

Descritores de forma baseados em *Tensor Scale*

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Fernanda Alcântara Andaló e aprovada pela Banca Examinadora.

Campinas, 02 de Março de 2007.

Prof. Dr. Ricardo da Silva Torres (Orientador)

Prof. Dr. Alexandre Xavier Falcão
(Co-orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Substitua pela ficha catalográfica

(Esta página deve ser o verso da página anterior mesmo no caso em que não se imprime frente e verso, i.é., até 100 páginas.)

Substitua pela folha com as assinaturas da banca

Descritores de forma baseados em *Tensor Scale*

Fernanda Alcântara Andaló¹

Fevereiro de 2007

Banca Examinadora:

- Prof. Dr. Ricardo da Silva Torres (Orientador)
- Prof. Dr. Roberto Marcondes Cesar Jr.
Instituto de Matemática e Estatística (IME) – USP
- Prof. Dr. Neucimar Jerônimo Leite
Instituto de Computação (IC) – Unicamp
- Prof. Dr. Siome Klein Goldenstein (Suplente)
Instituto de Computação (IC) – Unicamp

¹Financiado pelo CNPq (processo número 134169/2005-0), FAEPEX, FAPESP e projeto Microsoft Escience.

Prefácio

Recentemente o número de coleções de imagens disponíveis vem crescendo. Conseqüentemente, surge a demanda por sistemas de informação para armazenamento, indexação e busca destas imagens. Uma das principais soluções adotadas é a utilização de sistemas de recuperação de imagem por conteúdo que possuem a habilidade de, dada uma imagem de consulta, retornar as imagens mais similares em uma base de dados. Para viabilizar este tipo de consulta, é importante que o processo de caracterização do conteúdo seja automatizado, destacando-se, neste contexto, o uso de descritores de imagem baseados na cor, textura ou forma dos objetos contidos nas imagens. Neste trabalho, são propostos descritores de forma baseados em *Tensor Scale*. *Tensor Scale* é um parâmetro morfométrico que unifica a representação de orientação, espessura e anisotropia de estruturas locais na imagem, que pode ser utilizado em várias aplicações de visão computacional e processamento de imagem. Além dos descritores de forma baseados neste parâmetro morfométrico, este trabalho apresenta um estudo de algoritmos para cálculo do *Tensor Scale*.

As principais contribuições deste trabalho são: (i) estudo de descritores de imagens baseados em cor e textura e, mais extensivamente, descritores baseados em forma; (ii) estudo de algoritmos para cálculo do *Tensor Scale*; (iii) proposta e implementação de detector de saliências de contorno baseado em *Tensor Scale*; (iv) proposta e implementação de novos descritores de forma baseados em *Tensor Scale*; e (v) validação dos descritores propostos quanto à sua utilização em sistemas de recuperação de imagens por conteúdo, por meio de experimentos comparativos com outros descritores de forma relevantes, recentemente propostos.

Abstract

In the past few years, the number of image collections available has increased. In this scenery, there is a demand for information systems for storing, indexing, and retrieving these images. One of the main adopted solutions is to use content-based image retrieval systems (CBIR), that have the ability to, for a given query image, return the most similar images stored in the database. To answer this kind of query, it is important to have an automated process for content characterization and, for this purpose, the CBIR systems use image descriptors based on color, texture and shape of the objects within the images. In this work, we propose shape descriptors based on Tensor Scale. Tensor Scale is a morphometric parameter that unifies the representation of local structure thickness, orientation, and anisotropy, which can be used in several computer vision and image processing tasks. Besides the shape descriptors based on this morphometric parameter, we present a study of algorithms for Tensor Scale computation.

The main contributions of this work are: *(i)* study of image descriptors based on color, texture and shape descriptors; *(ii)* study of algorithms for Tensor Scale computation; *(iii)* proposal and implementation of a contour salience detector based on Tensor Scale; *(iv)* proposal and implementation of new shape descriptors based on Tensor Scale; and *(v)* validation of the proposed descriptors with regard to their use in content-based image retrieval systems, comparing them, experimentally, to other relevant shape descriptors, recently proposed.

Agradecimentos

A Deus pela vida e tudo o que nela tenho conquistado.

Aos meus pais, Fernando e Tânia, e à minha irmã Juliana, pela paciência em me ouvir e a sensatez ao me aconselhar e ajudar.

Ao meu querido namorado Diego, pelo amor, companheirismo, dedicação e conhecimento.

Ao professor Ricardo Torres, cujas qualidades, conhecimento e postura possibilitaram não só a conclusão deste trabalho, mas também meu amadurecimento como pesquisadora. E ao professor Alexandre Falcão, pelos ensinamentos que foram imprescindíveis para este trabalho.

Aos professores Roberto Marcondes Cesar Jr., Neucimar Jerônimo Leite e Siome Klein Goldenstein por gentilmente aceitarem avaliar este trabalho.

Aos professores da UnB que me ensinaram, com dedicação, muito do conhecimento que hoje emprego no trabalho e na vida, em especial Dra. Suzete Venturelli e Dr. Aluizio Arcela.

A todos os familiares: tios, primos e vó Lili; em especial à prima Lívia, pelo exemplo. E aos eternos amigos Flávia, Denise e Vinícius, pelas alegrias, amizade e compreensão.

A todos que, de alguma maneira, me ajudaram na conclusão deste trabalho: docentes, funcionários e colegas do Instituto de Computação da UNICAMP.

Ao CNPq, pelo apoio financeiro. Este trabalho também foi parcialmente financiado pela agência FAPESP, pelo fundo FAEPEX e pelo projeto Microsoft Escience.

Este trabalho é dedicado à minha avó Tana (*in memoriam*) por tudo o que ela significou na minha vida.

Sumário

Prefácio	v
Abstract	vi
Agradecimentos	vii
1 Introdução	1
2 Trabalhos relacionados	6
2.1 Terminologia e conceitos fundamentais	6
2.2 Recuperação de imagem por conteúdo	6
2.2.1 Arquitetura típica de sistemas CBIR	6
2.2.2 Sistemas CBIR	8
2.3 Descritores de imagem	9
2.3.1 Descritores de cor	10
2.3.2 Descritores de textura	12
2.4 Descritores de forma	12
2.5 Resumo	16
3 <i>Tensor Scale</i>	17
3.1 Conceitos de <i>Tensor Scale</i>	18
3.2 Algoritmos para cálculo do <i>Tensor Scale</i>	21
3.2.1 Algoritmo de Punam	21
3.2.2 Algoritmo de Miranda et al.	25
3.2.3 Algoritmo para cálculo do <i>Tensor Scale</i> via Transformada Imagem-Floresta	27
3.3 Descritor <i>Tensor Scale</i> – TSD	35
3.3.1 Função de extração do vetor de características (ϵ_{TSD})	35
3.3.2 Função de similaridade (δ_{TSD})	36
3.4 Resumo	37

4	Descritor de forma <i>Tensor Scale</i> por zonas de influência	38
4.1	Função de extração do vetor de características (ϵ_{TSDIZ})	38
4.1.1	Computação das elipses <i>Tensor Scale</i>	39
4.1.2	Mapeamento das orientações das elipses para os segmentos de contorno do objeto	39
4.1.3	Cálculo da média angular	40
4.1.4	Formação do vetor de características	42
4.2	Função de similaridade (δ_{TSDIZ})	42
4.3	Experimentos	44
4.3.1	Base de imagens	44
4.3.2	Resultados	45
4.4	Resumo	54
5	Descritor de forma baseado em saliências do contorno detectadas por <i>Tensor Scale</i>	55
5.1	Função de extração do vetor de características (ϵ_{TSCS})	56
5.1.1	Detecção de pontos de saliência	56
5.1.2	Cálculo dos valores das saliências detectadas	60
5.1.3	Formação do vetor de características	60
5.2	Função de similaridade (δ_{TSCS})	61
5.3	Experimentos	63
5.3.1	Base de imagens	63
5.3.2	Resultados	63
5.4	Resumo	67
6	Conclusões	68
	Bibliografia	71

Lista de Tabelas

4.1	Taxa de similaridade para o descritor BAS e para o TSDIZ, considerando diferentes números de segmento no contorno.	49
5.1	Medidas de eficácia para os métodos de esqueleto e TS. Para cada valor de limiar testado no método TS, a tabela apresenta o valor absoluto das medidas e, em parêntesis, a porcentagem de ganho em relação ao método baseado em esqueleto.	65

Lista de Figuras

1.1	Uma imagem contendo um objeto (a) e sua silhueta 2D (b).	2
1.2	Objetos facilmente identificáveis (formas da base de imagens MPEG-7 CE-shape-1 part B [2]).	3
2.1	<i>Framework</i> de sistema de recuperação de imagem por conteúdo.	7
2.2	Arquitetura típica de um sistema de recuperação de imagem por conteúdo. . . .	8
2.3	Uso de um descritor D para calcular a distância entre as imagens \hat{I}_A e \hat{I}_B	10
2.4	Classificação de técnicas de representação e descrição de forma. Figura adaptada de [77].	14
3.1	Fatores do <i>Tensor Scale</i>	18
3.2	Exemplo de visualização do <i>Tensor Scale</i> no espaço HSI.	19
3.3	Exemplo de elipses <i>Tensor Scale</i> para formas básicas.	20
3.4	Etapas do algoritmo de Punam para o cálculo do <i>Tensor Scale</i>	22
3.5	Exemplo de utilização dos limiares th_1 e th_2	26
3.6	Primeiro passo da perseguição de borda.	32
3.7	Segundo passo da perseguição de borda, após a situação ilustrada na Figura 3.6.	33
3.8	Terceiro passo da perseguição de borda, após a situação ilustrada na Figura 3.7.	35
3.9	Exemplo da primeira etapa de ε_{TSD} para duas imagens.	36
3.10	Exemplo da segunda etapa de ε_{TSD} para as duas imagens da Figura 3.9(a).	37
3.11	Exemplo da função δ_{TSD} para as duas imagens da Figura 3.9(a).	37
4.1	Contorno com 10 segmentos de igual comprimento e com rótulos diferentes.	39
4.2	Dados axiais circulares.	41
4.3	Exemplos de curvas de orientação obtidas com o descritor TSDIZ.	43
4.4	Curvas registradas das imagens presentes nas Figuras 4.3(a) e 4.3(b).	45
4.5	Curvas Precisão vs. Revocação para o descritor TSDIZ, considerando diferentes números de segmentos no contorno.	47
4.6	Curva de Precisão vs. Revocação para diversos descritores.	48
4.7	Taxas de similaridade do descritor TSDIZ, considerando diferentes números de segmento no contorno.	49

4.8	Exemplos de recuperação por similaridade nos quais o TSDIZ apresenta melhor resultado.	51
4.9	Exemplos de recuperação por similaridade nos quais o BAS apresenta melhor resultado.	51
4.10	Taxa de similaridade para todas as classes da base MPEG-7 CE-shape-1 part B.	52
4.11	Curva de separabilidade multiescala para a base de imagens MPEG-7 CE-shape-1 part B.	53
5.1	Registro de imagens a partir de pontos de saliência.	55
5.2	Mapeamento do <i>Tensor Scale</i>	57
5.3	Exemplo de valores de diferença ao longo do contorno de um objeto.	59
5.4	Exemplos de visualização dos pontos de saliência detectados para duas imagens.	59
5.5	Zonas de influência de pontos convexos (<i>a</i> , <i>b</i> , <i>d</i> e <i>e</i>) e côncavo (<i>c</i>).	60
5.6	Exemplo de formação do vetor de características do descritor TSCS.	61
5.7	Granularidade dos métodos baseados em esqueleto e em <i>Tensor Scale</i>	64
5.8	Exemplos de imagens de <i>ground-truth</i>	65
5.9	Curva de separabilidade multiescala para a base de imagens <i>Fish-shape</i>	66

Lista de Algoritmos

3.1	Transformada Imagem-Floresta.	29
3.2	Transformada de Distância Euclidiana via IFT.	31
3.3	Perseguição de bordas da elipse.	34
4.1	Mapeamento das orientações para os segmentos do contorno do objeto.	40
4.2	Similaridade entre dois vetores TSDIZ.	44
5.1	Mapeamento das orientações para o contorno do objeto.	58
5.2	Similaridade entre dois vetores TSCS.	62

Capítulo 1

Introdução

As novas tecnologias que auxiliam a aquisição e o armazenamento de imagens e o rápido crescimento e popularização da *World Wide Web* fazem crescer o número de imagens disponíveis. Conseqüentemente, este crescimento acarreta uma maior necessidade por sistemas de informação para armazenamento, indexação e busca destas imagens [64]. A realização de buscas de imagens consiste em um dos mais importantes serviços que sistemas deste tipo devem prover.

O desafio reside no fato de que quanto mais informação disponível sobre um determinado tópico, mais difícil é a localização acurada de informação relevante. E este paradoxo não é diferente para imagens: quanto maior forem as coleções, mais complexo é o problema de buscar as imagens mais relevantes para uma dada consulta.

Em um primeiro momento, a única maneira de realizar tais buscas em coleções de imagens era pela indexação de palavras-chave associadas a cada imagem [69]. Porém, com o crescimento das coleções de imagens ficou inviável o esforço para realizar tais anotações. Outro fator limitante reside na natureza subjetiva do processo de anotação, ou seja, o entendimento da imagem depende da percepção humana. Neste caso, usuários podem utilizar diversos termos para anotar uma determinada imagem ou o mesmo usuário, em momentos distintos, pode anotar diferentemente uma mesma imagem. Esta subjetividade no processo de anotação limita a efetividade de sistemas de busca de imagens baseados em palavras-chave.

Devido às inúmeras deficiências dos sistemas de busca baseados em palavras-chave, deu-se início à pesquisa em recuperação de imagens por conteúdo. A principal vantagem desta metodologia é a possibilidade de se automatizar o processo de recuperação das imagens, solucionando o problema primário relacionado às anotações.

O projeto de sistemas de recuperação de imagens por conteúdo (*Content-based image retrieval* – CBIR) tem sido um desafio e vem recebendo considerável atenção desde meados dos anos 90, quando esforços começaram a ser realizados para manipular e buscar eficientemente grandes volumes de imagens a partir de seus conteúdos (propriedades visuais de seus *pixels*).

Em sistemas CBIR, algoritmos de processamento de imagens são utilizados para a extração de vetores de características que representam as propriedades da imagem como cor, textura e forma. Neste sentido, é possível recuperar automaticamente imagens similares a um padrão de consulta (em geral, uma outra imagem) definido pelo usuário (*query-by-example*). Tais algoritmos devem ser capazes de captar a semântica presente na imagem, destacando-se, neste contexto, o uso de descritores dos objetos/regiões contidos na imagem. Estes descritores podem ser representados por duas funções: uma de extração de vetor de característica que representa as propriedades extraídas da imagem; e uma função de distância que mede a similaridade entre imagens a partir de seus vetores de características.

As características extraídas e a função de distância podem depender do domínio das imagens que estão sendo analisadas [64]. Um domínio específico tem variabilidade limitada e previsível em todos os aspectos relevantes do conteúdo, como em coleções de imagens de peixes, impressões digitais ou fotos de asas de mosca; em um domínio geral verifica-se uma variabilidade ilimitada e imprevisível, como ocorre em imagens disponíveis pela Internet.

Nestas coleções de imagens, dentre todos os aspectos referentes à informação visual, a forma certamente é uma característica importante para reconhecimento dos objetos. As formas em duas dimensões (2D) são formadas pelo mapeamento dos objetos tridimensionais (3D) em estruturas 2D, como a retina dos olhos ou lentes de uma câmera. Até mesmo as silhuetas de objetos 2D quase sempre possuem informação suficiente para o reconhecimento do objeto original [19], como pode ser observado na Figura 1.1.

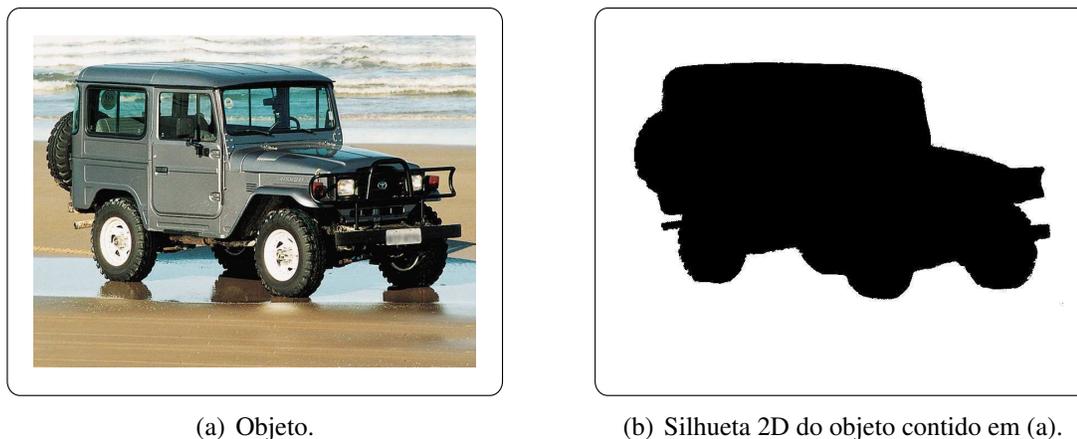


Figura 1.1: Uma imagem contendo um objeto (a) e sua silhueta 2D (b).

Seres humanos geralmente são capazes de reconhecer objetos apenas por sua forma, prova de que esta característica carrega informação semântica [9]. Na Figura 1.2, embora não tenhamos informação visual de cor, profundidade, textura e movimento, os objetos representados pelas silhuetas podem ser prontamente reconhecidos [19]. Desta maneira, a forma se distingue de outras características visuais, como a cor, movimento ou textura, que sozinhas não são suficientes para revelar a identidade de um objeto.

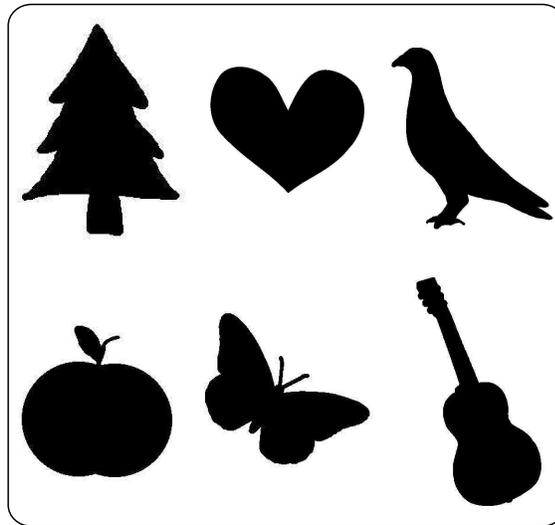


Figura 1.2: Objetos facilmente identificáveis (formas da base de imagens MPEG-7 CE-shape-1 part B [2]).

Muitas aplicações que envolvem análise e recuperação visual utilizam características da forma. Na área de segurança, por exemplo, pode-se utilizar análise da forma de objetos contidos em uma imagem na prevenção de crimes (detecção de impressão digital, face, íris, etc). Um sistema de vigilância por vídeo pode utilizar técnicas baseadas em forma para detectar intrusos [9]. Na engenharia, tem-se exemplo de aplicação da análise da forma no controle de qualidade (verificação da corretude dos componentes produzidos) [19].

Uma aplicação atual de bastante importância é a utilização de descritores em sistemas de biodiversidade, para auxílio na identificação e classificação de espécies de seres vivos (e.g., DrawWing [68]). Este tipo de sistema vem sendo usado com diversos objetivos como, por exemplo, controle de pestes na agricultura, formulação de legislação de conservação, assistência em aspectos farmacológicos e monitoramento da disseminação de doenças e da poluição [73], etc. Em [24], Torres et al. propuseram um *framework* de bibliotecas digitais para gerenciar dados de biodiversidade, combinando técnicas de recuperação de imagem por conteúdo e mecanismos de processamento de buscas textuais, para englobar, em um mesmo sistema, características

ecológicas, geográficas e visuais (cor, textura ou forma). Este sistema foi validado com experimentos baseados na forma de diversas espécies de peixes.

Mais especificamente no domínio da *Internet*, algumas aplicações de recuperação de imagem baseadas na forma dos objetos podem ser citadas: gerenciamento de galerias de arte e museus, sistemas de informação geográfica, gerenciamento de bases de *trademark* e *copyright* e arquivamento de fotografias [35].

Em aplicações médicas, a inclusão de características da forma causa grande impacto na área de ensino e em diagnósticos. No ensino, pode auxiliar nas aulas e também ajudar alunos que queiram pesquisar em repositórios de imagens educacionais, inspecionando visualmente os resultados encontrados. Em diagnósticos, por exemplo, esta inclusão permite a recuperação de casos com similaridade visual, que dificilmente seriam associados pois possuem diagnósticos diferentes [54].

Motivado por estas aplicações, o presente trabalho tem como objetivo a proposta, implementação e validação de descritores de forma que possam ser utilizados na recuperação de imagens por conteúdo, em um domínio geral de imagens.

A metodologia adotada utiliza um descritor previamente proposto – o descritor *Tensor Scale* (TSD) [52] – como base para o estudo. Tal descritor se baseia em parâmetros morfométricos locais, chamados *Tensor Scale* [61], que provêm uma representação unificada de anisotropia, orientação e espessura de estruturas locais contidas em objetos de uma imagem. Isto é, para qualquer ponto em uma imagem, seu *Tensor Scale* é representado pela maior elipse (2D), ou elipsóide (3D), centrada no ponto e dentro de uma mesma região homogênea.

