural Networks Society (INNS). A partir destes acontecimentos decorreu a fundação do INNS journal em 1989, seguido do Neural Computation e do IEEE Transactions on Neural Networks em 1990. Desde 1987, muitas universidades anunciaram a formação de institutos de pesquisa e programas de educação em neurocomputação.

3.3 Conceitos de redes neurais artificiais

Apesar de uma rede neural poder ser simulada e executada em um computador seqüencial, a rede está muito mais para o funcionamento cerebral do que para um computador convencional. Portanto os modelos neurais procuram aproximar o processamento (paralelo) dos computadores ao cérebro. As redes neurais possuem um grau de interconexão similar à estrutura do cérebro e em um computador convencional moderno a informação é transferida em tempos específicos dentro de um relacionamento com um sinal para sincronização.

A Tabela 3.1 traça um comparativo entre o cérebro humano e o computador:

Tabela 3.1 – Quadro comparativo entre o cérebro e o computador

Parâmetro	Cérebro	Computador
Material	Orgânico	Metal e plástico
Velocidade	Milissegundos	Nanossegundos
Tipo de Processamento	Paralelo	Seqüencial
Armazenamento	Adaptativo	Estático
Controle de Processos	Distribuído	Centralizado
Número de elementos processados	10^{11} a 10^{14}	10^5 a 10^6
Ligações entre elementos processados	10.000	< 10

O mesmo paralelo pode ser traçado comparando o computador com as redes neurais. Para tanto, a comparação não se dará com um computador específico encontrado no mercado, mas sim com o paradigma predominante nos computadores atuais.

Tabela 3.2 – Quadro comparativo entre computadores e neurocomputadores

Computadores	Neurocomputadores
Executa programas	Aprende
Executa operações lógicas	Executam operações não lógicas, transformações, comparações
Depende do modelo ou do programador	Descobre as relações ou regras dos dados e exemplos
Testa uma hipótese por vez	Testa todas as possibilidades em paralelo

Uma rede neural artificial, conforme definição de Teuvo Kohonen [Koh87], um importante pesquisador finlandês de redes neurais, é:

> "uma rede massivamente paralela interconectada de elementos e suas organizações hierárquicas que estão intencionadas para interar com objetos do mundo real do mesmo modo que um sistema nervoso biológico faz".

Um modelo neural é programado para aprender e essa palavra soa estranha aos ouvidos dos profissionais de informática, pois durante todo seu curso de graduação são passadas informações de que um computador faz apenas aquilo que for mandado fazer, ou seja, um computador apenas executa instruções para os quais foi prévia e devidamente programado.

Os conceitos de redes neurais estão datados desde a década de 40, o que significa que, na época, não havia computadores, não como os que se conhece hoje. Deduz-se, então, que as redes neurais independem de computadores para funcionarem, da mesma forma que um projetista de casas

(um engenheiro, por exemplo) independe de um computador para realizar os seus projetos. Contudo, é certo que um bom sistema de CAD (Computer Aided Design) ajudará reduzindo o tempo de elaboração do projeto de forma significativa, além de aumentar a qualidade do trabalho final. O tempo e a qualidade são aspectos de relevância significativa na era moderna, não só em computação ou engenharia, mas praticamente em todas as ciências conhecidas.

Para o caso de redes neurais, embora a situação não seja exatamente a mesma, é muito parecida. No início (na década de 40), os modelos de redes neurais eram calculados com papel e lápis na mão, talvez até uma caneta tinteiro. Hoje, esses mesmos modelos são jogados para dentro de um computador e esperado o resultado do processamento do modelo. Esse processamento antes era feito totalmente a mão, o que poderia significar incontáveis horas de cálculo, para não dizer impraticáveis.

3.4 Motivação

A partir do momento em que as máquinas começaram a evoluir, um grande desejo do homem tem sido a criação de uma máquina que possa operar independentemente do controle humano. Uma máquina cuja independência seja desenvolvida de acordo com seu próprio aprendizado e que tenha a capacidade de interagir com ambientes incertos (desconhecidos por ela), uma máquina que possa ser chamada de autônoma, inteligente ou cognitiva.

O sucesso de uma máquina autônoma dependeria única e exclusivamente de sua capacidade de lidar com uma variedade de eventos inesperados no ambiente em que opera. Estas máquinas teriam maior capacidade de aprender tarefas de alto nível cognitivo que não são facilmente manipuladas por máquinas atuais, e continuariam a se adaptar e realizar tais tarefas gradativamente com maior eficiência, mesmo que em condições de ambiente imprevisíveis. Então, seriam muito úteis onde as iterações humanas são perigosas, tediosas ou impossíveis; como em reatores nucleares, combate ao fogo, operações militares, exploração do espaço a distâncias em que um a nave espacial estaria fora do alcance do controle na terra porém enviando informações.

Organismos humanos são uma fonte de motivação para o desenvolvimento destas máquinas, e proporcionam diversas dicas para o desenvolvimento de algoritmos de aprendizado e adaptação. Assim, espera-se que algumas das características de organismos biológicos de aprendizado e adaptação estejam presentes nas mesmas.

Enquanto computadores funcionam de modo seqüencial, proporcionando maior eficiência na resolução de tarefas nas quais devem ser seguidas etapas, o cérebro humano funciona de modo paralelo, e sendo extremamente conectado é mais eficiente na resolução de tarefas que exigem várias variáveis.

O motivo pelo qual máquinas inspiradas na biologia são diferentes das máquinas atuais se encontra no fato de que as máquinas atuais baseiam seu processamento explicitamente em modelos matemáticos. Mecanismos de controle baseado em mecanismos neurais entretanto, não são baseados em modelos, utilizam cálculos matemáticos para efetuar suas operações porém podem coordenar diversos graus de liberdade durante a execução de tarefas manipulativas e em ambientes desestruturados. Eles são capazes de lidar com tarefas complicadas sem que tenham que desenvolver um modelo matemático e nem um modelo do ambiente em que operam.

Baseado nas características de seres biológicos, acredita-se que surgirá, em um futuro próximo, uma geração completa de novos sistemas computacionais, muito mais eficientes e inteligentes que os sistemas atuais.

3.5 Descrição de uma rede neural artificial

As redes neurais artificiais consistem em um método de solucionar problemas de inteligência artificial, constituindo um sistema que tenha circuitos que simulem o cérebro humano, inclusive seu comportamento, ou seja, aprendendo, errando e fazendo descobertas. São mais que isso, são técnicas computacionais que apresentam um modelo inspirado na estrutura neural de organismos inteligentes e que adquirem conhecimento através da experiência. Uma grande rede neural artificial pode ter centenas ou milhares de unidades de processamento, enquanto que o cérebro de um mamífero pode ter bilhões de neurônios.

Apesar da complexidade das redes neurais não permitir uma única definição, as linhas seguintes seguem como uma tentativa das inúmeras definições ou interpretações do que seja realmente uma rede neural.

Um grafo direcionado é um objeto geométrico que consiste de um conjunto de pontos, denominados nós, ao longo de um conjunto de segmentos de linhas direcionadas entre eles. Uma

rede neural é uma estrutura de processamento de informação, distribuída paralelamente na forma de um grafo direcionado, com algumas particulares restrições e definições.

Os nós deste grafo são chamados *elementos de processamento*. Suas arestas são conexões, que funcionam como caminhos de condução instantânea de sinais em uma única direção, de forma que seus elementos de processamento podem receber qualquer número de conexões de entrada. Estas estruturas podem possuir memória local, e também possuir qualquer número de conexões de saída desde que os sinais nestas conexões sejam os mesmos. Portanto, estes elementos têm na verdade uma única conexão de saída, que pode dividir-se em cópias para formar múltiplas conexões, sendo que todos carregam o mesmo sinal.

Então, a única entrada permitida para a função de ativação (que cada elemento de processamento possui) são os valores armazenados na memória local do elemento de processamento e os valores atuais dos sinais de entrada nas conexões recebidas pelo elemento de processamento. Os únicos valores de saída permitidos a partir da função de transferência são valores armazenados na memória local do elemento de processamento, e o sinal de saída do mesmo.

A função de transferência pode operar continuamente ou episodicamente. Sendo que no segundo caso, deve existir uma entrada chamada *activate* que causa a ativação da função de transferência com o sinal de entrada corrente e com valores da memória local, e produzir um sinal de saída atualizado (ocasionalmente alterando valores da memória). E no primeiro caso, os elementos estão sempre ativados, e a entrada *activate* chega através de uma conexão de um elemento de processamento agendado que também é parte da rede.

Sinais de entrada para uma rede neural a partir de fora da rede chegam através de conexões que se originam do mundo externo; saídas da rede para o mundo externo são conexões que deixam a rede.

De forma geral, a operação de uma célula da rede se resume em:

- Sinais são apresentados à entrada;
- Cada sinal é multiplicado por um peso que indica sua influência na saída da unidade;
- É feita a soma ponderada dos sinais que produz um nível de atividade;
- Se este nível excede um limite (threshold) a unidade produz uma saída (este é o comportamento dos neurônios iniciais. Atualmente a saída é gerada a partir da região não-linear da função de transferência);

3.5.1 O neurônio artificial

Assim como o sistema nervoso é composto por bilhões de células nervosas, a rede neural artificial também seria formada por unidades que nada mais são que pequenos módulos que simulam o funcionamento de um neurônio. Estes módulos devem funcionar de acordo com os elementos em que foram inspirados, recebendo e retransmitindo informações.

O fisiologista Warrem MacCulloch [MP43] interpretou o funcionamento do neurônio biológico como sendo um circuito de entradas binárias combinadas por uma soma ponderada (com pesos) produzindo uma entrada efetiva:

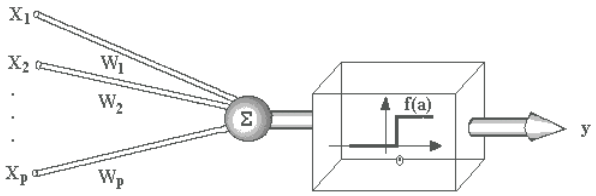


Figura 3.1 – Modelo de McCulloch e Pitts

No modelo geral de neurônio, representado na Figura 3.1, as entradas W_iX_i são combinadas usando uma função *f(a)*, para produzir um estado de ativação do neurônio (correspondente à freqüência de descarga do neurônio biológico). As entradas chegam através dos dendritos e tem um peso atribuído pela sinapse.

Pesos

Quanto aos pesos, atributo importantíssimo do neurônio, pode-se compará-los com os dendritos realizando as sinapses em outros neurônios. Graças a essa comparação, os pesos são denominados *pesos sinápticos*. Os pesos, representados por W (weight), são valores que representam o grau de importância que determinada entrada possui em relação àquele determinado neurônio. Ou seja, esse valor (o peso), muda em função da importância do sinal de entrada, e dessa forma, o peso muda o seu valor representativo para a rede. Significa que, quando uma entrada é bastante estimulada, acaba estimulando, também, o peso correspondente à sua conexão. Um peso quando é bastante estimulado, automaticamente terá, cada vez mais, mais influência no resultado do sinal de saída.

Os pesos podem ser vistos, matematicamente, como um vetor de valores $(w_1,w_2,...,w_n)$. Havendo mais de um neurônio na rede, pode-se então ter uma coleção de vetores, ou seja, uma matriz de pesos, onde cada vetor, corresponde a um neurônio. Quando as entradas $(x_1,x_2,...,x_n)$ são apresentadas para o neurônio, elas são multiplicadas pelos pesos, e a soma desses resultados é, então,

o sinal de excitação do neurônio. As entradas multiplicadas pelos pesos, recebem, depois desta operação, o nome de entradas ponderadas (unidade sigma - Σ).

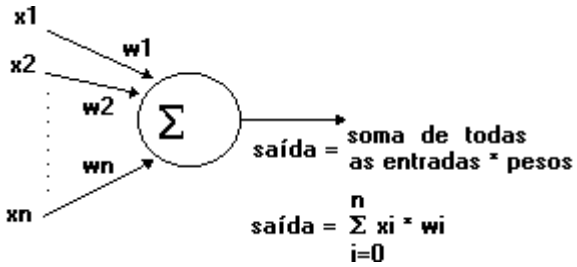


Figura 3.2 – Um neurônio com entradas e pesos definidos

A função do neurônio é, depois de acumulado o valor somado dos produtos ocorridos entre as entradas e os pesos, comparar esse valor com um limiar (um valor estipulado), e, atingindo-o, o valor é então passado adiante através da saída. Esse processo chama-se *função de ativação*. Caso contrário, se o valor não atinge o limiar, o sinal não é transferido adiante. Em ambos os casos, com sinal ou sem sinal, a resposta é significativa, pois afetará diretamente, ou a resposta final da rede, ou os neurônios da próxima camada. A lógica neural expõe, que a intensidade dos sinais de entrada, dispara, ou não, o sinal do neurônio, fazendo com que este estimule o neurônio seguinte.

Função de ativação e função de transferência

Têm-se agora duas funções diferentes, a função de ativação e a função de transferência. A função de ativação antecede a função de transferência, e tem por atribuição, repassar o sinal para a saída do neurônio. A função de ativação é uma função de ordem interna, cuja atribuição é fazer acontecer um nível de ativação dentro do próprio neurônio, ou seja, uma decisão tomada pelo neurônio sobre o que fazer com o valor resultante do somatório das entradas ponderadas. Essa decisão terá um efeito restrito ao próprio neurônio.

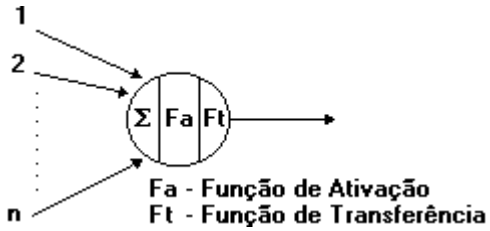


Figura 3.3 – Diferentes funções em um neurônio

Em modelos simples de redes neurais, a função de ativação pode ser a própria função de soma das entradas ponderadas do neurônio. Em modelos mais complexos, a função de ativação possui um processamento atribuído. Esse processamento pode usar, por exemplo, o valor prévio de saída como uma entrada para o próprio neurônio (uma auto-excitação). Após o valor ter sido processado pela

função de ativação, é então passado para a função de transferência que produzirá o valor de saída do neurônio.

A função de transferência pode ter muitas formas e métodos, podendo ser simples ou complexa. A função de transferência também é conhecida como limiar lógico (threshold). Essa função é quem define e quem envia para a saída do neurônio o valor passado pela função de ativação. Em alguns modelos de redes, o nível de saída produzido pela função de transferência é igual ao nível de ativação. Muitas vezes, essa função possui características sigmas ou ríspidas, assim, o neurônio pode não produzir efeito no neurônio seguinte se o valor de ativação estiver abaixo de um valor mínimo para sua ativação. As funções de transferências mais conhecidas são:

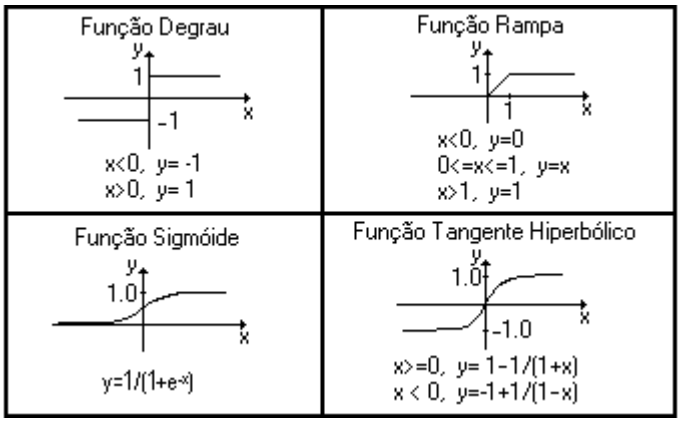


Figura 3.4 – Principais funções de transferência

As funções rampa e degrau possuem decisões ríspidas, principalmente para valores extremos. Essas funções refletem a saída dentro de uma faixa (digamos entre 0 e 1). Isto quer dizer que, ultrapassando um certo limite, a função dispara o valor 0, ou 1, ou -1, dependendo da função e da forma que será utilizada.

As funções sigmóide e tangente hiperbólico são utilizadas normalmente para decisões onde as saídas limites são disparadas quando existe uma saturação muito alta do valor de ativação. Em outras palavras, significa, que se fosse utilizado a função sigmóide $y = \frac{1}{1+e^{-x}}$, essa função produziria um 0 de saída somente quando o valor passado da função de ativação tivesse um grande valor negativo, e produziria o valor de saída 1 quando a ativação tivesse um grande valor positivo. Essa função faz a transição entre os extremos de forma suave.

Fator de aprendizado

O fator de aprendizado controla a velocidade do aprendizado, aumentando ou diminuindo o ajuste de pesos que é efetuado a cada iteração durante o treinamento. Intuitivamente, seu valor deve ser maior que 0 e menor que 1:

- um valor negativo faria com que a correção dos pesos se fizesse no sentido contrário ao do erro observado (a rede neural "desaprenderia" mais ainda, ou seja, seus resultados iriam divergir dos valores esperados na saída);
- um valor maior que 1 faria com que a correção fosse maior do que o erro observado, fazendo com que a rede neural ultrapassasse o ponto de aprendizado ótimo, tornando o processo de treinamento instável.

3.5.2 O Perceptron

O Perceptron foi criado em 1957 por Frank Rosenblatt. A partir desta data foi crescendo o interesse pelas redes neurais. O Perceptron, em sua origem, era uma simulação computacional para a retina, que demonstrou como o sistema nervoso visual reconhece padrões. Houve muito otimismo entre os pesquisadores, pois, aparentemente, parecia que os Perceptrons podiam resolver vários tipos de problemas. Iniciou-se, então, intensa pesquisa sobre redes neurais.

Marvin Minsky e Seymor Papert [MP69], porém, provavam, no livro intitulado Perceptrons, que redes neurais de uma única camada, do tipo Perceptron, não seriam capazes de resolver problemas interessantes. O Perceptron, por exemplo, não pode resolver o problema XOR (ou exclusivo). Assim, o Perceptron foi criticado e acabou sendo apelidado de rede *brain-damaged*, pois não poderia resolver nenhum problema realmente interessante. A pesquisa foi então desestimulada.

Entretanto, não se extinguiu de todo. Pesquisando em silêncio, para talvez evitar crítica de colegas, alguns, um pouco mais destemidos, continuavam trabalhando. E, em 1982, voltou a renascer o interesse pelas redes neurais, com a apresentação à National Academy of Sciences, por John Hopfield, de um trabalho sobre redes neurais (a rede que ele apresentou leva hoje o seu nome).

