

TEOBALDO JAMUNDÁ

**UM SISTEMA DE VIGILÂNCIA COM DETECÇÃO
DE INTRUSÃO UTILIZANDO INTELIGÊNCIA
ARTIFICIAL**

FLORIANÓPOLIS

2002

UNIVERSIDADE FEDERAL DE SANTA CATARINA

**PROGRAMA DE PÓS-GRADUAÇÃO
EM CIÊNCIA DA COMPUTAÇÃO**

**UM SISTEMA DE VIGILÂNCIA COM DETECÇÃO
DE INTRUSÃO UTILIZANDO INTELIGÊNCIA
ARTIFICIAL**

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

TEOBALDO JAMUNDÁ

Florianópolis, Agosto de 2002.

UM SISTEMA DE VIGILÂNCIA COM DETECÇÃO DE INTRUSÃO UTILIZANDO INTELIGÊNCIA ARTIFICIAL

Teobaldo Jamundá

Esta Dissertação foi julgada adequada para obtenção do Título de Mestre em Ciência da Computação, Área de Concentração *Sistemas de Conhecimento*, e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Santa Catarina.

Prof. Fernando Álvaro Ostuni Gauthier, Dr.

Coordenador do Programa de Pós-Graduação em Ciência da Computação

Banca Examinadora:

Prof. Paulo Sérgio da Silva Borges, Dr.

Orientador

Prof. Antonio Carlos Gay Thomé, Dr.

Prof. Aldo von Wangenheim, Dr. ner. nat.

Prof. Roberto Willrich, Dr.

“Não basta ensinar ao homem
uma especialidade,
porque o tornará assim
uma máquina utilizável, mas não
uma personalidade.
É necessário que
adquira um sentimento,
um senso prático daquilo
que vale a pena ser empreendido,
daquilo que é belo,
do que é moralmente correto”

Albert Einstein

Dedico este trabalho à minha família.

Agradecimentos

Agradeço ao Prof. Paulo S. S. Borges pelo apoio, confiança, amizade, orientação e conversas que contribuíram de forma singular.

Agradeço, também, a Universidade Federal de Santa Catarina (UFSC), especialmente ao Departamento de Informática e Estatística e ao CPGCC, que proporcionaram um ótimo ambiente para o meu desenvolvimento profissional e humano.

Agradeço aos meus pais Cícero e Corporina; e aos meus irmãos Taciana, Curcio e Woldemar. Obrigado pela força e pelo amor. Vocês foram mais importantes do que possam imaginar. Agradeço, assim, a toda a minha família.

Agradeço a todas as pessoas que cruzaram a minha vida, permitindo que este trabalho fosse realizado. Muito obrigado a todos que contribuíram, muitas vezes sem saber, para evolução deste ser.

Resumo

O desenvolvimento de sistemas inteligentes com visão computacional permite realizar o reconhecimento de objetos, tornando esses sistemas capazes de responder a modificações no ambiente em que operam. Neste contexto, este trabalho propõe um sistema de vigilância inteligente, baseado no emprego de uma rede neural artificial (RNA), para verificar se o objeto em movimento em determinada cena é uma pessoa. Após extrair-se o objeto a ser analisado pela RNA através de técnicas de processamento de imagem, obtém-se a partir do contorno os descritores da Transformada Discreta de Fourier (DFT), que são apresentados à RNA. Estes estímulos, descritores da DFT, são propagados através de uma rede neural previamente treinada, gerando como resultado um vetor que se refere ao objeto reconhecido, e, se for o caso, alertando que um ser humano está transitando no ambiente monitorado.

Abstract

The recognition of objects through computer vision techniques enables the development of intelligent systems that are able to respond to modifications that may occur in their operation environment. This work proposes an intelligent surveillance system based on the use of an artificial neural network (ANN) to distinguish if the object in motion in a scene is a person or not. After computationally extracting a bitmap of the object to be analyzed by the ANN through image processing, we get the descriptors of Discrete Fourier Transform (DFT) from the contour, which are introduced to an ANN. These stimuli, the descriptors of the DFT, are propagated through a previously trained neural network, generating as a result a vector that refers to the known object, if it is the case under appraisal. The output of the system is a warning regarding the nature of the object that is moving in the environment under surveillance, informing if it is a person or not. This research has particular interest regarding applications in security systems.

Sumário

Lista de Figuras	xi
Lista de Quadros	xii
Introdução	1
1.1 <i>Preâmbulo</i>	1
1.2 <i>Objetivos</i>	2
1.3 <i>Exemplos de Trabalhos Relacionados</i>	3
1.4 <i>Motivação</i>	4
1.5 <i>Organização</i>	4
Natureza do Problema	5
2.1 <i>Introdução</i>	5
2.2 <i>Sistemas de Vigilância</i>	5
2.2.1 <i>Detetores de Movimento</i>	6
2.2.2 <i>Componentes de um Sistema de Vigilância</i>	6
2.2.3 <i>Limitações</i>	7
2.3 <i>Reconhecimento de Objetos</i>	7
2.4 <i>Classes de Objetos</i>	8
Redes Neurais Artificiais: Uma Breve Revisão Teórica	9
3.1 <i>Introdução</i>	9
3.2 <i>Neurônio Artificial</i>	10
3.3 <i>Topologia das Redes Neurais</i>	12
3.4 <i>Redes Neurais Multicamadas Feedforward</i>	12
3.4.1 <i>Aprendizado Supervisionado</i>	14
3.4.2 <i>O Algoritmo Backpropagation</i>	15
3.4.3 <i>Gradiente Descendente</i>	17
3.4.4 <i>Aprendendo a cada Lote (batch, epoch) ou a cada Iteração (pattern)</i>	19
Processamento de Imagens	20
4.1 <i>Introdução</i>	20
4.2 <i>Etapas do Processamento de Imagens</i>	20
4.3 <i>Aquisição de Imagem</i>	21
4.4 <i>Imagem e ROI</i>	22
4.5 <i>Operações Básicas</i>	23
4.6 <i>Threshold</i>	25
4.6.1 <i>Threshold Adaptativo</i>	26
4.7 <i>Filtros (Convoluções)</i>	26

4.8 Operadores Morfológicos.....	28
4.9 Contornos	30
Reconhecimento de Formas.....	32
5.1 Introdução	32
5.2 Região e Contorno de um Objeto	32
5.3 Aproximação do Contorno por uma Série de Fourier.....	35
5.4 Identificação de Objeto na Cena Monitorada.....	37
5.5 Obtenção do Contorno e Reconhecimento de Objeto.....	38
5.6 Experimentos e Resultados	39
Um Sistema de Vigilância com Detecção de Intrusão Utilizando Inteligência Artificial	43
6.1 Introdução	43
6.2 Características e Funcionamento do Sistema Desenvolvido.....	44
6.3 Implementação do Sistema de Vigilância.....	45
6.3.1 Ferramentas de Desenvolvimento	45
6.3.2 Implementação de Módulos do Sistema	45
6.4 Algoritmo de Processamento de Imagens.....	46
6.4.1 Algoritmo de Detecção de Movimento	48
6.4.2 Algoritmo de Obtenção de Objeto Intruso	50
6.4.3 Algoritmo de Reconhecimento de Objeto Intruso	53
6.5 Software de Sistema de Vigilância	54
6.6 Teste e Avaliação do Sistema de Vigilância	55
Conclusões e Futuros Desenvolvimentos	59
Referências Bibliográficas	61

Lista de Figuras

Figura 1 - Exemplos de detetores de movimento.....	6
Figura 2 - Exemplos de extração da região correspondente aos objetos em movimento.....	8
Figura 3 - Estrutura de uma Rede Neural Feedforward.....	10
Figura 4 - Modelo do Neurônio Artificial.....	11
Figura 5 - Rede Neural Artificial Feedforward com uma camada interna e uma de saída.....	13
Figura 6 – Problema do gradiente descendente.....	18
Figura 7 - Exemplo de uma ROI sobre outra ROI [35].....	22
Figura 8 - Exemplo de uma imagem após a operação de <i>Threshold</i>	26
Figura 9 - Exemplificação da Convolução.....	27
Figura 10 - Exemplo dos dois tipos principais operações morfológicas.....	28
Figura 11 - Os Elementos Estruturantes padrão N4 e N8, respectivamente.....	29
Figura 12 - Exemplo de aplicações de morfologia.....	29
Figura 13 - Exemplos de análise de contornos.....	31
Figura 14 - Exemplo de identificação do contorno de um objeto em movimento.....	33
Figura 15 - Imagem antes de ressaltar a objeto.....	33
Figura 16 - Imagem depois de ressaltar o objeto.....	34
Figura 17 - Borda de um objeto.....	34
Figura 18 - Contorno em forma de "T" e representação usando 3, 6 e 20 coeficientes de Fourier..	38
Figura 19 - Objetos utilizados para obter o conjunto de treinamento da RNA.....	39
Figura 20 - Objetos utilizados para validar o processo de reconhecimento.....	40
Figura 21 - Algoritmo de Processamento de Imagens.....	47
Figura 22 - Algoritmo de detecção de movimento.....	49
Figura 23 – Exemplo da operação de subtração entre duas imagens.....	50
Figura 24 – Comparação entre resultados de subtração.....	51
Figura 25 – Algoritmo de obtenção de objeto intruso.....	52
Figura 26 - Algoritmo de Reconhecimento de Objeto Intruso.....	53
Figura 27 - Interface do Sistema de Vigilância.....	55
Figura 28 - Imagens modelo do conjunto de teste do Sistema de Vigilância.....	56

Lista de Quadros

Quadro 1 - Exemplo de vetores do conjunto de treinamento da RNA, objetos na posição frontal..	40
Quadro 2 - Relação (codificação) entre objeto e saídas desejadas	40
Quadro 3 - Saídas da RNA (real e com função limiar) para o conjunto de validação especificado.	41
Quadro 4 - Verificação dos contornos obtidos e resposta da RNA referente aos objetos intrusos. .	57
Quadro 5 - Relacionamento entre objeto intruso e reconhecimento efetuado pelo sistema.....	58

Capítulo 1

Introdução

1.1 Preâmbulo

A idéia de possibilitar a um dispositivo reconhecer o ambiente em que atua tem motivado pesquisadores a investir esforços no estudo do mais complexo dos sentidos humanos: a visão. Como a grande parte dos processos mentais e perceptivos do homem, a visão ainda apresenta uma série de aspectos desconhecidos para a ciência, embora os últimos 50 anos tenham sido decisivos para a compreensão dos mecanismos fisiológicos essenciais ao seu funcionamento. A visão é, antes de tudo, uma tarefa de representação e processamento de informações, sendo portanto adequada ao tratamento computacional.

Conceber uma definição precisa do funcionamento da percepção visual é quase tão complexo quanto descrever seus mecanismos. Entre alguns pesquisadores do estudo da Visão Computacional, encontra-se consenso em relação aos aspectos concernentes ao propósito e domínio da visão. Conforme Ballard & Brow [3],

“ Visão Computacional é a construção de descrições explícitas e significativas de objetos físicos através de imagens. ”

O objetivo de construir um modelo do mundo a partir de imagens também consta na definição apresentada por A. Rosenfeld [37].

“ O objetivo da análise de imagens é a construção de descrições de cenas com base nas informações extraídas de imagens ou seqüências de imagens. ”

A função do sistema visual é fornecer uma representação do ambiente à nossa volta e, mais importante do que isso, prover a informação de que necessitamos para interagir com o ambiente [12].

A visão é um processo perceptivo orientado por objetivos, cujo propósito é um fator determinante para a análise dos mecanismos de funcionamento. A abordagem utilizada pelo sistema visual frente aos estímulos recebidos se baseia na economia de esforço, atuando de forma a extrair prioritariamente as informações necessárias à execução da tarefa. Seres vivos inferiores utilizam a visão basicamente para funções de sobrevivência da espécie, como a identificação de alimentos e predadores naturais. À medida em que se

sobe na escala evolutiva das espécies, a visão ganha maior poder, sem contudo perder a característica de estar centrada em objetivos.

Isso pode ser facilmente verificado nas situações de nosso cotidiano, onde, focamos e analisamos apenas o que nos interessa de uma determinada cena, apesar de poder ocorrer diversas situações simultâneas naquele ambiente. Em contrapartida, pode-se estar observando uma determinada cena a espera de que um evento específico ocorra para realizar um procedimento. Um vigia, por exemplo, fica monitorando o perímetro de uma empresa a fim de evitar que pessoas não autorizadas entrem naquele local.

Assim, este trabalho propõe a especificação de um sistema de vigilância, que consiga realizar uma fração da tarefa rotineira do sistema visual, mais especificamente, o reconhecimento de determinados objetos. Aborda-se assuntos referentes a área de sistema de vigilância, visão computacional, processamento de imagem, inteligência artificial conexionista e transformada discreta de Fourier.

1.2 Objetivos

O objetivo principal é especificar um sistema de vigilância, baseado nas técnicas de Redes Neurais Artificiais (RNA) e processamento de imagens, que seja capaz de indicar se em um ambiente monitorado há uma pessoa transitando. A idéia é integrar aos sistemas de vigilância um *software* capaz de "verificar" se há um objeto movimentando-se em determinada cena, e tentar reconhecê-lo.

O funcionamento do sistema de vigilância consiste basicamente em coletar imagens de uma cena através de uma câmara, extrair os objetos em movimento, utilizando algoritmos de processamento de imagens, a fim de gerar vetores que servirão como entradas para a RNA, que por sua vez tentará reconhecer os objetos. Caso um dos objetos seja reconhecido, o sistema realizará um procedimento previamente definido.

Objetivos Detalhados

Detalhadamente, propõem-se os seguintes objetivos específicos para este trabalho:

- Projeto de um sistema de vigilância utilizando inteligência artificial;
- Implementação de rotinas de filtragem de imagens e pré-processamento de imagens, para obter os objetos em movimento na cena monitorada;
- Implementação de rotinas de processamento de imagens para obter o contorno de cada objeto;

- Implementação dos algoritmos da transformada de Fourier sobre os contornos dos objetos;
- Implementação de uma Rede Neural para efetuar o reconhecimento (classificação) dos objetos intrusos ao ambiente monitorado;
- Análise dos resultados obtidos.

Para o desenvolvimento do projeto, foi utilizado um computador tipo PC com processador Pentium® IV, *clock* de 1.6 GHz, e com 256 MB de memória RAM. Uma *webcam* comum, com ajuste automático de luminosidade, resolução máxima de 320 x 240 *pixels*, e com taxa de aquisição de até 30 *frames* por segundos. O computador possuía o *hardware* e os *softwares* necessários para a criação, desenvolvimento e implementação do sistema.

1.3 Exemplos de Trabalhos Relacionados

Recentemente tem havido um grande número de projetos sobre detecção e *tracking* (acompanhamento) de pessoas. Esses sistemas podem ser classificados em categorias, de acordo com o tipo de sensor (câmara única ou múltiplas, imagem colorida ou escala de cinza), e suas funcionalidades (pessoa isolada, múltiplas e área de atuação). A seguir são relatados alguns projetos e a forma que operam para tentar localizar pessoas [13].

- *Pfinder* [42] e W^A [13] usam um modelo estatístico de *background* (imagem modelo) para localizar pessoas. Entretanto, o *Pfinder* usa a distribuição Gaussiana de cor para cada *pixel* (menor componente de uma imagem), enquanto que W^A emprega a distribuição bimodal de intensidade em cada *pixel*. Ambos sistemas usam a silhueta para detectar as partes do corpo; sendo que o *Pfinder* assume que há somente uma pessoa na cena. W^A permite grupos de pessoas ou pessoas isoladas em diversas posturas.
- *TI's System* [33]: é um sistema de propósito geral para detecção e eventualmente reconhecimento. Também emprega a subtração de *background* para encontrar pessoas e objetos, usando a distribuição Gaussiana de intensidade de cada *pixel*. Foi desenvolvido para vigilância em ambientes fechados e não consegue manipular pequenos movimento de objetos pertencentes ao *background*.

1.4 Motivação

Um novo paradigma na área de processamento de imagens é a técnica de Redes Neurais Artificiais [4][7][8][24][26][31], baseada na emulação de certas funções elementares do cérebro humano, como a capacidade de aprender a partir de exemplos. Uma aplicação onde as Redes Neurais Artificiais (RNA) têm demonstrado grande eficiência é o reconhecimento de padrões, mesmo na presença de imperfeições, ruídos e perturbações. Em função destas características, as RNA apresentam-se como uma solução em potencial para problemas ligados à visão computacional, tal como, por exemplo, a verificação de determinados objetos em uma cena.

1.5 Organização

Este trabalho está organizada como se segue:

O presente capítulo — Introdução — apresenta uma visão geral e sucinta da pesquisa realizada, de seus objetivos, e do desenvolvimento do trabalho; O Capítulo 2 descreve a natureza do problema, sua importância, os métodos atuais de tratamento do mesmo e suas limitações; O capítulo 3 apresenta uma breve revisão bibliográfica sobre redes neurais artificiais; Em seguida, capítulo 4 aborda alguns conceitos e técnicas de processamento de imagem empregados nesse estudo; O capítulo 5 apresenta a especificação de um sistema de reconhecimento de formas através redes neurais, empregando os descritores de Fourier obtidos a partir do contorno do objeto, e resultados do método proposto; Finalmente, no capítulo 6, apresenta-se o sistema de vigilância desenvolvido para detecção de intrusão em ambientes monitorados. O documento termina, no capítulo 7, com conclusões sobre este trabalho e sugestões a serem abordadas em trabalhos futuros.

Capítulo 2

Natureza do Problema

2.1 Introdução

Apesar da visão parecer algo simples, pois ocorre naturalmente, seu funcionamento é bastante complexo. E intimamente relacionado ao processo de percepção visual está o de reconhecimento, seja de seres vivos, objetos, símbolos ou cores. O processo de reconhecimento de formas requer a representação das informações captadas do meio através de características que sejam capazes de diferenciar objetos de naturezas distintas.

O processo que realizamos para conseguir reconhecer um objeto é conhecido parcialmente, pois não há explicações de como armazenamos as informações percebidas. Por exemplo, como são armazenadas as informações referentes a uma bola em nosso cérebro? Se for apresentada uma bola de cor ou tamanho diferente também reconhecemos como sendo uma bola. Imagine então que informações extraímos e armazenamos sobre outra pessoa, a qual pode ser reconhecida por seu rosto, às vezes pela sua aparência física, voz ou até mesmo pelo jeito de andar.