O trabalho desenvolvido contribui não só com a proposta de descritores baseados em *Tensor Scale*, que visam acrescentar melhorias no método do TSD previamente proposto, como também com estudo de algoritmos mais eficientes para o cálculo destes parâmetros morfométricos. Como existem diversas aplicações de *Tensor Scale* no domínio de processamento e análise de imagens, estes algoritmos também podem ser aplicados na solução de outros problemas que não sejam especificamente relacionados à descrição de imagens, como segmentação, *clustering*, classificação, registro e filtragem de imagens.

As contribuições deste trabalho visam facilitar a recuperação de imagens por conteúdo. Mais especificamente, podem-se destacar:

- Estudo de descritores de imagens baseados em cor e textura e, mais extensivamente, descritores de forma (Capítulo 2);
- Estudo de novos algoritmos para cálculo do *Tensor Scale* (Capítulo 3);
- Proposta e implementação de detector de saliências de contorno baseado em *Tensor Scale* (Capítulo 5).

- Proposta e implementação de novos descritores de forma baseados em *Tensor Scale* (Capítulos 4 e 5);
- Validação dos descritores propostos quanto à recuperação de imagens por conteúdo, por meio de comparação com outros descritores de forma relevantes, recentemente propostos (Capítulos 4 e 5).

Este capítulo apresentou o contexto do trabalho, seus objetivos e contribuições. O restante da dissertação está organizado da seguinte forma:

- O capítulo 2 introduz alguns conceitos iniciais importantes para o entendimento do trabalho. Apresenta, também, uma revisão bibliográfica contendo os principais projetos nas linhas de pesquisa que este trabalho abrange: recuperação de imagem por conteúdo e descritores de imagem, com maior enfoque em descrição de forma.
- O capítulo 3 aborda os conceitos matemáticos nos quais os descritores propostos estão embasados, os algoritmos de cálculo do *Tensor Scale* e sua primeira utilização para a descrição de imagens.
- Os capítulos 4 e 5 contemplam a apresentação dos novos descritores propostos, com o detalhamento dos algoritmos de extração de características e de cálculo de similaridades. Nestes capítulos, os experimentos realizados também são apresentados e os resultados obtidos são avaliados.
- O capítulo 6 apresenta as conclusões do trabalho e as direções para trabalhos futuros.

Capítulo 2

Trabalhos relacionados

2.1 Terminologia e conceitos fundamentais

Uma imagem digital \hat{I} é um par (D_I, \vec{I}) , onde D_I é um conjunto de pontos no espaço Z^n (domínio da imagem), denominados *spels* (*space elements*), e \vec{I} é um mapeamento vetorial que associa a cada *spel* p em D_I um conjunto $\{I_1(p), I_2(p), \dots, I_k(p)\}$ de valores escalares, associados a alguma propriedade física. O valor de n refere-se à dimensão da imagem e o valor de k ao número de bandas [29].

Neste trabalho, as imagens digitais utilizadas são bidimensionais, ou seja, $D_I \subset Z^2$, e possuem apenas uma banda I ($k = 1$), onde os *spels* são chamados *pixels* (*picture elements*). Estas imagens são representadas, portanto, por uma matriz de tamanho $N \times M$ (N linhas e M colunas) e os valores $I(p)$ de cada *pixel* são obtidos por amostragem e quantização de uma função contínua $I_c(x, y)$ que descreve a propriedade física correspondente em uma dada região do espaço. No caso de uma foto, temos o brilho, por exemplo. Como os descritores propostos trabalham apenas com imagens binárias, os valores de $I(p)$ podem assumir somente dois valores: 0 e 1, ou seja, fundo e objeto.

2.2 Recuperação de imagem por conteúdo

Esta seção apresenta a arquitetura típica de sistemas CBIR (Subseção 2.2.1) e descreve alguns sistemas relevantes existentes (Subseção 2.2.2).

2.2.1 Arquitetura típica de sistemas CBIR

Um sistema CBIR simples pode ser ilustrado pelo *framework* presente na Figura 2.1. As setas numeradas representam o fluxo do sistema. Para uma coleção de imagens, vetores de características são extraídos e posteriormente armazenados e indexados (setas 1 e 2 da Figura 2.1).

Dada uma imagem de entrada, também chamada de imagem de consulta, o mesmo algoritmo de extração é aplicado (setas 4 e 5) e, a partir desta fase, são calculadas as distâncias entre as características extraídas da imagem de consulta e as extraídas de todas as imagens presentes na coleção (setas 3 e 6). O sistema deve, então, enviar as imagens mais próximas à imagem de entrada para um módulo de interface onde serão visualizadas (seta 8).

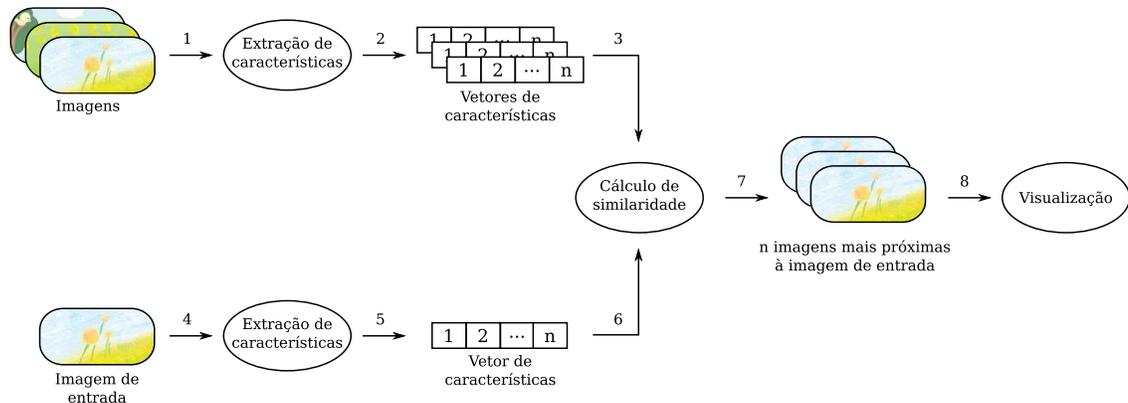


Figura 2.1: *Framework* de sistema de recuperação de imagem por conteúdo.

Este *framework* também pode ser entendido como dois subsistemas: inserção de dados (setas 4 e 5) e processamento de consulta (setas de 1 a 3 e de 6 a 8). A Figura 2.2 ilustra o entendimento do *framework* de forma a destacar estes dois subsistemas.

O subsistema de inserção de dados (módulos e setas tracejados na Figura 2.2) é responsável pela extração de vetores de características das imagens e posterior armazenamento em um banco de dados. O módulo de processamento de consulta se inicia e termina na interface, onde o usuário especifica uma consulta e visualiza as imagens similares recuperadas. É também neste módulo que ocorrem a extração do vetor de características da imagem de consulta e a aplicação de uma função de distância para avaliar a similaridade entre a imagem de consulta e as armazenadas no banco de dados. Por último, este módulo ordena as imagens do banco de dados em ordem decrescente de similaridade em relação à imagem de consulta e as exibe na interface. Na maioria dos casos, as imagens no banco de dados são indexadas de acordo com seus vetores de características, pelo uso de estruturas como *M-tree* [15] ou *Slim-tree* [11], para otimizar a computação da similaridade e da recuperação [21].

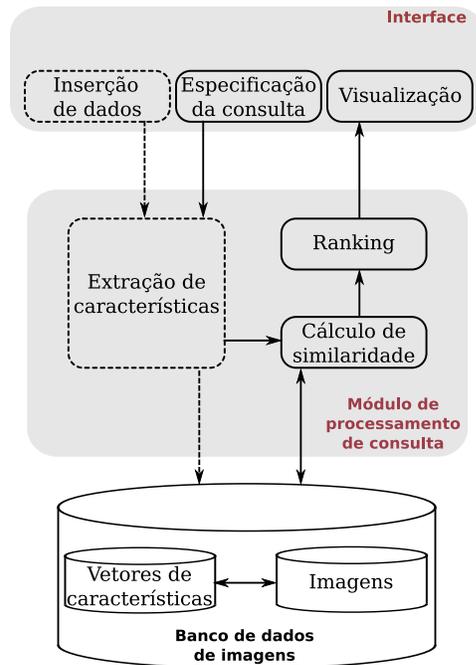


Figura 2.2: Arquitetura típica de um sistema de recuperação de imagem por conteúdo.

2.2.2 Sistemas CBIR

Os primeiros sistemas de recuperação de imagens por conteúdo são datados da década de 80 [13], porém os mais relevantes começaram a ser propostos no meio da década de 90, como o QBIC (*Query by Image Content*) da IBM [33]. Neste sistema, as consultas podem ser realizadas por imagens de exemplo, esboços realizados pelos usuários e/ou padrões de cor e textura. Para o casamento das imagens, são calculadas distâncias Euclidianas ponderadas entre o objeto de consulta e aqueles presentes no banco de dados, além dos casamentos de histogramas e de *templates*. As características de cor e textura são indexadas utilizando-se *R*-trees* [69].

A maioria dos sistemas disponíveis atualmente são acadêmicos, como o Candid [38], Photobook [58] e o Netra [46], e utilizam características simples de forma, cor e textura para descrever o conteúdo das imagens.

O projeto Netra [46] descreve um conjunto de ferramentas para pesquisa em grandes coleções de imagens e, para recuperação de imagens similares, utiliza cor, textura e informações de mais alto nível referentes às regiões segmentadas da imagem, assim como o sistema Blobworld [12]. Já o Photobook [58] implementa três diferentes métodos de representação de imagens para a realização de buscas: faces, formas 2D e textura.

O sistema PicHunter [16] é um *framework* Bayesiano utilizado para auxiliar o usuário na

formulação de sua busca. Ou seja, é uma ferramenta de navegação por imagens que auxilia os usuários a encontrar uma imagem específica, mostrando imagens que maximizam o ganho de informação em cada passo de *feedback*.

Mais recentemente, foi proposto o sistema SMURF (*Similarity-based Multimedia Retrieval Framework*) [71] que incorpora cor, textura e forma na recuperação de imagens.

Uma descrição mais detalhada desses e de outros sistemas CBIR pode ser encontrada em [69].

2.3 Descritores de imagem

Para que sistemas de recuperação de imagem por conteúdo sejam viáveis, é desejável que as imagens possam ser descritas pelas propriedades de seus objetos, tais como forma, textura e cor, normalmente representadas em vetores. Descritores de imagens são utilizados para extrair e comparar estes vetores de características, viabilizando a indexação e busca de imagens.

Formalmente, um vetor de características $\vec{v}_{\hat{I}}$ de uma imagem \hat{I} pode ser considerado como um ponto no espaço \mathbb{R}^n : $\vec{v}_{\hat{I}} = (v_1, v_2, \dots, v_n)$, onde n é a dimensão do vetor [21], e codifica o conteúdo da imagem.

Um possível vetor de características que armazena informações de cor em uma imagem é o histograma [66], que registra o número de *pixels* de cada cor em uma imagem.

Um descritor de imagem D é definido como uma tupla (ϵ_D, δ_D) [21], onde:

- $\epsilon_D : \{\hat{I}\} \rightarrow \mathbb{R}^n$ é uma função que extrai o vetor de características $\vec{v}_{\hat{I}}$ da imagem \hat{I} ;
- $\delta_D : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ é uma função que calcula a similaridade entre duas imagens, inversamente proporcional à distância entre seus vetores de características.

A Figura 2.3 ilustra o uso de um descritor D para calcular a similaridade entre duas imagens \hat{I}_A e \hat{I}_B . Primeiramente, o algoritmo de extração ϵ_D é utilizado para extrair os vetores de características $\vec{v}_{\hat{I}_A}$ e $\vec{v}_{\hat{I}_B}$ associados às imagens. Por último, a função de similaridade δ_D é utilizada para determinar a similaridade d entre as imagens [21].

Para facilitar o processo de indexação, os vetores de características geralmente são definidos em espaços métricos. Neste caso, a distância calculada $d(\vec{v}_1, \vec{v}_2)$ entre dois vetores quaisquer \vec{v}_1 e \vec{v}_2 deve obedecer os seguintes critérios:

- Positividade: $d(\vec{v}_1, \vec{v}_2) \geq 0$;
- Auto-similaridade: $d(\vec{v}_1, \vec{v}_1) = 0$;
- Simetria: $d(\vec{v}_1, \vec{v}_2) = d(\vec{v}_2, \vec{v}_1)$;
- Desigualdade triangular: $d(\vec{v}_1, \vec{v}_2) + d(\vec{v}_2, \vec{v}_3) \geq d(\vec{v}_1, \vec{v}_3)$, para um terceiro vetor \vec{v}_3 .

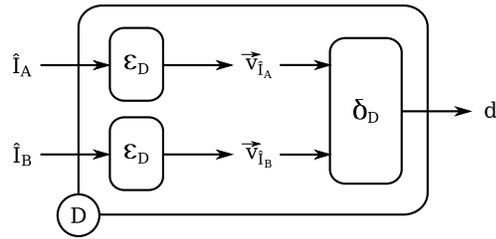


Figura 2.3: Uso de um descritor D para calcular a distância entre as imagens \hat{I}_A e \hat{I}_B .

São exemplos de métricas a distância Euclidiana [25] e o algoritmo de casamento de subsequências OCS – *Optimal Correspondent Subsequence* [72].

Descritores de imagem devem possuir propriedades que garantam sua efetividade quando usados na indexação e recuperação de imagens. A mais importante é a caracterização única de um determinado objeto [44] a partir do vetor de características, facilitando a distinção entre imagens visualmente diferentes, de acordo com alguma métrica. A segunda propriedade mais importante é a invariância do descritor a algumas classes de transformações, como a rotação e a translação [51]. Outras propriedades desejáveis são: insensibilidade a ruído, geração de vetores de características compactos (que requerem pouco espaço de armazenamento) e função de extração computacionalmente eficiente.

Nas próximas seções, será apresentada revisão bibliográfica de descritores de cor e textura. Como a principal contribuição deste trabalho é a proposta de descritores de forma, um estudo mais aprofundado desta classe será feito separadamente na Seção 2.4.

2.3.1 Descritores de cor

A cor é um poderoso descritor que geralmente simplifica a identificação de objetos e a sua posterior extração das cenas. Isso se deve ao fato de que seres humanos podem discernir milhares de tons e intensidades de cores, enquanto apenas duas dúzias de tons de cinza [34].

O descritor de cor mais comum é o histograma, que descreve o conteúdo global de uma imagem pelo percentual de *pixel* de cada cor [66]. Um histograma, para cada valor de cor presente na imagem, informa a probabilidade de um *pixel* da imagem possuir esta cor. A partir da representação do conteúdo da imagem por meio do histograma, podem-se utilizar as distâncias L_1 (*City-block*), L_2 (Euclidiana) ou L_∞ (*Chessboard*), por exemplo, para a comparação entre diferentes histogramas.

Em [65], é proposto um método de recuperação por conteúdo baseado em cor, chamado BIC (*Border/interior pixel classification*). Trata-se de método compacto e eficiente baseado na classificação dos *pixels* como sendo de borda ou de interior do objeto. Possui três componentes

principais: algoritmo de segmentação de imagens que classifica *pixels* como sendo da borda ou interior; uma representação compacta das propriedades visuais extraídas das imagens através de seus histogramas de cor nas regiões e nas bordas; e uma função de distância logarítmica para comparação de histogramas.

Em [20], é proposto um descritor de cor, chamado Coesão, baseado nas estatísticas relativas à distribuição das regiões coloridas na imagem. Este descritor utiliza informação cromática em conjunto com a distribuição espacial dos objetos na imagem. O descritor é avaliado com diferentes distâncias: L_1 , L_2 e L_∞ .

O projeto *Multimedia Content Description Interface* [2], ou MPEG, é um padrão para descrever dados de conteúdo multimídia, permitindo a interpretação da significância da informação, para ser passada ou acessada por um dispositivo ou código computacional. Dentre outros padrões, o MPEG-7 define quatro descritores de cor [47]:

- *Scalable color (SCD)*: é definido como um histograma no espaço de cor HSV (*hue*, *saturation* e *color*), com quantização fixa e uniforme neste espaço. Os histogramas são codificados utilizando a transformada de *Haar*, que facilita uma representação escalável. O SCD utiliza a métrica L_1 ou distância de *Hamming* para a comparação dos histogramas.
- *Color Structure (CSD)*: definido no espaço HMMD (*hue-min-max-difference*), objetiva identificar distribuições de cores localizadas na imagem, utilizando uma pequena janela estruturante. O vetor de características é formado por um histograma, cujo *bin* m representa o número de localidades nas quais um *pixel* de cor c_m se encontra dentro da janela estruturante. Qualquer distância (L_1 , L_2 ou L_∞) pode ser utilizada para computar a dissimilaridade entre os histogramas.
- *Dominant color (DCD)*: provê a distribuição das cores salientes em uma imagem, representando as cores presentes em uma região de interesse. O vetor de características é formado pelas cores representativas, suas porcentagens na região, coerência espacial das cores dominantes e a variância de cor para cada dominante. Uma medida de similaridade baseada na distância quadrática para histogramas de cores é proposta para este descritor.
- *Color Layout (CLD)*: captura o *layout* espacial das cores dominantes em uma grade sobreposta na região de interesse. Para isso, utiliza cores representativas em uma grade 8x8 e aplica a Transformada de Cosseno Discreta (DCT). O vetor de características é formado pelos coeficientes obtidos pela transformada. A função de distância está detalhada em [47].

Uma explanação mais detalhada e técnica destes e outros descritores de cor, bem como testes comparativos, podem ser vistos em [14] e [47].

2.3.2 Descritores de textura

A textura é uma medida do arranjo estrutural dos *pixels* em uma imagem. Embora não exista nenhuma definição formal para textura, os descritores de textura podem medir algumas propriedades como regularidade, orientação (direção), suavidade, granularidade, entre outras [34]. Assim como cor e forma, a textura é um poderoso descritor de baixo nível.

O projeto MPEG-7 propõe três principais classes de descritores de textura:

- *Texture Browsing* [75]: caracteriza atributos perceptivos em termos de direção, regularidade e granularidade da textura. A computação deste descritor consiste, simplificada, em filtrar a imagem com um banco de filtros de escala e orientação (modelados a partir de funções de Gabor), obtendo projeções nas orientações dominantes, que podem ser analisadas quanto à regularidade.
- *Homogeneous Texture* [60]: provê uma caracterização quantitativa de regiões de texturas homogêneas por recuperação de similaridades. É baseado na computação da estatística da frequência espacial local da textura e seu cálculo se dá de maneira semelhante ao *Texture Browsing Descriptor*. Um exemplo de aplicação desta classe de descritor de textura seria a procura, em uma base de imagens de satélites, por estacionamentos de carro. Carros estacionados em intervalos regulares são um ótimo exemplo de textura homogênea, quando vistos a uma certa distância.
- *Local Edge Histogram* [57]: é usado quando a região subjacente não é homogênea nas propriedades de sua textura. Este descritor representa a distribuição espacial de cinco tipos de borda (quatro direcionais e uma não-direcional). Como as bordas são importantes na percepção da imagem, este descritor é usado para recuperação de imagens com significado semântico similar.

Outros exemplos de descritores de textura podem ser encontrados em [26].

2.4 Descritores de forma

Para a aplicação de descritores de forma, muitas vezes as imagens devem passar previamente pelos passos de segmentação e representação. Entende-se por segmentação o processo de extração, ou distinção, de objetos relevantes em uma imagem. Como diferentes objetos requerem diferentes algoritmos de segmentação, torna-se complexa a tarefa de descrição de imagens por meio da forma. Por este motivo, geralmente as propostas de descritores não contemplam esta fase de segmentação, já supondo que as imagens foram segmentadas em um processo anterior. Esta estratégia também é utilizada neste trabalho.

Embora algumas vezes os dados obtidos na segmentação sejam utilizados diretamente para geração dos vetores de características, uma prática comum é a utilização de esquemas que compactem os dados em representações que são consideradas mais úteis na computação dos vetores [34]. Um exemplo é a utilização de *esqueletos*, que podem ser definidos como os lugares geométricos dos centros de todos os discos máximos de um objeto [43]. Muitos descritores já incluem a fase de representação em suas funções de extração de características.

Após o processo de segmentação, os objetos de interesse na imagem se encontram separados do fundo e de outros objetos irrelevantes, podendo ser utilizados diretamente pelos descritores de forma. Os métodos de descrição de forma podem se basear em características efetivas e perceptivas da forma presentes no contorno destes objetos – métodos baseados em contorno – e/ou no conteúdo interno destes objetos – métodos baseados em região. Cada classe se subdivide em estrutural ou global, dependendo se a forma é representada como um todo ou é dividida em segmentos ou seções. Esta classificação pode ainda se desmembrar em domínio espacial e domínio transformado, baseando-se no fato das características de forma serem derivadas do domínio espacial ou de um domínio transformado [77]. A Figura 2.4, adaptada de [77], mostra esta classificação dos métodos de descrição de forma. Outro tipo de classificação é encontrado em [44].