As redes Perceptron têm apenas uma camada de processamento. Vem daí a limitação de representação por redes Perceptrons de apenas problemas linearmente separáveis. Não existe um procedimento para ajustamento de pesos em redes Perceptron com mais de uma camada (pelo menos não havia quando o Perceptron foi criado). Estes problemas foram superados em modelos de redes que surgiram a partir dos anos 80 com o desenvolvimento de técnicas de aprendizado como, por exemplo, o *backpropagation*. Mas, estas redes receberam outros nomes. Deixaram de ser Perceptrons.

Generalização

Uma das principais características de redes neurais é sua habilidade em generalizar, isto é, classificar corretamente padrões que não foram previamente apresentados. Perceptrons multicamada generalizam detectando características da entrada padrão que são significativas. Portanto um padrão desconhecido é classificado juntamente com outros que compartilham as mesmas características distintas. Isto significa que aprender por exemplos é uma proposição possível, desde que apenas um conjunto representativo de padrões tenha que ser ensinado à rede, e, conseqüentemente, as propriedades de generalização permitirão a entradas similares serem classificadas corretamente. Isto também significa que entradas ruidosas serão classificadas, em virtude de sua similaridade com as entradas puras. É esta habilidade de generalização que permite a perceptrons multicamada desempenhar com maior sucesso problemas do mundo real do que outros tipos de reconhecimento de padrões ou métodos de sistemas especialistas.

Em geral, redes neurais são boas em interpolação, mas não tão boas em extrapolação. Elas são capazes de reconhecer padrões em suas entradas, mesmo sendo estados intermediários que não foram vistos ainda. Entretanto, entradas que são extensões da faixa de padrões são bem menos classificadas, já que existe pouco padrão com que as comparar. Por outro lado, dado um padrão não visto que é uma mistura intermediária de dois padrões previamente ensinados, a rede o classificará como um exemplo do padrão predominante. Se o padrão não corresponde a algum padrão similar para o qual a rede tenha visto antes, a classificação será muito mais pobre [BJ90].

3.5.3 A rede neural artificial

A rede neural artificial é um sistema de neurônios ligado por conexões sinápticas e dividido em neurônios de entrada, que recebem estímulos do meio externo, neurônios internos ou ocultos e neurônios de saída, que se comunicam com o exterior. A forma de arranjar perceptrons em camadas é denominada *Perceptron Multicamada* (*Multilayer Perceptron*). O perceptron multicamada foi concebido para resolver problemas mais complexos, os quais não poderiam ser resolvidos pelo modelo de neurônio básico. Um único perceptron ou uma combinação das saídas de alguns perceptrons poderia realizar uma operação XOR, porém, seria incapaz de aprendê-la. Para isto são necessárias mais conexões, os quais só existem em uma rede de perceptrons dispostos em camadas. Os neurônios internos são de suma importância na rede neural pois se provou que sem estes se torna impossível a resolução de problemas linearmente não separáveis. Em outras palavras pode-se dizer que uma rede é composta por várias unidades de processamento, cujo funcionamento é bastante

simples. Essas unidades, geralmente são conectadas por canais de comunicação que estão associados a determinado peso. As unidades fazem operações apenas sobre seus dados locais, que são entradas recebidas pelas suas conexões. O comportamento inteligente de uma rede neural artificial vem das interações entre as unidades de processamento da rede.

A maioria dos modelos de redes neurais possui alguma regra de treinamento, onde os pesos de suas conexões são ajustados de acordo com os padrões apresentados. Em outras palavras, elas aprendem através de exemplos. Arquiteturas neurais são tipicamente organizadas em camadas, com unidades que podem estar conectadas às unidades da camada posterior.

A rede neural passa por um processo de treinamento a partir dos casos reais conhecidos, adquirindo, a partir daí, a sistemática necessária para executar adequadamente o processo desejado dos dados fornecidos. Sendo assim, a rede neural é capaz de extrair regras básicas a partir de dados reais, diferindo da computação programada, onde é necessário um conjunto de regras rígidas pré-fixadas e algoritmos.

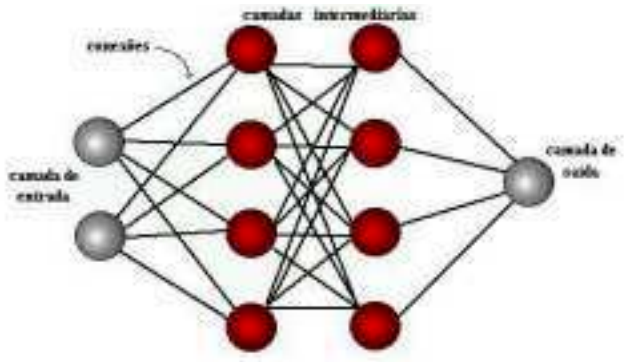


Figura 3.5 – Organização em camadas.

Usualmente as camadas são classificadas em três grupos:

- Camada de Entrada: onde os padrões são apresentados à rede;
- Camadas Intermediárias ou Ocultas: onde é feita a maior parte do processamento, através das conexões ponderadas; podem ser consideradas como extratoras de características;
- Camada de Saída: onde o resultado final é concluído e apresentado.

Redes neurais são também classificadas de acordo com a arquitetura em que foram implementadas, com a topologia, com características de seus nós, com regras de treinamento, e com tipos de modelos.

3.5.4 Propriedades de uma rede neural

Uma rede neural possui uma estrutura maciçamente paralela e distribuída e é capaz de aprender um dado tipo de conhecimento, armazená-lo e usá-lo para executar uma determinada tarefa. De acordo com Haykin [Hay94], as redes neurais apresentam as seguintes propriedades:

1. **Não linearidade.** Um neurônio é um dispositivo não-linear. Conseqüentemente, a rede neural, interconexão de neurônios, é uma rede não linear. Além disso, a não linearidade ocorre de um modo bastante especial, uma vez que é distribuída ao longo da rede.
2. **Mapeamento entrada-saída.** Um paradigma popular de aprendizagem denominado "*aprendizagem supervisionada*" envolve a modificação dos pesos sinápticos da rede neural pela aplicação de um conjunto de treinamento. Cada amostra do conjunto de treinamento consiste de uma única entrada e saída desejada. Os pesos sinápticos da rede são modificados de forma a minimizar a diferença entre a saída produzida pela rede e a saída desejada. O treinamento da rede é realizado com todas as amostras do conjunto de treinamento até que a diferença entre a saída produzida e a desejada atinja um mínimo aceitável. Desta forma, a rede aprende através dos exemplos, pela construção de um mapeamento entrada-saída para o problema em questão.
3. **Adaptabilidade**. As redes neurais apresentam uma capacidade intrínseca de adaptar seus pesos sinápticos de acordo com a situação em questão. Em particular, uma rede neural treinada para operar em um ambiente específico pode ser re-treinada para lidar com pequenas modificações nas condições ambientais de operação. Além disso, ao operar em um ambiente não-estacionário, uma rede neural pode ser projetada para modificar seus pesos sinápticos em tempo real.
4. **Tolerância a falhas**. Implementada em hardware, uma rede neural é potencialmente tolerante a falhas no sentido de que sua performance é apenas gradativamente degradada sob condições de operações adversas, tendo em vista a natureza distribuída da informação ao longo da rede.
5. **Possibilidade de implementação em VLSI.** A natureza maciçamente paralela de uma rede neural a torna potencialmente rápida para a realização de certas tarefas. Esta mesma característica a torna idealmente adequada para implementação por hardware paralelo em VLSI. Este tipo de implementação provê um meio de capturar o comportamento complexo de uma rede neural em uma feição extremamente hierárquica, possibilitando o seu uso para operação em tempo real de aplicações envolvendo reconhecimento de padrões, processamento de sinais e controle.

6. **Uniformidade.** Uma mesma notação é utilizada em todos os domínios que envolvem aplicações de redes neurais. Além disso, os neurônios, de uma forma ou de outra, representam o ingrediente comum a todas as redes, e neste contexto é possível aplicar teorias e algoritmos de aprendizagem em aplicações diversas.

3.6 Métodos de controle do aprendizado

A propriedade mais importante das redes neurais é a habilidade de aprender por meio de seu ambiente e com isso melhorar seu desempenho. Isso é feito através de um processo iterativo de ajustes aplicado a seus pesos, o treinamento. O aprendizado ocorre quando a rede neural atinge uma solução generalizada para uma classe de problemas.

Denomina-se algoritmo de aprendizado a um conjunto de regras bem definidas para a solução de um problema de aprendizado. Existem muitos tipos de algoritmos de aprendizado específicos para determinados modelos de redes neurais, estes algoritmos diferem entre si principalmente pelo modo como os pesos são modificados.

A rede neural se baseia nos dados para extrair um modelo geral. Portanto, a fase de aprendizado deve ser rigorosa e verdadeira, a fim de se evitar modelos espúrios. Todo o conhecimento de uma rede neural está armazenado nas sinapses, ou seja, nos pesos atribuídos às conexões entre os neurônios. De 50 a 90% do total de dados deve ser separado para o treinamento da rede neural, dados estes escolhidos aleatoriamente, a fim de que a rede "aprenda" as regras e não "decore" exemplos. O restante dos dados só é apresentado à rede neural na fase de testes a fim de que ela possa "deduzir" corretamente o inter-relacionamento entre os dados.

Aprender é o ato que produz um comportamento diferente a um estímulo externo recebido no passado e é, de uma certa forma, sinônimo de aquisição de conhecimento. Em IA é comum se falar de aprendizado pela máquina e aprender pode ser considerado como atributo fundamental de um comportamento inteligente.

As redes neurais possuem a capacidade de aprenderem por exemplos, e fazerem interpolações do que aprenderem. No aprendizado conexionista não se procura obter regras como na abordagem simbólica da IA, mas determinar a intensidade de conexões entre neurônios. Como o conhecimento é armazenado nas conexões, o uso de redes neurais está intimamente ligado ao que se chama de conexionismo.

A monitorização é o processo de controle do aprendizado da rede. Este processo de aprendizado acontece, basicamente, de duas formas: o aprendizado supervisionado (com professor) e o aprendizado não-supervisionado (sem professor).

3.6.1 Aprendizado supervisionado

Em aprendizado supervisionado, a rede neural é treinada com o auxílio de um professor, ou um treinador. Para tanto, a rede deverá possuir pares de entrada e saída, ou seja, um conjunto de entradas e um conjunto com as saídas desejadas para cada entrada. Toda vez que for apresentada à rede uma entrada, deverá ser verificado se a saída obtida (gerada a partir dos cálculos efetuados com os pesos que a rede possui) confere com a saída desejada para aquela entrada. Sendo diferente, a rede deverá ajustar os pesos de forma que armazene o conhecimento desejado. Essa iteratividade do treino deverá ser repetida com todo o conjunto de treinamento (entradas e saídas), até que a taxa de acerto esteja dentro de uma faixa considerada satisfatória. Essa forma de aprendizado é bem conhecida e tem demonstrado excelentes resultados em aplicações reais.

3.6.2 Aprendizado não-supervisionado

Este tipo de aprendizado também é conhecido como aprendizado auto-supervisionado e não requer saída desejada; e por isso é conhecido pelo fato de não precisar usar professores para o seu treinamento. Para o treinamento da rede, são usados apenas os valores de entrada. A rede trabalha essas entradas e se organiza de modo que acabe classificando-as, usando, para isso, os seus próprios critérios. Esse tipo de rede utiliza os neurônios como classificadores, e os dados de entrada, como os elementos para classificação. O processo de classificação fica a encargo da rede neural e o seu algoritmo de aprendizado. A auto-organização demonstrada em redes neurais não-supervisionadas, envolve o processo de competição e o processo de cooperação entre os neurônios da rede. Muitos pesquisadores têm utilizado esse tipo de rede como detector de características, dada a sua capacidade de aprender a discriminar estímulos ocorrendo em partes espacialmente diferentes. Em outras palavras, pode-se dizer que a rede pode aprender a distinguir a esquerda e a direita sem a presença de um professor para lhe ensinar.

3.7 Classificação de redes neurais artificiais

Um dos objetivos da pesquisa sobre redes neurais na computação é desenvolver morfologias neurais matemáticas, não necessariamente baseada na biologia, que podem realizar funções diversas. Na maior parte dos casos, modelos neurais são compostos de muitos elementos não lineares que operam em paralelo e que são classificados de acordo com padrões ligados à biologia.

Quando um processo é criado visando utilizar aspectos de redes neurais começam com o desenvolvimento de um neurônio artificial ou computacional baseado no entendimento de estruturas biológicas neurais, seguidas do aprendizado de mecanismos voltados para um determinado conjunto de aplicações. Ou em outras palavras, seguindo as três etapas:

- O desenvolvimento de modelos neurais, motivado por neurônios biológicos;
- Modelos de estruturas e conexões sinápticas;
- O aprendizado das regras (um método de ajuste de pesos ou forças de conexões internodais).

Por causa de diferenças entre algumas ou às vezes todas as entidades envolvidas, diferentes estruturas de redes neurais tem sido desenvolvidas por pesquisadores. Do ponto de vista estrutural, a arquitetura de redes neurais pode ser classificada como estática, dinâmica ou fuzzy, e de única camada ou múltiplas camadas. Além disso, diferenças computacionais surgem também quando se trata da maneira com que são feitas as conexões existentes entres os neurônios. Estas conexões podem ser estritamente no sentido de ida, no sentido de ida e volta, lateralmente conectadas, topologicamente ordenadas ou híbridas.

A aplicação de redes neurais pode ser classificada em classes distintas: reconhecimento de padrões e classificação; processamento de imagens e visão; identificação de sistema e controle e processamento de sinais. É importante verificar que uma determinada aplicação de um sistema baseado em rede neural não precisa necessariamente ser classificada em apenas uma das citadas acima.

3.8 Aplicações de redes neurais

A tecnologia de redes neurais é uma ferramenta nova em muitas áreas de aplicação. O uso dessa tecnologia já se encontra disseminado em inúmeras áreas, onde objetivos de desempenho ótimo, monitoramento de atividades e integração de sistemas são almejados. Como qualquer tecnologia

nova, sua eficácia em resolver problemas depende diretamente da adequação de suas potencialidades às características da tarefa a ser realizada. Conhecer a técnica, ou no caso as técnicas de redes neurais, é tão importante quanto conhecer o problema a ser abordado. A boa notícia, e que tem feito de redes neurais uma tecnologia muito popular nos últimos 5 a 10 anos, é que suas potencialidades são muitas e inúmeros casos de sucessos existem, onde redes neurais superam os métodos convencionais na solução.

Aplicações de redes neurais são inúmeras. Muitos tomam conhecimento lendo a respeito das técnicas no prognóstico de mercados financeiros. Grupos de investimentos conhecidos utilizam redes neurais para analisar pelo menos uma parte do mercado financeiro e fazerem suas seleções.

O reconhecimento ótico de caracteres (OCR) é outro tipo de aplicação que já existe e está crescendo, e em breve as pessoas estarão em constante contato com esse tipo de aplicação. Outras aplicações bem sucedidas das técnicas de redes neurais artificiais são: análise de pesquisa de mercado, controle de processos industriais, aplicações climáticas, e identificação de fraude de cartão de crédito. Um banco americano chamado Mellon Bank instalou um sistema de detecção de fraudes de cartão de crédito implementado com técnicas de redes neurais e os prejuízos evitados pelo novo sistema conseguiram cobrir os gastos de instalação em seis meses [www02]. Vários outros bancos começam a utilizar sistemas baseados em redes neurais para controlar fraudes de cartão de crédito. Estes sistemas têm a capacidade de reconhecer uso fraudulento com base nos padrões criados no passado com uma exatidão melhor que em outros sistemas.

Outro exemplo da utilização de redes neurais está no diagnóstico médico. Em seu aprendizado, são submetidos uma série de diagnósticos de pacientes, de várias características, com vários sintomas e os resultados de seus testes. Também serão fornecidos os diagnósticos médicos para cada doença. Então quando forem apresentados os dados de um novo paciente, com seus sintomas, a rede fornecerá um diagnóstico para os novos casos. Isto essencialmente criará um sistema com o conhecimento de vários médicos, e fornecerá um diagnóstico inicial em tempo real a um médico. É importante mencionar que com isso o que se pretende é implementar uma ferramenta de auxílio ao médico, e não um programa que o substitua.

Outras aplicações:

- análise e processamento de sinais;
- controle de processos;
- robótica;
- classificação de dados;
- reconhecimento de padrões em linhas de montagem ;

- análise de imagens;
- análise de voz;
- avaliação de crédito;
- análise de aroma e odor – um projeto em desenvolvimento, buscando analisar o odor via nariz eletrônico;
- análise e diagnóstico de descargas parciais pelo reconhecimento do padrão acústico– trata-se de uma tese de mestrado cujo objetivo é criar um sistema com capacidades de classificar o padrão acústico de uma descarga parcial;

Para estas e muitas outras aplicações existem diversas páginas na internet, inclusive com alguns simuladores, e listagens de programas fontes.

3.9 Considerações finais

Apesar da neurocomputação ter praticamente nascido juntamente com a computação programada nas décadas de 40 e 50, deve-se salientar que a implementação de uma rede neural naquela época era inviável, pois a fase de aprendizado, a fase mais difícil e demorada no desenvolvimento de uma rede, dependia (e ainda depende) de complicados algoritmos e de um número grande de iterações, algo que um ENIAC em 1946 não teria tanta disposição de fazê-lo. Hoje, com a tecnologia dos chips VLSI, a implementação das redes neurais tem sido facilitada.

Todas as informações aqui expostas nos levam a crer que o campo de redes neurais artificiais é acima de tudo extremamente vasto e promissor. Por ser um assunto que surgiu há muito tempo atrás, ganhou muita credibilidade, e devido às novas descobertas relacionadas a ela a cada instante, tornou-se bastante atrativo para profissionais de domínios distintos, tornando-se um assunto interdisciplinar. Os conhecimentos obtidos até hoje atraem o interesse de profissionais tais como psicólogos, neurofisiologistas, engenheiros, cientistas cognitivos, e cientistas da computação, que buscam, cada um em sua área, novos caminhos através da computação neural.

Capí tulo 4

Sistema de detecção das peças de xadrez

Este capítulo aborda na prática alguns conceitos, descritos em capítulos anteriores e apêndices, utilizados para implementar o sistema de detecção das peças de um tabuleiro de xadrez. Será visto detalhadamente todo o processo de implementação do sistema de detecção em modo *off-line* e *on-line*. No final serão apresentados resultados experimentais obtidos e serão analisados sucintamente alguns aspectos do desempenho do sistema.