Ao tratar esse assunto no campo da visão computacional tem-se as seguintes perguntas: que modelo ou estruturas de dados são necessárias para representar os objetos pertinentes ao problema? Qual estratégia deve ser utilizada para que uma máquina consiga reconhecer determinados objetos [25]?

Dentro da área de reconhecimento de objetos, visão computacional, este trabalho enfoca uma possível solução para um sistema de vigilância inteligente, que consiga identificar se o objeto em movimento no ambiente monitorado é uma pessoa ou não.

A seguir são apresentados, de forma sucinta, os dispositivos de um sistema de vigilância, os processos de reconhecimento, e o relacionamento entre o reconhecimento de objetos e suas classes.

2.2 Sistemas de Vigilância

Alguns sistemas de vigilância permitem delimitar uma área da cena, e disparam um alarme caso alguns *pixels* daquela região sofram alteração. De forma similar funcionam os

sistemas de segurança com detetores de movimento, que empregam infravermelho ou microondas, e acionam o alarme quando um objeto se move em seu campo de ação.

2.2.1 Detetores de Movimento

Há vários tipos de sensores disponíveis. Sensores mecânicos envolvem tipicamente um interruptor magnético e são usados em portas e janelas. Quando a porta ou janela é fechada, o ímã segura o interruptor em uma posição; quando a porta ou janela é aberta, o ímã muda o interruptor de posição, ativando-o.

Os detetores de movimento são disponíveis em três tipos (ver figura 1). O tipo mais comum é o sensor infravermelho (PIR - *Passive InfraRed*), que é utilizado para monitorar áreas e detectar a entrada de pessoas não autorizadas. Outro tipo de detetor de movimento é o sensor de microondas, que trabalha como um radar. Este sensor envia sinais de rádio e capta os movimentos através dos sinais refletidos pelos objetos.

Ambos tipos de sensores de movimento podem ser enganados por outras fontes de calor ou de som respectivamente, além de serem mais sensíveis a certos tipos de movimentos. Por esse motivo, muitos fabricantes produzem detetores de movimento que envolvem as duas tecnologias e são freqüentemente mais confiáveis que qualquer um individualmente.



Figura 1 - Exemplos de detetores de movimento.

2.2.2 Componentes de um Sistema de Vigilância

Os componentes básicos de um sistema de vigilância são as câmaras e os monitores. Os modelos de câmaras variam muito, bem como seus preços. Há desde as pequenas, do tamanho de um cartão de crédito, até as grandes, de uso profissional. Um tipo de câmara muito prática para uso externo são aquelas dotadas de um detetor de movimento, que podem inclusive emitir um som ou acionar a gravação de uma fita em um videocassete quando alguém se aproximar.

O correto posicionamento das câmaras é fundamental. A maior parte delas têm lentes fixas, e portanto um campo de visão e distância focal também fixos. É preciso estar certo de que a câmara irá cobrir a área que se pretende monitorar. Deve-se evitar fontes de luz no seu campo de visão, embora disponham de função "auto-iris", pois causam zonas escuras que prejudicam muito a qualidade da imagem.

Quanto aos monitores, existem vários tipos de monitores dedicados que funcionam apenas com as imagens do circuito fechado. No entanto, em residências é cada vez mais recomendável fazer uma integração entre o circuito fechado de televisão (CFTV) e o sistema de vídeo da casa (ou seja, TV a cabo, satélite ou antena), tornando possível aos moradores ter a imagem gerada pelo CFTV em qualquer um dos televisores da casa, num canal especialmente designado para este fim. Desejando um pouco mais de sofisticação, é ainda possível mudar o canal da TV (passando a monitorar a imagem do CFTV) sempre que alguém tocar a campainha da casa ou quando um sensor de presença predeterminado identificar um movimento estranho.

2.2.3 Limitações

Vários dispositivos de segurança patrimonial empregam detetores de movimento que disparam um alarme quando há movimento de um objeto no recinto. Porém, há ocasiões em que o alarme é ativado e quando faz-se a verificação do local percebe-se que o alarme disparou devido a um cachorro, gato, pássaro ou outro animal qualquer. Há alguns tipos desses dispositivos, para uso em interiores, que são acionados somente se o volume do objeto em movimento for superior a um determinado limite. Entretanto, o problema maior está em conseguir uma solução para áreas externas, tal como pátios, estacionamentos e etc. Como fazer para que os dispositivos somente acionem o alarme quando determinados objetos moverem-se na sua área de ação?

2.3 Reconhecimento de Objetos

A tarefa de reconhecimento implica na ativação de outros processos. O primeiro dentre eles é o acesso à memória onde a representação do objeto está armazenada. Representações apropriadas ou modelos são importantes itens de pesquisa em visão computacional. O segundo deve ser uma combinação de processos, onde observações são coletadas, normalizadas, e confrontadas com as representações armazenadas para efetuar o

reconhecimento. Em implementações computacionais, os modelos são usualmente representados como estruturas de dados [25].

2.4 Classes de Objetos

Em geral, os seres humanos formam classes a partir de algumas características do objeto que está sendo analisado, para futuramente poder identificar objetos semelhantes ao que foi analisado. Então, se for apresentado a uma pessoa que conheça veículos automotores um novo modelo de automóvel e de motocicleta, esta saberá identificar o automóvel e a motocicleta, embora esteja vendo ambos modelos pela primeira vez. Isto ocorre porque essa pessoa já possui formada em seu cérebro uma classe referente a automóvel e outra para motocicleta. O mesmo ocorre quando visualizamos os vários objetos existentes no ambiente em que vivemos, às vezes não sabemos indicar o modelo específico, mas conseguimos identificá-los devido ao conhecimento de sua classe.

O primeiro passo para uma pessoa identificar um objeto é conseguir distingui-lo dos demais, focalizá-lo. Em visão computacional ocorre processo semelhante, primeiro realiza-se a aquisição da imagem e posteriormente processos para conseguir extrair da imagem a região correspondente ao objeto que deseja-se classificar. A figura 2 exemplifica a obtenção de regiões referentes aos objetos em movimento em algumas cenas.

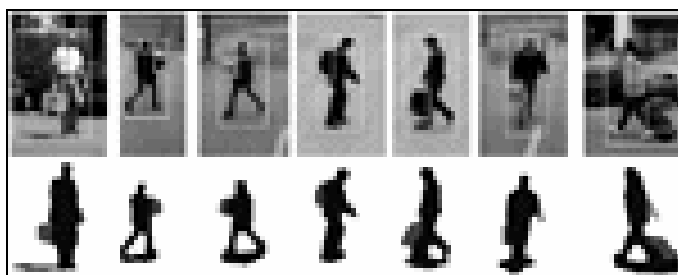


Figura 2 - Exemplos de extração da região correspondente aos objetos em movimento.

Após definido o objeto, o próximo passo é tentar classificá-lo. Vários pesquisadores criam modelos específicos para cada objeto a fim de identificá-los, entretanto neste trabalho optou-se em utilizar redes neurais artificiais para realizar esta tarefa, devido as suas características, que estão descritas no capítulo 3.

Capítulo 3

Redes Neurais Artificiais: Uma Breve Revisão Teórica

3.1 Introdução

As pesquisas neurológicas sobre o funcionamento do cérebro humano e suas características, possibilitaram que pesquisadores pudessem desenvolver um modelo que emulasse algumas de suas atividades, o qual convencionou-se chamar de “*Rede Neural Artificial*” (RNA).

Sendo o cérebro humano o centro do sistema nervoso composto por aproximadamente 10^{11} neurônios (um fator de 10 é razoável como expectativa de erro) [5], que recebe continuamente informação, percebe-a e toma decisões apropriadas. Buscou-se da mesma forma representá-lo como uma rede finita de elementos ativos capazes de interagirem entre si para compor o resultado do processamento das informações.

Desta forma, com inspiração no comportamento dos neurônios biológicos e nos sistemas nervosos, construiu-se as Redes Neurais Artificiais interligando “Neurônios Artificiais” (ver Figura 3). Devido a esta estrutura, as RNA possuem como forte característica o processamento de informação distribuído e paralelo, o que permite alta performance das suas implementações em componentes de hardware.

Buscando simular o comportamento do cérebro humano, as RNAs conseguiram desta forma adquirir algumas das características típicas deste, como a capacidade de generalização, aprendizado e de adaptar-se ao ambiente onde se encontra inserida. Diz-se então que a RNA pode aprender (*learn*). Para tal, existe uma grande variedade de algoritmos de treinamento, todos com seus pontos fortes e fracos [2][4][7][31][36].

Uma vez treinada (ensinada), uma resposta da RNA pode ser insensível a pequenas variações na sua entrada. Esta característica é essencial para o reconhecimento de padrões no mundo real, devido a ruídos ou distorções de padrões (imperfeições) que freqüentemente ocorrem. É importante notar que os resultados das RNAs são obtidos a partir de sua estrutura e não pelo uso de cognição humana embutida em alguma forma de programa de computador.

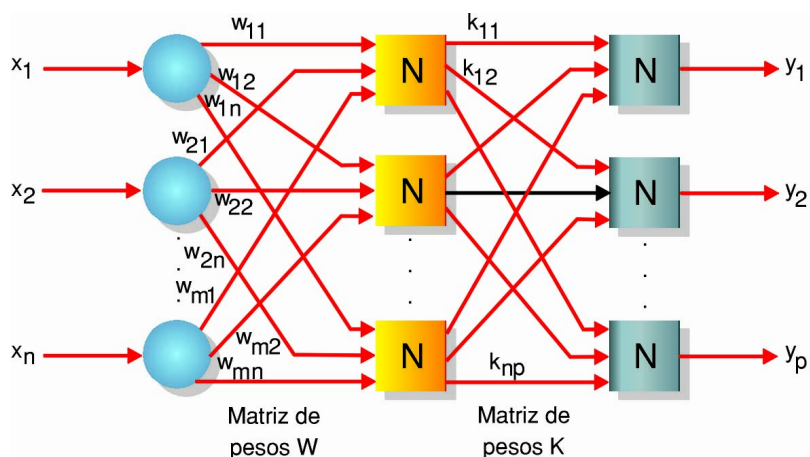


Figura 3 - Estrutura de uma Rede Neural Feedforward.

Devido às suas características de aprendizado e adaptação, as RNAs vêm sendo empregadas nos mais variados campos do conhecimento, principalmente em sistemas onde não se conhece uma equação matemática capaz de relacionar adequadamente a entrada com a saída. Por exemplo, em aplicações como: controle de processos, reconhecimento de padrões, controle de qualidade, análise preditiva do comportamento das bolsas de valores, identificação de objetos, processamento de imagens, entre outras.

As RNAs se destacam entre as demais técnicas de processamento de imagem, devido à sua capacidade de generalização, imunidade a imperfeições e ruídos, capacidade de aprender a reconhecer novas imagens e velocidade de processamento [31].

3.2 Neurônio Artificial

As RNAs são compostas por neurônios artificiais, Figura 4, que buscam emular o funcionamento de um neurônio biológico. O primeiro modelo baseado no comportamento do neurônio biológico foi desenvolvido por McCulloch e Pitts em 1943 [5][7][31]. Nesse modelo, a saída de um neurônio assumia o valor 1, se a soma ponderada de suas entradas não fosse negativa, e 0 caso contrário. Esta definição descreve a *propriedade tudo-ou-nada* do modelo McCulloch-Pitts [12].

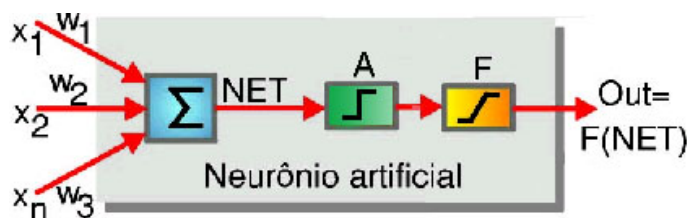


Figura 4 - Modelo do Neurônio Artificial.

O funcionamento deste neurônio é similar ao neurônio humano. Este recebe várias entradas possuindo um certo “*peso*”, que representa a força sináptica presente nas conexões com os outros neurônios, e combina-as por uma soma ponderada, produzindo a entrada efetiva do neurônio, também denominada de “*net*” do neurônio. Após a determinação do *net*, o valor da ativação do neurônio é atualizado através da função de ativação *A* e por fim, o resultado do processamento do neurônio é gerado pela função de saída *F(net)*.

Se a função de ativação *A* for considerada como constante, tem-se neurônios que não possuem “memória”, ou seja, o estado atual é igual aos estados anteriores e portanto o neurônio é conhecido como “neurônio estático”. Caso a função não seja constante, tem-se o conhecido como “neurônio dinâmico”, no qual o estado futuro depende do estado atual.

Este é o modelo básico amplamente adotado para representar o comportamento de primeira ordem do neurônio biológico. As diferenças adotadas dizem respeito à função de saída (ou transferência) do neurônio que pode ser de vários tipos. A seguir apresenta-se alguns exemplos comuns de função de saída:

$$F(net_i) = \frac{1}{1 + e^{-net_i}} \quad \text{Equação 1 – Função sigmóide}$$

$$F(net_i) = \frac{1 - e^{-net_i}}{1 + e^{-net_i}} \quad \text{Equação 2 - Função Tangente - Hiperbólica}$$

$$F(net_i) = a \cdot net_i \quad (a \text{ é um escalar constante}) \quad \text{Equação 3 - Função Linear}$$

$$F(net_i) = \begin{cases} 1, & \text{para } (net_i > 0) \\ 0, & \text{para } (net_i \leq 0) \end{cases} \quad \text{Equação 4 - Função Binária}$$

Cada função de saída diferente gera uma RNA com características distintas, o que em alguns casos pode vir a ser uma qualidade, atributo importante, pois permite que a rede adapte-se melhor para solucionar um dado problema.

Usualmente todos os neurônios de uma RNA possuem a mesma função de saída,. Entretanto, dependendo do problema, algumas vezes a camada de saída da rede tem a sua própria função [30].

3.3 Topologia das Redes Neurais

Conforme mencionado, as RNAs são compostas de inúmeros neurônios que seguem o modelo apresentado anteriormente. Dependendo da forma como os neurônios de uma rede se conectam, esta é classificada como pertencendo a uma topologia. Entre as mais comuns encontram-se [5]:

- Redes Diretas (“*Feedforward*”), cujo grafo não contém ciclo;
- Redes com Ciclos (“*Feedback*”, ou retroalimentação), cujo grafo de conectividade contém ao menos um ciclo. Quando além disto envolvem neurônios dinâmicos, contendo um retardo, são chamadas de *recorrentes*;
- Redes Simétricas, cuja matriz de conectividade é uma matriz simétrica. Trata-se de um caso particular das redes com ciclos.

A configuração mais conhecida, trata-se da *Rede Multicamadas Feedforward* [4][24], também conhecida como *Multi Layer Perceptron* (MLP), onde cada camada possui neurônios de um mesmo tipo (ver Figura 5).

Dentre os diversos modelos de redes neurais existentes, optou-se pela rede Multicamadas *Feedforward* pelos seguintes motivos [29]:

- Bem compreendida, com muitas aplicações de sucesso em reconhecimento de padrões, classificação, mapeamento e filtragem de sinal, entre outras;
- Operação rápida;
- Boa em formar representações internas das características dos dados de entrada.

3.4 Redes Neurais Multicamadas *Feedforward*

Uma Rede Neural Multicamada *Feedforward* é uma rede neural direta, composta por múltiplas camadas de neurônios, organizados em uma camada de entrada, uma ou várias

camadas intermediárias (também denominada de camada escondida ou oculta), e uma camada de saída que fornece o resultado da rede. O sinal de entrada se propaga através da rede camada por camada. Os neurônios de cada camada estão conectados aos neurônios da camada seguinte através de conexões sinápticas, que possuem um peso associado, chamado de *peso sináptico*. O peso sináptico é na verdade o local da rede onde o conhecimento¹ adquirido é armazenado [12]. A figura 8 representa um exemplo deste tipo de rede, com 3 camadas.

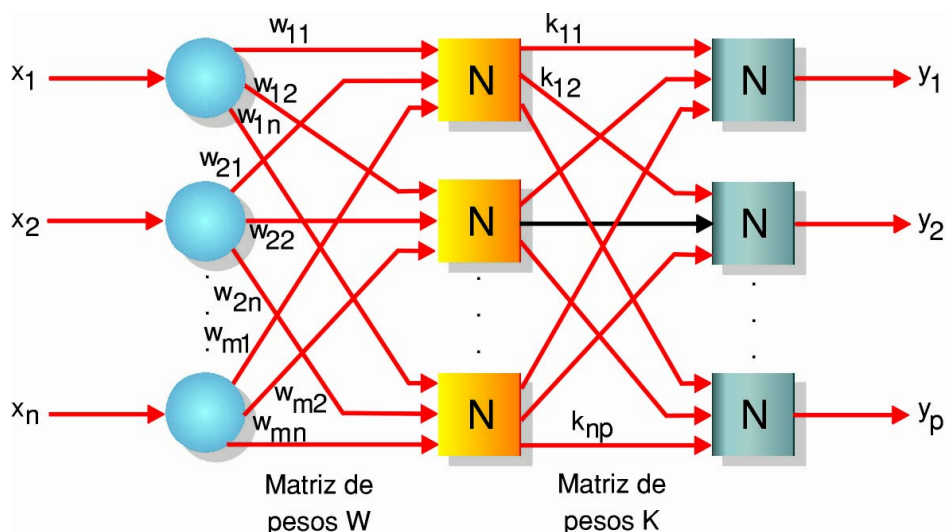


Figura 5 - Rede Neural Artificial Feedforward com uma camada interna e uma de saída.

A primeira camada não efetua nenhum processamento, somente repassa os dados recebidos na entrada da rede, sinais de excitações, para os neurônios da segunda camada. Cada neurônio da segunda camada recebe todos os dados de entrada da Rede Neural Artificial (RNA), realiza a soma das entradas ponderadas pelos valores sinápticos e calcula a saída de acordo com a sua função de saída estipulada (ver equações 1, 2, 3 e 4). Este resultado é então repassado à última camada que, de forma análoga à segunda camada, calcula a saída da rede.