Muitas técnicas de descrição de forma têm sido propostas e algumas delas são bastante relevantes ao presente trabalho:

- *Curvature Scale Space (CSS)*: O *Curvature Scale Space* [3] é um descritor aplicado ao contorno do objeto. Os contornos simplificados de um determinado objeto na imagem são obtidos por meio de sucessivas suavizações por uma função Gaussiana e cada estágio desta suavização do contorno representa um escala na curva (*scale-space*). Assim, o vetor do CSS representa uma organização multiescala dos pontos de curvatura zero no contorno. Neste sentido, a dimensão do vetor varia para formas diferentes, requisitando um algoritmo de casamento especial para o cálculo de dois vetores CSS [3].
- *Beam Angle Statistics (BAS)*: O BAS [6] é um descritor baseado em *beams* que se originam em pontos do contorno. *Beams* são conjuntos de linhas que conectam um determinado ponto de referência com todos os outros pontos do contorno. Em cada ponto do contorno, o ângulo entre um par de *beams* é calculado e o vetor de características é definido usando-se estatísticas de primeira, segunda e terceira ordem aplicadas aos ângulos, em um conjunto de sistemas de vizinhança. Um algoritmo de correspondência ótima de subsequências (OCS) [72] é usado para o cálculo de similaridade entre dois vetores BAS.
- *Contour Saliences (CS)*: O descritor *Contour Saliences* [22] utiliza as saliências do contorno do objeto para a representação de sua forma. As saliências de um contorno são definidas como sendo os pontos de mais alta curvatura presentes no contorno de um objeto. No descritor CS, os valores das saliências são definidos como as áreas de influência

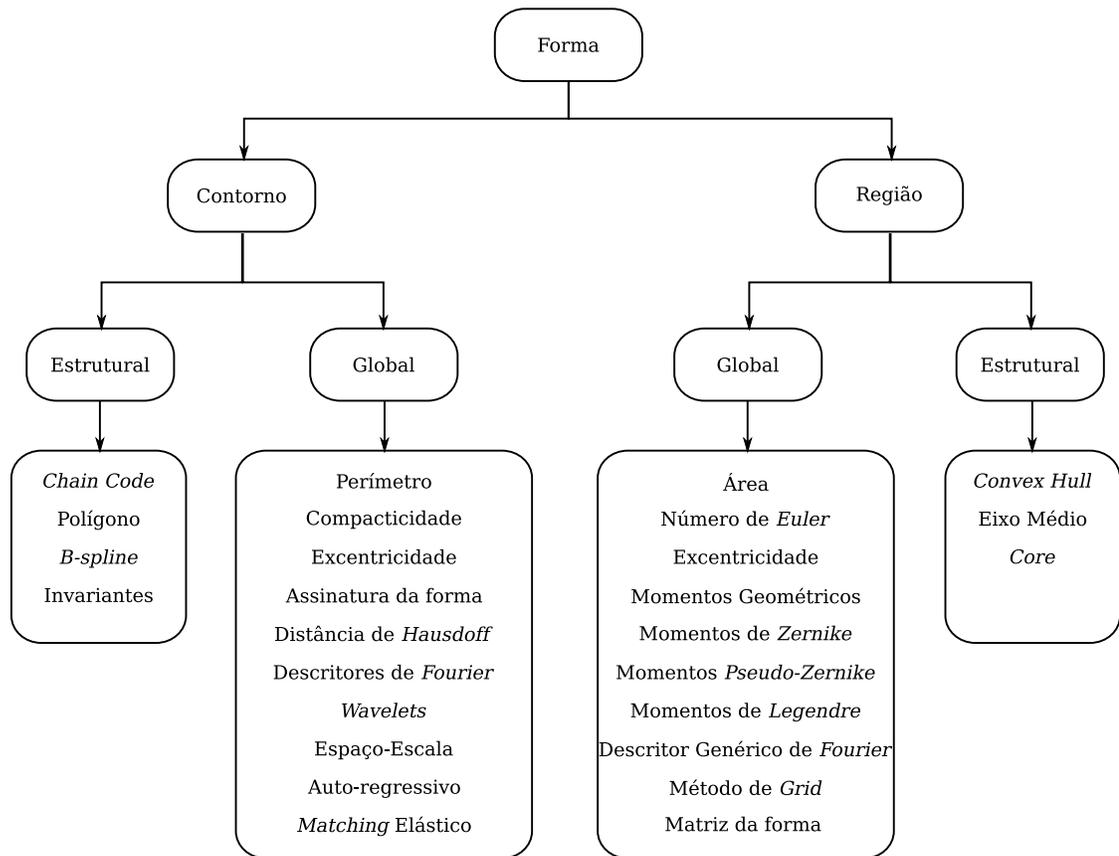


Figura 2.4: Classificação de técnicas de representação e descrição de forma. Figura adaptada de [77].

dos pontos de contorno de mais alta curvatura, considerando-se uma banda estreita em ambos os lados da curva e as regiões de *Voronoi* desses pontos. Um ponto do contorno é considerado convexo quando sua área de influência é maior dentro do que fora do contorno e, caso contrário, é considerado côncavo. Para o cálculo das saliências de contorno, o descritor utiliza as saliências dos esqueletos interno e externo da forma. O método determina os pontos de saliência e define um deles como referência para, na seqüência, computar a posição relativa dos demais pontos em função do ponto de referência, completando assim a representação do vetor de características. Como os vetores de objetos distintos podem possuir tamanhos diferentes, o CS utiliza um algoritmo heurístico de casamento entre os contornos, que registra os vetores em relação ao ponto de referência e computa a similaridade. Este algoritmo é baseado no proposto por Abassi e Mokhtarian para o descritor CSS [3] e é descrito na Seção 5.2 do Capítulo 5.

- *Segment Saliences (SS)*: O *Segment Saliences* [22] é uma variação do CS que incorpora duas melhorias: o valor de saliência de segmentos do contorno, no lugar de valores de saliência de pontos isolados; e um outro algoritmo que substitui o casamento heurístico por uma abordagem baseada no OCS [72]. Os valores de saliência são calculados e então o contorno é dividido em um número pré-definido de segmentos de mesmo tamanho. As áreas de influência internas e externas de cada segmento são computadas somando-se as áreas de influência de seus *pixels* correspondentes. Um segmento é considerado convexo se sua área acumulada externa é maior do que sua área acumulada interna e, caso contrário, é considerado côncavo. O número fixo de segmentos no contorno permite a utilização do algoritmo OCS para realizar o casamento dos vetores entre contornos.
- *Descritor de Fourier*: O descritor de Fourier de um contorno é formado por um vetor de características com os 126 coeficientes mais significativos de sua Transformada de Fourier, utilizando o método descrito em [34, 50]. A distância Euclidiana pode ser usada para medir a similaridade entre dois vetores do descritor de Fourier.
- *Invariantes de Momento*: Para o descritor Invariantes de Momento, cada objeto é representado por um vetor de características com 14 dimensões, incluindo dois conjuntos de invariantes de momento normalizados [37, 27], sendo um do contorno do objeto e outro da sua silhueta sólida. Neste descritor, também pode-se utilizar a distância Euclidiana para o cálculo da similaridade.
- *Dimensão Fractal Multiescala*: A dimensão fractal provê um meio de caracterizar auto-similaridade (ou auto-afinidade) de objetos reais ou abstratos. A dimensão fractal de Minkowski-Bouligand é definida como uma função $F = 2 - \lim_{r \rightarrow 0} \frac{\log(A(r))}{\log(r)}$, sendo $A(r)$ a área da forma dilatada com o raio r . Numericamente, pode ser estimada pela interpolação da curva logarítmica $A(r)$ em termos do raio de dilatação r , computando o coeficiente angular $A'(r)$ desta linha e considerando F como $F(r) = 2 - A'(r)$, como descrito em [23]. O vetor de características é formado por 50 amostras deste polinômio e a distância entre dois vetores é calculada por meio da distância Euclidiana.

O projeto MPEG-7 sugere a utilização de alguns descritores de forma. São eles [9]:

- *Descritor de forma 3-D (espectro da forma)*: é uma extensão do índice de forma [76] (*shape index*), usado anteriormente como uma medida local de formas 3-D, para malhas (*meshes*) 3-D. O vetor de características é definido como o histograma do índice de forma, computado para toda a superfície 3-D ou, para malhas 3-D, computado para cada vértice. Duas variáveis adicionais são usadas para formar o vetor: a primeira expressa a área de regiões de superfície planar em relação à área total da malha 3-D; e a segunda é a área relativa de todos os componentes poligonais cujo índice de forma não pôde ser estimado confiavelmente, com respeito à área total da malha 3-D.

- Descritor baseado em região (transformação angular radial): expressa a distribuição de *pixels* dentro de uma região de um objeto 3-D. Pode ser usado para descrever objetos complexos, consistindo de múltiplas regiões desconexas, bem como objetos com e sem buracos. É um descritor baseado em momentos e utiliza uma transformação angular radial 2-D complexa, definida em um disco unitário em coordenadas polares. O vetor é formado por 35 coeficientes extraídos desta transformação e quantizados em 4 *bits* por coeficiente.
- Descritor baseado em contorno: é baseado na representação CSS (*Curvature scale-space*) [3] do contorno. Como descrito anteriormente, o vetor consiste dos valores de excentricidade e circularidade dos contornos original e filtrado, o índice indicando o número de picos na imagem CSS, a altura do maior pico e a posição dos picos restantes.
- Descritor 2-D/3-D: pode ser utilizado na combinação de descritores 2-D para representar características visuais de um objeto 3-D, visualizado em diferentes ângulos. Qualquer descritor 2-D pode ser utilizado, como os baseados em contorno, região, cor ou textura. O descritor 2-D/3-D suporta então a integração da utilização de descritores na imagem 2-D, para descrever características do mesmo objeto em três dimensões.

2.5 Resumo

Primeiramente, foram apresentados alguns conceitos fundamentais para o entendimento deste trabalho, como a definição de imagem digital e outros elementos que a compõem.

A segunda seção mostrou a arquitetura simples de um sistema CBIR e detalhou alguns sistemas importantes existentes.

O capítulo também englobou a revisão bibliográfica de descritores de imagem, apresentando brevemente descritores de cor e textura e, mais extensivamente, descritores de forma.

Capítulo 3

Tensor Scale

Escala (*scale*) pode ser entendida como uma resolução ou, mais genericamente, um intervalo de resoluções necessárias para garantir uma representação eficaz e compacta de objetos [61]. A noção de escala é importante na determinação do compromisso (*trade-off*) entre suavização de ruídos e percepção/detecção de estruturas. Além disso, a escala é bastante útil na divisão de uma tarefa complexa de visão computacional ou processamento de imagens, por exemplo, em uma hierarquia de tarefas onde níveis mais altos lidam com estruturas maiores [61].

Witkin [74] e Koenderink [39] formularam matematicamente este conceito, na forma da teoria de espaço-escala. Representações discretas de espaço-escala têm sido utilizadas em diversas aplicações, como segmentação [70], *clustering* [42], classificação [45] e análise estrutural [31]. Embora representações de espaço-escala sejam de grande relevância, não é óbvio determinar [61]:

- como unificar as informações de imagens em diferentes escalas; e
- como identificar a escala ótima em cada ponto individual de uma imagem.

Estas questões geram bastante impacto em algumas tarefas de visão computacional e processamento de imagens. Por exemplo, o conhecimento de uma escala local permitiria a seleção ótima de vizinhança em diferentes processos, conduzindo à seleção de vizinhança pequena em regiões de muitos detalhes em uma imagem ou regiões próximas a contornos; e à vizinhanças maiores em interiores de objetos [61].

Este tipo de observação motivou diversos trabalhos e impulsionou estudos referentes à escala local. Utilizando o modelo de campo de força, Tabb e Ahuja [67] definiram a escala em cada localidade como a menor escala da função de peso Gaussiana na qual a força atrativa resultante se estabiliza. Elder e Zucker [28] descreveram escala local como a menor escala que provê uma medida de gradiente acima de um limiar estatístico confiável. *Wavelets* e funções elementares de Gabor (GEF) [10] provêm informação relacionada à escala local. As funções

GEF são filtros que podem ser otimamente localizados nos domínios do espaço e da frequência e, utilizando GEF assimétrica, é possível obter informação de orientação de padrões locais [61].

Entretanto, os métodos de escala local citados apenas inferem, estatisticamente, a frequência e orientação locais em uma região de tamanho fixo pré-determinado, não sendo óbvia a utilização destes métodos para determinar o tamanho, orientação e anisotropia de uma estrutura específica formada localmente por continuidade de homogeneidade. Informações morfométricas de escala para estruturas específicas são bastante úteis em várias aplicações como filtragem, detecção de borda, segmentação de objetos, registro, análise das propriedades de estruturas regionais, como orientação, espessura, forma, etc. Este tipo de escala é chamada de escala local morfométrica [61].

A noção de escala local morfométrica foi introduzida por Punam et al. [62] com a utilização de método de escala local baseado em um modelo esférico que foi aplicado à filtragem, registro e remoção de efeitos parciais de volume em renderização. O *Tensor Scale* [61] é uma evolução deste método, pois resolve a principal limitação do modelo precursor, que é a falta de informação de orientação e anisotropia das estruturas.

3.1 Conceitos de *Tensor Scale*

Tensor Scale, em qualquer ponto p da imagem, é a representação paramétrica da maior elipse centrada em p que está contida em uma mesma região homogênea, de acordo com um critério predeterminado (e.g., uma textura, uma diferença entre os brilhos dos *pixels*, etc). A elipse centrada na origem é unicamente definida por três fatores:

$$\text{Orientação}(p) = \theta; \quad (3.1)$$

$$\text{Anisotropia}(p) = \sqrt{1 - \frac{|t_2(p)|^2}{|t_1(p)|^2}}; \quad (3.2)$$

$$\text{Espessura}(p) = |t_2(p)|; \quad (3.3)$$

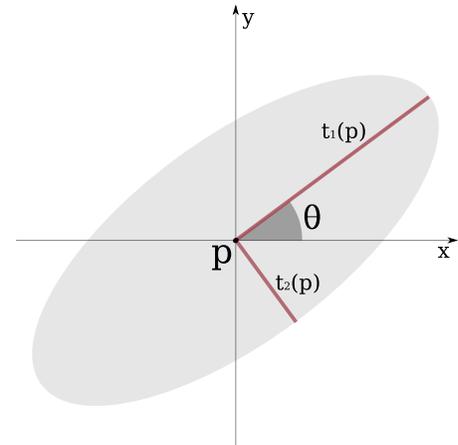


Figura 3.1: Fatores do *Tensor Scale*.

onde $|t_1(p)|$ e $|t_2(p)|$, sendo $|t_1(p)| \geq |t_2(p)|$, denotam o comprimento dos dois semi-eixos da elipse centrada em p ; e θ é o ângulo entre $|t_1(p)|$ e o eixo horizontal.

Na Figura 3.1, pode-se visualizar um exemplo dos três fatores do *Tensor Scale*. A elipse centrada em p possui a orientação representada pelo ângulo θ entre o eixo-x e o maior eixo da elipse; anisotropia, calculada a partir dos comprimentos dos semi-eixos $t_1(p)$ e $t_2(p)$; e a espessura, representada pelo semi-eixo menor $t_2(p)$.

Para efeito de visualização, os fatores *Tensor Scale* podem ser representados em uma imagem 2D, utilizando o sistema de cores HSI (*Hue, Saturation e Intensity*). É adotada a matiz (*hue*) para a representação da orientação, a saturação (*saturation*) para a representação da anisotropia e a intensidade (*intensity*) para a espessura normalizada. A Figura 3.2 mostra um exemplo de visualização do *Tensor Scale* no espaço HSI (Figura 3.2(b)) para a imagem da Figura 3.2(a). O círculo de codificação HSI representa a matiz para cada orientação possível que uma elipse pode assumir.

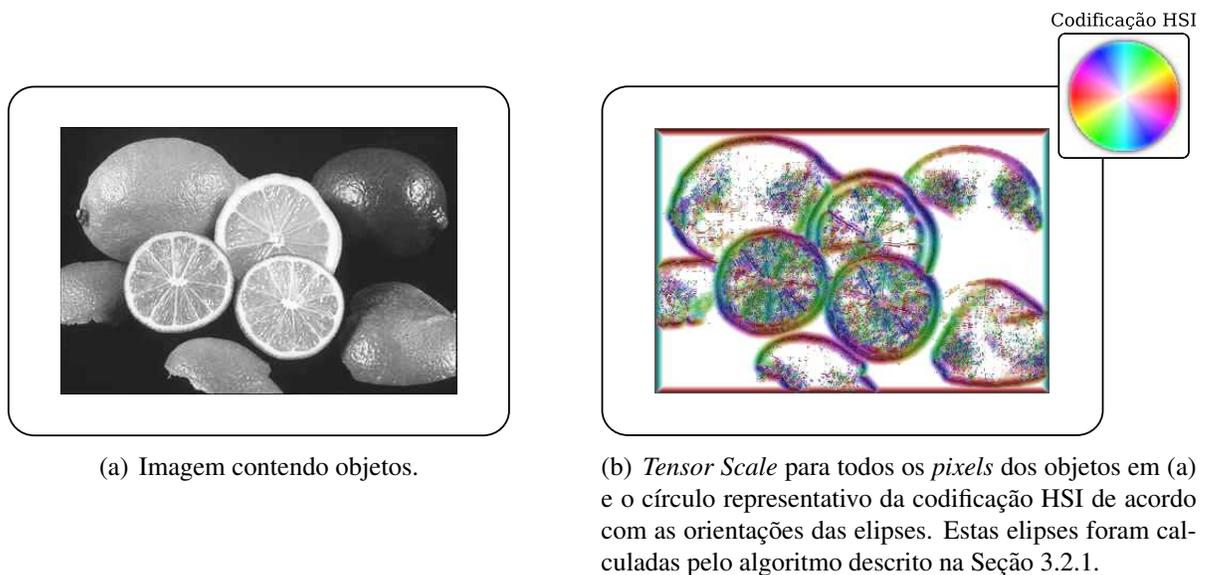


Figura 3.2: Exemplo de visualização do *Tensor Scale* no espaço HSI.

No círculo representativo da codificação HSI, ilustrado na Figura 3.2(b), nota-se que elipses com orientação no intervalo $[180^\circ, 360^\circ]$ assumem a mesma matiz que elipses no intervalo $[0^\circ, 180^\circ]$. Sendo assim, pode-se reduzir todas as orientações do intervalo $[180^\circ, 360^\circ]$ para $[0^\circ, 180^\circ]$. Ou seja, se a orientação da elipse A (θ_A) for maior que 180° , o valor $\theta_A - 180^\circ$ é adotado. Esta simplificação pode ser realizada sem perda de informação, pois as orientações de duas elipses A e B , sendo $\theta_A = 45^\circ$ e $\theta_B = 225^\circ$, por exemplo, são idênticas e, portanto, devem ser representadas com a mesma matiz.

A Figura 3.3 apresenta alguns exemplos de elipses *Tensor Scale* calculadas para algumas formas básicas. Para cada forma (i), pode-se observar as elipses calculadas (ii) para todos os pontos do objeto, codificadas no sistema HSI; e exemplos de elipses (iii) para alguns pontos da forma.

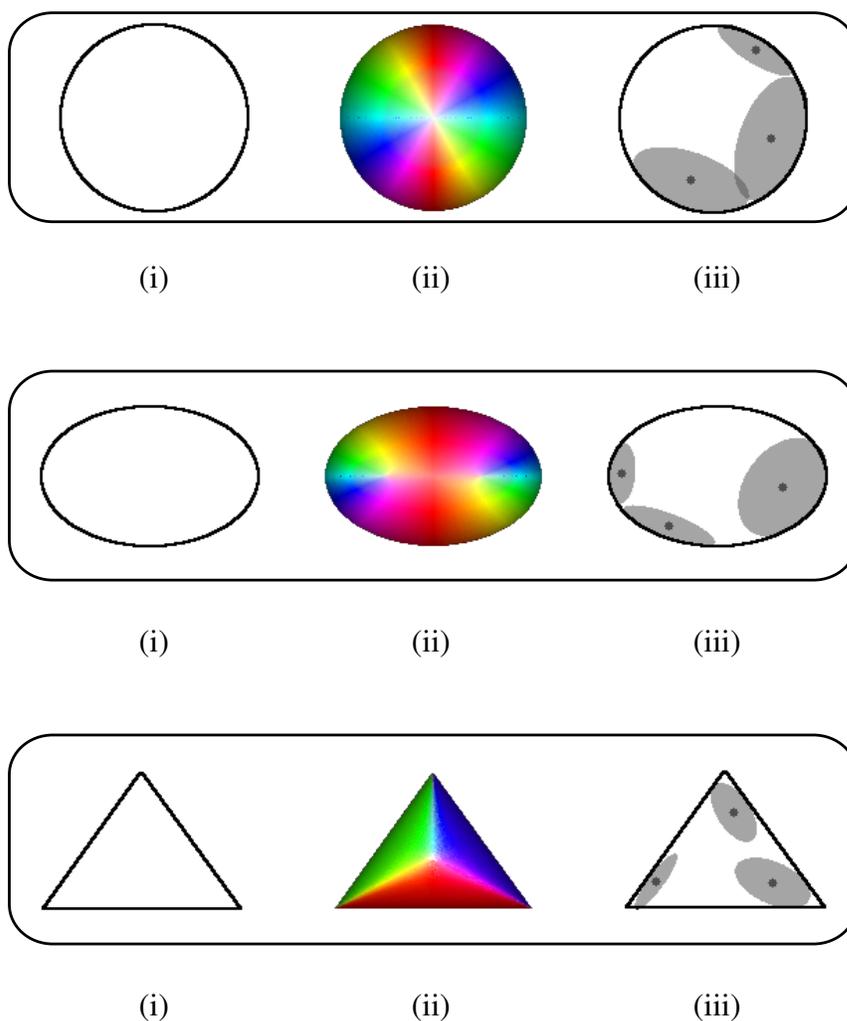


Figura 3.3: Exemplo de elipses *Tensor Scale* para formas básicas.

3.2 Algoritmos para cálculo do *Tensor Scale*

Esta seção apresenta os algoritmos já propostos para o cálculo do *Tensor Scale*, bem como o estudo de novos algoritmos.

3.2.1 Algoritmo de Punam

O primeiro algoritmo para o cálculo do *Tensor Scale* foi proposto por Punam [61] e, para seu projeto, foram considerados três aspectos do método:

- Continuidade da homogeneidade da região: em imagens reais, muitas vezes duas estruturas locais desconexas com intensidades similares, representando diferentes partes de um mesmo objeto ou de objetos diferentes, ficam próximas umas das outras. É desejável que a escala local não seja afetada por este tipo de evento.
- Representação paramétrica simples: uma escala está associada a cada *pixel* da imagem e uma representação paramétrica simples e intuitiva é essencial para seu uso eficiente.
- Modelo rico: juntamente com a simplicidade, é importante que o modelo seja rico o suficiente para poder representar as propriedades locais. *Tensor Scale* obedece este modelo, pois provê uma representação unificada de orientação, anisotropia e espessura das estruturas locais.

