

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL  
INSTITUTO DE INFORMÁTICA  
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

LUÍS RENATO CRUZ ERPEN

**Reconhecimento de Padrões em Imagens por  
Descritores de Forma**

Dissertação apresentada como requisito parcial  
para a obtenção do grau de Mestre em Ciência  
da Computação

Prof. Dr. Paulo Martins Engel  
Orientador

Porto Alegre, julho de 2004

## CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Erpen, Luís Renato Cruz

Reconhecimento de Padrões em Imagens por Descritores de Forma/Luís Renato Cruz Erpen – Porto Alegre: Programa de Pós-Graduação em Computação, 2004.

113f.:il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR – RS, 2004. Orientador: Paulo Martins Engel.

1. Descritores de Forma. 2. Reconhecimento de Padrões em Imagens 3. Classificadores Neurais 4. Classificadores Estatísticos. I. Engel, Paulo Martins. II. Título.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Reitora: Profa. Wrana Panizzi

Pró-Reitor de Ensino: Prof. José Carlos Ferraz Hennemann

Pró-Reitor Adjunto de Pós-Graduação: Prof. Jaime Evaldo Fernsterseifer

Diretor do Instituto de Informática: Prof. Philippe Olivier Alexandre Navaux

Coordenador do PPGC: Prof. Carlos Alberto Heuser

Bibliotecária-Chefe do Instituto de Informática: Beatriz Regina Bastos Haro

## AGRADECIMENTOS

Ao meu professor orientador, Paulo Engel, pela orientação e dedicação durante todo este tempo. Pelas palavras de incentivo e pelo apoio, mesmo nos momentos em que eu não me sentia capaz. A sua persistência em me manter focado no meu trabalho e não buscando mais métodos e técnicas.

Agradeço a minha família que sempre me confortou nos momentos difíceis, naqueles onde tudo parece não ter sentido. Eles foram a motivação para que este trabalho fosse desenvolvido e concluído. À minha namorada Ana Paula Hickmann que tolerou minhas ausências e momentos de reflexão.

Um agradecimento especial a duas pessoas que além de colegas tornaram-se grandes amigos: João Valiati e Tibério Caetano, com os quais aprendi muito durante essa convivência tanto no campo pessoal quanto no profissional. Diversos foram os momentos onde nossas discussões foram extremamente proveitosas para auxiliar-me a mudar ou reforçar alguns de meus muitos pensamentos.

Meus amigos e colegas no Instituto de Informática, Everaldo Daronco, Vinícius Serafim, César Ida, Filipe Lopes, Cláudio Pitbull Gomes e Carlos Michel Betemps. Ao pessoal do futebol, que além dos já citados, incluem-se Luís Otávio Soares (LO) e Jorge Cunha, meu muito obrigado.

Não poderia deixar de citar aos amigos Chico, Marguit, Carlão, Carlos Henrique, Miguel e Laone que sempre tentaram entender o que eu estava fazendo e me incentivaram a seguir buscando a conclusão de mais esta etapa de minha vida. Ao Chico que aceitou a empreitada de tirar as fotos das placas dos automóveis para meu estudo de caso.

Aos funcionários do PPGC, que pela solicitude, contribuíram para meu aprendizado.

Ao CNPQ, pelo apoio financeiro.

## SUMÁRIO

<b>LISTA DE ABREVIATURAS E SIGLAS .....</b>	<b>6</b>
<b>LISTA DE FIGURAS .....</b>	<b>7</b>
<b>LISTA DE TABELAS .....</b>	<b>9</b>
<b>RESUMO .....</b>	<b>10</b>
<b>ABSTRACT .....</b>	<b>11</b>
<b>1 INTRODUÇÃO.....</b>	<b>12</b>
1.1 Motivação .....	14
1.2 Objetivo do Trabalho .....	16
1.3 Organização do Trabalho .....	17
<b>2 RECONHECIMENTO DE PADRÕES EM IMAGENS.....</b>	<b>18</b>
2.1 Imagem Digital.....	18
2.2 Etapas do Reconhecimento de Padrões em Imagens.....	19
2.2.1 Aquisição da Imagem .....	19
2.2.2 Pré-processamento .....	19
2.2.3 Segmentação .....	20
2.2.4 Extração de Feições .....	20
2.2.5 Classificação .....	21
2.3 Descritores de Forma.....	21
2.3.1 Similaridade Baseada em Contorno.....	22
2.3.2 Similaridade Baseada em Região.....	30
2.4 Classificadores.....	41
2.4.1 Classificadores Estatísticos.....	42
2.4.2 Classificadores Neurais.....	45
2.5 Estado da Arte.....	51
2.5.1 Ferramentas de Processamento de Imagens.....	51
2.5.2 Ferramentas para Recuperação de Informação Visual.....	52
<b>3 DESCRIÇÃO DO TOOLBOX E ANÁLISE DOS DESCRITORES DE FORMA .....</b>	<b>54</b>
<b>4 ESTUDOS DE CASO .....</b>	<b>68</b>
4.1 Estudo de Caso - Comandos .....	69
4.1.1 Localização dos Retângulos.....	71

4.1.2	Isolando Caracteres.....	77
4.1.3	Análise de Variabilidade.....	78
<b>4.2</b>	<b>Estudo de Caso – Placas de Automóveis.....</b>	<b>84</b>
4.2.1	Localização dos Retângulos.....	88
4.2.2	Isolando Caracteres.....	89
4.2.3	Banco de Imagens.....	93
<b>5</b>	<b>ANÁLISE DOS RESULTADOS.....</b>	<b>95</b>
5.1	Comandos.....	95
5.2	Placas de automóveis.....	101
<b>6</b>	<b>CONCLUSÃO.....</b>	<b>106</b>
	<b>REFERÊNCIAS.....</b>	<b>109</b>

## LISTA DE ABREVIATURAS E SIGLAS

ADM	Absolute Difference Mask
ART	Angular Radial Transform
BAS	Beam Angle Statistics
CASCOR	Cascade Correlation
CSS	Curvature Scale Space
CSSI	Curvature Scale Space Image
DF	Descritores de Fourier
DM	Distância Mínima
FFT	Fast Fourier Transform
GDX	Gradiente Descendente com termo de momento e aprendizagem adaptativa
IRENE	Implementação de Redes Neurais
IRIS	Image Retrieval for Information Systems
LM	Levenberg-Marquardt
LVQ	Learning Vector Quantization
MI	Momentos Invariantes
MIA	Momentos Invariantes Afins
MLP	Multilayer Perceptron
MPZ	Momentos Pseudo-Zernike
MVG	Máxima Verossimilhança Gaussiana
MZ	Momentos de Zernike
QBIC	Query By Image Content
RNA	Rede Neural Artificial
RP	Resilient Propagation
RPROP	Resilient Propagation
SCG	Scaled Conjugate Gradient

## LISTA DE FIGURAS

Figura 1.1: Representação de imagens sob diversos aspectos .....	12
Figura 1.2: Exemplo de imagens recuperadas (b), (c) através de um descritor de forma da imagem (a) .....	14
Figura 1.3: Imagens visualmente similares para a percepção humana (a) e (c) e seus respectivos contornos (b) e (d).....	15
Figura 2.1: Etapas do reconhecimento de padrões em imagens .....	19
Figura 2.2: Exemplos de similaridade de forma baseada em contorno e região.....	22
Figura 2.3: Alguns descritores de forma.....	22
Figura 2.4: Exemplos de formas onde um descritor baseado em contorno é aplicável ..	24
Figura 2.5: Um objeto visual 2D e seu respectivo contorno.....	25
Figura 2.6: Uma fronteira digital e sua representação por uma seqüência complexa.....	25
Figura 2.7: Formas originais e sua reconstrução com 2,3,4 e 8 descritores, da esquerda para a direita, respectivamente.....	26
Figura 2.8: Cruzamentos por zero da curvatura.....	27
Figura 2.9: Evolução da forma e correspondente CSSI.....	28
Figura 2.10: Caractere e seu contorno .....	28
Figura 2.11: Diversas evoluções com $N = 64$ .....	29
Figura 2.12: Diversas evoluções com $N = 32$ .....	30
Figura 2.13: Exemplos de várias formas para descritores baseados em região .....	30
Figura 2.14: Experimento para letras.....	34
Figura 2.15: Experimento para formas geométricas .....	35
Figura 3.1: Detalhamento da etapa de Extração de Feições .....	54
Figura 3.2: Formas utilizados no experimento .....	55
Figura 3.3: Apresentação da função “ <i>descriptors</i> ” .....	55
Figura 3.4: Apresentação da função “ <i>extract</i> ” .....	57
Figura 3.5: Extração dos descritores.....	57
Figura 3.6: Os 7 Momentos Invariantes para as formas da Figura 3.2 .....	58
Figura 3.7: Os 10 Descritores de Fourier para as formas da Figura 3.2 .....	58
Figura 3.8: Momentos Invariantes – primeiro agrupamento .....	59
Figura 3.9: Momentos Invariantes – segundo agrupamento.....	59
Figura 3.10: Momentos Invariantes – terceiro agrupamento .....	60
Figura 3.11: Descritores de Fourier – primeiro agrupamento.....	60
Figura 3.12: Descritores de Fourier – segundo agrupamento .....	61
Figura 3.13: Descritores de Fourier – terceiro agrupamento .....	61
Figura 3.14: Célula “amostra” possuindo 9 classes .....	63
Figura 3.15: Atributos da primeira classe da célula “amostra” .....	63

Figura 3.16: Descritores para as primeiras duas imagens da primeira classe.....	63
Figura 3.17: Apresentação da função “ <i>preparaAmostras</i> ”.....	64
Figura 3.18: Apresentação da função “ <i>mlp</i> ”.....	65
Figura 3.19: Apresentação da função “ <i>mmle</i> ”.....	66
Figura 3.20: Apresentação da função “ <i>classifica</i> ”.....	66
Figura 4.1: Imagens utilizadas para o Estudo de Caso dos Comandos.....	71
Figura 4.2: Tarefas para a localização dos retângulos.....	71
Figura 4.3: (a) vizinhança 5x5 de P (b) níveis de cinza.....	72
Figura 4.4: Imagem original.....	73
Figura 4.5: Imagem original após filtro da mediana.....	73
Figura 4.6: Imagem após filtragem gaussiana.....	74
Figura 4.7: Imagem com direções calculadas.....	74
Figura 4.8: Imagem com bordas detectadas e com espessura de 1 pixel.....	75
Figura 4.9: Resultado da localização de <i>end-point</i> com restrições e das respectivas linhas.....	76
Figura 4.10: Retângulos obtidos.....	77
Figura 4.11: Isolando os caracteres do comando DIR.....	78
Figura 4.12: Isolando os caracteres do comando ANDE.....	78
Figura 4.13: Variabilidade das amostras para as classes “A” e “D”.....	79
Figura 4.14: Variabilidade das amostras para as classes “E” e “I”.....	80
Figura 4.15: Variabilidade das amostras para as classes “N” e “P”.....	80
Figura 4.16: Variabilidade das amostras para as classes “Q” e “R”.....	80
Figura 4.17: Variabilidade das amostras para a classe “S”.....	81
Figura 4.18: Variabilidade das amostras.....	81
Figura 4.19: Variabilidade das amostras para as classes “A” e “D”.....	82
Figura 4.20: Variabilidade das amostras para as classes “E” e “I”.....	82
Figura 4.21: Variabilidade das amostras para as classes “N” e “P”.....	82
Figura 4.22: Variabilidade das amostras para as classes “Q” e “R”.....	83
Figura 4.23: Variabilidade das amostras para a classe “S”.....	83
Figura 4.24: Variabilidade das amostras.....	84
Figura 4.25: Placas não utilizadas por impossibilidade de reconhecimento.....	84
Figura 4.26: Placas não utilizadas por problemas na iluminação.....	85
Figura 4.27: Placas comerciais (vermelhas).....	85
Figura 4.28: Placas em escala reduzida.....	86
Figura 4.29: Placas inclinadas e com iluminação não-uniforme.....	86
Figura 4.30: Placas em escala mediana.....	87
Figura 4.31: Placas enfumaçadas.....	87
Figura 4.32: Placas com caracteres desgastados.....	88
Figura 4.33: Placas com sombras e escala reduzida.....	88
Figura 4.34: Etapas para o Isolamento dos Caracteres.....	89
Figura 4.35: Histograma da Lua com limiares L1 e L2.....	90
Figura 4.36: Em (a) limiarização utilizando limiar L1 e em (b) com limiares L1 e L2.....	90
Figura 4.37: Amostras de placas binarizadas.....	91
Figura 4.38: Variação do total de amostras para as letras.....	94
Figura 4.39: Variação do total de amostras para os dígitos.....	94



## LISTA DE TABELAS

Tabela 2.1: Momentos invariantes afins das letras do Experimento 1 .....	35
Tabela 2.2: Momentos invariantes afins das formas do Experimento 2 .....	35
Tabela 2.3: Exemplos de caracteres reconstruídos com os MZ.....	38
Tabela 2.4: Número de momentos entre ordens 0 e 10 para os MZ .....	39
Tabela 2.5: Exemplos de caracteres reconstruídos com os MPZ.....	40
Tabela 2.6: Número de momentos entre ordens 0 e 10 para os MPZ.....	41
Tabela 4.1: Caracteres individuais dos comandos: PARE, ANDE, DIR e ESQ .....	79
Tabela 4.2: Total de Amostras para cada classe .....	93
Tabela 5.1: Classificação para os Momentos Invariantes Afins.....	96
Tabela 5.2: Classificação para os Momentos Invariantes.....	97
Tabela 5.3: Classificação para os Descritores de Fourier .....	98
Tabela 5.4: Classificação para o <i>Curvature Scale Space</i> .....	98
Tabela 5.5: Classificação para os Momentos de Zernike .....	99
Tabela 5.6: Classificação para os Momentos Pseudo-Zernike .....	100
Tabela 5.7: Classificação das letras das placas com RP .....	102
Tabela 5.8: Classificação das letras das placas com SCG .....	102
Tabela 5.9: Classificação dos dígitos das placas com RP.....	103
Tabela 5.10: Classificação dos dígitos das placas com SCG.....	104

## RESUMO

A idéia de capacitar uma máquina a reconhecer o ambiente em que atua tem motivado pesquisadores a investir esforços no estudo do mais complexo dos sentidos humanos, a visão. A visão é, antes de tudo, uma tarefa de representação e processamento de informações, sendo portanto adequada ao tratamento computacional.

Visto que ainda não se possuem métodos que tenham resultados equivalentes ao que seria obtido com um usuário humano, tem-se estudado intensamente a utilização de feições para um melhor aproveitamento de seu potencial. Dentre estas feições, a forma de um objeto proporciona um poderoso indício de sua identidade e funcionalidade, podendo ser utilizada para seu reconhecimento. Isso distingue a forma de outras feições visuais elementares, como a cor, o movimento ou a textura, que, apesar de igualmente importantes, normalmente não revelam a identidade de um objeto. Assim sendo, a possibilidade de avaliar a robustez e a estabilidade de técnicas alternativas para a representação de forma é vital para prever o desempenho de cada técnica na presença de alguma incerteza ou discrepância.

Neste trabalho, alguns descritores de forma descritos na literatura foram implementados e utilizados em estudos de caso para avaliar sua eficácia. Estes estudos de caso foram realizados utilizando-se caracteres, todavia, com finalidades bastante distintas. O primeiro estudo de caso é voltado para aplicações como a robótica móvel, com reconhecimento de comandos localizados no ambiente por parte do robô. Já o estudo de caso principal está direcionado para aplicações de reconhecimento de placas de automóveis, que poderia tanto ser utilizado para monitoramento e controle do fluxo de trânsito, quanto para controle de infrações.

Muitas aplicações, incluindo aquelas que envolvem a recuperação e indexação de objetos visuais, são apropriadas para a utilização de feições de forma.

Outra característica importante do presente trabalho é a de realçar que a seleção de um bom descritor reduz o esforço necessário na etapa de classificação, o qual é computacionalmente elevado.

**Palavras-Chave:** Descritores de Forma, Reconhecimento de Padrões em Imagens, Classificadores Neurais, Classificadores Estatísticos.

# Pattern Recognition in Images via Shape Descriptors

## ABSTRACT

The idea of enabling a machine to recognize the environment with which it interacts has motivated researchers to dedicate efforts in studying the most complex of the human senses: vision. Vision is essentially a task of information representation and processing, what makes it suitable for computational treatment.

Given that currently there are no methods that perform equivalently to humans, the use of features has been intensively studied in order to improve the performance of the existing methods. Among these features, the shape of an object provides a powerful sign of its identity and functionality, what enables the exploitation of this feature with the purpose of recognition. This evidence distinguishes shape from other visual features, such as color, motion or texture, which, although equally important, normally do not reveal the identity of an object. As a result, the possibility of evaluating the robustness and stability of alternate techniques for shape representation is essential in order to measure the performance of each technique in the presence of uncertainty.

In this work, some shape descriptors available in the literature were implemented and used in case studies aiming at evaluating their effectiveness. These case studies were carried out using characters, although, with very different purposes. The first case study is geared towards applications such as mobile robotics, where the robot recognizes commands available in the environment. The main case study is focused on applications of license plate recognition, which could be used both in situations of surveillance and traffic control and in situations of infraction.

Many applications, including those that involve the search and indexing of visual objects, are suited for the use of shape features.

Another important characteristic of this work is that it emphasizes that the selection of a good shape description reduces the effort during the classification step, which is computationally elevated.

**Keywords:** Shape Descriptors, Pattern Recognition in Images, Neural Classifiers, Statistical Classifiers.

# 1 INTRODUÇÃO

O uso de imagens em nossa comunicação é bastante comum, vemos educadores e escritores utilizarem-nas para ilustrações, engenheiros e arquitetos em seus projetos, médicos para efetuar diagnósticos, cineastas para contar histórias, repórteres para realçar informação textual, etc. Além disso, nos últimos anos, o crescimento do número de imagens em meio digital foi espantoso sendo que seu custo de processamento e armazenamento decresceu consideravelmente. A facilidade de distribuição advinda com a popularização da Internet e o aumento do poder computacional foram decisivos para que isso ocorresse.

Diversas são as áreas que contribuem para a geração de imagens digitais. Nestas, podemos citar as áreas de entretenimento, bibliotecas digitais e educação. Nestes casos, o objetivo é armazenar imagens de forma compacta e recuperar os itens relevantes a uma consulta efetuada a um banco de dados/imagens (ZHANG, 2002). Para tal, o conteúdo, ou seja, a informação contida na imagem, deve ser descrita, o que é feito através da extração de feições (*feature extraction*) da imagem.

A Figura 1.1 apresenta os diferentes aspectos sob os quais a informação de uma imagem pode ser representada. Dentro desta subdivisão, existem dois grandes grupos, sendo que o primeiro deles é formado pela comunidade da área de banco de dados, para os quais, toda a informação da imagem é descrita textualmente. O outro grupo é composto pela comunidade das áreas de processamento de imagens digitais, visão computacional, reconhecimento de padrões e áreas afins que pretendem descrever a imagem baseado na informação nela contida.

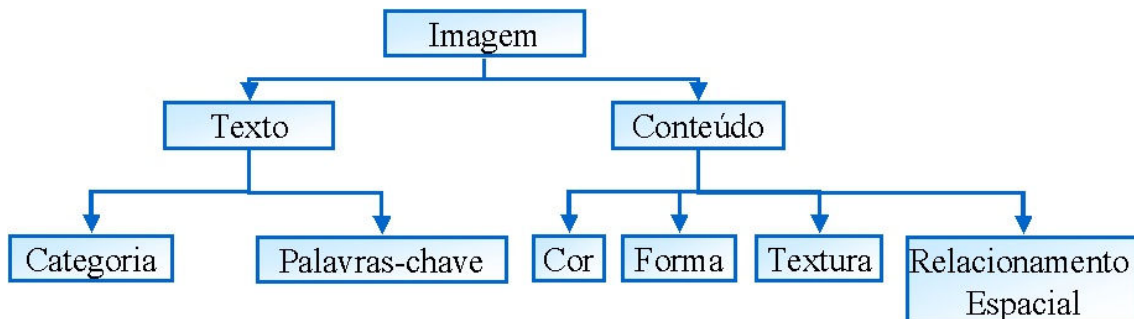


Figura 1.1: Representação de imagens sob diversos aspectos

A descrição da imagem por meio de palavras (textual), obviamente, não é eficiente pois existem diversas formas de descrever a mesma imagem, característica da

subjetividade presente na operação. Além disso, acentua-se o fato de que normalmente quem está recuperando a informação não é a mesma pessoa que realizou o armazenamento. Para realçar ainda mais a subjetividade, pode-se imaginar casos onde a descrição da imagem armazenada foi realizada pelos objetos nela presentes e a busca é realizada pelo que a imagem representa (imagem de pôr-do-sol, por exemplo). Assim, fica evidente, que o nível de detalhamento pode variar entre os operadores do sistema causando resultados indesejados.

O grupo de reconhecimento de padrões, denominação empregada no presente trabalho para caracterizar todas as comunidades anteriormente referidas, busca realizar a descrição baseado em características próprias da imagem, através de meios computacionais, evitando assim a subjetividade causada por um operador humano. Desta maneira, todas as imagens armazenadas possuiriam o mesmo nível de detalhamento.

Formalmente, o reconhecimento de padrões é definido como o processo pelo qual um padrão/sinal recebido é atribuído a uma classe dentre um número pré-determinado de classes (HAYKIN, 2001).

Sistemas de reconhecimento de padrões geralmente consideram um espaço de feições dentro do qual um vetor de observação é mapeado. Com este vetor de observação mapeado, pode-se verificar a qual classe ele pertence. O propósito da extração de feições é a redução da quantidade de dados através da observação de certas feições ou propriedades que distinguem os padrões de entrada. Na extração de feições transforma-se um vetor de observação em um vetor de feições empregando alguma função ortogonal ou não-ortogonal, de modo a se obter um espaço de características não correlacionado (KULKARNI, 1994).

Para realizar tal tarefa feições como cor, forma, textura e relacionamentos espaciais são utilizadas. Existem inúmeros métodos de caracterizar a imagem baseado em cada uma destas feições sendo que a escolha depende da natureza das imagens (aplicações/domínios).

Feições de imagens incluem feições primitivas e feições lógicas. As feições primitivas tais como cor, textura e forma são quantitativas por natureza, podendo ser extraídas automaticamente ou semi-automaticamente. Feições lógicas são qualitativas por natureza e proporcionam representações resumidas de dados visuais em vários níveis de detalhe sendo, tipicamente, extraídas manualmente ou semi-automaticamente. Uma ou mais feições podem ser utilizadas em uma aplicação específica. Por exemplo, em um sistema de informação por satélite, as feições de textura são importantes, enquanto forma e cor são feições mais importantes em sistemas de registro de logomarcas.

Naturalmente, um melhor desempenho em qualquer das aplicações citadas pode ser alcançado com a utilização de uma combinação de descritores de feições visuais. A Figura 1.2 (HÖYNECK; OHM, 2003) é utilizada para ilustrar mais adequadamente os resultados da utilização de apenas uma feição para a recuperação de imagens. Em (a), a referida figura apresenta uma imagem de onde são extraídas feições de forma, com base no contorno, e em (b) e (c) possíveis resultados do processo de recuperação. Fica claro que a adição de feições adicionais como cor ou textura poderia reduzir o número de respostas indesejadas como as obtidas neste exemplo. Este assunto, entretanto, não está no escopo desta dissertação podendo fazer parte de uma possível evolução do presente trabalho.

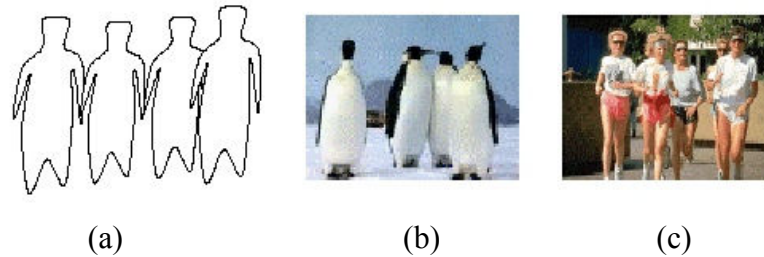


Figura 1.2: Exemplo de imagens recuperadas (b), (c) através de um descritor de forma da imagem (a)

Cabe salientar, que a extração de feições é apenas um dos elementos chave das pesquisas nesta área, que quando implementada computacionalmente, faz uso de técnicas estudadas em disciplinas como: processamento de imagens digitais, recuperação de informação visual, reconhecimento de padrões, estatística e inteligência artificial, caracterizando assim, sua forte interdisciplinariedade.

## 1.1 Motivação

A idéia de capacitar uma máquina a reconhecer o ambiente em que atua tem motivado pesquisadores a investir esforços no estudo do mais complexo dos sentidos humanos, a visão. A visão é, antes de tudo, uma tarefa de representação e processamento de informações, sendo portanto adequada ao tratamento computacional. Marr define: “Visão é o processo que produz, através de imagens do mundo externo, uma descrição que é útil ao observador e não sobrecarregada por informações irrelevantes” (1982, p.31).

Visto que ainda não se possuem métodos que tenham resultados equivalentes ao que seria obtido com um usuário humano, tem-se estudado intensamente a utilização de feições para um melhor aproveitamento de seu potencial. Dentre estas, a forma de um objeto proporciona um poderoso indício de sua identidade e funcionalidade, podendo ser utilizada para seu reconhecimento. Humanos podem reconhecer objetos característicos através de sua forma, prova de que a forma possui informação semântica. Isso distingue a forma de outras feições visuais elementares, como a cor, o movimento ou a textura, que, apesar de igualmente importantes, normalmente não revelam a identidade de um objeto (BOBER, 2001).

Muitas aplicações, incluindo aquelas que envolvem a recuperação e indexação de objetos visuais, são apropriadas para a utilização de feições de forma. Por exemplo, um sistema inteligente de segurança em vídeo pode utilizar a forma para determinar a identidade do intruso. O escopo de aplicações de um robô pode ser consideravelmente expandido dotando-o de um sistema de visão, um dos principais sentidos humanos, o que o faria responder ao ambiente de forma mais inteligente e flexível.

Dentre as aplicações que este trabalho visa auxiliar estão aquelas onde se faz necessário o reconhecimento de caracteres. Uma destas aplicações é o reconhecimento de placas de automóveis.

Segundo Campos (2001), na cidade de Porto Alegre – RS, um dispositivo eletrônico de controle de velocidade, como o pardal, aplica mais de mil multas diárias, sendo que o processo de reconhecimento das placas destes veículos é realizado manualmente por pessoas responsáveis pela análise destas imagens. Um sistema desta natureza pode ser desenvolvido com a utilização de feições de forma. Além disso, este sistema pode servir

de base a outras aplicações como, por exemplo, a navegação de um robô autônomo em um ambiente composto de salas identificadas. O robô poderia ser enviado de uma sala a outra buscando ou levando informações. Este mesmo robô poderia ser utilizado para detecção de intrusos com a utilização de sensores de movimento, onde poderia verificar em qual sala está o intruso e então alertar um serviço de segurança. Diversas são as aplicações, sendo que as aqui apresentadas são apenas uma pequena amostra do que é possível.

As feições são a representação de uma imagem através de um vetor numérico (vetor de feições) que tem por objetivo reduzir a redundância dos dados e também sua dimensionalidade. As feições extraídas devem possuir grande parte da informação necessária para a discriminação do conteúdo da imagem e, para isso, é desejável que apresentem pequena variação intra-classe e grande variação inter-classe. Desta forma, a realização adequada desta etapa é de suma importância para que a próxima etapa, ou seja, a classificação, alcance os índices de reconhecimento desejados. Quanto mais discriminantes forem as feições extraídas, menor será o esforço computacional requerido pelo classificador e maior será a taxa de reconhecimento alcançada.

A extração de feições quando tratada sob a ótica das feições de forma recebe a denominação de “*descritores de forma*”, denominação esta, empregada neste trabalho. Os descritores de forma são agrupados pelo critério de similaridade adotado, podendo ser baseados em contorno ou região.

Descritores de forma com similaridade baseada em contorno, como os Descritores de Fourier e os métodos baseados em curvatura, não são adequados para formas complexas que consistem de diversas regiões desconexas como logomarcas, emblemas, cenas da natureza, etc. O contorno pode sofrer mudanças drásticas caso exista uma pequena abertura ou caso ele toque em algum objeto próximo. Isto pode ser visto nas formas apresentadas nas imagens da Figura 1.3(a) e (c) que são muito similares para a percepção humana. Entretanto, seus contornos, mostrados na Figura 1.3(b) e (d), são muito diferentes quando a estrela interna toca ou não toca o círculo externo (KIM; KIM, 2000). Nesta figura, o objeto (b) possui o dobro de contornos que (d), o que os torna bastante distintos segundo este descritor. Além disso, muitos descritores de forma não são apropriados para descrever formas que sejam constituídas de regiões desconexas sendo então, mais adequados a imagens constituídas de um único contorno.

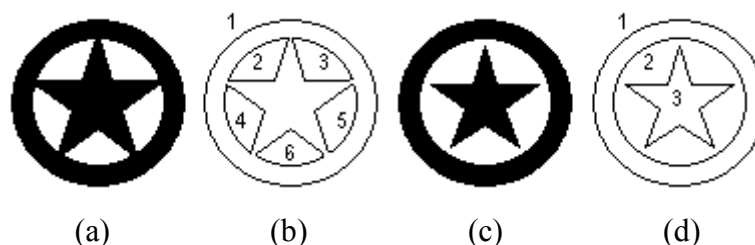


Figura 1.3: Imagens visualmente similares para a percepção humana (a) e (c) e seus respectivos contornos (b) e (d)

Descritores de forma com similaridade baseada em região, como os momentos, são mais adequados para formas que possuam contornos complexos, pois operam sobre todos os pixels da imagem, independentemente se estes estão no contorno ou no interior da imagem.

A escolha da melhor maneira de se descrever imagens, ou seja, por descritores baseados em contorno ou região, não é trivial, demandando uma série de testes para avaliação do desempenho de cada abordagem sobre a aplicação específica considerada.

Sabendo-se disso, o desenvolvimento de um *toolbox* Matlab para a extração destas feições seria bastante útil na investigação de técnicas alternativas para a resolução de problemas de diversas aplicações. A técnica pode ser avaliada e, então, após a verificação de sua eficácia, ser implementada numa linguagem e/ou plataforma específica como, por exemplo, o controlador de um robô.

Assim, propõe-se a extração de feições como um processo iterativo onde toma-se a decisão sobre qual técnica é mais promissora baseado nos experimentos realizados com os descritores de forma disponibilizados. Os descritores que obtiverem resultados mais significativos indicam, por exemplo, qual critério de similaridade deve ser empregado para a aplicação estudada. Esta decisão, por um critério de similaridade, pode nortear o rumo das futuras pesquisas. Desta forma, seria também interessante que o *toolbox* possuísse código aberto e caso não se alcancem resultados satisfatórios com a aplicação de suas técnicas, pode-se desenvolver outras com base nas já existentes, reduzindo assim, o esforço nesta etapa. O tempo ganho com a utilização do *toolbox* permite então que mais experimentos sejam realizados, que mais técnicas sejam avaliadas e também que as outras etapas do sistema sejam aprimoradas.

A escolha do ambiente de desenvolvimento Matlab deve-se ao fato de este possuir uma linguagem orientada a matrizes, proporcionando uma enorme facilidade na manipulação de imagens, as quais nada mais são do que visualização de matrizes. Com isso, tem-se uma maneira muito simples e rápida para expressar operações de processamento de imagens. Além disso, ele já possui um *toolbox* para o Processamento de Imagens que, por exemplo, disponibiliza operações de filtragem, equalização de histograma, detecção de bordas, plotagem de gráficos, etc.

## 1.2 Objetivo do Trabalho

Conforme foi apresentado, muitas são as motivações para este trabalho o que possibilita seu emprego em muitas aplicações, evidenciando sua potencialidade.

O objetivo principal deste trabalho é o desenvolvimento de um *toolbox* Matlab com funções para a extração de feições de forma, que possa ser utilizado como ferramenta para o desenvolvimento rápido de outras aplicações e validação de técnicas para novos domínios. Este *toolbox* pode ser utilizado por outras ferramentas de extração de feições onde poderão ser agregados novos algoritmos, por exemplo, para textura e cor, facilitando a pesquisa e o estudo dos interessados.

Para demonstrar a aplicabilidade das funções desenvolvidas este *toolbox* foi empregado em dois estudos de caso.

O primeiro estudo de caso foi motivado pela tese de doutorado de Rolf Fredi Molz (MOLZ, 2001), onde necessitava-se avaliar métodos de extração de feições de forma para reconhecimento de comandos escritos. Neste experimento, objetiva-se que os comandos **pare**, **ande**, **direita** e **esquerda**, que estão dispostos em retângulos com diferentes escalas e graus de rotação, sejam localizados automaticamente em um ambiente e então, após o processamento seus caracteres sejam reconhecidos. Esta é uma aplicação que poderia ser utilizada num sistema robótico autônomo.



Um segundo estudo de caso, envolve o reconhecimento dos caracteres de placas de automóveis. Neste estudo a localização das placas será realizada de forma manual e então, os métodos que obtiveram resultados satisfatórios no primeiro estudo de caso serão aplicados na tarefa de reconhecimento.

### **1.3 Organização do Trabalho**

O trabalho encontra-se organizado da seguinte maneira:

O Capítulo 1 é a presente introdução que apresenta uma visão geral do trabalho desenvolvido.

O Capítulo 2 apresenta os principais conceitos de reconhecimento de padrões em imagens, processamento de imagens, descritores de forma e classificadores que fazem parte dos conhecimentos necessários para o bom entendimento deste trabalho. Como as áreas são muito amplas, procurou-se direcionar o assunto para os tópicos de maior relevância.

No Capítulo 3 é feita, de uma forma geral, a descrição do *toolbox* desenvolvido neste trabalho. A diferença existente entre descritores com similaridade baseada em região e contorno é evidenciada através da demonstração das rotinas desenvolvidas.

O *toolbox* é avaliado através dos estudos de caso no Capítulo 4. Dois estudos de caso visando aplicações distintas são detalhados.

A análise dos resultados obtidos com os estudos de caso está descrita no Capítulo 5.

Por fim, o Capítulo 6 apresenta as considerações finais sobre o desenvolvimento do trabalho, suas contribuições e sugestões para trabalhos futuros.

## 2 RECONHECIMENTO DE PADRÕES EM IMAGENS

O Reconhecimento de Padrões em Imagens Digitais é uma área em constante evolução na Ciência da Computação, com muitas áreas de aplicação em várias ciências e na engenharia. Tais áreas incluem o sensoriamento remoto, transmissão e armazenamento de imagens para aplicações na área de negócios, processamento de imagens médicas, radar, sonar, robótica, inspeção automática de componentes industriais, entre outros (JAIN; DUIN; MAO, 2000; PICOLLI, 1999).

Cada uma dessas áreas requer soluções diferentes, sendo, dessa forma, importante o conhecimento sobre imagens digitais e também sobre as etapas que compõem um sistema de reconhecimento de padrões em imagens digitais.

### 2.1 Imagem Digital

Uma imagem pode ser descrita por uma função bidimensional de intensidade da luz  $f(x,y)$ , sendo seu valor, em qualquer ponto de coordenadas espaciais  $(x,y)$ , proporcional ao brilho da imagem naquele ponto (GONZALEZ; WOODS, 2000). A função  $f(x,y)$  descreve um conjunto bidimensional de pontos  $M \times N$ , os quais representam os elementos da imagem (*picture elements* ou pixels).

Existem diferentes meios de representação de imagens digitais, que dependem da natureza da imagem. Normalmente as imagens são classificadas como binárias, em níveis de cinza ou coloridas.

Em imagens binárias os pixels podem assumir apenas dois valores, 0 (preto) ou 1 (branco). Imagens com esta representação apresentam a vantagem de necessitar de pouco espaço para seu armazenamento, sendo que, além disso, facilitam o emprego de operadores lógicos (ex.: XOR, AND e OR).