4.1 Introdução

Inicialmente será mostrado o sistema robótico a ser acoplado ao sistema de visão. Depois será descrita a implementação do sistema de detecção *off-line* utilizado anteriormente ao início deste projeto de dissertação [RFC00], denominado NeuroMorfo. A seguir será descrito o projeto e a implementação de um novo sistema de detecção *off-line* proposto com o intuito de corrigir problemas inerentes ao sistema original, denominado NeuroMorfo com SPI. Este novo sistema utiliza princípios do Sistema de Previsão Inteligente [MCSS99].

4.2 O sistema robótico

No NEUROLAB (Laboratório de Redes Neurais e Automação Inteligente), são desenvolvidos robôs para pesquisas tomando como plataformas de testes a música e os jogos. Esses robôs utilizam somente sensores ópticos para detectar a posição dos eixos dos motores elétricos de corrente contínua. Portanto, a não existência de sensores de visão limitou a operação destes robôs. Para suprir esta deficiência, foi desenvolvido o AutoXad, um sistema robótico com realimentação visual, projetado para fazer a interface entre o computador e o tabuleiro do jogo de xadrez [FFCAF99]. A Figura 4.1 mostra uma foto do AutoXad, constituído de uma câmara QuickCam P/B (indicada pela seta), de um braço robótico, e de um microcomputador PENTIUM que realiza as tarefas de visão do tabuleiro de xadrez, e as tarefas de controle do sistema.

Figura 4.1 – O AutoXad.

O robô manipulador é composto de uma base, três braços (L1, L2 e L3) e uma garra (G), conforme ilustrados na Figura 4.2. As juntas dos braços são acionadas por motores de passo, sendo que a junta 1, responsável pelo movimento do braço L1, gira em torno da normal ao Eixo Base (Figura 4.3), executando movimentos para a direita e esquerda. A junta 2, responsável pelo movimento do braço L2, executando movimentos para cima e para baixo e a junta 3, responsável pelo movimento do braço L3, executando movimentos também para cima e para baixo. A localização dos motores (M1, M2 e M3) é ilustrada na Figura 4.2. O motor M1 movimenta o braço L1 através de um conjunto de roldanas. O motor M2 preso à base do braço L1 movimenta o braço L2 com um sistema de transmissão por cabos, e o motor M3 preso ao braço L2 movimenta o braço L3 também com um sistema de transmissão por cabos. O braço L3 possui uma garra eletromagnética (eletroímã) em sua extremidade (G) [FFCAF99].

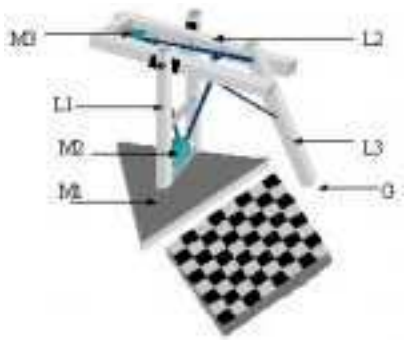


Figura 4.2 – Modelo do AutoXad

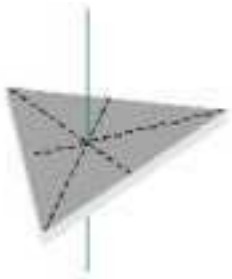


Figura 4.3 – O Eixo Base

4.3 Sistema de detecção *off-line* das peças

4.3.1 O NeuroMorfo

A QuickCam [Qui95] possui foco de 34,5 cm para o infinito e está a uma altura de 34,7 cm em relação à superfície do tabuleiro. Cada casa do tabuleiro possui uma dimensão de 2,7cm x 2,7 cm, de tal forma que uma casa na imagem pode ser enquadrada por uma moldura de 27 x 27 pixels, havendo uma correspondência, aproximada, de 1 pixel na imagem capturada pela câmara para 1 mm na superfície do tabuleiro. A Figura 4.4 mostra a tela principal do projeto original do NeuroMorfo com uma imagem do tabuleiro de xadrez (formato BMP) obtida pela QuickCam.

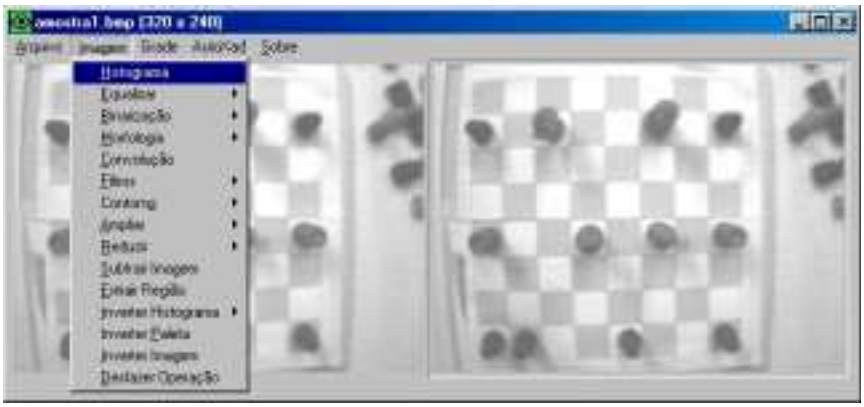


Figura 4.4 – NeuroMorfo

A imagem do tabuleiro mostrada na tela principal do NeuroMorfo foi obtida pela câmera via software original da QuickCam, denominado QuickPICT, e armazenada em arquivo. Posteriormente o NeuroMorfo lê a imagem do arquivo e a exibe em sua interface.

O laboratório é iluminado por três lâmpadas fluorescentes. A disposição do tabuleiro em relação às lâmpadas gera uma variação da luminosidade ao longo do mesmo tal que, as casas que

estão próximas à parede na região inferior da imagem, na Figura 4.5, apresentam níveis de cinza menores que as que estão na frente.

A equalização da imagem é feita através do espalhamento e deslocamento dos níveis de cinza, multiplicando-se cada nível por um fator obtido pela divisão entre 255 e o maior nível encontrado na imagem. Na Figura 4.5 pode-se observar a sombra das peças ao longo do tabuleiro. Vale ressaltar que esta não era uma equalização de fato, apenas um espalhamento dos níveis de cinza. Na seção seguinte (seção 5.2.2) será comentada a implementação da equalização de histograma na versão modificada do NeuroMorfo por meio de uma função de distribuição de probabilidade.

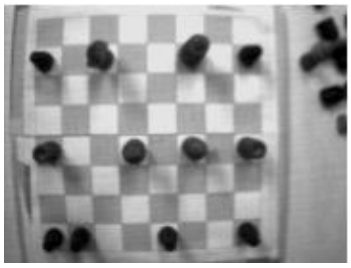


Figura 4.5 – Imagem equalizada do tabuleiro de xadrez.

A imagem do tabuleiro está contida em um quadrado de 216mm de lado, e a imagem da QuickCam tem a dimensão 320 x 240 pixel, isto é, a imagem contém informações redundantes. Necessitou-se isolar o tabuleiro do restante da imagem. Para isto, desenhou-se (virtualmente no computador) uma grade com as dimensões aproximadas do tabuleiro (8 x 8 casas de 27 x 27 pixels cada) e ajustada manualmente sobre o mesmo. A grade possibilita ao NeuroMorfo reconhecer as coordenadas das casas do tabuleiro de xadrez. Na Figura 4.6 é apresentada a imagem do tabuleiro de xadrez juntamente com a grade. Observe-se que, devido à distorção na imagem gerada pela QuickCam, a grade não fica perfeitamente ajustada sobre o tabuleiro (em particular nas bordas). Observe-se também que a região central das casas do tabuleiro é a que apresenta a menor distorção.

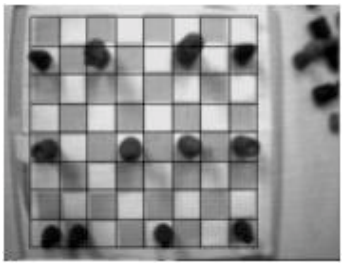


Figura 4.6 – O tabuleiro de xadrez com a grade

Na Figura 4.7 apresenta-se a imagem da ampliação (“zoom” de 300%) sobre a imagem de uma peça do tabuleiro localizada na posição 4D (quarta coluna a partir da esquerda e quarta linha a partir da base do tabuleiro de xadrez da Figura 4.5). Convencionou-se como região central da casa do tabuleiro uma área de 9 x 9 pixels (cada casa possui uma dimensão de 27 x 27) deslocada em 9 pixels a partir das coordenadas da casa (quadrado central na Figura 4.7).

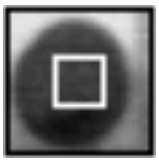


Figura 4.7 – Ampliação da imagem de uma casa do tabuleiro

A Figura 4.8 mostra a representação visual do conjunto de amostras extraídas do tabuleiro da Figura 4.5, sendo que a ordem na qual a extração ocorre é da esquerda para a direita, e de cima para baixo. Cada coluna na Figura 4.8 representa os níveis de cinza presentes na região central de uma casa como mostrado na Figura 4.7.

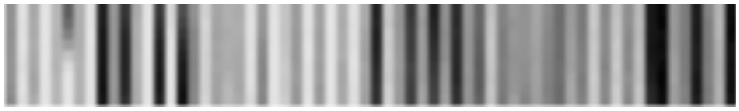


Figura 4.8 – Amostras extraídas do tabuleiro

Análise da imagem usando redes neurais

A análise da imagem da casa do tabuleiro feita na região central com 9 x 9 pixels (81 pixels) do quadrado da Figura 4.7 foi feita utilizando redes neurais artificiais. Devido às características da rede neural utilizada na análise da imagem, decidiu-se reduzir a dimensão dos dados (81 pixels) a serem analisados. Calculou-se a média aritmética de cada linha da amostra de 9 x 9 pixels e obteve-se um vetor com nove valores. Este vetor foi aplicado à entrada da rede neural.

Na Figura 4.9 apresenta-se o conteúdo do vetor (valor numérico entre 0 e 255) das casas representadas por 8TD (oitava linha torre de dama), 8CD e 8BD. Essas casas não contêm peça e portanto são claras, apresentando níveis altos de intensidade, principalmente por estarem numa das zonas mais iluminadas do tabuleiro.

138 142 146 147 147 146 143 146 144
222 226 225 220 224 226 225 224 227
172 164 163 168 169 172 170 171 167
Sem Peca

Figura 4.9 – Valores dos pixels de três casas sem peças do tabuleiro

A Figura 4.10 mostra o conteúdo correspondente às três casas representadas por 7TD, 7CD e 7BD. As casas 7TD e 7BD estão ocupadas por uma peça, os níveis correspondentes à peça possuem valores muito baixos. A casa 7CD não é ocupada por uma peça.

20 18 19 21 21 24 24 29 30
152 153 153 152 150 149 150 150 152
46 38 36 34 34 36 39 46 52
Com Peça

Figura 4.10 – Valores dos pixels de três casas do tabuleiro

Utilizou-se uma rede neural multicamada (RNMC) [RM86] do tipo Perceptron com 3 camadas (Figura 4.11). A RNMC possui 9 neurônios na camada de entrada, 8 neurônios na camada escondida, e 1 neurônio na camada de saída. Os 9 neurônios da camada de entrada possuem função de ativação linear, servindo apenas para transformar os valores da entrada em partes por unidade (peso igual a 255). Os demais neurônios da RNMC possuem função de ativação sigmóide. O treinamento dos pesos e parâmetros da rede (W, β, θ, τ) foi realizado usando o algoritmo da propagação retroativa do erro (APR). Tal função de ativação sigmóide é representada pela Equação 4.1.

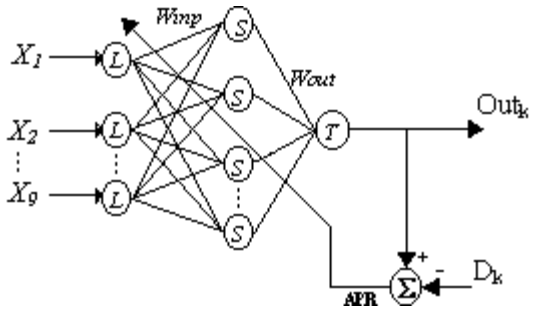


Figura 4.11 – Rede Neural Artificial Multicamada.

$$S(Y_i,\theta,\beta,\delta) = \frac{\theta}{1+e^{-\beta((\sum X_i W_i)+\tau)}} \qquad (4.1)$$

onde:

$S(\ )$ representa a função de ativação sigmóide;

$Y_i = \sum X_i W_i$, é um número real que especifica o grau de ativação do neurônio;

X_i é um vetor com os valores de entrada da rede neural;

W_i é um vetor com os valores dos pesos da rede neural;

θ é o valor máximo da sigmóide;

β representa a declividade da sigmóide;

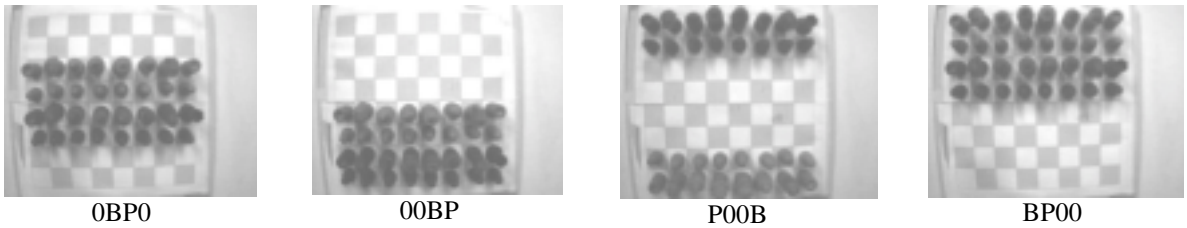
τ representa o deslocamento da sigmóide.

Atribuíram-se os valores 1 (um) quando há uma peça sobre a casa e 0 (zero) quando não há peça. A Figura 4.12 mostra a janela representando a existência (valor 1) e a ausência de peças no tabuleiro apresentado na Figura 4.5.

Figura 4.12 – Classificação do tabuleiro efetuada pela RNMC

O conjunto de valores utilizados no treinamento da RNMC foi escolhido a partir da região central das casas do tabuleiro. A rede aprenderá através do conjunto de treinamento apresentado à mesma (conjunto de vetores representando padrões). A escolha de um conjunto de treinamento que represente de forma adequada o problema visado, e tenha um tamanho adequado, é de fundamental importância para o bom desempenho da rede. Considerando tal importância, foi utilizado um conjunto de quatro imagens básicas mostradas na Figura 4.13 como amostras para realizar o treinamento da rede. O treinamento da rede está relacionado a três estados possíveis:

- Presença de peça branca (B);
- Presença de peça preta (P);
- Ausência de peça (0).



0BP0 00BP P00B BP00

Figura 4.13 – Imagens básicas utilizadas para formar o conjunto de treinamento

Depois que as amostras são obtidas, o conjunto de treinamento está formado como mostra a Figura 4.14. Cada linha, representando o conjunto de treinamento para cada casa, contém quatro vetores com nove elementos cada (intercalados por um vetor com nove elementos nulos).

Figura 4.14 – Conjunto de treinamento

A Tabela 4.1 ilustra mais claramente o conjunto de treinamento utilizado para treinar a rede neural (Figura 4.14). Esta é uma disposição das amostras que melhor representa o conjunto de treinamento ideal pois faz a rede neural aprender através de seqüências repetitivas e desordenadas das amostras (P – 0 – B).

Tabela 4.1 – Representação do conjunto de treinamento.

Amostras / Casas :	**1**	**2**	**3**	**4**
1 - 16	0	0	P	B
17 - 32	B	0	0	P
33 - 48	P	B	0	0
48 – 64	0	P	B	0

A RNMC foi treinada com 1000 iterações utilizando as 4 amostras básicas. No final foram realizados os testes com a imagem do tabuleiro de xadrez apresentada na Figura 4.5. A Figura 4.12 mostra o resultado da classificação das casas do tabuleiro.

4.3.2 O NeuroMorfo com SPI

Nesta seção são descritas as modificações utilizadas sobre o projeto original apresentado na seção anterior com o intuito de aperfeiçoar o sistema de detecção das peças no tabuleiro de xadrez. Tais modificações estão baseadas na utilização do Sistema de Previsão Inteligente como nova ferramenta de detecção das peças. Todo o código referente ao processo de detecção no sistema original foi retirado nesta nova versão. O resultado esperado com estas mudanças é uma melhor detecção das peças com menor quantidade de erros e com um menor tempo de treinamento possível. No final do capítulo serão apresentadas comparações gráficas entre o NeuroMorfo original e o NeuroMorfo com SPI.

O NeuroMorfo é capaz de realizar diversas funções de processamento de imagens. Ele está apto para exibir o histograma da imagem além de equalizá-la e binarizá-la. Pode também realizar algumas operações da morfologia matemática como erosão e dilatação binária da imagem. É capaz de efetuar a convolução da imagem, filtragem da média e extração de contorno por meio de algoritmos específicos como o de Sobel, Prewitt, Laplaciano e Morfológico. Pode também ampliar ou reduzir (*zoom in – zoom out*) uma imagem várias vezes, subtrair uma imagem da outra, extrair regiões da imagem, inverter o histograma da imagem (gerando uma imagem negativa), e inverter também a própria imagem.

Nesta fase do projeto o NeuroMorfo foi modificado para melhor detectar as peças no tabuleiro de xadrez. Embora a sua nova interface, tela gráfica mostrada na Figura 4.15, não tenha

sido sensivelmente modificada, sua maior modificação está em sua estrutura interna onde foi inserido o algoritmo do Sistema de Previsão Inteligente (SPI), além de alterar o algoritmo responsável pela equalização de histograma.

Figura 4.15 – Interface do NeuroMorfo

Equalização de histograma

O NeuroMorfo original não realizava uma verdadeira equalização sobre a imagem, apenas um espalhamento nos seus níveis de cinza. Na verdade, a equalização de histograma é uma técnica a partir da qual se procura redistribuir os valores de tons de cinza dos pixels em uma imagem, de modo a obter um histograma uniforme, no qual o número (percentual) de pixels de qualquer nível de cinza é praticamente o mesmo. Para tanto, utiliza-se uma função auxiliar, denominada função de transformação. O algoritmo de equalização de histograma utilizado baseou-se na função da distribuição de probabilidade expressa pela Equação 2.1. A Figura 4.16 ilustra o resultado obtido com este novo algoritmo de equalização sobre a imagem de um tabuleiro de xadrez.

Imagem original

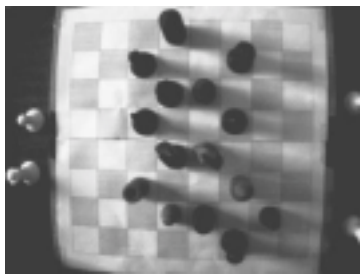


Imagem equalizada

Figura 4.16 – Resultado da equalização de uma imagem.

A Figura 4.17 exibe graficamente a distribuição de pixels na imagem original e na equalizada da Figura 4.16. Percebe-se que este novo algoritmo de equalização tenta distribuir uniformemente os pixels ao longo da faixa de níveis de cinza, tornando as partes escuras da imagem mais claras.