Cada neurônio i na rede é uma simples unidade de processamento que calcula a sua saída s_i , de acordo com as suas excitações de entrada, representada por net_j :

$$net_j = \sum_{j \in p(i)} s_j w_{ij} + \theta_i$$

Equação 5

¹ Conhecimento se refere à utilização da informação armazenada ou a modelos utilizados por uma pessoa ou máquina para interpretar, prever e responder apropriadamente ao mundo exterior

onde, $p(i)$ define o conjunto dos neurônios da camada anterior i , w_{ij} corresponde ao peso da conexão (peso sináptico) entre o neurônio i e o neurônio j , e θ_j contém o valor do bias. Para simplificar, considera-se θ_j como sendo o peso que relaciona o neurônio j com um neurônio “bias” cuja a saída é sempre igual a 1. Desta forma, pode-se proceder manipulando o bias como sendo um peso qualquer, sem perda de generalidade e simplificando a sua análise.

A resposta ao estímulo do neurônio i , (s_i) é calculada a partir de net_i através de uma função de saída. A título de exemplificação e sem perda de generalidade, empregarse-á como função de saída uma das funções unipolares mais utilizadas, a função sigmóide:

$$s_i = F(net_i) = \frac{1}{1 + e^{-net_i}} \quad \text{Equação 6}$$

Esta função tem uma característica muito importante, que é possuir uma derivada fácil de ser calculada, que em geral é empregada nos algoritmos de aprendizado da rede:

$$\frac{\partial s_i}{\partial net_i} = F'(net_i) = s_i \cdot (1 - s_i) \quad \text{Equação 7}$$

Para obter a saída da rede, basta efetuar o cálculo da função de saída s_i para cada neurônio da rede, começando a partir da primeira camada interna, propagando os resultados pela rede até a última camada. O resultado dos neurônios da camada de saída indicará o resultado da rede.

3.4.1 Aprendizado Supervisionado

No treinamento supervisionado, o objetivo é ajustar os valores dos pesos da rede de forma que esta forneça um mapeamento (relação) da entrada na sua saída, ou seja, para um dado vetor de entrada conhecido a rede deverá fornecer a saída desejada. O mapeamento é dado por um conjunto de exemplos P , que relacionam a entrada com a saída da rede, sendo a capacidade de efetuar este relacionamento que a rede deverá aprender [2][4][7][31][36].

Cada par de vetores no conjunto P , consiste de um vetor de entrada x e a correspondente saída desejada t . Após o treinamento, quando uma vetor igual a x for

inserido como entrada da rede, está deverá fornecer como saída um vetor igual a t . A diferença entre o vetor desejado t e o vetor real s , em outras palavras a qualidade do mapeamento da rede, é representado pela seguinte função custo ou energia:

$$E = \frac{1}{2} \sum_{p \in P} \sum_n (t^2 - s_n^2)^2 \quad \text{Equação 8 - Equação do Erro da Rede}$$

onde n é o número de neurônios na camada de saída da rede.

Resolver o problema do algoritmo de treinamento, é equivalente a obter um mínimo global para E [4], pois este representa o quanto a rede é capaz de fornecer a saída desejada. Os pesos sinápticos da rede são modificados no sentido de se obter o mínimo global de E , ou seja, modifica-se os pesos na direção $d(t)$ na procura do mínimo global da superfície de erro formada por E :

$$\begin{aligned} \Delta w(t) &= \varepsilon \cdot d(t) \\ w(t+1) &= w(t) + \Delta w(t) \end{aligned} \quad \text{Equação 9}$$

onde o parâmetro de treinamento ε indica o tamanho do passo de treinamento.

A fim de determinar a direção do ajuste dos pesos, a direção de procura do mínimo global, utiliza-se comumente a informação fornecida pela primeira derivada de E , chamado o método do gradiente.

$$\nabla E = \frac{\partial E}{\partial w} \quad \text{Equação 10}$$

Considerando que se deseja encontrar o mínimo global existente na superfície de erro fornecida por E , utiliza-se a direção oposta do gradiente de E , a fim de obter a direção do mínimo global.

O algoritmo de treinamento *backpropagation*, apresentado a seguir, utiliza a idéia de retropropagar o gradiente do erro da camada de saída para as camadas internas, a fim de corrigir os pesos.

3.4.2 O Algoritmo *Backpropagation*

O algoritmo *Backpropagation* foi um dos primeiros algoritmos de treinamentos desenvolvidos, em 1974 por Werbos, e continua sendo um dos mais aplicados devido sua popularização em 1986 através do trabalho desenvolvido por Rumelhart, Hinton e Williams. Utiliza o mesmo princípio da Regra Delta, a minimização de uma função custo,

no caso, a soma dos erros quadráticos sobre um conjunto de treinamento, utilizando a técnica de busca do gradiente descendente. Ou seja, o algoritmo se baseia fundamentalmente no cálculo do gradiente da função erro E , e a partir deste procura localizar o mínimo global da função [2][4][7][31][36].

A idéia básica, utilizada para calcular derivadas parciais $\partial E/\partial w$ para cada peso na rede, é repetidamente aplicar a regra da cadeia:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial s_i} \frac{\partial s_i}{\partial w_{ij}} \quad \text{Equação 11}$$

onde,

$$\frac{\partial s_i}{\partial w_{ij}} = \frac{\partial s_i}{\partial net_i} \frac{\partial net_i}{\partial w_{ij}} = F'(net_i) s_j \quad \text{Equação 12}$$

Para calcular $\partial E/\partial s_i$, ou a influência da saída s_i do neurônio i no erro global E , distingui-se os seguintes dois casos:

a) Se i é um neurônio da camada de saída, então:

$$\frac{\partial E}{\partial s_i} = \frac{1}{2} \frac{\partial (t_i - s_i)^2}{\partial s_i} = -(t_i - s_i) \quad \text{Equação 13}$$

b) Se i não é um neurônio da camada de saída, então o cálculo de $\partial E/\partial s_i$ é um pouco mais complicado. Agora, utiliza-se a idéia de propagar o erro da camada de saída para dentro da rede, de forma a estimar o erro das camadas internas e assim poder corrigir os pesos associados a estas. Novamente, a regra da cadeia é aplicada:

$$\begin{aligned} \frac{\partial E}{\partial s_i} &= \sum_{k \in \text{suc}(i)} \frac{\partial E}{\partial s_k} \frac{\partial s_k}{\partial s_i} \\ \frac{\partial E}{\partial s_i} &= \sum_{k \in \text{suc}(i)} \frac{\partial E}{\partial s_k} \frac{\partial s_k}{\partial net_k} \frac{\partial net_k}{\partial s_i} = \sum_{k \in \text{suc}(i)} \frac{\partial E}{\partial s_k} F'(net_k) w_{ki} \end{aligned} \quad \text{Equação 14}$$

onde $\text{suc}(i)$ indica o conjunto de todos os neurônios k na camada sucessiva (sucessiva significa camada seguinte na direção da camada de saída) onde o neurônio i tem um peso w_{ki} diferente de zero.

A equação 14 assume que é conhecido o valor de $\partial E / \partial s_k$ para os neurônios na camada sucessiva na qual i está conectado. Isto é garantido começando-se o cálculo a partir da camada de saída (equação 11) e então sucessivamente calculando as derivadas das camadas precedentes, aplicando a equação 12.

Em outras palavras, a informação do gradiente é sucessivamente propagado da camada de saída em direção a camada de entrada. Por isso, este algoritmo de treinamento é chamado “*Error Backpropagation*” (Retropropagação do Erro).

3.4.3 Gradiente Descendente

Uma vez que as derivadas são conhecidas, o próximo passo no algoritmo “Backpropagation” é calcular o valor do ajuste dos pesos que deverá ser aplicado. A idéia básica é ajustar os pesos dando um passo na direção do gradiente, em outras palavras a derivada negativa é multiplicada por uma constante, a taxa de aprendizado ε . Esta técnica de minimização é normalmente conhecida como “gradiente descendente” [4][24][30][36]:

$$\Delta w(t) = -\varepsilon \cdot \nabla E(t) \quad \text{Equação 15}$$

ou, para um peso específico:

$$\Delta w_{ij}(t) = -\varepsilon \cdot \frac{\partial E}{\partial w_{ij}}(t) \quad \text{Equação 16}$$

Embora a regra básica do treinamento seja simples, é freqüentemente complicado determinar o valor apropriado para a taxa de aprendizado ε para uma dada aplicação. Uma boa escolha depende da forma da função de erro, a qual claramente muda de acordo com o problema de aprendizado [4][36]. Um valor pequeno da taxa de aprendizado resultará em um tempo muito longo de convergência se a superfície do erro for muito plana, enquanto que uma taxa de aprendizado grande implicará em uma grande oscilação da convergência do algoritmo, impedindo que este alcance o valor mais adequado. Outro aspecto é que este algoritmo pode convergir para um mínimo local da função E , não garantido a obtenção do mínimo global e, portanto, a obtenção da melhor solução de aprendizado.

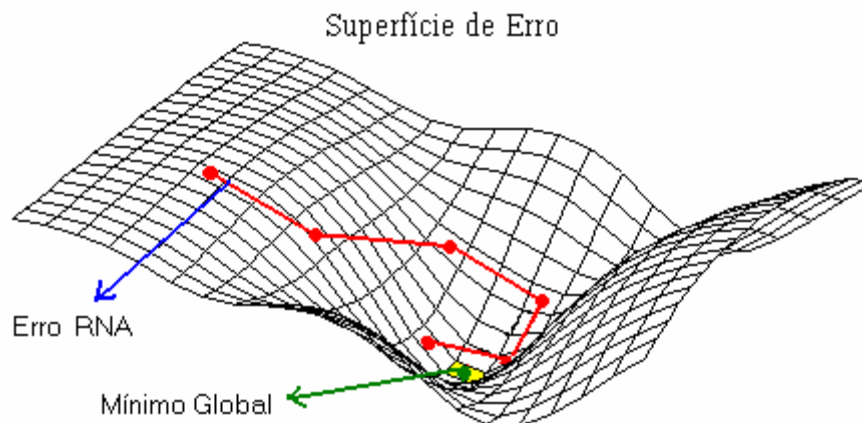


Figura 6 – Problema do gradiente descendente.

Outro problema com o gradiente descendente é que este vai contra a intuição com relação a influência da derivada do erro no ajuste dos pesos. Por exemplo, se a função de erro decresce lentamente indicando que o sistema se encontra numa região quase plana distante do mínimo, a derivada implicará em passos pequenos de ajuste do peso, demorando a convergir para o mínimo. Outro caso é se a função de erro decresce rapidamente indicando a proximidade de um mínimo, a derivada fornecerá passos longos ao ajuste dos pesos, o que implicará na dificuldade de convergir a um ponto específico e desta forma gerando oscilações, conforme exemplificado na Figura 6.

Com o intuito de tornar o aprendizado mais estável e capaz de transpor a estagnação em mínimos locais, introduziu-se um termo de momento à lei de aprendizado [2][4][24][36]:

$$\Delta w_{ij}(t) = -\varepsilon \cdot \frac{\partial E}{\partial w_{ij}}(t) + \mu \cdot \Delta w_{ij}(t-1) \quad \text{Equação 17}$$

O parâmetro do momento μ indica o valor da influência do ajuste dos pesos ocorridos nas iterações anteriores no ajuste dos pesos na iteração atual, ou seja, haverá uma tendência a exercer-se sobre os ajustes subseqüentes, e por isso este termo é chamado de momento. Deve-se notar que embora essa técnica funcione bem na maioria das aplicações, não é um método geral para permitir ganho de estabilidade e ao mesmo tempo velocidade de convergência. Em alguns casos, pode-se até obter melhores resultados sem a utilização do termo de momento. Usualmente, quando utiliza-se o gradiente descendente com momento, deve-se diminuir a taxa de aprendizado ε para evitar que o algoritmo de treinamento torne-se instável.

3.4.4 Aprendendo a cada Lote (*batch, epoch*) ou a cada Iteração (*pattern*)

Basicamente, existem duas maneiras possíveis para calcular e atualizar as modificações nos pesos da rede, dependendo de quando é realizada a atualização destas modificações.

No método de aprendizado a cada iteração (*learning by pattern*) [2][36], o valor dos pesos é atualizado após a apresentação de cada par de vetores exemplos e o cálculo do respectivo gradiente. Este método é conhecido também como aprendizado *on-line* ou estocástico, pois tenta minimizar o erro global através da minimização do erro de cada par de exemplos individualmente e, portanto, a cada iteração do algoritmo. Este método funciona bem especialmente para grandes conjuntos de exemplos contendo informação redundante, ou seja, para um conjunto grande que possua exemplos parecidos ou já conhecidos.

Um método alternativo, conhecido como “aprendizado por lote” (*learning by batch* ou *epoch*) [2][36], primeiramente soma os resultados provenientes do cálculo do gradiente para cada par de vetores do conjunto de exemplos, para depois atualizar os pesos da rede. Cada peso será ajustado no sentido de minimizar o somatório do erro do conjunto de exemplos, em outras palavras a função erro definida (equação 8). Este método é mais eficiente pois a informação gerada pela soma de todos os gradientes contém as informações mais relevantes, mantendo a forma da função de erro global que deseja-se minimizar. Entretanto, necessita-se armazenar um volume maior de dados, requerendo assim um volume maior da memória de computador para implementar este método.

Para utilizar RNA são necessários dados para realizar seu treinamento, a fim de que a rede consiga aprender a reconhecer os padrões de entradas e gerar as respectivas saídas. Neste trabalho lida-se com imagens, e para conseguir extrair informações delas é preciso executar vários procedimentos, que são abordados no capítulo seguinte.

Capítulo 4

Processamento de Imagens

4.1 Introdução

O processamento de imagens digitais é caracterizado por soluções específicas, envolvendo procedimentos que são geralmente expressos em forma algorítmica. Assim, com exceção da aquisição e exibição de imagens, a maioria das funções de processamento de imagens podem ser implementadas em *software* [10]. Desse modo, técnicas que funcionam bem em uma área podem se mostrar totalmente inadequadas em uma outra área.

A seguir são apresentadas as principais características e os resultados das técnicas de processamento de imagens aplicadas no projeto deste sistema de vigilância para extrair informações das imagens.

4.2 Etapas do Processamento de Imagens

Embora constituindo um processo integrado, o processamento de imagens digitais (visão computacional) pode ser compreendido através de seis grandes etapas principais [28]:

- **Aquisição:** dois elementos são necessários para a aquisição de imagens digitais. O primeiro é um dispositivo físico sensível a uma banda do espectro de energia eletromagnética (raios X, ultravioleta, visível ou banda infravermelha, etc) e que produza um sinal elétrico de saída proporcional a um nível de energia percebida. O segundo, chamado digitalizador, converte a saída elétrica de um dispositivo de sensoriamento físico para a forma digital [10]. A qualidade das imagens obtidas são de fundamental importância para as demais etapas do processo.
- **Pré-processamento:** As imagens obtidas na etapa de aquisição podem apresentar ruídos e baixa definição de detalhes. A utilização de técnicas de Processamento Digital de Imagens (PDI) prepara a imagem para as etapas subsequentes, eliminando ruídos, suavizando efeitos indesejáveis e realçando detalhes importantes para a detecção de objetos.
- **Segmentação:** etapa em que a imagem é dividida em regiões que constituem os diversos objetos nela representados. A identificação de um objeto se baseia na detecção

de descontinuidades ou similaridades na imagem, gerando uma representação abstrata de seu contorno ou da região que ocupa.

- **Descrição:** cada objeto identificado no processo de segmentação é analisado para extração de algumas de suas características. A esse conjunto de características denomina-se *padrão*, o qual representará o objeto nas etapas seguintes.
- **Reconhecimento:** nesta etapa, o padrão de cada objeto identificado é comparado às classes padrões já conhecidas com o objetivo de se decidir à qual grupo ele pertence. Metodologias de reconhecimento devem estabelecer um compromisso entre eficiência e confiabilidade, além de possibilitarem revisão do conhecimento através da experiência adquirida.
- **Interpretação:** conforme a natureza do problema tratado, pode ser necessária uma análise de cena, onde os objetos são relacionados entre si, buscando-se uma consciência descritiva do ambiente em que se encontram.

O pré-processamento de uma imagem é um componente muito importante do sistema de visão, pois a partir do mesmo pode-se ter informações sobre a posição e orientação do objeto, eliminar ruídos e salientar detalhes de interesse, bem como gerar os vetores referentes ao contorno do objeto determinado. Esses vetores podem servir como dados de entrada para a RNA, de forma a permitir que esta reconheça o objeto. Por esta razão, a qualidade do sistema de visão depende diretamente da qualidade do pré-processamento da imagem [31].

4.3 Aquisição de Imagem

As imagens digitais são obtidas através do seguinte princípio básico [22]:

1. Inicialmente, o sistema de iluminação (LEDs, estrobos, lâmpadas, luz natural etc.) produz um tipo de luz que incide sobre a cena, especificamente na superfície do objeto e no fundo da cena (*background*);
2. Essa luz é refletida da cena (objeto e fundo) em direção à lente da câmara;
3. A lente refrata a luz para o sensor da câmara, que transforma a imagem recebida em uma imagem eletrônica², onde cada *pixel* representa a intensidade de luz incidida sobre uma fotocélula do sensor da câmara.

² Defini-se *Imagem Eletrônica* como uma imagem representada por um sinal elétrico analógico.

4. Em seguida, a placa de captura (*frame-grabber*) digitaliza a imagem, formando uma matriz numérica, onde cada elemento da matriz recebe um valor compatível com a intensidade de luz do *pixel* correspondente.

Por fim, tem-se então a imagem digitalizada, que é armazenada na memória do computador na forma de uma matriz de dados, onde cada elemento representa um *pixel* da imagem. Esta matriz numérica é, então, processada pelo computador para extrair as informações desejadas da imagem. Cada sistema desenvolvido processa as imagens digitalizadas e extrai as informações necessárias para execução de seus procedimentos. Esta tarefa pode ser simples em alguns casos, porém, geralmente, é uma das tarefas mais complexas de um sistema de visão. Diversos algoritmos estão sendo desenvolvidos para auxiliar a análise de imagens [35].

4.4 Imagem e ROI

Imagem digital, como foi dito anteriormente, é uma matriz numérica onde cada elemento representa a intensidade luminosa em um determinado ponto [23]. ROI (*Region Of Interest*) caracteriza-se como um segmento desta imagem, ou uma sub-imagem. A Figura 7 apresenta um exemplo de ROI. Nota-se que o centro da imagem foi selecionado em uma sub-imagem, facilitando a análise desta parte da imagem.