As imagens em níveis de cinza, ou imagens monocromáticas, possuem pixels que assumem valores entre 0 (preto) e  $N$  (branco). Todos os outros valores intermediários serão tons de cinza. Geralmente,  $N+1$  é uma potência de dois; no caso de ser 256, necessitará de 8 bits (1 byte) para cada pixel sendo suficiente para representar todas as tonalidades que o olho humano é capaz de distinguir (LIBERMAN, 1997).

Por sua vez, as imagens coloridas podem ser representadas de duas formas: através de uma tabela de cores ou pelo modelo RGB (*Red, Green e Blue*). No modelo de tabela, existem códigos para cada cor, organizados em forma de tabela. Cada pixel da imagem representa um índice da tabela, também denominada *palette*. Já no modelo RGB, cada pixel ocupa 24 bits (3 bytes), onde são representadas as intensidades de vermelho, verde

e azul do pixel. Existem outros modelos de cores, como o HSI (*Hue, Saturation e Intensity*), o XYZ, o CMY (*Cyan, Magenta e Yellow*),  $L^*u^*v$ ,  $L^*a^*b$ , dentre outros (BIMBO, 1999).

## 2.2 Etapas do Reconhecimento de Padrões em Imagens

O reconhecimento de padrões é a parte essencial de qualquer sistema de análise de imagens de alto-nível. Para reduzir a dimensão e a redundância dos dados, um conjunto de valores numéricos, conhecidos como feições, é extraído das imagens e utilizado para representá-las. Um sistema de reconhecimento flexível deve ser capaz de reconhecer um objeto independentemente de sua orientação, tamanho e localização no campo de visão. Estes requisitos são obtidos com a extração de feições invariantes a rotação, escala e translação (KHOTANZAD; LU, 1989). Muitos destes sistemas compartilham uma estrutura geral composta de 5 etapas, representadas abaixo, na Figura 2.1:

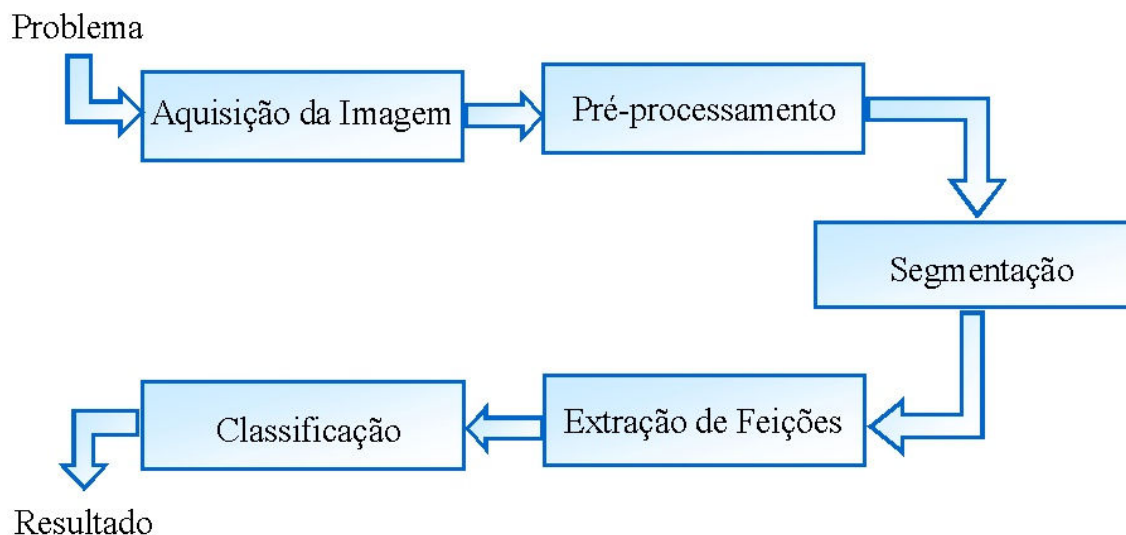


Figura 2.1: Etapas do reconhecimento de padrões em imagens

### 2.2.1 Aquisição da Imagem

O primeiro estágio de um sistema de reconhecimento de padrões em imagens é o processo de aquisição. Durante esta etapa, sensores ópticos são responsáveis pela captura de sinais emitidos por fontes de luz. Paralelamente, estes sinais são digitalizados por conversores AD (Analogico/Digital), armazenados e processados pelas etapas subsequentes.

Existem diversos equipamentos com a função de capturar imagens, como por exemplo, câmeras de vídeo e *scanners*. Estes aparelhos possuem conjuntos de sensores, que capturam as imagens de interesse. A captura é realizada de acordo com uma determinada taxa de amostragem que determina a resolução da imagem. De acordo com a necessidade e finalidade, as imagens capturadas são então convertidas para um formato de interesse e devidamente armazenadas (GONZALEZ; WOODS, 2000; JAIN, 1989).

### 2.2.2 Pré-processamento

Pré-processamento é o nome usado para operações em imagens com baixo nível de abstração, onde a entrada e a saída são imagens (SONKA; HLAVAC; BOYLE, 1999).

Esta etapa é opcional e objetiva corrigir imperfeições e defeitos incorporados à imagem durante a aquisição. As causas destas imperfeições podem ser diversas, desde características do sistema até falhas no processo de aquisição. Sendo assim, filtragem de ruídos, aumento de contraste e até mesmo operações de rotação e escalamento podem ser considerados exemplos de pré-processamento. A realização adequada desta etapa aumenta as chances de sucesso dos processos seguintes.

### **2.2.3 Segmentação**

Nesta etapa, a imagem é considerada do ponto de vista da informação nela presente, podendo ser entendida como o particionamento da mesma em regiões que apresentam propriedades semelhantes.

A detecção destas regiões pode ser feita de dois modos: através do agrupamento de pontos vizinhos com características semelhantes, ou através da determinação da borda da região. Além disso, a detecção de regiões pode ser feita visando um dos seguintes objetivos: extrair uma determinada região ou particionar a imagem num conjunto de regiões disjuntas cuja união representa a imagem inteira.

As partes nas quais deseja-se segmentar a imagem geralmente são distinguíveis com base nos níveis de cinza ou propriedades de cor dos pixels individuais que compõem estas partes. Por exemplo, em uma imagem de caracteres impressos ou escritos, os pixels pertencentes aos caracteres normalmente apresentam-se mais escuros do que aqueles pertencentes ao fundo. Além disso, deve-se considerar os relacionamentos espaciais dos pixels.

### **2.2.4 Extração de Feições**

Nesta etapa busca-se alguma informação quantitativa de interesse ou que seja básica para a discriminação entre classes de objetos (GONZALEZ; WOODS, 2000). Para isso, é necessário escolher a forma de representação dos dados, que pode ser por fronteiras (externa) ou por regiões completas (interna).

A representação por fronteira é adequada quando o interesse é baseado em características externas, tais como cantos e pontos de inflexão (*zero-crossings*). A representação por regiões é adequada quando o interesse está baseado em características internas (os pixels que compõem a região), como parâmetros estatísticos, cor e textura. Nada, entretanto, inviabiliza a utilização das duas formas de representação conjuntamente, isso ficando a cargo da tarefa a ser realizada.

Assim que for tomada a decisão sob qual das formas de representação será empregada, deve-se decidir quais serão as feições utilizadas para tal tarefa. Por exemplo, uma região pode ser representada por sua fronteira, com esta última sendo descrita por feições tais como tamanho e número de concavidades da fronteira.

Geralmente, uma representação externa é escolhida quando a atenção primária estiver voltada para feições de forma. Por outro lado, uma representação interna é selecionada quando a atenção estiver voltada para propriedades como cor ou textura. Em qualquer um dos casos, as feições selecionadas como descritores devem ser afetadas o menos possível por variações como mudanças de tamanho, rotação e translação.

Em outras palavras, na extração de feições, a imagem é representada como um vetor numérico (vetor de feições) que tem por objetivo diminuir a redundância dos dados e também efetuar a redução de dimensionalidade. Estas feições devem possuir grande parte da informação realmente útil para a discriminação. Assim, a extração de boas

feições é uma etapa crucial para que a próxima etapa (classificação) atinja seu objetivo conforme esperado. Boas feições são aquelas que satisfazem dois requisitos:

- pequena variação intra-classe – formas ligeiramente diferentes mas com características gerais similares devem possuir valores próximos;
- grande variação inter-classe – feições de diferentes classes devem possuir valores bastante diferentes.

Além disso, um sistema de reconhecimento flexível deve reconhecer um objeto independente de sua orientação, tamanho e localização. Estes requisitos são traduzidos em propriedades invariantes a rotação, escala e translação.

### **2.2.5 Classificação**

A classificação é a parte mais abstrata do processo de visão por computador. Ela representa o “alto nível” e permite obter a compreensão e a descrição final da imagem analisada. A classificação parte da premissa que a similaridade entre objetos implica que eles possuam características similares, formando classes/agrupamentos. O resultado da classificação pode ser percentual (indicando o % de chance da ocorrência de alguma classe) ou também pode ser uma imagem com algumas características enfatizadas para auxiliar o especialista em sua tomada de decisão. De outra maneira, podemos considerar que a fase de classificação consiste em reconhecer um objeto, uma forma ou, de modo geral, uma entidade particular da imagem. Dado um conjunto de classes e um padrão apresentado como entrada para o sistema, o problema consiste em decidir a que classe o padrão pertence. Deve haver a alternativa de rejeição do padrão (DAHMER, 1998; GONZALEZ; WOODS, 2000; LIBERMAN, 1997; WHELAN; MOLLOY, 2001).

A seguir serão apresentadas em mais detalhes as etapas de extração de feições (através dos descritores de forma) e classificação que fazem parte do *toolbox* desenvolvido.

## **2.3 Descritores de Forma**

A noção de forma do objeto, apesar de intuitivamente clara, pode ter muitos significados. Muitos dos objetos do mundo real são 3D, entretanto, imagem e vídeo normalmente contêm projeções 2D dos mesmos (BOBER, 2001).

No caso 2D, existem duas noções de similaridade que são: baseada em região e baseada em contorno. Isto pode ser visto na Figura 2.2. Objetos na primeira linha possuem uma distribuição espacial similar dos pixels, sendo semelhantes de acordo com critérios baseados em região. Entretanto, possuem contornos diferentes. Quando a similaridade baseada em contorno é empregada, os objetos mostrados em cada coluna são similares. Caso fosse efetuada uma consulta a partir de uma imagem exemplo tal como a localizada na primeira linha e primeira coluna, as imagens consideradas semelhantes seriam as da primeira linha (se a similaridade fosse baseada em região) ou da primeira coluna (se a similaridade fosse baseada em contorno) (BOBER, 2001).

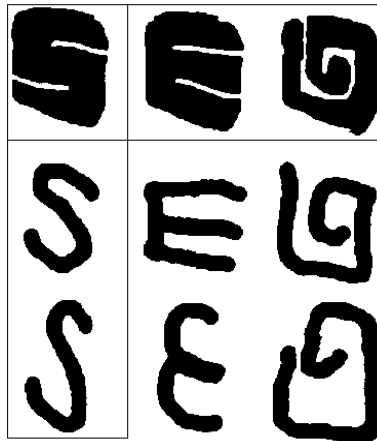


Figura 2.2: Exemplos de similaridade de forma baseada em contorno e região

Faz-se necessário comentar a necessidade da busca de métodos que possuam invariância a mudanças de orientação (rotação e translação) bem como a variações de escala, pois, por exemplo, nas aplicações consideradas, objetos semelhantes mas com tamanhos diferentes devem ser considerados similares independente de seu tamanho. Assim sendo, são considerados métodos eficazes aqueles que possuem tais características aliadas também à robustez em relação a ruído.

Na Figura 2.3 alguns destes descritores são apresentados. Cabe salientar que os métodos apresentados nesta figura não são os únicos existentes, sendo apenas uma pequena porção dentre todas as possibilidades. Para um estudo mais aprofundado ou para verificar a diversidade de métodos existentes, podem ser consultados os artigos de (MEHTRE; KANKANHALLI; LEE, 1997; SAFAR; SHAHABI; SUN, 2000).

Neste trabalho, vários descritores de forma serão apresentados, principalmente os que fazem uso de similaridade baseada em região. Um grande número de descritores desta natureza está descrito nos artigos clássicos de (TEH; CHIN, 1988; PROKOP; REEVES, 1992).

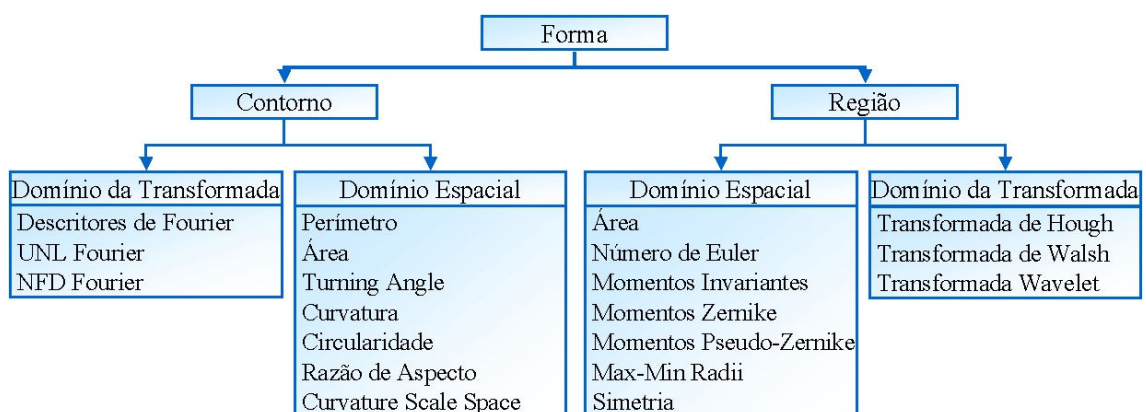


Figura 2.3: Alguns descritores de forma

### 2.3.1 Similaridade Baseada em Contorno

Os descritores baseados em contorno expressam as propriedades das formas pelo seu esboço (contorno).

As propriedades de forma do contorno são importantes para recuperação de objetos semanticamente similares. Busca-se em um descritor desta categoria, segundo a especificação MPEG-7, que ele seja muito eficiente em aplicações onde uma alta variabilidade na forma é esperada, devido por exemplo a deformações no objeto. A descrição deve ser robusta na presença de ruído no contorno. A seguir, uma lista com algumas características importantes buscadas em descritores de contorno:

- Na Figura 2.4 (a) todos os objetos seriam distinguidos eficientemente por um descritor desta natureza pois as características das formas estão contidas no contorno;
- Capacidade de distinguir entre formas que possuam propriedades de forma por região semelhantes mas diferentes propriedades da forma do contorno. Os objetos apresentados na Figura 2.4 (b) possuem uma distribuição de pixels por região similar, mas diferentes propriedades de contorno e, assim, um descritor de contorno pode eficientemente diferenciar entre cada forma;
- Deve ser capaz de realizar buscas para formas que são semanticamente similares para humanos, onde uma significativa variabilidade intra-classe existe. A Figura 2.4 (c) demonstra esta capacidade, onde as imagens deveriam ser consideradas similares;
- Robustez a deformações não-rígidas significantes, como as encontradas na Figura 2.4 (d);
- Robusto a distorções no contorno ocorridas por transformações perspectivas, as quais são comuns em imagens e vídeos Figura 2.4 (e).

As informações apresentadas acima podem ser resumidas da seguinte maneira: um descritor de contorno eficiente deve ser capaz de distinguir as imagens da Figura 2.4 (a) e (b), dentro de cada grupo; e deve considerar similares, apesar da variação intra-classe, as imagens da Figura 2.4 (c), (d) e (e).

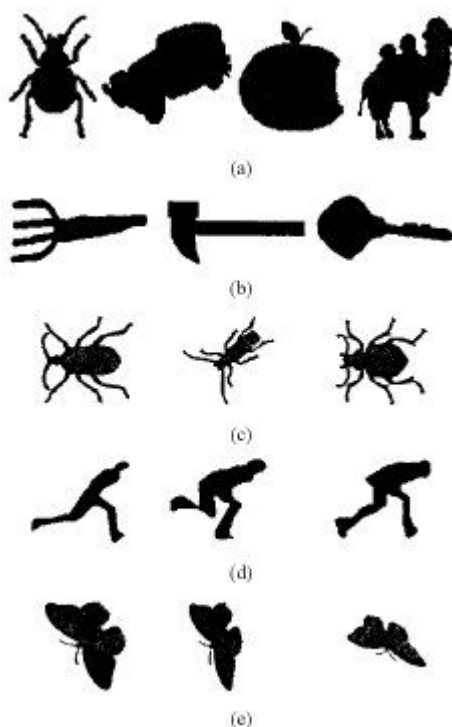


Figura 2.4: Exemplos de formas onde um descritor baseado em contorno é aplicável

A seguir os dois métodos baseados em contorno selecionados para este trabalho serão apresentados. O método dos Descritores de Fourier é tido como o método clássico, possuindo invariância a rotação e a escala. Além deste, o método padronizado para o MPEG-7, conhecido como *Curvature Scale Space* (CSS) também será descrito. Estes métodos permitem a visualização de algumas de suas características, através da reconstrução da imagem nos Descritores de Fourier (Figura 2.7) e através da geração das imagens com a assinatura das figuras do CSS (Figura 2.9), imagens estas conhecidas como *CSS Images* (CSSI). Na Figura 2.5, pode-se verificar a representação de uma imagem para descritores de forma baseados em contorno, onde é necessária a extração de um contorno fechado do objeto. Essa tarefa de segmentação é realizada de maneira semi-automática, na maioria dos casos.

É de suma importância salientar que a segmentação é essencial para a utilização deste tipo de descritor, havendo a necessidade de extrair um contorno fechado. Assim, de modo geral, a tarefa deve ser assistida por um operador humano que fará as correções necessárias para que o contorno mantenha sua integridade. Na imagem da Figura 2.5, por exemplo, se o método de segmentação não possuísse supervisão, provavelmente, a listra branca próxima a cabeça do peixe seria tomada como uma nova região, alterando completamente a forma final do contorno pois haveria uma abertura cruzando verticalmente quase a totalidade do animal. Isto, entretanto, não faria com o que o contorno pudesse ser caracterizado como aberto, mas causaria uma alteração muito grande no mesmo. O operador então, teria por tarefa unificar as duas regiões de interesse, compostas pela parte branca e pelo restante do peixe, caracterizado pela parte em amarelo.



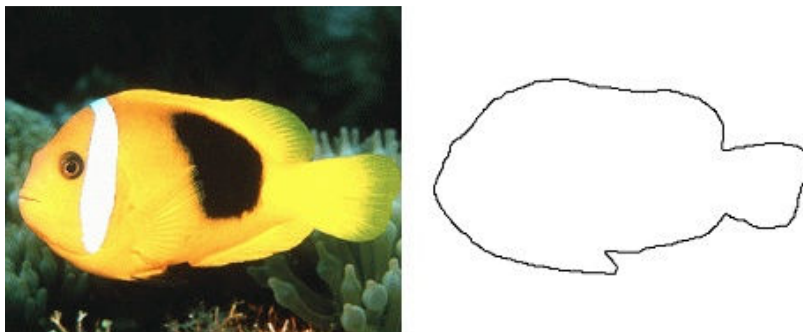


Figura 2.5: Um objeto visual 2D e seu respectivo contorno

### 2.3.1.1. Descritores de Fourier

A Figura 2.6 mostra uma fronteira digital de  $N$  pontos no plano  $xy$ . Começando de um ponto arbitrário  $(x_0, y_0)$ , pode-se encontrar os pares coordenados  $(x_0, y_0)$ ,  $(x_1, y_1)$ ,  $(x_2, y_2)$ , ...,  $(x_{N-1}, y_{N-1})$  ao longo da fronteira (por exemplo) no sentido anti-horário. Essas coordenadas podem ser expressas na forma  $x(k) = x_k$  e  $y(k) = y_k$ . Com essa notação, a própria fronteira pode ser representada como uma seqüência de coordenadas  $s(k) = [x(k), y(k)]$ , para  $k = 0, 1, 2, \dots, N-1$ . Além disso, cada par ordenado pode ser tratado como um número complexo da forma (GONZALEZ; WOODS, 2000; JÄHNE, 1997).

$$s(k) = x(k) + jy(k) \quad (2.1)$$

para  $k = 0, 1, 2, \dots, N-1$ .

Ou seja, o eixo  $x$  é tratado como o eixo real, enquanto que o eixo  $y$  como o eixo imaginário de uma seqüência de números complexos. Embora a interpretação da seqüência tenha sido reformulada, a própria natureza da fronteira não foi mudada. Obviamente, essa representação possui uma grande vantagem: ela reduz um problema de duas dimensões a uma só dimensão.

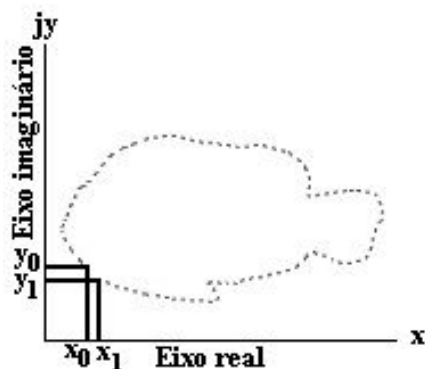


Figura 2.6: Uma fronteira digital e sua representação por uma seqüência complexa

A transformada discreta de Fourier (DFT) de  $s(k)$  é definida como:

$$a(u) = \frac{1}{N} \sum_{k=0}^{N-1} s(k) \exp[-j2\pi uk / N] \quad (2.2)$$

para  $u = 0, 1, 2, \dots, N-1$ . Os coeficientes complexos  $a(u)$  são chamados de *descritores de Fourier* da fronteira e possuem invariância a rotação e a escala.

A transformada inversa de Fourier de  $a(u)$  reconstrói  $s(k)$ , ou seja:

$$s(k) = \sum_{u=0}^{N-1} a(u) \exp[j2\pi uk / N] \quad (2.3)$$

para  $k = 0, 1, 2, \dots, N-1$ .

Suponha, entretanto, que no lugar de todos os valores  $a(u)$ , apenas os primeiros  $M$  coeficientes sejam usados. Isso é equivalente a fazer  $a(u) = 0$  para  $u > M-1$  na Equação (2.3). O resultado é a seguinte aproximação de  $s(k)$ :

$$\hat{s}(k) = \sum_{u=0}^{M-1} a(u) \exp[j2\pi uk / N] \quad (2.4)$$

para  $k = 0, 1, 2, \dots, N-1$ .

Embora apenas os  $M$  termos sejam usados na obtenção de cada componente, a quantidade de pontos da imagem reconstruída permanece sendo  $N$ . Ou seja, o mesmo número de pontos existe na fronteira de aproximação, mas não mais tantos termos são usados na reconstrução destes pontos. Se o número de pontos da fronteira for grande, então  $M$  é geralmente selecionado como sendo uma potência de 2, de maneira que um algoritmo de FFT (*Fast Fourier Transform*) possa ser usado no cálculo dos descritores. Cabe salientar que na transformada de Fourier os componentes de alta frequência geralmente estão relacionados a detalhes finos, enquanto que os componentes de baixa frequência determinam a forma global. Portanto, quanto menor for  $M$ , mais detalhes são perdidos na fronteira. Isso pode ser constatado na Figura 2.7 (JÄHNE, 1997).

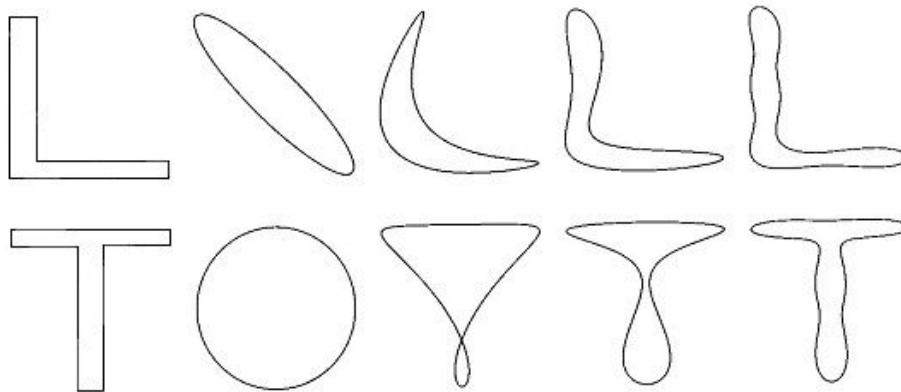


Figura 2.7: Formas originais e sua reconstrução com 2,3,4 e 8 descritores, da esquerda para a direita, respectivamente

### 2.3.1.2. Curvature Scale Space

Mokhtarian e Mackworth (1992) propuseram um método confiável e bastante rápido para a recuperação de formas por similaridade em grandes bases de dados chamado *Curvature Scale Space* (CSS). Este método é muito robusto em relação a ruído, mudanças de escala e orientação de um objeto. O CSS foi recomendado como o padrão para descritores de forma baseados em contorno para o MPEG-7 e sua representação é extremamente compacta, cerca de 14 bytes em média.

Neste trabalho, o método foi empregado para a tarefa de classificação, ao invés de recuperação como originalmente projetado.

Para criar uma descrição CSS de uma imagem, deve-se obter o contorno fechado desta imagem, como o apresentado na Figura 2.5. Para criar a descrição CSS de uma forma,  $N$  pontos equidistantes são selecionados do contorno, iniciando em um ponto arbitrário e seguindo no sentido anti-horário. Esses pontos são escalonados pelo comprimento do arco para um intervalo entre 0 e 1. O cálculo do comprimento do arco, nada mais é do que a geração de um índice para cada coordenada do contorno. Em outras palavras, partindo-se da primeira posição do contorno até a última, verifica-se se as coordenadas das posições subseqüentes estão na mesma linha ou coluna, caso estejam o comprimento do arco para esta posição é 1, caso contrário é  $\sqrt{2}$ . Esse processo é feito para todas as posições do contorno. Feito isso, utiliza-se esse vetor contendo o valor do comprimento do arco para cada posição do contorno para gerar, através de interpolação, um novo contorno com  $N$  pontos, gerando assim, novas coordenadas do contorno, desta vez entre 0 e 1. Isso é realizado em todas as imagens, com o intuito de que todas possuam um contorno com a mesma quantidade de pontos o que proporciona ao método a propriedade de invariância à escala.

As novas coordenadas de  $x$  e  $y$  são armazenadas em dois conjuntos  $X$  e  $Y$  que são empregados no cálculo da curvatura, conforme equação abaixo:

$$K(j,k) = \frac{X_u(j,k)Y_{uu}(j,k) - X_{uu}(j,k)Y_u(j,k)}{(X_u(j,k)^2 + Y_u(j,k)^2)^{1,5}} \quad (2.5)$$

onde  $X_u(j,k), Y_u(j,k)$  são a derivada primeira de  $X$  e  $Y$  respectivamente e  $X_{uu}(j,k), Y_{uu}(j,k)$  são a sua derivada segunda.

Todas as inflexões (cruzamentos por zero – Figura 2.8) obtidas através da Equação (2.5) devem ser armazenadas para a geração de uma imagem que servirá de assinatura da forma, essa imagem é conhecida como *CSS Image (CSSI)*.



Figura 2.8: Cruzamentos por zero da curvatura

O contorno do objeto, então, é gradualmente suavizado por repetidas aplicações de um filtro passa-baixas (0,25; 0,5; 0,25) nas coordenadas de  $X$  e  $Y$ . A esse processo dá-se o nome de evolução da forma, o qual acarreta o arredondamento da imagem. Para cada nível de evolução, calcula-se a curvatura novamente e armazenam-se as inflexões.

A CSSI possui em sua base o índice das coordenadas do contorno reamostrado, contendo  $N$  pontos, correspondendo à ordem do percurso sobre o contorno. A ordenada da CSSI representa a iteração do processo de evolução da forma. A CSSI registra os pontos de inflexão encontrados durante as iterações do processo de evolução da forma. Assim, para cada conjunto (iteração, posição no contorno), calcula-se a curvatura e verifica-se se houve uma inflexão, caso exista, este ponto é marcado na CSSI. Ao final, têm-se a formação dos picos característicos da CSSI que nada mais são do que as inflexões ocorridas durante o processo de evolução nas múltiplas escalas. É importante salientar que a imagem mesmo após várias iterações sempre possui a mesma quantidade

de pontos sendo, desta forma, possível acompanhar o que ocorre em cada posição do contorno durante todo o processo.

A evolução da forma de um objeto e sua respectiva CSSI em vários níveis de evolução podem ser vistas na Figura 2.9.

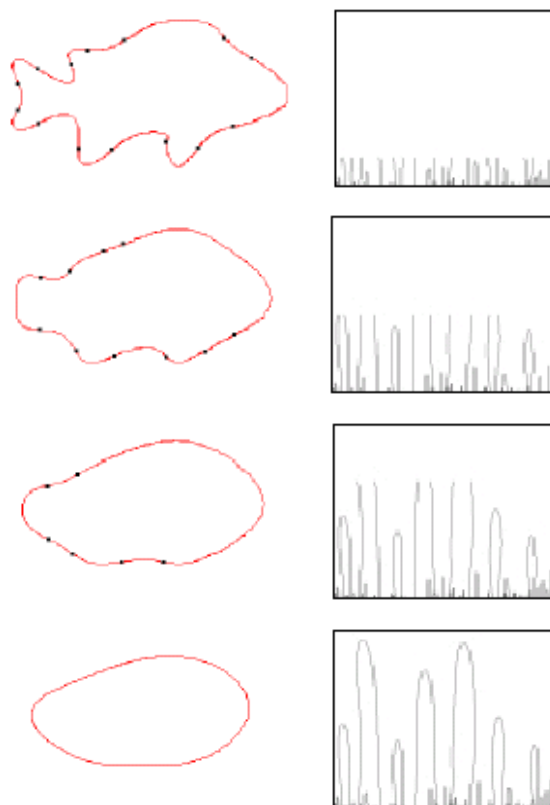


Figura 2.9: Evolução da forma e correspondente CSSI

Após várias evoluções, obtém-se uma imagem onde não ocorrem mais inflexões que é o indicativo de que o processo deve ser suspenso. Nesse instante, buscam-se os picos da CSSI os quais são armazenados como a assinatura da forma, ou seja, como o vetor de feições que será utilizado como índice para caracterizar a imagem. Apenas os picos que possuam uma altura acima de um percentual relacionado ao maior pico, cerca de 20% são armazenados, os demais são desprezados por serem o ruído oriundo dos primeiros níveis de evolução. Picos que se mantêm durante muitos níveis de evolução representam os pontos que caracterizam a forma.

Na Figura 2.10, é apresentada uma imagem de um caractere e seu respectivo contorno. O processo de arredondamento deste contorno em vários níveis de evolução pode ser verificado nas Figuras 2.11 e 2.12.

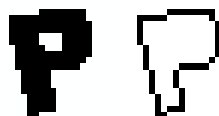


Figura 2.10: Caractere e seu contorno

Na Figura 2.11 o contorno original foi reamostrado utilizando-se  $N = 64$  pontos, enquanto na Figura 2.12 o referido contorno foi reamostrado utilizando-se  $N = 32$  pontos.

O tempo necessário para a extração das inflexões é proporcional à quantidade de pontos do contorno, assim, faz-se necessário escolher adequadamente este valor.

A fim de apresentar estas diferenças, um caractere teve seu contorno extraído (Figura 2.10) e então o processo de evolução foi acompanhado, sendo que as inflexões estão marcadas em vermelho.

Na Figura 2.11 (a) pode-se visualizar o contorno reamostrado com 64 pontos, sendo em (b) o resultado do primeiro nível de evolução onde já é possível verificar o processo de suavização da forma. Em (c)-(f), respectivamente, apresentam-se a segunda, terceira, quarta e quinta evoluções. As imagens restantes, (g)-(v) trazem as evoluções a cada cinco níveis, sendo desta forma, (g) o nível 10 e (v) o nível 85. Em (w) o último nível 89 é apresentado, sendo que o mesmo não possui nenhuma inflexão, indicativo de que o processo de evolução deve ser encerrado. É importante salientar, que em todos os níveis a quantidade de pontos é constante, no caso 64, possibilitando assim acompanhar o comportamento de cada ponto individualmente e assim, criar a CSSI.

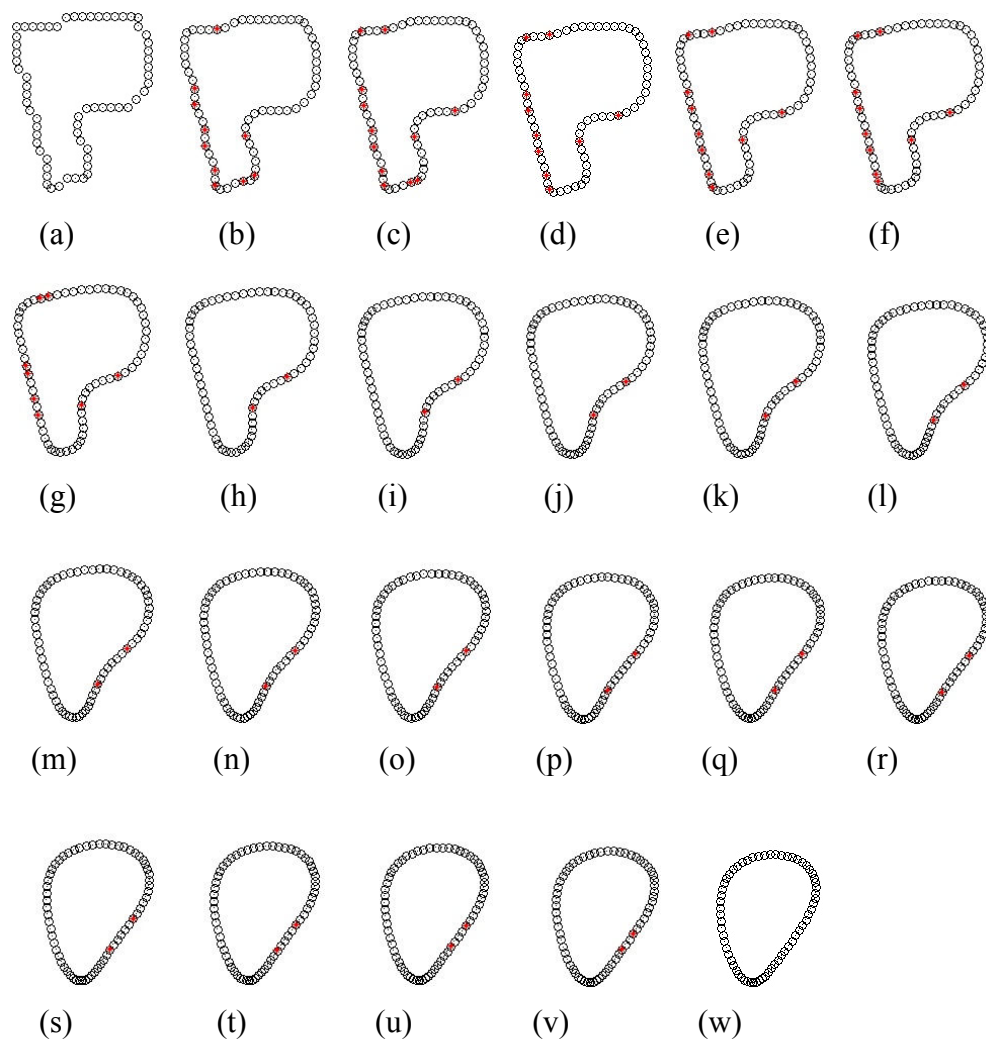


Figura 2.11: Diversas evoluções com  $N = 64$

A Figura 2.12(a) possui um contorno com 32 pontos e as imagens de (b)-(f) são as evoluções para os níveis de 1 a 5, da mesma maneira que na Figura 2.11. Desta forma, é possível efetuar uma comparação visual entre as duas imagens. As imagens (g), (h) e (i), representam a evolução para os níveis, 10, 15 e 20. Em (j), o nível 21, é apresentado não trazendo nenhuma inflexão.