Histograma original Histograma equalizado

Figura 4.17 – Distribuição gráfica do percentual de pixels versus níveis de cinza.

Sistema de Previsão Inteligente

As medições, chamadas de amostras, de uma dada característica de um fenômeno ou variável de um sistema, a cada intervalo regular de tempo, são denominadas de série temporal. Assim, uma série temporal é qualquer conjunto de observações ordenadas no tempo [MTGM89]. Quando a grandeza em observação é de natureza aleatória, refere-se à série temporal como uma realização de um processo estocástico. Neste caso, e de modo geral, pode-se dizer que os modelos mais utilizados para descrever as séries temporais são processos descritos por leis probabilísticas.

As séries temporais têm sido estudadas por diversas perspectivas, das quais destaca-se a análise paramétrica e a análise espectral [Pri96]. O objetivo principal destas análises constitui-se, em linhas gerais, na determinação de algumas regularidades, tais como tendência e periodicidade, e/ou na estimativa de valores futuros da série.

Uma nova abordagem para a solução deste antigo e estimulante problema pode ser encontrada no campo de Sistemas Inteligentes. O termo Inteligente significa buscar, identificar e emular a forma de processamento da informação executada pelo cérebro humano em situações complexas. Nestas situações não se usam conceitos explicitados em equações matemáticas, mas, sim a experiência adquirida mediante um processo específico de aprendizagem.

Usando a abordagem dos Sistemas Inteligentes, baseada em redes neurais artificiais (RNA), Lógica Fuzzy [Zad94] e Algoritmo Genético [Hol92], desenvolveu-se um aplicativo denominado Sistema de Previsão Inteligente (SPI). Porém, neste projeto de dissertação, procurou-se simplificar o sistema utilizando apenas uma parte do SPI. Portanto eliminou-se do SPI os blocos referentes a Lógica Fuzzy e Algoritmo Genético, trabalhando-se somente com redes neurais artificiais.

O SPI

O Diagrama de Blocos do SPI é mostrado na Figura 4.18, nele as amostras de uma série temporal são normalizadas e depois são agrupadas no bloco “Janela” usando para tanto o conhecimento existente sobre a série no bloco “Base do Conhecimento”.

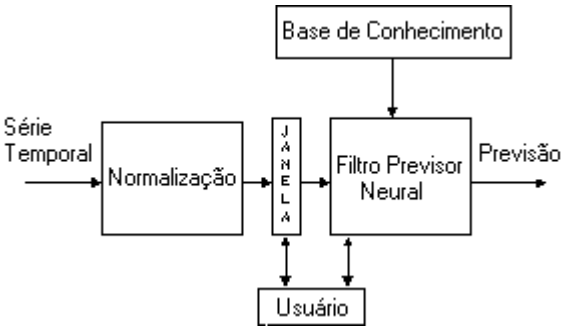


Figura 4.18 – Sistema de Previsão Inteligente.

A fase de definição do CJT (Conjunto Janelas de Treinamento) é executada de uma forma supervisionada pelo bloco "Usuário" em iteração com o bloco "Janela". O FPN (Filtro Previsor Neural), devidamente treinado utilizando o CJT fornecido pelo bloco "Usuário", executará previsões de futuras Janelas Alvos. A saída do bloco "Filtro Previsor Neural – FPN" fornece previsões de valores futuros da série [MC98].

Configuração da rede neural

O uso do SPI destinou-se apenas ao sistema de detecção de peças sobre o tabuleiro de xadrez, com o intuito de melhorar o seu desempenho. Deve-se salientar que o projeto de dissertação utilizou conceitos do Sistema de Previsão Inteligente de forma bem simplificada. Portanto, não foram utilizados processos de fuzzificação e algoritmo genético no projeto.

A rede neural foi estruturada de forma a prever os níveis de cinza da casa seguinte seguindo a ordem indicada pelas setas na Figura 4.19. Esta ordem garante uma seqüência perfeita de casas brancas e casas pretas. Isto é importante para o bom treinamento da rede neural.

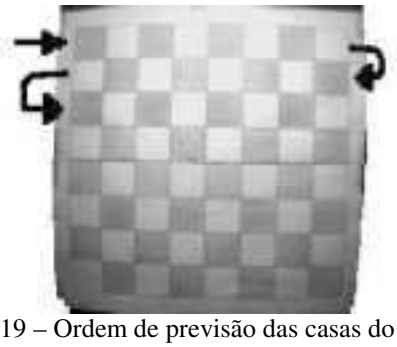


Figura 4.19 – Ordem de previsão das casas do tabuleiro

Utiliza-se no treinamento da RNA um conjunto constituído de igual número de janelas de entradas e janelas alvos, denominado *Conjunto de Janelas de Treinamento – CJT* [MCSS99]. A Figura 4.20 mostra um CJT constituído de três janelas, cada uma com o mesmo número de pontos na Janela Entrada (9 pontos) e Janela Alvo (6 pontos). Cada pequeno retângulo na figura representa um ponto (uma amostra) da série.

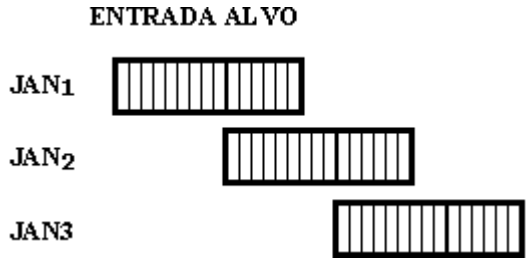


Figura 4.20 – Um CJT de três janelas.

Uma Janela Entrada representa uma casa no tabuleiro de xadrez, uma Janela Alvo representa uma parte da casa seguinte (seguindo a ordem estabelecida na Figura 4.19), e uma amostra da série (representado pelos pequenos retângulos) corresponde a um valor do nível de cinza (dentre nove valores) presente em uma casa.

Será visto agora como o SPI foi implementado no NeuroMorfo a fim de realizar a detecção das peças no tabuleiro de xadrez. O SPI é um sistema que utiliza ferramentas baseadas em Redes Neurais Artificiais desenvolvido para previsão de valores futuros de uma série temporal. Tal sistema utiliza uma Rede Neural Multicamada (RNMC) do tipo Perceptron com algoritmo de treinamento de retropropagação do erro ("*error backpropagation*"). Portanto o problema de detecção de peças no tabuleiro é abordado no campo de Sistemas Inteligentes.

O NeuroMorfo original utilizava um modelo de RNA para detecção de peças no tabuleiro de xadrez restrito a quatro situações diferentes, ilustradas na Figura 4.13, como amostras para treinamento da rede neural. Embora tal modelo apresente bons resultados, ele requer do microcomputador um número de processamento razoável para encontrar o conjunto de pesos ideal. Portanto passou-se a estudar o uso do Sistema de Previsão Inteligente (SPI) como forma de aprendizagem da Rede Neural Artificial com o intuito de fazer a detecção das peças no tabuleiro de xadrez.

Inicialmente foi necessário adquirir os dados da série. Tais dados correspondem aos níveis de cinza de cada casa do tabuleiro. Na verdade, para reduzir ao máximo a quantidade de dados, cada casa foi representada por nove valores, onde cada valor corresponde à média aritmética dos nove níveis de cinza presentes em cada linha da região central da casa (Figura 4.7).

Como cada casa é representada por nove valores (variando entre 0 e 255), e o tabuleiro é constituído por 64 casas, a série é composta por 576 valores distribuídos em 64 janelas (cada janela representa uma casa do tabuleiro). Sabendo-se disso, decidiu-se implementar uma RNMC formada por 9 neurônios na camada de entrada, 20 neurônios na camada escondida e 6 neurônios na camada de saída, como mostra a Figura 4.21. A camada de entrada recebe os nove níveis de cinza de cada

casa, e a camada de saída fornece os seis valores previstos da casa seguinte. Portanto, dos nove níveis de cinza da casa seguinte, a rede neural é capaz de prever os valores de seis níveis de cinza (valores normalizados).

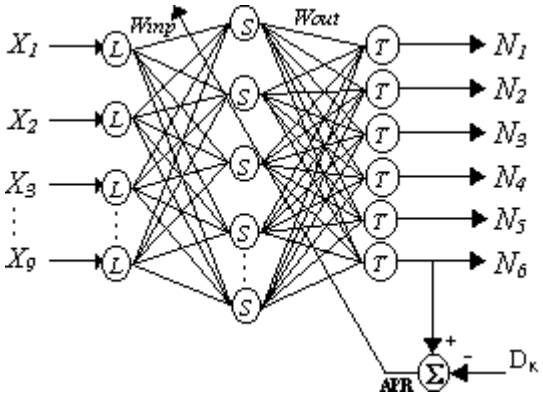


Figura 4.21 – Rede neural multicamada.

Depois de adquirido os dados, o passo seguinte foi treinar a RNMC. Decidiu-se treinar a rede utilizando os dados (níveis de cinza de cada casa) correspondentes ao tabuleiro vazio (casas sem peças), onde se pode garantir uma boa periodicidade à rede através das seqüências de casas brancas e casas pretas do tabuleiro. Para uma boa aprendizagem da rede neural, utilizaram-se dados ideais para serem previstos, ou seja, seis valores 0.8 para casa preta e seis valores 0.9 para casa branca. Tais dados foram armazenados em arquivo. Assim o sistema detecta as peças quando a diferença entre o valor previsto pela rede (obtido durante a fase de aprendizagem, quando o tabuleiro está vazio) e o valor real fornecido na saída da rede neural (obtido na fase de teste, quando o tabuleiro contém peças) for grande, já que a presença de peça em uma casa implica em um valor pequeno do nível de cinza (variando entre 0.1 e 0.3) (valores normalizados). Resumindo, quando o erro produzido em determinada casa for maior do que o permitido (obtido empiricamente), então muito provavelmente há peça nesta casa. Isto porque como foi dito anteriormente a RNMC foi treinada sob um tabuleiro vazio (sem peças).

4.4 Sistema de detecção *on-line* das peças

Serão descritos nesta seção os processos envolvidos para realizar a detecção das peças no tabuleiro de xadrez em tempo real. Isto abrange o desenvolvimento do sistema de aquisição de imagens diretamente do NeuroMorfo. Todo o processo de aprendizagem da rede neural permanece inalterado e segue o mesmo esquema descrito na seção anterior (NeuroMorfo com SPI). Inicialmente será visto os procedimentos para capturar imagens estáticas e também como exibir imagens continuamente na

interface do NeuroMorfo. Depois será visto como detectar as peças em tempo real onde qualquer deslocamento das peças sobre o tabuleiro é percebido pelo sistema.

4.4.1 Aquisição das imagens em tempo real

Como foi visto anteriormente, o NeuroMorfo original adquiria as imagens somente em arquivo numa forma *off-line*. Tal arquivo era gerado por um software proprietário da QuickCam [Qui95]. Isto impedia qualquer intenção de detecção *on-line*. Como descrito no apêndice A, foi necessário implementar um sistema de aquisição de imagens diretamente no NeuroMorfo. A Figura 4.22 mostra a nova interface do software com destaque ao menu responsável pela aquisição da imagem diretamente da câmera. O uso deste menu faz ativar as diversas funções implementadas no NeuroMorfo e descritas na seção A.2.6.

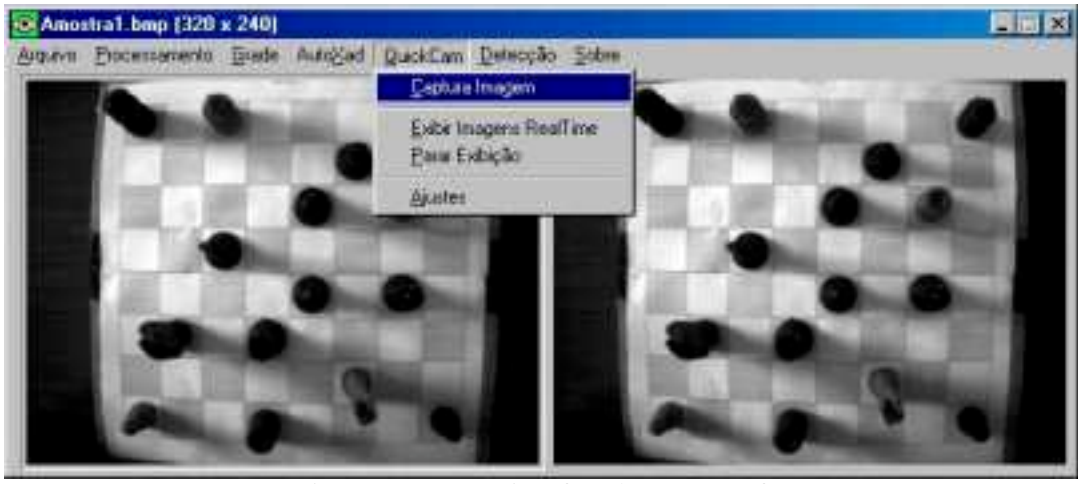


Figura 4.22 – Nova interface do NeuroMorfo.

Neste menu, a opção "*Captura Imagem*" permite a aquisição de uma imagem estática (uma imagem a cada chamada desta opção) através da câmera. A opção "*Exibir Imagens RealTime*" permite exibir as imagens em tempo real. Portanto qualquer alteração no ambiente é percebido automaticamente, embora com um certo atraso devido à demora no processamento das imagens. As imagens são adquiridas diretamente através da porta paralela, o que permite a detecção das peças em tempo real. A opção "*Parar Exibição*", ao ser escolhida, implica na parada da aquisição de imagens em tempo real. Esta opção é necessária quando o usuário desejar efetuar alguma outra tarefa no NeuroMorfo. A opção "*Ajustes*" exibe uma janela (Figura 4.23) onde é possível realizar ajustes no brilho, no contraste e no equilíbrio do branco da imagem.

Figura 4.23 – Ajustes da imagem da câmera.

Alguns pontos importantes para a implementação do sistema de aquisição de imagens no NeuroMorfo foram apresentados no apêndice A. A funcionalidade deste sistema permite adquirir tanto imagens estáticas como dinâmicas possibilitando a detecção das peças no tabuleiro de xadrez tanto no modo *off-line* como *on-line*.

4.4.2 Detecção *on-line* das peças

Para realizar a detecção das peças em tempo real deve-se seguir alguns poucos passos via interface do software. Primeiro é necessário o NeuroMorfo reconhecer a posição do tabuleiro na imagem. Portanto precisa-se desenhar uma grade sobre o tabuleiro (Figura 4.6) para obter as coordenadas do tabuleiro na imagem e isto é efetuado manualmente via a janela "*Grade*" apresentada na Figura 4.24. Depois que a grade coincide com o tabuleiro basta salvar a posição por meio do botão "*Salvar Posição*" situado no canto inferior esquerdo da janela "*Grade*".

Figura 4.24 – Janela "*Grade* " utilizada para posicionar uma grade sobre o tabuleiro.

O passo seguinte é definir o número de iterações e o valor do passo, ou taxa de aprendizagem. Isto é feito via a janela "*Treinamento*", mostrada na Figura 4.25, usando o botão "*Salvar Iteração e Passo*".

Figura 4.25 – Janela "*Treinamento* ".

Finalmente o passo seguinte é iniciar o processo de detecção *on-line* via o menu destacado na interface do NeuroMorfo da Figura 4.26.

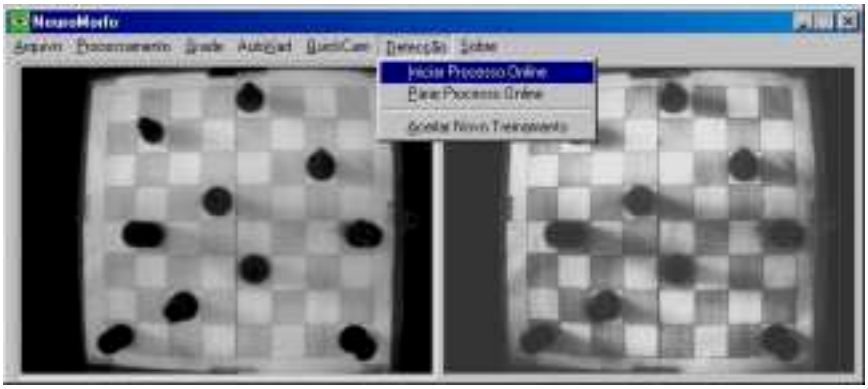


Figura 4.26 – Interface do NeuroMorfo e menu "*Detecção* ".

A opção "*Iniciar Processo On-line*" inicia o processo de detecção *on-line*. Este processo realiza automaticamente o treinamento da rede neural com base nos dados salvos nos passos anteriores e depois informa quantas peças foram detectadas e onde estão posicionadas por meio da janela mostrada na Figura 4.27. Esta janela é exibida continuamente no processo de detecção *on-line* e atualiza seus valores de três em três segundos aproximadamente. Este foi um tempo estimado para que o usuário pudesse deslocar uma peça sobre o tabuleiro. O objetivo é que posteriormente este sistema de visão seja acoplado a um sistema robótico e, conseqüentemente, este tempo será adaptado ao tempo que o braço robótico leva para deslocar uma peça sobre o tabuleiro sem que sua estrutura mecânica fosse também capturada pela câmera. A opção "*Parar Processo On-line*" finaliza o processo de detecção *on-line* e a opção "*Aceitar Novo Treinamento*" permite que a rede neural seja novamente treinada.

Figura 4.27 – Janela "*Detectando Peças* ".

4.5 Testes e resultados experimentais obtidos

Até o momento foi comentada detalhadamente a funcionalidade do sistema de detecção de peças no tabuleiro de xadrez tanto no NeuroMorfo original (que utiliza um conjunto de treinamento específico mostrado na Figura 4.14) como na última versão deste software (que utiliza princípios do Sistema de Previsão Inteligente). Como foi visto, os princípios utilizados em cada um deles são bastante diferentes. Agora serão mostrados e comentados os resultados obtidos de uma série de testes realizados sobre as duas versões do NeuroMorfo.

Para comparar as duas versões, foram estabelecidas condições iguais de ambiente e de aquisição das imagens. Os dois sistemas foram analisados sob uma iluminação comum formada apenas por uma lâmpada incandescente de 40 Watts. Para isso foi necessário encontrar empiricamente, por meio da janela mostrada na Figura 4.23, um conjunto ideal de valores de brilho, contraste e equilíbrio do branco na imagem adquirida. A aquisição de imagens era feita distintamente para as duas versões. Ou seja, para o NeuroMorfo original a imagem era adquirida via software da QuickCam (com 64 níveis de cinza), e para o NeuroMorfo com SPI a imagem era adquirida diretamente do próprio software (com 256 níveis de cinza). As duas imagens eram iguais (mesmo valores de brilho, contraste e equilíbrio do branco e contendo a mesma distribuição de peças no tabuleiro) e foram armazenadas em arquivo para uma análise posterior.