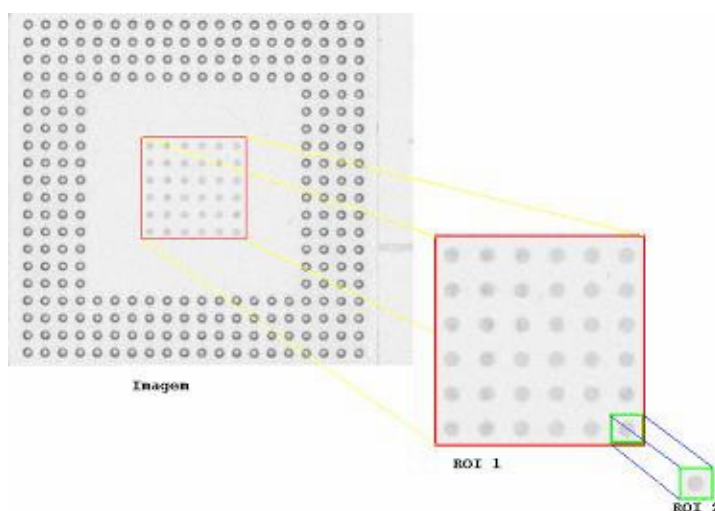


Figura 7 - Exemplo de uma ROI sobre outra ROI [35].

As ROIs servem, justamente, para separar partes da Imagem para serem analisadas segundo algoritmos específicos. Existem dois tipos de ROI [35]:

1. **ROI Normal:** esta copia a região da imagem selecionada para uma nova imagem (nova matriz de dados) com as dimensões da área selecionada, permitindo que operações sejam executadas sobre esta parte da imagem sem afetar a imagem original. Esta também é denominada como **ROI Primária** ou **ROI Pai**;
2. **ROI Virtual:** esta representa uma determinada região da imagem sem realizar uma cópia (utiliza a mesma matriz de dados que a imagem original), ou seja, operações realizadas sobre esta ROI alteram o conteúdo da imagem de referência na região que a ROI representa. Esta também é denominada como **ROI Secundária** ou **ROI Filha**.

Outra característica da ROI é a sua hierarquia múltipla, por exemplo, cria-se uma ROI referenciando-se uma região da imagem. Entretanto, pode-se criar uma segunda ROI referenciando-se uma região de uma ROI pré-existente (ver Figura 7). Isto significa que pode-se referenciar ROI em relação umas com as outras e em relação com imagens.

4.5 Operações Básicas

Uma das operações mais simples realizadas sobre imagens em sistemas de visão são operações aritméticas e lógicas [16][17][23][40] entre imagens e entre imagens e um valor constante. Como as imagens digitais são na verdade matrizes numéricas, geralmente com valores inteiros, pode-se aplicar a estas todos os tipos de operações efetuadas com matrizes.

Uma característica que diferencia as operações de imagens das operações com matrizes é a forma como a saturação é implementada. Quando os *pixels* de uma imagem são processados por uma determinada operação, se o valor resultante for maior que a resolução da imagem, o valor será saturado no valor máximo suportado pela imagem. Da mesma forma ocorre se o resultado da operação ultrapassar o limite mínimo, sendo, portanto, saturado para o valor mínimo [35]. Por exemplo, suponha que em uma imagem monocromática com resolução de 8 bits (0 – 255 tons de cinza) seja adicionada o valor 100. Todos os *pixels* que possuírem valores iniciais superiores a 155, terão os seus valores finais (resultado da operação) saturados em 255, pois este é o valor máximo suportado pela resolução da imagem. Esta característica serve para manter a coerência das informações na imagem .

As operações lógicas consideram que um *pixel* possui valor ‘verdadeiro’ se o seu conteúdo for diferente de ‘0’(zero), senão o valor será ‘falso’. O resultado da operação é sempre uma imagem binária, onde os *pixels* com conteúdos verdadeiros assumem o valor máximo (255, em uma imagem em tons de cinza com resolução de 8 bits) e os *pixels* com conteúdos falsos assumem o valor ‘0’.

A seguir, apresentam-se algumas das operações básicas comumente realizadas com imagens[16][17][35]:

- **Cópia:** copiar uma imagem para outra imagem;
- **Soma:** somar os *pixels* de duas imagens com dimensões idênticas, ou *pixels* de uma imagem com uma constante;
- **Subtração:** realiza a subtração dos *pixels* de duas imagens com dimensões idênticas, ou *pixels* de uma imagem com uma constante;
- **Inversão:** este algoritmo inverte uma imagem, ou seja, os *pixels* com cor ‘branca’ passam a conter a cor ‘preta’ e vice-versa (cálculo do complemento). Em uma imagem com 255 tons de cinza, este algoritmo opera da seguinte forma: $\text{valor_do_pixel}(x) = 255 - \text{valor_do_pixel}(x)$; Um algoritmo semelhante é aplicado no caso de imagens coloridas;
- **Multiplicação:** multiplicação de uma imagem por outra ou por um valor constante;
- **Operação Binária AND:** realiza-se uma operação binária AND entre os *pixels* de duas imagens ou entre os *pixels* de uma imagem e uma constante;
- **Operação Binária OR:** realiza-se uma operação binária OR entre os *pixels* de duas imagens ou entre os *pixels* de uma imagem e uma constante;
- **Operação Binária XOR:** realiza-se uma operação binária XOR (“ou exclusivo”) entre os *pixels* de duas imagens ou entre os *pixels* de uma imagem e uma constante;
- **Operação Binária NOT:** operação idêntica a inversão da imagem, ou seja, calcula o complemento desta;
- **Mínimo:** retorna o valor mínimo presente em uma imagem;
- **Máximo:** retorna o valor máximo presente em uma imagem;
- **Soma dos Pixels:** soma o valor de todos os *pixels* presentes na imagem;
- **Num. dos Pixels Nulos:** calcula quantos *pixels* na imagem possuem o valor ‘0’;

- **Num. dos *Pixels* Não Nulos:** calcula quantos *pixels* na imagem possuem os valores diferentes de '0';
- **Média:** calcula a média dos valores de todos os *pixels* presentes na imagem.
- **Desvio padrão:** calcula o desvio padrão dos valores de todos os *pixels* presentes na imagem;
- **Norma:** calcula a norma dos valores dos *pixels* presentes na imagem, tanto a norma simples com a norma da diferença em relação a outro valor (constante ou outra imagem);
- **Operação Lógica AND:** realiza-se uma operação lógica AND entre os *pixels* de duas imagens ou entre os *pixels* de uma imagem e uma constante;
- **Operação Lógica OR:** realiza-se uma operação lógica OR entre os *pixels* de duas imagens ou entre os *pixels* de uma imagem e uma constante;
- **Operação Lógica XOR:** realiza-se uma operação lógica XOR entre os *pixels* de duas imagens ou entre os *pixels* de uma imagem e uma constante;
- **Igualdade:** operação lógica para dizer se duas imagens são idênticas ou se uma imagem é idêntica ao valor de uma constante (todos os *pixels* são idênticos a esta constante);
- **Maior Que:** indica se todos os *pixels* de uma imagem são maiores que os *pixels* de outra imagem ou maiores que uma constante;
- **Menor Que:** indica se todos os *pixels* de uma imagem são menores que os *pixels* de outra imagem ou menores que uma constante;

4.6 Threshold

Threshold [16][19][23] é uma técnica desenvolvida para aumentar o contraste de um objeto em relação ao fundo da imagem, ou seja, ressaltar o objeto em relação ao fundo.

Threshold consiste em analisar a imagem, definindo um valor mínimo ou máximo de intensidade luminosa para que um *pixel* pertença a um objeto. Valores fora deste patamar, serão apagados da imagem e valores dentro do intervalo serão ressaltados ou mantidos.

$$I'(i, j) = \begin{cases} 255; & \text{if}(I(i, j) > \text{Threshold}) \\ 0; & \text{else} \end{cases}$$

Equação 18

Normalmente, utiliza-se como referência um valor mínimo de intensidade (equação 1). *Pixels* com intensidade menor que este valor, recebem o valor nulo ('0') de intensidade. *Pixels* com intensidade acima deste valor, recebem o valor máximo ('255', em uma imagem de 8 bits de resolução) de intensidade. A Figura 8 mostra um exemplo de uma imagem analisada com o algoritmo de *threshold*, onde o valor de intensidade luminosa utilizado como referência é igual a '92'.

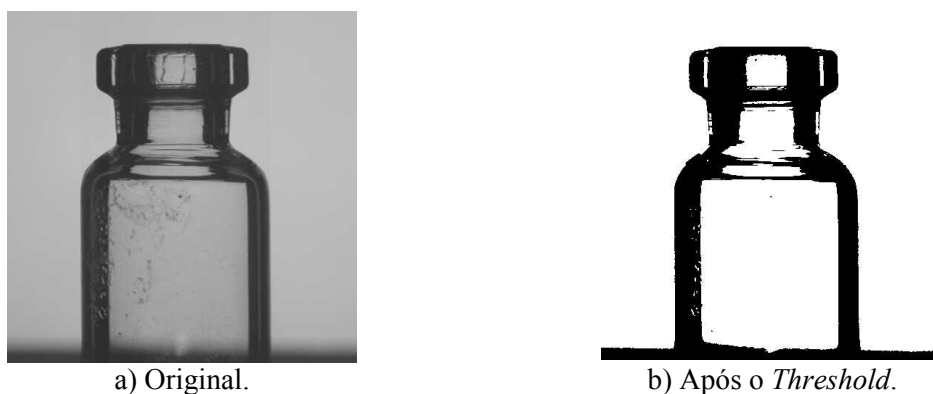


Figura 8 - Exemplo de uma imagem após a operação de *Threshold*.

4.6.1 *Threshold* Adaptativo

Threshold adaptativo [19] nada mais é que um tipo especial de *threshold*, onde os patamares mínimo e máximo de intensidade luminosa são definidos a partir de dados estatísticos da imagem. Nesta técnica, procura-se localizar parâmetros de *threshold* que otimizem o processo de separação dos *pixels* pertencentes ao fundo da imagem em relação aos *pixels* do objeto.

4.7 Filtros (Convoluções)

Filtros [16][19][23][40] são transformações aplicadas na imagem considerando o valor de um determinado *pixel* e de seus vizinhos. Os filtros produzem diferentes tipos de efeitos sobre a imagem, alguns deles sendo listados a seguir:

- atenuar ruídos na imagem;
- suavizar imagens;
- segmentar regiões com valores de intensidade parecidos;
- detectar contornos na imagem.

Os filtros são, geralmente, implementados através de convoluções sobre a imagem. Convolução [16][19][23][40] é uma operação sobre imagens que utiliza uma máscara

(núcleo de convolução ou *kernel*) para processar um determinado *pixel*, considerando o valor deste e de seus vizinhos na imagem.

A Figura 9 apresenta uma ilustração da técnica de convolução. A Figura 9a mostra uma matriz que representa uma imagem monocromática 5x8, com 8 bits de resolução. Deseja-se aplicar uma convolução nesta imagem utilizando uma máscara de tamanho 3x3. Desta forma, cria-se uma matriz 3x3 contendo os valores a serem utilizados pela máscara de convolução (ver Figura 9c). A imagem resultante é calculada a partir da equação:

$$I'(x,y) = \frac{\sum_{i=0}^3 \sum_{j=0}^3 (a(i,j) \cdot I(x+i-1, y+j-1))}{\sum_{i=0}^3 \sum_{j=0}^3 a(i,j)}; \quad \text{Equação 19}$$

Por exemplo, o valor do *pixel* (2,4) após esta convolução, será:

$$I'(2,4) = \frac{50 + 150 + 100 + 200 + 5 \cdot 255 + 150 + 150 + 100 + 100}{13} = 175; \quad \text{Equação 20}$$

Assim, calcula-se esta equação para todos os elementos da matriz da imagem, gerando a imagem resultante.

						a20	
						a21	
						a22	
7	10	10	25	10	10		
	50	50	50	50	10		
	10	50	150	100	25	a10	
	10	200	255	150	25	a11	
	50	150	100	100	20	a12	
	10	100	50	50	10	a00	
	10	10	10	10	10	a01	
0	0				4	a02	

1	1	1
1	5	1
1	1	1

a)Exemplo de imagem. b)Núcleo de Convolução. c)Exemplo de Convolução 3x3.

Figura 9 - Exemplificação da Convolução.

A partir do princípio da convolução, implementa-se uma série de filtros conhecidos apenas definindo-se a matriz de convolução apropriada.

Os seguintes filtros são aplicados neste sistema de visão:

- **Sobel:** este filtro aplica um operador de derivada na imagem, ressaltando os contornos. Entretanto, ruídos presentes na imagem são amplificados;

- **Laplace:** este filtro também aplica um operador de derivada na imagem, ressaltando os seus contornos. Entretanto, ruídos presentes na imagem são amplificados;
- **Min:** este é um filtro não-linear. Ele substitui o valor de determinado *pixel* pelo valor do *pixel* com menor intensidade na sua vizinhança de convolução;
- **Max:** este é um filtro não-linear. Ele substitui o valor de determinado *pixel* pelo valor do *pixel* com maior intensidade na sua vizinhança de convolução;
- **Média:** este é um filtro não-linear. Ele substitui o valor de determinado *pixel* pelo valor médio contido nos *pixels* da sua vizinhança de convolução
- **Gauss:** este é um filtro muito utilizado para suavizar imagens e atenuar ruídos, utilizando uma curva de Gauss para calcular os ganhos da matriz de convolução.

4.8 Operadores Morfológicos

Define-se Morfologia como a forma e a estrutura de um objeto, ou ainda, os arranjos e inter-relacionamentos entre as partes de um objeto. Morfologia digital é uma estratégia para descrever e analisar a forma da representação digital de um objeto.

Operadores morfológicos [16][19][23][39][40] realizam um conjunto de operações a partir da sobreposição (ou subtração) da forma de uma imagem pela forma de um elemento estruturante (outra imagem). O elemento estruturante está para a morfologia assim como o núcleo de convolução (*kernel*) está para teoria de filtragem linear.

A Figura 10 apresenta os dois tipos principais de operações morfológicas: dilatação e erosão. Tanto o conjunto de *pixels* em 'A' quanto o conjunto de *pixels* em 'B' podem ser considerados como sendo uma imagem, entretanto 'A' é usualmente considerado com sendo a imagem a ser analisada e 'B' como sendo o elemento estruturante.

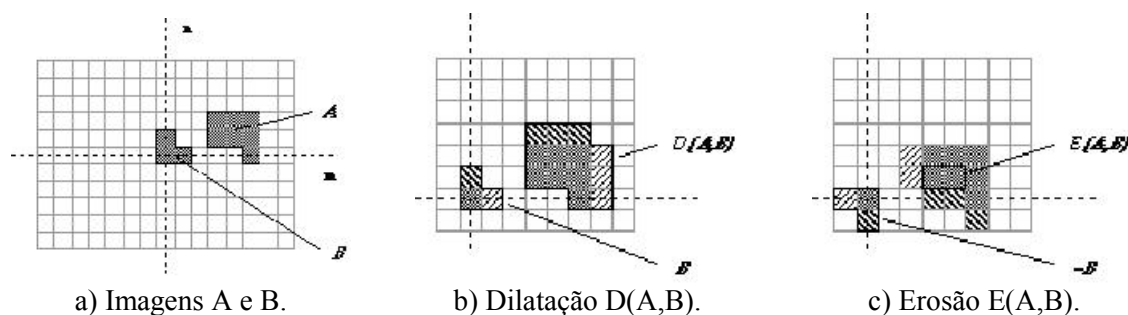


Figura 10 - Exemplo dos dois tipos principais operações morfológicas.

A operação morfológica de dilatação, em geral, faz com que o objeto dilate, enquanto que a erosão faz com que o objeto encolha. O modo e a proporção (magnitude)

da expansão ou redução da imagem dependem, necessariamente, do elemento estruturante ‘B’. Aplicar uma dilatação ou erosão numa imagem sem especificar um elemento estruturante não causará nenhum efeito nesta. Os dois elementos estruturantes mais comuns (olhando num plano cartesiano) são os conjuntos 4-conexões e 8-conexões, ‘N4’ e ‘N8’ (ver Figura 11).



Figura 11 - Os Elementos Estruturantes padrão N4 e N8, respectivamente.

- Pode-se, ainda, combinar a dilatação com a erosão para construir operadores mais importantes:
- **Abertura (*Open*):** definida como uma erosão seguida de uma dilatação. Esta operação ressalta furos internos do objeto e tende a separar elementos na extremidade do objeto (ver Figura 12a);
- **Fechamento (*Close*):** definida como uma dilatação seguida de uma erosão. Esta operação tende a fechar furos internos no objeto e a conectar elementos na extremidade do objeto (ver Figura 12b);
- **Acerto e Erro (*Hit and Miss*):** aplica os operadores de abertura e fechamento para ressaltar os contornos do objeto (ver Figura 12d);
- **Gradiente:** aplica os operadores de abertura e fechamento de forma a calcular o gradiente da imagem;
- **Top-Hat:** aplica os operadores de abertura e fechamento para ressaltar os contornos do objeto, principalmente no caso de objetos claros com fundo escuro;
- **Black-Hat:** aplica os operadores de abertura e fechamento para ressaltar os contornos do objeto, principalmente no caso de objetos escuros com fundo claro.

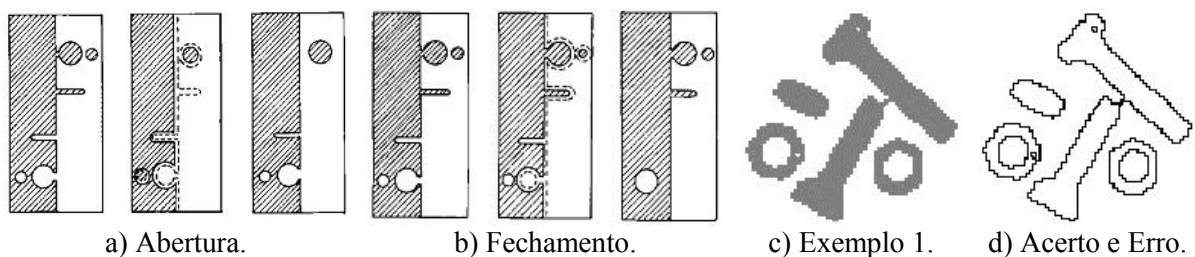


Figura 12 - Exemplo de aplicações de morfologia.