A elipse do *Tensor Scale* é calculada a partir de linhas de amostragem (*sample lines*) traçadas a partir de um *pixel*, de 0° a 179° (Figura 3.4(a)). Os eixos das elipses são determinados pela computação de intensidade, ao longo de cada linha de amostragem, e a localização de dois pontos ótimos de contorno nessas linhas (Figura 3.4(b)). O próximo passo é o reposicionamento dos pontos de contorno para pontos equidistantes do dado *pixel*, seguindo a simetria axial da elipse (Figura 3.4(c)). A computação da elipse mais próxima dos pontos encontrados é feita por *Principal Component Analysis* (PCA) [36] (Figura 3.4(d)).

Observa-se na Figura 3.4(d) que a elipse não está totalmente contida no objeto. O cálculo do *Tensor Scale* permite que isto ocorra, sendo necessário acrescentar passos no algoritmo para impedir este tipo de situação. Porém, o aumento na complexidade não justifica a pouca informação que a total inclusão da elipse traria. Mesmo utilizando elipses que não estão totalmente contidas no objeto, o *Tensor Scale* pode ser utilizado para descrição de formas.

Cada passo será detalhado a seguir, assim como foi originalmente proposto em [61].

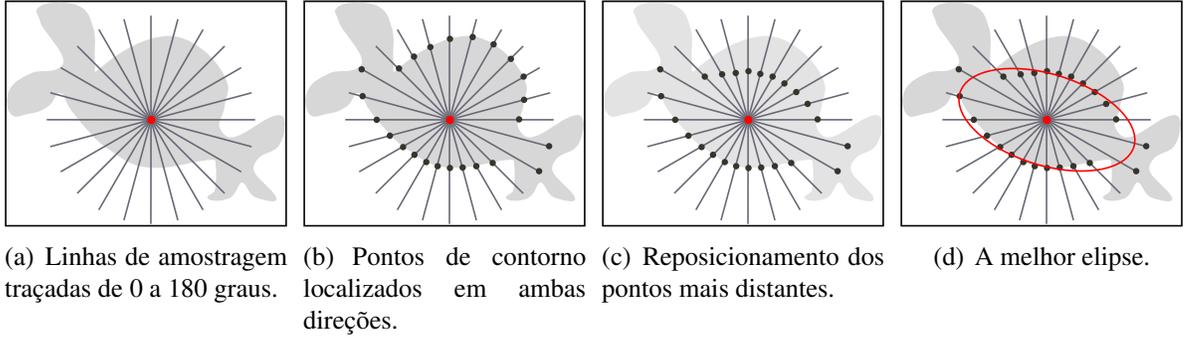


Figura 3.4: Etapas do algoritmo de Punam para o cálculo do *Tensor Scale*.

Computação de intensidade ao longo de cada linha de amostragem

Sejam $(\vec{\tau}_1, \vec{\tau}'_1), (\vec{\tau}_2, \vec{\tau}'_2), \dots, (\vec{\tau}_m, \vec{\tau}'_m)$, m pares de vetores unitários mutuamente opostos, com distribuição espacial uniforme em todo o espaço angular em torno da origem para garantir invariância à rotação. Uma linha de amostragem saindo do *pixel* p na direção do vetor $\vec{\tau}_i$ é denotada por $\pi_i^{(p)} : [0, L] \rightarrow \mathbb{R}^2$ e seu segmento oposto com direção $\vec{\tau}'_i$ é denotado por $\pi_i'^{(p)}$.

$$\pi_i^{(p)}(\lambda) = p + \lambda \vec{\tau}_i \quad | \quad \lambda \in [0, L]. \quad (3.4)$$

A Equação 3.4 representa uma linha de amostragem, onde L denota o comprimento de cada linha e a variável λ representa a distância entre o ponto $\pi_i^{(p)}(\lambda)$ e o ponto p . O comprimento L das linhas de amostragem é informado como parâmetro de entrada, representando a maior escala local, ou seja, este parâmetro limita a extensão de localidade.

O perfil (*profile*) de intensidade $f_i^{(p)}(\lambda)$ é computado em cada segmento, interpolando-se as intensidades da imagem em pontos selecionados em intervalos regulares. Perfis de intensidade são utilizados para armazenar valores de intensidade ao longo de segmentos de linha.

Localização de borda em uma linha de amostragem

A localização da borda em cada segmento é baseada no máximo $\mu_{\pi_i^{(p)}}^{UP}$ e no mínimo $\mu_{\pi_i^{(p)}}^{DN}$ do perfil de intensidades ao longo do segmento.

$$\mu_{\pi_i^{(p)}}^{UP}(\lambda) = \max_{x=1,2,\dots,\lambda} f_i^{(p)}(x), \quad (3.5)$$

$$\mu_{\pi_i^{(p)}}^{DN}(\lambda) = \min_{x=1,2,\dots,\lambda} f_i^{(p)}(x). \quad (3.6)$$

O principal objetivo é tratar separadamente os casos onde ocorrem *step-up* (Equação 3.5) e *step-down* (Equação 3.6) na borda, filtrando estruturas desconectadas localmente mas que tenham intensidade similar. *Step-down* e *step-up* são a diminuição e o aumento, respectivamente, no valor de intensidade ao longo da linha de amostragem.

Um detector de bordas convencional é então aplicado para localizar um ponto de borda no máximo e no mínimo do perfil de intensidades. Punam adotou a função *LoG*, proposta em [49], para computar a borda ótima em $\mu_{\pi_i^{(p)}}^{UP}$ e $\mu_{\pi_i^{(p)}}^{DN}$. A localização da borda é dada pelo primeiro ponto de curvatura zero, a partir de p , onde o gradiente de intensidade esteja acima de um determinado limiar. O gradiente de intensidade é computado pela convolução do mínimo e máximo dos perfis com a primeira derivada de uma Gaussiana.

Reposicionamento dos pontos de borda

Sejam $\epsilon_1^{(p)}, \epsilon_1'^{(p)}, \epsilon_2^{(p)}, \epsilon_2'^{(p)}, \dots, \epsilon_m^{(p)}, \epsilon_m'^{(p)}$ os pontos ótimos de borda nas linhas de amostragem $\pi_1^{(p)}, \pi_1'^{(p)}, \pi_2^{(p)}, \pi_2'^{(p)}, \dots, \pi_m^{(p)}, \pi_m'^{(p)}$, respectivamente, utilizando-se o método descrito anteriormente. Seguindo a simetria axial de uma elipse, dois pontos radialmente opostos no contorno de uma elipse são sempre equidistantes do centro. Por isso, os pontos de borda $\epsilon_i^{(p)}$ e $\epsilon_i'^{(p)}$ devem ser equidistantes de p para $i = 1, 2, \dots, m$. Porém, o método de localização não garante esta simetria axial, sendo necessária uma fase de reposicionamento dos pontos de borda calculados.

Esta etapa é realizada pela análise dos pontos de borda em cada par conjugado de linhas de amostragem. O ponto mais perto de p entre $\epsilon_i^{(p)}$ e $\epsilon_i'^{(p)}$ é escolhido e refletido em sua linha de amostragem complementar.

Computação da melhor elipse

A computação da melhor elipse (*best-fit*) é efetuada em dois sub-passos:

1. Determinação da orientação da elipse;
2. Cálculo dos comprimentos dos semi-eixos a e b .

Punam utiliza método baseado em *Principal Component Analysis* (PCA) [36] para implementar o primeiro sub-passo. A orientação θ de uma elipse é dada pelo menor ângulo entre o auto-vetor associado ao maior auto-valor da matriz de co-variância Σ e o eixo horizontal, onde os elementos $\Sigma_{i,j}$ são assim determinados:

$$\Sigma_{1,1} = \frac{1}{2m} \sum_{i=1,2,\dots,2m} (x_i - x_p)^2, \quad (3.7)$$

$$\Sigma_{2,2} = \frac{1}{2m} \sum_{i=1,2,\dots,2m} (y_i - y_p)^2, \quad (3.8)$$

$$\Sigma_{1,2} = \Sigma_{2,1} = \frac{1}{2m} \sum_{i=1,2,\dots,2m} (x_i - x_p)(y_i - y_p), \quad (3.9)$$

onde (x_i, y_i) , com $i = 1, 2, \dots, 2m$, são as coordenadas dos pontos de borda depois da fase de reposicionamento; e (x_p, y_p) é a coordenada do *pixel* p .

O segundo sub-passo é realizado pela minimização da função de erro (Equação 3.10).

$$f_{err} = \sum_{i=1,2,\dots,2m} \left[1 - \frac{u_i^2}{a^2} - \frac{v_i^2}{b^2} \right]^2, \quad (3.10)$$

onde (u_i, v_i) , com $i = 1, 2, \dots, 2m$ são as coordenadas relativas dos pontos de borda reposicionados em relação ao ponto p e depois da rotação pelo ângulo $-\theta$, tal que o maior semi-eixo das elipses seja alinhado com o eixo horizontal. Atribuindo-se zero às derivadas parciais de f_{err} em relação a a e b , pode-se provar que os valores a e b que minimizam f_{err} são determinados pelas seguintes expressões:

$$a = \sqrt{\frac{\gamma * \gamma - \delta * \zeta}{\gamma * \beta - \alpha * \zeta}}, \quad (3.11)$$

$$b = \sqrt{\frac{\gamma * \gamma - \delta * \zeta}{\alpha * \gamma - \delta * \beta}}, \quad (3.12)$$

onde α , β , γ e ζ correspondem a:

$$\alpha = \sum_{i=1,2,\dots,2m} u_i^2, \quad (3.13)$$

$$\beta = \sum_{i=1,2,\dots,2m} v_i^2, \quad (3.14)$$

$$\gamma = \sum_{i=1,2,\dots,2m} u_i^2 v_i^2, \quad (3.15)$$

$$\delta = \sum_{i=1,2,\dots,2m} u_i^4, \quad (3.16)$$

$$\zeta = \sum_{i=1,2,\dots,2m} u_i^4. \quad (3.17)$$

A Figura 3.2(b) apresenta o resultado do cálculo do *Tensor Scale* para a Figura 3.2(a), utilizando-se a codificação para o espaço de cor HSI, como foi explicado anteriormente.

Como estes cálculos são realizadas para todo *pixel* da imagem e o algoritmo proposto em [61] para calcular o *Tensor Scale* é computacionalmente caro, sua utilização em métodos que auxiliam sistemas CBIR é quase proibitiva. Por esta razão, Miranda et al. [52] propuseram uma implementação mais simples e efetiva do método original, que é descrita na próxima seção.

3.2.2 Algoritmo de Miranda et al.

O fato do algoritmo proposto por Punam ser computacionalmente caro é a principal motivação do trabalho de Miranda et al. [52], que propõe uma implementação do *Tensor Scale* efetiva e bem mais simples. Ao invés da utilização do esquema complexo de detecção de borda, foi proposto outro critério para definir a região homogênea em torno de um determinado *pixel*. O método não necessita da fase de reposicionamento e também possui uma fase menos complexa de computação da elipse.

A primeira mudança proposta é na fase de localização dos pontos de contorno, na qual o método adotado é a computação da intensidade ao longo de segmentos opostos, alternadamente, ao invés de realizar o cálculo em cada segmento inteiro de cada vez. Quando a localização da borda é encontrada em um segmento, o ponto de borda oposto já está na posição correta e, assim, a fase de reposicionamento não é mais necessária.

A segunda mudança é a utilização de dois limiares (*threshold*) conectados, th_1 e th_2 , para simplificar o método de detecção do contorno. O limiar th_1 é o máximo valor de diferença de intensidade absoluta dentro de regiões homogêneas da imagem; e o limiar th_2 é o máximo valor de diferença de intensidade absoluta acumulada dentro de regiões homogêneas. Ambos os limiares são calculados em relação ao *pixel* central p . O limiar th_1 tenta capturar as descontinuidades abruptas, enquanto th_2 objetiva terminar o processo de detecção de borda, no caso de transições de intensidade suaves.

Segmentos opostos são percorridos alternadamente, no mesmo laço do algoritmo. O valor de diferença de intensidade absoluta entre o ponto sendo visitado e p é calculado e comparado a th_1 . Se o valor for menor ou igual a th_1 , então é acumulado em uma variável cujo valor é comparado a th_2 . O processo termina quando o valor da diferença for maior que th_1 , ou seja, quando a transição for abrupta, ou quando o valor acumulado for maior que th_2 (transição suave). No caso de imagens binárias, esta localização é trivial. A detecção pára quando uma das linhas sendo percorridas cruzar o objeto em direção ao fundo.

Um exemplo da utilização dos limiares th_1 e th_2 pode ser visto na Figura 3.5. Analisando a variação de intensidade na linha \overline{PB} , percebe-se uma transição suave, mas com uma separação clara entre objeto e fundo que deve ser detectada. Para que isto ocorra, o valor acumulado das intensidades na linha deve ser maior que th_2 no ponto B . Já na linha \overline{PA} , a intensidade se mantém praticamente inalterada, até que uma transição abrupta ocorre em A . Neste ponto, a diferença absoluta entre A e P deve ser maior que th_1 .

A terceira e última mudança é uma melhoria na fase de computação da elipse. Miranda et al. propuseram a função $g(\gamma)$ para cálculo do ângulo diretamente, ao invés do uso de PCA [36].

$$g(\gamma) = \sum_{i=1,2,\dots,2m} [x_{i\gamma}^2 - y_{i\gamma}^2], \quad (3.18)$$

onde

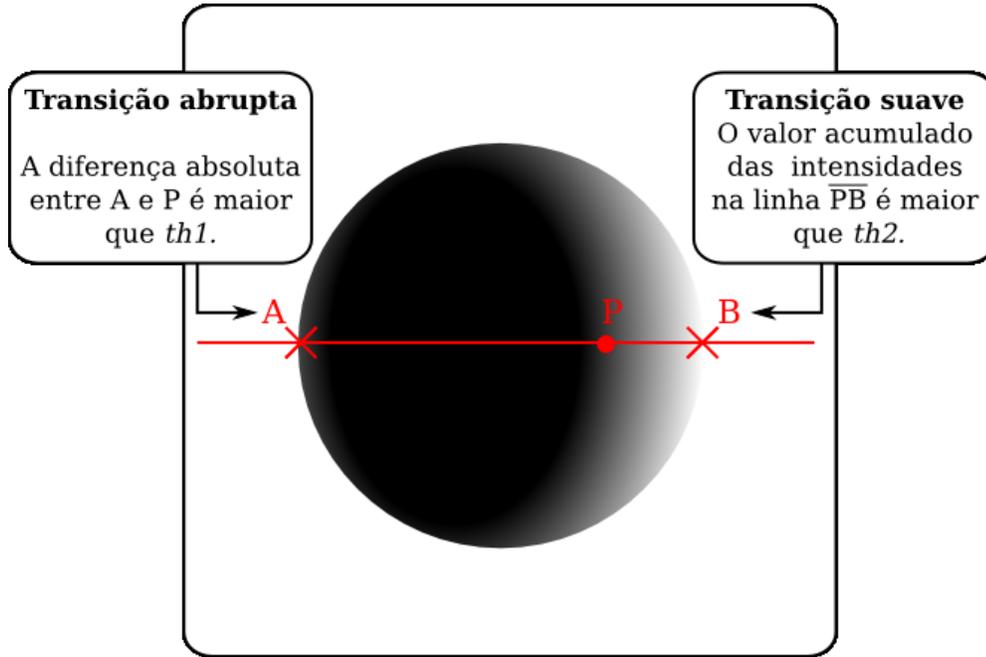


Figura 3.5: Exemplo de utilização dos limiares th_1 e th_2 .

$$x_{i_\gamma} = x_i \cos(\gamma) - y_i \sin(\gamma), \quad (3.19)$$

$$y_{i_\gamma} = x_i \sin(\gamma) + y_i \cos(\gamma), \quad (3.20)$$

(x_i, y_i) são as coordenadas relativas dos pontos de borda em relação ao ponto central $p = (x_p, y_p)$ da elipse; e $(x_{i_\gamma}, y_{i_\gamma})$ são as novas coordenadas depois de aplicada uma rotação pelo ângulo γ .

A orientação da elipse é obtida do valor de γ que minimiza a função g . Atribuindo-se zero à derivada de $g(\gamma)$ ($g'(\gamma) = 0$), pode-se mostrar que γ é dado pela seguinte expressão:

$$\gamma = \frac{\arctan \left[\frac{2 * \alpha}{\beta - \zeta} \right]}{2}, \quad (3.21)$$

onde

$$\alpha = \sum_{i=1,2,\dots,2m} x'_i y'_i, \quad (3.22)$$

$$\beta = \sum_{i=1,2,\dots,2m} y_i'^2, \quad (3.23)$$

$$\zeta = \sum_{i=1,2,\dots,2m} x_i'^2. \quad (3.24)$$

O fato da derivada de $g(\gamma)$ ser zero também implica que g é mínima e a diferença entre o máximo e o mínimo é de $\frac{\pi}{2}$. Este fato pode ser corrigido adicionando-se ou subtraindo-se $\frac{\pi}{2}$ de γ , dependendo dos valores de α e γ (se α e γ forem negativos, então adiciona-se $\frac{\pi}{2}$; caso contrário subtrai-se).

Com a realização destas melhorias, o *Tensor Scale* pode ser utilizado para descrever objetos, como mostrado em [52]. Neste trabalho, são utilizadas apenas imagens binárias para teste e, por isso, várias melhorias são realizadas para simplificar o cálculo do *Tensor Scale*, como mostra a próxima seção.

3.2.3 Algoritmo para cálculo do *Tensor Scale* via Transformada Imagem-Floresta

No algoritmo proposto por Miranda et al., a melhoria na fase de localização dos pontos de contorno é bastante útil para a utilização do *Tensor Scale* em imagens em níveis de cinza. Porém, como neste trabalho utilizam-se apenas imagens binárias nos testes realizados com os descritores, podem-se realizar outras melhorias na fase de localização.

A primeira mudança é simples e consiste apenas na eliminação dos limiares th_1 e th_2 . Estes limiares não são mais necessários, visto que as imagens de teste já se encontram segmentadas e, sendo assim, a fase de localização dos pontos de contorno é óbvia.

Além disso, podem-se utilizar técnicas para encontrar mais facilmente os pontos de borda (pontos que separam o objeto do fundo) nas direções das linhas de amostragem. Tais técnicas compreendem a utilização da Transformada de Distância Euclidiana, calculada a partir da Transformada Imagem-Floresta, como será mostrado a seguir.

Transformada Imagem-Floresta

A Transformada Imagem-Floresta, ou IFT (do inglês, *Image-Foresting Transform*), é uma metodologia de resolução de problemas de processamento de imagens, que consiste na redução de um problema ao cálculo de florestas de caminhos mínimos [30]. Nesta redução, a IFT trata a imagem digital $\hat{I} = (D_I, \vec{I})$ como se fosse um grafo $G = (V, E)$, no qual os vértices em V correspondem aos *pixels* em D_I e as arestas em E são definidas por uma relação de adjacência. A relação de adjacência A é uma relação binária entre *pixels* da imagem, e $A(p)$ representa o conjunto de *pixels* adjacentes ao *pixel* p de acordo com A [29].

No grafo definido por A , o caminho π é uma seqüência de *pixels* adjacentes $\langle p_1, p_2, \dots, p_n \rangle$, onde $(p_i, p_{i+1}) \in A$, $i = 1, 2, \dots, n - 1$. Para o cálculo de florestas de caminhos de custo mínimo, os caminhos definidos no grafo devem possuir um custo para serem percorridos e, para tanto, o grafo formado a partir de uma imagem deve ser ponderado. Associa-se então, a cada caminho no grafo, o custo para percorrê-lo calculado pela função de custo mínimo $pf(\pi)$. A IFT requer ainda a escolha de um conjunto de *pixels* $S = \{s_1, s_2, \dots, s_n\}$, cujos elementos, chamados de *pixels* sementes, correspondem aos pontos de início dos caminhos mínimos. Cada semente recebe um rótulo de identificação, que deverá ser propagado a todos os *pixels* que pertencem ao caminho que a semente inicia.

O problema se resume, então, em particionar a imagem I em regiões R_i , de tal forma que um *pixel* p pertença a R_i se a distância de s_i a p , seguindo o critério de pf , for menor que a distância de s_j a p , para todo $s_j \in S, s_j \neq s_i$. Ou seja, deseja-se obter uma partição ótima de I a partir das sementes em S , de acordo com o critério de distância definido por pf , sendo que cada região obtida corresponde à zona de influência da semente correspondente [17].

Sendo assim, o algoritmo da IFT (Algoritmo 3.1) requer cinco parâmetros: a imagem I de entrada, o conjunto S de sementes, a função de custo de caminho pf , a relação de adjacência A e uma função de rotulação λ que atribui a cada *pixel* semente um rótulo diferente. A saída da IFT é a anotação da imagem correspondente à floresta de caminhos mínimos calculada. Para cada *pixel* p , existe a anotação de rótulo, que indica a região a qual o *pixel* p pertence; o custo do caminho mínimo da raiz de sua árvore até p ; e o seu predecessor no caminho ótimo. O rótulo anotado do *pixel* p é o mesmo rótulo atribuído à semente de seu caminho pela função λ .

A prova da corretude do algoritmo da IFT (Algoritmo 3.1) pode ser encontrada em [17].

Algoritmo 3.1 Transformada Imagem-Floresta.