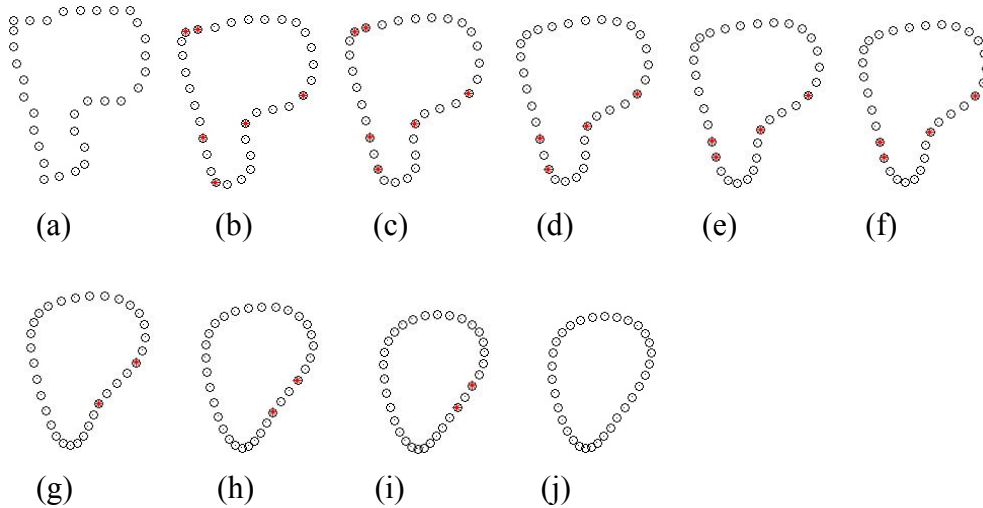


Figura 2.12: Diversas evoluções com  $N = 32$

As CSSI para as Figuras 2.11 e 2.12 não são apresentadas aqui por possuírem resolução muito baixa para análise visual a qual requer cerca de 256 pontos.

Variações do CSS e também outras de suas aplicações, como detecção de cantos, podem ser encontradas em (MOKHTARIAN, 1995; MOKHTARIAN; ABBASI; KITTLER, 1996; MOKHTARIAN; SUOMELA, 1998; ISO, 2001).

### 2.3.2 Similaridade Baseada em Região

Os descritores de forma baseados em região expressam distribuições de pixel como um objeto 2D, podendo descrever objetos complexos que consistem de múltiplas regiões desconexas como também objetos simples com ou sem furos (Figura 2.13). As técnicas baseadas em momentos são as principais representantes desse tipo de descritor.

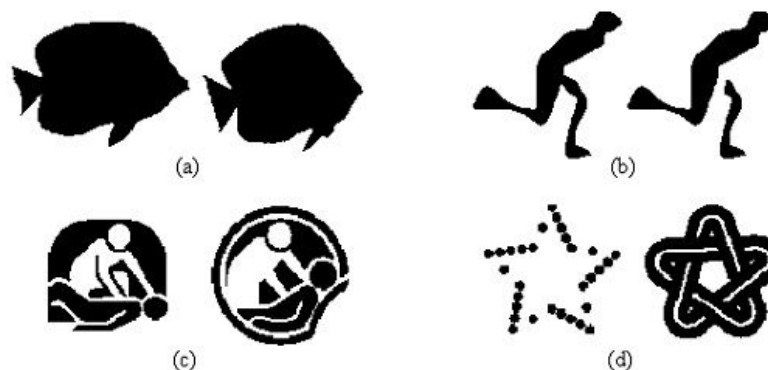


Figura 2.13: Exemplos de várias formas para descritores baseados em região

Na Figura 2.13 (ISO, 2001), observa-se que as imagens são similares em seus próprios conjuntos e dissimilares em relação a outros. Exemplificando, o conjunto em (a) é similar e o conjunto em (d) também o é, entretanto, (a) e (d) não são similares entre si.

As características que este tipo de descritor deve possuir são:

- Fornecer uma maneira compacta e eficiente de descrever propriedades de múltiplas regiões disjuntas simultaneamente;
- Algumas vezes durante o processo de segmentação, um objeto pode ser dividido em sub-regiões desconexas. Cada um dos objetos pode ser recuperado, desde que se saiba como a segmentação foi realizada;
- Robustez a ruídos, como por exemplo, ruído *salt & pepper*.

Dentre os descritores de forma com similaridade baseada em região existentes na literatura, serão abordados os utilizados neste trabalho, que são: Momentos Invariantes, Momentos Invariantes Afins, Momentos de Zernike e Momentos Pseudo-Zernike.

Os Momentos Invariantes foram escolhidos para este trabalho por serem amplamente apresentados na literatura da área e também por serem utilizados em muitas aplicações comerciais como, por exemplo, o QBIC. Os Momentos Invariantes Afins por sua vez foram selecionados pois foram apresentados como uma alternativa eficiente para os Momentos Invariantes tradicionais.

Estes dois métodos são derivados a partir dos momentos geométricos e possuem uma característica indesejada que é a redundância de informação. Para evitar a redundância existente nestes momentos recomenda-se o emprego dos momentos ortogonais os quais são representados principalmente pelos Momentos de Zernike e pelos Momentos Pseudo-Zernike.

Além disso, os Momentos de Zernike foram o método padrão de descritor de forma com similaridade baseada em região durante grande parte do processo de padronização do MPEG-7 coincidindo com o período de pesquisas dessa dissertação. Ao final da padronização do MPEG-7 este método foi substituído pelo *Angular Radial Transform* (ART). Entretanto, muitas pesquisas estão em andamento com os Momentos de Zernike e Momentos Pseudo-Zernike, enquanto pouca coisa é apresentada sobre o ART.

### 2.3.2.1. Momentos Geométricos

No caso de uma imagem contínua bidimensional  $f(x,y)$ , o momento geométrico de ordem  $(p+q)$  é definido como (GONZALEZ; WOODS, 2000).

$$m_{pq} = \int_{-\infty-\infty}^{+\infty+\infty} x^p y^q f(x, y) dx dy \quad (2.6)$$

para  $p, q = 0, 1, 2, \dots$

Um teorema de unicidade afirma que se  $f(x,y)$  for contínua por partes e possuir valores não-nulos apenas em uma parte finita do plano, então existem os momentos de todas as ordens e a seqüência de momentos  $m_{pq}$  é unicamente determinada por  $f(x,y)$ . Por outro lado,  $m_{pq}$  determina de maneira única  $f(x,y)$ . Para propósitos de invariância, é mais conveniente utilizar os momentos centrais, os quais, por definição, são invariantes a translação. Os momentos centrais podem ser expressos como

$$\mu_{pq} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} (x - \bar{x})^p (y - \bar{y})^q f(x, y) dx dy \quad (2.7)$$

Onde o centróide da imagem é obtido através de

$$\bar{x} = \frac{m_{10}}{m_{00}} \quad \text{e} \quad \bar{y} = \frac{m_{01}}{m_{00}}$$

No caso de uma imagem digital, a Equação 2.7 torna-se

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q f(x, y) \quad (2.8)$$

Os momentos centrais até a ordem 3 são

$$\mu_{00} = m_{00}$$

$$\mu_{01} = 0$$

$$\mu_{10} = \sum_x \sum_y (x - \bar{x})^1 (y - \bar{y})^0 f(x, y) = m_{10} - \frac{m_{10}}{m_{00}} (m_{00}) = 0$$

$$\mu_{11} = \sum_x \sum_y (x - \bar{x})^1 (y - \bar{y})^1 f(x, y) = m_{11} - \frac{m_{10} m_{01}}{m_{00}}$$

$$\mu_{20} = \sum_x \sum_y (x - \bar{x})^2 (y - \bar{y})^0 f(x, y) = m_{20} - \frac{2m_{10}^2}{m_{00}} + \frac{m_{10}^2}{m_{00}} = m_{20} - \frac{m_{10}^2}{m_{00}}$$

$$\mu_{02} = \sum_x \sum_y (x - \bar{x})^0 (y - \bar{y})^2 f(x, y) = m_{02} - \frac{m_{01}^2}{m_{00}}$$

$$\mu_{30} = \sum_x \sum_y (x - \bar{x})^3 (y - \bar{y})^0 f(x, y) = m_{30} - 3\bar{x}m_{20} + 2\bar{x}^2 m_{10}$$

$$\mu_{12} = \sum_x \sum_y (x - \bar{x})^1 (y - \bar{y})^2 f(x, y) = m_{12} - 2\bar{y}m_{11} - \bar{x}m_{02} + 2\bar{y}^2 m_{10}$$

$$\mu_{21} = \sum_x \sum_y (x - \bar{x})^2 (y - \bar{y})^1 f(x, y) = m_{21} + 2\bar{x}m_{11} - \bar{y}m_{20} + 2\bar{x}^2 m_{01}$$

$$\mu_{03} = \sum_x \sum_y (x - \bar{x})^0 (y - \bar{y})^3 f(x, y) = m_{03} - 3\bar{y}m_{02} + 2\bar{y}^2 m_{01}$$

Os momentos centrais normalizados, são invariantes a mudanças de escala e definidos por

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\frac{p+q}{2}}} \quad (2.9)$$

em que



$$\gamma = \frac{p+q}{2} + 1 \quad (2.10)$$

para  $p + q = 2, 3, \dots$

Mais informações sobre estes momentos e suas variações podem ser encontradas em (SONKA; HLAVAC; BOYLE, 1999; JAIN, 1989; REISS, 1993; KULKARNI, 1994; BIMBO, 1999; CHIN; KASSIM; IBRAHIM, 1999), dentre outros. A seguir, os Momentos Invariantes e os Momentos Invariantes Afins, representantes deste tipo de momentos serão descritos.

### Momentos Invariantes

Hu (1962) definiu um conjunto de sete momentos invariantes que são invariantes a translação, rotação e escala e que podem ser derivados a partir dos segundo e terceiro momentos centrais normalizados (Equações 2.11-2.17). A principal desvantagem da técnica é que não há garantia de que estas sete funções formem um conjunto completo de descritores para os padrões de entrada (KULKARNI, 1994).

$$\phi_1 = \eta_{20} + \eta_{02} \quad (2.11)$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \quad (2.12)$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \quad (2.13)$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} - \eta_{03})^2 \quad (2.14)$$

$$\phi_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \quad (2.15)$$

$$\phi_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \quad (2.16)$$

$$\phi_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \quad (2.17)$$

Observe que  $\phi_1 \dots \phi_6$  são invariantes também a reflexão enquanto  $\phi_7$  foi definido de maneira a mudar o sinal sob reflexão (SZMURLO, 1995; REISS, 1993).

### Momentos Invariantes Afins

Flusser e Suk (1993) deduziram um novo conjunto de invariantes, estas entretanto, invariantes a transformações afins em geral. A Figura 2.14 apresenta um de seus experimentos originais para o reconhecimento de letras, sendo em (a) a imagem original e em (b) a imagem de teste. Os resultados deste experimento estão na Tabela 2.1. Na Figura 2.15 outro de seus experimentos, este para formas geométricas. Os resultados do reconhecimento estão na Tabela 2.2. Para estes experimentos apenas 3 dos 4 momentos

derivados (Equação 2.18-2.21) foram utilizados pois, segundo os autores, para imagens simples, os 3 primeiros momentos já são eficazes.

A vantagem deste método em relação aos Momentos Invariantes é a sua habilidade de reconhecer formas que sofreram transformações afins, como as deformações apresentadas nas Figuras 2.14 (b) e 2.15 (b).

Observe que os experimentos apresentados para este descritor foram extraídos de Flusser e Suk (1993) onde o método foi proposto.

$$I_1 = (\mu_{20}\mu_{02} - \mu_{11}^2) / \mu_{00}^4 \quad (2.18)$$

$$I_2 = (\mu_{30}\mu_{03}^2 - 6\mu_{30}\mu_{21}\mu_{12}\mu_{03} + 4\mu_{30}\mu_{12}^3 + 4\mu_{21}^3\mu_{03} - 3\mu_{21}^2\mu_{12}^2) / \mu_{00}^{10} \quad (2.19)$$

$$I_3 = (\mu_{20}(\mu_{21}\mu_{03} - \mu_{12}^2) - \mu_{11}(\mu_{30}\mu_{03} - \mu_{21}\mu_{12}) + \mu_{02}(\mu_{30}\mu_{12} - \mu_{21}^2)) / \mu_{00}^7 \quad (2.20)$$

$$\begin{aligned} I_4 = & (\mu_{20}^3\mu_{03}^2 - 6\mu_{20}^2\mu_{11}\mu_{12}\mu_{03} - 6\mu_{20}^2\mu_{02}\mu_{21}\mu_{03} \\ & + 9\mu_{20}^2\mu_{02}\mu_{12}^2 + 12\mu_{20}\mu_{11}^2\mu_{21}\mu_{03} \\ & + 6\mu_{20}\mu_{11}\mu_{02}\mu_{30}\mu_{03} - 18\mu_{20}\mu_{11}\mu_{02}\mu_{21}\mu_{12} \\ & - 8\mu_{11}^3\mu_{30}\mu_{03} - 6\mu_{20}\mu_{02}^2\mu_{30}\mu_{12} + 9\mu_{20}\mu_{02}^2\mu_{21}^2 \\ & + 12\mu_{11}^2\mu_{02}\mu_{30}\mu_{12} - 6\mu_{11}\mu_{02}^2\mu_{30}\mu_{21} \\ & + \mu_{02}^3\mu_{20}^2) / \mu_{00}^{11} \end{aligned} \quad (2.21)$$

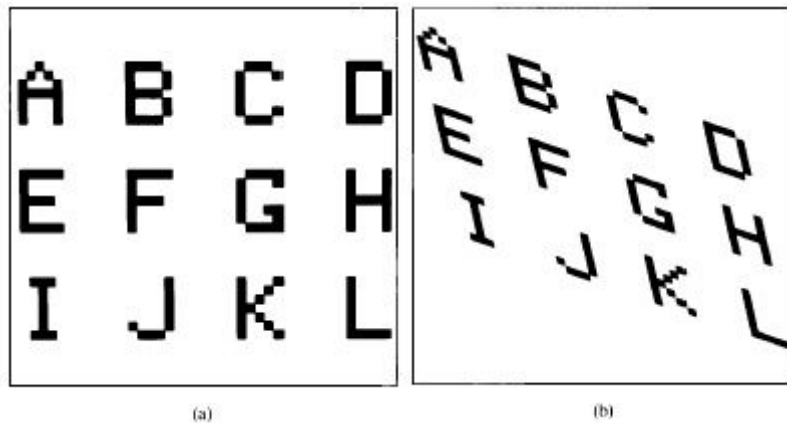


Figura 2.14: Experimento para letras

Tabela 2.1: Momentos invariantes afins das letras do Experimento 1

Letra	Figura 2.14 (a)			Figura 2.14 (b)		
	$I_1[10^{-4}]$	$I_2[10^{-8}]$	$I_3[10^{-6}]$	$I_1[10^{-4}]$	$I_2[10^{-8}]$	$I_3[10^{-6}]$
A	330	6	-133	336	10	-137
B	292	0	-15	294	0	-16
C	775	1286	192	779	1290	186
D	468	-8	-64	470	-8	-65
E	367	120	62	376	123	62
F	315	-245	-318	315	-234	-311
G	516	-30	-139	519	-33	-146
H	408	0	0	411	0	0
I	218	0	0	219	0	0
J	601	3423	-1240	599	3405	-1227
K	511	550	187	518	586	195
L	523	-2295	-1950	526	-2110	-1952

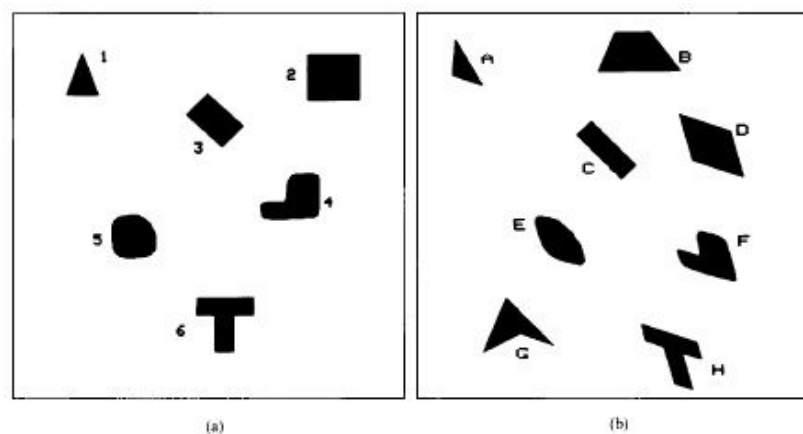


Figura 2.15: Experimento para formas geométricas

Tabela 2.2: Momentos invariantes afins das formas do Experimento 2

Objeto	Figura 2.15 (a)			Objeto	Figura 2.15 (b)		
	$I_1[10^{-4}]$	$I_2[10^{-8}]$	$I_3[10^{-6}]$		$I_1[10^{-4}]$	$I_2[10^{-8}]$	$I_3[10^{-6}]$
1	92	-32	-55	A	93	-32	-55
2	69	0	0	B	75	-2	-14
3	70	0	0	C	69	0	-1
4	89	-7	-25	D	70	0	-1
5	64	0	-1	E	64	0	-1
6	143	-168	-155	F	90	-7	-25
				G	145	-241	-202
				H	143	-170	-157

### 2.3.2.2. Momentos Ortogonais

Momentos geométricos como definidos pela Equação 2.6 têm a forma de uma projeção da imagem  $f(x,y)$  sobre o monômio  $x^p y^q$ . Ao conjunto base  $x^p y^q$  faltam algumas características desejadas, como a ortogonalidade. Conseqüentemente, as feições resultantes dos momentos geométricos não são ótimas por possuírem redundância de informação (KHOTANZAD; HONG, 1990; PROKOP; REEVES, 1992; SZMURLO, 1995; TRIER; JAIN; TAXT, 1996; SQUIRE, 1996; MEHTRE; KANKANHALLI; LEE, 1997; LIAO; PAWLAK, 1998; KIM; KIM, 2000).

Teague (1980) sugeriu os momentos ortogonais baseados na teoria dos polinômios ortogonais para resolver os problemas associados com os momentos geométricos. Os momentos de Zernike usados neste estudo são da classe dos momentos ortogonais e foram escolhidos em relação a outros momentos ortogonais porque possuem a propriedade de invariância a rotação. Rotacionar a imagem não altera a magnitude dos momentos de Zernike. Desta forma, eles podem ser usados para representar imagens com feições invariantes a rotação. Outra propriedade importante dos momentos de Zernike é a facilidade de reconstrução da imagem a partir deles. A ortogonalidade proporciona separar a contribuição individual gerada para os momentos de cada ordem no processo de reconstrução. A adição, destas contribuições individuais gera a imagem reconstruída. A seguir, são dadas as definições dos momentos de Zernike e dos momentos Pseudo-Zernike.

#### Momentos de Zernike

Os polinômios de Zernike  $V_{nm}(x,y)$  são um conjunto de polinômios complexos que forma um conjunto ortogonal completo no interior de um círculo unitário:

$$V_{nm}(x, y) = V_{nm}(r, \Theta) = R_{nm}(r) e^{im\Theta} \quad (2.22)$$

onde  $n \geq 0$ ,  $|m| \leq n$ ,  $n - |m|$  é par,  $r = \sqrt{x^2 + y^2}$ ,  $\Theta = \tan^{-1}(y/x)$ ,  $i = \sqrt{-1}$

$$R_{nm}(x, y) = R_{nm}(r) = \sum_{s=0}^{\frac{n-|m|}{2}} (-1)^s \frac{(n-s)!}{s! \left(\frac{n-|m|}{2} - s\right)! \left(\frac{n+|m|}{2} - s\right)!} r^{n-2s} \quad (2.23)$$

Observe que  $R_{n,-m}(r) = R_{nm}(r)$ .

Estes polinômios são ortogonais e satisfazem

$$\iint_{x^2+y^2 \leq 1} [V_{nm}(x, y)]^* V_{pq}(x, y) dx dy = \frac{\pi}{n+1} \delta_{np} \delta_{mq} \quad (2.24)$$

onde o símbolo \* representa o operador complexo conjugado e

$$\delta_{ab} = \left\{ \begin{array}{l} 1 \text{ se } a = b, \\ 0 \text{ caso contrário} \end{array} \right\}$$

Os momentos de Zernike (MZ) de ordem  $n$  com repetição  $m$  de uma função  $f$ ,  $f(x,y)=0$  para  $x^2 + y^2 > 1$ , são definidos por:

$$A_{nm} = \frac{n+1}{\pi} \iint_{x^2+y^2 \leq 1} f(x,y)[V_{nm}(x,y)]^* dx dy \quad (2.25)$$

No caso das imagens digitais, as integrais são substituídas por somatórios:

$$A_{nm} = \frac{n+1}{\pi} \sum_x \sum_y f(x,y)[V_{nm}(x,y)]^* \quad (2.26)$$

onde  $x^2 + y^2 \leq 1$

Para calcular os momentos de Zernike de uma imagem, o centro da imagem é considerado como origem e as coordenadas dos pixels são mapeadas para o intervalo do círculo unitário, isto é,  $x^2 + y^2 \leq 1$ . Aqueles pixels que permanecerem fora deste círculo não são utilizados no cálculo. Observe também que  $A_{nm}^* = A_{n,-m}$ .

Devido à propriedade de ortogonalidade de  $V_{nm}(x,y)$ , a reconstrução de  $f$  baseada nos primeiros  $N$  momentos pode ser calculada através de:

$$\hat{f}(x,y) = \sum_{n=0}^N \sum_m A_{nm} V_{nm}(x,y) \quad (2.27)$$

com as mesmas restrições anteriores de  $m$ .

Em relação às características de invariância, pode-se afirmar que as magnitudes  $|A_{nm}|$  são invariantes a rotação.

Invariância a escala e translação podem ser obtidas deslocando e escalando a imagem antes de calcular os momentos de Zernike. Isto pode ser feito com a utilização dos momentos geométricos, isto é,  $m_{pq}$  de cada imagem. Invariância à translação é obtida transformando a imagem em outra, onde os momentos de ordem  $m_{01}$  e  $m_{00}$ , são iguais a zero. Isto é feito pela transformação da imagem original  $f(x,y)$  em outra com  $f(x+\bar{x}, y+\bar{y})$  onde  $\bar{x}$  e  $\bar{y}$  são a localização do centróide da imagem original.

Em outras palavras, a origem é movida para o centróide antes do cálculo dos momentos.

Invariância a escala pode ser obtida pelo aumento ou redução de cada forma fazendo com que  $m_{00}$  seja igual a um valor pré-determinado de  $\beta$ , onde  $\beta$  é o tamanho pré-determinado da imagem. Observe que, para imagens binárias,  $m_{00}$  é o número total de pixels da forma na imagem.

A imagem  $f(x,y)$  é representada por sua versão escalada através de  $f(x/a,y/a)$  onde  $a = \sqrt{\beta/m_{00}}$ .


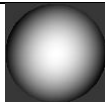

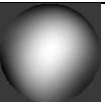
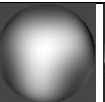
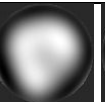


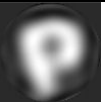
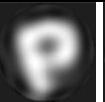
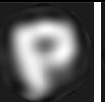
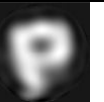



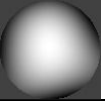
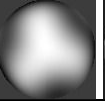
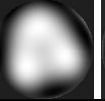
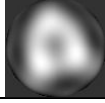
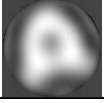

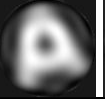
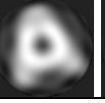
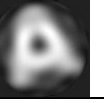




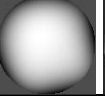
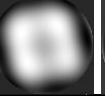











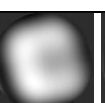




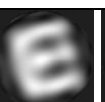
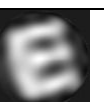
Por fim, uma imagem  $f(x,y)$  pode ser normalizada em escala e translação transformando-a em  $g(x,y)$ , onde

$$g(x,y) = f\left(\frac{x}{a} + \bar{x}, \frac{y}{a} + \bar{y}\right) \quad (2.28)$$

A derivação das equações aqui apresentadas pode ser verificada em (KHOTANZAD; HONG, 1990).

Para facilitar o entendimento do processo de reconstrução das imagens através dos momentos de Zernike, realizou-se um teste apresentado na Tabela 2.3 que utiliza para a reconstrução os momentos de ordem 2 até 13.

Tabela 2.3: Exemplos de caracteres reconstruídos com os MZ

Imagem Original	Imagens reconstruídas com momentos de ordem 2, 3, 4..13, da esquerda para a direita					
<b>P</b>						
						
<b>A</b>						
						
<b>R</b>						
						
<b>E</b>						
						

A Tabela 2.4 apresenta as restrições de  $n$  e  $m$  e as ordens de 0 a 10 válidas para os momentos de Zernike. Conforme definição,  $m = -m$  e assim, apenas os  $m \geq 0$  são utilizados.

Tabela 2.4: Número de momentos entre ordens 0 e 10 para os MZ

$n \geq 0$	$ m  \leq n, n -  m  = \text{par}$	Número de Momentos	
		$\pm m$	$+m$
0	0	1	1
1	-1, 1	2	1
2	-2, 0, 2	3	2
3	-3, -1, 1, 3	4	2
4	-4, -2, 0, 2, 4	5	3
5	-5, -3, -1, 1, 3, 5	6	3
6	-6, -4, -2, 0, 2, 4, 6	7	4
7	-7, -5, -3, -1, 1, 3, 5, 7	8	4
8	-8, -6, -4, -2, 0, 2, 4, 6, 8	9	5
9	-9, -7, -5, -3, -1, 1, 3, 5, 7, 9	10	5
10	-10, -8, -6, -4, -2, 0, 2, 4, 6, 8, 10	11	6

### Momentos Pseudo-Zernike

Teh e Chin (1988) propuseram uma modificação nos momentos de Zernike. As principais propriedades, principalmente a ortogonalidade e a invariância a rotação são compartilhadas com os momentos de Zernike sendo que os momentos Pseudo-Zernike (MPZ)  $P_{nm}$  diferem de  $A_{nm}$  pela definição do polinômio radial:

$$R'_{nm}(x, y) = R'_{nm}(r) = \sum_{s=0}^{n-|m|} (-1)^s \frac{(2n+1-s)!}{s!(n-|m|-s)!(n+|m|+1-s)!} r^{n-s} \quad (2.29)$$

Os momentos Pseudo-Zernike são então definidos por:



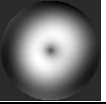
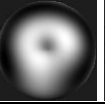
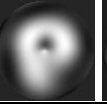
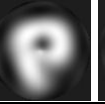


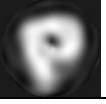

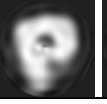
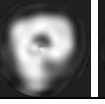




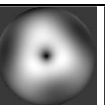
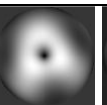
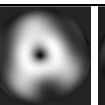
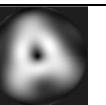
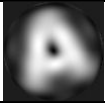
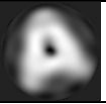
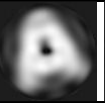
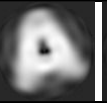
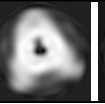
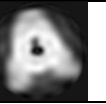


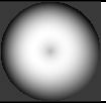
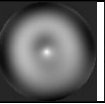
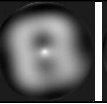
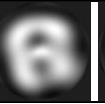


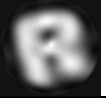
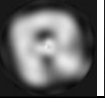
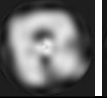
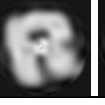








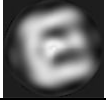
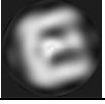
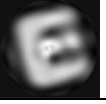


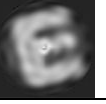
$$P_{nm} = \frac{n+1}{\pi} \sum_x \sum_y f(x, y) R'_{nm}(x, y) e^{-im\Theta} \quad (2.30)$$

onde  $n \geq 0$  e  $|m| \leq n$ . A condição  $n - |m|$  é par, é eliminada.

Por isso, para uma dada ordem  $n$ , existe aproximadamente o dobro de momentos Pseudo-Zernike,  $(n+1)^2$ , em comparação com os momentos de Zernike convencionais  $(n+1)(n+2)/2$ . Os momentos de ordem 0 a 10 para os MZ e MPZ podem ser verificados nas Tabelas 2.4 e 2.6, respectivamente.

Os momentos  $P_{nm}$  teoricamente mostram ser menos sensíveis ao ruído do que os momentos  $A_{nm}$ . A Tabela 2.5 apresenta a reconstrução das imagens através dos momentos Pseudo-Zernike de ordem 2 até 13.

Tabela 2.5: Exemplos de caracteres reconstruídos com os MPZ

Imagem Original	Imagens reconstruídas com momentos de ordem 2, 3, 4..13, da esquerda para a direita						
<b>P</b>							
							
<b>A</b>							
							
<b>R</b>							
							
<b>E</b>							
							

A Tabela 2.6 apresenta as restrições de  $n$  e  $m$  e as ordens de 0 a 10 válidas para os momentos Pseudo-Zernike. Conforme definição,  $m = -m$  e assim, apenas os  $m \geq 0$  são utilizados.



Tabela 2.6: Número de momentos entre ordens 0 e 10 para os MPZ

$n \geq 0$	$ m  \leq n$	Número de Momentos	
		$\pm m$	$+m$
0	0	1	1
1	-1, 0, 1	3	2
2	-2, -1, 0, 1, 2	5	3
3	-3, -2, -1, 0, 1, 2, 3	7	4
4	-4, -3, -2, -1, 0, 1, 2, 3, 4	9	5
5	-5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5	11	6
6	-6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6	13	7
7	-7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7	15	8
8	-8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8	17	9
9	-9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9	19	10
10	-10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10	21	11

## 2.4 Classificadores

Sabendo que os descritores não são o único responsável pelo desempenho do sistema faz-se necessário uma descrição de outra etapa fundamental de um sistema de reconhecimento de padrões denominada classificação.

Os classificadores apresentados aqui estão divididos em estatísticos e neurais. Os estatísticos são mais empregados pela comunidade oriunda das áreas de Engenharia e Estatística e mais particularmente pelos profissionais que trabalham com processamento de imagens digitais e reconhecimento de padrões. Os profissionais que possuem sua formação na Inteligência Artificial costumam fazer uso de classificadores neurais, em alusão ao processamento realizado pelos neurônios no cérebro.

A maioria dos classificadores estatísticos tem como base a regra de decisão de Bayes, que assume uma distribuição normal da população. Apesar dessa regra de decisão ser muito simples, é difícil aplicá-la na prática porque a probabilidade a priori normalmente não é conhecida e, desta forma, deve ser estimada a partir das amostras.

Na classificação com uso de redes neurais não é necessário fazer-se nenhuma consideração das funções de densidade de probabilidade; contudo, existe o problema da definição da arquitetura da RNA, visto que não há nenhuma teoria para se determinar, a priori, a arquitetura que melhor resultado apresentaria na classificação.

Com o intuito de restringir o espectro de abrangência, apenas os classificadores utilizados neste trabalho serão descritos nas seções subseqüentes.

### 2.4.1 Classificadores Estatísticos

As abordagens de decisão teórica para o reconhecimento baseiam-se na utilização de funções de decisão. Seja  $\mathbf{x}=(x_1,x_2,\dots,x_n)^T$  um vetor de padrões  $n$ -dimensional. Para  $M$  classes de padrões  $w_1, w_2,\dots,w_M$ , o problema básico em reconhecimento de padrões por decisão teórica é encontrar  $M$  funções de decisão  $g_1(\mathbf{x}),g_2(\mathbf{x}),\dots,g_M(\mathbf{x})$ , com a propriedade que, se o padrão  $\mathbf{x}$  pertencer à classe  $w_i$ , então (GONZALEZ; WOODS, 2000).

$$g_i(\mathbf{x}) > g_j(\mathbf{x}) \quad i = 1,2,\dots,M; j \neq i \quad (2.31)$$

Em outras palavras, um padrão desconhecido  $\mathbf{x}$  pertence à  $i$ -ésima classe de padrões se a substituição de  $\mathbf{x}$  em todas as funções de decisão fizer com que  $g_i(\mathbf{x})$  tenha o maior valor numérico. Empates são resolvidos arbitrariamente.

Desta forma, o processo de classificação aplicado ao nosso problema consiste de duas etapas. Primeiramente, estima-se um modelo para cada uma das classes a partir de exemplos – etapa de treinamento ou de estimação.

A segunda etapa consiste em determinar qual das funções de decisão tem maior valor para um dado padrão de entrada desconhecido – etapa de teste. A que possuir maior valor será considerada como o modelo mais provável para representar este novo padrão desconhecido.

Uma das características principais dos classificadores desta natureza é a de que estes necessitam de informações detalhadas a respeito das estatísticas sobre as imagens a serem classificadas.

Este fato mostra-se como uma limitação na utilização dos mesmos, já que processos de manipulação dos dados, anteriores à classificação propriamente dita, tornam-se indispensáveis, ou, principalmente, quando nenhuma informação *a priori* a respeito dos dados está disponível. Os classificadores estatísticos freqüentemente assumem que valores de atributos possuem distribuição normal, e então usam os dados fornecidos para determinar média e covariância da distribuição.

A seguir, descrevemos os classificadores estatísticos por Máxima Verossimilhança Gaussiana (MVG) e por Distância Mínima (DM).

#### 2.4.1.1. Máxima Verossimilhança Gaussiana

O classificador por Máxima Verossimilhança Gaussiana (MVG) baseia-se na estimação de *modelos estatísticos* para cada uma das classes envolvidas no processo de classificação. Estes modelos representam uma função cujo argumento é um vetor de feições e cuja saída é um número relacionado à *probabilidade* de este vetor pertencer ao modelo em questão.

Este modelo é representado por uma função gaussiana multivariada em  $n$  dimensões, onde  $n$  é o número de feições. A estimação consiste na determinação do vetor média e da matriz de covariância, que são os parâmetros que definem uma gaussiana multivariada. Os valores estimados podem ser obtidos a partir de (DUDA; HART, 1973; RICHARDS, 1993):

$$\boldsymbol{\mu} = \frac{1}{n} \sum_{k=1}^n \mathbf{x}_k \quad (2.32)$$

$$\Sigma = \frac{1}{n} \sum_{k=1}^n (\mathbf{x}_k - \boldsymbol{\mu})(\mathbf{x}_k - \boldsymbol{\mu})^T \quad (2.33)$$

onde  $\boldsymbol{\mu}$  é o vetor média e  $\Sigma$  a matriz de covariâncias. A função gaussiana resultante é representada por

$$p(\mathbf{x}|w_i) = (2\pi)^{-n/2} |\Sigma_i|^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^T \Sigma_i^{-1}(\mathbf{x} - \boldsymbol{\mu}_i)\right) \quad (2.34)$$

Assim, este classificador atribui um padrão  $\mathbf{x}$  a uma classe  $w_i$  se

$$p(\mathbf{x} | w_i)P(w_i) > p(\mathbf{x} | w_j)P(w_j) \quad i = 1, 2, \dots, M; j \neq i \quad (2.35)$$

onde  $p(\mathbf{x}|w_i)$  é a função densidade de probabilidade dos padrões da classe  $w_i$  e  $P(w_i)$  é a probabilidade de ocorrência da classe  $w_i$ . Entretanto, é mais conveniente trabalhar com o logaritmo natural da função de decisão, devido à forma exponencial da densidade gaussiana. Em outras palavras, podemos utilizar

$$g_i(\mathbf{x}) = \ln\{p(\mathbf{x} | w_i)P(w_i)\} = \ln p(\mathbf{x} | w_i) + \ln P(w_i) \quad (2.36)$$

A substituição da Equação (2.34) na Equação (2.36) leva a

$$g_i(\mathbf{x}) = \ln P(w_i) - \frac{n}{2} \ln 2\pi - \frac{1}{2} \ln |\Sigma_i| - \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^T \Sigma_i^{-1}(\mathbf{x} - \boldsymbol{\mu}_i) \quad (2.37)$$

O termo  $(n/2) \ln 2\pi^2$  é comum para todas as classes, de maneira que pode ser eliminado da Equação (2.37), que se torna

$$g_i(\mathbf{x}) = \ln P(w_i) - \frac{1}{2} \ln |\Sigma_i| - \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^T \Sigma_i^{-1}(\mathbf{x} - \boldsymbol{\mu}_i) \quad (2.38)$$

A implementação da regra de decisão da máxima verossimilhança é realizada através da Equação (2.38) em que um vetor de padrões  $\mathbf{x}$  é atribuído à classe  $w_i$  se  $g_i(\mathbf{x}) > g_j(\mathbf{x})$  para todo  $j \neq i$ .