Para realizar os testes decidiu-se adquirir dez imagens diferentes do tabuleiro, cada uma delas contendo uma distribuição aleatória de 32 peças (16 peças pretas e 16 peças brancas). Cada uma destas imagens está representada no eixo *x* dos gráficos mostrados nas Figuras 4.28 e 4.29. A distribuição 0 representa uma imagem com 32 peças distribuídas aleatoriamente, a distribuição 1 representa uma outra imagem com outra distribuição de 32 peças, e assim por diante.

Figura 4.28 – Acertos e falsas detecções obtidos do NeuroMorfo original.

O resultado dos testes referentes ao NeuroMorfo original está representado na Figura 4.28. A Figura 4.29 mostra os resultados obtidos com a última versão do NeuroMorfo. Nos dois casos, a rede neural foi treinada com 1000 iterações.

Figura 4.29 – Acertos e falsas detecções obtidos do NeuroMorfo com o SPI.

Os gráficos representam o acerto (detecção correta) e a falsa detecção (detecção de uma peça onde de fato não existe peça) em função das diferentes distribuições de peças sobre o tabuleiro.

Verifica-se portanto que as duas versões possuem desempenho bastante semelhante. Foram detectadas todas as peças corretamente em todas as onze diferentes distribuições. Não houve nenhuma falsa detecção com o NeuroMorfo original. No NeuroMorfo com SPI houve falsas detecções em duas distribuições (1 e 2, conforme Figura 4.29). De fato houve apenas uma falsa detecção em cada uma destas duas distribuições. Isto ocorreu porque a nova versão do NeuroMorfo mostrou-se mais sensível à deformação tipo barril presente nas imagens. Esta deformação faz a imagem de uma peça, principalmente as peças maiores, ocupar parte de uma casa vizinha. Isto ocorre nas extremidades do tabuleiro.

Vale ressaltar que embora o desempenho das duas versões do NeuroMorfo tenha sido bastante semelhante, a grande diferença advém do número de iterações utilizadas para o treinamento da rede neural nos dois sistemas. Para o NeuroMorfo original verificou-se que era necessário treinar a rede neural com no mínimo 60 iterações para obter um desempenho satisfatório. Enquanto que no NeuroMorfo com SPI foi necessária apenas uma única iteração para que a rede neural gerasse um conjunto de pesos satisfatórios para o bom desempenho do sistema (os gráficos mostrados nas Figuras 4.30 e 4.31 refletem bem isso). Esta diferença repercute diretamente no tempo de treinamento da rede neural. No NeuroMorfo original este tempo foi apenas de 1,7s gerando uma quantidade de pesos e parâmetros (W, β, θ e τ) igual a 6848. No NeuroMorfo com SPI o tempo de treinamento praticamente é inexistente (questão de milissegundos) gerando uma quantidade de pesos e parâmetros igual a 378 (cerca de 18 vezes menos).

Os gráficos nas Figuras 4.30 e 4.31 foram obtidos realizando testes de acerto e falsa detecção em função da variação do número de iterações sobre a distribuição de peças 0 analisada anteriormente.

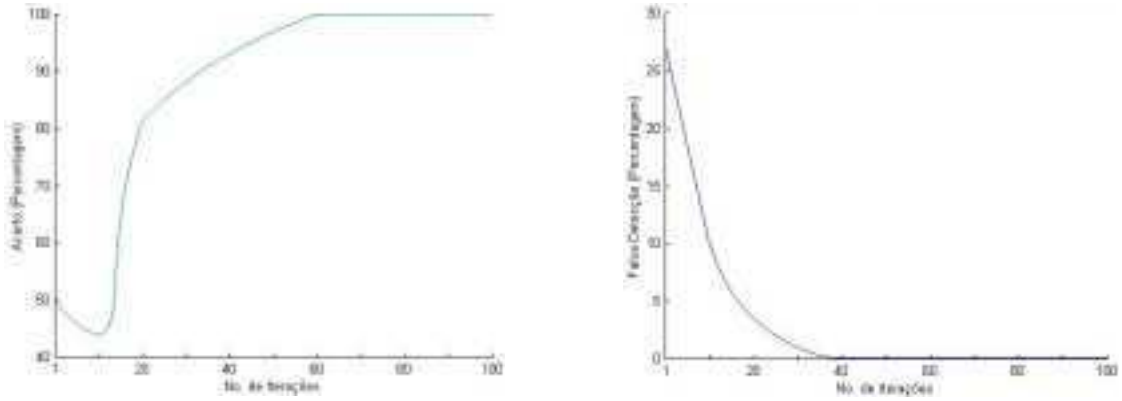


Figura 4.30 – Acerto e falsa detecção em função do número de iterações para o NeuroMorfo original.

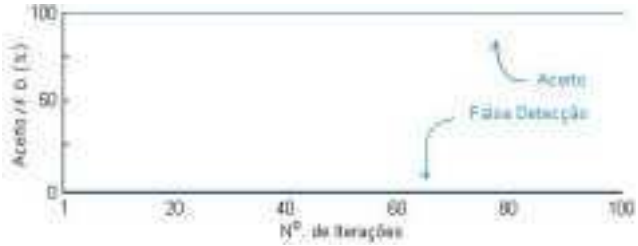


Figura 4.31 – Acerto e falsa detecção em função do número de iterações para o NeuroMorfo com SPI.

Verifica-se pelos gráficos acima que para o NeuroMorfo original apresentar um desempenho satisfatório é preciso haver no mínimo 60 iterações durante a fase de treinamento para a rede neural aprender satisfatoriamente. Enquanto que para o NeuroMorfo com SPI, a partir de uma única iteração a rede neural aprende suficientemente bem e o sistema detecta todas as peças corretamente. Esta aprendizagem super-rápida ocorre porque, como foi dito em seções anteriores, a rede neural foi projetada para prever valores (níveis de cinza normalizados) correspondentes a casas pretas e casas brancas cuja periodicidade é perfeita.

4.6 Considerações finais

Os testes realizados indicam uma boa confiabilidade das duas versões analisadas. A vantagem da nova versão sobre a original reside num tempo de treinamento praticamente inexistente com uma quantidade de pesos e parâmetros consideravelmente menor. Um ponto a ser destacado é o bom desempenho do sistema sob uma condição de iluminação não muito apropriada.

Embora não tenham ocorrido erros significativos, o sistema de detecção está condicionado a várias fontes geradoras de erro. Uma forte fonte de erro provém da distorção tipo barril intrínseco à câmera QuickCam. Esta distorção pode gerar facilmente falsas detecções principalmente quando

existem peças grandes como o rei e o cavalo nas bordas do tabuleiro. Uma outra fonte de erro consiste nas pequenas variações de brilho ao longo do tempo gerado pela câmera. Como se trata de uma câmera popular e barata, a qualidade de suas imagens não é muito boa, e, portanto, pequenas variações contínuas de brilho estão presentes. Estas variações podem causar tanto falsa detecção como a não-detecção. Sombras também são fontes de erro e causam falsa detecção. Uma outra situação que pode causar a não-detecção ocorre quando a peça está muito deslocada da região central da casa.

Capí tulo 5

Sistema de reconhecimento das peças de xadrez

Neste capítulo, apresenta-se a implementação do sistema de reconhecimento das peças no tabuleiro de xadrez. Será visto o conjunto de treinamento utilizado para treinar a rede neural bem como o método usado para extrair o vetor característica de cada peça de xadrez. Serão comentadas também as dificuldades encontradas para implementar este sistema. No final será comentado o desempenho do sistema, baseado num conjunto de testes realizados, como também as possíveis melhorias sobre o mesmo.

5.1 Introdução

Neste capítulo será visto inicialmente como ficou a versão final do NeuroMorfo. Depois serão descritos detalhes do processo de reconhecimento, do processamento da imagem obtida e da rede neural utilizada. Finalmente serão mostrados os resultados obtidos, analisando o desempenho do sistema de reconhecimento das peças de xadrez.

Será apresentado a seguir um pequeno trecho da mitologia grega com a finalidade de expor a dificuldade de reconhecimento de objetos por meio de um único sensor de visão.

Na mitologia grega o gigante Polifemo, um dos Ciclopes, era filho do Deus Netuno, e só tinha um olho. Por só ter um olho ele tinha problemas para reconhecer as formas tridimensionais dos objetos. Antes de ser cegado por Odisseu, como cantou o grande poeta Homero, na escuridão da caverna, era difícil ao gigante reconhecer quem era um grego vestido de peles ou um carneiro. Para reconhecê-los, o Ciclope sentado na sua grande cadeira, com a sua mão, tangia o ser peludo que podia ser um homem ou um carneiro. Os gregos só eram reconhecidos pelo gigante quando posicionados num certo ângulo de visão relativo ao seu olho e à fogueira acesa na caverna.

5.2 O NeuroMorfo

Durante o desenvolvimento do projeto, a versão original do NeuroMorfo sofreu uma série de modificações em sua estrutura interna e interface. A Figura 5.1 mostra a interface da última versão do NeuroMorfo.

Figura 5.1 – Interface da versão final do NeuroMorfo.

Esta versão final do NeuroMorfo inclui todos os recursos já desenvolvidos para o software como aquisição de imagens estáticas e dinâmicas, processamento digital das imagens (com todos os recursos apresentados na seção 4.1.2), detecção *on-line* e *off-line* das peças sobre o tabuleiro, e reconhecimento *off-line* das peças de xadrez.

5.3 Descrição do processo de reconhecimento

Nesta versão final do NeuroMorfo o reconhecimento das peças sobre o tabuleiro de xadrez está sendo realizado em modo *off-line*. As imagens capturadas pela câmera QuickCam são armazenadas

em arquivos do tipo BMP para posterior reconhecimento das peças. O processo de reconhecimento envolve uma série de etapas que será descrita a seguir.

A câmera Quickcam deve ser posicionada no sistema de forma que seu ângulo de visão englobe todo o tabuleiro e que a peça a ser reconhecida esteja centralizada na imagem. A Figura 5.2 exibe o modelo proposto.

Figura 5.2 – Perfil da peça rei.

A melhor posição da câmera para a captura do perfil da peça de xadrez é aquela que gera uma imagem semelhante à da Figura 5.2. A câmera deve estar situada a aproximadamente 18cm do tabuleiro e a 16cm de altura em relação ao mesmo, formando um ângulo de aproximadamente 52^o em relação à normal à superfície do tabuleiro. Ela também deve ser posicionada de tal forma que a peça a ser reconhecida esteja centralizada na imagem, como mostrado na Figura 5.2, para evitar a influência da distorção tipo barril presente na imagem.

O processo de reconhecimento pode ser resumido em alguns passos mostrados na Figura 5.3 [Cas96]. Inicialmente a imagem é adquirida digitalmente e depois ela é segmentada e o objeto extraído da imagem. O passo seguinte é gerar o vetor característica do objeto a ser reconhecido. O processo termina com a etapa de classificação do objeto. Esta etapa é implementada seguindo uma de várias possíveis soluções. Neste projeto de dissertação, foi adotada uma solução baseada em Inteligência Artificial usando redes neurais artificiais como ferramenta de classificação.

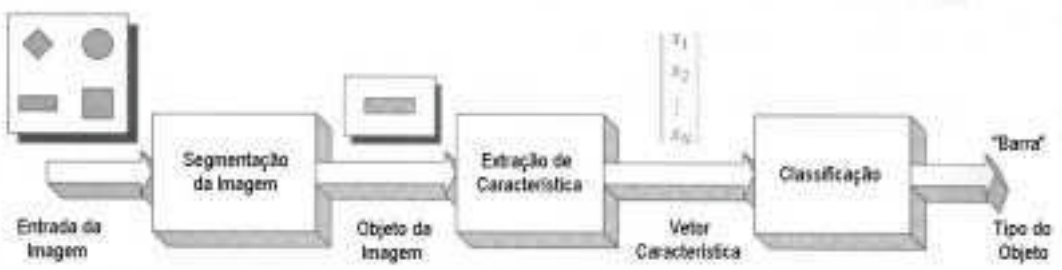


Figura 5.3 – Etapas do reconhecimento de um objeto.

Depois que uma peça de xadrez é reconhecida em uma casa no tabuleiro, uma outra peça será reconhecida em outra casa se o posicionamento da câmera estiver obedecendo às distâncias acima informadas com a peça centralizada na imagem para evitar problemas relacionados à distorção.

5.4 Extração de característica

Depois que a imagem da peça é capturada pela câmera QuickCam e armazenada em arquivo do tipo BMP de 8 bits, a mesma é processada para se obter como resultado final o *Vetor Característica* da peça analisada. A primeira etapa a ser realizada é o processo de segmentação da peça. Para isso são obtidas, via processamento, a largura e a altura da peça, extraindo a peça do tabuleiro. A Figura 5.4 mostra a peça extraída ao efetuar um clique do mouse sobre a mesma. Depois que a peça é extraída o passo seguinte é gerar o vetor característica. Este vetor é gerado a partir da imagem da peça extraída do tabuleiro.

Figura 5.4 – Peça rei extraída do tabuleiro.

Optou-se por gerar o vetor característica a partir dos níveis de cinza da cabeça da peça. Observou-se heuristicamente que ao se analisar apenas a cabeça da peça obtém-se algumas vantagens consideráveis como:

- Menor número de pixels a serem processados;
- Menor tempo de processamento para efetuar os cálculos envolvidos;
- Vetor característica com tamanho reduzido.

Portanto, ao trabalhar apenas com a cabeça da peça, obteve-se uma imagem de tamanho 32x24 (32 pixels de largura por 24 pixels de altura). A Figura 5.5 mostra a cabeça da peça rei ampliada 200% em relação ao tamanho original.

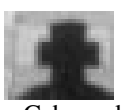


Figura 5.5 – Cabeça da peça rei.

Desta forma obtém-se um vetor formado por 768 dados (32x24 pixels), mas isto ainda é relativamente grande para servir como entrada para a rede neural. Logo se decidiu reduzir o tamanho da imagem da cabeça por um fator 4x4 (a Figura 5.6 ilustra esta redução), obtendo um vetor característica de tamanho 8x6 = 48 dados (pixels) de entrada para a rede neural. Este vetor característica é formado por dados contendo valores entre 0 e 255 correspondentes aos níveis de cinza presentes na imagem.

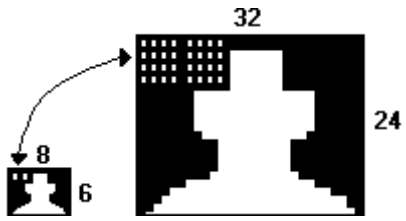


Figura 5.6 – Redução de tamanho da cabeça da peça rei.

Todo este processo envolvido para gerar o vetor característica é realizado funcionalmente pelo NeuroMorfo via os itens "*Ajustar Linha*" e "*Amostras*" do menu "*Reconhecimento*" mostrado na Figura 5.1. "*Ajustar Linha*" é um recurso que desenha virtualmente na imagem do tabuleiro uma pequena linha. A função do usuário é ajustar a localização desta linha sobre a peça por meio da janela mostrada na Figura 5.7. Depois de ajustada, os valores das posições *X* e *Y* são armazenados na memória via o botão "*Salva Posição*". A linha é uma referência que facilita o processo de segmentação da peça.

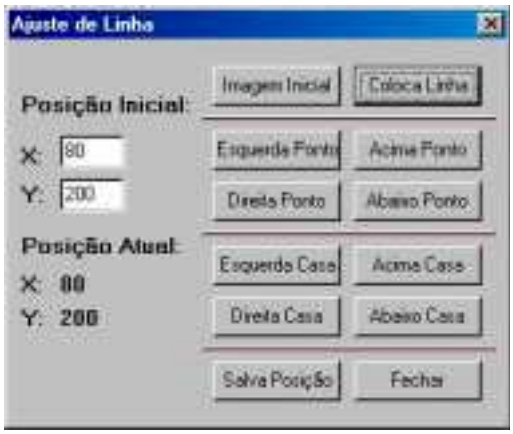


Figura 5.7 – Janela "*Ajuste de Linha* ".

O item "*Amostras*" permite gerar o conjunto de treinamento, necessário à aprendizagem da rede neural, por meio da janela apresentada na Figura 5.8. Portanto, ao carregar a imagem da peça rei (Figura 5.2), clica-se no botão "*Rei*". Isto gera o vetor característica correspondente à peça rei. Repete-se o procedimento para todas as peças e no final obtém-se o conjunto de treinamento ilustrado na Figura 5.9 (imagem ampliada).

Figura 5.8 – Janela "*Amostras* ".

Figura 5.9 – Conjunto de treinamento.

As etapas seguintes correspondem às definições dos valores de erro e passo (taxa de aprendizagem), e ao treinamento da rede neural via os itens "*Atualizar Dados*" e "*Treinar Rede*", respectivamente. Estas etapas são discutidas na seção seguinte.

5.5 Rede neural artificial

Após gerar o vetor característica, uma rede neural multicamada recebe os dados deste vetor onde os neurônios de entrada normalizam estes dados, ou seja, convertem os dados em p.u. dividindo cada valor pelo maior valor de nível de cinza presente na imagem (255). Como cada valor do vetor característica varia entre zero e 255, a entrada da rede neural varia entre os valores 0.0 e 1.0. Utilizou-se uma rede neural multicamada (RNMC) [RM86] do tipo Perceptron composta por 48 neurônios na camada de entrada (que recebe os níveis de cinza da cabeça de uma peça), 15 na camada escondida e 6 na camada de saída (que reconhece a peça), como mostra a Figura 5.10. Cada saída da rede neural está relacionada a uma peça específica.

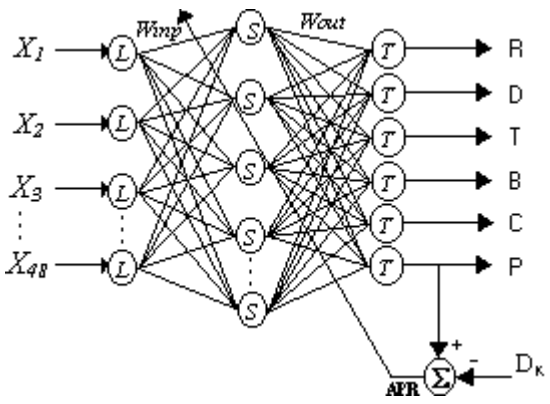


Figura 5.10 – Rede neural multicamada.