Entrada: Uma imagem I , um conjunto S de sementes, uma relação de adjacência A , uma função de rotulação λ e uma função de custo $pf(\pi_p)$, que calcula o custo de se percorrer o caminho π_p de uma semente até o *pixel* p .

Saída: Um mapa de custo, um mapa de predecessores e um mapa de rótulos.

Estruturas de dados auxiliares: Uma fila de prioridade Q , uma variável auxiliar $custo'$ e uma lista L de *pixels* que já foram processados.

```

for all  $p \in I$  do
   $custo[p] \leftarrow +\infty$ ;
   $rotulo[p] \leftarrow NIL$ ;
   $predecessor[p] \leftarrow NIL$ ;
end for
for all  $p \in S$  do
   $rotulo[p] \leftarrow \lambda(p)$ ;
   $custo[p] \leftarrow 0$ ;
  insira  $p$  em  $Q$ ;
end for
while  $Q$  não estiver vazia do
  remova de  $Q$  o pixel  $p$  tal que  $custo[p]$  seja mínimo;
  insira  $p$  em  $L$ ;
  for all  $q$  tal que  $q \in A(p)$  e  $q \notin L$  do
     $custo' \leftarrow pf(\pi_p. < (p, q) >)$ , representando o custo do caminho para se chegar a  $q$ 
    passando por  $p$ ;
    if  $custo' < custo[q]$  then
      if  $q \notin Q$  then
        insira  $q$  em  $Q$ ;
      else
        atualize a posição de  $q$  em  $Q$ ;
      end if
       $custo[q] \leftarrow custo'$ ;
       $predecessor[q] \leftarrow p$ ;
       $rotulo[q] \leftarrow rotulo[p]$ ;
    end if
  end for
end while

```

Transformada de Distância Euclidiana (TDE)

Transformadas de distâncias são operações que associam, a cada *pixel* em uma imagem binária, a distância dele ao *pixel* semente mais próximo, de acordo com alguma métrica como, por exemplo, a Euclidiana. A distância Euclidiana D_{euc} entre dois *pixels* $p = (x_p, y_p)$ e $q = (x_q, y_q)$ é definida como:

$$D_{euc}(p, q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}. \quad (3.25)$$

Para o cálculo da TDE através da IFT, deve-se representar a imagem binária de entrada como um grafo ponderado não direcionado, no qual os vértices correspondem aos *pixels* [17]. Além disso, a função de custo é tal que o custo do caminho da semente s para o *pixel* p na floresta é a distância Euclidiana entre s e p .

A computação desta transformada exige a relação Euclidiana A e a função de custo f_{euc} definida para qualquer caminho $\pi = \langle p_1, p_2, \dots, p_n \rangle$ no grafo como:

$$q \in A(p) \Rightarrow (x_q - x_p)^2 + (y_q - y_p)^2 \leq \rho^2, \quad (3.26)$$

$$f_{euc} = \begin{cases} (x_{p_n} - x_{p_1})^2 + (y_{p_n} - y_{p_1})^2, & \text{if } p_1 \in S, \\ +\infty, & \text{caso contrário,} \end{cases} \quad (3.27)$$

onde ρ é o raio de adjacência e (x_{p_i}, y_{p_i}) são as coordenadas (x, y) do *pixel* p_i da imagem. $\rho = 1$ define vizinhança-4 e $\rho = \sqrt{2}$ define vizinhança-8. Esta relação é simétrica e invariante à translação.

A IFT também pode retornar o mapa de raízes, que associa a cada *pixel* p da imagem o *pixel* semente s que inicia o caminho a qual pertence. Este mapa facilita o cálculo da TDE e foi incorporado ao Algoritmo 3.2.

O mapa de rótulos indicará a qual região cada *pixel* da imagem pertence, ou seja, qual o rótulo da semente que influencia a região na qual o *pixel* da imagem se encontra. O mapa de raízes associa o número do *pixel* semente ao invés do rótulo.

A vantagem da utilização da IFT para o cálculo da TDE é a obtenção do mapa de rótulos, calculado dinamicamente em tempo linear, juntamente com as outras informações relevantes, como o mapa de raízes.

Algoritmo 3.2 Transformada de Distância Euclidiana via IFT.

Entrada: Uma imagem binária I , um conjunto S de sementes, uma relação de adjacência Euclidiana A e uma função de rotulação λ .

Saída: Uma mapa de custo, um mapa de predecessores, um mapa de rótulos e um mapa de raízes.

Estruturas de dados auxiliares: Uma fila de prioridade Q e uma variável auxiliar $custo'$.

```

for all  $p \in I$  do
   $custo[p] \leftarrow +\infty$ ;
   $rotulo[p] \leftarrow NIL$ ;
   $predecessor[p] \leftarrow NIL$ ;
   $raiz[p] \leftarrow NIL$ ;
end for
for all  $p \in S$  do
   $custo[p] \leftarrow 0$ ;
   $rotulo[p] \leftarrow \lambda(p)$ ;
   $raiz[p] \leftarrow p$ ;
  insira  $p$  em  $Q$ ;
end for
while  $Q$  não estiver vazia do
  remova de  $Q$  o pixel  $p = (x_p, y_p)$  tal que  $custo[p]$  seja mínimo;
  for all  $q = (x_q, y_q)$  tal que  $q \in A(p)$  e  $custo[q] > custo[p]$  do
     $custo' \leftarrow \sqrt{(x_q - x_{raiz[p]})^2 + (y_q - y_{raiz[p]})^2}$ , onde  $raiz[p] = (x_{raiz[p]}, y_{raiz[p]})$  é o pixel raiz do pixel  $p$ ;
    if  $custo' < custo[q]$  then
      if  $custo[q] \neq +\infty$  then
        remova  $q$  de  $Q$ ;
      end if
       $custo[q] \leftarrow custo'$ ;
       $rotulo[q] \leftarrow rotulo[p]$ ;
       $predecessor[q] \leftarrow p$ ;
       $raiz[q] \leftarrow raiz[p]$ ;
      insira  $q$  em  $Q$ ;
    end if
  end for
end while

```

Cálculo do *Tensor Scale* para imagens binárias

O cálculo do *Tensor Scale* para imagens binárias é bem mais simples do que o cálculo para imagens em níveis de cinza. Isto é verdade porque uma das fases do cálculo do *Tensor Scale* é exatamente a de localização de borda nas direções das linhas de amostragem. Porém, nas imagens binárias têm-se os conceitos de fundo e objeto, o que torna óbvia a diferença entre o que deve e o que não deve fazer parte do contorno do objeto. Para introduzir essa facilidade no cálculo, pode-se utilizar a Transformada de Distância Euclidiana.

No cálculo da Transformada de Distância Euclidiana (Algoritmo 3.2), se os *pixels* do contorno de um objeto contido em uma imagem forem considerados como sementes, tem-se o caminho de custo mínimo de qualquer *pixel* da imagem para o *pixel* de contorno mais próximo, considerando-se a métrica Euclidiana. Então, no mapa de custo retornado pela TDE, $custo[p]$ armazena o custo do caminho do *pixel* p até o ponto de contorno mais próximo.

A utilização desta transformada justifica-se pelo seguinte fato: se tivermos a menor distância de p ao contorno, não há necessidade de procurar por pontos de borda dentro de um raio de tamanho $custo[p]$, já que o ponto de borda mais próximo está a uma distância de $custo[p]$ de p .

A Figura 3.6 ilustra uma situação em que a TDE é aplicada. A área cinza corresponde ao círculo formado pelo raio de tamanho $custo[p]$ em torno do ponto p . Esta área representa o espaço que não precisará ser percorrido nas linhas de amostragem, pois é garantido que nela não há pontos de borda.

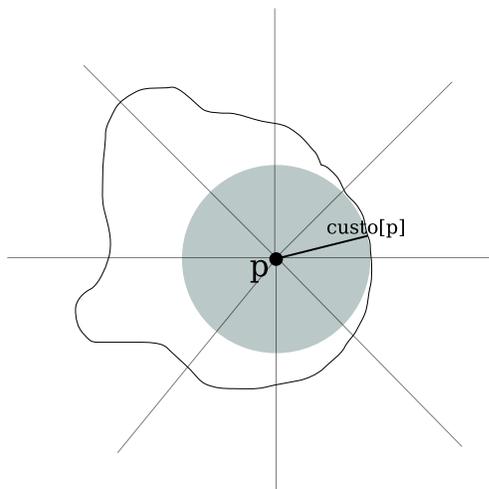


Figura 3.6: Primeiro passo da perseguição de borda.

Seguindo o algoritmo de Miranda et al., procura-se o ponto de borda ao longo de segmentos opostos, alternadamente. Porém, ao invés de realizar esta procura *pixel a pixel*, o algoritmo

realiza pulos ao longo das linhas, não passando exatamente nas áreas definidas pelos círculos de raio referente aos custos.

Na Figura 3.7, observa-se o segundo passo da situação ilustrada na Figura 3.6. Os pontos q e r são visitados ao mesmo tempo, pois estão em linhas de amostragem opostas. A partir deles, realiza-se novamente a procura de borda fora da área definida pelo custos $custo[q]$ e $custo[r]$.

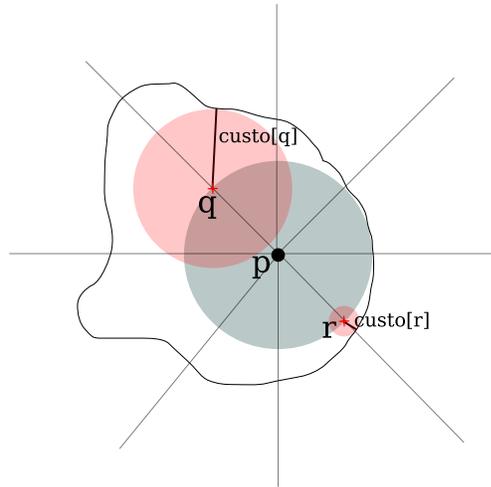


Figura 3.7: Segundo passo da perseguição de borda, após a situação ilustrada na Figura 3.6.

No exemplo, percebe-se que a borda foi encontrada no ponto $raiz[r]$, ou seja, no ponto de contorno mais próximo a r . Com isso, o algoritmo define que os dois pontos de borda nesta direção se encontram em (x, y) , que é a coordenada do ponto $raiz[r]$ em relação ao ponto p ; e (x', y') , que é a coordenada do ponto diametralmente oposto a (x, y) , em relação a p . Este passo pode ser observado na Figura 3.8.

Realizando-se este procedimento para todas as linhas de amostragem, o algoritmo define todos os pontos de borda nas direções das linhas. Com isso, pode-se utilizar a mesma fórmula definida por Miranda et al. (Equação 3.18) para o cálculo da orientação da elipse.

A perseguição de bordas da elipse centrada no *pixel* p está definida no Algoritmo 3.3. Neste algoritmo, qualquer função de rotulação pode ser aplicada desde que o conjunto S seja formado pelos *pixels* do contorno do objeto contido na imagem. No Capítulo 4, será mostrado que a função de rotulação independente é outra vantagem da utilização da IFT para o cálculo da TDE neste trabalho.

Algoritmo 3.3 Perseguição de bordas da elipse.

Entrada: Um *pixel* $p = (x_p, y_p)$, o número m de linhas de amostragem e o mapa *custo* retornado pelo Algoritmo 3.2.

Saída: O vetor *borda* que contém m pares de pontos de borda localizados nas linhas de amostragem.

```

for all  $\theta$  de  $0^\circ$  a  $179^\circ$ , com incrementos de  $\frac{180^\circ}{m}$  do
   $v \leftarrow custo[p]$ ;
   $p_1 \leftarrow NIL$ ;
   $p_2 \leftarrow NIL$ ;
   $q_1 \leftarrow 0$ ;
   $q_2 \leftarrow 0$ ;
  while  $p_1 \neq 0$  e  $p_2 \neq 0$  do
     $x \leftarrow v * \cos(\theta)$ ;
     $y \leftarrow v * \sin(\theta)$ ;
    if  $q_1 = 0$  then
       $temp \leftarrow (x_p + x, y_p + y)$ ;
       $p_1 \leftarrow custo[temp]$ ;
    end if
    if  $q_2 = 0$  then
       $temp \leftarrow (x_p - x, y_p - y)$ ;
       $p_2 \leftarrow custo[temp]$ ;
    end if
     $d \leftarrow \min(p_1, p_2)$ ;
     $v \leftarrow v + d$ ;
     $q_1 \leftarrow p_1 - d$ ;
     $q_2 \leftarrow p_2 - d$ ;
  end while
   $borda[\theta] \leftarrow ((x, y), (x', y'))$ , onde  $(x', y')$  é a coordenada do ponto diametralmente oposto a  $(x, y)$ , em relação a  $p$ ;
end for

```

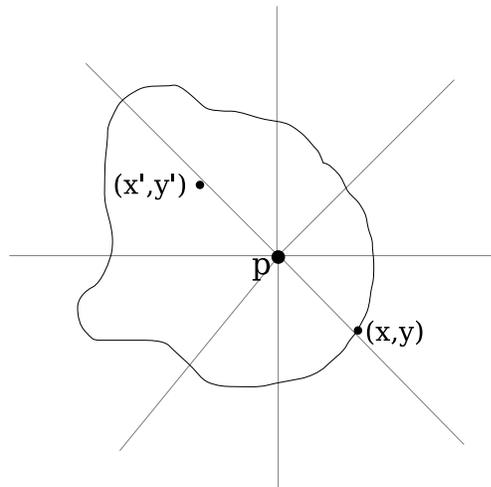


Figura 3.8: Terceiro passo da perseguição de borda, após a situação ilustrada na Figura 3.7.

3.3 Descritor *Tensor Scale* – TSD

O descritor TSD [52] baseia-se no fato de que objetos distintos geralmente apresentam diferentes distribuições das orientações do *Tensor Scale* em sua forma. A partir desta suposição, uma função de extração é proposta, tentando captar exatamente esta distribuição de orientações. A seguir, serão apresentadas as funções de extração do vetor de características e similaridade deste descritor.

3.3.1 Função de extração do vetor de características (ϵ_{TSD})

A primeira etapa é o cálculo dos fatores *Tensor Scale* para todos os *pixels* da imagem. Este cálculo é feito a partir da utilização do algoritmo apresentado na Seção 3.2.2. A Figura 3.9 ilustra um exemplo dessa primeira etapa da função de extração. O *Tensor Scale* é calculado para duas imagens: uma asa de mosca e a mesma imagem rotacionada.

A segunda etapa é a construção de histograma para as orientações calculadas. Este histograma possui 180 *bins*, representando as orientações quantizadas em intervalos de 1 grau. O histograma de orientações é computado apenas para elipses de alta anisotropia e com o valor de espessura dentro de um certo intervalo. Esta restrição é utilizada, pois elipses com baixa anisotropia ou muito espessas não têm uma orientação bem definida.

A Figura 3.10 mostra os histogramas das orientações do *Tensor Scale* representadas na Figura 3.9(b).

Este histograma deve ser analisado de forma circular, pois elipses que influenciaram os

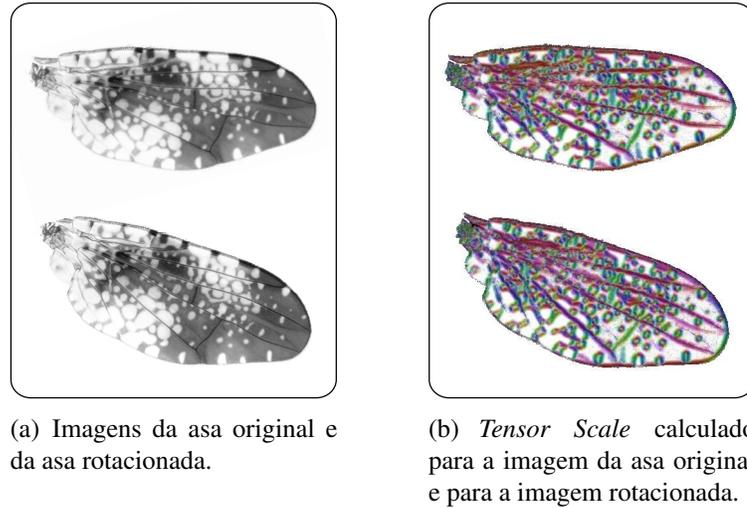


Figura 3.9: Exemplo da primeira etapa de ϵ_{TSD} para duas imagens.

bins próximos a 180° , por exemplo, têm orientação próxima às orientações das elipses que influenciaram os *bins* próximos a 0° .

Analisando o histograma de forma circular, percebe-se que o efeito de uma rotação θ na imagem original sob o histograma é apenas o deslocamento de θ no histograma da imagem rotacionada em relação ao histograma da imagem original.

3.3.2 Função de similaridade (δ_{TSD})

O casamento dos histogramas de orientação de duas imagens é realizado em duas etapas: a primeira corrige, por correlação, o deslocamento entre os histogramas; e a segunda calcula a diferença absoluta de área entre eles. A correlação que resulta na mínima diferença entre os dois histogramas representa o deslocamento entre eles e, conseqüentemente, a rotação entre as imagens. Sendo assim, este processo é invariante à rotação. Para torná-lo insensível à escala, os histogramas devem ser normalizados antes da correlação.

O casamento dos histogramas presentes na Figura 3.10 está representado na Figura 3.11.

Este deslocamento calculado pela correlação, quando convertido para graus, pode ser entendido como a diferença de angulação entre as imagens. Por este motivo, pode também ser usado para registro de imagens.

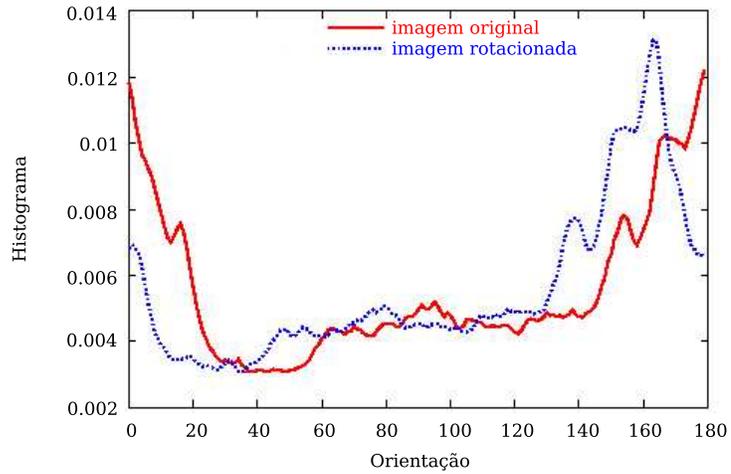


Figura 3.10: Exemplo da segunda etapa de ϵ_{TSD} para as duas imagens da Figura 3.9(a).

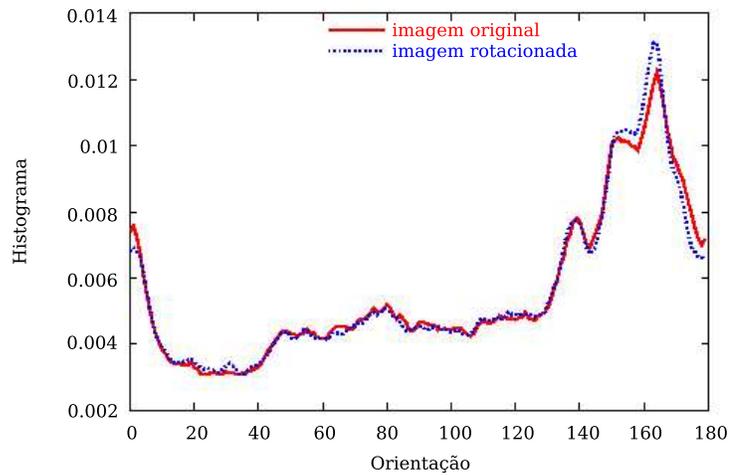


Figura 3.11: Exemplo da função δ_{TSD} para as duas imagens da Figura 3.9(a).

3.4 Resumo

Neste capítulo foram apresentados os conceitos relativos a *Tensor Scale*, bem como diversos algoritmos que podem ser utilizados para seu cálculo. Por último, o descritor baseado em *Tensor Scale* (*Tensor Scale Descriptor* – TSD), previamente proposto em [52], foi descrito para embasar o trabalho que será apresentado nos dois próximos capítulos.

Capítulo 4

Descritor de forma *Tensor Scale* por zonas de influência

Neste capítulo, propõe-se um descritor baseado em região e em contorno, que pode ser utilizado na recuperação de imagens de domínio geral. Este descritor, inspirado no descritor *Tensor Scale* (TSD) [52], introduz um novo método para explorar a orientação das elipses *Tensor Scale*, incluindo informação espacial.

Por utilizar, em conjunto, as informações das elipses em uma mesma zona de influência dentro da forma, este descritor é nomeado Descritor de forma baseado em *Tensor Scale* por zonas de influência, ou simplesmente descritor *Tensor Scale* por zonas de influência (*Tensor Scale Descriptor with Influence Zones* – TSDIZ).

O TSDIZ é constituído por duas funções: função de extração do vetor de características (ϵ_{TSDIZ}), apresentada na Seção 4.1; e função de similaridade (δ_{TSDIZ}), apresentada na Seção 4.2.

4.1 Função de extração do vetor de características (ϵ_{TSDIZ})

Primeiramente, o *Tensor Scale* é computado para todos os *pixels* do objeto. A seguir, o contorno é dividido em um número n_s predefinido de segmentos de mesmo tamanho. Em um terceiro passo, o TSDIZ mapeia a média angular das orientações das elipses encontradas nas zonas de influência correspondentes ao segmento, para o segmento no contorno.

O vetor de características é formado por n_s médias angulares, na ordem em que são mapeadas para os segmentos do contorno.

Cada passo da extração será detalhado a seguir.