Para distribuições amostrais que satisfazem a condição de normalidade, é bem conhecido o resultado que afirma que a classificação por MVG é ótima, no sentido em que ela minimiza o erro de classificação (DUDA; HART, 1973). Para que essa condição de ser ótima seja verdadeira, entretanto, as funções densidade de probabilidade dos padrões de cada classe, bem como a probabilidade de ocorrência de cada classe, devem ser conhecidas. Essa última restrição normalmente não é um problema pois, por exemplo, se todas as classes puderem ocorrer com a mesma probabilidade, então  $P(w_i) = 1/M$ . Mesmo se essa relação não for verdadeira, essas probabilidades geralmente podem ser inferidas a partir do conhecimento prévio sobre o problema (GONZALEZ; WOODS, 2000).

Erbert e Haertel (2003) citam que com dados que possuem alta dimensionalidade, é possível separar classes que possuem vetores média iguais, desde que as matrizes de covariâncias difiram suficientemente entre si. O emprego de dados em alta dimensionalidade apresenta entretanto, o problema da estimação dos parâmetros como a matriz de covariâncias. Sempre que o número de amostras de treinamento é pequeno, se comparado com a dimensionalidade dos dados, a estimativa dos parâmetros torna-se incerta, resultando numa degradação da performance do classificador.

Para que seja possível estimar o vetor média e a matriz de covariâncias necessita-se de uma quantidade representativa de amostras de treinamento para cada uma das diferentes classes. Segundo Richards (1993), para um espaço de  $N$  dimensões, pelo menos  $N+1$  amostras são necessárias para que a matriz de covariâncias não seja singular. Caso isto não ocorra não é possível obter a inversa na função de decisão. Fora esta consideração, é extremamente importante possuir a maior quantidade possível de amostras de treinamento, principalmente quando se trabalha com alta dimensionalidade pois alguma destas dimensões pode não ser devidamente representada. Recomenda-se na prática, para o treinamento, pelo menos  $10N$  amostras de cada classe, sendo  $100N$  altamente desejável.

Existem diversas pesquisas que objetivam estimar a matriz de covariâncias a partir de um conjunto limitado de dados de treinamento, e, apenas para exemplificar, citam-se os trabalhos de (HOFFBECK; LANDGREBE, 1996; ERBERT; HAERTEL, 2003).

#### 2.4.1.2. Distância Mínima

A eficácia da MVG depende da obtenção de uma acurada estimação de  $\boldsymbol{\mu}$  e  $\boldsymbol{\Sigma}$  para cada classe. Para que isto ocorra, deve existir um número suficiente de amostras para cada uma destas classes. Quando isto não ocorre obtêm-se estimações incorretas dos elementos de  $\boldsymbol{\Sigma}$  resultando num fraco desempenho de classificação. Quando o número de amostras de treinamento por classe é limitado pode ser mais eficaz recorrer a um classificador que não faz uso da covariância, dependendo somente da média das classes, observando que para um dado número de amostras estes podem ser estimados mais eficazmente que as covariâncias (RICHARDS, 1993). O classificador por distância mínima (DM) possui estas características, consistindo em um caso particular da classificação por MVG onde a matriz de covariâncias para as diferentes classes é igual e, além disso, múltipla da matriz identidade. Em outras palavras, se todas as matrizes de covariância forem iguais,  $\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}$ ,  $i=1,2,\dots,M$  e eliminando-se todos os termos independentes de  $i$  da Equação (2.38) teríamos

$$g_i(\mathbf{x}) = \ln P(w_i) + \mathbf{x}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_i - \frac{1}{2} (\boldsymbol{\mu}_i^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_i) \quad (2.39)$$

que são funções de decisão linear para  $i=1,2,\dots,M$ .

Se, além do mais, a matriz de covariâncias for múltipla da matriz identidade então

$$g_i(\mathbf{x}) = \ln P(w_i) + \mathbf{x}^T \boldsymbol{\mu}_i - \frac{1}{2} (\boldsymbol{\mu}_i^T \boldsymbol{\mu}_i) \quad (2.40)$$

Caso as probabilidades *a priori* sejam iguais,  $\ln P(w_i)$  é ignorado e então a função de decisão para um classificador de distância mínima se reduz a

$$g_i(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\mu}_i - \frac{1}{2} (\boldsymbol{\mu}_i^T \boldsymbol{\mu}_i) \quad (2.41)$$

Entretanto, por não fazer uso da covariância ele não é tão flexível como o classificador por MVG, não modelando adequadamente classes alongadas (RICHARDS, 1993). Assim, um desempenho substancialmente superior é esperado quando se faz uso da classificação por MVG.

Para um estudo mais aprofundado sobre a dedução das fórmulas e sobre as técnicas de classificação supervisionada através dos métodos de decisão teórica aconselha-se a consulta a (RICHARDS, 1993; GONZALEZ; WOODS, 2000).

#### 2.4.2 Classificadores Neurais

Uma Rede Neural Artificial (RNA) pode ser definida como um sistema computacional capaz de adquirir, representar e computar mapeamentos de um espaço multivariado para informações de outro espaço, dado um conjunto de dados representativos do mapeamento. Nas RNAs, o relacionamento matemático entre as variáveis não é especificado. As RNAs têm a habilidade de aprender a partir dos exemplos apresentados a elas podendo produzir respostas corretas para dados que não foram vistos, o que é conhecido como generalização.

As RNAs são dispositivos não-lineares, inspirados na funcionalidade dos neurônios biológicos, aplicados no reconhecimento de padrões, na otimização e na previsão de sistemas complexos e são compostas por diversas unidades computacionais paralelas, interconectadas parcial ou totalmente. Cada uma dessas unidades (neurônios artificiais) efetua um certo número de operações simples e transmite seus resultados às unidades vizinhas com as quais possui conexão. Através de um processo de treinamento, as RNAs passam a ser capazes de reconhecer padrões, mesmo que os dados utilizados nesse treinamento sejam incompletos, não-lineares ou até mesmo contraditórios. A habilidade de manipular esses dados imprecisos faz com que as RNAs sejam extremamente eficazes em tarefas onde um conjunto de regras não pode ser facilmente formulado (LIBERMAN, 1997).

Podemos classificar as RNAs de acordo com as seguintes características: tipo de treinamento e regra de aprendizado, topologia da rede e interconexão dos neurônios (FREEMAN; SKAPURA, 1991; HAYKIN, 2001).

Em tarefas de classificação, o treinamento pode ser supervisionado ou não-supervisionado, sendo que o supervisionado baseia-se na apresentação à rede de um padrão a ser reconhecido juntamente com a resposta que a rede deve fornecer. Neste tipo de treinamento temos uma regra de aprendizado do tipo correção de erros. O treinamento não-supervisionado consiste em apresentar apenas os padrões que se deseja reconhecer à rede e esta deverá ser capaz de agrupar os padrões que possuem características similares. Neste tipo de treinamento temos uma regra de aprendizado do tipo competitivo a qual se caracteriza pelas conexões laterais dos neurônios com seus vizinhos levando a rede a um estado estável (LIBERMAN, 1997).

Em relação à topologia, as RNAs podem ter uma única camada ou mais de uma camada (*multilayer*).

Quanto à interconexão dos neurônios, as RNAs podem ser com realimentação (*feedback*) ou sem realimentação (*feed-forward*). As RNAs sem realimentação possuem neurônios cujas saídas conectam-se apenas com os níveis superiores da rede, não havendo realimentação para suas entradas. No tipo com realimentação, a saída de um neurônio pode se conectar a uma de suas próprias entradas ou à entrada de um neurônio em um nível inferior à rede. Esta rede possui neurônios cuja saída é capaz de influenciar de uma maneira direta ou indireta o seu próprio comportamento (HAYKIN, 2001; LIBERMAN, 1997).

Os classificadores empregados neste trabalho são conhecidos como MLP (*multilayer perceptron*), que são RNAs multicamadas com treinamento supervisionado.

O treinamento de redes neurais multicamadas é um problema de otimização não-linear de uma função de custo, que mede o erro quadrático médio produzido pela saída da rede neural frente a uma saída desejada. Este fato traz grandes vantagens devido à vasta literatura existente sobre métodos de otimização não-linear, onde podemos encontrar uma grande variedade de métodos poderosos para otimização que podem ser prontamente aplicados ao problema de treinamento de redes neurais, caso se disponha de informações suficientes acerca da função de erro a ser minimizada (IYODA, 2000).

O algoritmo de retropropagação, que é o algoritmo padrão para esta família de redes neurais, é uma implementação de um método conhecido como *método do gradiente*. No método do gradiente, o vetor de parâmetros (pesos) é ajustado na direção oposta ao do vetor gradiente. O método do gradiente pode ser classificado como um método de *primeira ordem*, já que utiliza apenas a informação do gradiente da função de erro para ajustar os pesos da rede. Os métodos de primeira ordem são conhecidos por serem ineficientes no tratamento de problemas de larga escala, pois apresentam taxas de convergência muito pobres, especialmente em regiões próximas a mínimos locais (IYODA, 2000).

Em vista disso, e do fato de que os dados para treinamento geralmente apresentam grande dimensionalidade, é justificável a utilização de um método de otimização não-linear de segunda ordem. Nos métodos de segunda ordem, além do vetor gradiente da função objetivo, fazemos uso também da matriz *hessiana* (matriz de derivadas de segunda ordem) da função erro. Apesar de notadamente superiores aos métodos de primeira ordem, os métodos de segunda ordem também apresentam desvantagens, sendo a principal delas o alto custo computacional associado ao cálculo e armazenamento da matriz hessiana. Mais informações sobre métodos de primeira e segunda ordens podem ser obtidos em (BISHOP, 1995; MØLLER, 1993; SMAGT, 1994; CASTRO, 1998).

Neste trabalho vários algoritmos de MLP disponíveis no *toolbox* de Redes Neurais do Matlab foram testados, sendo que quatro dentre estes foram selecionados. O primeiro é um método de primeira ordem conhecido como Gradiente Descendente com termo de momento e taxa de aprendizado adaptativa (GDX). O segundo método é um algoritmo de otimização de segunda ordem chamado *Levenberg-Marquardt* (LM). Os outros métodos também são de otimização, sendo conhecidos por *Resilient Propagation* (RPROP) e *Scaled Conjugate Gradient* (SCG).

#### 2.4.2.1. Gradiente Descendente com termo de momento e taxa de aprendizado adaptativa

A implementação do aprendizado de retropropagação atualiza os pesos da rede e *bias* na direção em que a função de desempenho decresce mais rapidamente; isto é, o negativo do seu gradiente. Isto pode ser representado como:

$$\Delta w_k = -\eta_k p_k + \alpha_k \Delta w_{k-1} \quad (2.42)$$

onde  $\Delta w_k$  é o ajuste do vetor com os pesos e *bias* atuais,  $p_k$  é o gradiente atual,  $\eta_k$  é a taxa de aprendizado e  $\alpha_k$  é a constante de momento. Esta técnica é conhecida como “*steepest (gradient) descent*”. A variação nos pesos e *bias* é obtida pela multiplicação da taxa de aprendizado pelo gradiente negativo. Quanto maior for a taxa de aprendizado, maior é o passo. Se a taxa de aprendizado for muito grande o algoritmo fica instável. Se a taxa de aprendizado for muito pequena a convergência demora a ocorrer (ABDUL-KAREEM et. al., 2001).

O termo de momento possui como principal função acelerar o aprendizado sem produzir oscilações. A probabilidade da convergência esbarrar em mínimos locais é reduzida, uma vez que o termo ignora as variações de alta frequência na superfície de erro. A inclusão do momento se baseia em fazer com que as mudanças nos pesos das conexões sejam somadas a uma fração da última alteração nestes pesos determinada pela regra de aprendizagem. Desta forma, se a alteração anterior foi realizada num determinado sentido da superfície de erro, parte da atualização atual nos pesos das conexões será realizada no mesmo sentido (VALIATI, 2000).

Como anteriormente mencionado, se a taxa de aprendizado,  $\eta_k$  for muito alta o algoritmo pode sofrer oscilações e ficar instável. Se a taxa de aprendizado for muito pequena o algoritmo demorará a convergir (ABDUL-KAREEM et. al., 2001).

#### 2.4.2.2. Levenberg-Marquardt

Este método é bastante eficiente quando estamos tratando de redes que não possuem mais do que algumas centenas de conexões a serem ajustadas. Isto deve-se, principalmente, ao fato de que este algoritmo necessita armazenar uma matriz quadrada cuja dimensão é da ordem do número de conexões da rede (CASTRO, 1998).

Se considerarmos como funcional do erro a soma dos erros quadráticos (SSE), e ainda levarmos em conta que o problema pode ter múltiplas saídas, obtemos a seguinte expressão para o funcional do erro:

$$J(\theta) = \sum_{i=1}^N \sum_{j=1}^m (g_{ij}(\mathbf{x}) - \hat{g}_{ij}(\mathbf{x}, \theta))^2 = \sum_{k=1}^q r_k^2 \quad (2.43)$$

onde  $J$  é a superfície de erro,  $\theta$  é o vetor de parâmetros,  $g(\cdot)$  é a função a ser aproximada,  $\hat{g}(\cdot, \theta)$  é o modelo de aproximação,  $N$  é o número de amostras,  $l$  é o número de unidades intermediárias,  $r$  o erro residual,  $m$  o número de saídas e  $q$  o produto  $N \times m$ .

Seja  $\mathbf{J}$  o *Jacobiano* (matriz das derivadas primeiras) do funcional  $J$  dado pela Equação (2.43). Esta matriz pode ser escrita da seguinte forma:

$$\mathbf{J} \equiv \begin{bmatrix} \nabla r_1^T \\ \vdots \\ \nabla r_q^T \end{bmatrix} \quad (2.44)$$

Diferenciando a Equação (2.43) obtemos:

$$\nabla J = 2\mathbf{J}^T \mathbf{r} = 2 \sum_{k=1}^q r_k \nabla \mathbf{r}_k \quad (2.45)$$

$$H \equiv \nabla^2 J = 2 \left( \mathbf{J}^T \mathbf{J} + \sum_{k=1}^q r_k \nabla^2 \mathbf{r}_k \right) \quad (2.46)$$

A matriz de derivadas segundas do funcional de erro é chamada *de matriz hessiana*. Quando os erros residuais são suficientemente pequenos, a matriz hessiana pode ser aproximada pelo primeiro termo da Equação (2.46), resultando em:

$$\nabla^2 J \approx 2\mathbf{J}^T \mathbf{J} \quad (2.47)$$

Esta aproximação geralmente é válida em um mínimo de  $J$  para a maioria dos propósitos e é a base para o algoritmo de Gauss-Newton. A lei de atualização torna-se então:

$$\Delta\theta = [\mathbf{J}^T \mathbf{J}]^{-1} \mathbf{J}^T \mathbf{r} \quad (2.48)$$

A modificação de Levenberg-Marquardt para o método de Gauss-Newton é:

$$\Delta\theta = [\mathbf{J}^T \mathbf{J} + \mu \mathbf{I}]^{-1} \mathbf{J}^T \mathbf{r} \quad (2.49)$$

O efeito da matriz adicional  $\mu \mathbf{I}$  é adicionar  $\mu$  a cada autovalor de  $\mathbf{J}^T \mathbf{J}$ . Uma vez que a matriz  $\mathbf{J}^T \mathbf{J}$  é semi-definida positiva e portanto o autovalor mínimo possível é zero, qualquer valor positivo, pequeno, mas numericamente significativo, de  $\mu$  será suficiente para restaurar a matriz aumentada e produzir uma direção descendente de busca.

Os valores de  $\mu$  podem ser escolhidos de várias maneiras; a mais simples é escolhê-lo zero a menos que a matriz hessiana encontrada na iteração  $i$  seja singular. Quando isso ocorrer, um valor pequeno como  $\mu = 10^{-4} \sum_{i=1}^N (\mathbf{J}^T \mathbf{J})_{ii}$  pode ser usado (CASTRO, 1998). Mais informações a respeito deste algoritmo podem ser encontradas em (BISHOP, 1995).

#### 2.4.2.3. Resilient Propagation (RPROP)

O princípio básico do RPROP (RIEDMILLER; BRAUN, 1992; RIEDMILLER, 1994) é eliminar a influência destrutiva do tamanho da derivada parcial do peso de um determinado ciclo. Como consequência, apenas o sinal da derivação é considerado para indicar a “direção” da atualização do peso. O tamanho da mudança do peso é determinado exclusivamente por um peso específico, conhecido por *valor de atualização*  $\Delta_{ij}^{(t)}$ :

$$\Delta \mathbf{w}_{ij}^{(t)} = \begin{cases} -\Delta_{ij}^{(t)}, & \text{se } \frac{\partial E^{(t)}}{\partial w_{ij}} > 0 \quad \text{erro aumenta} \\ +\Delta_{ij}^{(t)}, & \text{se } \frac{\partial E^{(t)}}{\partial w_{ij}} < 0 \quad \text{erro diminui} \\ 0, & \text{outros casos} \end{cases} \quad (2.50)$$

onde  $\partial E^{(t)} / \partial w_{ij}$  denota a informação dos gradientes somados considerando-se todos os padrões de um conjunto (*batch learning*).



O segundo passo do aprendizado RPROP é determinar o novo valor de atualização  $\Delta_{ij}^{(t)}$ . Isto é baseado em um processo adaptativo dependente do sinal.

$$\Delta_{ij}^{(t)} = \begin{cases} \eta^+ * \Delta_{ij}^{(t-1)}, & \text{se } \frac{\partial E^{(t-1)}}{\partial w_{ij}} * \frac{\partial E^{(t)}}{\partial w_{ij}} > 0 \\ \eta^- * \Delta_{ij}^{(t-1)}, & \text{se } \frac{\partial E^{(t-1)}}{\partial w_{ij}} * \frac{\partial E^{(t)}}{\partial w_{ij}} < 0 \\ \Delta_{ij}^{(t-1)}, & \text{outros casos} \end{cases} \quad (2.51)$$

onde  $0 < \eta^- < 1 < \eta^+$ .

A regra adaptativa funciona da seguinte maneira: toda vez que a derivada parcial do peso correspondente  $w_{ij}$  muda seu sinal, indicando que a última atualização dos pesos foi muito grande e o algoritmo pulou sobre um mínimo local, o valor de atualização  $\Delta_{ij}^{(t)}$  é decrescido pelo fator  $\eta^-$ . No caso da derivada manter seu sinal, o valor de atualização é ligeiramente aumentado de modo a acelerar convergências em regiões aproximadamente planas. Os valores dos parâmetros de decremento,  $\eta^-$ , e do incremento,  $\eta^+$ , foram fixados pelos autores, respectivamente em 0,5 e 1,2.

O algoritmo possui dois parâmetros: o valor de atualização inicial  $\Delta_0$  e o limite máximo de atualização,  $\Delta_{\max}$ .

Quando o aprendizado começa, todos os valores de atualização são inicializados com  $\Delta_0$  que é definido em 0,1. A escolha deste parâmetro entretanto não é crítica pois ele é adaptado durante o treinamento.

#### 2.4.2.4. Scaled Conjugate Gradient (SCG)

Os métodos do gradiente conjugado (*Conjugate Gradient-CG*) encontram a direção de busca a cada iteração através de uma combinação linear entre a direção do gradiente atual e a anterior, onde o *steepest descent* é utilizado como a direção da primeira iteração. O algoritmo CG original, baseado na teoria das direções conjugadas aplicada a funções quadráticas faz uso de uma matriz hessiana  $\mathbf{H}$  das derivadas de segunda ordem e inversa em duas operações básicas: o cálculo do tamanho do passo e a combinação linear da direção atual e anterior do gradiente para encontrar a nova direção de busca. O cálculo e a inversão de uma matriz  $WxW$ , sendo  $W$  o número total de pesos da rede, são dois procedimentos com alto custo computacional para serem realizados a cada iteração durante o treinamento da rede. Além disso, acarreta a necessidade do cálculo explícito das derivadas de segunda ordem as quais não são diretamente obtidas por *backpropagation*. Considerando a inconveniência de manipular a Hessiana, uma aproximação foi derivada para substituir a matriz verdadeira no cálculo da direção conjugada enquanto o tamanho do passo é obtido por busca unidimensional. Com isto, uma versão simplificada do algoritmo que depende somente da função e do gradiente é obtida o que é considerado como sendo o gradiente conjugado padrão para as RNAs (MOREIRA; FIESLER, 1995). Como somente uma breve descrição é apresentada aqui, aconselha-se a leitura de (MØLLER, 1993) para uma derivação completa e para a

descrição de todos os princípios. Nesta subseção, o vetor contendo o somatório dos vetores gradientes negativos para todos os padrões na época  $j$  ( $-\nabla J(\theta_j)$ ) é representado por  $r_j$ .

1. Inicialização – o vetor de pesos  $\theta_l$  é inicializado. A direção inicial  $p_l = r_l = -\nabla J(\theta_l)$  para  $j=1$ .
2. A busca unidimensional é realizada para encontrar o  $\alpha_j$  que minimiza  $J(\theta_j + \alpha_j d_j)$ .
3. O vetor de pesos é atualizado:  $\theta_{j+1} = \theta_j + \alpha_j d_j$ .
4. Uma nova direção  $p_{j+1}$  é calculada. Se  $(j \bmod W) = 0$  então o algoritmo é reiniciado com  $p_{j+1} = r_{j+1}$ . Caso contrário  $p_{j+1} = r_{j+1} + \beta_j p_j$ .
5. Se o mínimo foi atingido então a busca é terminada. Caso contrário, uma nova iteração é realizada:  $j = j+1$  e o procedimento é recomeça no passo 2.

O parâmetro  $\beta_j$  é sempre calculado para forçar que a próxima direção seja conjugada. Existem diferentes fórmulas para calcular  $\beta_j$ .

Quando aplicados em funções quadráticas, o método do gradiente converge em no máximo  $W$  iterações.

Møller (1993) propôs uma variação do algoritmo do gradiente conjugado, conhecido como *Scaled Conjugate Gradient* (SCG), que evita a busca em linha (*line search*) a cada iteração utilizando uma técnica que combina o algoritmo Levenberg-Marquardt com o gradiente conjugado cujo objetivo é fazer um escalonamento do passo de ajuste  $\alpha$ .

Esta técnica utiliza um parâmetro  $\lambda$  para ajustar a hessiana garantindo que ela seja positiva em cada iteração, evitando que seja preciso realizar uma busca em linha. A idéia utilizada por Møller é estimar o termo denominado  $s_j = \nabla^2 J(\theta_j) d_j$  do método do gradiente conjugado por uma aproximação da forma

$$s_j = \nabla^2 J(\theta_j) d_j \approx \frac{\nabla J(\theta_j + \sigma_j d_j) - \nabla J(\theta_j)}{\sigma_j}, \quad 0 < \sigma_j < 1 \quad (2.52)$$

Combinando esta estratégia com a abordagem do gradiente conjugado e Levenberg-Marquardt, obtém-se um algoritmo diretamente aplicável ao treinamento de redes MLP

$$s_j = \frac{\nabla J(\theta_j + \sigma_j d_j) - \nabla J(\theta_j)}{\sigma_j} + \lambda_j d_j \quad (2.53)$$

O valor de  $\sigma$  não influi no comportamento do algoritmo. A qualidade da aproximação obtida para a hessiana é medida em cada iteração. Assim, se a qualidade medida não é satisfatória, o parâmetro  $\lambda$  é ajustado e o cálculo do tamanho do passo e o ajuste de pesos não são realizados.

O parâmetro  $\beta_j$  para o SCG é calculado segundo

$$\beta_j = \frac{\|r_{j+1}\|^2 - r_{j+1}^T r_j}{\mu_j} \quad (2.54)$$

O SCG apresenta algumas vantagens em relação a outros algoritmos de segunda ordem:

- não possui nenhum parâmetro crítico dependente de definição por parte do usuário;
- nenhum procedimento de busca em linha é necessário. Estes procedimentos são extremamente custosos computacionalmente e dispensáveis no SCG.

## 2.5 Estado da Arte

Como estado da arte foram relacionadas aplicativos utilizados no processamento de imagens e também na recuperação de informação visual.

### 2.5.1 Ferramentas de Processamento de Imagens

Os aplicativos Khoros, Matlab, *Machine Vision Toolbox* e IRENE estão brevemente descritos abaixo.

O ambiente **Khoros** é proposto como uma ferramenta para desenvolvimento de aplicações voltadas ao processamento de imagens. A estrutura do ambiente disponibiliza diversos módulos, denominados *toolbox*, onde são implementadas rotinas próprias de processamento de imagens, como por exemplo, operações aritméticas, filtros, conversões, transformadas, operações sobre histogramas, entre outras.

Além da possibilidade de desenvolvimento de protótipos e aplicações, o Khoros possui ferramentas para a confecção de *toolbox*.

Uma virtude deste sistema é a facilidade de obtenção. O Khoros é um software para ambiente Unix, de livre acesso para pesquisadores, sendo que o código fonte está disponível via ftp.

O **Matlab** é o núcleo de um ambiente de computação numérica baseado em matrizes que integra: funções de tratamento numérico de alta performance, sofisticados recursos de geração de gráficos para visualização de dados e uma poderosa linguagem de programação de alto nível.

Existem diversos módulos, chamados *toolbox*, dentre os quais *toolbox* de Processamento de Imagens onde estão implementadas funções de filtragem, transformadas, detectores de arestas, conversões, etc.

Possui o inconveniente de ser um software pago e que necessita também que os *toolbox* sejam adquiridos. Existem *toolbox* na Web, os quais podem ser utilizados sem custo.

O **Machine Vision Toolbox** oferece funções que são úteis em visão computacional, visão de máquina e áreas relacionadas. É uma coleção que reflete o interesse de seu criador em áreas como fotometria, colorimetria e também em filtragem. Possui funções para leitura e escrita de imagens em arquivos, filtragem, segmentação, calibração de câmera, exibição, conversão de espaços de cor, etc. Este *toolbox* é superior ao *toolbox* de processamento de imagens do Matlab quando nos referimos ao tratamento de imagens coloridas e também à utilização de detectores de arestas. É disponível gratuitamente na Web em <http://www.cat.csiro.au/cmst/staff/pic/vision-tb.html>.

O **IRENE** (Implementação de Redes Neurais) foi proposto em (ENGEL; NUNES, 1994) e sua idéia inicial era desenvolver um sistema modular que desempenhasse as funções básicas necessárias para o tratamento de imagens multiespectrais e que permitisse futuras extensões. Foram implementados 15 “canais” intercambiáveis que podem armazenar 15 imagens, antes e depois de processadas.

O IRENE possui uma interface com as funções básicas de manipulação de imagens: carregamento e salvamento, chaveamento entre bandas, zoom, etc. Além disso, funções de processamento de imagem em baixo nível, como extração de bordas, controles do nível de brilho e contraste, equalização de histogramas e segmentação por quadrees também foram implementadas.

Buscou-se também a capacidade de criação e simulação de estruturas neurais, sendo implementados os módulos da rede de Kohonen e dos classificadores LVQ. No entanto, novos módulos são facilmente “plugáveis” ao sistema. Esta flexibilidade de criação inclui a definição do número de neurônios, fatores de aprendizado e vizinhança, topologia da malha de neurônios e número de iterações (NUNES, 1995).

### 2.5.2 Ferramentas para Recuperação de Informação Visual

Os descritores de forma implementados neste trabalho são utilizados para a recuperação de informação visual em ferramentas como o QBIC, IRIS, PICASSO, Photobook e o NETRA.

O **QBIC** (*Query By Image Content*) desenvolvido pela IBM talvez seja o software de recuperação de informação visual mais conhecido na atualidade.

Partindo-se do fato de que as coleções de imagens online estão crescendo rapidamente e que ferramentas para gerenciar, organizar e navegar essas coleções eficientemente são necessárias, a IBM desenvolveu o sistema QBIC que é capaz de realizar consultas em grandes bases de imagens. Esse sistema é capaz de trabalhar com imagens estáticas e também com vídeos. Para imagens estáticas, funções de consulta por similaridade global de cor, local de cor, textura, por forma e relacionamento espacial estão disponíveis. Outras formas de consulta também estão disponíveis, como por esboço (FALOUTSOS et. al., 1994; BIMBO, 1999). Este software é comercial.

O sistema **IRIS** (*Image Retrieval for Information Systems*) foi um projeto conjunto entre o German Software Development Laboratory da IBM e o grupo de Inteligência Artificial da Universidade de Bremen. Dispõe de funções para: consulta através de conceitos de alto-nível, similaridade de cor global, similaridade por textura e similaridade de forma (HERMES et. al., 1995; BIMBO, 1999).

**PICASSO** foi desenvolvido na Universidade de Florença, Itália para recuperação de imagens estáticas. As funções disponíveis são: consulta por similaridade de cor global, similaridade de cor por região, similaridade de textura, similaridade de forma e partes da forma. Este sistema em particular foi desenvolvido para bases de dados de arte (BIMBO, 1999).

**Photobook** é um conjunto de ferramentas iterativas para navegar e buscar imagens. A idéia principal desta ferramenta é comprimir a imagem preservando sua semântica, o que reduz a imagem a um pequeno conjunto de coeficientes significativamente perceptíveis. O Photobook consiste de três sub-livros. Eles são o *Appearance Photobook*, *Shape Photobook* e *Texture Photobook*. Usuários podem consultar uma imagem baseado nas feições que correspondem a cada um dos três sub-livros ou através da combinação dos diferentes mecanismos com uma descrição textual (BIMBO, 1999; PENTLAND; PICARD; SCLAROFF, 1995).

**NETRA** é um sistema protótipo para a recuperação de imagens desenvolvido em Java onde são utilizadas feições de cor, textura, forma e localização espacial em regiões da imagem segmentada para buscar e recuperar regiões similares da base de dados.

Neste sistema foi incorporado um algoritmo automático e robusto para a segmentação de imagens que permite a busca por regiões. A segmentação de imagem aumenta significativamente a qualidade da recuperação quando as imagens contém múltiplos objetos complexos (MA; MANJUNATH, 1999).

### 3 DESCRIÇÃO DO TOOLBOX E ANÁLISE DOS DESCRITORES DE FORMA

A ênfase deste trabalho está nas etapas de Extração de Feições e na Classificação as quais fazem parte de um sistema de reconhecimento de imagens (Figura 2.1).

As feições de cor, forma e textura são as mais utilizadas para a Extração de Feições. Neste trabalho enfatiza-se a forma, representada pelos Descritores de Forma detalhados na Seção 2.3. Conforme já apresentado, estes descritores são divididos quanto ao seu critério de similaridade podendo ser por contorno ou região (Figura 3.1).

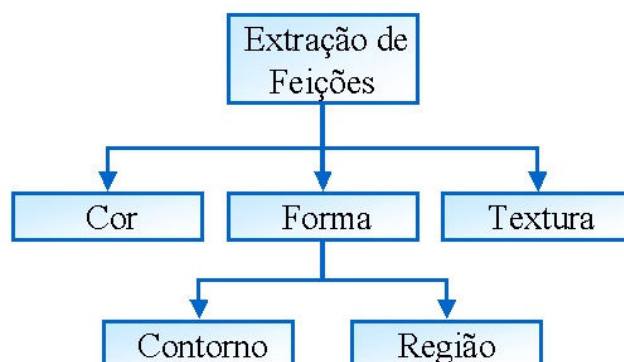


Figura 3.1: Detalhamento da etapa de Extração de Feições

Apesar de aparentemente simples a decisão de empregar um descritor com similaridade baseada em contorno ou região não é uma tarefa trivial. Por esta razão, propõe-se a realização de uma análise preliminar do conjunto de imagens que se deseja classificar quanto ao seu agrupamento pelo critério de similaridade dos descritores disponíveis.

O *toolbox* desenvolvido possui diversas funções implementadas, permitindo que esta análise seja realizada de uma maneira bastante eficiente. As funções estão documentadas, possibilitando um melhor aproveitamento e um ganho de produtividade por parte de pesquisadores e estudantes das áreas de reconhecimento de padrões através de formas e recuperação de informação visual.

Visando um melhor entendimento das funcionalidades do *toolbox*, de seu modo de operação e também para auxiliar no entendimento das características dos descritores com similaridade baseada em contorno e região realizou-se um pequeno experimento. A Figura 3.2 empregada neste experimento foi apresentada e comentada anteriormente na Seção 2.3. As formas desta figura foram processadas através de dois descritores

clássicos, sendo um baseado em contorno (Descritores de Fourier) e outro baseado em região (Momentos Invariantes). As nove imagens existentes nesta figura foram isoladas e nomeadas recebendo o prefixo “img” acrescido de um número que indica sua posição na imagem original. A primeira coluna recebeu números de 01 até 03, a segunda coluna de 04 até 06 e a terceira coluna de 07 até 09. Após, as imagens foram armazenadas a fim de serem processadas pelas funções desenvolvidas para o *toolbox*.

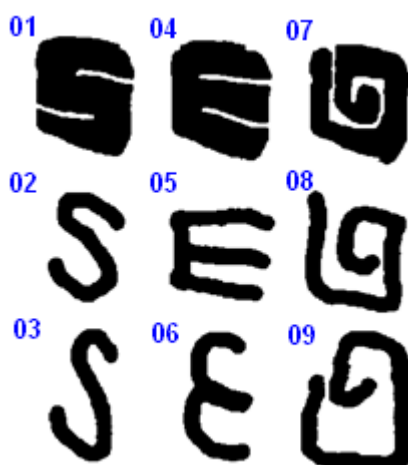


Figura 3.2: Formas utilizados no experimento

Cada uma das imagens deve ser carregada utilizando-se a função *imread* própria do *toolbox* de Processamento de Imagens do Matlab. Após a leitura da imagem, uma função (Figura 3.3) responsável pela chamada dos descritores de forma implementados é executada com o intuito de verificar a integridade dos parâmetros recebidos, fazendo ajustes quando possível ou indicando o erro e interrompendo o processamento. Caso não ocorram erros, ou seja, caso a validação dos parâmetros ocorra de maneira adequada, então, realiza-se a extração das feições de forma.

***d = descriptors(I,method, param)***

***Parâmetros de entrada:***

- *I*: imagem que deverá ser processada;
- *method*: descritor de forma que será empregado;
- *param*: número de coeficientes, exibição de imagens, etc.

***Parâmetro de saída:***

- *d*: descritores extraídos para a imagem *I*

Figura 3.3: Apresentação da função “*descriptors*”

Os descritores de forma desenvolvidos neste trabalho podem ser executados diretamente sem a utilização da função *descriptors*, entretanto, aconselha-se seu emprego pois todos os parâmetros são verificados garantindo assim uma maior

confiabilidade dos resultados. Em suma, *descriptors* deve ser utilizada para validar os parâmetros recebidos e invocar os descritores. Seus parâmetros estão apresentados na Figura 3.3.

Conforme descrito na Seção 2.3, foram implementados os seguintes descritores: Momentos Invariantes Afins, Momentos Invariantes, Descritores de Fourier, *Curvature Scale Space*, Momentos de Zernike e Momentos Pseudo-Zernike, os quais são reconhecidos pela função *descriptors*, através de seu parâmetro *method* respectivamente por: MIA, MI, DF, CSS, ZERNIKE e PZERNIKE.

Cada um destes descritores possui parâmetros distintos de execução que serão brevemente descritos.

Os Descritores de Fourier possuem um único parâmetro indicando a quantidade de coeficientes que deverá ser utilizada.

Os Momentos Invariantes e os Momentos Invariantes Afins também possuem um único parâmetro, entretanto, estão limitados respectivamente a no máximo 7 e 4 momentos, pois estes são o número máximo de momentos de cada um dos métodos.

O *Curvature Scale Space* possui um parâmetro que objetiva prover ao método invariância à escala, o qual, indica a quantidade de pontos que o contorno deverá possuir. Além deste, um parâmetro adicional tem por função indicar se o processo de evolução da forma será exibido. Os pontos do contorno não são definidos pelo usuário e são gerados durante a execução da função *descriptors*. Entretanto, isto poderia ser realizado na própria implementação do método, mas por detalhes de implementação preferiu-se fazê-lo em separado.