4.9 Contornos

Em uma imagem bidimensional, pode-se descrever um objeto a partir da sua coloração ou textura e a partir da sua forma. A forma de um objeto pode ser perfeitamente representada por um conjunto de contornos que delimitam o objeto externa e internamente. Seguindo este raciocínio, conclui-se que a detecção e análise de contornos são ferramentas fundamentais para a correta interpretação de uma imagem [23][31][40].

Contornos podem ser ressaltados nas imagens através de diferentes técnicas, ou mesmo, da combinação destas técnicas:

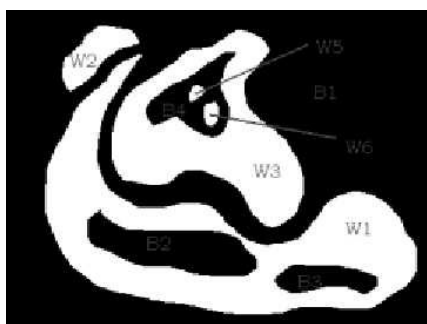
- Filtragem (Sobel, Laplace, Canny, *Hipass*, etc.);
- *Threshold*;
- Morfologia (*Hit and Miss*, *Top-Hat*, *Black-Hat* e Gradiente);
- Segmentação;

Uma vez que a imagem foi binarizada (ver Figura 13), mantendo-se somente os pontos pertencentes ao objeto com valores acima de zero ('0'), basta aplicar um algoritmo de varredura simples [1] para detectar todos os contornos existentes e armazená-los em vetores de pontos, para posterior análise. Um método de varredura de contornos pode ser obtido em [23].

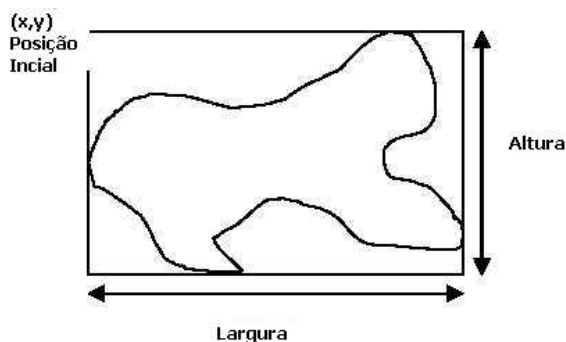
A partir destes vetores que definem os contornos existentes na imagem, pode-se calcular diversos parâmetros, como os abaixo relacionados:

- medição de dimensões (largura, altura, raio máximo, raio mínimo etc.);
- área do contorno;
- momento;
- centróide;
- orientação;
- cálculo das extremidades superior, inferior, lateral esquerda e lateral direita (ver Figura 13).

Através dos contornos, pode-se também compactar uma imagem, representando-a apenas pelos seus contornos.



a) Detecção de Contornos.



b) Cálculo das Extremidades do Contorno.

Figura 13 - Exemplos de análise de contornos.

Diversos algoritmos de detecção de padrões (*pattern matching*) [23][31][40], utilizam a comparação dos contornos dos objetos para definir se um contorno é igual a outro. Anteriormente, comparava-se todos os *pixels* que formavam o objeto, requerendo um alto tempo de processamento para executar essa operação.

Tendo sido realizado uma revisão sobre processamento de imagens, e anteriormente sobre RNAs, o próximo capítulo demonstra um método que utiliza o que foi explanado para realizar o reconhecimento de formas a partir de imagens.

Capítulo 5

Reconhecimento de Formas

5.1 Introdução

Neste capítulo, será demonstrado uma técnica de inteligência artificial que possibilita o reconhecimento de objetos a partir de seu contorno (borda), desde que o conjunto de pontos que o formam sejam devidamente tratados. Uma das formas empregadas para transformar esses pontos em entradas para rede neural artificial (RNA) é o emprego da transformada de Fourier, que será abordada no item 5.3.

5.2 Região e Contorno de um Objeto

Em geral, o sistema visual humano consegue identificar facilmente os objetos de uma paisagem ou cena do cotidiano quando a observarmos. Todavia, em algumas ocasiões só é possível fazê-lo caso o objeto se mexa. Isto ocorre principalmente quando o objeto possui características de texturas semelhantes ao local em que se encontra. Este é o princípio da camuflagem, empregado amplamente por animais e insetos.

Vale salientar que em análise de imagens acontece fato similar, na qual uma das formas de identificar um objeto é ter uma imagem com a situação imediatamente anterior, sem a presença do objeto, e outra com a atual.

No caso de um sistema de vigilância, *a priori* necessita-se que este seja capaz de verificar se há algum objeto em movimento na cena que está sendo monitorada, e se houver, conseguir capturar sua forma, como exemplificado na Figura 14, abaixo.

Uma das formas de executar essa tarefa é possuir um detetor de movimento, que pode ser implementado via *software*, e uma imagem do local monitorado sem objetos em movimento. Essa imagem pode ser atualizada em intervalos constantes de tempo ou quando ocorrerem mudança referentes a luminosidade do ambiente, desde que o detetor de movimentos não esteja acionado. Realizando uma operação de subtração entre a imagem atual e a previamente armazenada obtém-se a região correspondente ao objeto em movimento. Sobre a região selecionada, dependendo da aplicação, emprega-se um filtro para evitar que regiões demasiadamente pequenas ou grandes sejam processadas pelas etapas seguintes sem necessidade.



Figura 14 - Exemplo de identificação do contorno de um objeto em movimento.

A partir da região, o objetivo é selecionar o contorno do objeto presente na imagem adquirida e aplicar a *Transformada Discreta de Fourier (DFT)* nos pontos pertencentes ao contorno com o intuito de aproximar este por uma série de Fourier. Posteriormente, no capítulo 3, serão descritas vantagens em utilizar esta transformada.

Para obter-se o contorno do objeto presente na imagem, precisa-se primeiramente ressaltar o objeto em relação ao fundo da imagem. No exemplo a seguir, está sendo considerado o fundo da imagem próximo de zero (preto) e o objeto próximo do valor máximo 31 (branco), devido o emprego de uma câmara que utiliza 5 bits. Pode-se ressaltar o objeto analisando a intensidade de cinza de cada *pixel* (menor unidade que constitui uma imagem digital) e alterando o seu valor para zero caso esta possua uma intensidade menor que um dado valor, por exemplo 25 (utiliza-se aqui a idéia de um filtro passa-alta) [31].

Por exemplo, na figura abaixo (ver Figura 15) tem-se uma imagem digitalizada antes de ressaltar o objeto.

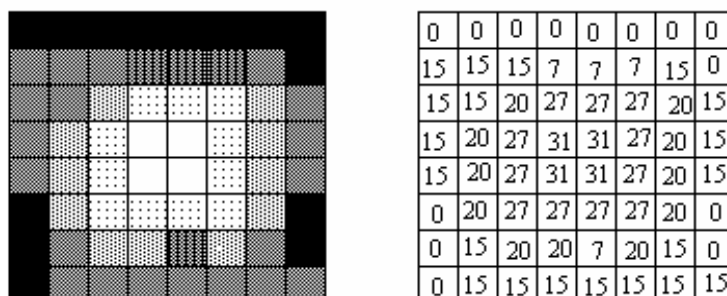


Figura 15 - Imagem antes de ressaltar a objeto.

Alterando-se para zero os valores de intensidade de cinza menores que 25 obter-se-á a imagem a seguir (ver Figura 16).

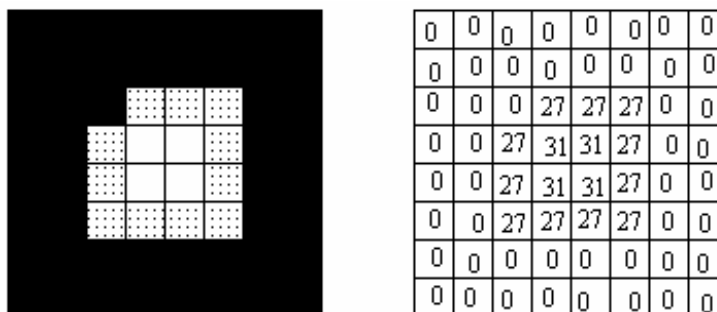


Figura 16 - Imagem depois de ressaltar o objeto.

Desta forma, ressaltou-se o objeto em relação ao fundo da imagem, possibilitando tratar somente as informações contidas na imagem que dizem respeito ao objeto, reduzindo assim as influências do fundo da imagem no resultado do processamento.

Após concluída esta etapa, aplica-se um algoritmo para determinar a borda do objeto [1]. A borda então é composta por um vetor de pontos bidimensionais (x, y). Por exemplo, aplicando-se o algoritmo de detecção da borda no objeto da figura 4 obtém-se o contorno da Figura 17.

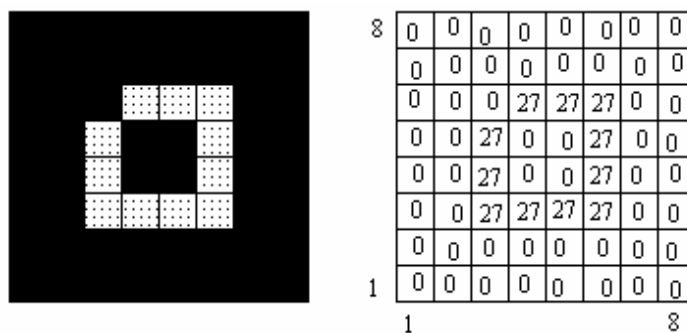


Figura 17 - Borda de um objeto.

O contorno é composto pelos seguintes pontos:

$$\text{Borda} = \{(3,3); (4,3); (5,3); (6,3); (6,4); (6,5); (6,6); (5,6); (4,6); (3,5); (3,4)\}.$$

Contudo, para que uma rede neural artificial possa gerar saídas é necessário haver entradas (estímulos). Neste trabalho as entradas serão os descritores de Fourier, que representam o contorno do objeto que está sendo analisado. Assim, será abordado a seguir como esses descritores são calculados

5.3 Aproximação do Contorno por uma Série de Fourier

Dado um vetor contendo elementos de pontos (x,y) referentes ao contorno de determinado objeto e considerando-se o índice da borda obtida como o tempo t e o ponto correspondente (x, y) como o resultado de uma função complexa $Y(t)$, pode-se então interpretar a borda de um objeto como sendo um sinal discreto no tempo e periódico, isto é possível pois o contorno volta sobre si mesmo (após o ponto final do contorno tem-se o ponto inicial) [10][30][31], permitindo empregar a série de Fourier para aproximá-la. Ou seja,

$$\begin{aligned} \text{Borda} &= \sum Y(t); \text{ onde } Y(t) \in \text{contorno}(c) \\ Y(t) &= x + i.y \end{aligned} \quad \text{Equação 21}$$

Seja $F(t)$ satisfazendo as seguintes condições [6][26][31]:

- $F(t)$ é definida no intervalo $c < t < c + 2l$;
- $F(t)$ e $F'(t)$ são seccionalmente contínuas em $c < t < c + 2l$;
- $F(t + 2l) = F(t)$, isto é, $F(t)$ é periódica com período $2l$.

Então, em todo ponto de continuidade, temos:

$$F(t) = \frac{\alpha_0}{2} + \sum_{n=1}^{\infty} \left(\alpha_n \cos\left(\frac{n\pi t}{l}\right) + \beta_n \text{sen}\left(\frac{n\pi t}{l}\right) \right) \quad \text{Equação 22}$$

Contorno de um objeto, onde:

n = índice dos coeficientes da série de Fourier;

l = período da função $F(t)$;

t = índice da função $F(t)$, correspondente a função complexa relativa ao contorno.

$$\alpha_n = \frac{1}{l} \int_c^{c+2l} (F(t) \cdot \cos\left(\frac{n\pi t}{l}\right)) \quad \text{Equação 23}$$

$$\beta_n = \frac{1}{l} \int_c^{c+2l} (F(t) \cdot \text{sen}\left(\frac{n\pi t}{l}\right)) \quad \text{Equação 24}$$

A série (equação 22) com os coeficientes (equações 23 e 24) é chamada “Série de Fourier” que aproxima a função $F(t)$. As condições acima são muitas vezes chamadas “Condições de Dirichlet” e são suficientes (mas não necessárias) para a convergência da Série de Fourier [31]. Em notação complexa, as Séries de Fourier e os seus coeficientes podem ser escritos como:

$$F(t) = \sum_{n=-\infty}^{\infty} (c_n \cdot e^{j(\frac{n\pi t}{l})}) \quad \text{Equação 25}$$

onde, considerando-se a constante $c = -l$, tem-se:

$$c_n = \frac{1}{2l} \int_{-l}^l (F(t) \cdot e^{-j(\frac{n\pi t}{l})}) dt \quad \text{Equação 26}$$

Considerando-se o período da função como sendo T ($T = 2l$) e que se pode escrever a função $F(t)$ da forma, $F(m) = x(m) + i.y(m)$; com $-l \leq t \leq l$. Desta forma, obtêm-se os coeficientes da forma discreta da transformada de Fourier [10][38] para $0 \leq k \leq (T-1)$ dados por:

$$\alpha_{[k]} = \frac{1}{T} \sum_{m=1}^T x[m] \cdot e^{-j(\frac{2k\pi m}{T})} \quad \text{Equação 27}$$

$$\beta_{[k]} = \frac{1}{T} \sum_{m=1}^T y[m] \cdot e^{-j(\frac{2k\pi m}{T})} \quad \text{Equação 28}$$

Neste momento apresenta-se o primeiro resultado fornecido pelo cálculo da DFT [26]. Analisando-se os coeficientes $\alpha_{[0]}$ e $\beta_{[0]}$, nota-se que estes compõem o centro de massa do contorno representado por $F(m)$, ou seja, o ponto P_0 ($\alpha_{[0]}$, $\beta_{[0]}$) indicará a posição do centro de massa do objeto em relação a referência da câmara.

Define-se os “*Descritores de Fourier*” a partir do módulo dos coeficientes de Fourier (equação 29). Os descritores são utilizados para a identificação inteligente dos objetos [26] quando utilizados em conjunto com redes neurais artificiais (RNA), pois estes são invariantes com a rotação e translação do objeto e conseguem caracterizar de forma muito eficiente os contornos dos objetos. Os “*Descritores de Fourier*” serão utilizados como os vetores de entrada para a RNA que caracterizam o objeto [31].

$$r_{[k]} = \sqrt{|\alpha_{[k]}|^2 + |\beta_{[k]}|^2} \quad \text{Equação 29}$$

$$s_{[k]} = \frac{r_{[k]}}{r_{[1]}} \quad \text{Equação 30}$$

onde:

$r_{[k]}$: invariante com a translação e rotação do objeto;

$s_{[k]}$: invariante com a translação, rotação e com a dimensão do objeto.

Utiliza-se o descritor $s_{[kj]}$ quando se deseja caracterizar o objeto independentemente da dimensão deste. Este descritor é importante em um sistema de vigilância, pois a dimensão do objeto modifica dependendo da sua distância à câmara, porém esse descritor não.

Baseando-se em uma análise empírica, percebeu-se também que o primeiro descritor de Fourier, $r_{[1]}$, é diretamente proporcional ao perímetro do objeto, ou seja, existe uma constante c que relaciona o descritor de Fourier $r_{[1]}$ com o perímetro do contorno analisado. Em uma aplicação prática, isto implica que se colocarmos um objeto com perímetro conhecido sob a câmara, pode-se calcular esta constante e, a partir dela, calcular o perímetro de qualquer outro contorno, desde que a distância entre os demais objetos e a câmara seja a mesma [31].

5.4 Identificação de Objeto na Cena Monitorada

A identificação de um objeto na imagem pressupõe a consciência de quais pontos pertencem a ele e quais pertencem ao fundo ou outros objetos. Para facilitar essa operação, antes de iniciar a monitoração de um local deve-se ter a imagem do mesmo, sem objetos intruso (estranhos ao ambiente monitorado). Esta imagem servirá como matriz, para posteriormente extrair o objeto que se move na cena.

Através de um detetor de movimento, que pode ser implementado via *software* ou *hardware*, ter-se-á a indicação de que algum objeto intruso está movendo-se naquele ambiente. Nesse instante, o sistema faz a aquisição de imagem do local que está sendo vigiado, e utilizando algoritmos de processamento de imagem, subtração e filtros, obtém a região da imagem referente ao objeto em movimento. Então, todos os *pixels* correspondentes à região selecionada são setados para mesma cor, por exemplo branca, e o fundo para outra cor que permita distinguir facilmente os *pixels* que não pertencem a região, tal como a cor preta. Como resultado, tem-se uma imagem do mesmo tamanho da original, contendo em branco a silhueta respectiva ao objeto que se encontra em movimento na cena.

5.5 Obtenção do Contorno e Reconhecimento de Objeto

Entende-se por contorno o subconjunto de pontos de um objeto que o separa do restante da imagem. Como representação, constitui-se de uma curva fechada de pontos conectados, a partir da qual se consegue reconstruir a silhueta do objeto.

Nesta fase, trabalhar-se-á com a imagem obtida no procedimento citado no item anterior, e empregando um algoritmo determina-se a borda do objeto. Então, obtém-se um vetor de pontos bidimensionais (x, y) que corresponde ao contorno do objeto que será analisado.

A partir desse vetor, conforme explicado acima, pode-se obter a aproximação do contorno do objeto por uma série de Fourier. Em consequência, tem-se os descritores de Fourier referentes ao contorno do objeto.

A Figura 18a apresenta como exemplo o contorno de objeto em forma de "T". Observa-se na seqüência, Figura 18b até Figura 18d, o contorno deste objeto empregando-se os primeiros 3, 6 e 20 coeficientes da série de Fourier, respectivamente, possibilitando analisar a contribuição dos vários coeficientes da série para a representação do contorno.

Os descritores, obtidos a partir dos coeficientes, serão então apresentados à rede neural artificial que foi previamente treinada para reconhecer determinadas formas (objetos). A RNA a ser usada para esta tarefa, pelos motivos expostos no capítulo 3, será a rede *Feedforward* com 3 camadas (uma de entrada, uma intermediária e uma de saída).

Após a rede neural ter realizado o processamento serão verificadas as saídas geradas, a fim de determinar se objeto foi ou não reconhecido. Em caso positivo, o sistema executará o procedimento previamente estabelecido.