4.1.1 Computação das elipses *Tensor Scale*

A primeira fase é realizada com a aplicação do algoritmo de cálculo de *Tensor Scale* via Transformada Imagem-Floresta, apresentado na Subseção 3.2.3 do Capítulo 3, por sua maior eficiência em comparação com os outros algoritmos.

O *Tensor Scale* deve ser calculado para todos os pontos do objeto obtendo-se, assim, os valores de anisotropia, orientação e espessura das elipses centradas em cada *pixel* do objeto.

4.1.2 Mapeamento das orientações das elipses para os segmentos de contorno do objeto

A primeira etapa é dividir o contorno em um número n_s predefinido de segmentos de mesmo tamanho. Cada segmento receberá um rótulo diferenciado, seguindo a ordem em que estão dispostos no contorno. A Figura 4.1 ilustra um contorno dividido em 10 partes iguais com rótulos diferentes.

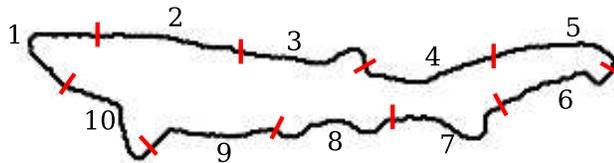


Figura 4.1: Contorno com 10 segmentos de igual comprimento e com rótulos diferentes.

A partir das elipses computadas, utiliza-se a o mapa de rótulos da Transformada de Distância Euclidiana para mapear as orientações destas elipses para os segmentos de contorno do objeto. Esta TDE é calculada via IFT (Algoritmo 3.2), formando-se as zonas de influência referentes a cada segmento do contorno. Cada *pixel* q dentro de uma zona de influência possuirá $rotulo[q] = rotulo[raiz[q]]$, que é o mesmo rótulo do segmento no qual $raiz[q]$ está contido.

Pode-se observar que a TDE utilizada na computação das elipses *Tensor Scale* pode retornar tanto o mapa de raízes, como o mapa de rótulos que é utilizado nesta fase. Neste caso, a função de rotulação λ atribui a cada segmento do contorno um rótulo diferente. Assim, o rótulo de cada segmento é propagado na imagem, formando as zonas de influência (regiões discretas de Voronoi) de cada segmento.

O mapeamento é feito de acordo com o Algoritmo 4.1.

Algoritmo 4.1 Mapeamento das orientações para os segmentos do contorno do objeto.

Entrada: Uma imagem binária I contendo um único objeto O , número n_s de segmentos no contorno e os vetores Ani e Ori que contêm a anisotropia e a orientação das elipses *Tensor Scale*, computadas para todos os *pixels* de O , respectivamente.

Saída: Vetor de características F que contém as orientações mapeadas para cada segmento do contorno.

Estruturas de dados auxiliares: O vetor V de listas para armazenar informações das elipses em cada zona de influência.

```

for all  $p \in O$  do
    acrescentar o par  $(Ani[p], Ori[p])$  à lista  $V[rotulo[p]]$ , sendo  $rotulo[p]$  o rótulo da zona de
    influência na qual o pixel  $p$  está contido;
end for
for all  $i \in [1, \dots, n_s]$  do
     $F[i] = MédiaAngularPonderada(V[i]);$ 
end for

```

A função $MédiaAngularPonderada(V[i])$ retorna a média angular ponderada das orientações das elipses contidas na zona de influência de rótulo i , considerando-se as anisotropias destas elipses como peso. O cálculo da média angular será visto na seção a seguir.

O vetor de características é formado pelos valores $F[i]$, para $i = 1, 2, \dots, n_s$.

4.1.3 Cálculo da média angular

Medidas angulares são utilizadas em diversas áreas do conhecimento, assim como na Meteorologia (direção de ventos e correntes marinhas) e Geologia (orientação da fratura em uma rocha) [41]. Neste trabalho, as orientações das elipses *Tensor Scale* também são consideradas medidas angulares, ou variáveis circulares. A propriedade fundamental de uma medida circular é que o início e o fim de seu intervalo coincidem (e.g., $0^\circ = 360^\circ$; ou no caso das orientações das elipses *Tensor Scale*, $0^\circ = 180^\circ$).

Uma variável circular é representada por um ponto P sobre a circunferência de círculo unitário centrado na origem O do sistema de coordenadas cartesianas e, portanto, \overrightarrow{OP} é um vetor unitário em \mathbb{R}^2 . Assim como as orientações das elipses *Tensor Scale*, esta variável pode ainda ser representada como um ângulo θ , formado pelo vetor \overrightarrow{OP} e o semi-eixo positivo O_x , no sentido anti-horário [41].

Considerando esta última representação, uma variável circular pode ser vista como um par ordenado $(\cos(\theta), \sin(\theta))$, em coordenadas cartesianas. Porém, existem casos em que as variáveis não possuem direção, como por exemplo duas elipses *Tensor Scale* de mesma orientação e angulações 45° e 225° . Este tipo de dado é denominado axial, diferentemente

do anterior, denominado vetorial [32]. Para utilizar a representação em par ordenado em função de θ , o dado axial deve ser convertido em vetorial, duplicando-se o ângulo θ e reduzindo o valor módulo 360° [48]. Isso se deve ao fato de que, se o dado não possui uma direção, o ângulo pode ser interpretado tanto como θ ou como $\theta + 180^\circ$. Para que ambas representações sejam igualmente consideradas durante uma análise estatística, é preciso que elas possuam o mesmo valor e, sendo assim, $2\theta = 2(\theta + 180^\circ) = 2\theta + 360^\circ$. Na Figura 4.2, pode-se observar esta situação, onde as orientações θ e $\theta + 180^\circ$ são representadas na mesma direção.

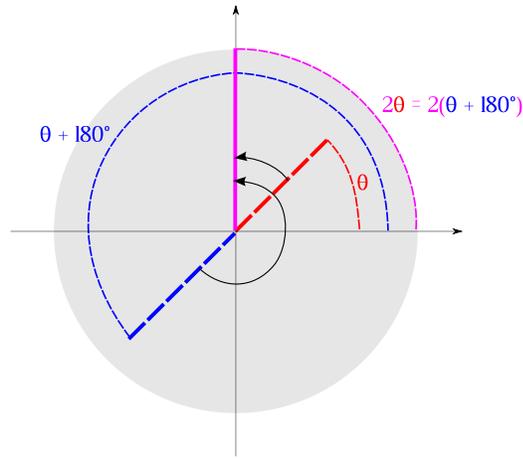


Figura 4.2: Dados axiais circulares.

Para o cálculo da média angular entre variáveis do tipo circular, é incorreto utilizar a mesma média de variáveis lineares, como se pode observar no exemplo a seguir. Sejam $\theta_A = 1^\circ$ e $\theta_B = 179^\circ$ duas orientações de elipses *Tensor Scale*. Observa-se que tais orientações são bem próximas entre si e equidistantes do semi-eixo positivo O_x . Se considerarmos estas orientações como variáveis lineares, a média seria $\frac{1^\circ + 179^\circ}{2} = 90^\circ$ que representa um orientação muito diferente das originais. Sendo assim, deve-se utilizar o modelo para média direcional apresentado em [32].

Sejam $T = \{\theta_1, \theta_2, \dots, \theta_n\}$ um conjunto de n orientações (variáveis circulares) e os vetores unitários $\vec{OP}_1, \vec{OP}_2, \dots, \vec{OP}_n$ associações vetoriais a cada elemento de T , respectivamente.

A média direcional de $\theta_1, \theta_2, \dots, \theta_n$ é definida como sendo o ângulo $\bar{\theta}$, correspondente ao vetor resultante da soma $\vec{OP}_1 + \vec{OP}_2 + \dots + \vec{OP}_n$. Este ângulo resultante deve satisfazer as condições $\cos(\bar{\theta}) = S$ e $\sin(\bar{\theta}) = C$, onde

$$S = \sum_{i=1}^n \cos(\theta_i) \quad \text{e} \quad C = \sum_{i=1}^n \sin(\theta_i) \quad (4.1)$$

$$C = \sum_{i=1}^n \cos(\theta_i). \quad (4.2)$$

Sendo assim, $\bar{\theta}$ é calculado algebricamente como:

$$\bar{\theta} = \begin{cases} \arctan\left(\frac{S}{C}\right), & \text{se } S \geq 0 \text{ e } C > 0, \\ \arctan\left(\frac{S}{C}\right) + 180^\circ, & \text{se } C < 0, \\ \arctan\left(\frac{S}{C}\right) + 360^\circ, & \text{se } S < 0 \text{ e } C > 0 \text{ e} \\ 90^\circ, & \text{caso contrário.} \end{cases} \quad (4.3)$$

No caso das orientações *Tensor Scale*, que são dados axiais, os valores considerados no cálculo devem ser duplicados e a média é definida como $\frac{\bar{\theta}}{2}$ [48].

Como cada elipse ocupa uma área diferente dentro do objeto, é natural que cada elipse possua um peso diferente no cálculo da média. Sendo assim, devem-se utilizar as anisotropias dessas elipses como peso para calcular a média ponderada. Então, para a média ponderada de elipses *Tensor Scale*, substitui-se S e C , na Equação 4.3, por S_2 e C_2 , respectivamente:

$$S_2 = \sum_{i=1}^n \text{anisotropia}_i * \text{sen}(2\theta_i) \text{ e} \quad (4.4)$$

$$C_2 = \sum_{i=1}^n \text{anisotropia}_i * \text{cos}(2\theta_i). \quad (4.5)$$

4.1.4 Formação do vetor de características

O vetor de características do descritor TSDIZ é formada por n_s posições correspondentes às orientações médias das zonas de influência mapeadas para os segmentos.

A Figura 4.3 mostra as curvas de orientação para uma imagem e a mesma imagem rotacionada em 45° .

4.2 Função de similaridade (δ_{TSDIZ})

A função de similaridade deve ser capaz de determinar a diferença de rotação das orientações entre dois contornos. Além disso, deve determinar o ponto, em cada contorno, no qual os vetores devem ser alinhados, a fim de obter o melhor casamento entre eles.

O algoritmo exaustivo para realizar esta tarefa consiste em registrar as curvas de orientação. Sejam $\alpha = 0^\circ, \dots, 179^\circ$ e $j = 1, \dots, n_s$. O algoritmo computa, para cada rotação α e para cada

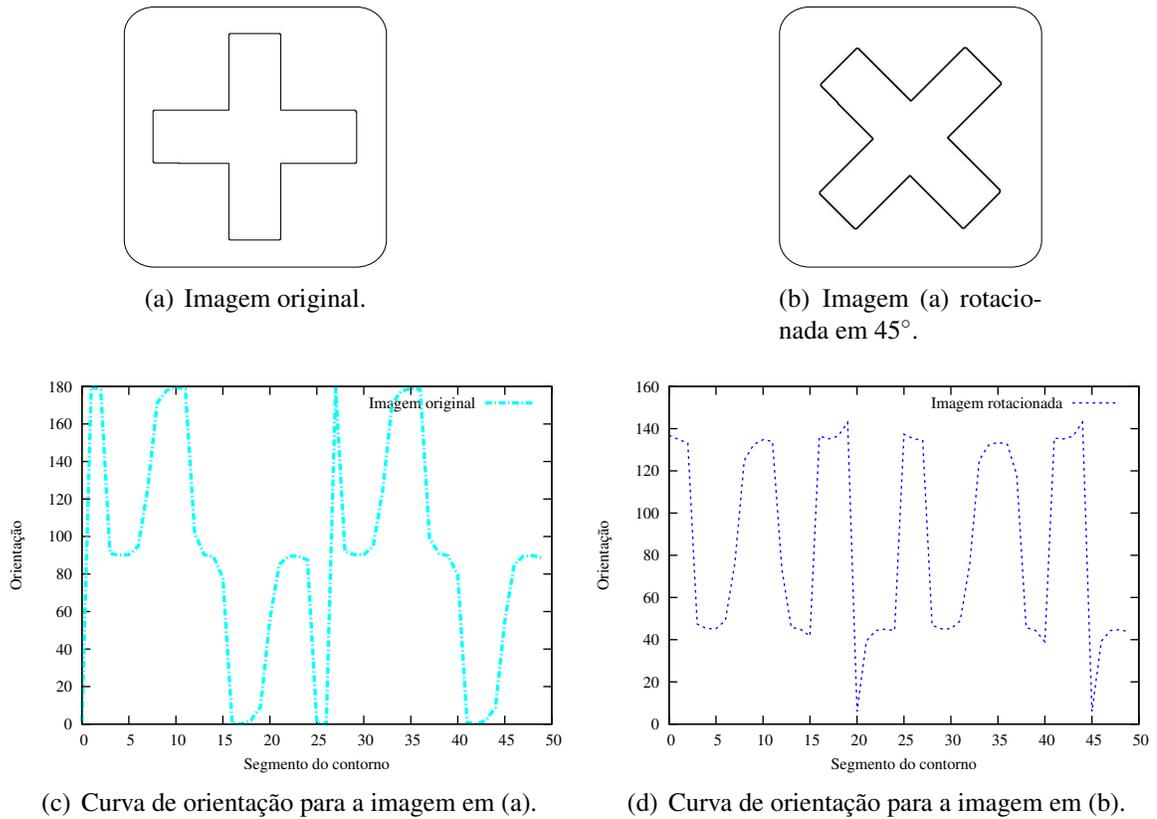


Figura 4.3: Exemplos de curvas de orientação obtidas com o descritor TSDIZ.

deslocamento j , a diferença entre os vetores após a rotação das orientações de um vetor por α e o deslocamento circular do mesmo vetor em j posições. A diferença mínima calculada corresponde à distância entre os dois vetores.

A desvantagem deste método é sua complexidade $O(c * n_s^2)$, onde c é uma constante, neste caso 180° . O Algoritmo 4.2 mostra os passos descritos para o cálculo da distância entre dois vetores. A função *DistânciaAngular*, no Algoritmo 4.2, retorna o menor ângulo entre as orientações.

A Figura 4.4 mostra as curvas registradas das imagens presentes nas Figuras 4.3(a) e 4.3(b). Como pode ser observado, a função de distância permite caracterizar o método como invariante à rotação. Para este exemplo, a menor distância encontrada pelo Algoritmo 4.2 foi com $j = 27$ e $\alpha = 45^\circ$.

Uma função mais simples para o cálculo da similaridade consiste na utilização da média das orientações das elipses *Tensor Scale* calculadas para todos os *pixels* do objeto. Esta média é calculada assim como foi apresentada na Subseção 4.1.3 e é realizada na fase de extração

Algoritmo 4.2 Similaridade entre dois vetores TSDIZ.

Entrada: Dois vetores de características F_A e F_B .

Saída: Distância $dist$ entre F_A e F_B .

Estruturas de dados auxiliares: Variável $dist_{aux}$ para armazenar temporariamente uma distância.

```

 $dist \leftarrow \infty;$ 
for all  $j \in [1, \dots, n_s]$  do
  for all  $\alpha \in [0^\circ, \dots, 179^\circ]$  do
    for all  $i \in [1, \dots, n_s]$  do
       $dist_{aux} \leftarrow Dist\grave{a}nciaAngular(\{F_B[(j - i) \bmod n_s] + \alpha\} \bmod 180^\circ, F_A[i]);$ 
      if  $dist_{aux} < dist$  then
         $dist \leftarrow dist_{aux};$ 
      end if
    end for
  end for
end for

```

de características. O valor obtido é somado a cada orientação (reduzindo-se módulo 180° o resultado da soma) antes de serem incorporadas ao vetor de características. Recebendo os vetores já com as orientações rotacionadas, a função de similaridade não precisa mais do passo no qual são testadas todas as possíveis rotações de 0° a 179° .

Esta versão simplificada da função é uma aproximação do melhor registro. No entanto, sua complexidade é $O(n_s^2)$. Para o exemplo do registro das curvas para as imagens presentes nas Figuras 4.3(a) e 4.3(b), este método determina que a rotação entre as imagens é de $44,84^\circ$, ao invés de 45° .

4.3 Experimentos

Nesta seção, apresentam-se os resultados dos experimentos executados para avaliar o descritor TSDIZ.

4.3.1 Base de imagens

Para os experimentos, a mesma base do experimento MPEG-7 CE-shape-1 part B [2] foi utilizada. A base é formada por mil e quatrocentas imagens, sendo setenta classes com vinte imagens cada. É composta por silhuetas de diversos objetos, como frutas e animais.

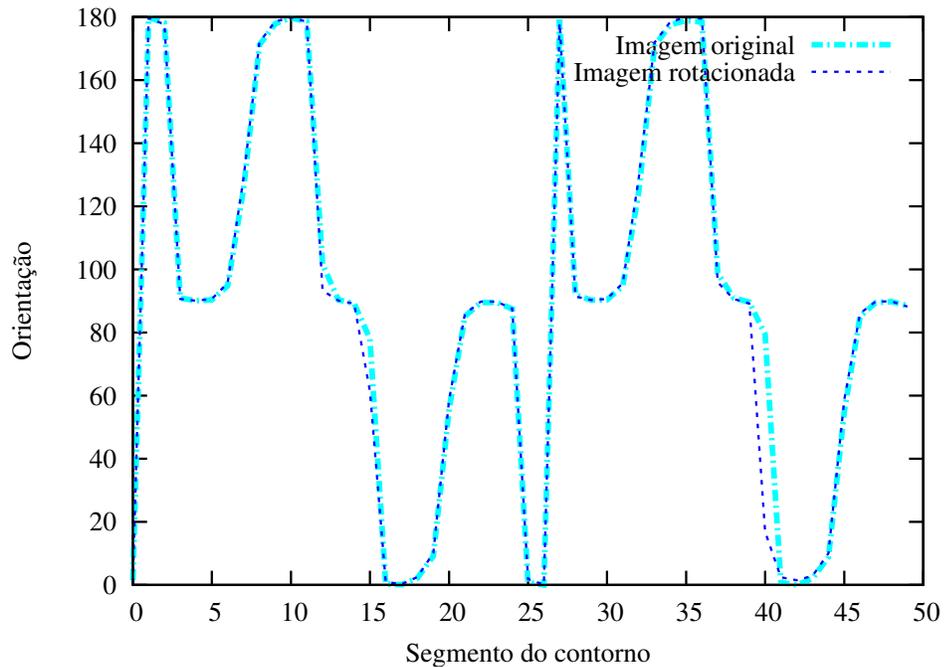


Figura 4.4: Curvas registradas das imagens presentes nas Figuras 4.3(a) e 4.3(b).

4.3.2 Resultados

Os experimentos consistem no cálculo de três medidas – separabilidade multiescala, precisão vs. revocação (medidas gráficas) e revocação em 40 (medida de único valor) – para o TSDIZ e outros descritores de forma, a título de comparação, com a utilização da base de imagens MPEG-7 CE-shape-1 part B.

Cada uma das medidas de eficácia utilizadas avalia um aspecto da recuperação de imagens, sendo que uma análise da combinação das medidas ajuda a caracterizar melhor a eficácia dos descritores. Nas próximas seções serão apresentados os experimentos realizados, utilizando-se as três medidas.

Precisão vs. Revocação

As curvas de Precisão vs. Revocação (*Precision vs. Recall*) [8, 55] são as medidas mais utilizadas para avaliação de eficácia no domínio de CBIR. Precisão é definida com sendo a fração das imagens recuperadas que são relevantes à busca. Em contraste, revocação mede a proporção de imagens relevantes dentre as imagens recuperadas. Ou seja,

$$Precisão = \frac{\{\text{imagens relevantes}\} \cap \{\text{imagens recuperadas}\}}{\{\text{imagens recuperadas}\}}; \quad (4.6)$$

$$Revocação = \frac{\{\text{imagens relevantes}\} \cap \{\text{imagens recuperadas}\}}{\{\text{imagens relevantes}\}}. \quad (4.7)$$

A curva de Precisão vs. Revocação, ou simplesmente curva PR, indica o compromisso entre as duas medidas e, geralmente, a curva mais alta no gráfico indica uma melhor eficácia do descritor sendo avaliado.

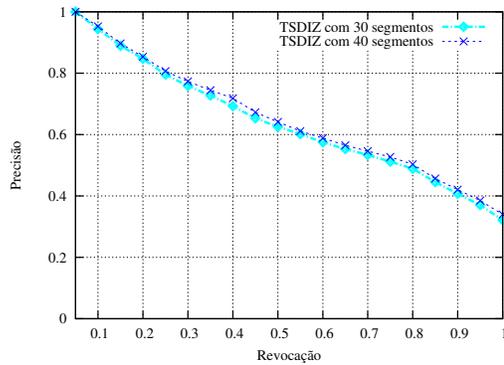
Neste experimento, cada imagem da base foi utilizada como entrada e as imagens relevantes consideradas foram as de mesma classe da imagem de entrada.

Como a eficácia do descritor TSDIZ pode variar com a quantidade de segmentos em que o contorno de um objeto foi subdividido, vários valores foram testados. As curvas PR para estes valores é apresentada na Figura 4.5.

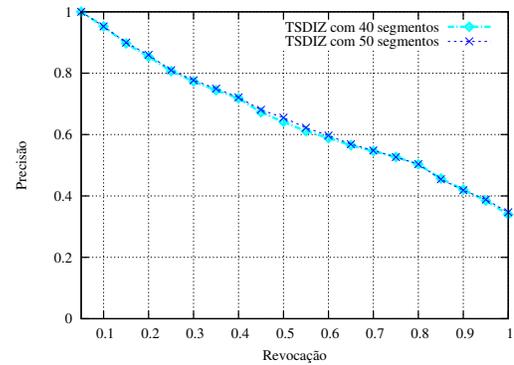
Como pode ser observado na Figura 4.5, a eficácia do descritor não foi muito alterada pela variação do número de segmentos no contorno. Por isso, outras medidas foram utilizadas para confirmar esta avaliação, como será visto nas próximas seções. As curvas foram representadas duas a duas porque, como a variação é muito pequena, a visibilidade das curvas poderia ficar comprometida, caso fossem representadas em um mesmo gráfico.