Os Momentos de Zernike e Pseudo-Zernike, por sua vez, possuem quatro parâmetros. Os dois primeiros são a ordem e a repetição do polinômio os quais possuem restrições que serão verificadas pelas funções desenvolvidas. Estas restrições foram apresentadas na Seção 2.3.2.2. Os demais parâmetros referem-se à exibição do processo ou da apresentação da animação com a reconstrução da forma adicionando-se os momentos de altas ordens.

Estas funções foram descritas brevemente, mas sua implementação e especificação poderão ser verificadas livremente pelo usuário do *toolbox*.

Um comentário em relação à documentação das funções desenvolvidas se faz necessário antes de dar prosseguimento à descrição do *toolbox*. O padrão de desenvolvimento Matlab foi seguido, no quesito documentação. Assim, utilizando-se o comando “*help*” seguido do nome da função obter-se-ão seus parâmetros de entrada, saída, bem como uma descrição de sua funcionalidade. Em alguns casos, um exemplo também é apresentado.

A partir deste momento, um experimento que faz uso das funções apresentadas, aplicadas às imagens da Figura 3.2 será discutido. Para este experimento, uma função denominada *extract* (Figura 3.4) foi desenvolvida. Esta função é responsável por buscar todas as imagens que estão na pasta corrente, dentro da subpasta “*imagens*”. Além de fazer a leitura das imagens, esta função invoca a função já descrita, *descriptors*.



**$p = \text{extract}(\text{metodo}, \text{numcoeffs})$**

**Parâmetros de entrada:**

- *metodo*: descritor de forma que será empregado;
- *numcoeffs*: número de coeficientes

**Parâmetro de saída:**

- *p*: padrões extraídos para cada uma das imagens

Figura 3.4: Apresentação da função “*extract*”

O experimento realizado faz duas chamadas à função *extract*, sendo, respectivamente, uma para os Descritores de Fourier e outra para os Momentos Invariantes (Figura 3.5).

**$pDF = \text{extract}('DF', 10);$**

**$pMI = \text{extract}('MI', 7);$**

Figura 3.5: Extração dos descritores

Assim, na variável “*pDF*” estarão armazenados os 10 primeiros Descritores de Fourier para as 9 imagens do experimento e na variável “*pMI*” estarão armazenados os 7 Momentos Invariantes também para as 9 imagens. Para uma melhor visualização, o resultado da operação realizada é apresentado em gráficos, sendo que no eixo das abscissas do gráfico o índice do descritor é apresentado e no eixo das ordenadas o respectivo valor obtido.

O gráfico apresentado na Figura 3.6 apresenta o resultado do cálculo dos momentos invariantes para cada uma das imagens da Figura 3.2. Observa-se que visualmente temos um padrão que aparentemente separa algumas imagens de outras, principalmente nos momentos de ordem mais elevada. Nestas figuras, as linhas unindo os pontos são utilizadas apenas para agrupar os valores dos descritores de cada imagem.

O primeiro agrupamento é formado pelas *img01*, *img04* e *img07*. Já o segundo agrupamento é composto pelas *img02*, *img05* e *img08*. Por fim, *img03*, *img06* e *img09* formam o último agrupamento.

Já na Figura 3.7 o resultado do cálculo dos Descritores de Fourier é apresentado e aparentemente existe um maior entrelaçamento entre os valores obtidos para cada um dos caracteres.

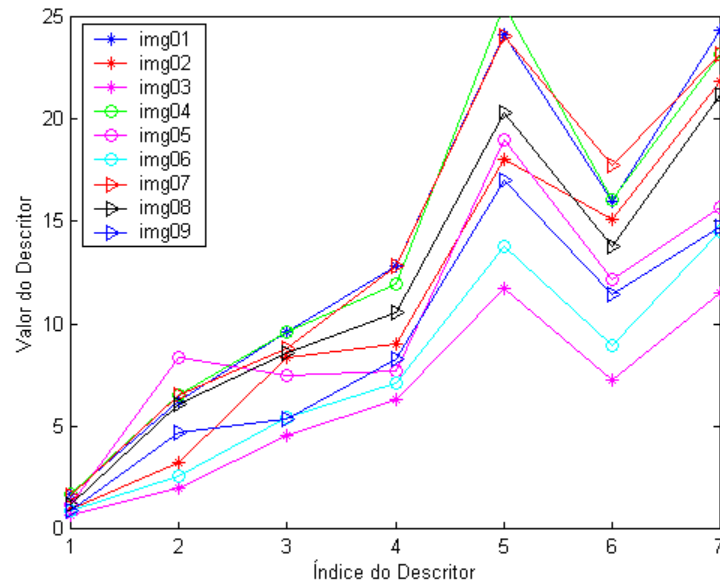


Figura 3.6: Os 7 Momentos Invariantes para as formas da Figura 3.2

Analisando-se as duas figuras lado-a-lado tem-se a impressão de que o método dos Momentos Invariantes apresenta-se mais adequado para o conjunto de dados em questão, separando-o mais eficientemente. Entretanto, após a submissão dos resultados de ambos os descritores a um clusterizador, obteve-se 3 agrupamentos que trazem uma nova perspectiva.

O algoritmo para a clusterização empregado foi o *k-means* implementado no *Clustering Toolbox*, disponível gratuitamente (CORNEY, 2003).

As Figuras 3.8-3.10 apresentam, respectivamente, os resultados para o primeiro, segundo e para o terceiro agrupamento utilizando-se os Momentos Invariantes.

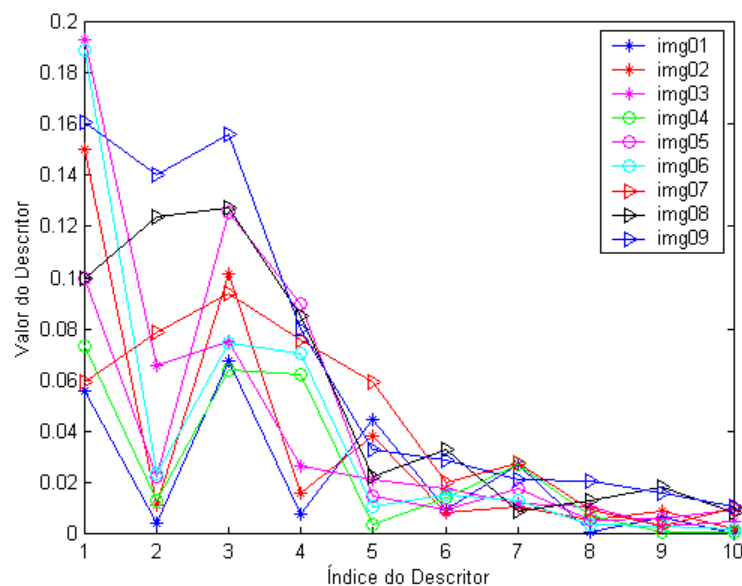


Figura 3.7: Os 10 Descritores de Fourier para as formas da Figura 3.2

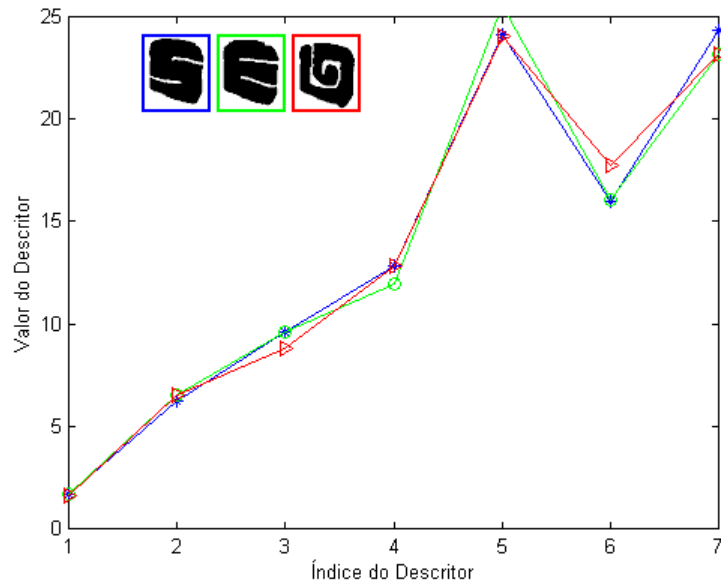


Figura 3.8: Momentos Invariantes – primeiro agrupamento

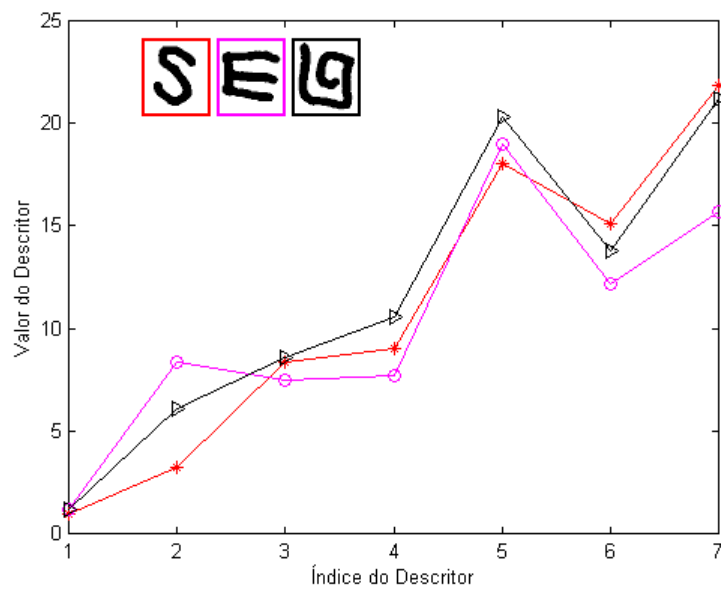


Figura 3.9: Momentos Invariantes – segundo agrupamento

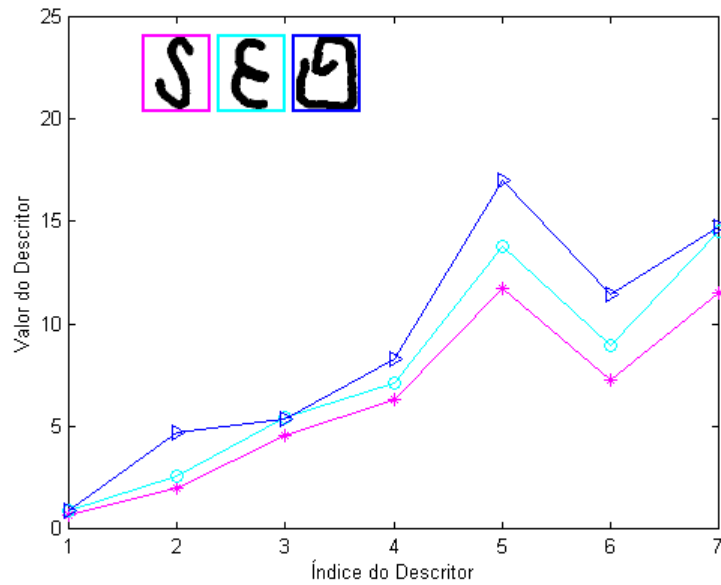


Figura 3.10: Momentos Invariantes – terceiro agrupamento

Através da análise destes resultados, é possível verificar que os Momentos Invariantes foram eficazes na caracterização das imagens com distribuição de pixels semelhante. Entretanto, antes que qualquer nova afirmação possa ser realizada, é importante observar-se os resultados obtidos com o método com similaridade baseada em contorno.

As Figuras 3.11-3.13 referem-se aos Descritores de Fourier onde novamente temos 3 agrupamentos, sendo que a Figura 3.11 refere-se ao primeiro, a Figura 3.12 ao segundo e a Figura 3.13 ao terceiro agrupamento.

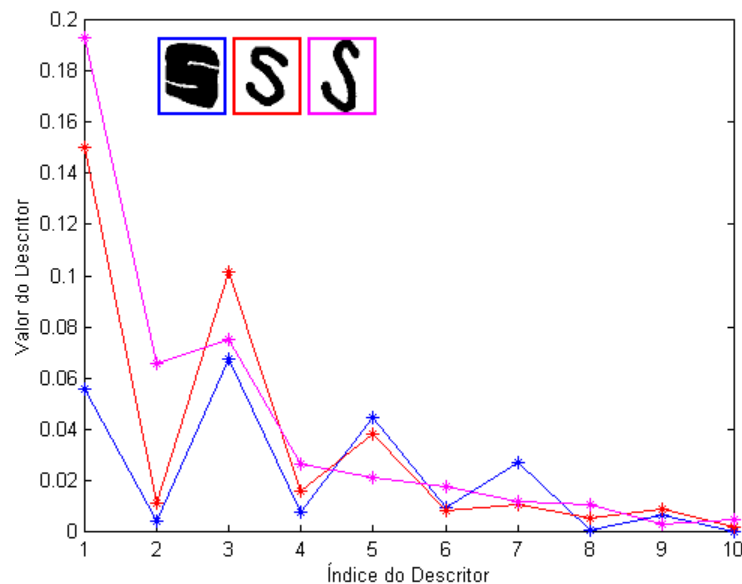


Figura 3.11: Descritores de Fourier – primeiro agrupamento

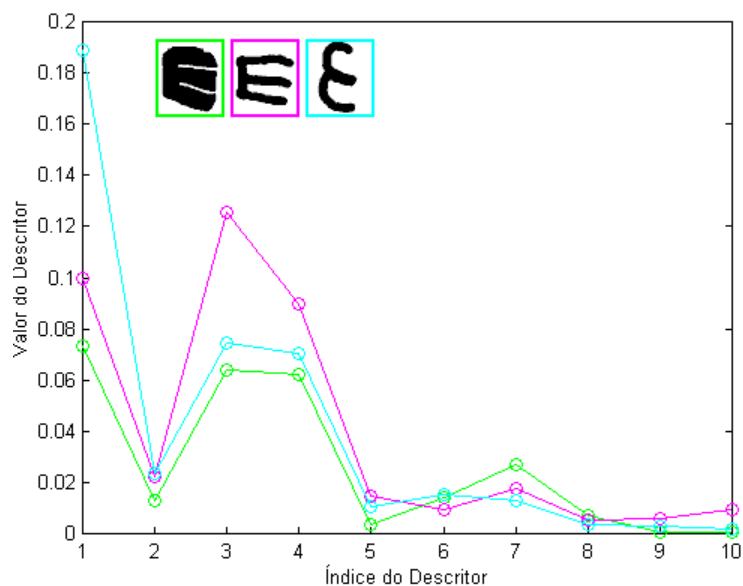


Figura 3.12: Descritores de Fourier – segundo agrupamento

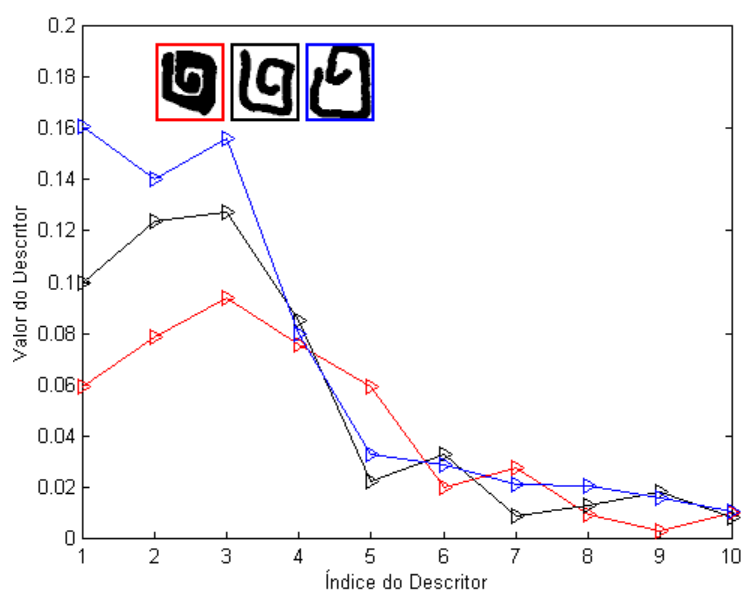


Figura 3.13: Descritores de Fourier – terceiro agrupamento

Analisando-se as imagens dos agrupamentos gerados para o método com similaridade baseado em contorno verificamos que o comportamento dos descritores que não podia ser visualizado eficazmente na Figura 3.7 foi explicitado pela geração dos agrupamentos das Figura 3.11-3.13. Nestas, podemos constatar pelo comportamento dos picos uma característica altamente desejada, caracterizada por uma baixa variação intra-classe e uma alta variação inter-classe.

Observa-se através da análise das imagens dos agrupamentos de ambos os métodos que a similaridade baseada em contorno pode ser empregada com uma expectativa de sucesso mais elevada quando os contornos forem conexos. Por outro lado, métodos com similaridade baseada em região serão mais indicados se possuímos regiões desconexas ou se nossas imagens possuírem problemas oriundos de uma etapa de segmentação não

adequada. Percebemos então, que dependendo da natureza da aplicação, um método poderá ser mais indicado que outro.

Através deste experimento, é possível verificar as sutilezas existentes na utilização de formas em imagens e também verificar a importância de um trabalho desta natureza, pois os resultados obtidos para cada abordagem são extremamente distintos. Assim sendo, deve-se ter certeza de qual é o objetivo que se deseja alcançar e então, decidir se serão utilizados métodos baseados em contorno ou em região. A análise de agrupamentos pode auxiliar nesta decisão e é uma alternativa que deve ser melhor explorada. Provavelmente em experimentos mais complexos a análise de agrupamentos não obterá resultados tão exatos quanto os obtidos neste experimento, mas os mesmos deverão ser suficientemente relevantes para reduzir o tempo de decisão do critério de similaridade que será adotado. Assim, após esta decisão parte-se para uma nova decisão, que é a de avaliar qual dos métodos será utilizado dentre as inúmeras opções disponíveis.

O *toolbox* aqui desenvolvido, deve auxiliar na tomada destas decisões e também acelerar o processo de desenvolvimento da aplicação em sua totalidade, possibilitando, por exemplo, um melhor estudo sobre as técnicas de segmentação e/ou pré-processamento requeridas.

Deve-se ressaltar que o resultado do agrupamento dos descritores, confirma que os Momentos Invariantes formam classes de objetos similares por região e os Descritores de Fourier formam classes de objetos similares por contorno, conforme já previsto na Seção 2.3.

Após a apresentação das características dos descritores de forma baseados em região e contorno, retoma-se a explicação das funções do *toolbox*. As funções que serão apresentadas referem-se a próxima etapa apresentada na Figura 2.1, denominada Classificação.

Pelo fato de se empregarem classificadores supervisionados neste *toolbox* faz-se necessário a geração dos conjuntos de treinamento e teste. Para tal, deve-se definir quais são as imagens pertencentes a cada uma das classes e então dividi-las respectivamente em conjunto de treinamento e conjunto de teste. Neste trabalho, isto é feito através de rotinas específicas para cada um dos estudos de caso e desta forma, não estarão descritos no texto. Entretanto, as rotinas utilizadas nos estudos de caso, mesmo que específicas, estarão disponíveis para estudo. Estas rotinas foram desenvolvidas propositadamente de maneiras diferentes nos dois estudos de caso a fim de aumentar a facilidade de entendimento, sendo uma implementação bastante simplificada e trabalhosa e outra bem mais elaborada, proporcionando assim uma forma de compreensão incremental.

Por facilidades disponíveis no ambiente de desenvolvimento e por simplicidade de entendimento estabeleceu-se uma **célula** que contém os descritores extraídos para cada uma das classes já rotuladas. O conceito de célula em Matlab é semelhante ao registro na linguagem Pascal, entretanto, operações de criação e manipulação de estruturas desta natureza em Matlab são extremamente simples. Um exemplo da célula aplicada, denominada “*amostra*”, pode ser verificada na Figura 3.14. Nesta célula, estão definidas nove classes. Cada uma destas classes é uma nova estrutura, contendo atributos que possuem diferentes tamanhos. Isto ocorre pois nem sempre se possui a mesma quantidade de amostras para cada classe sendo este um fato bastante comum em aplicações práticas.

Para verificar, por exemplo, o que está definido para a primeira classe basta digitar no ambiente de trabalho do Matlab: `amostra{1}`. O resultado desta operação pode ser verificado na Figura 3.15.

```
>> whos amostra

Name      Size      Bytes Class
amostra   1x9       20520 cell array

Grand total is 837 elements using 20520 bytes
```

Figura 3.14: Célula “amostra” possuindo 9 classes

```
>> amostra{1}

ans =

metodo: 'DF'
      d: {1x18 cell}
```

Figura 3.15: Atributos da primeira classe da célula “amostra”

Percebemos que a estrutura é composta por dois atributos (isto é, variáveis), o primeiro, “*metodo*” no presente exemplo possui como conteúdo DF, caracterizando que os descritores ali existentes foram gerados através dos Descritores de Fourier. O segundo atributo, “*d*” é outra célula que possui os descritores extraídos para cada uma das dezoito imagens da primeira classe. A Figura 3.16 apresenta para a primeira classe (`amostra{1}`) os valores dos 5 Descritores de Fourier da primeira e segunda imagens.

```
>> amostra{1}.d{1}

ans =

0.0126  0.1216  0.0370  0.0074  0.0063

>> amostra{1}.d{2}

ans =

0.0177  0.1322  0.0311  0.0085  0.0047
```

Figura 3.16: Descritores para as primeiras duas imagens da primeira classe

Após esta breve e necessária explicação sobre o funcionamento das células no ambiente Matlab pode-se então retomar a descrição do *toolbox*. É importante salientar, que a célula “*amostra*” é preenchida com os valores obtidos com a execução da função *descriptors*, já apresentada, para cada uma das imagens de cada classe. Feito isto, utiliza-se esta estrutura para a preparação das amostras de treinamento e teste, tarefa realizada pela função *preparaAmostras*, apresentada na Figura 3.17.

Como pode ser verificado, *preparaAmostras* possui além da célula contendo os descritores já extraídos para todas as classes, um segundo parâmetro que indica o percentual do conjunto amostral que será utilizado para a etapa de treinamento (estimação) e teste. Este percentual é utilizado para cada classe, e, em casos extremos, garante que pelo menos um exemplar de cada classe seja selecionado. É importante frisar que cada classe pode possuir diferentes quantidades de amostras, requerendo então que cada conjunto de treinamento (estimação) seja tratado de maneira distinta. Assim, por exemplo, se uma classe possui 18 amostras e outra possui 55 amostras, selecionando-se 75% dos dados para o treinamento, então obter-se-ia 13 amostras para o treinamento da primeira classe e 41 amostras para a segunda classe.

O terceiro parâmetro de *preparaAmostras* pode ser “*neural*” ou “*estatístico*”. Esta distinção se faz necessária pois os dados das saídas requeridas por um classificador neural e um classificador estatístico são bastante diferentes. Além disso, é importante que os dados apresentados para um classificador neural estejam ordenados aleatoriamente, objetivando um melhor desempenho do classificador.

Pode-se verificar então, a relevância desta função para o *toolbox*, pois a mesma centraliza e gerencia várias características pertencentes a cada uma das abordagens de classificação.

***[pTreino,pTeste,tTreino,tTeste] =  
preparaAmostras(amostra,percentual,classificador)***

***Parâmetros de entrada:***

- *amostra: estrutura contendo os descritores;*
- *percentual: % dos dados que será utilizado para o treinamento (estimação);*
- *classificador: indica se os dados serão utilizados por um classificador ‘neural’ ou ‘estatístico’*

***Parâmetro de saída:***

- *pTreino: padrões utilizados para o treinamento (estimação);*
- *pTeste: padrões utilizados para o teste;*
- *tTreino: saídas desejadas para o treinamento (estimação);*
- *tTeste: saídas desejadas para o teste.*

Figura 3.17: Apresentação da função “*preparaAmostras*”

Neste instante, estamos aptos a descrever as funções de classificação neural e estatística. A classificação neural é realizada pela função *mlp* que engloba a chamada aos classificadores neurais do *toolbox* de Redes Neurais Artificiais do Matlab. Esta função é apresentada na Figura 3.18.



Os padrões para o treinamento, teste e as saídas desejadas são obtidos da função *preparaAmostras*. Dentre os parâmetros gerais de uma RNA, apenas o número de neurônios é definido pelo usuário, os demais foram definidos experimentalmente e estão na própria função *mlp*. O modelo de RNA pode ser um dentre os seguintes: GDX, LM, RP e SCG.

Esta função faz a chamada as funções do *toolbox* de Redes Neurais Artificiais do Matlab e desta forma, pode ser substituída sem nenhum comprometimento pelas referidas funções. Entretanto, visando aumentar a facilidade de utilização das RNAs por parte de usuários não-especialistas, definiu-se pela criação da função *mlp* visando garantir um resultado se não ótimo pelo menos satisfatório. Assim, alguém que nunca utilizou RNAs poderia, utilizando esta função simplificada, obter resultados satisfatórios com as técnicas e assim que estivesse familiarizado poderia explorar com maior profundidade os classificadores neurais e então, definir seus parâmetros ou aplicar outros algoritmos disponíveis no próprio *toolbox* Matlab.

***[net,tr] = MLP(p,t,neuron,modelo)***

***Parâmetros de entrada:***

- *p*: padrões para o treinamento;
- *t*: saídas desejadas;
- *neuron*: número de neurônios da camada oculta;
- *modelo*: algoritmo de treinamento da RNA

***Parâmetro de saída:***

- *net*: estrutura contendo a configuração e os pesos da RNA após o treinamento;
- *tr*: parâmetro opcional que mostra o comportamento da RNA em cada época.

Figura 3.18: Apresentação da função “*mlp*”

Por fim, a classificação estatística faz uso da mesma função *preparaAmostras*, já citada e, após, realiza-se a estimação dos parâmetros, média e covariância. Para realizar esta estimação, utiliza-se a função *mml*, apresentada na Figura 3.19.

Conforme já apresentado na Seção 2.4.1, as estimações devem ser realizadas para cada classe e então, através da aplicação das funções de decisão, decide-se pela de maior valor. Assim, *mml* é chamado para obter-se o vetor média e a matriz de covariância para cada uma das classes. Esta informação é armazenada em uma célula contendo também o rótulo da classe. Esta informação deve ser armazenada manualmente, pois nenhuma rotina foi implementada para tal tarefa. Para o classificador DM a matriz de covariância é substituída pela matriz identidade. Realizada a etapa de estimação, parte-se para a etapa de testes, através da função *classifica*, apresentada na Figura 3.20.

***[media,covariancia] = mmle(p,nomeClass)***

***Parâmetros de entrada:***

- *p*: padrões para a estimação;
- *nomeClass*: nome do classificador, podendo ser, *MVG* ou *DM*

***Parâmetro de saída:***

- *media*: média para os padrões da referida classe;
- *covariancia*: matriz de covariância para os padrões da referida classe.

Figura 3.19: Apresentação da função “*mmle*”

***resultado = classifica(estimacao,p, prob)***

***Parâmetros de entrada:***

- *estimacao*: célula contendo os atributos (rotulo, media e covar);
- *p*: padrões para a estimação;
- *prob*: probabilidade a priori

***Parâmetro de saída:***

- *resultado*: vetor contendo os rótulos das classes de cada padrão.

Figura 3.20: Apresentação da função “*classifica*”

A função *classifica* recebe a célula contendo o rótulo, a média e a covariância de todas as classes e então faz as comparações necessárias para obtenção da classe que possui o maior valor para a função de decisão. A probabilidade *a priori* para os experimentos foi definida em 50% para todos os casos.

As explicações dadas neste capítulo visam propiciar um melhor entendimento das várias funções desenvolvidas, sendo que diversas delas não foram descritas por terem sido englobadas pelas funções principais, as quais foram descritas e comentadas.

Buscou-se com o desenvolvimento deste *toolbox*, reduzir a quantidade de funções necessárias para que o usuário trabalhe eficazmente sem a constante preocupação com a utilização de dezenas de funções disponíveis. Obviamente, o usuário poderá, caso seja de sua vontade, utilizar as funções puras, sem as funções de centralização e gerenciamento, mas arcando com os prováveis problemas causados por isso. Também é permitido ao usuário explorar as funções desenvolvidas em sua totalidade e então, modificá-las a fim de melhorá-las ou entendê-las.

É altamente recomendável que o *Toolbox de Descrição de Forma* seja aprimorado com novas funções, baseadas ou não nas já existentes. Correções de eventuais “bugs” devem ser realizados a fim de criar um pacote cada vez mais completo e robusto.

Os principais descritores de forma existentes estão implementados o que já provê ao interessado um excelente ambiente de estudo, coisa que até então, não estava disponível.

Os estudos de caso descritos no próximo capítulo servirão também de base de estudos, pois suas implementações serão disponibilizadas junto ao *toolbox*.

## 4 ESTUDOS DE CASO

Este capítulo descreve em detalhes as etapas requeridas pelos estudos de caso desenvolvidos.

O primeiro estudo de caso refere-se aos comandos: **pare**, **ande**, **dir** e **esq**. As imagens pertencentes a este estudo de caso são apresentadas e utilizadas com o intuito de avaliar o comportamento dos descritores de forma em associação aos classificadores. Neste estudo de caso, o processo é desenvolvido na íntegra, desde a localização dos retângulos até a identificação dos caracteres dos comandos neles contidos. Com isso, objetiva-se avaliar quais combinações são mais promissoras e então utilizá-las no segundo estudo de caso, mais complexo.

Placas de automóveis são utilizadas no segundo estudo de caso. Visando cobrir muitos dos problemas encontrados em aplicações reais de reconhecimento de placas de automóveis, criou-se um banco de imagens possuidor de uma grande variedade de situações do cotidiano. Assim, neste banco de imagens encontram-se placas com diversas escalas e graus de rotação, iluminação não-uniforme, desgaste nas informações das placas e também, variações de cores, oriundas da atividade a que se destina o veículo.

Em razão destas características, um conjunto de métodos que prove sua eficiência no reconhecimento das imagens de tal banco poderá ser utilizado em uma vasta gama de aplicações, pois cobrirá a grande maioria das situações reais encontradas se não a sua totalidade. Assim, os mesmos poderão ser plenamente empregados às aplicações de controle de segurança ou mais especificamente, às aplicações de controle de acesso a condomínios e garagens e aos controladores eletrônicos de velocidade.

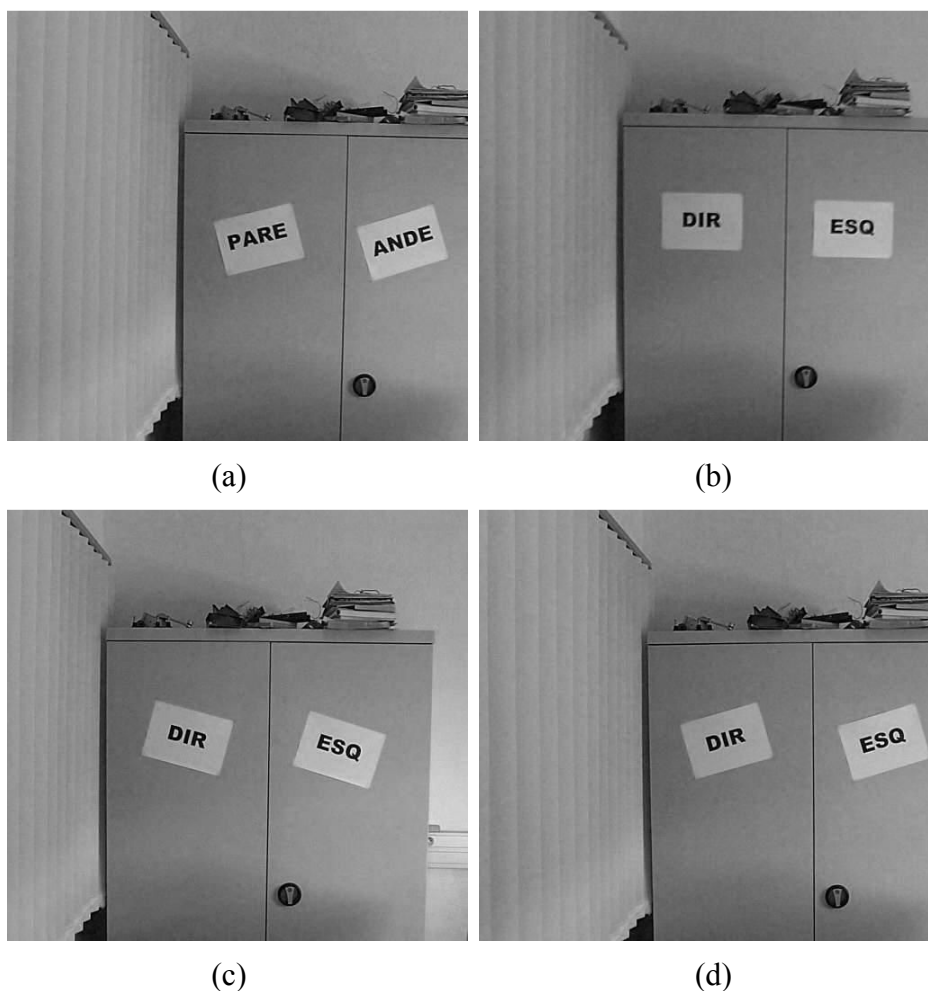
É importante realçar que os sistemas de reconhecimento de placas de automóveis são desenvolvidos com um propósito único, ou seja, para uma aplicação distinta, possuindo características que os diferem de outras aplicações que também fazem uso de placas de automóveis. Obviamente, isto possibilita que o mesmo obtenha um performance superior à obtida por um sistema de propósito geral, como o deste estudo de caso.

Após esta breve introdução, os estudos de caso propriamente ditos serão descritos sendo que todas as etapas do processo serão detalhadas de uma maneira que se avalia suficiente para o entendimento do problema em questão.

#### 4.1 Estudo de Caso - Comandos

O estudo de caso inicial trata do reconhecimento dos caracteres dos comandos específicos em um ambiente restrito. Para realizar tal tarefa, têm-se a necessidade de localizar os retângulos onde os comandos estão contidos. É importante observar que o número de retângulos dispostos no ambiente é desconhecido sendo então necessário recuperá-los através de um conjunto de regras. Após a localização de cada retângulo, deve-se localizar os comandos em seu interior e então segmentá-los, visando o isolamento de cada um dos caracteres existentes. A partir deste momento, realiza-se a etapa de reconhecimento.

O conjunto de imagens utilizado é apresentado na Figura 4.1, sendo que as 12 imagens originalmente estão armazenadas em níveis de cinza, possuindo 512 x 480 pixels.





(e)



(f)



(g)



(h)



(i)



(j)

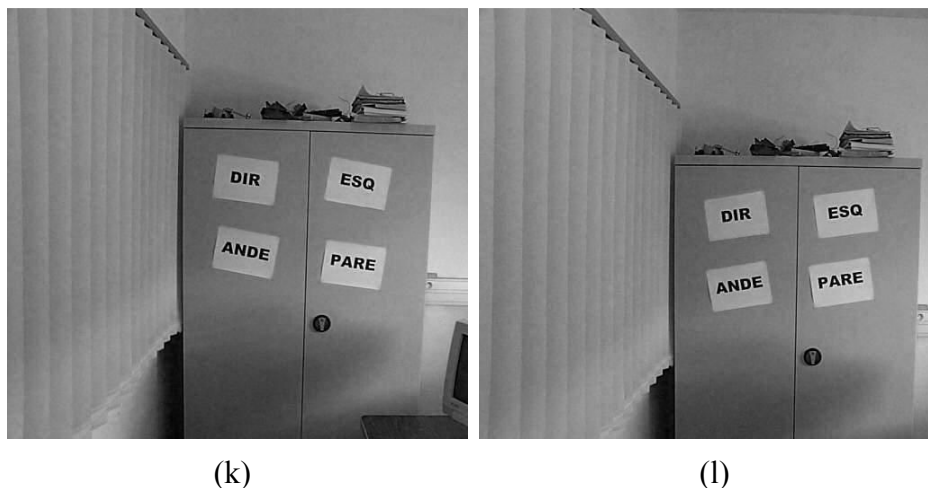


Figura 4.1: Imagens utilizadas para o Estudo de Caso dos Comandos

A localização de retângulos é realizada através do algoritmo descrito em (MOLZ, 2001) o qual é facilmente implementado em *hardware*. Após a localização do retângulo passa-se a buscar os caracteres individuais a fim de gerar sua representação. Isto é feito através de um algoritmo de crescimento de regiões, típico da área de Processamento de Imagens Digitais. A seguir, detalham-se as etapas de Localização dos Retângulos e Isolamento dos Caracteres.