Os 48 neurônios da camada de entrada possuem função de ativação linear, servindo apenas para transformar os valores da entrada em partes por unidade. Os demais neurônios da RNMC

possuem função de ativação sigmóide. O treinamento dos parâmetros da rede (W, β, θ, δ) foi realizado usando o algoritmo da propagação retroativa do erro (APR). A função de ativação sigmóide é representada pela Equação 4.1 (apresentada no capítulo anterior). O conjunto de treinamento é formado por seis vetores representando as peças (Rei, Dama, Torre, Bispo, Cavalo e Peão) e está assim representado:

Tabela 5.1 – Representação do conjunto de treinamento.

Amostra / Linha :	**1**	**2**	**3**	**4**
1	R	R	P	P
2	T	D	T	C
3	C	T	R	B
4	P	B	D	T
5	B	C	B	D
6	D	P	C	R

As letras na tabela correspondem às iniciais dos nomes de cada peça. A Figura 5.9 mostra uma imagem ampliada do conjunto de treinamento, onde cada célula contém o vetor característica da peça correspondente na tabela acima.

O conjunto de treinamento utilizado com a disposição mostrada na Tabela 5.1 foi gerado devido à capacidade da rede neural aprender por exemplos (aprendizagem supervisionada). Esta foi a melhor disposição encontrada pois faz a rede neural aprender decorrente da repetição das amostras. Portanto não se deve apresentar um conjunto de treinamento ordenado, mas misturado, desordenadamente, pois apresentá-los ordenadamente seria como apresentar um fato (vetor característica) de cada vez, e treinar a rede 100% para o fato apresentado.

Para melhor entender a importância do conjunto de treinamento, supõe-se, por exemplo, que o mesmo fosse composto pelas amostras das peças simplesmente nesta ordem: R-D-T-B-C-P. Primeiro a rede aprende a peça rei e ajusta os seus pesos para reter esse aprendizado. Depois, a rede deve aprender a peça dama, e novamente ajusta os seus pesos para captar essa nova informação. Nesse ajuste, as modificações serão muito constantes e contínuas, e a partir dessa insistência, também serão profundos os ajustes efetuados. Na terceira apresentação, a peça torre, a rede muito provavelmente, já terá esquecido a peça rei em virtude dos ajustes para a peça dama, e os novos ajustes para a peça torre.

Conforme a Tabela 5.1, os fatos são apresentados desordenadamente, e também, reincidentemente. Para esta situação, acontecerá que, quando a rede já estava quase esquecendo a peça rei, ela aparece, e, assim, novamente, acaba reforçando o seu aprendizado (reforço sináptico).

A forma desordenada provoca uma soma dos ajustes de pesos para obter o conjunto completo treinado, enquanto que a forma ordenada, literalmente, faz o novo fato passar por cima do fato anteriormente aprendido. Deve-se considerar também que a insistência, contínua e constante, de um fato, acaba forçando demais as sinapses em um sentido, fazendo com que se esqueça o que já havia aprendido antes (lavagem cerebral).

Uma outra particularidade da rede neural utilizada foi a adoção de seis saídas. Cada saída está intrinsecamente relacionada a uma peça. A saída 1 representa a peça rei, a saída 2 representa a peça dama, a saída 3 representa a peça torre, a saída 4 representa a peça bispo, a saída 5 representa a peça cavalo, e a saída 6 representa a peça peão. Portanto depois que a rede neural for treinada, uma única saída irá se sobrepor às outras de acordo com o vetor característica apresentado à sua entrada. A taxa de reconhecimento (variando entre 0% e 100%) para uma determinada saída será superior às demais.

A condição de parada para o treinamento da rede está baseada no valor do erro final obtido. A Figura 5.11 mostra a dinâmica do erro exibido durante o treinamento. Esta janela foi importante no início da implementação do sistema de reconhecimento pois fornece informações valiosas sobre o comportamento da rede neural durante o treinamento. Através dela, verificava-se, por exemplo, se a rede neural estava divergindo ou não. Portanto, dependendo de seu comportamento, fazia-se modificações no número de neurônios da camada escondida da RNMC e também no valor da taxa de aprendizagem, com o intuito de convergir o treinamento da rede neural.

Figura 5.11 – Erros exibidos durante o treinamento.

Verificou-se que a rede aprende satisfatoriamente quando o erro final a ser atingindo for igual a 0.1, e o passo (taxa de aprendizagem) igual a 0.03 (valores definidos na janela da Figura 5.12). Estabelecendo estas condições a rede neural levou aproximadamente 5 segundos para ser treinada. Conclui-se que a fase de aprendizagem é muito rápida considerando que a rede neural é constituída de 48 neurônios na camada de entrada, 15 na camada escondida e 6 na camada de saída. Isto gera cerca de 870 dados, entre pesos e parâmetros da rede (θ, β, τ).

Figura 5.12 – Janela "*Atualizar Medidas* ".

Estas medidas são importantes para o treinamento da rede neural. O erro gerado durante a fase de aprendizagem é a diferença entre o valor de saída fornecido pela rede e o valor de saída ideal (aprendizagem supervisionada). A taxa de aprendizagem corresponde ao ajuste que deve ser dado aos pesos a cada iteração durante a aprendizagem. Deve ser um valor muito pequeno para que não faça a rede neural divergir durante o treinamento, permanecendo treinando indefinidamente. O Quadro 5.1 descreve o algoritmo de treinamento utilizado para treinar a rede neural.

Quadro 5.1 – Algoritmo de treinamento simplificado.

```
Gera_Pesos_Aleatórios ( );
Para linha = 1 até 6
{          // pega amostra de uma linha do CJT
   Pega_Amostra_Peça (Bitmap, linha);
   Pega_Pontos_Amostra ( );  // pega pontos da amostra
   Enquanto true  // loop infinito
   {
      Para amostra = 1 até 4
      {       // gera saída da rede neural
         EntraSai_RedeNeural (pontos [amostra]);
         Para k = 1 até 6
            erro [amostra] [k] = Saída_Desejada ( ) – Saída_RedeNeural ( );
         BackPropagation_RedeNeural (linha, amostra, passo);
      }
      Se erro < limite
         Break;      // sai do loop infinito
   }
}
```

Os passos do algoritmo da propagação retroativa do erro (APR) para o treinamento da rede neural multicamada são os seguintes:

1. Inicializa os pesos sinápticos e parâmetros de rede (θ, β, τ). Estes devem ser inicializados com pequenos valores aleatórios. A razão da estreita faixa é reduzir a probabilidade dos neurônios saturarem, ou seja, encontrarem uma solução estável que não provê uma saída correta.
2. Seleciona o par do conjunto de treinamento, aplicando os valores de entrada e de saída desejada.
3. Calcula a saída atual.

4. Calcula o erro entre o valor de saída obtido e o valor desejado.
5. Ajusta os pesos da rede visando minimizar o erro.
6. Repete os passos 2 a 5 para outros pares do conjunto de treinamento até que o erro global (obtido para todo o conjunto) seja suficientemente baixo.

O Quadro 5.2 descreve sucintamente os procedimentos utilizados no algoritmo da propagação retroativa do erro (APR), também denominado *backpropagation*.

Quadro 5.2 – Backpropagation

1) A cada iteração, modifique os valores de entrada à rede neural.
2) Modifique os pesos e parâmetros da rede neural de acordo com as equações abaixo.

$$E_k = D_k - Z_k$$
$$W[j]_{k+1} = W[j]_k + \varphi^* x_j^* E_k$$
$$\theta[j]_{k+1} = \theta[j]_k + \varphi^* x_j^* E_k$$
$$\beta[j]_{k+1} = \beta[j]_k + \varphi^* x_j^* E_k$$
$$\tau[j]_{k+1} = \tau[j]_k + \varphi^* x_j^* E_k$$

5.6 Reconhecimento das peças

Terminada a fase de treinamento começa a fase de testes onde o sistema está habilitado a reconhecer qualquer uma das seis peças (neste momento o formato do cursor do mouse muda para uma pequena mão quando o mesmo percorre a imagem). Para isso basta que o operador use o mouse dando um clique sobre a peça na imagem. Neste momento será exibida uma janela identificando a peça tal como mostrado na Figura 5.13.

Figura 5.13 – Reconhecimento da peça rei com um clique do mouse.

A janela exibe o nome da peça reconhecida juntamente com os valores das 6 saídas da rede neural. Cada valor foi convertido em percentagem e representa a taxa de reconhecimento da peça analisada, fornecida pelo classificador, quando houve o clique do mouse sobre a mesma.

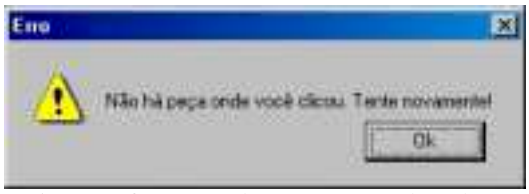


Figura 5.14 – Informação de erro provocado pelo usuário.

Esta foi uma maneira escolhida para tornar o processo de reconhecimento simples e interativo com o usuário. Caso o clique do mouse seja realizado fora da peça de xadrez o sistema retorna um erro como mostrado na Figura 5.14. De fato um simples clique do mouse sobre a peça gera uma seqüência de processamento interno na seguinte ordem:

1. a imagem é segmentada (peça extraída do tabuleiro);
2. a peça é segmentada (cabeça da peça extraída da peça);
3. gera-se o vetor característica da peça;
4. apresenta-se o vetor característica à entrada da rede neural;
5. rede neural gera a saída (classificação);
6. exibição da classificação em uma janela como mostrado na Figura 5.13.

Na verdade esta seqüência segue o raciocínio apresentado na Figura 5.3 para o reconhecimento de objetos. Esta seqüência é descrita no Quadro 5.3 em forma de um algoritmo simplificado.

Quadro 5.3 – Algoritmo simplificado para o reconhecimento de uma peça de xadrez.

```
Se treinou = true
{
   Pega_Níveis_Cinza (Bitmap);
   Extrai_Peça ( );
   Extrai_Cabeça ( );
   Reduz_4x4 (Bitmap);
   Entrada_RedeNeural (vetor);
   Para k = 1 até 6
      saída [k] = Saída_RedeNeural (k);
   Exibe_Saída (saída);
}
```

5.7 Testes efetuados e resultados obtidos

Depois que a rede neural foi treinada iniciou-se os testes de reconhecimento das peças de xadrez. Verificou-se que o sistema foi capaz de reconhecer todas as peças que constituíam o conjunto de treinamento. O passo seguinte foi verificar a capacidade de reconhecimento das peças que fossem capturadas pelo sistema de aquisição de imagens do NeuroMorfo e que não pertenciam ao conjunto de treinamento. Comprovou-se novamente sua capacidade de reconhecer todas as peças com uma boa taxa de acerto (como mostram as Tabelas 5.2 e 5.3). Deve-se ter o cuidado de clicar com o mouse na região central da peça, caso contrário o sistema reconhece erradamente a peça pois é gerado um vetor característica muito diferente daquele que a rede neural aprendeu. Isto ocorre porque o algoritmo de segmentação da peça gera extrações de peças levemente diferentes para cada região da peça clicada com o mouse. Estas pequenas diferenças são suficientes para gerar vetores características diferentes. Portanto o algoritmo de segmentação precisa ser aperfeiçoado.

Tabela 5.2 – Resultados do reconhecimento para $\varepsilon = 0.3$

Peça	***Total***	***Erro** (%)*	***Rec.** (%)*
Rei	25	8	92
Dama	25	12	88
Torre	25	16	84
Bispo	25	16	84
Cavalo	25	12	88
Peão	25	4	96
Total	150	11,33	88,67

Tabela 5.3 – Resultados do reconhecimento para $\varepsilon = 0.1$

Peça	***Total***	***Erro** (%)*	***Rec.** (%)*
Rei	25	4	96
Dama	25	8	92
Torre	25	12	88
Bispo	25	12	88
Cavalo	25	4	96
Peão	25	0	100
Total	150	6,67	93,33

Verifica-se que há um aumento na taxa de reconhecimento quando o limite de erro final (ε), que é um parâmetro para o treinamento da rede neural e serve como um critério de parada, diminui. Para $\varepsilon = 0.1$, o percentual de reconhecimento da peça peão foi de 100%. Isto ocorreu porque, além de a rede neural ter sido bem treinada, o vetor característica desta peça é bem diferenciado dos vetores das demais peças. Os testes foram realizados com todas as peças situadas em diferentes casas no tabuleiro, deslocando a câmera lateralmente e para frente e para trás manualmente para manter uma

distância fixa entre a câmera e a peça (cujas medidas foram comentadas na seção 5.3). A Tabela 5.4 descreve o tempo de treinamento gasto pelo microcomputador para fazer a rede neural aprender para um limite de erro (ε) igual a 0.1 e a 0.3.

Tabela 5.4 – Tempo de treinamento para diferentes critérios de parada.

	Tempo (segundos)
ε = 0.1	4,18
ε = 0.3	2,86

Um detalhe verificado nesta fase de testes foi a capacidade do sistema reconhecer algumas peças em casas vizinhas à casa principal (casa central livre das distorções tipo barril). Isto aconteceu, por exemplo, com as peças cavalo e bispo (Figura 5.15) que possuem vetores características diferenciados dos demais.

Figura 5.15 – Peças reconhecidas pelo sistema.

Um outro detalhe a ser considerado é o cuidado com o perfil das peças cavalo e rei a serem reconhecidos. Como são peças assimétricas, elas devem ser posicionadas no tabuleiro de forma que tenham um perfil igual ao perfil treinado na fase de aprendizagem, embora o sistema tenha demonstrado uma certa tolerância com a peça cavalo e foi capaz de reconhecê-la muitas vezes nos seus dois perfis antagônicos.

5.8 Arquivo final do projeto

O arquivo final do projeto é um executável denominado *NeuroMorfo.exe*, gerado em C++ Builder versão 4.0. Ao ser executado apresenta a interface do sistema como mostra a Figura 5.16.

Figura 5.16 – Interface do NeuroMorfo

Este arquivo possui o tamanho de 915 Kbytes e foi gerado a partir da compilação de 85 outros arquivos do tipo .CPP, .H, .BPR, .DFM e .RES.

5.9 Considerações finais

Diferentemente do sistema de detecção analisado em capítulos anteriores, o sistema de reconhecimento não foi implementado a partir de um sistema original. Muitos dos algoritmos necessários para o reconhecimento tiveram que ser implementados, como por exemplo o algoritmo de segmentação, o da geração do vetor característica e o da interface com o usuário.

Uma das grandes dificuldades iniciais foi encontrar a maneira mais simples e eficiente para gerar o vetor característica da peça de xadrez. Surgiram diversas idéias diferentes para implementar o algoritmo de extração de característica como por exemplo trabalhar com a peça inteira (ao invés da cabeça), trabalhar com as dimensões da peça, ou mesmo trabalhar com imagens da peça reduzida à metade ou a um quarto do seu tamanho. O resultado foi desanimador pois a rede neural ou não aprendia corretamente ou, como ocorria muitas vezes, não convergia e permanecia treinando indefinidamente.

Uma outra dificuldade encontrada foi conseguir gerar um conjunto de treinamento ideal para a aprendizagem da rede. Teria que ser um conjunto que fizesse a rede aprender à medida que fossem apresentados a ela novos fatos (vetores características da peças de xadrez) sem esquecer os fatos anteriores. Depois de tentar vários conjuntos diferentes, obteve-se um conjunto de treinamento satisfatório formado a partir da idéia do conjunto de treinamento utilizado no sistema de detecção original discutido no capítulo 4 e representado na Tabela 4.1.

Apesar das dificuldades encontradas para tornar o sistema eficiente, foi produtivo desenvolver uma pesquisa contínua para encontrar um bom conjunto de treinamento, uma estrutura de rede neural apropriada e uma amigável interface com o usuário. O sistema está reconhecendo todas as peças corretamente, no entanto ainda é necessário um aperfeiçoamento pois para reconhecer a peça deve-se clicar em sua região central, caso contrário o sistema irá reconhecer erradamente a peça porque será gerado um vetor característica muito diferente do que a rede aprendeu. Vale ressaltar o poder que a rede neural exerce sobre o sistema, pois, devido à sua capacidade de generalização (abordada na seção 3.7.1), a rede é capaz de reconhecer a peça mesmo com pequenas variações nos valores contidos no vetor característica (48 valores correspondendo aos níveis de cinza entre 0 e 255 presentes na cabeça da peça).

Atualmente está sendo construída uma versão deste sistema que funciona em tempo real. Para isso utiliza-se um sistema de controle para um posicionador (manipulador com dois graus de liberdade com movimentos nos eixos coordenados X e Y) que faz a câmera se deslocar apropriadamente sobre o tabuleiro.

Uma outra tarefa a ser realizada para minimizar os erros é desenvolver um algoritmo que corrija as distorções, tipo barril, presentes nas imagens capturadas pela câmera. Pretende-se utilizar um sistema de previsão inteligente [MCSS99], disponível na nossa universidade, para corrigir essas distorções.

Capí tulo 6

Conclusão

Os resultados obtidos dos sistemas implementados permitiram uma análise detalhada do desempenho alcançado. É importante ressaltar que a implementação dos sistemas de detecção e reconhecimento não foi obtida rápida e facilmente. Foi, na verdade, um desenvolvimento contínuo e árduo ao longo de dois anos em que se pretendia obter sistemas eficientes e confiáveis.

Como descrito em capítulos anteriores, os sistemas de detecção e reconhecimento possuem uma eficiência satisfatória mesmo sob condições de iluminação não muito boas. Isto demonstra a robustez do sistema como um todo.

Como visto ao longo desta dissertação, o NeuroMorfo é um software poderoso por possuir uma variedade de recursos computacionais e por permitir análises laboratoriais. No entanto há uma série de melhorias a serem realizadas e que são comentadas no final desta conclusão.

Deve-se considerar que algumas alterações deverão ser realizadas quando este sistema de visão for acoplado ao sistema robótico. Uma das principais alterações está relacionada ao tempo de ociosidade entre captura de imagens seqüenciais durante a detecção *on-line* das peças sobre o tabuleiro de xadrez. Este tempo de ociosidade deve levar em conta o tempo que um software específico (a ser desenvolvido) de análise de jogadas gasta para encontrar a melhor jogada possível a ser executada pelo sistema robótico, e o tempo que o braço robótico gasta para deslocar uma peça de uma casa para outra e sair do raio de ação da câmera.

Uma das dificuldades confrontadas durante o desenvolvimento do projeto estão relacionadas a questões técnicas, onde tive que aprender a trabalhar com uma plataforma de programação que não

conhecia (C++ Builder). Além disso, é normal que todo algoritmo desenvolvido, ao ser implementado, apresente inicialmente alguns "bugs" em seu funcionamento e estas foram dificuldades menores enfrentadas durante o mestrado e grande parte delas já superadas. Alguns pequenos problemas que ainda permanecem estão comentados na seção 6.2.