A curva do TSDIZ com 60 segmentos foi comparada com a curva de alguns descritores de forma: *Beam Angle Statistics* (BAS), Dimensão Fractal Multiescala (MS Fractal), Invariantes de Momento (MI), Descritor de Fourier (Fourier), Descritor *Tensor Scale* (TSD) e Saliências de Segmento (SS). Todos estes descritores de forma estão detalhados na Seção 2.4. A Figura 4.6 mostra a comparação entre as curvas dos descritores.

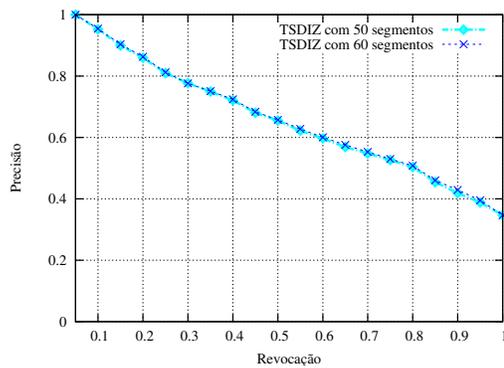
Percebe-se que o TSDIZ apresenta a segunda melhor curva de PR dentre os descritores analisados. Como o descritor BAS apresentou a melhor curva PR, uma análise comparativa mais profunda com este descritor será realizada.



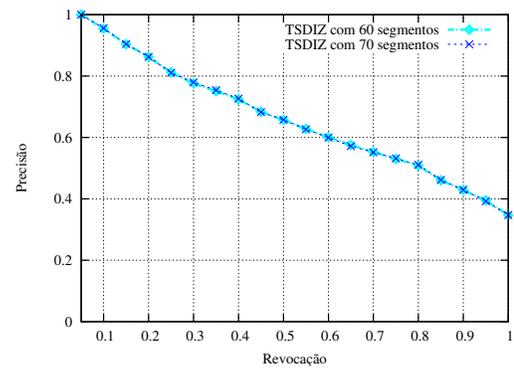
(a) TSDIZ com 30 e 40 segmentos.



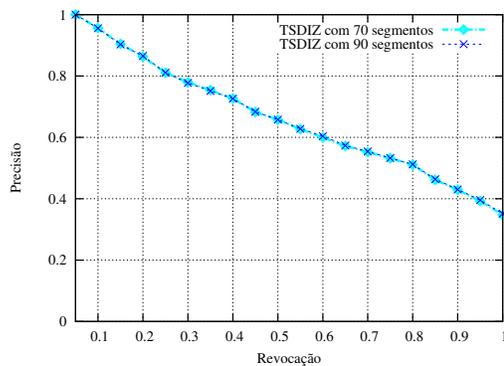
(b) TSDIZ com 40 e 50 segmentos



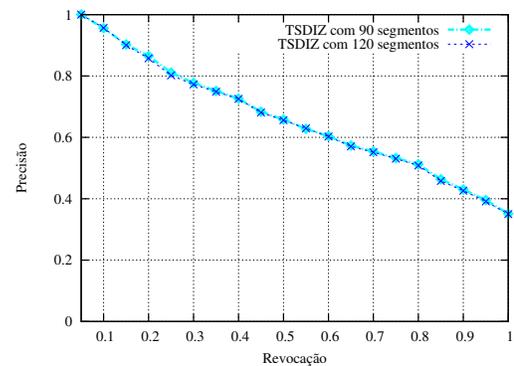
(c) TSDIZ com 50 e 60 segmentos



(d) TSDIZ com 60 e 70 segmentos



(e) TSDIZ com 70 e 90 segmentos



(f) TSDIZ com 90 e 120 segmentos

Figura 4.5: Curvas Precisão vs. Revocação para o descritor TSDIZ, considerando diferentes números de segmentos no contorno.

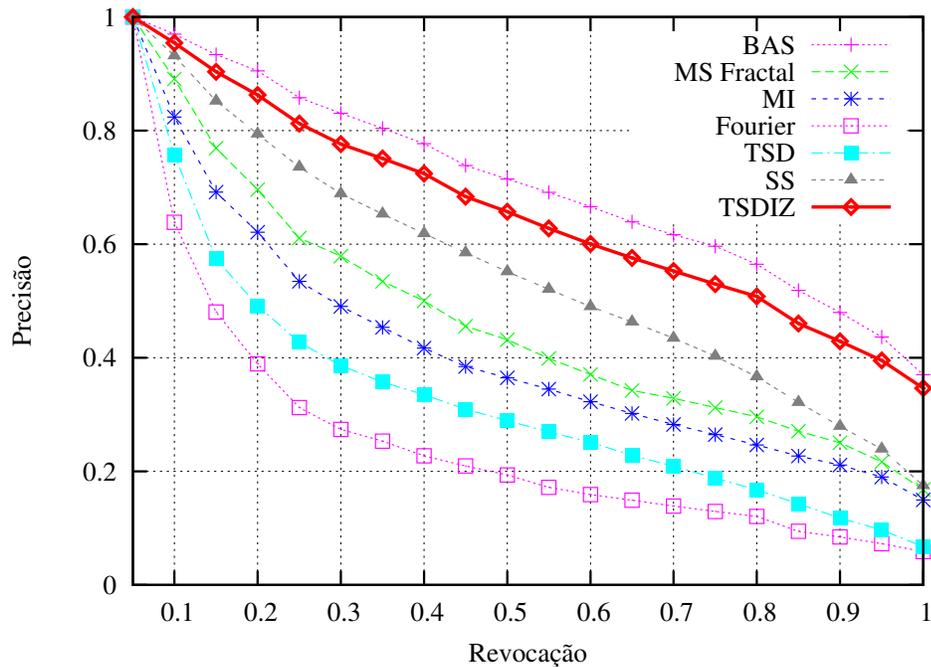


Figura 4.6: Curva de Precisão vs. Revocação para diversos descritores.

Revocação em 40

Muitas vezes é importante que tenhamos medidas de eficácia expressas apenas por um único número, para que possa ser colocado em uma escala obtendo valores absolutos e relativos entre descritores. Um exemplo deste tipo de medida é o valor da revocação após um número x de imagens recuperadas. Para os experimentos, utilizou-se $x = 40$, pois este é o valor recomendado pelo experimento MPEG-7 CE-shape-1 part B [40].

Todas as 1400 imagens da base foram utilizadas como imagens de consulta e o número de imagens relevantes (as que pertencem à mesma classe da imagem de entrada) dentre as 40 primeiras imagens recuperadas foi contado. Como o número máximo de acertos para uma dada imagem de entrada é 20 (pois cada classe da base utilizada contém 20 imagens), a soma dos acertos de todas as imagens é 28000.

O gráfico apresentado na Figura 4.7 indica a taxa de similaridade do descritor TSDIZ em função do número de segmentos no qual o contorno do objeto foi subdividido. Esta taxa de similaridade é calculada pela razão entre o número de acertos obtidos e o número total de acertos possíveis.

Como esperado, a quantidade de segmentos no contorno não altera significativamente a taxa de similaridade, porém o melhor resultado foi obtido com 60 segmentos.

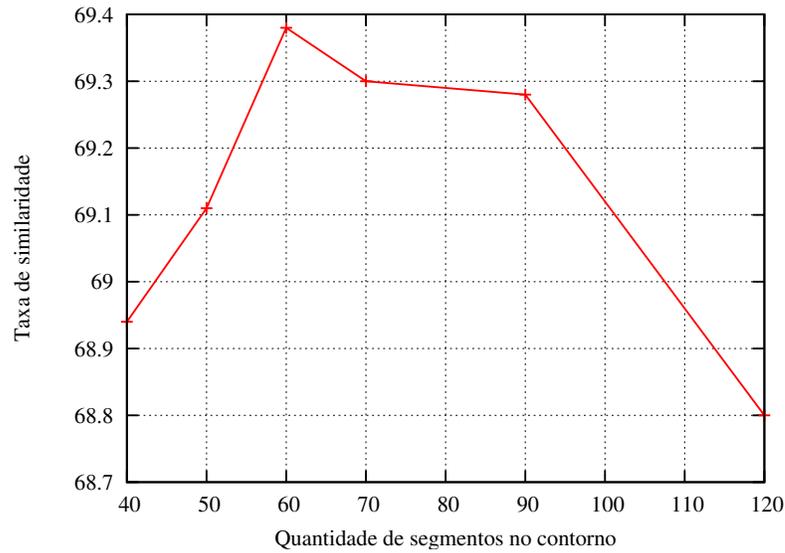


Figura 4.7: Taxas de similaridade do descritor TSDIZ, considerando diferentes números de segmento no contorno.

A Tabela 4.1 mostra as taxas de similaridade para o descritor BAS e para o TSDIZ, considerando diferentes números de segmento no contorno.

Descritores	Taxa de Similaridade
BAS	75.13
TSDIZ 40	68.94
TSDIZ 50	69.11
TSDIZ 60	69.38
TSDIZ 70	69.30
TSDIZ 90	69.28
TSDIZ 120	68.80

Tabela 4.1: Taxa de similaridade para o descritor BAS e para o TSDIZ, considerando diferentes números de segmento no contorno.

O valor para o descritor BAS também foi obtido experimentalmente e é diferente do valor apresentado em [6]. Esta diferença encontrada se deve principalmente à definição de contorno utilizada. Enquanto neste trabalho o contorno é formado por todos os *pixels* que possuem pelo menos um vizinho-8 não pertencente ao objeto, os experimentos realizados para o trabalho em [6], segundo o autor, consideram o contorno como formado pelos *pixels* que possuem pelo menos um vizinho-8 que possua um vizinho-8 não pertencente ao objeto.

Não obstante o BAS ter taxa de similaridade geral maior, o TSDIZ 60 consegue melhores resultados em muitas das classes da base de imagens. A Figura 4.8 mostra exemplos de recuperação do BAS e do TSDIZ 60, para cinco imagens de entrada. As imagens destacadas são as que não pertencem à mesma classe da imagem de entrada (imagens não relevantes).

Algumas classes possuem características que o TSDIZ não consegue captar. A Figura 4.9 ilustra situações em que isto ocorre. Pelos exemplos pode-se perceber que o TSDIZ não é um bom descritor para objetos que possuem muita variação de orientação dentro da forma, como é o caso das imagens apresentadas na Figura 4.9. Nestes casos, o TSDIZ tende a achar as imagens com formas mais simples como mais próximas da imagem de entrada. Isso ocorre pois, por mais que a imagem de entrada com muita variação de orientação e uma imagem com uma forma simples sejam diferentes visualmente, estas possuirão menor discordância de orientações do que a imagem de entrada com qualquer outra forma mais complexa da base.

Em contrapartida, percebe-se que o TSDIZ é um bom descritor para objetos que possuem ruído no contorno, como pode ser observado nas imagens do quarto exemplo apresentado na Figura 4.8. Isso ocorre, pois o BAS utiliza informação diretamente do contorno e, sendo assim, qualquer variação afetará bastante o resultado. Já o TSDIZ utiliza informação de dentro do objeto e, por isso, ruídos no contorno não afetam muito o resultado.

O resultado da taxa de similaridade para cada uma das classes da base de imagens pode ser observado na Figura 4.10. A última coluna do gráfico representa a média da taxa de similaridade, considerando-se todas as classes.

Esta base de imagens utilizada para avaliar o TSDIZ foi construída para o experimento MPEG-7 CE-shape-1 part B, do projeto MPEG-7. Neste experimento, são comparados descritores baseados em contorno, baseados em região e baseados em esqueleto, utilizando-se a medida Revocação em 40 [40]. Como resultado deste experimento, observou-se que os descritores baseados em contorno apresentaram melhor resultado quando comparados aos descritores baseados em região ou esqueleto. Por este razão, pode-se concluir que os objetos desta base de imagens são melhor descritos com informações do contorno, justificativa que pode ser aplicada a este trabalho também.

Por causa das conclusões obtidas, os experimentos realizados contam com mais uma etapa, que será descrita a seguir, utilizando-se outra medida de eficácia.

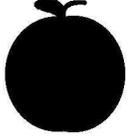
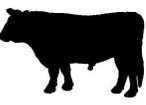
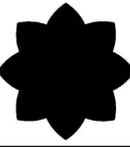
Imagem de entrada	Descritor	1	2	3	4	5	6	7	8	9	10
	TSDIZ										
	BAS										
	TSDIZ										
	BAS										
	TSDIZ										
	BAS										
	TSDIZ										
	BAS										
	TSDIZ										
	BAS										

Figura 4.8: Exemplos de recuperação por similaridade nos quais o TSDIZ apresenta melhor resultado.

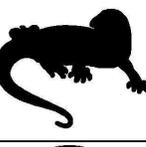
Imagem de entrada	Descritor	1	2	3	4	5	6	7	8	9	10
	TSDIZ										
	BAS										
	TSDIZ										
	BAS										

Figura 4.9: Exemplos de recuperação por similaridade nos quais o BAS apresenta melhor resultado.

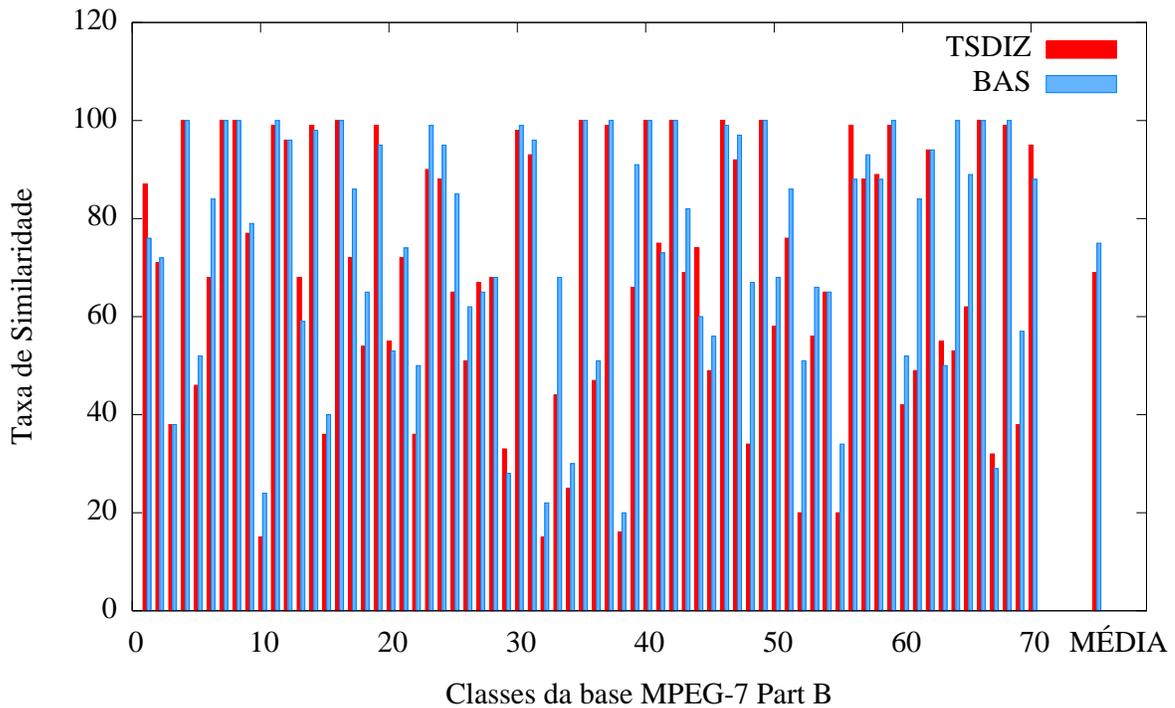


Figura 4.10: Taxa de similaridade para todas as classes da base MPEG-7 CE-shape-1 part B.

Separabilidade Multiescala

Como cada descritor representa o vetor de características como um ponto no espaço métrico correspondente, sua eficácia será maior quanto mais separado forem os agrupamentos de vetores relevantes no espaço métrico. Por causa disso, uma boa medida de eficácia de descritores deve capturar o conceito de separabilidade [22].

Separabilidade indica a habilidade de discriminação de objetos que pertençam a classes distintas. Este conceito é muito usado em análise de agrupamentos (*clusters*) e foi introduzido na área de CBIR por Torres et al. [22]. Na área de CBIR, a medida mais utilizada são as curvas de Precisão vs. Revocação (PR). Porém, em [22], um exemplo é usado para ilustrar que a medida PR não captura o conceito de separabilidade e, portanto, não deve ser usada como medida de eficácia.

Seja Σ o conjunto de κ formas organizadas em classes. A separabilidade $\psi_D(C)$ de um descritor D para uma dada classe C é definida como a seguir. Uma forma arbitrária r_C é escolhida como referência para a classe C e a distância $\overline{\Delta_D(r_C, i)} = \frac{\Delta_D(r_C, i)}{M}$, na qual $M = \max_{i \in \Sigma, \forall r_C} \{\Delta_D(r_C, i)\}$, é computada para todas as formas $i \in \Sigma$.

O raio de distância é quantizado em um certo número de valores de x a 1.0, por exemplo

50 ($x = 0.02$ e intervalos de 0.02). Seja $\eta_{r_C}(x)$ o número de formas cuja distância da forma de referência é menor ou igual a x ($\Delta(r_C, i) \leq x$) e não pertençam à mesma classe C ($i \notin C$). Para cada valor de distância de x a 1.0, a separabilidade $\psi_D(C)$ de um descritor D em relação à classe C é definida como:

$$\psi_D(C) = 1 - \frac{\eta_{r_C}(x)}{\kappa}.$$

Os valores de separabilidade definem uma curva multiescala para a classe C ao longo de x . A separabilidade do descritor D é definida como a média da separabilidade multiescala para todas as classes.

A curva de separabilidade calculada para os descritores TSDIZ 60 e BAS é apresentada na Figura 4.11. Como pode-se observar, os descritores apresentam eficácia equivalente para raios de busca menores que 10% da distância máxima. Deste ponto em diante, a curva de separabilidade do BAS decresce rapidamente, indicando que este descritor não é robusto ou eficaz para raios maiores que 20%.

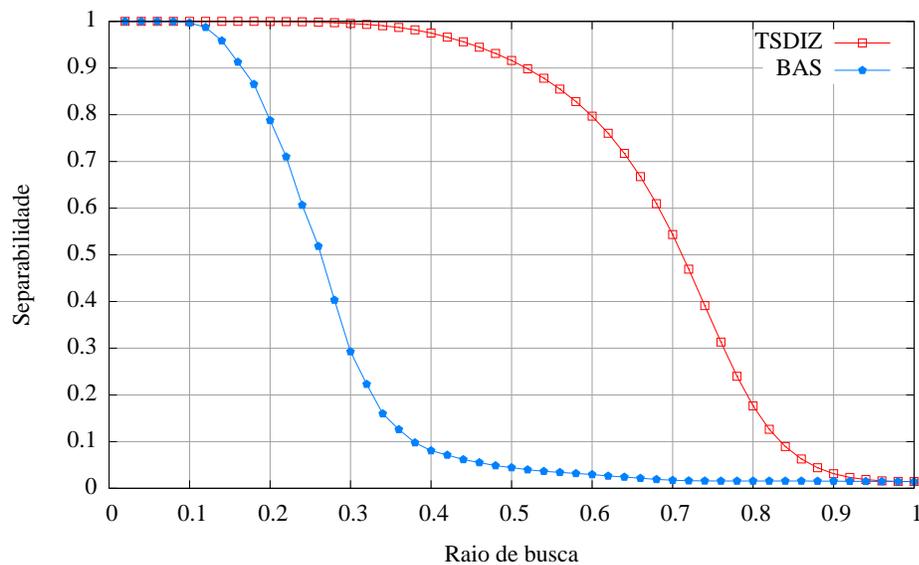


Figura 4.11: Curva de separabilidade multiescala para a base de imagens MPEG-7 CE-shape-1 part B.

A utilização do descritor TSDIZ resulta em uma melhor separabilidade entre as classes, embora as outras medidas não tenham captado esta característica do descritor.

Invariância às transformações geométricas

A partir dos experimentos realizados na base de imagens MPEG-7 CE-shape-1 part B, pode-se analisar o comportamento do descritor mediante a presença de transformações geométricas:

- Translação: percebe-se que o descritor TSDIZ é invariante à translação, pois todo o seu cálculo é referente ao mapeamento de propriedades de dentro do objeto para seu contorno. Sendo assim, independe da localização deste objeto na imagem.
- Rotação: este tipo de transformação é corrigida na fase de casamento das curvas TSDIZ. Como a rotação da imagem só causa a rotação das elipses *Tensor Scale* calculadas, o deslocamento no vetor por j e a rotação de seus elementos por α , na função de similaridade, devem ser capazes de resolver este tipo de transformação.
- Escala: embora um estudo mais aprofundado deste tipo de transformação deva ser realizado, a princípio pode-se verificar que a escala (igual em ambas as direções) não afeta o descritor. Isto pode ser considerado, pois utiliza-se uma medida estatística (média ponderada) para o cálculo da orientação final de cada segmento e, sendo assim, um aumento na quantidade de elipses dentro do objeto não deve mudar o resultado desta medida estatística.

4.4 Resumo

Este capítulo apresentou o descritor *Tensor Scale* por zona de influência (TSDIZ) que utiliza, em conjunto, as informações das elipses *Tensor Scale* em uma mesma zona de influência dentro do objeto.

Os experimentos consistiram na comparação do TSDIZ com outros descritores de forma, utilizando-se as medidas Precisão vs. Revocação (PR), Revocação em 40 e separabilidade multiescala. Além disso, alguns exemplos de recuperação de imagens por similaridade foram apresentados.