#### 4.1.1 Localização dos Retângulos

Conforme já citado, a tarefa de localização dos retângulos utilizada foi proposta em (MOLZ, 2001) sendo apenas acrescentada a filtragem pela mediana (Figura 4.2).

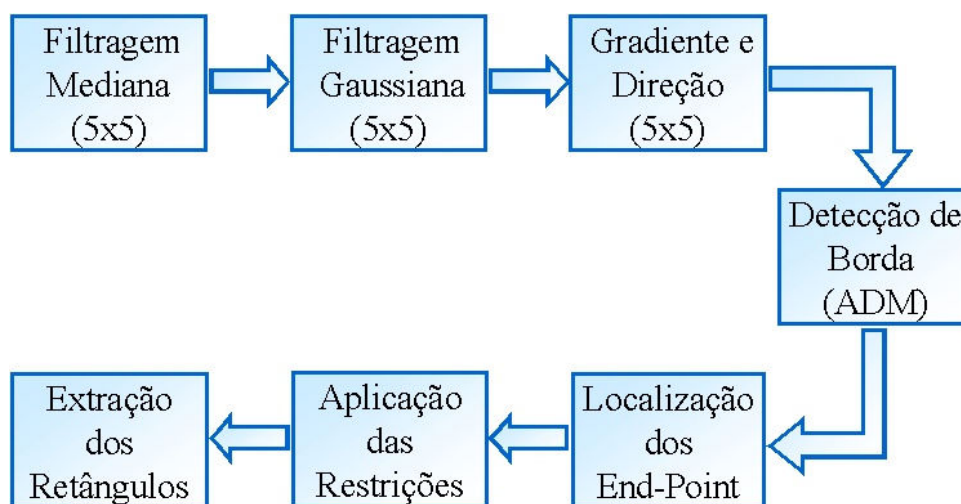


Figura 4.2: Tarefas para a localização dos retângulos

O filtro da mediana é uma solução eficiente para alcançar a redução de ruído sem borramento. Isto acontece pois o nível de cinza de cada pixel é substituído pela mediana dos níveis de cinza na vizinhança daquele pixel.

A mediana  $m$  de um conjunto de valores é tal que metade dos valores do conjunto são menores do que  $m$  e metade dos valores são maiores do que  $m$ . Para calcular a filtragem por mediana em uma vizinhança de um pixel  $P$  (Figura 4.3 (a)), primeiramente selecionamos os valores do pixel e de seus vizinhos, determinamos a mediana e atribuímos este valor ao pixel (GONZALEZ; WOODS, 2000). Por exemplo, em uma vizinhança 3x3, a mediana é o 5º maior valor, já em nosso caso, a vizinhança é 5x5 (Figura 4.2) sendo que a mediana é dada pelo 13º maior valor.

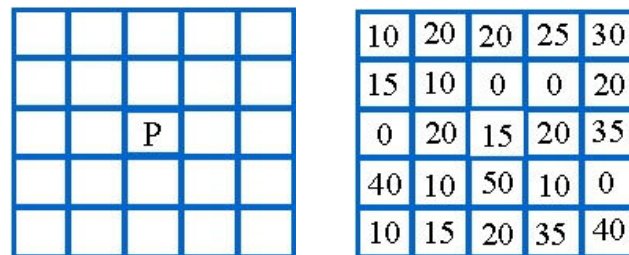


Figura 4.3: (a) vizinhança 5x5 de P (b) níveis de cinza

Para o caso da Figura 4.3 (b), onde vários valores se repetem, devemos ordená-los da seguinte forma (0; 0; 0; 0; 10; 10; 10; 10; 10; 15; 15; 15; 20; 20; 20; 20; 20; 20; 25; 30; 35; 35; 40; 40; 50) e então, selecionar a 13ª posição que possui o valor 20. Assim, a função principal da filtragem por mediana é forçar pontos com intensidades distintas a assemelharem-se a seus vizinhos, efetivamente eliminando intensidades que apareçam isoladas. Este tipo de filtro é utilizado para filtragem do ruído *speckle*.

O segundo filtro utilizado foi o gaussiano, com máscara (0,25; 0,5; 0,5; 0,5; 0,25; 0,5; 0,75; 1; 0,75; 0,5; 0,5; 1; 2; 1; 0,5; 0,5; 0,75; 1; 0,75; 0,5; 0,25; 0,5; 0,5; 0,5; 0,25). Este filtro atua como uma função suavizadora, devendo ser utilizado principalmente para minimizar os ruídos provenientes do processo de digitalização da imagem e da introdução de sombras e luzes que diferem de uma imagem para outra (MOLZ, 2001).

As tarefas pertencentes à segmentação responsáveis pela detecção de bordas, extração de linhas e detecção dos retângulos são descritas com maior detalhamento em (MOLZ, 2001).

Molz (2001) visava a implementação em *hardware* e desta forma não utilizou nenhum dos métodos tradicionais para a detecção de bordas, tais como Sobel, Prewitt e Canny (GONZALEZ; WOODS, 2000; JAIN, 1989), fazendo uso, então, de um algoritmo denominado *Absolute Difference Mask* (ADM) (ALZAHIRANI; CHEN, 1997). Este método realiza a detecção de bordas utilizando o método do gradiente e da direção que cada pixel possui em relação a uma dada região de vizinhança. O resultado da aplicação do processo a uma imagem é mostrado nas Figuras (4.4-4.8).





Figura 4.4: Imagem original



Figura 4.5: Imagem original após filtro da mediana



Figura 4.6: Imagem após filtragem gaussiana

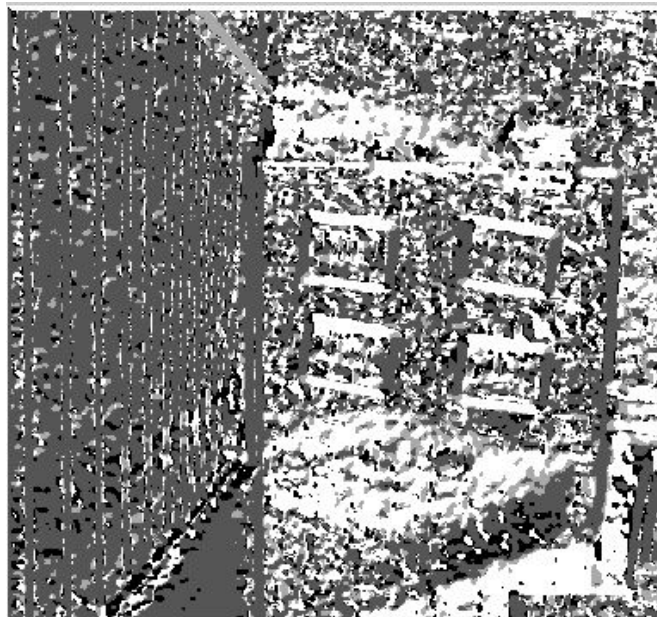


Figura 4.7: Imagem com direções calculadas

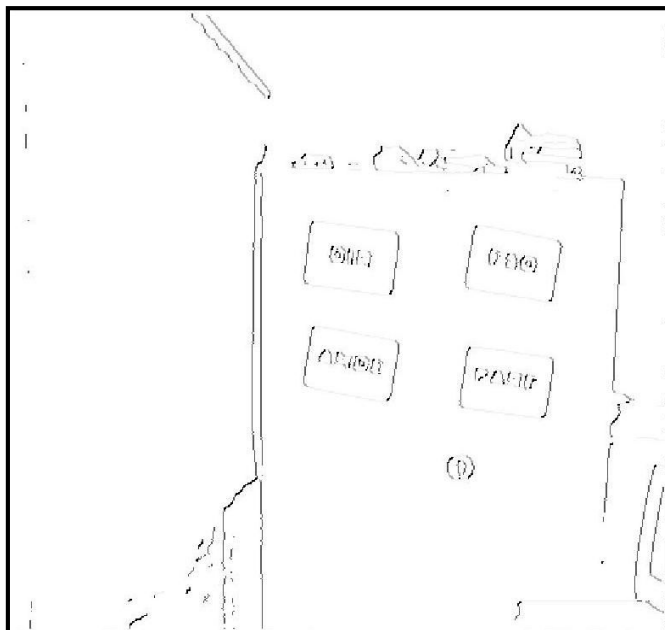


Figura 4.8: Imagem com bordas detectadas e com espessura de 1 pixel

Os processos descritos até este ponto não localizam as linhas. O algoritmo somente detecta a existência de linhas (bordas), com alguma dimensão e algum ângulo dentro da imagem. Para se obter a localização exata destas linhas, utilizou-se o método de localização de *end-point* (COCQUEREZ et al., 1995). Este algoritmo é aplicado sobre as bordas detectadas e devidamente afinadas. Para esta tarefa ser realizada, efetua-se uma busca na imagem (Figura 4.8) de cima para baixo e da esquerda para a direita, de um pixel de uma linha que possua uma das quatro direções relativas possíveis. A seguir, a direção é perseguida, enquanto a mesma persistir, por meio da verificação de uma vizinhança 3x3. Cada pixel visitado é marcado para evitar nova comparação. Este laço persiste até não existirem mais pixels não marcados na imagem. Como resultado este algoritmo nos fornece as posições inicial e final de cada linha, sendo que com estas informações pode-se calcular o ângulo de inclinação e o comprimento de cada linha existente na imagem.

As duas próximas tarefas, Aplicação das Restrições e a Extração dos Retângulos, são necessárias para a solução dos problemas propostos como estudo de caso. Caso o problema seja diferente do que localizar formas retangulares, estas duas tarefas deverão ser substituídas pelas tarefas necessárias para a solução deste problema em particular.

A Aplicação das Restrições está relacionada com a dimensão mínima e com o ângulo máximo.

A restrição de dimensão mínima existe por causa da ocorrência de *gaps* (espaços entre pixels consecutivos) no processo de segmentação.

A restrição de ângulo máximo também é ligada à existência destes *gaps*. Tem sido verificado, por experiências, que entre  $50^\circ$  e  $130^\circ$  (exceção para  $90^\circ$ ) o processo de segmentação produz vários *gaps* em uma mesma linha dificultando as demais tarefas. Por causa desta restrição, este algoritmo não é capaz de localizar linhas cujo ângulo seja superior a  $50^\circ$  (exceção para  $90^\circ$ ).

Ao final da tarefa de aplicação destas restrições, obtém-se uma imagem com linhas que respeitam as restrições impostas. A Figura 4.9 mostra as linhas obtidas por este algoritmo sobrepostas à imagem original.

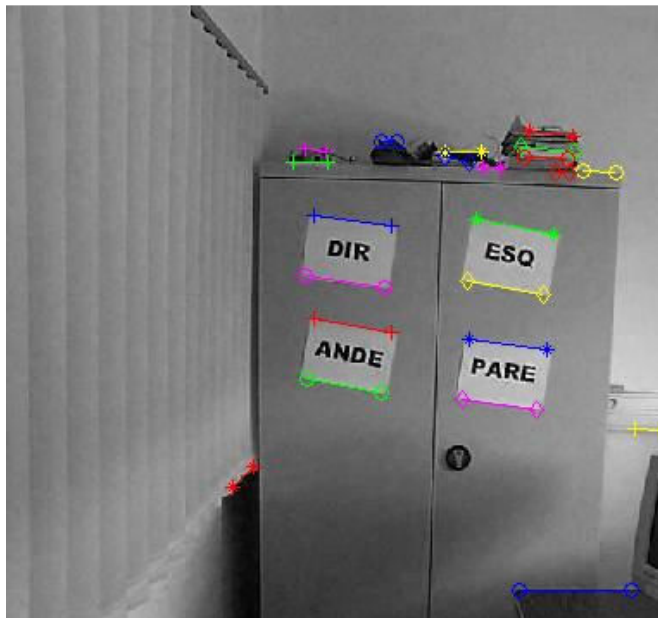


Figura 4.9: Resultado da localização de *end-point* com restrições e das respectivas linhas

Finalmente, estas linhas devem ser associadas a possíveis formas retangulares, através da procura de similaridades entre as linhas obtidas. As similaridades levadas em conta são: comprimentos das linhas, ângulos das linhas e posições iniciais relativas ao eixo X.

Com este procedimento de associação de linhas, obtém-se dois lados da forma retangular. Assim, pode-se calcular os demais lados e trabalhar com base em dois lados somente. A Figura 4.10 mostra o resultado do algoritmo completo sobre a imagem original.

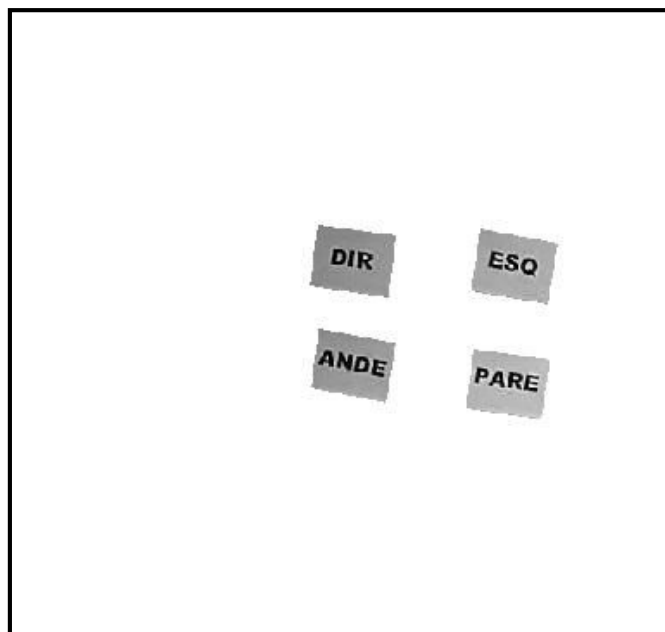


Figura 4.10: Retângulos obtidos

#### 4.1.2 Isolando Caracteres

Para cada uma das imagens que contém um dos comandos deste estudo de caso, faz-se a localização dos retângulos e então a detecção e a segmentação dos caracteres. Para que este processo seja entendido, dois retângulos do banco de imagens tiveram seu processo acompanhado e são apresentados aqui. Foram escolhidos dois comandos de imagens distintas contendo diferentes orientação e escala. Além disso, optou-se por apresentar comandos diferentes, no caso: **dir** e **ande**.

A técnica de detecção e segmentação dos caracteres é uma abordagem intuitiva baseada no procedimento de crescimento de regiões (*region growing*) (GONZALEZ; WOODS, 2000; JAIN, 1989; SONKA; HLAVAC; BOYLE, 1999), utilizado na área de Processamento de Imagens Digitais.

A técnica funciona da seguinte forma: depois de definido quais retas pertencem a cada retângulo, projeta-se uma linha que cruza a imagem seguindo a mesma orientação das retas que definem o retângulo (linha azul). Isto pode ser visto nas Figuras 4.11-4.12. Isso é feito pois nesta linha, próximo ao centro da imagem, sempre deverá existir pelo menos uma pequena fração de cada caractere, o que já é suficiente para o funcionamento adequado da técnica. O algoritmo percorre os pontos desta linha em busca de pontos que estejam próximos a preto e então, utiliza o primeiro ponto encontrado como semente (marcado em vermelho nas imagens) para o processo de crescimento de regiões. Após isto, a vizinhança (conectividade-8) deste ponto é verificada com o intuito de observar se algum dos vizinhos também atende ao critério de similaridade.

Esse processo é recursivo e a cada vez que um pixel atende ao critério de similaridade esta posição é marcada para que o algoritmo não a percorra novamente. Esse procedimento é realizado até que não existam mais pixels que atendam ao critério de similaridade. Neste momento, um novo ponto é buscado e o processo se repete.

Este processo é demonstrado utilizando-se as Figura 4.11-4.12. Note que, nas imagens mostradas, o processo de detecção de retângulos já está realizado, sendo que as próximas etapas do processamento serão realizadas sobre os caracteres já isolados.

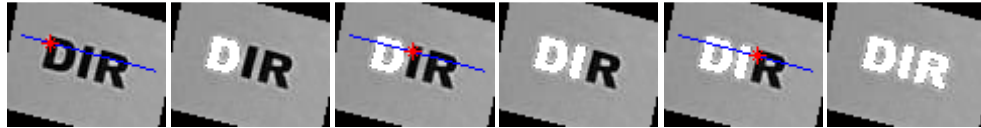


Figura 4.11: Isolando os caracteres do comando DIR



Figura 4.12: Isolando os caracteres do comando ANDE

Cabe salientar que não existe limite pré-estabelecido em relação ao número de caracteres existentes nos retângulos, sendo possível a detecção de qualquer número de caracteres.

Conforme descrito anteriormente, estas imagens passam pelo processo de localização dos retângulos e após ao isolamento dos caracteres. Quando o método foi proposto por Molz (2001) apenas a filtragem gaussiana era efetuada como pré-processamento, neste trabalho, entretanto, observou-se que a utilização do filtro da mediana em primeiro lugar fazia com que apenas os retângulos que possuíam os comandos fossem localizados reduzindo assim o tempo que haveria para a verificação de quais retângulos eram válidos e quais não eram.

#### 4.1.3 Análise de Variabilidade

A fim de possibilitar um maior entendimento sobre as características das imagens deste estudo de caso, fez-se um estudo de variabilidade empregando-se os Descritores de Fourier e os Momentos Invariantes. Objetiva-se que esta variabilidade possa ser comprovada através de análise visual para o descritor com similaridade baseada em contorno. Isto ocorre pois existe uma pequena quantidade de classes e também pela excelente qualidade da segmentação. O método com similaridade por região, de modo geral, apresenta resultados bastante próximos para o sistema visual humano o que exige o emprego de métodos computacionais para a análise visto que as classes tendem a ocorrer muito próximas.

A Tabela 4.1 apresenta os caracteres pertencentes às imagens da Figura 4.1 onde verifica-se a variabilidade das amostras de uma maneira bastante simplificada. Através de comparações na referida tabela, percebe-se que existe uma variabilidade elevada na quantidade de amostras das classes onde algumas possuem poucas amostras enquanto outras possuem um número bastante elevado. Caso o treinamento dos classificadores não seja realizado adequadamente pode-se ter uma tendência de resposta maior as classes que possuem maior quantidade de amostras. A fim de reduzir esta possibilidade, gerou-se 5 conjuntos de treinamento (estimação) aleatórios para cada configuração de classificador através da função *preparaAmostras*.

A média das respostas obtidas destes experimentos, para cada um dos classificadores, será analisada no próximo capítulo.

Tabela 4.1: Caracteres individuais dos comandos: PARE, ANDE, DIR e ESQ

Total de amostras	Amostras
18	<b>A A A A A A A A A A A A A A A A A A</b>
19	<b>D D D D D D D D D D D D D D D D D D</b>
28	<b>E E E E E E E E E E E E E E E E E E</b> <b>E E E E E E E E E E E E E E E E E E</b>
10	<b>I I I I I I I I I I</b>
9	<b>N N N N N N N N N N</b>
9	<b>P P P P P P P P P P</b>
10	<b>Q Q Q Q Q Q Q Q Q Q</b>
19	<b>R R R R R R R R R R R R R R R R R R</b>
10	<b>S S S S S S S S S S</b>

Nas Figuras 4.13-17 são apresentados os Momentos Invariantes para cada uma das classes.

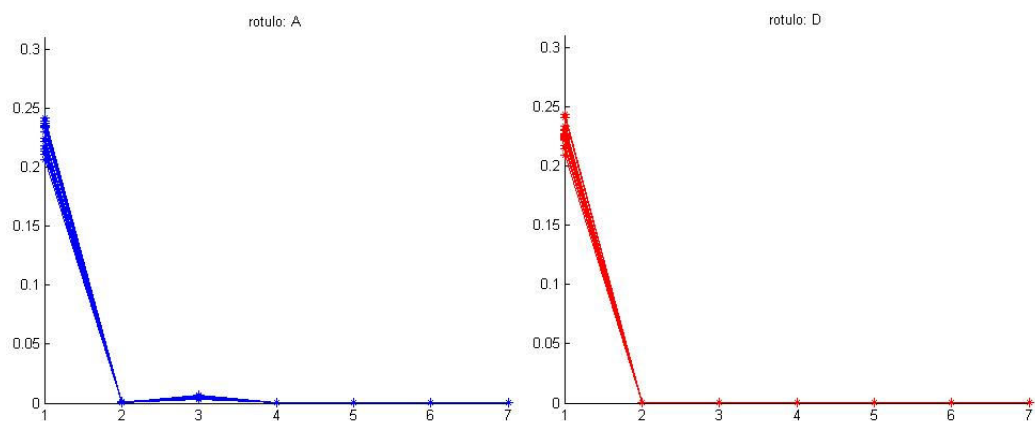


Figura 4.13: Variabilidade das amostras para as classes “A” e “D”

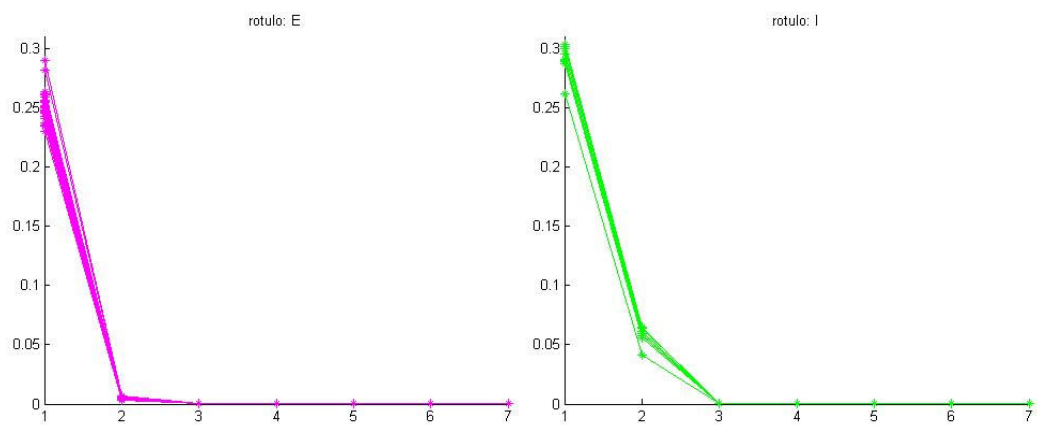


Figura 4.14: Variabilidade das amostras para as classes “E” e “I”

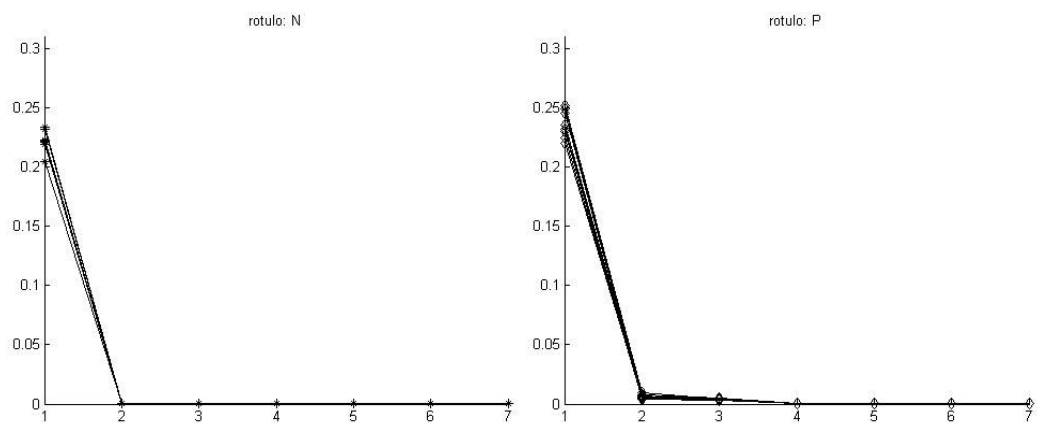


Figura 4.15: Variabilidade das amostras para as classes “N” e “P”

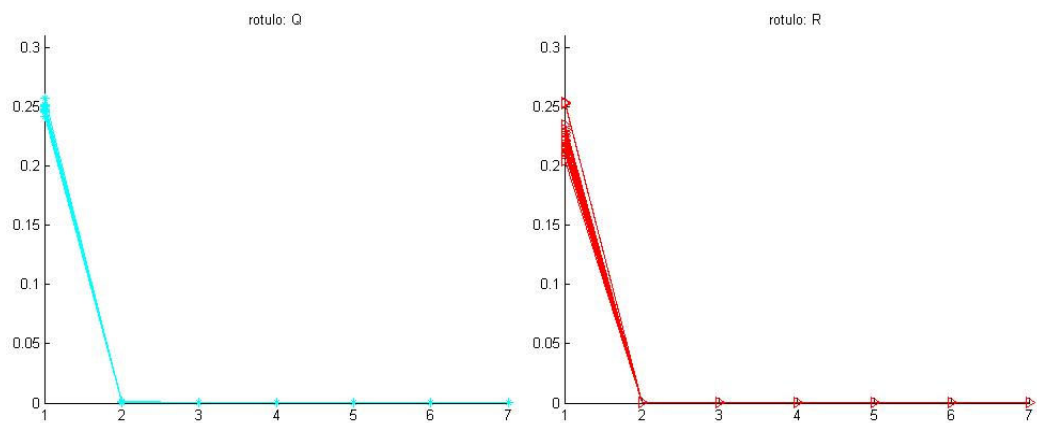


Figura 4.16: Variabilidade das amostras para as classes “Q” e “R”



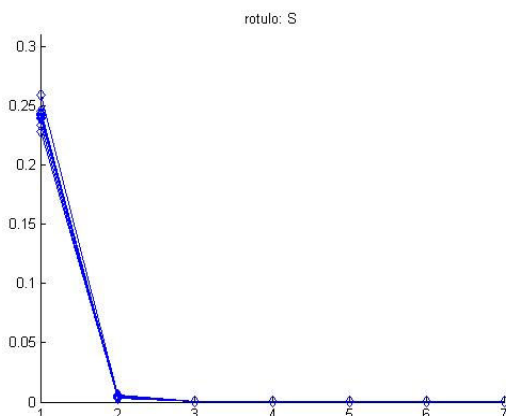


Figura 4.17: Variabilidade das amostras para a classe “S”

Com as Figuras 4.13-4.17 podemos verificar a variabilidade existente entre as amostras de cada classe. As amostras possuem uma pequena variação intra-classe e uma também pequena variação inter-classe. A segunda característica não é desejável pois aumenta a possibilidade de erros de classificação nas amostras que fazem parte das fronteiras entre as classes. Com isto, os classificadores necessitam de maior robustez e poder de processamento a fim de minorar este problema. A Figura 4.24 apresenta todas as classes deste estudo de caso e observa-se que as classes realmente ocorrem num pequena área. De modo geral, apenas a classe “I” pode ser distinta através da análise visual.

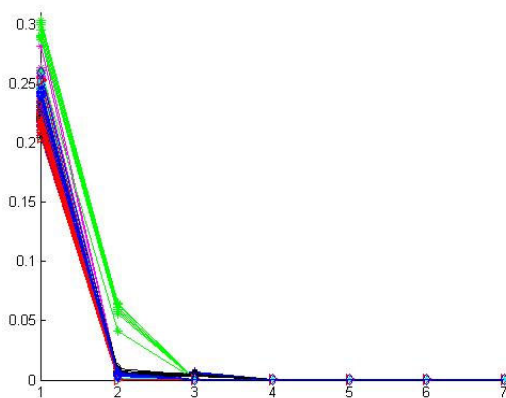


Figura 4.18: Variabilidade das amostras

Nas Figuras 4.19-23 são apresentados os 5 primeiros Descritores de Fourier para cada uma das classes. As letras junto aos gráficos visam mostrar a variabilidade das amostras naquele ponto específico para aquela classe.

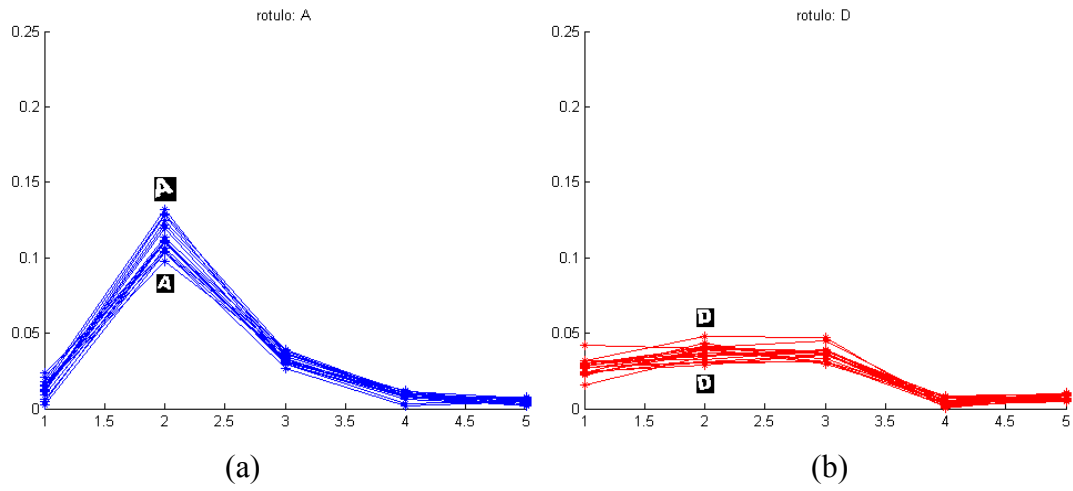


Figura 4.19: Variabilidade das amostras para as classes “A” e “D”

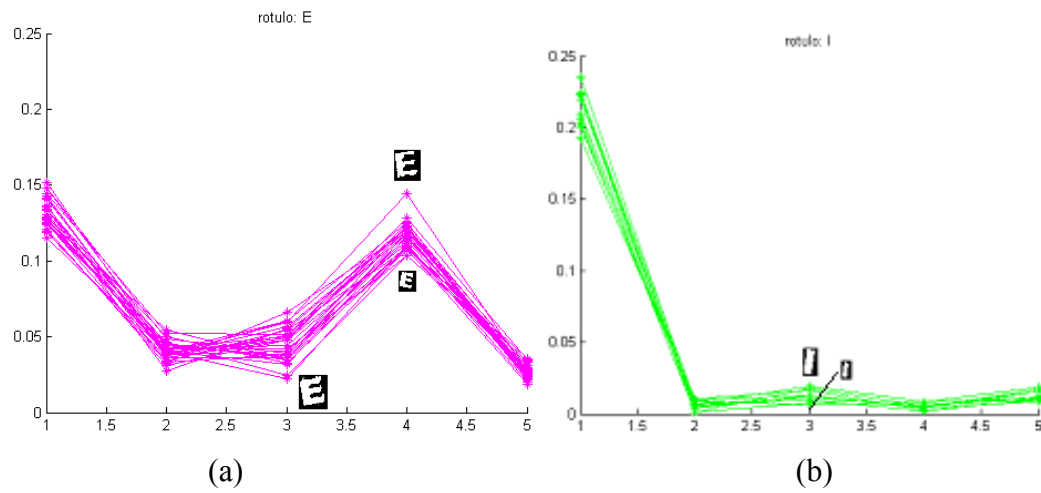


Figura 4.20: Variabilidade das amostras para as classes “E” e “I”

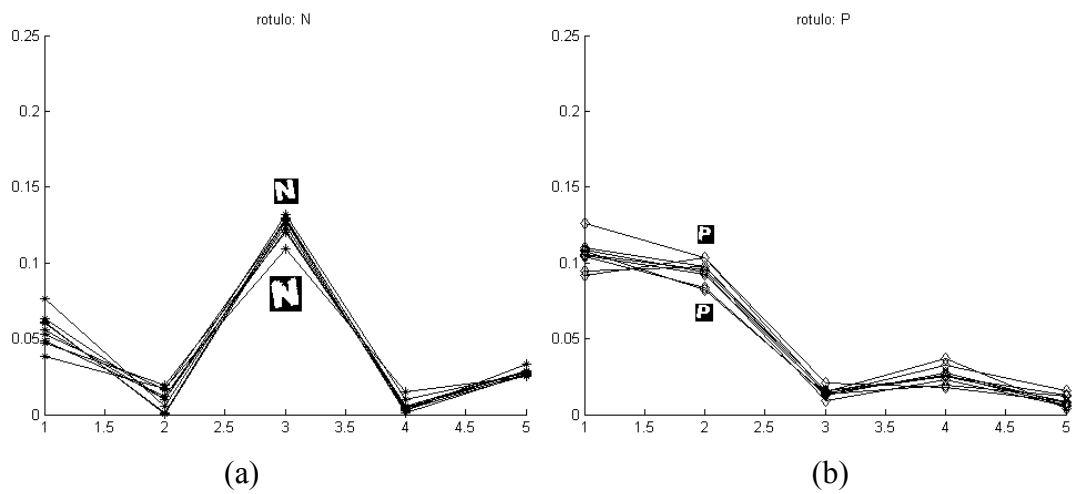


Figura 4.21: Variabilidade das amostras para as classes “N” e “P”

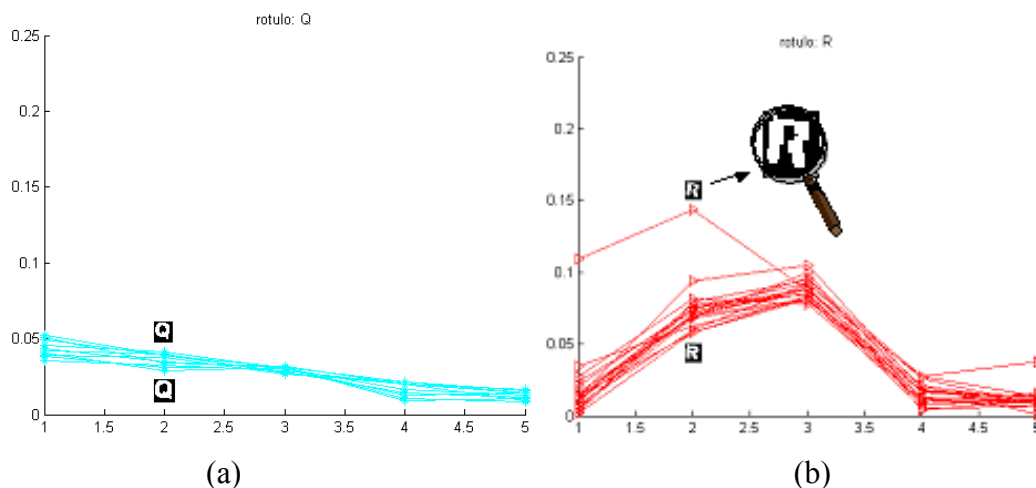


Figura 4.22: Variabilidade das amostras para as classes “Q” e “R”

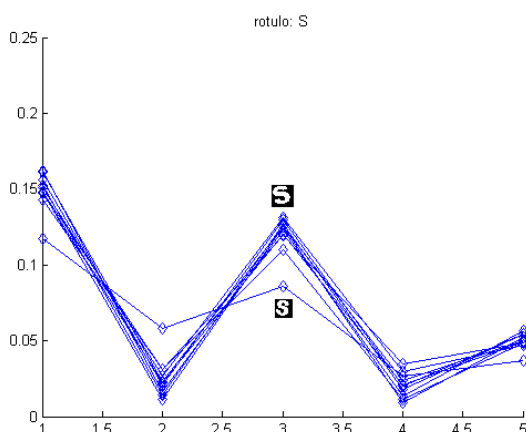


Figura 4.23: Variabilidade das amostras para a classe “S”

Com as Figuras 4.19-4.23 podemos verificar a variabilidade existente entre as amostras de cada classe. De modo geral, as amostras possuem uma pequena variação intra-classe e uma grande variação inter-classe, evidenciando um excelente descritor de forma. Isto propicia, por exemplo, uma arquitetura bastante simplificada para os classificadores neurais pois o descritor já foi capaz de separar cada uma das classes existentes. Na Figura 4.22 (b), entretanto, a amostra realçada apresenta um problema na segmentação. Isto não resultou em problemas para o reconhecimento pois existe uma pequena quantidade de classes e além disso, cada uma das classes é bastante diferente se analisarmos o seu contorno. Apesar das imagens serem aparentemente simples, percebe-se a necessidade de realizar uma segmentação eficiente que resolva ou reduza problemas de segmentação inadequados. Problemas como este tendem a agravarem-se com o incremento da quantidade de classes devendo então ser estudados com maior ênfase.