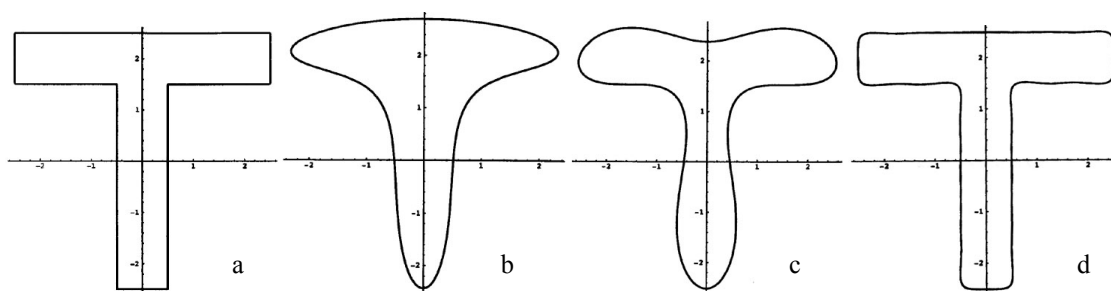


Figura 18 - Contorno em forma de "T" e representação usando 3, 6 e 20 coeficientes de Fourier.

5.6 Experimentos e Resultados

Para verificação e validação do processo de reconhecimento supracitado, foi empregada uma rede neural *feedforward* constituída por uma camada de entrada com 19 neurônios, correspondentes ao número de descritores de Fourier; uma camada intermediária com 8 neurônios e uma camada de saída com 3 neurônios; sendo caracterizada por neurônios com função de transferência do tipo *sigmóide*, e treinada com o algoritmo de aprendizado *backpropagation*. Não existe um método preciso para definir o número de neurônios na camada intermediária. Contudo, uma regra empírica que vem demonstrando resultados satisfatórios nesta aplicação é a utilização de um número de neurônios 150% maior que o da camada de saída [35].

Neste experimento, para facilitar a extração do contorno, os objetos utilizados foram dispostos sobre um fundo com textura única e fotografados.

Como em um sistema de vigilância, salvo algumas exceções, não é possível obter a imagem de um objeto sempre na mesma posição (frontal, perfil e etc.), neste experimento a RNA deveria reconhecer objetos independente da rotação em relação ao eixo vertical dos mesmos. Assim, o conjunto de treinamento foi composto pelos descritores normalizados de Fourier correspondentes ao contorno dos objetos em cada uma das oito direções estabelecidas (direção norte-sul, leste-oeste, e etc.), totalizando 24 amostras. Para extração do contorno cada objeto foi rotacionado cerca de 45°, conforme exemplificado na Figura 19. A aproximação do contorno do objeto em cada uma das direções foi obtida empregando os vinte descritores iniciais da transformada discreta de Fourier, que posteriormente foram normalizados (divididos pelo primeiro descritor), sendo dezenove deles aplicados na camada de entrada da RNA (quadro 1), com a respectiva saída desejada.

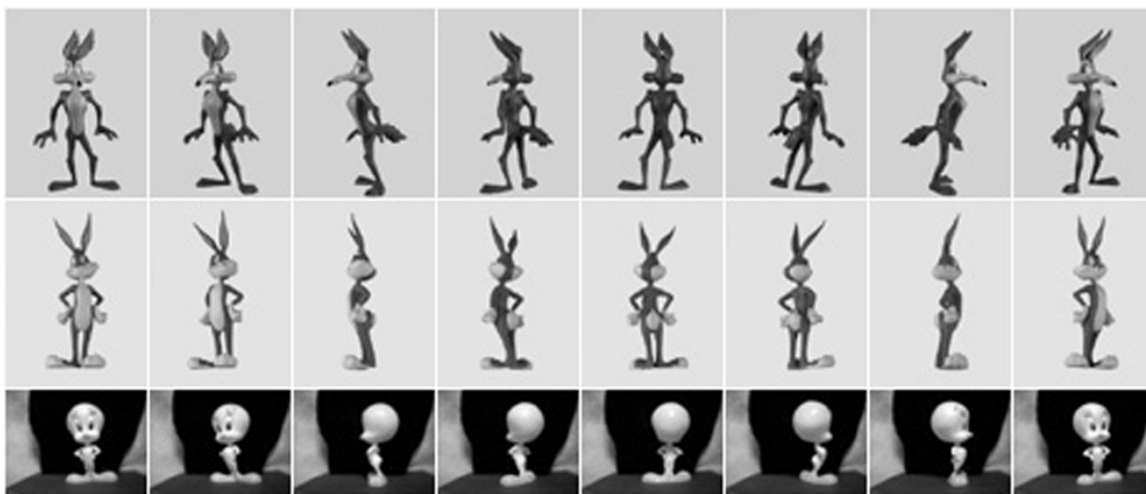


Figura 19 - Objetos utilizados para obter o conjunto de treinamento da RNA.

Entradas da RNA (descritores de Fourier normalizados)																			Saídas		
.189	.220	.222	.241	.145	.066	.051	.075	.039	.025	.022	.033	.048	.041	.025	.022	.028	.037	.020	0	1	0
.077	.080	.054	.130	.074	.076	.049	.009	.031	.033	.020	.018	.029	.010	.024	.010	.015	.008	.014	1	0	0
.168	.289	.056	.077	.034	.083	.032	.023	.012	.016	.031	.004	.013	.010	.010	.012	.006	.003	.001	0	0	1

Quadro 1 - Exemplo de vetores do conjunto de treinamento da RNA, objetos na posição frontal.

A normalização de cada vetor de descritores de Fourier consistiu em dividir todos elementos do vetor pelo primeiro elemento. Assim, o primeiro elemento do vetor normalizado sempre era igual a 1, e desnecessário para o treinamento da rede neural artificial.

As saídas desejadas da RNA estão relacionadas com os objetos empregados em seu treinamento conforme mostra o quadro 2.

Objeto (Personagem)	Saídas Desejadas		
³ Pernalonga	1	0	0
³ Coiote	0	1	0
³ Piu-piu	0	0	1

Quadro 2 - Relação (codificação) entre objeto e saídas desejadas.

Após concluído o aprendizado da rede neural, para efetuar a validação do processo de reconhecimento citado utilizou-se cinco objetos, vide Figura 20, dos quais dois deles não haviam sido empregados no treinamento. Os outros foram usados em posições, distância e direções distintas àquelas apresentadas à RNA durante sua fase de aprendizado.

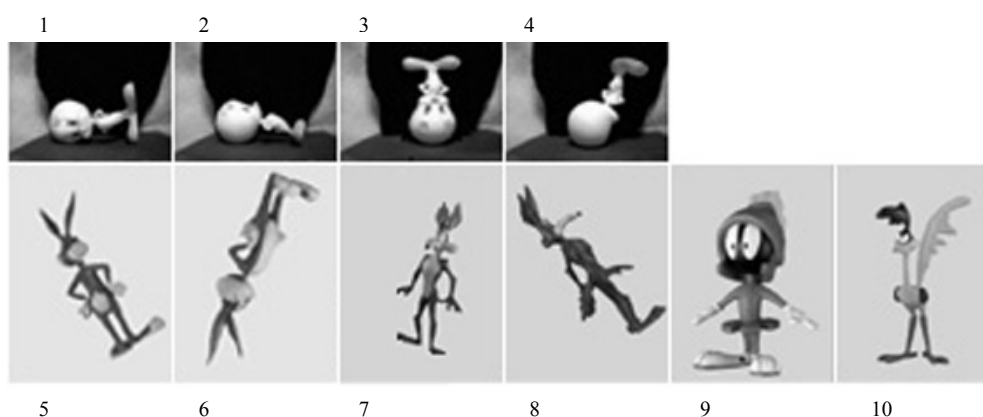


Figura 20 - Objetos utilizados para validar o processo de reconhecimento.

³ Marca registrada de Warner Bros, © 2002.

O quadro 3, mostra as saídas da RNA referentes aos objetos usados na validação do processo de reconhecimento empregado, conforme a ordem em que aparecem na Figura 20 (da esquerda para direita, de cima para baixo). Vale salientar que para tarefa de reconhecimento (classificação) aos valores de saídas da RNA em geral são aplicados uma função limiar, a fim de obter-se valores conforme os especificados no treinamento. Neste caso, utilizou-se os critérios a seguir:

- se valor de saída real da RNA maior ou igual a 0.9 então saída RNA igual a 1;
- se valor de saída real da RNA menor ou igual a 0.1 então saída RNA igual a 0;
- senão saída RNA igual ao valor de saída real da RNA.

Nr Ordem (figura 13)	Saídas da RNA (valor real)			Saídas da RNA (após função limiar)		
1	0,00000000	0,00015668	0,99999818	0	0	1
2	0,00003884	0,00000042	0,99991322	0	0	1
3	0,00000001	0,00000781	0,99999979	0	0	1
4	0,00000222	0,00829127	0,99599039	0	0	1
5	0,99486227	0,00138679	0,00000649	1	0	0
6	0,98997485	0,00034466	0,00003447	1	0	0
7	0,00001574	0,99999980	0,00004659	0	1	0
8	0,00493577	0,99656861	0,00009704	0	1	0
9	0,00000001	0,99992149	0,23999533	0	1	0,2399
10	0,00000066	0,98494276	0,66316851	0	1	0,6631

Quadro 3 - Saídas da RNA (real e com função limiar) para o conjunto de validação especificado.

Pode-se verificar no quadro 3, que a rede neural somente não reconheceu os objetos dos itens 9 e 10 (os personagens ³Marvin e ³Papa-léguas), pois os mesmos não constavam no conjunto de treinamento. Eventualmente a RNA poderia indicar um outro objeto que possuísse características próximas aos que foram utilizados em seu treinamento como sendo um objeto reconhecido. Isto demonstraria a capacidade da RNA em classificar objetos semelhantes como pertencentes a um mesmo grupo. Entretanto, referindo-se a

forma (contorno), os objetos referentes aos itens 9 e 10 possuem poucas características em comum aos existentes no conjunto de treinamento.

Tendo-se apresentado noções sobre processamento de imagens, o funcionamento da RNA tipo *feedforward*, o cálculo dos descritores de Fourier referentes ao contorno do objeto, e mostrado a eficiência do processo para reconhecimento de objetos, será abordado no capítulo 6, a seguir, a um sistema de vigilância inteligente utilizando os métodos supracitados, operando em tempo real.

Capítulo 6

Um Sistema de Vigilância com Detecção de Intrusão Utilizando Inteligência Artificial

6.1 Introdução

O sistema de vigilância ideal seria aquele que realizasse todas as funções pertinentes ao vigia, possibilitando que áreas mais isoladas e fronteiras pudessem ser monitoradas constantemente. No entanto, além de detectar movimento em uma cena teria que conseguir reconhecer todos possíveis objetos daquele escopo, independente de estarem parcialmente ocultos e, sob diversas condições luminosas e climáticas. Com certeza, a solução perfeita seria aquela que disponibilizasse andróides semelhantes ao T-800 e T-1000⁴, que pudessem ver o ambiente, analisar a situação e tomar as decisões quando necessário. Todavia, referindo-se somente à percepção visual, com base no que existe atualmente em visão computacional, pode-se dizer que isto ainda está distante de acontecer.

A tarefa de vigilância de um local realizada através de monitores por um vigia está intimamente ligada ao sistema de percepção visual. Além de ser monótona e às vezes tediosa, um evento no ambiente monitorado pode ocorrer no instante em que, por exemplo, o vigia virou-se para beber água ou café. Assim, seria melhor que um sistema integrado aos monitores disparasse um alarme ao ocorrer um fato, alertando ao vigia que algo anormal está acontecendo, tal como uma pessoa caminhando no pátio de entrada da empresa. Então, o vigia verificaria o monitor e tentaria identificar a pessoa. Se fosse uma pessoa autorizada iria deixa-la entrar, caso contrário executaria o procedimento determinado para aquela ocasião. Entretanto, se um gato estivesse atravessando o pátio o sistema não precisaria alertar o vigia, e este poderia continuar lendo seu livro ou desempenhando outra atividade.

Seguindo esta linha de raciocínio, o sistema teria que extrair e reconhecer objetos intrusos que estão em movimento em uma cena. Há diversos artigos publicados sobre reconhecimento de padrões utilizando métodos estatísticos e/ou redes neurais, e outros tratando sobre métodos empregados para seguir (*tracking*) objetos que estejam movimentando-se em uma cena, através de modelos matemáticos e análise de imagens.

⁴ Andróides do filme de ficção científica O Exterminador do Futuro 2, de James Cameron e Gale Anne Hurd, que além de possuírem capacidades físicas superiores à humana, apresentavam comportamento inteligente.

Dessa forma, este trabalho propõe um modelo que une as técnicas de processamento de imagens com a eficiência em reconhecimento de padrões das redes neurais artificiais, objetivando capacitar um sistema de vigilância a indicar a presença de uma pessoa quando esta mover-se em um ambiente.

6.2 Características e Funcionamento do Sistema Desenvolvido

Um sistema de vigilância inteligente, que detecte pessoas em ambientes fechados e aberto, deve conseguir classificar objetos que estejam em movimento no ambiente monitorado em tempo-real. Para fazer esta tarefa, dentre outros, foram implementados os seguintes processos:

- módulo para aquisição de imagem em tempo real;
- detecção de objeto intruso em movimento no ambiente monitorado;
- algoritmos para identificar o(s) objeto(s) intruso(s) ao ambiente monitorado, desprezando os pertencentes à cena;
- procedimentos para obtenção do contorno referente ao(s) objeto(s) e descritores de Fourier;
- algoritmo para verificar se o objeto em movimento é uma pessoa, independente das características físicas e direção do movimento em relação à câmara, através do emprego de uma rede neural artificial;
- conjunto de procedimentos a serem adotados em determinadas situações, de acordo com o reconhecimento ou não de objeto(s).

Dos itens supracitados, o terceiro apresenta maior dificuldade de ser implementado, principalmente em ambientes externos. Este fato ocorre devido a possível mudança brusca de luminosidade, condições climáticas, sombra projetada pelo objeto em movimento, movimento das folhas de árvores e etc.

O funcionamento do sistema de vigilância consiste basicamente em coletar imagens de uma cena através de uma câmara, extrair os objetos em movimento aplicando algoritmos de processamento de imagens e a transformada discreta de Fourier, a fim de gerar-se vetores que servirão como entradas para a RNA, que por sua vez tentará reconhecer os objetos. E, caso reconheça um dos objetos, o sistema realizará um procedimento preestabelecido.

6.3 Implementação do Sistema de Vigilância

No desenvolvimento deste projeto, fez-se necessária a implementação em *software* de diversas estruturas e algoritmos. Algumas destas estruturas e algoritmos são comuns a diferentes tipos de aplicação. Desta forma, a implementação do *software* deste projeto foi realizada de forma modular, permitindo que estes componentes possam ser reaproveitados em outros projetos.

6.3.1 Ferramentas de Desenvolvimento

As seguintes ferramentas e bibliotecas foram utilizadas no desenvolvimento deste sistema:

- Sistema Operacional: Windows[®] 98 SE;
- Ambiente de Desenvolvimento Delphi 5.0;
- Bibliotecas da Intel para processamento de imagens (IPL) e sistema de visão (OpenCV);
- Bibliotecas referentes ao DirectX 8.1, da Microsoft, para vídeo captura.

A Intel[®] oferece um conjunto de bibliotecas gratuitas para: sistemas de visão [18], processamento de imagens [15], reconhecimento de padrões [19], processamento de sinais [21] e de operações matemáticas [17]. Todas as bibliotecas foram desenvolvidas na linguagem computacional C, otimizadas para os processadores Intel[®] (no entanto, estas podem ser executadas em processadores da família AMD[®]) e suportam o ambiente Windows[®] e o ambiente Unix. Os empregos destas bibliotecas reduzem o tempo de desenvolvimento e permitem alto desempenho para aplicações tempo-real, ressaltando a importância da biblioteca da Intel estar otimizada para o conjunto de instruções em *assembly* de seus processadores.

6.3.2 Implementação de Módulos do Sistema

Embora haja diversos *softwares* disponíveis para reconhecimento de padrões através de RNA, que possuem diferentes topologias de redes neurais, algoritmos de aprendizado e etc, tal como Matlab, houve a necessidade de implementação de um módulo de RNA para permitir a integração deste com o sistema de vigilância desenvolvido. De forma similar os demais módulos do sistemas foram implementados, sempre levando em consideração a necessidade de otimizar o tempo de processamento, já que trata-se de uma aplicação em tempo-real.

6.4 Algoritmo de Processamento de Imagens

Devido à necessidade de detecção e obtenção de contorno do objeto intruso, em movimento no ambiente monitorado, principalmente quando for ambiente externo, fez-se necessário o desenvolvimento de um algoritmo de processamento de imagens que conseguisse filtrar pequenos movimentos e minimizar a sombra gerada pelo objeto. A estratégia adotada utiliza a comparação entre uma imagem modelo (ambiente sem objetos em movimento) e a imagem que está sendo capturada pela câmara, para efetuar a detecção de objeto intruso na cena. Nesta seção apresenta-se o algoritmo desenvolvido. Este algoritmo utiliza as técnicas de processamento de imagens apresentada no capítulo 4. As técnicas de processamento de imagens empregadas foram selecionadas devido ao resultado que estas produzem no tratamento de imagens. Estas são técnicas clássicas, amplamente difundidas no meio científico e descritas na literatura [40][8][10][12][26][32][40]. Outra vantagem destas técnicas é que estas estão disponíveis nas bibliotecas de processamento de imagens da Intel [15][16][18][19]. Desta forma, estas técnicas encontram-se já implementadas em linguagem C++ de forma otimizada para a linha de processadores da Intel. O que garante um alto desempenho em aplicações de sistemas de visão e uma grande portabilidade (Unix e Windows[®]). Por estas razões e buscando gerar uma solução tecnológica e economicamente viável que possa, realmente, ser aplicada no seguimento de segurança patrimonial, estas técnicas de processamento de imagens foram utilizadas.