Uma outra questão a ser analisada é o porque da utilização de redes neurais artificiais como classificador para os sistemas desenvolvidos. Primeiro, o desenvolvimento do projeto proposto no início do mestrado está condicionado à área de Inteligência Artificial do Departamento de Sistemas e Computação (DSC). Segundo, o projeto já estava em desenvolvimento no NEUROLAB, onde existe uma grande quantidade de pesquisas e estudos relacionados a redes neurais artificiais como ferramenta de Inteligência Artificial. E terceiro, redes neurais, geralmente, se mostram mais eficientes que outros métodos para reconhecimento de padrões, como Veloso [Vel98] verificou em sua pesquisa comparativa entre redes neurais e o método estrutural ou sintático para reconhecimento de caracteres.

Embora o sistema de reconhecimento tenha sido elaborado para reconhecer especificamente as peças de xadrez, é possível adaptá-lo a uma aplicação industrial adicionando alguns recursos de hardware como, por exemplo, uma câmera analógica de boa qualidade e um frame grabber para a digitalização e captura de imagens dinâmicas (com 30 quadros por segundo ou mais). Certamente seria possível montar, por exemplo, um sistema de visão para reconhecimento de frutas sobre uma esteira rolante.

6.1 Contribuições

Ao longo dos dois anos de desenvolvimento do projeto podem ser citados como contribuições ao acervo de pesquisas do laboratório:

- Implementação de algoritmos para aquisição de imagens estáticas e dinâmicas diretamente no NeuroMorfo, tornando-o independente do software proprietário da QuickCam para a aquisição das imagens. Além da aquisição, foram elaborados algoritmos específicos para ajustes de:
  - Brilho
  - Contraste
  - Equilíbrio de branco

- Como conseqüência da implementação descrita no item anterior, foi possível desenvolver um sistema de detecção *on-line* das peças sobre o tabuleiro de xadrez.
- Desenvolvimento de um novo sistema de detecção das peças de xadrez a partir do sistema original fazendo uso de conceitos relacionados ao Sistema de Previsão Inteligente desenvolvido por Melo [MCSS99].
- Desenvolvimento de um novo sistema de reconhecimento das peças de xadrez. Isto ocasionou o desenvolvimento de algoritmos específicos como o de:
  - Segmentação
  - Extração de característica
  - Interface com o usuário

6.2 Trabalhos futuros

Analisando tudo o que foi feito no desenvolvimento do projeto ao longo desta dissertação, sugere-se alguns estudos para a continuação das atividades de pesquisa desenvolvidas:

- Correção da distorção tipo barril. Isto seria uma realização importante pois reduziria bastante a possibilidade de erro de detecção das peças de xadrez nas extremidades do tabuleiro, onde a distorção tipo barril é bem evidenciado. Também reduziria a taxa de erro de reconhecimento das peças de xadrez. Portanto seria necessário elaborar um bom algoritmo para corrigir este tipo de distorção inerente à câmera.
- Ajuste automático da qualidade da imagem. Para que os sistemas de detecção e de reconhecimento atuem eficientemente, é preciso que as imagens adquiridas tenham um bom ajuste de brilho, contraste e equilíbrio de branco, de acordo com a iluminação do ambiente. Atualmente isto é realizado manualmente, observando a qualidade da imagem. Kartalopoulos [Kar96] afirma que Lógica Fuzzy é uma ferramenta ideal para automatizar estes ajustes.
- Reconhecimento de peças brancas. O atual sistema de reconhecimento reconhece apenas peças pretas. Fica como trabalho futuro alterá-lo para reconhecer peças brancas também.
- Sistema de reconhecimento *on-line*. Atualmente o sistema de reconhecimento está funcionando em modo *off-line*. Este sistema tornar-se-á completo quando estiver reconhecendo em tempo real. Para isto será necessário utilizar um sistema de controle para

um posicionador (manipulador com dois graus de liberdade com movimentos nos eixos coordenados X e Y) que faz a câmera se deslocar apropriadamente sobre o tabuleiro.

Concluo esta dissertação crendo ter contribuído para o acréscimo do acervo de pesquisas do Laboratório de Redes Neurais e Automação Inteligente deste departamento, principalmente no que se refere à elaboração do sistema de aquisição de imagens em tempo real e do sistema de reconhecimento de peças que até então não tinham sido desenvolvidos. Conseqüentemente isto servirá de incentivo a outros pesquisadores da área, permitindo o desenvolvimento de novos trabalhos nesta linha.

Referências bibliográficas

[BJ90] Beale, R., e Jackson, T., "Neural Computing: An introduction", London, UK, Institute of Physics Publishing, 1990.

[Cas96] Castleman, K.R., "Digital Image Processing", Prentice Hall, 1996.

[DG87] Dougherty, E.R. e Giardina, C.R., "Matrix Structured Image Processing", Prentice-Hall, 1987.

[DL95] Dougherty, E.R., Laplante, P.A., "Real-Time Imaging", New York, USA, IEEE Press, 1995.

[Eps94] EPSON FX870/1170 Manual do usuário, São Paulo, SP, Epson do Brasil Ltda, 1994.

[Fac96] Facon, J., "Morfologia Matemática: Teoria e Exemplos", Champagnat, PUC-PR, Editora Universitária, 1996.

[FCA98] Ferreira, J.R.S., Cavalcanti, J.H.F., Alsina, P.J., "Interface Controladora de Motores de Corrente Contínua", 1998.

[FFCAF99] Ferreira, C.N.M.A., Ferreira, J.R.S., Cavalcanti, J.H.F., Alsina, P.J., e Franca, J.E., "Posicionamento Inteligente de um Braço Robótico" COBEM'99 - Congresso Brasileiro de Engenharia Mecânica, Águas de Lindóia, SP, 1999.

[FN99] Filho, O.M., Neto, H.V., "Processamento Digital de Imagens", Rio de Janeiro, RJ, Brasport, 1999.

[Gro89] Grob, B., "Televisão e Sistemas de Vídeo", Rio de Janeiro, RJ, Guanabara, 1989.

[GW92] Gonzalez, R.C. e Woods, R.E., "Digital Image Processing", 3nd Edition, Addison-Wesley, 1992.

[Hay94] Haykin, S., "Neural Networks – A Comprehensive Foundation", IEEE Computer Society Press, 1994.

[Heb50] Hebb, D.O., "The Organization of Behavior", 1950.

[Hol92] Holland, J.H., "Adaptation in Natural and Artificial Systems". MIT Press/Bradford Books edition, 1992.

[Hop82] Hopfield, J.J., "Neural Networks and Physical Systems with Emergent Collective Computational Abilities", 1982. Republicado por Sanchez-Sinencio, E., e Lau, C., "Artificial Neural Networks", New York, IEEE Press, 1992.

[Kar96] Kartalopoulos, S.V., "Understanding Neural Networks and Fuzzy Logic - Basic Concepts and Applications", New York, NY, IEEE Press, Inc., 1996.

[Koh87] Kohonen, T., "An Introduction to Neural Computing", Helsinki, Finlândia, University of Technology, 1987.

[Kov96] Kovács, Z.L., "Redes Neurais Artificiais: Fundamentos e Aplicações", São Paulo, SP, Edição Acadêmica São Paulo, 1996.

[MC98] Melo, H., & Cavalcanti, J.H.F., "Filtro Previsor Neural", Uberlândia MG, XII Brazilian Automatic Control Conference-XII CBA'98, pp.319-324, 1998.

[MCSS99] Melo, H., Cavalcanti, J.H.F., Silva, J.F., Silva, V.P.R., "Sistema de Previsão Inteligente", São José dos Campos, SP, IV Brazilian Conference on Neural Networks – IV CBRN'99 – ITA, Brazil, 1999.

[MP43] McCulloch, W.S., e Pitts, W., "Logical of the Ideas Immanent in Nervous Activity", publicado inicialmente em 1943. Republicado por Anderson, J.A., e Rosenfeld, E., "Neurocomputing Foundations of Researchs". Cambridge, Massachusets, USA, MIT Press, 1988.

[MP69] Minsky, M., e Papert, S., "Perceptron", 1969. Republicado por Anderson, J.A., e Rosenfeld, E., "Neurocomputing Foundations of Researchs". Cambridge, Massachusets, USA, MIT Press, 1988.

[MTGM89] Morettin, P.A., Toloi, C.M.C., Gait, N., e Mesquita, A.R., "Analysis of the Relationships Between Some Natural Phenomena: Atmospheric Precipitation, Mean Sea Level and Sunspots". Relatório Técnico do Departamento de Estatística da USP, pp. 2-35, 1989.

[Pra91] Pratt, W. K., "Digital Image Processing", 2nd Edition, Wiley Interscience, 1991.

[Pri96] Priestley, M.B., "Spectral Analysis and Time Series", Academic Press Inc, 1996.

[Qui95] QuickCam User Guide, Connectix Corporation, November 1995.

[RFC00] Ribeiro, S. F., França, J. E. M., Cavalcanti, J. H. F., "Um Sistema de Visão para Reconhecimento de Peças em um Tabuleiro de Xadrez", Natal RN, CONEM2000, 2000.

[Ric88] Rich, E., "Inteligência Artificial", São Paulo, SP, McGraw-Hill, 1988.

[Rip93] Ripps, D.L., "Guia de Implementação para Programação em Tempo Real", Rio de Janeiro, RJ, Editora Campus, 1993.

[RM86] Rumelhart, D.E. & McClelland, J.L., "Parallel Distributed Processing", Volume 1, Cambridge, USA, MIT Bradford Press, 1986.

[Ros58] Rosenblatt, F., "The Perceptron: a Probabilistic Model for Information Storage and Organization in the Brain", 1958.

[Sch89] Schalkoff, R.J., "Digital Image Processing and Computer Vision", Wiley, 1989.

[Ser82] Serra, J., "Image Analysis and Mathematical Morphology", Academic Press, 1982.

[Ste94] Stewart, Z., "IBM Parallel Port FAQ/Tutorial", 1994.

[TXF96] Tafner, M.A., Xerez, M., e Filho, I.W.R., "Redes Neurais Artificiais – Introdução e Princípios de Neurocomputação", Blumenau, SC, Editora da FURB, 1996.

[Vel98] Veloso, L.R., "Reconhecimento de Caracteres Numéricos Manuscritos", Campina Grande, PB, COPELE, UFPB, 1998.

[WH60] Widrow, B., e Hoff, M.E., "Adaptative Switching Circuits", primeira publicação em 1960.

[www01] [Doc – WWW: http://www.uem.br] Universidade Estadual de Maringá. UEM. Maringá, SP.

[www02] [Doc – WWW: http://www.din.uem.br/ia/neurais]. Universidade Estadual de Maringá. UEM, Maringá, SP, setembro de 2000.

[www03] [Doc – WWW: http://www.connectix.com]. Connectix, fabricante das câmeras QuickCam..

[www04] [Doc – WWW: http://rampages.onramp.net/~garym]. Código fonte QCDOS.

[Zad94] Zadeh, L.A., "Fuzzy Logic Technology and Applications", New York, IEEE Technical Activities Board, preface, p.xvii-xviii, 1994.

Apêndices

Os apêndices seguintes descrevem alguns sistemas utilizados como suporte ao desenvolvimento do projeto de dissertação. O apêndice A descreve o sistema de processamento em tempo real, utilizado para detectar as peças em tempo real. O apêndice B descreve a plataforma de programação em ambiente Windows, utilizada para desenvolver uma interface amigável entre o usuário e o NeuroMorfo.

Estes sistemas deram ao NeuroMorfo a capacidade de realizar as detecções *on-line* e *off-line* e o reconhecimento *off-line* das peças sobre o tabuleiro de xadrez. O sistema de processamento em tempo real, particularmente, foi muito importante por oferecer o dinamismo que o NeuroMorfo precisava para trabalhar em tempo real, livrando-se de outros softwares para realizar as tarefas de aquisição de imagens.

Apêndice A

Sistema de processamento em tempo real

A programação em tempo real está relacionada com a implementação de programas que controlam comutadores telefônicos, pilotos automáticos de aviões, braços de robô, ou, no nosso caso, a exibição de imagens continuamente. Uma propriedade fundamental de um programa em tempo real é que algumas, ou todas as suas entradas, são recebidas de forma *assíncrona*, em relação a qualquer trabalho que o programa já esteja fazendo. O programa deve ser capaz de interromper sua atividade corrente e então executar algum código predefinido para capturar ou responder a essa entrada. Finalizado este processo, o computador está apto para retornar a sua atividade original [Rip93].

Uma outra característica de um sistema de processamento em tempo real é que o mesmo é determinístico. Um sistema é dito ser determinístico se para cada estado possível, e cada conjunto de entradas, um único conjunto de saída e o próximo estado do sistema podem ser determinados.

Esta seção aborda vários aspectos relacionados diretamente com a implementação do processamento em tempo real utilizado neste projeto de pesquisa, como o uso de imagens bitmap, o uso da câmera QuickCam, conhecimentos sobre porta paralela, alocação dinâmica de memória, e a aquisição de imagens em tempo real.

A.1 Imagens bitmap

Existem diversos formatos de arquivos gráficos, que armazenam dados de diferentes maneiras. Tradicionalmente, os formatos de arquivos gráficos eram divididos em matriciais, vetor e metafiles, que representavam respectivamente dados matriciais, vetoriais e dados matricial e vetorial no mesmo arquivo. Esta seção aborda principalmente arquivos gráficos matriciais representados por imagens bitmap.

As imagens bitmap são exibidas em tempo real na interface do software NeuroMorfo e podem ser armazenadas em arquivos com extensão BMP. Este é um formato bitmap do Windows,

criado e mantido pela Microsoft Corporation. O BMP é um formato bitmap simples que suporta uma larga faixa de dados de imagem RGB.

Os arquivos bitmap podem ser encontrados compactados ou descompactados. Quando estão compactados não há qualquer tipo de perda, em termos de resolução, em relação a um descompactado. Neste caso os dados são comprimidos por meio de um algoritmo de compressão de dados RLE (Run Length Encoded) de 4 bits ou 8 bits.

Um arquivo bitmap possui quatro seções, nesta ordem, assim conhecidas:

- Bitmap File Header;
- Bitmap Information Header;
- Color Table / Paleta;
- Raster Data.

A.1.1 Bitmap file header

Esta seção identifica o bitmap. Um arquivo é um bitmap válido se **bfType = 19778**, que representa 'BM', ou 4D42, em hexadecimal. A Tabela A.1 apresenta os campos BFH.

Tabela A.1 – Seção Bitmap File Header

```
TBitmapFileHeader = packed record
   bfType      : Word;        // 4D42 Hex, Identifica o bitmap
   bfSize      : LongInt;     // Tamanho do arquivo
   bfReserved1 : Word;
   bfReserved2 : Word;
   bfOffBits   : LongInt;     // Posição, em bytes, no arquivo do Raster Data
end;
```

A.1.2 Bitmap info header

Os atributos do bitmap são armazenados no BIH apresentado na Tabela A.2. O cabeçalho encontra-se no começo do arquivo e contém informações sobre os dados encontrados em todos os outros lugares do arquivo. Todos os arquivos bitmap possuem algum tipo de cabeçalho. Tipicamente, o cabeçalho é composto de campos fixos e possui informações como a paleta de cores (descrito pelo campo biClrUsed), a largura e altura da imagem (campos biWidth e biHeight), o tamanho da imagem (campo biSizeImage), o número de bits por pixel (campo biBitCount), o tipo de compressão utilizado (campo biCompression), etc.

Tabela A.2 – Seção Bitmap Info Header

```
TBitmapFileHeader = packed Record
   biSize            : LongInt;    // Tamanho da estrutura TBitmapFileHeader
   biWidth           : LongInt;    // Largura da imagem
   biHeight          : LongInt;    // Altura da imagem
   biPlanes          : Word;
   biBitCount              : Word;      // Bits Por Pixel: 1, 2, 4, 8, 24 ou 32
   biCompression     : LongInt;    // Método de Compactação
   biSizeImage       : LongInt;    // Tamanho da Imagem
   biXPelsPerMeter : LongInt;
   biYPelsPerMeter : LongInt;
   biClrUsed         : LongInt;    // Quantidade de entradas na paleta
   biClrImportant    : LongInt;    // Quantidade de entradas importantes, 0 se todas
end;
```

A.1.3 Color table / paleta

O bitmap pode ter ou não uma paleta de cores. Só existe paleta para bitmaps de até 256 cores, acima disto os pixels são armazenados em triplas/quádruplos de bytes, na seguinte ordem: Red, Green, Blue (RGB). A paleta serve para aumentar as possibilidades de tonalidades de cores básicas dos dispositivos. A Tabela A.3 mostra a relação entre paleta e cores. Por exemplo, quando o bitmap é de 16 cores, cada índice de cor no arquivo, variando de 0 a 15, aponta para uma posição na paleta (tabela) de no máximo 16 entradas de 4 bytes (1 para o Red, 1 para o Green, 1 para o Blue e 1 não utilizado).

Tabela A.3 – Relação entre cores e paleta

Bits por Pixel	Cores	Paleta
1	2	Tem (002*4 bytes)
4	16	Tem (016*4 bytes)
8	256	Tem (256*4 bytes)
24	16.777.216	Não tem
32	4.294.967.296	Não tem

Deve-se ter o cuidado ao ler a paleta, pois o tamanho de entradas realmente utilizado encontra-se em **biClrUsed** no **TbitmapInfoHeader**.

A.1.4 Raster data

É nesta seção que a imagem é armazenada. Sendo armazenada de baixo para cima (down-up), com a quantidade de bytes por linha representada por um número múltiplo de 4. Em muitos formatos, os

dados bitmap vêm imediatamente após o fim do cabeçalho. Porém estes dados poderiam estar em qualquer outro lugar do arquivo, para acomodar a paleta de cores ou outra estrutura de dados presente. Se os dados não estão imediatamente depois do cabeçalho, no cabeçalho deve estar indicado a posição inicial destes dados.

O valor bits por pixel indica a quantidade de bits utilizados por cada pixel; quando ele é 8 significa que cada byte representa um índice de cor para o pixel; se for 4 significa que em 1 byte conterá no máximo dois pixels (4 bits mais à esquerda para um e 4 mais à direita para outro); se for 1 significa que cada pixel ocupará um bit, logo em 1 byte conterá no máximo 8 pixels da esquerda para direita.

Quando o valor bits por pixel for 24, como já foi dito não haverá paleta, e cada pixel terá a definição exata de RGB, portanto para cada pixel serão necessários 3 bytes.

A.1.5 Vantagens e desvantagens do formato bitmap

Os arquivos bitmap são especialmente indicados para o armazenamento de imagens do mundo real. Estes arquivos possuem as seguintes qualidades:

- Podem ser facilmente criados a partir de vetores de pixels existentes na memória.
- Os dados podem ser modificados individualmente ou em grandes grupos, alterando a paleta, se esta estiver presente.
- Podem ser facilmente traduzidos para dispositivos de saída como TRCs (Tubos de Raios Catódicos) e impressoras.