A Figura 4.24 apresenta todas as classes deste estudo de caso e através de sua análise fica evidente que os Descritores de Fourier são uma excelente alternativa para aplicações onde existe uma preservação eficiente do contorno dos objetos.

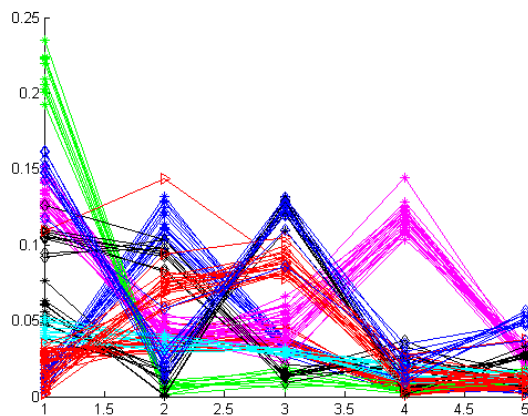


Figura 4.24: Variabilidade das amostras

## 4.2 Estudo de Caso – Placas de Automóveis

O banco de imagens das Placas de Automóveis é composto por imagens com 640x480 pixels adquiridas com câmera digital nas ruas das cidades de Recife e Natal. Utilizaram-se 100 das cerca de 120 imagens obtidas. A não utilização da totalidade das imagens deve-se a características das mesmas, onde, em certos casos, até mesmo um analista humano teria dificuldades em efetuar o reconhecimento. A grande maioria das imagens descartadas encontrava-se apagada ou distante, prejudicando a etapa de segmentação. Por fim, por simplicidade de cálculos optou-se por utilizar exatamente 100 imagens, pois assim, existem exatamente 300 letras e 400 dígitos (cada imagem possui 3 letras e 4 dígitos).

Após este esclarecimento, inicia-se a apresentação de algumas imagens deste estudo de caso e posteriormente as explicações relativas a este experimento.

A fim de esclarecimento, algumas das imagens não utilizadas estão apresentadas nas Figura 4.25 e 4.26. As imagens originalmente coloridas foram convertidas para níveis de cinza para o processamento. Nas Figura 4.27-33 algumas das imagens empregadas no experimento podem ser verificadas.



(a)

(b)

Figura 4.25: Placas não utilizadas por impossibilidade de reconhecimento



Figura 4.26: Placas não utilizadas por problemas na iluminação

A Figura 4.25 apresenta imagens que não foram utilizadas devido a pequena dimensão e baixa resolução que impossibilita uma segmentação adequada dos caracteres da placa. As imagens da Figura 4.26 não puderam ser utilizadas pois em (a) a inclinação elevada faz com que os dígitos fiquem muito próximos acarretando um problema de segmentação. Já a Figura 4.26 (b) não pode ser utilizada pela degradação dos caracteres agregada à iluminação não uniforme, ou seja, reflexos e sombras.



Figura 4.27: Placas comerciais (vermelhas)

Em contrapartida, as imagens da Figura 4.27 foram segmentadas de modo a obter os sete caracteres existentes. Observa-se que as imagens em tons de cinza são invertidas, possuindo fundo escuro e letras claras. Assim, durante a etapa que visa isolar cada um dos caracteres houve a verificação da cor predominante da placa, a fim de identificar qual sua cor de fundo e efetuar adequadamente a segmentação. Este processo será detalhado no decorrer da seção 4.2.2.



(a)

(b)



(c)

(d)

Figura 4.28: Placas em escala reduzida

As imagens da Figura 4.28 apesar de apresentarem-se em escala reduzida, puderam ser segmentadas adequadamente pois existe variação uniforme entre a tonalidade dos caracteres e o fundo da placa.



(a)

(b)

Figura 4.29: Placas inclinadas e com iluminação não-uniforme

As inclinações e a iluminação não-uniforme presentes nas imagens da Figura 4.29 não acarretaram problemas de segmentação pois o fundo da placa é mais claro que os caracteres.



Figura 4.30: Placas em escala mediana

As imagens que possuem alta nitidez, como as da Figura 4.30, possuem caracteres escuros e fundo claro bastante homogêneo. Imagens semelhantes a estas poderiam ser de uma aplicação que fizesse o controle de acessos a estacionamentos.



Figura 4.31: Placas enfumaçadas

Imagens enfumaçadas, Figura 4.31, normalmente encontradas em veículos de empresas de transporte de cargas são difíceis de serem segmentadas pelo baixo contraste existente. Neste estudo de caso, optou-se por efetuar a escolha de um limiar para a binarização das placas através da análise de algumas de suas características. Esta etapa será apresentada e explicada mais abaixo.



Figura 4.32: Placas com caracteres desgastados

Desgastes nos caracteres, como os observados na Figura 4.32 são comuns e nem sempre obtém-se uma segmentação eficiente. Pode-se considerar, inclusive, que isto só é possível na ausência de iluminação não-uniforme. Dificilmente, a ocorrência dos dois problemas em uma mesma imagem possibilitará a escolha de um limiar adequado para a segmentação.



Figura 4.33: Placas com sombras e escala reduzida

As sombras nas placas de certos tipos de automóveis são comuns e nem sempre podem ser tratadas de maneira eficiente. Na Figura 4.33 pode-se constatar duas placas que contêm sombras mas que tiveram seus caracteres isolados.

A seguir, detalham-se as etapas de Localização dos Retângulos e Isolamento dos Caracteres.

#### 4.2.1 Localização dos Retângulos

Devido à natureza do banco de imagens criado, o método de localização de retângulos apresentado no Estudo de Caso para os Comandos não pôde ser aplicado eficientemente neste Estudo de Caso. Isto se deve, dentre outras coisas, à impossibilidade de utilizar adequadamente a etapa de Aplicação de Restrições. Como não se possui um tamanho padrão para as placas, não é possível estabelecer uma



dimensão mínima e máxima para cada retângulo. Além disso, os ângulos de rotação são bastante variados resultando em *gaps* de naturezas diversas.

Por fim, a ênfase deste trabalho está no emprego de Descritores de Forma e também na Classificação Neural e Estatística e, desta forma, optou-se por realizar manualmente a localização dos retângulos.

O desenvolvimento de técnicas eficientes de localização de retângulos com diferentes escalas, graus de rotação, distorções e também que apresentem iluminação não-uniforme em ambientes naturais é um tema interessante para trabalhos futuros.

#### 4.2.2 Isolando Caracteres

Para que se obtenha os caracteres isolados, como desejado neste trabalho, são necessárias três etapas principais apresentadas na Figura 4.34. A seguir, estas etapas serão descritas com ênfase maior à primeira delas, a Binarização das Placas.



Figura 4.34: Etapas para o Isolamento dos Caracteres

Resumidamente, a etapa de Binarização das Placas é responsável por separar os objetos de interesse, no caso as letras e os dígitos, do restante da imagem, ou seja, da placa propriamente dita. A etapa de Projeção da Linha é necessária para localizar os caracteres das placas. Visa encontrar a linha próxima ao centro de cada retângulo que venha a conter pelo menos uma pequena fração de cada um dos sete caracteres. Por fim, a última etapa, o Isolamento dos Caracteres, é realizado através do algoritmo de crescimento de regiões, já descrito, aplicado em conjunto as restrições de tamanho que visam identificar se a região encontrada pertence a um dos sete caracteres da placa.

A segmentação de imagens possui uma abordagem chamada limiarização que se baseia em características do histograma da imagem. A limiarização visa agrupar regiões a partir de seu nível de cinza. Assim, gera-se um histograma de uma imagem e a partir deste, escolhe-se um ou mais pontos que serão utilizados como pontos limite para a geração de uma nova imagem. Este processo é melhor entendido através das Figuras 4.35 e 4.36. Na Figura 4.35 tem-se o histograma gerado para a imagem da Lua bem como dois limiares, L1 e L2.

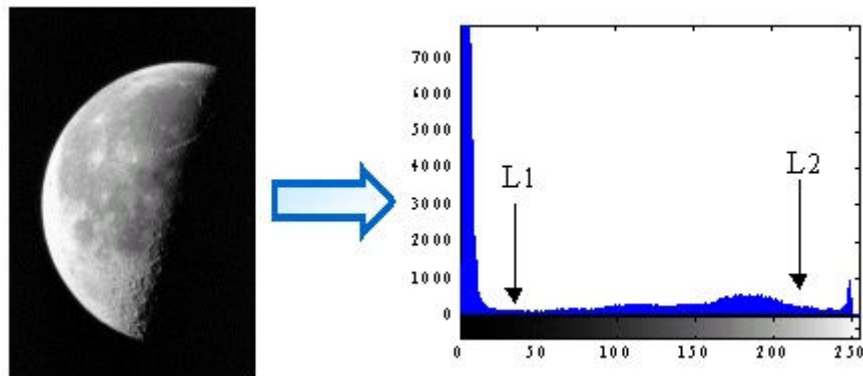


Figura 4.35: Histograma da Lua com limiares L1 e L2

A Figura 4.36 apresenta as imagens resultantes do processo de limiarização para a imagem da Lua. Na Figura 4.36 (a) utilizou-se apenas o limiar L1. Assim, todos os níveis de cinza com valor inferior a L1 foram convertidos para preto e todos os valores superiores a L1 foram convertidos para branco. Imagens como esta, que possuem apenas branco e preto são denominadas imagens binárias. Desta forma, dá-se o nome de binarização ao processo de limiarização que utiliza um único limiar. Na Figura 4.36 (b) tanto o limiar L1 quanto o limiar L2 foram utilizados. Todos os níveis de cinza com valores inferiores a L1 foram convertidos para preto e todos os valores superiores a L2 foram convertidos para branco. Valores que pertencem ao intervalo entre L1 e L2 foram convertidos para cinza.

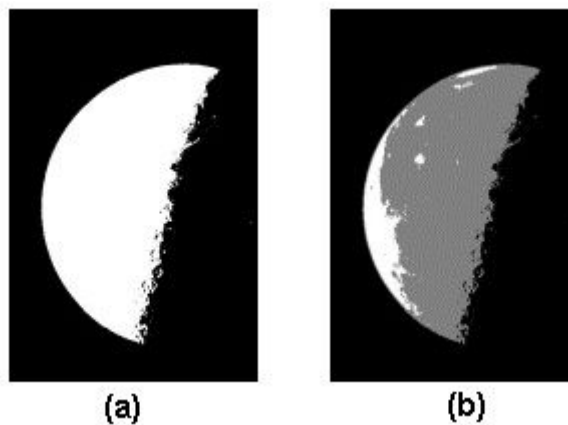


Figura 4.36: Em (a) limiarização utilizando limiar L1 e em (b) com limiares L1 e L2

No presente estudo de caso, binariza-se as imagens das placas de maneira que os caracteres sejam pretos e o fundo da placa seja branco. Isto é realizado com o intuito de simplificar o isolamento dos caracteres, realizado através do algoritmo de crescimento de regiões. A Figura 4.37 traz o resultado da binarização para uma das imagens das Figuras 4.27-4.33.

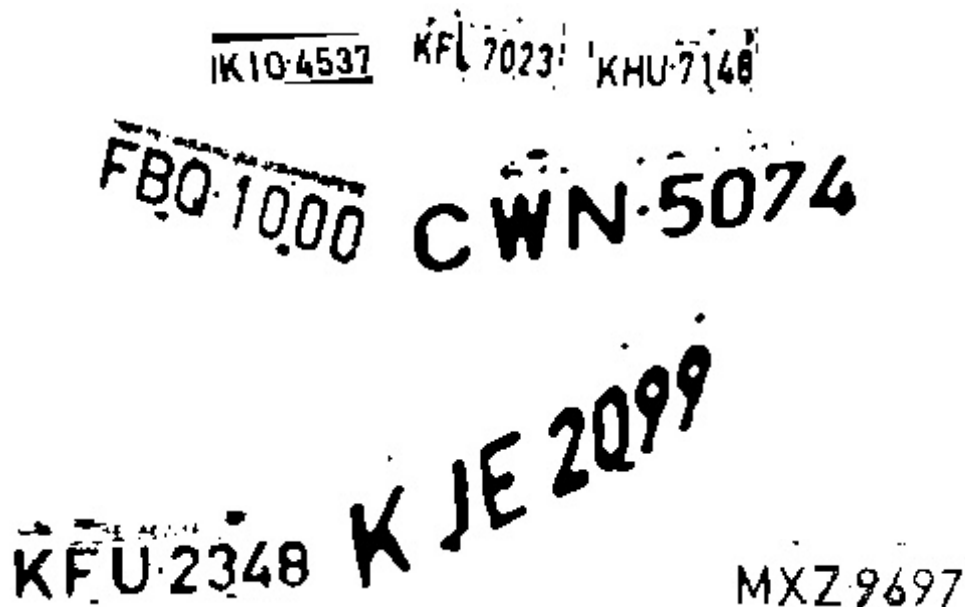


Figura 4.37: Amostras de placas binarizadas

Pode-se observar na Figura 4.37 que a etapa de binarização não resulta apenas nos sete caracteres desejados. Entretanto, de modo geral, obteve-se um resultado satisfatório para a maioria das imagens apresentadas. Para as imagens menores, primeira linha da Figura 4.37, constata-se a união de várias regiões aos caracteres o que obviamente não é desejado, apesar de esperado.

No presente trabalho, as imagens foram utilizadas como foram geradas, sem nenhum pós-processamento a fim de verificar a robustez dos métodos diante das imperfeições e distorções. Fica evidente que o desempenho dos descritores será prejudicado devido aos resultados da segmentação.

A binarização aqui empregada é composta de várias sub-tarefas que visam reduzir a quantidade de informação irrelevante. Durante os experimentos ficou evidente que não era possível binarizar as imagens através de um limiar único necessitando-se adicionar outros tratamentos com o intuito de obter melhores resultados. Operadores morfológicos são exemplos de uma das sub-tarefas empregadas. A seguir as sub-tarefas mais importantes serão descritas.

A primeira rotina tem por objetivo identificar se a placa é vermelha ou cinza o que é feito através da análise do histograma da imagem. Mediante testes verificou-se que as imagens que continham uma maior contagem para a primeira metade dos 256 níveis de cinza eram vermelhas e em caso contrário eram cinza.

Outra importante rotina, a aplicação do filtro da mediana, está relacionada ao tamanho das imagens, podendo ser: pequena, média ou grande. As imagens de tamanho médio são filtradas através de um filtro 3x3 e as grandes através de um filtro 5x5. Imagens pequenas não sofrem a ação da filtragem.

A partir deste instante, é possível realizar a binarização das imagens o que em alguns casos é realizado através da função *im2bw* pertencente ao *toolbox* de Processamento de Imagens Digitais do Matlab ou através da rotina de binarização ótima *opthr* disponível para download na página da MathWorks, fabricante do Matlab.

Existem algumas condições que são responsáveis pela seleção de qual destas funções será utilizada. A primeira condição está relacionada à área de concentração das maiores contagens do histograma. Se a imagem possuir uma distribuição próxima à gaussiana, com um pico no centro do histograma, é necessário utilizar limiares pré-definidos. Assim, se as imagens forem cinza utiliza-se um limiar de 0,80 e no caso de serem vermelhas os limiares podem ser 0,45 ou 0,22, dependendo de quão escura é a imagem. Caso a imagem não possua um único pico central pode-se utilizar o limiar adaptativo através da função *opthr*. Assim, realiza-se a primeira parte da binarização da imagem.

O processamento morfológico é realizado apenas nas imagens que possuam placa cinza e que sejam consideradas grandes. Nestas, verificou-se a viabilidade de efetuar o melhoramento da imagem acrescentando um pequeno esforço. Este melhoramento é obtido com o emprego da função *bwmorph* pertencente ao *toolbox* Matlab de Processamento de Imagens Digitais. A função recebe como parâmetros uma imagem binária, composta de zeros e uns e um operador. Os operadores empregados foram: *majority* e *spur*. O primeiro destes operadores faz com que um pixel seja 1 se cinco ou mais pixels de sua vizinhança 3x3 também sejam 1. *Spur* remove as extremidades (*end-points*) das linhas sem remover os pequenos objetos completamente.

As imagens que no início do processo foram classificadas como vermelhas são invertidas após todos estes processamentos a fim de simplificar o isolamento dos caracteres. Por fim, remove-se as laterais das imagens, buscando-se assim evitar que algum ponto pertencente a placa em si unifique dois ou mais caracteres o que resultaria em uma segmentação ineficiente.

Após a imagem ter sido binarizada necessita-se encontrar uma linha que cruze por todos os caracteres para que então se possa isolá-los. A Projeção da Linha segue os mesmos princípios empregados no primeiro estudo de caso e desta forma não necessita ser descrita.

O Isolamento dos Caracteres é feito através do algoritmo de crescimento de regiões já descrito. Neste experimento, existe a necessidade de buscar um número pré-determinado de caracteres, no caso, sete. Além disso, efetua-se uma restrição em relação ao tamanho da região encontrada. Esta restrição é necessária pois uma das regiões encontradas pode ser o separador das letras e dos dígitos ou até mesmo um ruído oriundo da segmentação. Assim, para as imagens que forem classificadas como grande, são desconsideradas as regiões encontradas que possuem menos de setenta pixels. Para as demais imagens, observa-se a altura da imagem e a quantidade de pixels encontradas na região. Caso a altura da imagem seja inferior a 30 pixels e a região encontrada possua menos de 10 pixels, a região é desconsiderada. O último caso é aplicado as imagens com altura inferior a 40 pixels e quantidade de pixels da região inferior a 30 pixels.

Cabe realçar que devido aos objetivos deste trabalho, ou seja, a análise do comportamento dos descritores de forma e dos classificadores, optou-se por isolar os caracteres de cada uma das 100 imagens e armazená-los a fim de que a etapa de Extração de Feições de Forma para os vários métodos diferentes fosse realizada diretamente sobre os caracteres já segmentados reduzindo o tempo necessário para os experimentos.

### 4.2.3 Banco de Imagens

Objetivando-se apresentar um pouco mais sobre o banco de imagens de placas de automóveis, gerou-se a Tabela 4.2 e as Figuras 4.38 e 4.39 onde é possível verificar o total de amostras para cada uma das classes. Salienta-se que o banco de imagens foi gerado a partir de fotos do trânsito e de estacionamentos das cidades de Natal e Recife.

A Tabela 4.2 apresenta os caracteres pertencentes às imagens das placas e pode-se verificar a variabilidade das amostras. Através de comparações na referida tabela, percebe-se que existem caracteres que possuem um elevado número de amostras enquanto outros, em alguns casos, possuem menos de 10% de amostras em relação a classe mais representativa. Os dados da Tabela 4.2 também podem ser visualizados nos gráficos da Figura 4.38 que consiste das letras e da Figura 4.39 que consiste dos dígitos.

Tabela 4.2: Total de Amostras para cada classe

Caracteres	Total de Amostras	Caracteres	Total de Amostras
A	5	B	8
C	5	D	3
E	3	F	14
G	10	H	16
I	10	J	12
K	60	L	13
M	42	N	4
O	6	P	5
Q	5	R	8
S	4	T	2
U	4	V	5
W	5	X	29
Y	15	Z	7
0	46	1	36
2	33	3	34
4	41	5	48
6	31	7	45
8	46	9	40

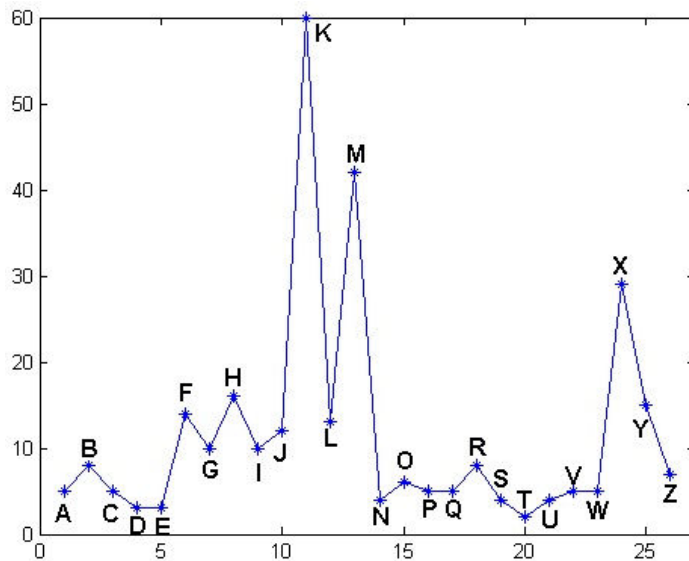


Figura 4.38: Variação do total de amostras para as letras

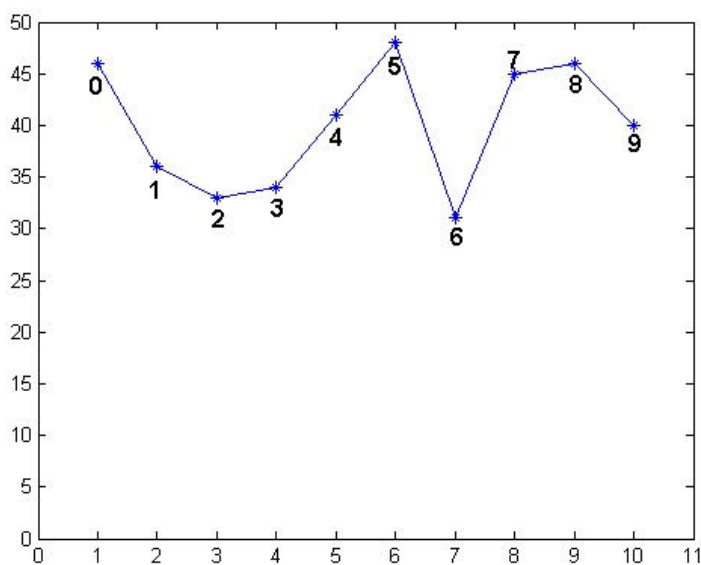


Figura 4.39: Variação do total de amostras para os dígitos

O próximo capítulo traz os detalhes sobre os experimentos realizados e suas respectivas análises. Os resultados são analisados com base na taxa de erro médio e no desvio padrão.

## 5 ANÁLISE DOS RESULTADOS

Este capítulo aborda os experimentos realizados bem como a apresentação e a análise dos resultados de cada um dos Estudos de Caso comprovando a importância do *toolbox* desenvolvido.

A métrica utilizada para analisar o desempenho da combinação entre os descritores de forma e os classificadores foi a taxa de erro médio em relação aos padrões de três execuções de cada combinação e seu respectivo desvio padrão. Tanto a taxa de erro médio quanto o desvio padrão estão expressos em percentagem.

Foram utilizadas 75% das amostras para a etapa de treinamento e os 25% restantes para a etapa de teste.

Nas tabelas deste capítulo, quando não se obteve resultados confiáveis ou quando não houve convergência optou-se por empregar a abreviatura NC.

### 5.1 Comandos

Este primeiro experimento foi realizado com o intuito de verificar quais métodos são mais indicados para o reconhecimento de caracteres por parte de um robô móvel. Além disso, também visa verificar quais os descritores de forma e classificadores mais promissores para o segundo experimento, mais complexo. Em função disso apresenta um número elevado de dados a serem analisados, dados estes, que estão dispostos em 6 tabelas distintas, sendo uma para cada descritor de forma.

Na primeira coluna de cada uma destas tabelas estão apresentados os classificadores utilizados e quando os mesmos forem neurais, entre parênteses apresenta-se o número de neurônios de sua camada oculta. Também é indicado se os mesmos são estatísticos ou neurais. Baseado nos diversos experimentos realizados optou-se pelo emprego de 5, 10 e 15 neurônios na camada oculta.

As demais colunas referem-se aos descritores de forma, conforme abreviaturas já definidas anteriormente, sendo que entre parênteses é apresentado o número de feições empregada.

O número de épocas para treinamento dos classificadores neurais foi de 10.000 com exceção do classificador Levenberg-Marquardt para o qual foram empregadas 500 épocas. Segundo a literatura pesquisada, este classificador não apresenta melhoras a partir da centésima época. Isto foi comprovado durante os experimentos. Além disso, o tempo de duração de cada época é bastante superior ao obtido com os demais classificadores neurais utilizados.

Os primeiros comentários são referentes aos resultados obtidos com os Momentos Invariantes Afins (Tabela 5.1) com 3 e 4 descritores. O emprego de 3 descritores deve-se ao fato de que, segundo os autores do referido método, para imagens com baixa complexidade este número é suficiente e o emprego de 4 descritores deve-se ao fato de este ser o número máximo de momentos definidos para este descritor.

Tabela 5.1: Classificação para os Momentos Invariantes Afins

Classificador		MIA (3)	MIA (4)
		Erro $\pm$ Desvio Padrão	Erro $\pm$ Desvio Padrão
Estatísticos	DM	32,43 $\pm$ 2,70	30,63 $\pm$ 8,26
	MVG	9,91 $\pm$ 3,12	19,82 $\pm$ 4,13
Neurais	GDX (5)	NC	NC
	LM (5)	28,83 $\pm$ 1,56	30,63 $\pm$ 3,12
	RP (5)	12,61 $\pm$ 1,56	17,12 $\pm$ 3,12
	SCG (5)	24,32 $\pm$ 8,11	14,41 $\pm$ 3,12
	GDX (10)	15,32 $\pm$ 4,13	14,41 $\pm$ 6,24
	LM (10)	19,82 $\pm$ 7,80	16,22 $\pm$ 4,68
	RP (10)	14,41 $\pm$ 4,13	6,31 $\pm$ 1,56
	SCG (10)	16,22 $\pm$ 5,41	13,51 $\pm$ 2,70
	GDX (15)	9,01 $\pm$ 4,13	9,91 $\pm$ 6,24
	LM (15)	18,02 $\pm$ 4,13	16,22 $\pm$ 2,70
	RP (15)	9,91 $\pm$ 5,63	10,81 $\pm$ 5,41
	SCG (15)	16,22 $\pm$ 7,15	9,01 $\pm$ 1,56

Analisando os resultados obtidos para os MIA podemos observar que o melhor resultado obtido foi obtido com o emprego de 4 momentos e utilização do classificador neural RP com 10 neurônios na camada oculta. Além disso, também podemos observar que de modo geral os resultados podem ser considerados excelentes com exceção dos resultados obtidos com o LM com 5 neurônios na camada oculta e com a DM. Os demais classificadores obtêm em média 85% de reconhecimento. É possível verificar também em muitos casos um desvio padrão de aproximadamente 5% comprovando-se a necessidade de realizar mais de uma execução com conjuntos de treinamento e teste diferenciados. Para as RNAs ainda existe a inicialização aleatória dos pesos que pode alterar consideravelmente os resultados obtidos.

Na Tabela 5.2 temos os Momentos Invariantes propostos por Hu (1962). Empregou-se 4 momentos para que fosse possível efetuar alguma comparação com os MIA e os 7 momentos por serem o número máximo de momentos possíveis para este método.

De modo geral, os MI obtiveram resultados superiores aos MIA. Novamente os piores resultados foram obtidos com o LM com 5 neurônios na camada oculta e com a DM. Utilizando como referência apenas o erro médio é possível verificar que os classificadores estatísticos foram os únicos a não superar o desempenho obtido com os MIA. Outra característica bastante visível é o desempenho equivalente com os dois descritores para o GDX com 10 e 15 neurônios na camada oculta. Os melhores resultados novamente foram obtidos com os classificadores RP e SCG.

Verificando-se os resultados destes classificadores podemos notar que muitos obtiveram taxa de erro médio inferior a 3%.



Tabela 5.2: Classificação para os Momentos Invariantes

Classificador		MI (4)	MI (7)
		Erro $\pm$ Desvio Padrão	Erro $\pm$ Desvio Padrão
Estatísticos	DM	46,85 $\pm$ 1,56	49,55 $\pm$ 5,63
	MVG	9,91 $\pm$ 4,13	30,63 $\pm$ 1,56
Neurais	GDX (5)	13,51 $\pm$ 9,74	15,32 $\pm$ 4,13
	LM (5)	21,62 $\pm$ 4,68	27,03 $\pm$ 2,70
	RP (5)	8,11 $\pm$ 5,41	1,80 $\pm$ 1,56
	SCG (5)	4,50 $\pm$ 4,13	3,60 $\pm$ 1,56
	GDX (10)	15,32 $\pm$ 8,26	13,51 $\pm$ 7,15
	LM (10)	5,41 $\pm$ 2,70	3,60 $\pm$ 1,56
	RP (10)	2,70 $\pm$ 2,70	5,41 $\pm$ 2,70
	SCG (10)	2,70 $\pm$ 2,70	8,11 $\pm$ 7,15
	GDX (15)	7,21 $\pm$ 3,12	9,91 $\pm$ 3,12
	LM (15)	6,31 $\pm$ 6,80	9,91 $\pm$ 6,80
	RP (15)	1,80 $\pm$ 1,56	1,80 $\pm$ 1,56
	SCG (15)	6,31 $\pm$ 4,13	5,41 $\pm$ 2,70

Dado a inexistência de imagens com transformações mais complexas não foi possível verificar se os MIA são mais efetivos do que os MI. Para verificar isto, seria necessário a existência de imagens com deslocamento no plano ou com alguma transformação semelhante ao cisalhamento.

Os Descritores de Fourier são apresentados na Tabela 5.3 e possuem similaridade baseada em contorno. Vale lembrar que os métodos apresentados nas Tabelas 5.1 e 5.2 possuem similaridade baseada em região.

O número de descritores empregados deve-se exclusivamente a fase de experimentação onde atingiu-se uma taxa de reconhecimento próxima à ideal em diversas combinações e, também, com um baixo número de descritores.

Com este descritor é possível verificar que os classificadores estatísticos que até então vinham obtendo resultados bastante inferiores aos demais obtiveram resultados também excelentes. Isto é um indicativo da utilidade deste descritor para o reconhecimento de padrões em imagens.

O classificador com os piores resultados neste experimento foi o LM, principalmente com 5 neurônios na camada oculta.

Tabela 5.3: Classificação para os Descritores de Fourier

Classificador		DF (3)	DF (5)
		Erro $\pm$ Desvio Padrão	Erro $\pm$ Desvio Padrão
Estatísticos	DM	3,60 $\pm$ 1,56	0,00 $\pm$ 0,00
	MVG	0,90 $\pm$ 1,56	5,41 $\pm$ 2,70
Neurais	GDX (5)	4,50 $\pm$ 5,63	5,41 $\pm$ 7,15
	LM (5)	25,23 $\pm$ 1,56	23,42 $\pm$ 4,13
	RP (5)	3,60 $\pm$ 1,56	1,80 $\pm$ 1,56
	SCG (5)	7,21 $\pm$ 5,63	4,50 $\pm$ 1,56
	GDX (10)	0,90 $\pm$ 1,56	1,80 $\pm$ 3,12
	LM (10)	7,21 $\pm$ 3,12	1,80 $\pm$ 1,56
	RP (10)	6,31 $\pm$ 1,56	0,00 $\pm$ 0,00
	SCG (10)	2,70 $\pm$ 2,70	0,00 $\pm$ 0,00
	GDX (15)	3,60 $\pm$ 1,56	2,70 $\pm$ 2,70
	LM (15)	8,11 $\pm$ 2,70	1,80 $\pm$ 1,56
	RP (15)	4,50 $\pm$ 1,56	2,70 $\pm$ 2,70
	SCG (15)	4,50 $\pm$ 3,12	0,90 $\pm$ 1,56

A seguir, na Tabela 5.4, um outro descritor com similaridade baseada em contorno é analisado. A não obtenção de resultados com o MVG é resultante de problemas com a estimação da matriz de covariâncias acarretada pela quantidade de amostras empregada. Este classificador precisa de um elevado número de amostras para que possa efetuar uma estimação correta desta matriz.

Tabela 5.4: Classificação para o *Curvature Scale Space*

Classificador		CSS (32)	CSS (64)
		Erro $\pm$ Desvio Padrão	Erro $\pm$ Desvio Padrão
Estatísticos	DM	32,43 $\pm$ 4,68	27,93 $\pm$ 11,25
	MVG	NC	NC
Neurais	GDX (5)	45,95 $\pm$ 18,72	57,66 $\pm$ 10,92
	LM (5)	41,44 $\pm$ 4,13	54,05 $\pm$ 4,68
	RP (5)	46,85 $\pm$ 1,56	67,57 $\pm$ 0,00
	SCG (5)	33,33 $\pm$ 10,92	52,25 $\pm$ 6,80
	GDX (10)	22,52 $\pm$ 8,26	31,53 $\pm$ 4,13
	LM (10)	21,62 $\pm$ 0,00	34,23 $\pm$ 4,13
	RP (10)	48,65 $\pm$ 9,74	57,66 $\pm$ 6,24
	SCG (10)	26,13 $\pm$ 8,26	42,34 $\pm$ 3,12
	GDX (15)	21,62 $\pm$ 0,00	37,84 $\pm$ 5,41
	LM (15)	21,62 $\pm$ 8,11	49,55 $\pm$ 1,56
	RP (15)	45,05 $\pm$ 4,13	59,46 $\pm$ 5,41
	SCG (15)	18,92 $\pm$ 4,68	29,73 $\pm$ 9,74

O CSS costuma ter seu contorno reamostrado com 128 ou 256 pontos. Como as imagens empregadas neste experimento possuem formas bastante simples não é possível reamostrá-las com tal quantidade de pontos. Isto ocorre pois a interpolação acaba reduzindo o número de pontos de curvatura existentes fazendo com que os mesmos sejam considerados ruído, o qual é desprezado. Assim, em imagens com contornos simples como os de caracteres este método não apresenta resultados tão satisfatórios quanto os que seriam obtidos com formas mais complexas.

Independentemente disso, pode-se observar que o contorno com 32 pontos apresenta erro próximos a 22% o que em alguns casos é bastante satisfatório. Por sua vez o contorno com 64 pontos apresenta resultados bastante inferiores, indicando que a escolha de 32 pontos foi acertada.

Este método não é empregado em classificação sendo utilizado para recuperação de informação visual em bancos de imagens, onde utiliza-se uma medida de distância para verificar quais imagens são semelhantes a uma dada imagem de consulta.

Assim, este trabalho apresentou resultados bastante promissores o que indica que tal descritor deve ser estudado para fins de classificação.

Após a apresentação dos resultados dos descritores com similaridade baseada em região, Tabelas 5.1 e 5.2, derivados dos momentos geométricos e dos resultados dos descritores com similaridade baseada em contorno, Tabelas 5.3 e 5.4, serão apresentados os Momentos de Zernike e Pseudo-Zernike, Tabelas 5.5 e 5.6, também descritores com similaridade baseada em região mas derivados dos momentos ortogonais.

Tabela 5.5: Classificação para os Momentos de Zernike

Classificador		MZ (10)	MZ (18)
		Erro $\pm$ Desvio Padrão	Erro $\pm$ Desvio Padrão
Estatísticos	DM	34,23 $\pm$ 8,26	32,43 $\pm$ 5,41
	MVG	NC	NC
Neurais	GDX (5)	NC	NC
	LM (5)	27,03 $\pm$ 2,70	34,23 $\pm$ 1,56
	RP (5)	52,25 $\pm$ 10,92	NC
	SCG (5)	17,12 $\pm$ 6,80	17,12 $\pm$ 9,49
	GDX (10)	NC	13,51 $\pm$ 9,36
	LM (10)	11,71 $\pm$ 15,84	19,82 $\pm$ 5,63
	RP (10)	1,80 $\pm$ 3,12	6,31 $\pm$ 3,12
	SCG (10)	0,90 $\pm$ 1,56	0,00 $\pm$ 0,00
	GDX (15)	3,60 $\pm$ 6,24	0,90 $\pm$ 1,56
	LM (15)	11,71 $\pm$ 10,23	8,11 $\pm$ 2,70
	RP (15)	1,80 $\pm$ 3,12	4,50 $\pm$ 3,12
	SCG (15)	0,00 $\pm$ 0,00	0,00 $\pm$ 0,00

Nos artigos de (KHOTANZAD, LU, 1990; KHOTANZAD; HONG, 1990) foi realizado um estudo sobre o número de ordem máximo a ser empregado com os MZ e MPZ sendo o mesmo definido em 12. Prokop e Reeves (1992) em seu artigo clássico recomendam 10. A partir destas recomendações e através de experimentação foram

definidas que as ordens máximas empregadas seriam 10 e 18. O número de momentos para os MZ para a ordem 10 pode ser visto na Tabela 2.4.