A Figura 21 apresenta o algoritmo de processamento de imagens composto das seguintes etapas básicas:

1. **Início:** usuário inicia o programa aplicativo;
2. **Inserir Modelo:** o usuário insere a imagem de um ambiente a ser monitorado para ser utilizada como modelo pelo sistema. Nesta etapa, o usuário deve cuidar para fornecer uma imagem em que não há objeto intruso ao ambiente;
3. **Adquirir Imagem:** o usuário inicia o processo de monitoração de um ambiente;
4. **Detectar Movimento:** cada imagem digitalizada pela câmara (*frame*), a uma taxa de 10 *frames* por segundo, é comparada com a imagem capturada no instante imediatamente anterior ($t-1$), permitindo que seja verificado se houve movimento na cena. É indicado um movimento caso a subtração das imagens resulte em número de *pixels* superior ao valor limite, parâmetro do sistema.
5. **Obter Objeto Intruso:** aplica-se um conjunto de algoritmos para obter o objeto intruso ao ambiente e tentar eliminar sua sombra, caso possua.

6. **Obter Contorno:** uma vez que o objeto intruso foi localizado na imagem e filtros aplicados, a partir do seu contorno externo obtém-se os descritores de Fourier.
7. **Reconhecer Objeto:** a partir do contorno externo do objeto intruso, aplica-se o processo descrito no capítulo 5 para tentar reconhecê-lo

A aquisição da imagem do ambiente monitorado pelo sistema, foi realizada por uma *webcam* e de forma automática, através da porta USB (*Universal Serial Bus*). As imagens adquiridas de forma contínua são digitalizadas em formato bitmap (BMP). Estas imagens são então utilizadas pelo aplicativo, desenvolvido no âmbito deste trabalho, para efetuar a vigilância de um ambiente.

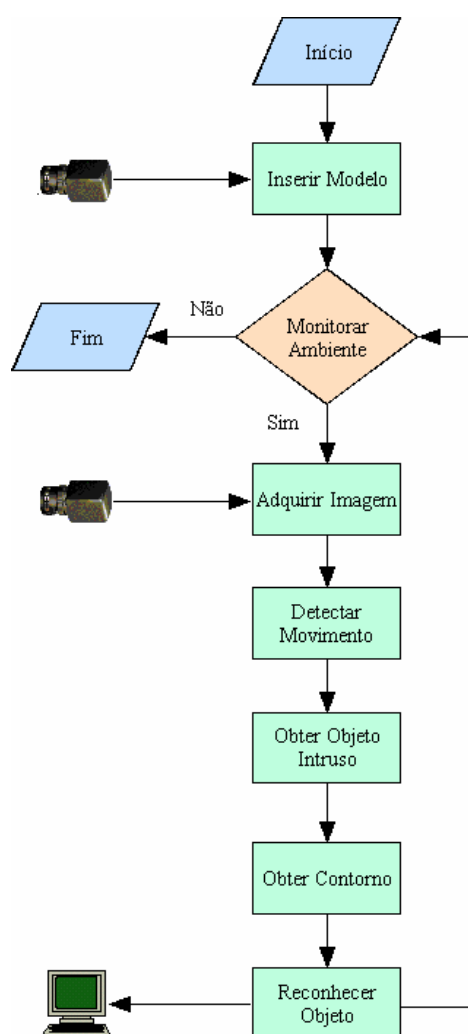


Figura 21 - Algoritmo de Processamento de Imagens.

A seguir, apresentam-se as técnicas empregadas nas etapas mais importantes deste algoritmo.

6.4.1 Algoritmo de Detecção de Movimento

O algoritmo de detecção de movimento funciona da seguinte maneira, ilustrada na Figura 22:

1. Criam-se duas imagens novas em tons de cinza, 8 *bits* (onde 0 corresponde a cor preta e 255 a cor branca), uma a partir da imagem modelo colorida (RGB) do ambiente monitorado e a outra da imagem colorida atual, ambas adquiridas através da câmara;
2. As imagens em tons de cinza, oriundas das coloridas, capturadas pela câmara, em geral, possuem ruídos devido à variação da iluminação quando capturadas pela câmara. O valor de um *pixel* situado na mesma posição em *frames* consecutivos, pode possuir pequena variação. Por isso, emprega-se um filtro de Média (núcleo de convolução 3x3) para “corrigir” o valor. Deste ponto em diante, essas imagens em tons de cinza serão chamadas de imagem atual em cinza e imagem modelo em cinza;
3. Compara-se a imagem atual em cinza com a imagem modelo em cinza, através de uma operação de subtração de imagens (ver item 4.5). O resultado é uma imagem contendo os *pixels* que diferem nas duas imagens. Se a quantidade de *pixels* na imagem resultante for superior a um número definido, parâmetro do sistema que pode ser alterado conforme a resolução empregada, então considera-se que há movimento no ambiente monitorado. Neste caso, o parâmetro foi definido como sendo 250 *pixels*.
4. Caso, o processo do item anterior indique que não há movimento na cena, então a imagem atual em cinza é copiada para imagem modelo em cinza. Isto possibilita evitar, principalmente em ambiente externo, que os movimentos das folhas de árvores ou pequena mudança de luminosidade sejam indicadas como objeto intruso.

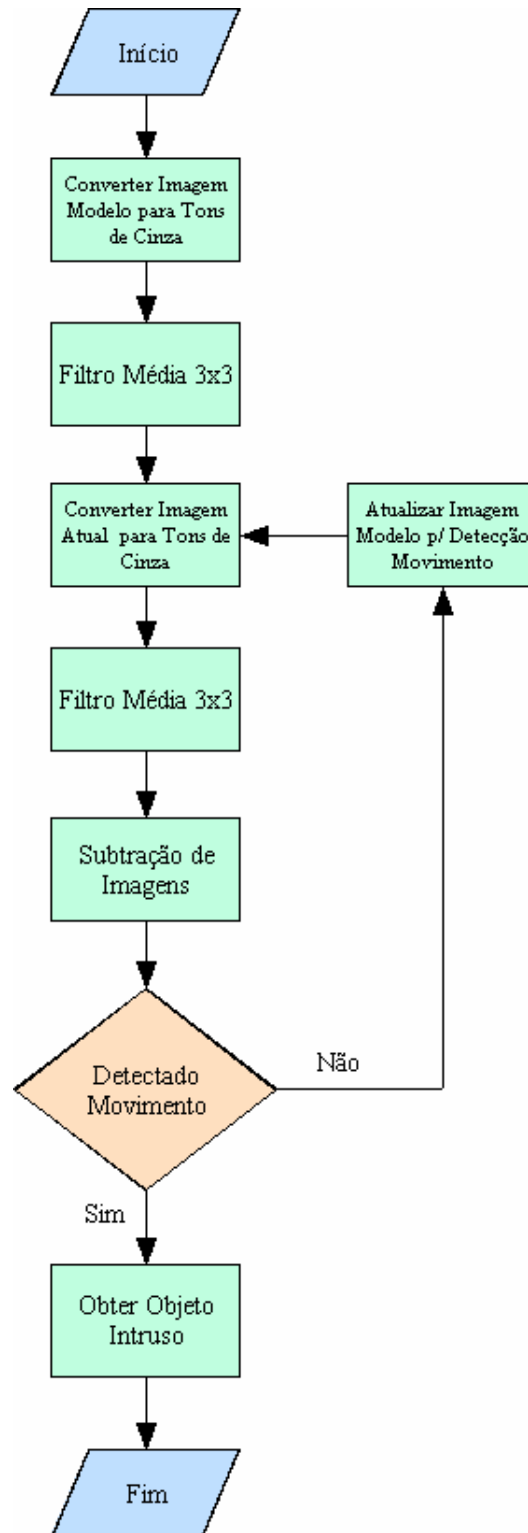


Figura 22 - Algoritmo de detecção de movimento

6.4.2 Algoritmo de Obtenção de Objeto Intruso

Após ter sido detectado movimento na cena vigiada, prossegue-se com a obtenção da região da imagem referente ao objeto intruso e tentativa de eliminação da sombra do mesmo. Este algoritmo funciona da seguinte maneira, ilustrada na Figura 25 :

1. A partir da imagem modelo e imagem atual (já convertidas em tons de cinza), realiza-se a operação de subtração, conforme exemplificado na Figura 23. Primeiro, faz-se a subtração da imagem modelo e atual, obtendo-se como resultado uma imagem $S1$. Nesta imagem encontram-se os *pixels* candidatos à sombra do objeto, pois esta imagem ($S1$) contém todos os *pixels* que possuem intensidade maior na imagem modelo do que na imagem atual. Assim, para reduzir/eliminar áreas referentes à sombra optou-se por aplicar um *threshold* nesta imagem, com valor 48 (obtido de forma empírica).



a) Imagem Modelo [41]



b) Imagem Atual [41]



c) Subtração Imagem Modelo e Imagem Atual ($S1$)



d) Subtração Imagem Atual e Imagem Modelo ($S2$)

Figura 23 – Exemplo da operação de subtração entre duas imagens



a) Resultado da subtração entre as imagens

a) Imagem (R), resultado da subtração entre as imagens utilizando um *threshold* na imagem ($S1$).

Figura 24 – Comparação entre resultados de subtração

Na Figura 24, pode-se comparar o resultado obtido sem e com o emprego de *threshold* para tentar eliminar sombras do objeto.

2. Na imagem resultante do processo de subtração (R) são aplicados operadores morfológicos, erosão e dilatação, funcionando como um filtro para suprir eventuais ruídos.
3. Então, esta imagem R é utilizada como uma máscara na imagem atual para destacar o objeto intruso ao ambiente monitorado. Também através da imagem R aplica-se um algoritmo de detecção de contornos (ver item 4.9). Os contornos localizados são armazenados em vetores de pontos.

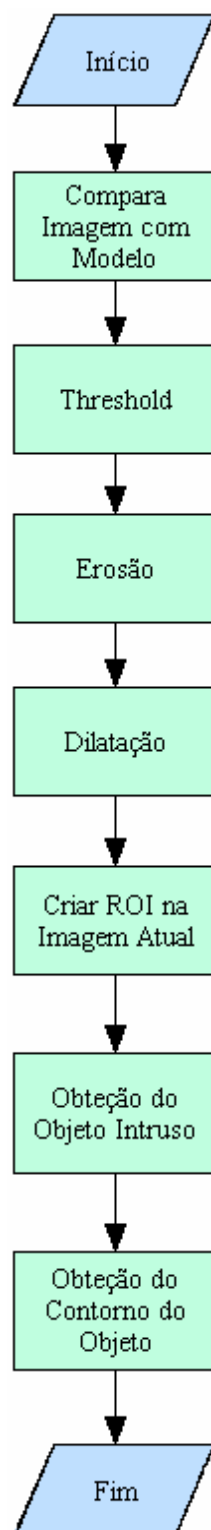


Figura 25 – Algoritmo de obtenção de objeto intruso

6.4.3 Algoritmo de Reconhecimento de Objeto Intruso

Assim, o contorno obtido pelo algoritmo acima, que emprega funções da biblioteca da Intel[®], é utilizado para tentar reconhecer o objeto intruso ao local monitorado. Para executar esta tarefa, utiliza-se a técnica descrita e demonstrada no capítulo 5.

1. A fim de evitar que o processo de reconhecimento seja executado em objetos demasiadamente pequenos, os quais provavelmente não seriam uma pessoa, ou, ainda estaria muito longe do local, há um filtro que somente realiza os demais procedimentos caso o contorno do objeto possui mais de 40 *pixels*.
2. Calcula-se os descritores de Fourier a partir do contorno pertencente ao objeto em movimento. Os '19' primeiros descritores são, então, normalizados e utilizados como entrada para a Rede Neural.
3. Em seguida, a Rede Neural processa o vetor contendo os descritores de Fourier normalizados. Por fim, a saída da Rede Neural é interpretada para verificar se o objeto intruso é uma pessoa ou não. Este algoritmo está sintetizado na Figura 26.

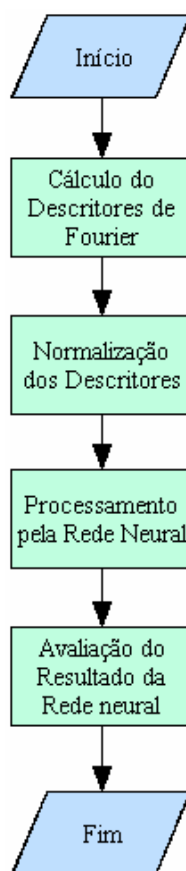


Figura 26 - Algoritmo de Reconhecimento de Objeto Intruso

A Rede Neural utilizada é do tipo *Feedforward* (direta) com apenas uma camada interna de neurônios. Os neurônios possuem função de ativação do tipo sigmóide. Como, no conjunto de amostras utilizadas para testar o sistema de medição, existiam apenas dois tipos de desgaste de ferramenta (desgaste de flanco e quebra da ferramenta), a Rede Neural foi configurada para apenas classificar estes dois tipos. Desta forma, a Rede Neural possui a seguinte configuração: 18 neurônios na camada de entrada, 3 neurônios na camada interna e 2 neurônios na camada de saída. Se os neurônios de saída possuírem a seqüência (1 - 0) então o objeto intruso foi classificado pela RNA como sendo uma pessoa.

A Rede Neural foi treinada, utilizando-se 37 contornos, dos quais 21 pertenciam à pessoas e 16 à outras formas diversas, através do algoritmo de aprendizado *Backpropagation* (ver item 3.4.2) com taxa de aprendizado igual a 0,5 e parâmetro de momento igual a 0,6. A Rede Neural leva, em média 270 segundos para aprender a classificar os dois tipos de contornos, utilizando um computador do tipo PC com processador Pentium IV 1.6 Ghz e 256 MB RAM.

6.5 Software de Sistema de Vigilância

Para a implementação deste sistema de vigilância, desenvolveu-se um *software* para Windows[®] que implementa as técnicas descritas neste documento. Este software utiliza os módulos supracitados e bibliotecas da Intel[®] para processamento de imagens.

Todos os algoritmos descritos neste trabalho foram implementados neste aplicativo. Desta forma, torna-se desnecessário descrever a estrutura detalhada do *software*, uma vez que as técnicas empregadas já foram descritas neste documento.

Na Figura 27 apresenta-se a tela de interface com o usuário. A interface do usuário contém uma grande área que exibe a imagem do ambiente monitorado. No canto esquerdo inferior há um quadro que exibe o(s) objeto(s) intruso(s) ao ambiente monitorado. Através do botão “Configurações”, no canto superior direito, o usuário pode modificar algumas características da imagem que está sendo capturada, bem como a fonte de vídeo. Ao pressionar o botão “Iniciar Monitoramento”, no lado esquerdo superior, a imagem que está sendo exibida naquele instante é automaticamente registrada como imagem modelo do ambiente vigiado e inicia-se o processo de monitoramento. Os botões “Capturar Bitmap” e “Capturar Vídeo”, realizam a captura da imagem que está sendo exibida, gravando um arquivo no formato bitmap ou um vídeo (formato AVI), respectivamente. O usuário pode modificar a resolução de captura da imagem através do *Combo-Box* (botão do *Windows*

com uma lista de opções para a escolha do usuário) existente no canto superior esquerdo da interface do sistema.

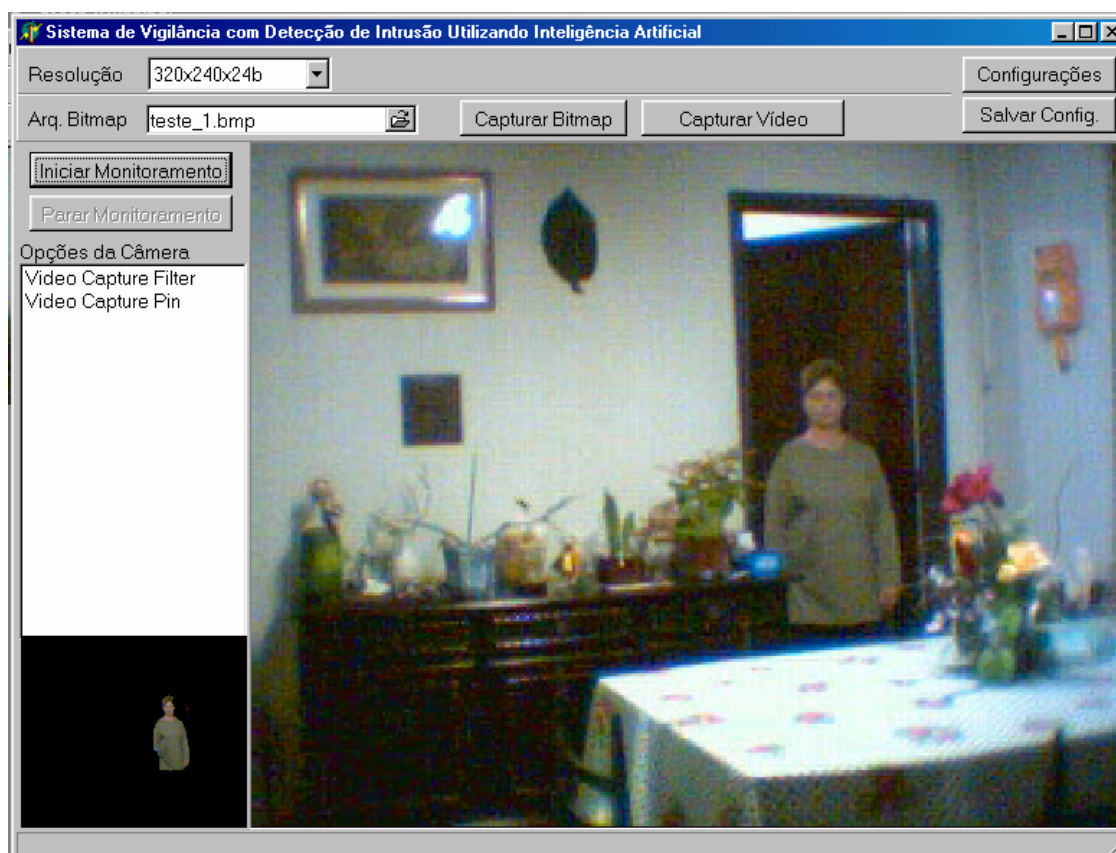


Figura 27 - Interface do Sistema de Vigilância


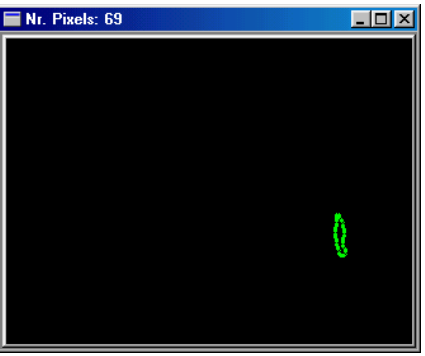

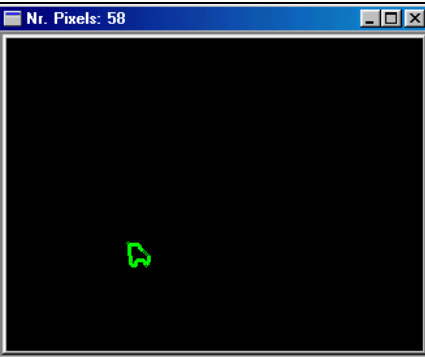

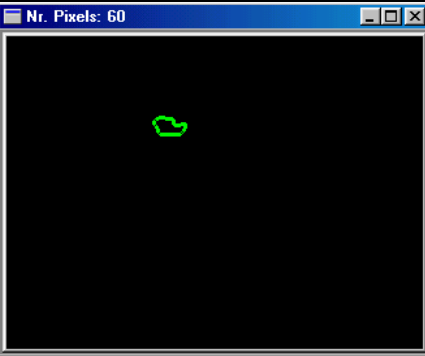
6.6 Teste e Avaliação do Sistema de Vigilância