Contudo, eles apresentam as seguintes desvantagens:

- Podem ser muito grandes, especialmente se a imagem contiver uma quantidade muito grande de cores. Técnicas de compressão de dados podem reduzir o tamanho dos dados a serem armazenados, mas estes dados terão de ser descompactados antes de serem usados, o que torna mais lenta a utilização destes dados.
- Pode ser difícil modificar o tamanho das imagens, sendo às vezes conveniente imprimir a imagem na sua resolução original.

No projeto do software NeuroMorfo, as imagens são exibidas em tamanho 320x240 com 256 níveis de cinza. No capítulo 4 foi visto como o NeuroMorfo trabalha com as imagens capturadas pela câmera.

A.2 Câmera QuickCam

A câmera QuickCam tem uma característica interessante. Ao invés de produzir imagens em vídeo NTSC (sinal composto de vídeo proposto por um grupo de estudos denominado National Television System Committee), possui uma conexão direta para a porta paralela do PC. Desta forma não é preciso usar uma placa digitalizadora de vídeo. A digitalização é feita pela própria câmera. A qualidade da imagem não é tão boa quanto à obtida por outros sistemas mais dispendiosos.

A câmera utilizada no NEUROLAB é uma câmera digital QuickCam preto e branco. O coração da QuickCam é um circuito CCD ("Charge Coupled Device"), que consiste de uma matriz de células semicondutoras fotossensíveis que atuam como capacitores, armazenando carga elétrica proporcional à energia luminosa incidente. A Figura A.1 ilustra o formato físico da câmera utilizada no laboratório.

Figura A.1 – Câmera QuickCam preto e branco

A QuickCam possui foco de 33,5 cm para o infinito, e um campo de visão de aproximadamente 60^{o}. As imagens podem ser capturadas no modo 4 bits por pixel (16 tons de cinza), 6 bits por pixel (64 tons de cinza) ou 8 bits por pixel (256 tons de cinza),e podem possuir um tamanho de até 320x240 pixels. O modo de resolução é freqüentemente chamado "profundidade" do pixel. No modo 8 bpp, 0x00 corresponde ao preto e 0xFF ao branco; no modo 6 bpp, 0x00 corresponde ao branco e 0x3F ao preto; no modo 4 bpp, 0x00 corresponde ao preto, 0x01 ao branco, e 0x02 até 0x0F corresponde a uma graduação suave do quase branco para o quase preto.

Para se utilizar uma câmera QuickCam com melhor desempenho, é importante que a porta paralela do microcomputador esteja configurada para o modo bidirecional ou mais rápido. Existem três tipos de modos de comunicação que a porta paralela de um microcomputador pode usar para se comunicar com a QuickCam [www03]:

- Nibble (ou compatível),o modo mais lento;
- Bit (ou bidirecional), modo de velocidade média; e
- ECP, o modo mais rápido.

Todas as máquinas classe Pentium são capazes de comunicação bidirecional; entretanto, elas geralmente são configuradas para o modo mais lento. Para configurar o computador para o modo bidirecional ou mais rápido, é necessário modificar o seu BIOS Setup. A próxima subseção aborda conceitos de porta paralela com mais detalhes.

A.3 A porta paralela do microcomputador

A comunicação com a QuickCam é feita via porta paralela ou porta de impressora. Enquanto a câmera está em "uso", nenhuma outra aplicação (incluindo uma que queira utilizar a câmera) pode acessar a porta. Essa porta é amplamente conhecida por ser a interface de comunicação do microcomputador com a impressora. Por esse motivo, também, ela mantém uma padronização para todos os microcomputadores do tipo IBM-PC. Pode-se instalar até três portas desse tipo em um microcomputador pessoal. Cada porta contém 25 sinais que implementam a comunicação entrada/saída, sendo estes divididos funcionalmente em:

- 8 sinais de saída (para dados);
- 4 sinais de saída (para controle);
- 5 sinais de entrada (para controle);
- 8 sinais ligados ao terra.

Essa estrutura é apropriada para a sua utilização com interface controladora de robôs. O tempo de ativação ou desativação dos seus sinais também facilita a sua utilização, pois cada sinal pode ser ativado em até 200ns [Eps94]. E o fato de poder utilizar até 8 sinais diferentes para dados de saída possibilita uma comunicação mais rápida advinda do paralelismo inerente destes sinais.

A porta paralela de um microcomputador possui um endereço privativo que é dado a ela pela ROM-BIOS quando da "inicialização" do microcomputador e que permite a comunicação direta entre o microprocessador e a mesma. Este endereço pode ser alterado em placas que possuem configuração manual através de *jumps*. As placas do tipo "Plug-and-Play" ou as do tipo "on-board", possuem configuração automática e algumas permitem a mudança deste endereço através do "setup". Entretanto, seja qual for o tipo de placa instalada no microcomputador, os possíveis endereços para

uma porta paralela são somente: 0x278, 0x378 e 0x3BC. A notação "0x" à frente de cada endereço indica que os mesmos estão representados em hexadecimal. Um endereço indica o primeiro byte da porta paralela. Existem mais dois bytes para a manipulação dos sinais da porta paralela conforme descrito na tabela abaixo.

Tabela A.4 – Endereços das portas paralelas.

Porta	**Endereço de dados (saída - byte 1)**	**Endereço de estado (entrada - byte 2)**	**Endereço de controle (saída - byte 3)**
LPT1	0x278	0x279	0x27A
LPT2	0x378	0x379	0x37A
LPT3	0x3BC	0x3BD	0x3BE

Observe que o endereço da porta paralela é igual ao endereço de dados. Os outros são obtidos somando-se uma unidade para o endereço de estado e duas unidades para o endereço de controle. O relacionamento entre os endereços 0x278, 0x378 e 0x3BC com os nomes LPT1, LPT2 e LPT3 é aleatório e pode diferir de um microcomputador para outro [FCA98].

A.3.1 Modos de comunicação

Os computadores PC atuais permitem transferência bidirecional de dados em portas paralelas. Esta característica não era suportada por alguns computadores PC mais antigos. A porta paralela foi originalmente projetada para enviar bytes de dados para a impressora e receber uns poucos bits de estado como "printer *on-line*" ou "out of paper". Portas bidirecionais permitem receber bytes de dados também. Isto faz a transferência de imagens cerca de 2,5 vezes mais rápida comparado com transferências unidirecionais.

Existem três tipos de porta paralela: SPP (Standard Parallel Port), EPP (Extended Parallel Port) e ECP (Extended Capability Port). As portas EPP e ECP são extensões da SPP que permitem transferência I/O rápida para dispositivos com uma porta EPP ou ECP. A porta SPP foi projetada como saída para uma impressora. Ela não foi planejada para entrada eficiente. Entretanto ela foi posteriormente estendida para permitir entradas com velocidade mais alta, chamando-se modo bidirecional. Desde o advento do EPP e ECP (que também permitem entradas rápidas) a presença de um modo SPP bidirecional não é tão importante, assim manuais de computadores freqüentemente não mencionam a capacidade do SPP bidirecional [Ste94]. A Connectix [www03] recomenda usar o modo EPP ou o SPP bidirecional.

A.4 Alocação dinâmica de memória

A habilidade para alocar memória dinamicamente é importante na construção e manutenção de pilhas (área de memória reservada para dados e programas) na memória necessária ao sistema operacional. Embora a alocação dinâmica possa consumir tempo, ela é muito importante para muitos algoritmos de processamento de imagens que necessitam trabalhar com imagens continuamente. Listas, árvores, pilhas e outras estruturas de dados dinâmicas usadas em aplicações de processamento de imagens em tempo real podem se beneficiar da economia introduzida pela alocação dinâmica. E em casos onde apenas um ponteiro é usado para passar uma estrutura de dado, o "overhead" (sobrecarga) para alocação dinâmica pode ser bastante razoável. Ao escrever programas de processamento de imagens, cuidados devem ser tomados para assegurar que o compilador passará ponteiros para grandes estruturas de dados e não cópias dos dados.

Linguagens que não permitem alocação dinâmica de memória requerem estrutura de dados de tamanho fixo. Embora isto pode tornar o sistema mais rápido, a flexibilidade é sacrificada e requisições de memória são exigidas. As linguagens procedurais do tipo C ou C++ utilizadas no desenvolvimento do NeuroMorfo possuem facilidades de alocação dinâmica, enquanto muitas versões de outras linguagens procedurais do tipo Fortran, por exemplo, não possuem essas facilidades [DL95].

A.5 Aquisição de imagens em tempo real

No início do trabalho de dissertação, tínhamos algumas metas a alcançar e uma das principais era realizar o processamento de imagens em tempo real. Isto faria o NeuroMorfo tornar-se independente do software fornecido pela fabricante da câmera QuickCam (denominado QuickPICT). O QuickPICT não permitia ao NeuroMorfo adquirir as imagens diretamente da câmera, conseqüentemente forçava o NeuroMorfo a trabalhar em modo *off-line*. Durante a aquisição de imagens o QuickPICT arquivava as imagens no formato BMP. O NeuroMorfo posteriormente abria o arquivo da imagem no formato BMP para depois analisar e processar a imagem.

Para alcançar tal meta foi necessário implementar um algoritmo que realizasse a aquisição das imagens em tempo real diretamente da câmera sem o auxílio do software fornecido pela QuickCam. O desenvolvimento de tal algoritmo capaz de obter imagens em tempo real exige um conhecimento

adequado sobre o funcionamento da câmera. Como não havia disponibilidade de tempo para dedicar-se a uma implementação a partir do zero, optou-se por procurar na web (internet) algum software que permitisse adquirir imagens em tempo real (*on-line*) utilizando uma câmera digital QuickCam P/B. Depois de pesquisar bastante em vários "sites" na internet foi encontrado um código escrito na linguagem C, que ao ser compilado, gerava um arquivo executável que exibia imagens em ambiente DOS, com tamanho de 160x120 pixels [www04]. O software utilizava funções gráficas do MS-DOS tal como putpixel( ).

Embora o código encontrado tenha facilitado a implementação, o trabalho continuou árduo pois foi preciso estudar o código, desenvolvido para o MS-DOS, para adaptá-lo ao NeuroMorfo que trabalha sob o sistema operacional Windows. Para isso foi necessário criar um algoritmo que gerasse imagens bitmap além de ter sido necessário mudar o tamanho das imagens para 320x240 pixels.

Uma diferença encontrada entre as imagens adquiridas pelo software acoplado à QuickCam e as adquiridas agora pelo NeuroMorfo é que as primeiras têm resolução de 6 bits (64 níveis de cinza) e as últimas de 8 bits (256 níveis de cinza). Embora haja tal diferença, o NeuroMorfo foi projetado para ser compatível com as versões anteriores. Portanto é possível realizar uma equalização tanto em imagens de 6 bits como de 8 bits.

A.6 Funções implementadas

Basicamente a aquisição das imagens em tempo real é realizada por três funções:

- **QC_scan( )** – "escaneia" um quadro de imagem completo;
- **QC_reset( )** – "reseta" a câmera. Isto é sempre necessário depois de um "escaneamento" completo;
- **QC_setup( )** – possibilita modificar brilho, contraste, equilíbrio de branco e resolução da imagem.

No entanto, antes de utilizar estas funções, é necessário reservar um espaço na memória para armazenar a imagem. Isto é possível devido a funções como:

- **CreateTBitmap(***bitmap*, *largura*, *altura***)** – gera o cabeçalho bitmap com todas as informações sobre a imagem (detalhes na seção A.1);
- **GetTBitmap(***bitmap***)** – obtém dados da imagem bitmap, como largura e altura, e cria um buffer na memória para armazenar a imagem;
- **SetTBitmap(***bitmap*, *bits*, *paleta***)** – exibe a imagem bitmap na interface do NeuroMorfo.

Como visto em seção anterior, esta reserva de memória para armazenar a imagem é realizada via alocação dinâmica de memória.

A função "*QC_scan( )*" chama diversas outras funções no algoritmo, mas de fato existem duas funções "operárias" que realizam a comunicação entre o computador e a câmera:

- **outportb(***WORD porta*, *BYTE valor***)** – realiza a saída de dados enviando um byte para a porta paralela;
- **inportb(***WORD porta***)** – realiza a entrada de dados recebendo um byte pela porta paralela.

É importante ressaltar que estas duas últimas funções já existem para a biblioteca Borland Graphics Interface (BGI) em MS-DOS. Mas como o NeuroMorfo trabalha em ambiente Windows, estas funções não existem e, portanto, tiveram que ser implementadas.

Vale destacar que durante as tentativas de adaptar o código encontrado na web, que funciona em ambiente MS-DOS, ao nosso sistema que funciona em ambiente Windows, não se conseguia visualizar as imagens em tempo real e o sistema congelava embora todo o código estivesse aparentemente bem adaptado. Depois de exaustivas tentativas em eliminar o problema, verificou-se que, em ambiente Windows, era necessário utilizar uma função para interromper a execução de uma aplicação de forma que o Windows possa processar uma fila de mensagens. Descobriu-se que o C++ Builder possui tal função, assim expressa:

```
Application –> ProcessMessages( );
```

Fila de mensagens é o lugar na memória onde se armazenam mensagens que são transmitidas entre aplicações. A função "*ProcessMessages*", ao ser chamada, permite ao Windows processar as mensagens que estão correntemente na fila de mensagens.

Portanto ao inserir esta função em um laço infinito responsável pela exibição das imagens em tempo real eliminou-se o problema de congelamento pois a função permite que as imagens sejam exibidas e que o laço continue com os ciclos.

Apêndice B

Plataforma de programação

O NeuroMorfo, assim como diversos outros projetos implementados no NEUROLAB, foi desenvolvido de forma a apresentar ao usuário uma interface amigável, compreensível e de fácil interação. Isto foi possível devido ao uso do C++ Builder da Borland como plataforma de desenvolvimento.

O C++ Builder é um ambiente visual, orientado a objetos, que tem por finalidade desenvolver rapidamente aplicações para o sistema operacional Windows. Estas aplicações podem ser de propósitos gerais ou cliente/servidor. Usando o C++ Builder, o usuário pode criar eficientes aplicações Windows com o mínimo de codificação manual.

O C++ Builder disponibiliza uma extensa biblioteca de componentes reutilizáveis e um ambiente de ferramentas RAD (Rapid Application Development). Quando o usuário inicia o Builder, ele é imediatamente posto diante de um ambiente de programação visual. É com este ambiente que o Builder disponibiliza todas as ferramentas que o usuário necessita para criar, desenvolver, testar, e depurar suas aplicações. A Figura B.1 mostra a interface completa do C++ Builder.

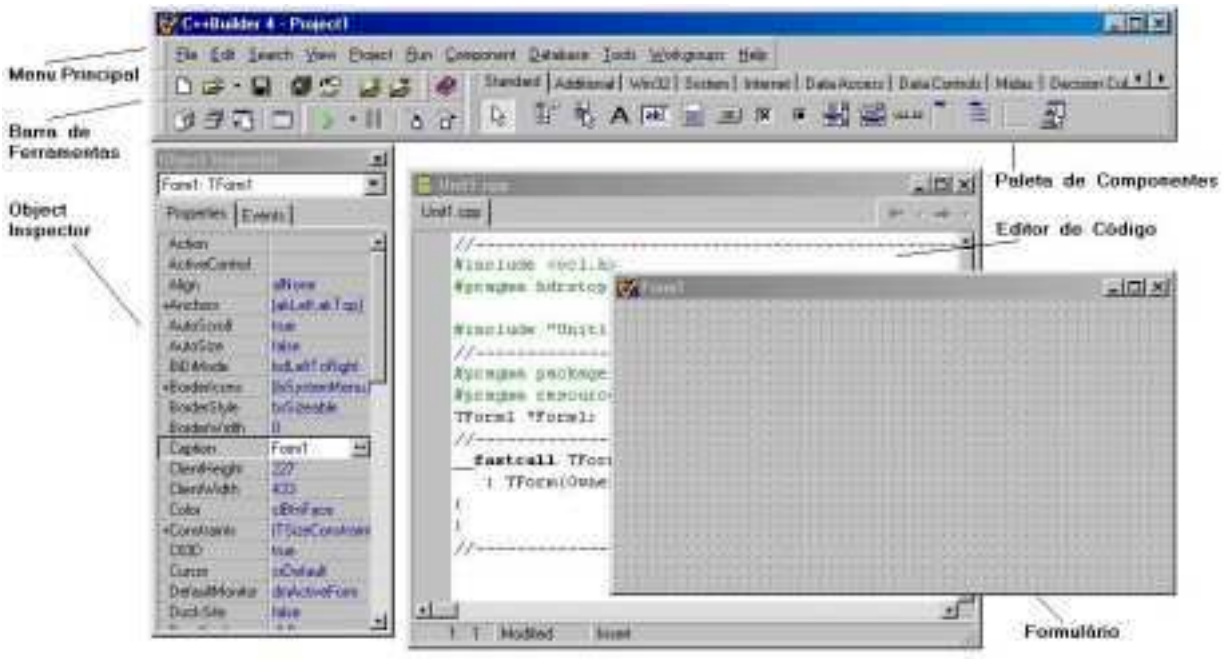


Figura B.1 – Interface da plataforma C++ Builder.

O ambiente de desenvolvimento do C++ Builder (também chamado de IDE) é formado por uma janela principal contendo o menu principal, a barra de ferramentas e a paleta de componentes;

pelo Object Inspector, que permite modificar parâmetros da aplicação; pelo Editor de Código; e pelo formulário, que será a interface da aplicação.

O C++ Builder pode ser usado para criar aplicações de 32-bits de qualquer tipo. Desde aplicações de propósitos gerais, até sofisticados programas de acesso a banco de dados.

B.1 O que é um projeto C++ Builder?

Um projeto C++ Builder é uma coleção de todos os arquivos que juntos, fazem uma aplicação executável ou uma DLL. No C++ Builder versão 4.0, os arquivos do projeto são organizados no arquivo .BPR.

À medida que a aplicação cresce, mais e mais dependências entre diferentes arquivos ocorrem e conseqüentemente a complexidade aumenta. Um programa Windows pode compor-se de script de recursos, LIB, OBJ, e códigos fonte. Cada tipo de arquivo requer um ajuste especial para ser compilado e "linkeditado" na aplicação final. Um projeto combina um ou mais arquivos fonte para produzir um arquivo final. O arquivo final pode ser, por exemplo, um .OBJ, .DLL, ou .EXE. Cada arquivo final depende de todos os arquivos usados para sua criação. Arquivos fontes são do tipo .C, .CPP, .H, e .HPP.