Na Tabela 5.5 verificamos várias ocorrências de não convergência e problemas na estimação da matriz de covariâncias. Além disso, também é possível observar vários resultados de classificação ideal. Ambas as configurações obtiveram resultados excelentes com 10 e 15 neurônios na camada oculta. Analisando-se os MZ(10) e MZ(18) em função do erro médio e do desvio padrão constatamos resultados equivalentes, entretanto, observando o desvio padrão é possível verificar que o mesmo é inferior em MZ(18), o que pode indicar uma probabilidade maior de obtenção de resultados superiores para os casos de poucas execuções do classificador.

Outra característica que pode ser observada é a baixa capacidade de convergência do GDX quando o número de neurônios na camada oculta é baixo. O LM também apresenta desempenho abaixo do desejado, sendo inferior aos demais classificadores neurais em todas as configurações. O RP e o SCG apresentam desempenho equivalente, sendo que com 15 neurônios na camada oculta o SCG atingiu resultados ideais em todas as suas execuções.

Para os resultados dos MPZ, Tabela 5.6, pode-se empregar a mesma análise realizada para os MZ. Como diferenças cita-se apenas que para os MZ o RP com 5 neurônios na camada oculta não obteve convergência com MZ(18) e que para os MPZ ele não obteve para 10 e 18. Para os MZ e MPZ o único classificador neural a apresentar resultados satisfatórios com 5 neurônios na camada oculta foi o SCG.

Tabela 5.6: Classificação para os Momentos Pseudo-Zernike

Classificador		MPZ (10)	MPZ (18)
		Erro $\pm$ Desvio Padrão	Erro $\pm$ Desvio Padrão
Estatísticos	DM	37,84 $\pm$ 5,41	27,03 $\pm$ 2,70
	MVG	NC	NC
Neurais	GDX (5)	NC	NC
	LM (5)	48,65 $\pm$ 16,44	26,13 $\pm$ 6,24
	RP (5)	NC	NC
	SCG (5)	14,41 $\pm$ 6,24	12,61 $\pm$ 4,13
	GDX (10)	18,92 $\pm$ 4,68	8,11 $\pm$ 0,00
	LM (10)	3,60 $\pm$ 1,56	18,02 $\pm$ 5,63
	RP (10)	3,60 $\pm$ 3,12	7,21 $\pm$ 6,80
	SCG (10)	0,00 $\pm$ 0,00	1,80 $\pm$ 3,12
	GDX (15)	7,21 $\pm$ 12,48	8,11 $\pm$ 0,00
	LM (15)	11,71 $\pm$ 3,12	23,42 $\pm$ 18,00
	RP (15)	1,80 $\pm$ 1,56	1,80 $\pm$ 1,56
	SCG (15)	0,00 $\pm$ 0,00	0,00 $\pm$ 0,00

Conforme apresentado no início desta seção, este estudo de caso tem por objetivo verificar quais métodos poderiam ser utilizados em um sistema de reconhecimento de comandos escritos, por exemplo, instalado em um robô móvel. Assim, foi feita a análise dos resultados obtidos na qual verificou-se que os classificadores com melhor desempenho e estabilidade são o RP e o SCG, superiores aos demais classificadores

para todos os descritores. Em relação aos descritores de forma, os DF, MZ e MPZ obtiveram 100% de acerto em algumas de suas combinações.

Desta maneira, estes métodos foram selecionados para o segundo estudo de caso. Além dos citados, optou-se por acrescentar também os MI pois os mesmos fazem parte dos momentos geométricos, os quais desejava-se que fossem avaliados frente ao segundo experimento.

## 5.2 Placas de automóveis

Este experimento tem por objetivo analisar o desempenho dos métodos selecionados em uma aplicação de reconhecimento de caracteres em placas de automóveis em condições variáveis, havendo automóveis em movimento, ou seja, no trânsito, e automóveis em estacionamentos. Existe uma enorme variabilidade nas condições de iluminação, ângulo de rotação e escala das imagens utilizadas.

A existência de tal variabilidade é proposital, pois deseja-se testar o desempenho dos descritores e classificadores frente a problemas diversos. É sabido, entretanto, que não existem aplicações reais que fazem uso de tais condições pois as aplicações são desenvolvidas para um propósito específico, seja ele para controle da entrada de um estacionamento ou seja ele para o controle de velocidade em rodovias ou cidades. Assim, espera-se que os métodos desenvolvidos neste *toolbox* e aqui estudados tenham um desempenho superior se aplicados a uma aplicação de propósito específico.

Para evitar erros de classificação entre letras como “O” e dígitos como o “0”, dentre outros, optou-se por criar um classificador para as letras e outro para os dígitos. Assim, a melhor combinação pode ser composta por diferentes classificadores e descritores de forma.

Em vista disso, as tabelas apresentam um formato diferente do apresentado na seção anterior.

Os descritores de forma são apresentados na primeira coluna contendo entre parênteses o número de coeficientes empregado. As demais colunas são referentes ao número de neurônios da camada oculta, podendo ser 5, 10, 15 ou 20. Cada tabela é referente a um único classificador. Em relação ao que foi apresentado no primeiro experimento acrescentou-se uma nova configuração com 20 neurônios na camada oculta. O objetivo desta modificação era verificar se os MI poderiam obter um desempenho mais próximo do obtido pelos demais descritores.

Durante os novos experimentos isto não se confirmou, entretanto, obteve-se melhores resultados para os outros descritores quando houve esse aumento no número de neurônios.

É provável que se houvesse novos aumentos no número de neurônios da camada oculta fosse obtido um erro de classificação inferior ao obtido. Isto não foi feito pois objetivava-se testar a robustez dos métodos frente ao ruído e a configurações com baixo número de neurônios.

Devido à quantidade de amostras e aos ruídos oriundos da segmentação foram utilizadas 25.000 épocas para o treinamento dos classificadores. Também devido aos ruídos e a quantidade de classes, acrescentou-se aos DF mais duas possibilidades, que são com 10 e 15 descritores. Para os MZ e MPZ isto não foi feito pois já se estava utilizando o número aconselhado pela literatura pesquisada.

Os resultados da Tabela 5.7 são referentes às letras das placas classificadas com o RP. O melhor resultado foi obtido com 20 neurônios na camada oculta, utilizando-se os Descritores de Fourier com 10 coeficientes. Analisando com mais atenção podemos verificar um desempenho para o melhor caso próximo a 69% de acerto.

Este resultado não é ideal, entretanto, é satisfatório dado a variabilidade das imagens empregadas. De modo geral os resultados são de cerca de 64% de reconhecimento.

Os MZ e MPZ não convergiram com a utilização de 5 neurônios na camada oculta, mas tiveram seu desempenho melhorado com o aumento da quantidade de neurônios, chegando nos melhores casos próximo aos 60% de reconhecimento. Os MI permaneceram com seu desempenho inferior aos demais métodos.

Tabela 5.7: Classificação das letras das placas com RP

Descritor	RP			
	05	10	15	20
DF (3)	51,76 ± 7,71	50,98 ± 5,92	50,98 ± 2,45	57,25 ± 2,96
DF (5)	45,88 ± 4,08	37,25 ± 6,48	36,08 ± 1,36	36,47 ± 3,53
DF (10)	40,00 ± 6,55	36,47 ± 4,08	31,76 ± 4,24	30,98 ± 3,40
DF (15)	45,49 ± 9,14	40,39 ± 5,43	38,04 ± 3,78	35,29 ± 3,11
MI (4)	60,00 ± 6,55	57,65 ± 6,55	60,39 ± 2,72	58,82 ± 7,35
MI (7)	59,22 ± 5,43	56,47 ± 3,11	61,57 ± 4,75	62,35 ± 5,13
MZ (10)	NC	52,94 ± 8,88	43,14 ± 3,59	40,78 ± 5,30
MZ (18)	NC	64,71 ± 0,00	44,71 ± 4,08	45,49 ± 7,47
MPZ (10)	NC	NC	46,67 ± 2,96	45,10 ± 6,48
MPZ (18)	NC	70,59 ± 5,13	62,35 ± 9,63	52,55 ± 3,78

Tabela 5.8: Classificação das letras das placas com SCG

Descritor	SCG			
	05	10	15	20
DF (3)	68,24 ± 10,05	54,51 ± 15,13	55,29 ± 5,13	60,39 ± 10,68
DF (5)	52,94 ± 10,46	45,10 ± 2,72	52,55 ± 20,66	46,67 ± 11,78
DF (10)	41,96 ± 10,01	44,71 ± 5,13	47,06 ± 20,91	27,84 ± 2,96
DF (15)	54,90 ± 7,09	40,39 ± 6,48	39,22 ± 4,75	36,47 ± 8,24
MI (4)	63,53 ± 0,00	54,12 ± 5,39	65,49 ± 7,09	61,18 ± 12,23
MI (7)	57,25 ± 1,80	59,61 ± 2,45	58,04 ± 6,79	57,65 ± 2,04
MZ (10)	74,51 ± 0,68	58,82 ± 11,59	50,59 ± 4,71	41,57 ± 2,96
MZ (18)	87,84 ± 11,43	55,29 ± 5,13	55,29 ± 5,88	39,61 ± 3,40
MPZ (10)	NC	65,10 ± 3,40	51,76 ± 7,35	56,86 ± 16,06
MPZ (18)	79,22 ± 1,36	64,71 ± 7,35	48,24 ± 7,06	49,41 ± 5,13

A classificação das letras das placas com o SCG pode ser visualizada na Tabela 5.8. Da mesma forma que com o RP, no SCG o melhor desempenho, cerca de 72%, foi obtido pelos 10 primeiros Descritores de Fourier, utilizando-se 20 neurônios na camada oculta. Assim, para o presente estudo, pode-se afirmar que os 10 primeiros Descritores de Fourier são capazes de caracterizar mais adequadamente a forma das letras, mesmo

com ruído. A adição de mais coeficientes, no caso 15, não se mostrou eficiente, devendo já apresentar parte do ruído existente nas imagens.

Os resultados obtidos com ambos os classificadores foram semelhantes, possibilitando que ambos sejam empregados em aplicações desta natureza.

O comportamento dos MI, dos MZ e MPZ para o SCG é semelhante ao obtido com o RP, tornando válida também a análise dos resultados já efetuada.

Observando-se as Tabelas 5.7 e 5.8 verifica-se a existência em alguns casos de um desvio padrão bastante elevado. Assim sendo, fica evidente a necessidade de realizar o treinamento sucessivas vezes.

Os dígitos das placas para o RP são apresentados na Tabela 5.9. O melhor resultado obtido, cerca de 79% de reconhecimento, ocorreu com os 5 primeiros Descritores de Fourier, com 10 neurônios na camada oculta. Os melhores resultados obtidos também foram com os 5 primeiros Descritores de Fourier para 15 e 20 neurônios na camada oculta. Os melhores resultados com os MZ são de cerca de 75% de reconhecimento e para os MPZ de 77%. Os MI novamente apresentaram os piores resultados.

O desvio padrão é bastante baixo e os resultados são significativamente superiores aos obtidos com as letras.

Tabela 5.9: Classificação dos dígitos das placas com RP

Descritor	RP			
	05	10	15	20
DF (3)	35,58 ± 2,54	34,29 ± 2,00	38,14 ± 1,47	38,14 ± 3,64
DF (5)	25,64 ± 0,56	20,83 ± 2,00	21,47 ± 1,47	25,00 ± 5,00
DF (10)	27,24 ± 2,00	24,04 ± 1,92	27,24 ± 4,93	26,92 ± 1,67
DF (15)	31,41 ± 3,89	32,37 ± 1,47	28,53 ± 2,22	28,85 ± 2,54
MI (4)	39,10 ± 3,38	41,99 ± 4,74	39,10 ± 4,74	41,99 ± 5,30
MI (7)	41,35 ± 3,33	43,27 ± 2,54	45,51 ± 3,89	42,95 ± 4,84
MZ (10)	66,35 ± 8,33	28,85 ± 5,85	25,32 ± 1,11	29,17 ± 5,63
MZ (18)	74,04 ± 0,00	40,38 ± 9,27	29,81 ± 3,47	28,53 ± 1,47
MPZ (10)	64,42 ± 0,00	40,38 ± 4,19	23,72 ± 1,47	29,49 ± 1,47
MPZ (18)	70,19 ± 6,66	52,24 ± 8,01	38,46 ± 4,81	34,29 ± 3,89

Apesar disso, os melhores resultados para os dígitos foram obtidos com o SCG, Tabela 5.10, onde em vários casos obteve-se mais de 75% de reconhecimento e chegando-se aos melhores casos com 20 neurônios a cerca de 78% de reconhecimento. Os DF, MZ e MPZ com 20 neurônios na camada oculta possuem resultados bastante semelhantes. Com 10 neurônios também foram obtidos resultados próximos a 78%.

Após a análise das Tabelas 5.9 e 5.10 constata-se que, para os dígitos, os resultados de ambos os classificadores são equivalentes, independentemente se adotado o DF, MZ ou MPZ como descritor de forma. Cabe realçar, no entanto que o número de feições empregado nos DF é muito inferior ao dos MZ e MPZ.

Tabela 5.10: Classificação dos dígitos das placas com SCG

Descritor	SCG			
	05	10	15	20
DF (3)	36,86 ± 1,11	40,38 ± 1,92	40,38 ± 12,94	49,04 ± 11,34
DF (5)	29,49 ± 8,72	21,47 ± 2,42	21,79 ± 3,09	43,59 ± 26,41
DF (10)	26,92 ± 2,54	31,41 ± 5,30	27,24 ± 3,89	28,53 ± 7,83
DF (15)	29,81 ± 7,87	27,88 ± 10,40	28,21 ± 8,18	21,79 ± 5,30
MI (4)	45,51 ± 2,00	39,42 ± 4,19	37,82 ± 3,89	40,06 ± 2,42
MI (7)	50,32 ± 10,72	54,49 ± 11,55	43,91 ± 2,22	48,40 ± 12,80
MZ (10)	57,05 ± 5,47	35,58 ± 5,00	25,96 ± 2,88	26,28 ± 3,89
MZ (18)	62,82 ± 11,59	36,22 ± 9,49	28,53 ± 2,00	26,92 ± 1,67
MPZ (10)	56,09 ± 15,63	27,56 ± 6,82	25,32 ± 2,00	22,12 ± 5,35
MPZ (18)	62,18 ± 6,75	33,01 ± 4,74	25,32 ± 2,22	21,79 ± 4,93

A existência de mais amostras para o treinamento dos dígitos em relação às letras representa um indicativo da significativa melhora nos resultados obtidos. Além disso, a regularidade na quantidade de amostras dos dígitos é bastante superior a das letras, como pôde ser verificado nas Figuras 4.38 e 4.39.

Pode-se verificar que a única maneira de minimizar este problema de distribuição das amostras é através da utilização de um número mais elevado de imagens de placas de automóveis. Deve-se observar, entretanto, que as placas sempre possuirão um número mais elevado de letras específicas pois isto está relacionado com as unidades federativas.

As imagens utilizadas foram adquiridas nas ruas das cidades de Recife e Natal e possuem uma repetição das letras “K” e “M” bem superior a das outras letras. Além das características já citadas, deve-se considerar a dificuldade de obtenção das imagens.

Neste instante é importante trazer uma observação sobre a propriedade de invariância a rotação. Conforme amplamente comentado no decorrer do texto, os métodos aqui apresentados possuem invariância a rotação, escala e translação. Para a presente aplicação, entretanto, seria interessante que os métodos não possuíssem tal propriedade pois dígitos como o “9” e o “6” e letras como o “M” e o “W” podem ser facilmente confundidas se considerarmos a invariância a rotação.

No caso específico deste experimento, optou-se por não acrescentar nenhum indicativo da orientação correta aos classificadores. Principalmente para o caso dos dígitos verificou-se que os classificadores foram capazes de discernir corretamente entre as classes pois apesar da proximidade entre as feições extraídas, elas não são idênticas.

Em função disso, pode-se verificar a robustez dos classificadores para ambos os estudos de caso. Assim, para aprimorar o desempenho de experimentos como estes, pode-se também acrescentar um feição extra indicando a orientação correta do caractere.

Com o intuito de melhorar o desempenho do presente estudo deve-se empregar métodos de segmentação de imagens mais robustos, principalmente os que utilizam imagens coloridas. O presente trabalho foi desenvolvido com ênfase nos descritores de forma e nos classificadores, buscando-se extrair o máximo destes métodos e desta



maneira não foram aplicados algoritmos robustos de segmentação que certamente melhorariam os resultados finais.

## 6 CONCLUSÃO

Este trabalho apresenta o desenvolvimento de um *toolbox* para o reconhecimento de formas através de imagens, tendo como estudos de caso o reconhecimento de caracteres de comandos escritos e o reconhecimento de caracteres de placas de automóveis.

No decorrer do mesmo, fundamentou-se o Reconhecimento de Padrões em Imagens e suas etapas, com ênfase aos Descritores de Forma e aos Classificadores Estatísticos e Neurais.

Detalhou-se os tipos de similaridade, região e contorno, possíveis aos Descritores de Forma. Exemplos práticos foram apresentados a fim de facilitar o entendimento. Os principais descritores de forma com similaridade baseada em região e contorno disponíveis na literatura durante os estudos desta dissertação foram implementados e estão disponíveis no *toolbox*.

Além disso, Classificadores Estatísticos e Neurais foram apresentados e descritos, com o intuito de comparar a aplicação e desempenho de cada uma das abordagens, enriquecendo ainda mais a pesquisa realizada. Através do estudo destes métodos de descrição de formas e classificação desenvolveu-se um *toolbox* que visa auxiliar estudantes e pesquisadores da área de reconhecimento de padrões em imagens e interessados em reconhecimento de formas em imagens. Até a conclusão deste *toolbox* não existia nenhuma ferramenta com código aberto e com os métodos mais relevantes existentes na literatura já implementados.

Os estudos de caso apresentaram resultados bastante animadores. Verificou-se a importância de um bom descritor e de um bom classificador através da experimentação das combinações implementadas. Também foi possível observar que existem combinações de descritores e classificadores mais eficientes e que podem ser utilizados com uma probabilidade maior de obtenção de sucesso em aplicações de natureza semelhante as empregadas nos estudos de caso.

Verificou-se no estudo de caso dos caracteres dos comandos, que os classificadores neurais foram superiores aos classificadores estatísticos, sendo que os classificadores neurais mais estáveis foram o RP (*Resilient Propagation*) e o SCG (*Scaled Conjugate Gradient*). Efetuando a análise dos descritores por grupos, constatou-se que os Momentos Invariantes (MI) foram os mais efetivos dentre os métodos de similaridade por região baseados em momentos geométricos. Os Momentos de Zernike (MZ) e Momentos Pseudo-Zernike (MPZ), também com similaridade por região, mas baseados em momentos ortogonais obtiveram resultados equivalentes em seu grupo. Os

Descritores de Fourier (DF) foram mais efetivos dentre os métodos de similaridade por contorno.

Assim sendo, neste estudo, tanto os descritores com similaridade por região quanto por contorno obtiveram resultados equivalentes, com uma leve vantagem aos por contorno, pois o número de feições empregado é menor bem como o custo computacional.

No estudo de caso das placas, o mesmo pôde ser verificado. Entretanto, um classificador neural é capaz, em grande parte das vezes, de adequar-se e conseguir realizar a classificação de maneira eficiente mesmo que as feições não sejam totalmente discriminantes. Resultados equivalentes foram obtidos com os classificadores RP (*Resilient Propagation*) e o SCG (*Scaled Conjugate Gradient*). Para decidir qual classificador é mais adequado para uma aplicação prática necessita-se de uma análise estatística. Para este estudo, os Descritores de Fourier (DF) foram mais efetivos. Resultados animadores obtidos com baixo número de descritores e neurônios na camada oculta.

Os resultados obtidos para as Placas indicam que o emprego da combinação de feições e classificador escolhidos melhora o desempenho geral de reconhecimento em relação a outros trabalhos desenvolvidos na UFRGS.

Assim, é possível concluir que o *Toolbox de Descrição de Forma* desenvolvido cumpre com seu propósito, disponibilizando diversos métodos e possibilitando ao pesquisador/interessado analisar de uma forma mais rápida suas opções visando encontrar os métodos mais adequados para a solução do problema pesquisado.

A criação de um banco de imagens com placas de automóveis já será de grande valia para outros pesquisadores principalmente por possibilitar a realização de comparações de desempenho.

Para trabalhos futuros, sugere-se o estudo de métodos de segmentação de imagens coloridas para adição ao *toolbox*, por exemplo, o algoritmo *Mean Shift*. Com isto, a robustez e utilização do *toolbox* será elevada, pois permitirá o desenvolvimento de aplicações como o controle de tráfego em tempo real.

Sugere-se também o estudo de outros descritores de forma, como o ART (*Angular Radial Transform*) (HÖYNCK; OHM, 2003), os Momentos de Legendre (PROKOP; REEVES, 1992; CHONG; RAVEENDRAN; MUKUNDAM, 2004) e os Descritores de Fourier Generalizados (ZHANG; LU, 2002). Além desses, que são baseados em região, sugere-se a pesquisa de métodos com similaridade baseada em contorno, como a UNLUFF (RAUBER, 1994), BAS (*Beam Angle Statistics*) (ARICA; VURAL, 2003) e os métodos baseados na Transformada Wavelet. O *Curvature Scale Space* (CSS) também poderia ser mais estudado para o propósito de classificação pois apresentou resultados bastante satisfatórios neste estudo preliminar. Algoritmos como o *Quickprop* (FAHLMAN, 1988) e redes neurais construtivas como o CASCOR (*Cascade Correlation*) (FAHLMAN; LEBIERE, 1991) poderiam ser agregados aos classificadores neurais.

Além disso, feições de textura e cor poderiam ser agregadas ao *toolbox* visando atender uma gama maior de aplicações e domínios. O uso de feições combinadas daria um aumento considerável à aplicação do *toolbox*.

E, por fim, partindo-se do que já foi desenvolvido neste trabalho a criação de uma ferramenta para a recuperação de informação visual seria extremamente interessante, servindo de laboratório para novas técnicas e também para capacitar o Instituto de Informática a desenvolver projetos nesta área que está em crescente evolução.

## REFERÊNCIAS

- ABDUL-KAREEM, S.; BABA, S.; ZUBAIRI, Y. Z.; PRASAD, U.; WAHID, M. I. A. Back Propagation Neural Network for Medical Prognosis: A Comparison of Different Training Algorithms, **Electronic Journal of School of Advanced Technologies - AIT**, v.3, Issue 1, Apr 2001.
- ALZHRANI, F. M.; CHEN, T. A Real-Time Edge Detector: **Algorithm and VLSI Architecture. Real-Time Imaging**. [S.l.], v.3, p. 363-378, 1997.
- ARICA, N.; VURAL, F. T. Y. BAS: a perceptual shape descriptor based on the beam angle statistics. **Pattern Recognition Letters**, v.24, p.1627-1639, 2004.
- BIMBO, A. **Visual Information Retrieval**. San Francisco: Morgan Kaufmann, 1999.
- BISHOP, C. M. **Neural Networks for Pattern Recognition**. New York: Oxford University Press, 1995.
- BOBER, M. MPEG-7 Visual Shape Descriptors. **IEEE Transactions on Circuits and Systems for Video Technology**, New York, v.11, n.6, p.716-719, June 2001.
- CAMPOS, T. J. de. **Reconhecimento de Caracteres Alfanuméricos de Placas em Imagens de Veículos**. 2001. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.
- CASTRO, L. N. **Análise e Síntese de Estratégias de Aprendizado para Redes Neurais Artificiais**. 1998. Dissertação (Mestrado em Engenharia Elétrica e de Computação), UNICAMP, Campinas.
- CHIN, Y. C.; KASSIM, A. A.; IBRAHIM, Y. Character recognition using statistical moments. **Image and Vision Computing**, Amsterdam, v.17, p.299-307, 1999.
- CHONG, C.; RAVEENDRAN, P.; MUKUNDAN, R. Translation and scale invariants of Legendre Moments. **Pattern Recognition**, Oxford, v.37, p.119-129, 2004.
- COCQUEREZ, J.; et al. **Analyse d'images: filtrage et segmentation**. Paris: Masson, 1995.
- CORNEY, D. **Clustering Toolbox**. Disponível em: <http://www.cs.ucl.ac.uk/staff/D.Corney/ClusteringToolbox.zip>. Acesso em: 27 ago. 2003.
- DAHMER, A. **Segmentação de Imagens Ecocardiográficas Utilizando Redes Neurais e Medidas de Textura**. 1998. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

- DUDA, R. O.; HART, P. E. **Pattern Classification and Scene Analysis**. New York: Wiley, 1973.
- ENGEL, P. M.; NUNES, R. V. IRENE: Intelligent Processing of Multispectral Images by Self-Organizing Maps. **International Symposium on Artificial Neural Networks**, Tainan, Taiwan, December 15-17, 1994.
- ERBERT, M.; HAERTEL, V. Estudo sobre Técnicas de Regularização da Matriz Covariância no Processo de Classificação de Dados em Alta Dimensionalidade. **Anais XI SBSR**, Belo Horizonte, Brasil, 05- 10 abril 2003, INPE, p. 1061-1068.
- FAHLMAN, S. E. An Empirical Study of Learning Speed in Back-Propagation Networks. **Technical Report CMU-CS-88-162**, Carnegie-Mellon University, School of Computer Science, Pittsburgh, 1988.
- FAHLMAN, S. E.; LEBIERE, C. The Cascade-Correlation Learning Architecture. **Technical Report CMU-CS-90-100**, Carnegie-Mellon University, School of Computer Science, Pittsburgh, 1991.
- FALOUTSOS, C.; EQUITZ, W.; FLICKNER, M.; NIBLACK, W.; PETKOVIC, D.; BARBER, R. Efficient and Effective Querying by Image Content, **Journal of Intelligent Information Systems**, v.3, n.3-4, p.231-262, 1994.
- FLUSSER, J.; SUK, T. Pattern Recognition by Affine Moment Invariants. **Pattern Recognition**, Oxford, v.26, p.167-174, 1993.
- FREEMAN, J. A.; SKAPURA, D. M. **Neural Networks: Algorithms, Applications, and Programming Techniques**. Addison-Wesley, 1991.
- GONZALEZ, R.; WOODS, R. E. **Processamento de Imagens Digitais**. São Paulo: E. Blücher, 2000.
- HAYKIN, S. **Redes Neurais: princípios e prática**. 2.ed. Porto Alegre: Bookman, 2001.
- HERMES, T.; KLAUCK, C.; KREYS, J; ZHANG, J. Image Retrieval for Information Systems. In: **Storage and Retrieval for Image and Video Databases**, v.2420, p.394-405, San Jose, 1995. Disponível em: <[http:// citeseer.ist.psu.edu/hermes95image.html](http://citeseer.ist.psu.edu/hermes95image.html)>. Acesso em 1 dez. 2003.
- HOFFBECK, J. P.; LANDGREBE, D. A. Covariance Matrix Estimation and Classification with Limited Training Data. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v.18, n.7, p.763-767, 1996.
- HÖYNCK, M.; OHM, J. Shape Retrieval with Robustness against Partial Occlusion. In: **Proceeding of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'03)**, Hong Kong, China, v.3, p. 593-596, 2003.
- HU, M. K. Visual Pattern Recognition by Moment Invariants. **IRE Transactions on Information Theory**, [S.l.], v.8, p.179-187, Feb. 1962.
- ISO. **ISO/IEC JTC1/SC29/WG11/N3914: MPEG-7 Visual part of eXperimental Model Version 9.0**. Pisa, 2001.
- IYODA, E. M. **Inteligência Computacional no Projeto Automático de Redes Neurais Híbridas e Redes Neurofuzzy Heterogêneas**. 2000. Dissertação (Mestrado em Engenharia Elétrica e de Computação) – Faculdade de Engenharia Elétrica e de Computação, UNICAMP, Campinas.

- JAIN, A. K. **Fundamentals of digital image processing**. Englewood Cliffs: Prentice Hall, 1989.
- JAIN, A. K.; DUIN, R. P. W.; MAO, J. Statistical Pattern Recognition: A Review. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v.22, n.1, p.4-37, 2000.
- JÄHNE, B. **Digital Image Processing : Concepts, Algorithms, and Scientific Applications**. Springer-Verlag, New York, 1997.
- KHOTANZAD, A.; HONG, Y. H. Invariant Image Recognition by Zernike Moments. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, New York, v.12, n.5, p.489-497, 1990.
- KHOTANZAD, A.; LU, J. Object Recognition Using a Neural Network and Invariant Zernike Features. **IEEE Computer Society Conference on Computer Vision and Pattern Recognition**, San Diego, p.200-205, 1989.
- KHOTANZAD, A.; LU, J. Classification of Invariant Image Representations Using a Neural Network. **IEEE Transactions on Acoustics, Speech, and Signal Processing**, New York, v.38, n.6, p.1028-1038, 1990.
- KIM, W.; KIM, Y. A Region-based Shape Descriptor Using Zernike Moments. **Signal Processing: Image Communication**, n.16, p.95-102, 2000.
- KULKARNI, A. D. **Artificial Neural Networks for Image Understanding**. New York, USA: Van Nostrand Reinhold, 1994.
- LIAO, S. X.; PAWLAK, M. On the Accuracy of Zernike Moments for Image Analysis. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, New York, v.20, n.12, p.1358-1364, 1998.
- LIBERMAN, F. **Classificação de Imagens Digitais por Textura usando Redes Neurais**. 1997. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.
- MA, W.; MANJUNATH, B. Netra: A Toolbox for Navigating Large Image Databases. **Multimedia Systems**, Springer-Verlag, v.7, n.3, p.184-198, 1999. Disponível em: <<http://citeseer.ist.psu.edu/ma99netra.html>>. Acesso em 1 dez. 2003.
- MARR, D. **Vision: a Computational Investigation into the Human Representation and Processing of Visual Information**. New York: W. M. Freeman and Company, 1982.
- METHRE, B. M.; KANKANHALLI, M. S.; LEE, W. F. Shape Measures For Content Based Image Retrieval: a Comparison. **Information Processing & Management**, v.33, n.3, p.319-337, 1997.
- MOKHTARIAN, F.; MACKWORTH, A. A Theory of Multiscale, Curvature-Based Shape Representation for Planar Shapes. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, New York, v.14, n.8, p.789-805, 1992.
- MOKHTARIAN, F. Silhouette-Based Isolated Object Recognition through Curvature Scale Space. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, New York, v.17, n.5, p.539-544, 1995.
- MOKHTARIAN, F.; ABBASI, S.; KITTLER, J. Robust and Efficient Shape Indexing through Curvature Scale Space. In: **Proceedings of the 1996 British Machine na Vision Conference BMVC'96**, Edinburgh: Scotland, Sep, 1996.

MOKHTARIAN, F.; SUOMELA, R. Robust Image Corner Detection Through Curvature Scale Space. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, New York, v.20, n.12, p.1376-1381, 1998.

MOLZ, R. F. **Uma Metodologia para o Desenvolvimento de Aplicações de Visão Computacional utilizando um projeto conjunto de Hardware e Software**. 2001. Tese (Doutorado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

MOREIRA, M.; FIESLER, E. Neural Networks with Adaptive Learning Rate and Momentum Terms, **Technical Report 95-04**, Institut Dalle Molle D'Intelligence Artificielle Perceptive, Valais, Suisse, 1995.

MØLLER, M. F. A Scaled Conjugate Gradient Algorithm for fast Supervised Learning. **Neural Networks**, v.6, n.4, p.525-533, 1993.

NUNES, R. V. **Uma Abordagem Neural para Tratamento de Imagens Multiespectrais**. 1995. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

PENTLAND, A.; PICARD, W.; SCLAROFF, S. Photobook: Content-Based Manipulation of Image Databases. In: **SPIE Storage and Retrieval for Image and Video Databases II**, San Jose, n.2185, 1995.

PICCOLI, L. **Segmentação e Classificação de Imagens Ecocardiográficas Utilizando Redes Neurais**. 1999. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

PROKOP, R. J.; REEVES, A. P. A Survey of Moment-Based Techniques for Unoccluded Object Representation and Recognition. **CVGIP: Graphical Models and Image Processing**, v.54, n.5, p.438-460, 1992.

RAUBER, T. W. Two-Dimensional Shape Description. **Technical Report GR UNINOVA-RT-10-94**, Universidade Nova de Lisboa, Faculdade de Ciências e Tecnologia & Intelligent Robotics Center, Lisboa, Portugal, 1994.

REISS, T. H. **Recognition Planar Objects Using Invariant Image Features**. Berlin: Springer-Verlag, 1993.

RICHARDS, J. A. **Remote Sensing Digital Image Analysis**. Berlin: Springer-Verlag, 1993.

RIEDMILLER, M.; BRAUN, H. RPROP – A Fast Adaptive Learning Algorithm. **Technical Report (Also Proceedings of ISCIS VII)**, Universität Karlsruhe, 1992.

RIEDMILLER, M. RPROP – Description and Implementation Details. **Technical Report**, Universität Karlsruhe, 1994.

SAFAR, M.; SHAHABI, C.; SUN, X. Image Retrieval By Shape: A Comparative Study. **IEEE International Conference on Multimedia and Expo (I)**, p.141-154, 2000. Disponível em: <<http://citeseer.nj.nec.com/safar99image.html>>. Acesso em 6 ago. 2003.

SMAGT, P. P. V. D Minimisation Methods for Training Feedforward Neural Networks, **Neural Networks**, v.7, n.1, p.1-11, 1994.

SONKA, M.; HLAVAC, V.; BOYLE, R. **Image Processing, Analysis, and Machine Vision**. 2nd. Ed. Pacific Grove, CA: PWS Publishing, 1999.



- SQUIRE, D. M. **Model-Based Neural Networks For Invariant Pattern Recognition**. 1996. Thesis (Phd) – Curtin Univesrisy of Technology.
- SZMURLO, M. **A Comparative Study of Statistically Classifiable Features Used Within The Area of Optical Character Recognition**. 1995. Thesis (Master) – University of Oslo, Norway. Disponível em: <<http://falcone.info.unicaen.fr/publications/papiers/master,thesis.ps.gz>>. Acesso em: 7 set. 2003.
- TEAGUE, M. Image analysis via the general theory of moments. **Journal Optical Society of America**, v.70, n.8, p.920-930, 1980.
- TEH, C.; CHIN, R. On Image Analysis by the Methods of Moments. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, New York, v.10, n.4, p.496-513, 1988.
- TRIER, Ø. D.; JAIN, A. K.; TAXT, T. Feature Extraction Methods for Character Recognition – A Survey. **Pattern Recognition**, v.29, n.4, p. 641-662, 1996.
- VALIATI, J. F. **Reconhecimento de Voz para Comandos de Direcionamento por meio de Redes Neurais**. 2000. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.
- WHELAN, P. F.; MOLLOY, D. **Machine Vision Algorithms in Java: Techniques and Implementation**. London: Springer Verlag, 2001.
- ZHANG, D. **Image Retrieval Based on Shape**. 2002. Thesis (Phd) - Faculty of Information Technology, Monash. Disponível em: <<http://www.gscit.monash.edu.au/~dengs/resource/papers/thesis.zip>>. Acesso em: 23 jul. 2003.
- ZHANG, D.; LU, G. Shape-based image retrieval using generic Fourier Descriptors. **Signal Processing: Image Communication**, n.17, p.825-848, 2002.