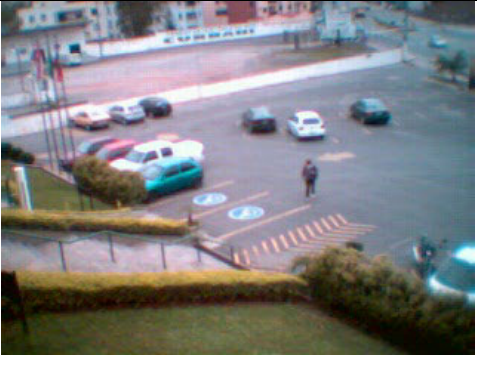
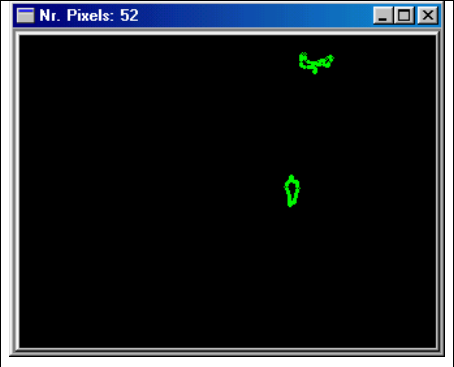

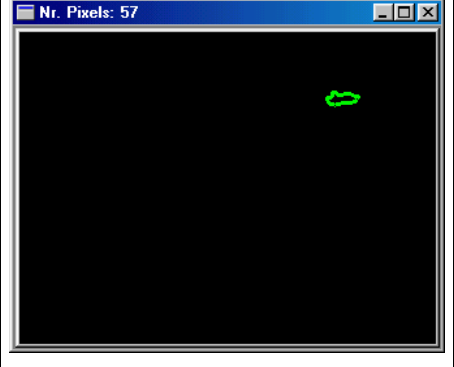

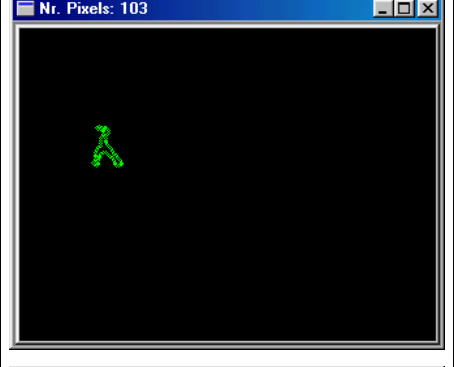

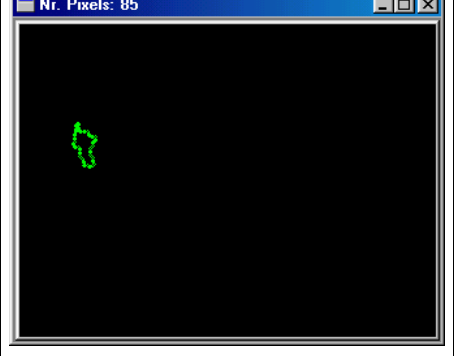
Para testar o sistema de vigilância e reconhecimento de objetos intrusos fez-se três sessões de monitoração do pátio de estacionamento de uma empresa, que contém ao fundo uma rodovia com tráfego intenso de veículos. Para realizar esta tarefa foi empregada uma *webcam* da empresa LEGO, que possui ajuste automático de luminosidade, resolução máxima de 320 x 240 *pixels* e taxa de aquisição de até 30 *frames* por segundo (*fps*), conectada, via porta USB, a um computador do tipo PC com processador Pentium® IV 1.6 GHz e 256 MB RAM.

Abaixo, a Figura 28 ilustra imagens usadas como modelo para a execução do processamento de imagens pelo sistema desenvolvido, com alguns resultados apresentados no quadro 4. Este, contém imagens com objetos intrusos, os contornos obtidos e processados através do algoritmo de reconhecimento de objetos (ver item 6.4.3).



Figura 28 - Imagens modelo do conjunto de teste do Sistema de Vigilância.

Item	Imagem com Movimento Detectado	Contorno Obtido	Saída RNA
1			Pessoa
2			Não é Pessoa
3			Não é Pessoa

4			<p>Não é Pessoa e Pessoa</p>
5			<p>Não é Pessoa</p>
6			<p>Pessoa</p>
7			<p>Pessoa</p>

Quadro 4 - Verificação dos contornos obtidos e resposta da RNA referente aos objetos intrusos.

Nota-se no quadro 4, acima, que no item 1 o objeto em movimento foi classificado como sendo uma pessoa, e no item 2, o mesmo objeto foi classificado como não sendo uma pessoa. Este problema está diretamente ligado ao processamento de imagem para obtenção do contorno. Devido a pessoa ter passado atrás do corrimão existente naquele local, seu

contorno foi dividido, havendo a separação de seus membros inferiores. O contorno destes não apareceram na imagem de contornos porque provavelmente possuíam menos de 40 *pixels*, não satisfazendo a condição do algoritmo implementado (ver item 6.4.2).

O quadro 5 apresenta a relação entre a quantidade de respostas fornecidas pelo sistema de vigilância implementado, para cada um dos possíveis resultados, após a análise de 55 *frames* não contínuos, e verificado qual deveria ser a resposta correta a ser dada pelo sistema para aquela situação. Isso não significa que a pessoa não foi identificada pelo sistema durante todo seu percurso pelo ambiente monitorado, mas sim, que naquele *frame* que foi analisado a resposta fornecida pelo sistema não foi compatível com a realidade do objeto.

Objeto Intruso	Como o Sistema Reconheceu o Objeto Intruso			Total
	Pessoa	Não Pessoa	Não Detectou	
Pessoa	26	6	2	34
Não Pessoa	5	16	-	21
Total	31	22	2	55

Quadro 5 – Relacionamento entre objeto intruso e reconhecimento efetuado pelo sistema.

Através do quadro acima, pode-se calcular, por exemplo, qual a probabilidade do sistema indicar corretamente que uma pessoa está movimentando-se no ambiente monitorado. Logo, temos que a probabilidade é de 76,5 %. Parte desses resultados devem-se também ao fato da baixa resolução propiciada pela câmara de aquisição (320 x 240 *pixels*), que dificulta a extração do contorno dos objetos quando estes encontram-se em posições mais distantes.

Capítulo 8

Conclusões e Futuros Desenvolvimentos

A partir dos resultados obtidos (ver item 6.6), conclui-se que o sistema de vigilância utilizando inteligência artificial (RNA) é viável, porém necessita sofrer pequenos ajustes a fim de melhorar a taxa de acerto, que está diretamente vinculada aos processos para extração de objetos intrusos ao ambiente. Entretanto, sempre deve haver a preocupação com o tempo de processamento, já que este é um sistema que deve operar em tempo real.

Vale salientar que as mesmas técnicas descritas podem ser empregadas para reconhecer outros objetos, possibilitando a realização de interação entre processos (automação). A grande vantagem em utilizar os descritores de Fourier como estímulo à rede neural artificial está na invariabilidade destes em relação à rotação e, principalmente, à dimensão do objeto, já que o tamanho de um objeto na imagem é inversamente proporcional à distância entre ele e a câmara. Outro fator importante refere-se a quantidade de neurônios necessários na camada de entrada da rede neural artificial. Conforme apresentado na Figura 18, os primeiros 20 descritores são suficientes para representar a forma genérica do objeto. Dependendo da quantidade de descritores utilizados para representar o contorno de um objeto pode-se separar os objetos em classes mais específicas ou genéricas.

A escolha de empregar RNA para reconhecer objetos em vez de modelos matemáticos, deve-se ao fato dela ser uma generalizadora universal de funções, aprender por exemplos e ser boa interpoladora. Assim, se desejar que outro objeto também seja reconhecido, basta apresentar exemplos à RNA e realizar o treinamento. Embora o treinamento da RNA, neste caso, requeira bastantes exemplos devido a diversificação de formas do objeto a ser reconhecido (pessoa em movimento), verificou-se que a probabilidade do sistema em classificar corretamente os objetos intrusos é de aproximadamente 77%, quando analisados *frames* aleatoriamente.

Para aumentar essa probabilidade, os algoritmos de processamento de imagens precisam ser refinados a fim de executarem a obtenção do contorno do objeto intruso com maior precisão, sobretudo quando o objeto for pequeno em relação ao tamanho da imagem ou possuir oclusão parcial, e talvez uma câmara de aquisição de imagens que possua maior

resolução. Todavia, o sistema mostrou ser bastante flexível quanto a mudança de luminosidade e pequenos movimentos, tal como folhas e ramos de plantas.

Uma sugestão para trabalhos futuros é desenvolver algoritmos para determinar a distância do objeto intruso em relação à câmara e em conseqüência a determinados pontos de interesse, tal como portas, armazéns e etc. Também seria interessante saber a direção e velocidade do movimento do objeto, bem como realizar seu acompanhamento por toda área monitorada, inclusive através de várias câmaras, se o ambiente monitorado as possuir.

Outras evoluções interessantes no sistema de vigilância são:

- Uso de câmaras que possam operar durante o período noturno (*infrared*) e desenvolvimento dos respectivos módulos para processamento dessas imagens;
- Módulo para que o sistema possa ser operado e comunicar-se remotamente, incluindo uma pequena base de conhecimento para tomada de algumas decisões;
- Módulo para registro de fatos ocorridos.

Neste trabalho foi desenvolvido o protótipo de um sistema direcionado ao atingimento do objetivo proposto. Apesar de o sistema ter sido testado apenas para reconhecer se o objeto introduzido e movimentando-se em uma área monitorada é ou não uma pessoa, a metodologia pesquisada é bastante genérica, podendo ser aplicada para reconhecer outras formas além da humana, e empregada em outras finalidades, por exemplo em controle de semáforos de trânsito. Futuramente, tem-se a intenção de transformar essa tecnologia em um produto comercial, disponibilizando-a para o mercado.

Referências Bibliográficas

- [1] ARCELLI, Carlo; BAJA, Gabriella Sanniti Di: *A Width-Independent Fast Thinning Algorithm*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-7, n° 4, July 1985.
- [2] ATIYA, A. F.; FERNANDEZ, B.; MUTHUSAMI, J.; PARLOS, A.G.; e TSAI, W.K.: *An Accelerated Learning Algorithm for Multilayer Perceptron Networks*. IEEE Transactions on Neural Network, vol. 5, no. 3, Março de 1994.
- [3] BALLARD, D. H.; BROWN, C. M.: *Computer Vision*. Prentice-Hall, 1982.
- [4] BARRETO, Jorge M.: *Redes Neurais Artificiais: Fundamentos, Aplicações e Implementações*. II CIPEEL
- [5] BARRETO, Jorge M.: *Inteligência Artificial no Limiar do Século XXI*. ppp Edições, 3ª Edição, 2001
- [6] CARVALHO, Carlos A. A. de: *Séries de Fourier e Problemas de Valores de Contorno*. Editora Guanabara Dois S.A., 1978.
- [7] DEMUTH, Howard; BEALE, Mark: *Neural Network Toolbox User's Guide - For Use with MatLab*. The Maths Works Inc.
- [8] FISHER, R.: *CVonline: Image Transformations and Filters*, em <http://www.dai.ed.ac.uk/CVonline/transf.htm>, Outubro, 2001.
- [9] FREEMAN, James A.; SKAPURA, David M.: *Neural Networks - Algorithms, Applications and Programming Techniques*, Addison Wesley, 1991.
- [10] GOMES, J. e VELHO, L.: *Computação Gráfica: Imagem*, Rio de Janeiro, IMPA/SBM, 1994.
- [11] GONZALEZ, R. C. e WOODS, R. E.: *Processamento de Imagens Digitais*. Editora Edgard Blücher Ltda, 1992.
- [12] HARALICK, R. M. e SHAPIRO, L. G.: *Computer and Robot Vision*, ed. Addison-Wesley, 1992.
- [13] HARITAUGLO, I., HARWOOD, D. e DAVIS, S. L.: *W4: Real-Time Surveillance of People and Their Activities*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 22, No. 8, Agosto 2000.
- [14] HAYKIN, Simon.: *Redes Neurais: princípios e práticas*, Editora Bookman, 2ª Edição, 2001.
- [15] INTEL: *Intel Image Processing Library*, em <http://developer.intel.com/software/products/perflib/ipl>, Intel, EUA, 2001.

- [16] INTEL: *Intel Image Processing Library – Reference Manual*, em <http://developer.intel.com/software/products/perflib/ipl/index.htm>, Intel, EUA, 2001.
- [17] INTEL: *Intel Math Kernel Library*, em <http://developer.intel.com/software/products/mkl/>, Intel, EUA, 2001.
- [18] INTEL: *Intel Open Source Computer Vision Library*, em <http://www.intel.com/research/mrl/research/opencv>, Intel, EUA, 2001.
- [19] INTEL: *Intel Open Source Computer Vision Library – Reference Manual*, em <http://www.intel.com/research/mrl/research/opencv/index.htm>, Intel, EUA, 2001.
- [20] INTEL: *Intel Recognition Primitives Library*, em <http://developer.intel.com/software/products/perflib/rpl/>, Intel, EUA, 2001.
- [21] INTEL: *Intel Signal Processing Library*, em <http://developer.intel.com/software/products/perflib/spl/>, Intel, EUA, 2001.
- [22] JÄHNE, B.: *Handbook of Computer Vision and Applications – Volume 1 Sensors and Imaging*, ed. Academic Press, Heidelberg, Alemanha, 1999.
- [23] JÄHNE, B.: *Handbook of Computer Vision and Applications – Volume 2 Signal Processing and Pattern Recognition*, ed. Academic Press, Heidelberg, Alemanha, 1999.
- [24] JAIN, A. K.; JIANCHANG Mao e MOHIUDDIN, K. M.: *Artificial Neural Networks: A Tutorial*. Computer, Março 1996.
- [25] JAIN, Ramesh C.; Jain, K. Anil.(ed): *Analysis and Interpretation of Range Images*, Springer-Verlang, 1990.
- [26] JONKER, P.: *Image Processing Fundamentals*, at <http://www.ph.tn.tudelft.nl/Courses/FIP/noframes/fip-Contents.html>, Março, 1998.
- [27] KIM, H; NAM, K.: *Object Recognition of One – DOF Tool by a Backpropagation Neural Net*, IEEE Transactions on Neural Networks, vol. 6 n° 2, March 1995.
- [28] KING-SUN FU, R. Gonzalez; Lee, C.: *Robotics: Controls, Sesing, Vision and Inteligence*. McGraw-Hill, 1987.
- [29] LOESCH, Cláudio e SARI, Solange T.: *Redes Neurais Artificiais: Fundamentos e Modelos*. Editora FURB, 1996.
- [30] MASTER, Tomothy: *Signal and Image Processing with Neural Networks*. Academic Press, 1993.
- [31] MCLAUGHLIN, R. A. e ALDER, M. D.: *Technical Report – The Hough Transform versus the UpWrite*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, n° 4, pag. 396 – 400, Abril, 1998.

- [32] NCC.: *Image Processing Resource List*, em http://www.ncc.com/misc/ipt_sites.html, Network Cybernetics Corporation, Outubro, 2001.
- [33] OLSON, T. e BRILL, F.: *Moving Object Detection and Event Recognition Algorithms for Smart Cameras*, Proc. DARPA Image Understanding Workshop, pp. 159-175, 1997.
- [34] ORTH, Alexandre.: *Projeto RAP*. Desenvolvimento e Implementação de um Sistema Inteligente Baseado em Redes Neurais Artificiais para o Reconhecimento Automático de Peças, 1999.
- [35] ORTH, Alexandre.: *Desenvolvimento de um Sistema de Visão para Medir o Desgaste de Flanco de Ferramentas de Corte*, Dissertação de Mestrado, UFSC, Dezembro/2001.
- [36] RIEDMILLER, M.: *Advanced Supervised Learning in Multi-layer Perceptrons – From Backpropagation to Adaptive Learning Algorithms*, Journal of Computer Standards and Interfaces, Special Issues on Neural Networks (5), 1994.
- [37] ROSENFELD, A.: *Image analysis: Problems, progress and prospects*. Morgan Kaufmann Publishes, 1983.
- [38] SPEIGEL, M. R.: *Transformadas de Laplace*, Coleção Schaum, Editora McGraw-Hill do Brasil. 1975.
- [39] SIMON, G. e CHATTERJJE, M.: *Introduction to Image Morphology*, em <http://cobb.ece.psu.edu/projects/interact/morphology.html>, Penn State University, Electrical Engineering, The Multidimensional Image Processing Lab., 2001.
- [40] SIMON, G. e CHATTERJJE, M.: *Introduction to Image Morphology*, em <http://cobb.ece.psu.edu/projects/interact/morphology.html>, Penn State University, Electrical Engineering, The Multidimensional Image Processing Lab., 2001.
- [41] WANGENHEIM, A. von.: *Seminário de Introdução a Visão Computacional*, em www.inf.ufsc.br/~visao, UFSC, Brasil, 2001.
- [42] WREN, C., AZARBAYEJANI, DAREELL, T. e PENTLAND, A.: *Pfinder: Real-Time Tracking of the Human Body*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 19, No. 7, Julho 2000.