Os resultados experimentais mostraram a superioridade do TSDIZ em relação aos outros descritores, utilizando-se curva PR. Como a curva PR do descritor BAS foi superior à curva do TSDIZ, uma comparação mais detalhada entre os dois descritores foi realizada. Usando-se, por exemplo, a medida separabilidade multiescala, o TSDIZ mostrou-se mais eficaz e robusto do que o descritor BAS.

Capítulo 5

Descritor de forma baseado em saliências do contorno detectadas por *Tensor Scale*

As saliências de uma forma são definidas como os pontos de maior curvatura ao longo do contorno [23], ou vértices ao longo do contorno com descontinuidade na derivada primeira [18]. A sua detecção é a chave para várias aplicações em processamento de imagens e visão computacional. Alguns exemplos são: registro de imagens (Figura 5.1), aproximação poligonal [63], análise de movimento [59] e descrição de forma [22].

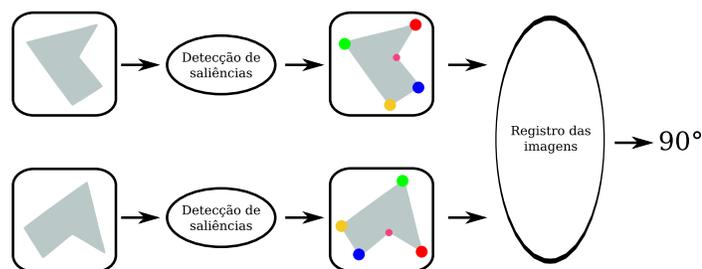


Figura 5.1: Registro de imagens a partir de pontos de saliência.

A Figura 5.1 mostra uma aplicação de pontos de saliência para o registro de duas imagens. Correlacionando-se os pontos de saliências detectados no contorno dos objetos contidos nas imagens, o ângulo de rotação entre as imagens (no exemplo, 90°) pode ser encontrado.

Um detector de saliências deve satisfazer alguns critérios importantes [53], como:

- Todas as saliências verdadeiras devem ser detectadas;
- Falsas saliências não devem ser detectadas;
- Pontos de saliência detectados devem ser bem localizados;
- O detector de saliências deve ser robusto a ruídos (e.g., cantos arredondados ou picos no contorno do objeto);
- O detector deve ser eficiente.

A idéia da utilização de saliências na descrição de forma é baseada no fato apontado por Attneave, em [7]. Attneave afirmou que a informação visual é altamente redundante no espaço e no tempo. Ele demonstrou que um tipo de redundância está relacionado a segmentos de reta ao longo do contorno de uma forma, significando que os pontos de alta curvatura concentram mais informação do que outros pontos na forma. Por esta razão, pode-se concluir que a curvatura, por concentrar muita informação da forma, é uma importante característica para a identificação de aspectos geométricos.

Baseando-se nas redundâncias da informação visual apontadas por Attneave [7], propõe-se a utilização das saliências como pontos-chave na descrição da forma. O descritor de forma baseado em saliências do contorno detectadas por *Tensor Scale*, ou *Tensor Scale Contour Saliences Descriptor* (TSCS), é constituído por duas funções: função de extração do vetor de características (ϵ_{TSCS}), apresentada na Seção 5.1; e função de similaridade (δ_{TSCS}), apresentada na Seção 5.2.

5.1 Função de extração do vetor de características (ϵ_{TSCS})

A extração de características neste descritor inicia-se pela detecção dos pontos de saliência ao longo do contorno de um objeto. Logo após, os valores das saliências detectadas são calculados. Em seguida, estes valores são usados para a formação de vetores de características. Estas três fases serão detalhadas a seguir.

5.1.1 Detecção de pontos de saliência

A detecção de pontos de saliências de um contorno ocorre em três etapas:

- Computação das elipses *Tensor Scale* para todos os *pixels* do objeto;
- Mapeamento das orientações das elipses para o contorno do objeto; e

- Aplicação da função de diferença nos pontos de contorno com as orientações mapeadas, com posterior limiarização dos valores obtidos.

Cada uma das etapas será detalhada a seguir.

Computação das elipses *Tensor Scale*

Assim como no descritor TSDIZ, calcula-se, para todos os *pixels* do objeto contido na imagem, o *Tensor Scale* com o algoritmo via Transformada Imagem-Floresta, apresentado na Subseção 3.2.3 do Capítulo 3. Desta forma, obtêm-se os valores de anisotropia, orientação e espessura das elipses centradas em cada *pixel* do objeto.

Mapeamento das orientações das elipses para o contorno do objeto

A partir das elipses computadas, utiliza-se a Transformada de Distância Euclidiana para mapear as orientações destas elipses. Ao invés de mapear a orientação média das elipses em uma zona de influência para o segmento do contorno, a exemplo do TSDIZ, o TSCS mapeia a orientação da elipse de maior anisotropia na zona de influência para o *pixel* raiz no contorno.

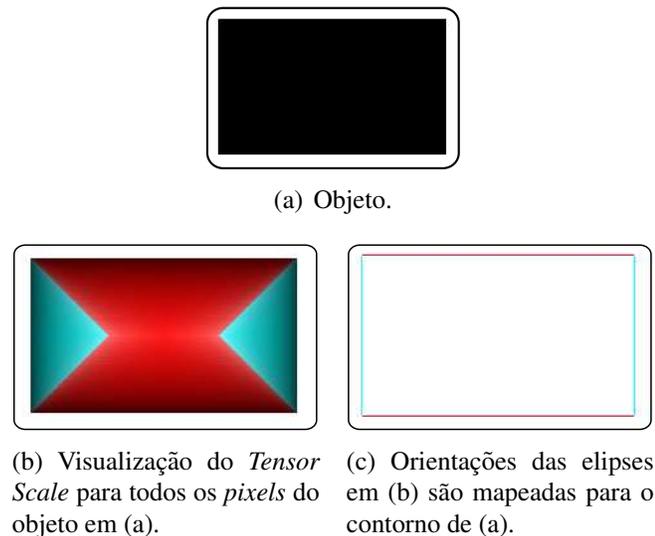


Figura 5.2: Mapeamento do *Tensor Scale*.

Calculando-se a TDE por meio da IFT (Algoritmo 3.2), tem-se que todo *pixel* s em S (sendo S o conjunto de sementes ou, neste caso, o conjunto que compreende os *pixels* de contorno seguindo a ordem em que circunscrevem o objeto, a partir de um *pixel* de partida) é a raiz de uma região, ou zona de influência, formada pelos *pixels* do objeto que são mais próximos

a s do que a qualquer outra raiz em S . Cada *pixel* q nesta região possuirá $raiz[q] = s$. A idéia, então, é mapear para s a orientação da elipse com a maior anisotropia em sua zona de influência (Algoritmo 5.1).

Algoritmo 5.1 Mapeamento das orientações para o contorno do objeto.

Entrada: Uma imagem binária I contendo um único objeto O , um conjunto S de *pixels* de contorno em O , o mapa de raízes $raiz$ resultado da TDE e os vetores Ani e Ori que contêm a anisotropia e a orientação das elipses *Tensor Scale*, computadas para todos os *pixels* de O , respectivamente.

Saída: Versão atualizada dos vetores Ani e Ori ;

```

for all  $p \in S$  do
   $Ori[p] \leftarrow 0$ ;
   $Ani[p] \leftarrow 0$ ;
end for
for all  $p \in O$  do
  if  $Ani[raiz[p]] < Ani[p]$  then
     $Ani[raiz[p]] \leftarrow Ani[p]$ ;
     $Ori[raiz[p]] \leftarrow Ori[p]$ ;
  end if
end for

```

O Algoritmo 5.1 tem como saída dois vetores (Ori e Ani) que são atualizados tal que $Ori[s]$ e $Ani[s]$, para todo *pixel* $p \in S$, contêm a orientação da elipse de maior anisotropia em sua área de influência correspondente e o valor desta anisotropia, respectivamente. Caso a semente não possua zona de influência na imagem, o valor assumido é o mesmo de um de seus vizinhos no contorno.

A Figura 5.2(c) ilustra o contorno de 5.2(a) com as orientações mapeadas das elipses mostradas em 5.2(b).

Diferença entre orientações mapeadas no contorno

Para localizar os pontos de saliência no contorno, calculam-se as diferenças entre orientações mapeadas adjacentes em S . Isso é possível, pois pontos de alta curvatura causam mudança abrupta de orientação ao longo do contorno.

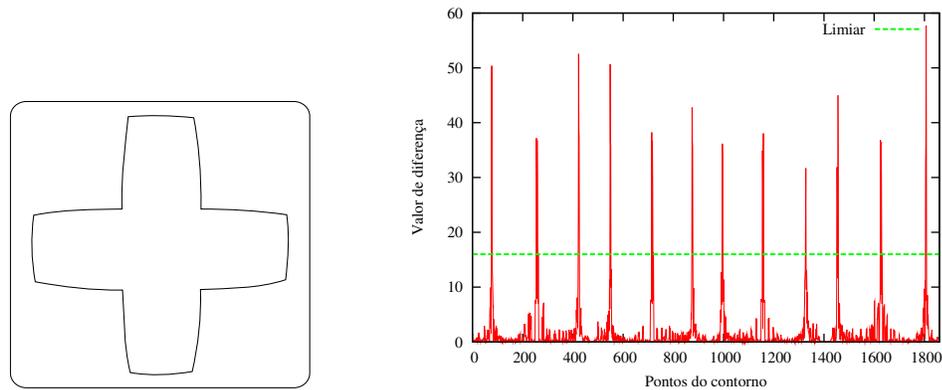
O valor da diferença para todo *pixel* $s \in S$ é:

$$Diferença(s) = DistânciaAngular(Ori[s-1], Ori[s+1]); \quad (5.1)$$

onde a função $DistânciaAngular(\alpha, \beta)$ retorna o menor ângulo entre as orientações α e β .

Agora pode-se utilizar um valor de limiar para eliminar valores baixos de diferença ao longo do contorno.

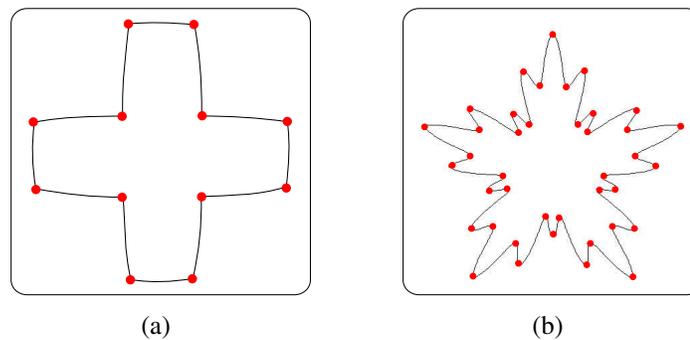
A Figura 5.3(b) apresenta o gráfico dos valores de diferença calculados para todos os pontos do contorno do objeto da Figura 5.3(a). No mesmo gráfico também está representado o limiar adotado para obter o resultado mostrado na Figura 5.4(a). A Figura 5.4 mostra as saliências detectadas (pontos vermelhos) para duas imagens, utilizando-se um limiar de 16° , isto é, saliências relacionadas a diferenças de orientação menores que 16° não foram representadas.



(a) Objeto

(b) Valores de diferença calculados para todos os pontos do contorno de (a).

Figura 5.3: Exemplo de valores de diferença ao longo do contorno de um objeto.



(a)

(b)

Figura 5.4: Exemplos de visualização dos pontos de saliência detectados para duas imagens.

5.1.2 Cálculo dos valores das saliências detectadas

Após a fase de detecção da localização dos pontos de saliência, precisa-se determinar o valor de saliência nestes pontos. É sabido que a área das zonas de influência dos pontos de alta curvatura é maior que a área das zonas de influência de outros pontos no contorno [22]. Também sabe-se que a área da zona de um ponto convexo é maior fora do contorno do que dentro dele, e vice-versa para pontos côncavos [22]. A Figura 5.5 ilustra este fato. As áreas hachuradas representam as zonas de influência internas (pontos côncavos) e as áreas cinzas representam as zonas de influência externas (pontos convexos).

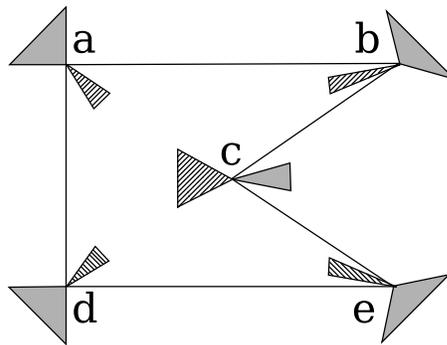


Figura 5.5: Zonas de influência de pontos convexos (a , b , d e e) e côncavo (c).

Para cada ponto detectado, utiliza-se, como valor de saliência, a área da zona de influência dos *pixels* em S . A zona de influência de cada *pixel* é calculada a partir do mapa de raízes gerado pela TDE. Gera-se um histograma do mapa de raízes resultante, para cada lado do contorno, restrito a uma pequena vizinhança da curva para eliminar influência cruzada de partes opostas. Cada *bin* do histograma indica a área da zona de influência do respectivo lado da curva do contorno. Valores negativos de área são utilizados para pontos côncavos e valores positivos para pontos convexos. Este método é o mesmo utilizado no descritor *Contour Salience* [22].

5.1.3 Formação do vetor de características

O vetor de características deste descritor é formado com o mesmo método do descritor CS. A única diferença entre o método proposto e o método do descritor CS é a qualidade dos pontos de saliência detectados ao longo do contorno.

Um ponto arbitrário é utilizado como referência e o método computa a posição relativa de cada ponto de saliência ao ponto de referência. O vetor é composto pelos valores de saliência e suas posições relativas.

A Figura 5.6 ilustra o vetor de características calculado para o contorno de um objeto. A Figura 5.6(a) mostra este objeto e os pontos de saliência detectados, enquanto o gráfico da Figura 5.6(b) representa os valores de saliência *versus* a posição relativa dos pontos, ao longo do contorno, em função do ponto *a*.

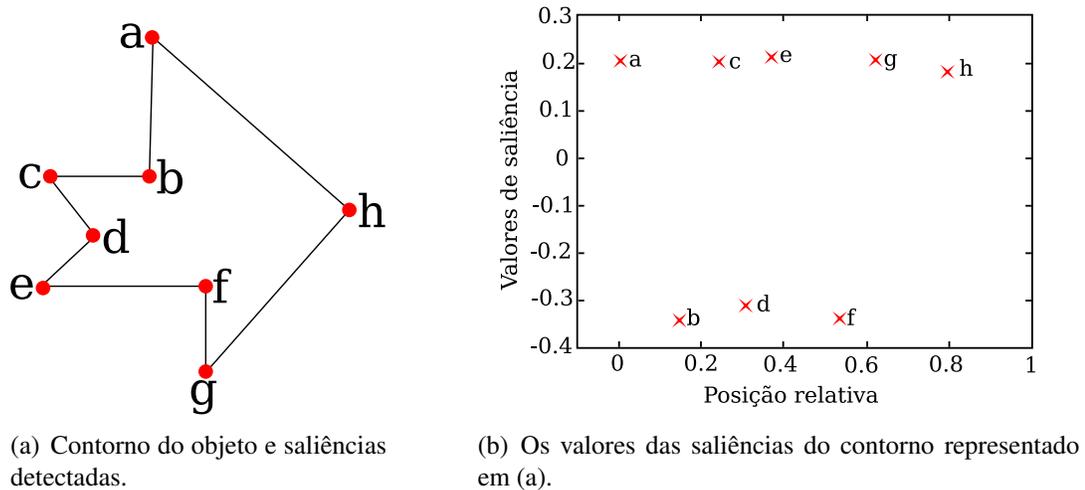


Figura 5.6: Exemplo de formação do vetor de características do descritor TSCS.

5.2 Função de similaridade (δ_{TSCS})

Como utilizamos a mesma formação de vetor de características do descritor CS, a função de similaridade também pode ser a mesma.

A função de similaridade precisa levar em conta duas propriedades do vetor de características gerado: o ponto de referência pode não ser o mesmo para diferentes vetores e o tamanho dos vetores pode ser diferente. Sendo assim, a função de similaridade é um algoritmo heurístico que registra os vetores utilizando-se dos pontos de referência e computa a similaridade entre eles levando em consideração a diferença no tamanho dos vetores. Este algoritmo é baseado no utilizado pelo descritor *Curvature Scale Space* (CSS) [3] e está descrito no Algoritmo 5.2.

A valor *d* retornado pelo Algoritmo 5.2 representa a distância entre os dois vetores de características que foram fornecidos como entrada do algoritmo.

Algoritmo 5.2 Similaridade entre dois vetores TSCS.

Entrada: Dois vetores de características $S_A = \{(u_{A_1}, s_{A_1}), \dots, (u_{A_n}, s_{A_n})\}$ e $S_B = \{(u_{B_1}, s_{B_1}), \dots, (u_{B_m}, s_{B_m})\}$ dos contornos A e B , onde (u_{A_i}, s_{A_i}) representa o i -ésimo valor de saliência s_{A_i} na posição $u_{A_i} \in [0, 1]$ ao longo de A .

Saída: Distância d entre S_A e S_B .

Estruturas de dados auxiliares: Uma lista L de pares de pontos candidatos para casamento (*matching*), de S_A e S_B ; e três vetores de características auxiliares S'_A , S'_B e S''_A .

criar $S'_A = \{(u'_{A_1}, s'_{A_1}), \dots, (u'_{A_n}, s'_{A_n})\}$ e $S'_B = \{(u'_{B_1}, s'_{B_1}), \dots, (u'_{B_m}, s'_{B_m})\}$, pela ordenação de S_A e S_B de acordo com a ordem decrescente dos valores de saliência;

for all $i \in [1, n]$ **do**

for all $j \in [1, m]$ **do**

if $|s'_{A_i} - s'_{B_j}| \leq 0.2s'_{A_1}$ **then**

 adicionar par $((u'_{A_i}, s'_{A_i}), (u'_{B_j}, s'_{B_j}))$ à lista L ;

end if

if $|s'_{B_j} - s'_{A_i}| \leq 0.2s'_{B_1}$ **then**

 adicionar par $((u'_{B_j}, s'_{B_j}), (u'_{A_i}, s'_{A_i}))$ à lista L ;

end if

end for

end for

for all par candidato $P_{ij} = ((u'_{A_i}, s'_{A_i}), (u'_{B_j}, s'_{B_j})) \in L$ **do**

 achar o parâmetro de deslocamento α , sendo $\alpha = u'_{A_i} - u'_{B_j}$;

 deslocar os pontos em S_A por α , formando o vetor $S''_A = \{(u''_{A_1}, s''_{A_1}), \dots, (u''_{A_n}, s''_{A_n})\}$;

$$d_{ij} \leftarrow \sum_{k=1}^{\min(n,m)} d_k, \text{ onde } d_k = \begin{cases} \sqrt{(u''_{A_k} - u_{B_k})^2 + (s''_{A_k} - s_{B_k})^2}, & \text{se } |u''_{A_k} - u_{B_k}| \leq 0.2, \\ s''_{A_k} + s_{B_k}, & \text{caso contrário;} \end{cases}$$

if $n \neq m$ **then**

$d_{ij} \leftarrow d_{ij} + s$, onde s é a altura dos pontos não casados;

end if

end for

repetir o laço anterior considerando o par candidato na forma $P_{ij} = ((u'_{B_j}, s'_{B_j}), (u'_{A_i}, s'_{A_i})) \in L$;

$d \leftarrow \min(d_{ij})$;

5.3 Experimentos

Nesta seção, apresentam-se os resultados dos experimentos executados para avaliar o detector de saliências proposto, bem como o descritor de forma baseado nas saliências detectadas.

5.3.1 Base de imagens

Para os experimentos, duas bases de imagens (MPEG-7 CE-shape-1 part B [2] e *Fish-shape* [1]) foram utilizadas parcial ou inteiramente.

A base do MPEG-7 foi descrita no Capítulo 4. Já a base *Fish-shape* original é formada por mil e cem imagens de silhuetas de peixes. As classes foram construídas por dez variações de cada imagem original com rotação e escala. Assim, a base inteira usada nos experimentos possui mil e cem classes com dez imagens cada.

5.3.2 Resultados

Os experimentos foram realizados em duas etapas. A primeira objetiva avaliar a qualidade dos pontos de saliência detectados. A segunda visa avaliar o descritor de forma proposto, comparando-o com o trabalho apresentado em [22]. Neste trabalho, dois descritores baseados em saliência são propostos: *Contour Saliences* (CS) – que utiliza os pontos de saliência do contorno – e *Segment Saliences* (SS) – que utiliza as saliências de segmentos do contorno. Como o descritor proposto também utiliza saliências do contorno, será comparado apenas com o descritor CS.

Para simplificação, o método de detecção de saliências utilizado no CS será referenciado como método baseado em esqueleto, pois utiliza-se dos esqueletos multiescala, interno e externo, para detectar as saliências. Já o detector proposto neste trabalho será referenciado como método baseado em *Tensor Scale*, ou método baseado em TS.

Detector de saliências

A primeira consideração a ser feita na comparação dos métodos de detecção é referente à granularidade. O método baseado em TS é executado localmente, realizando cálculos para cada orientação mapeada e seus vizinhos, ao longo do contorno. O método baseado em esqueleto é mais global, pois utiliza os esqueletos interno e externo de toda a forma, para a detecção das saliências. Essa diferença na granularidade resulta em uma detecção menos robusta por parte do método baseado em esqueleto, pois um limiar é aplicado aos esqueletos multiescala para a obtenção dos pontos de saliência. Este limiar representa uma suavização no contorno e, conseqüentemente, a perda de algumas saliências importantes. O método baseado em TS também é dependente de limiar, porém é muito menos complexo determinar um bom limiar geral para

uma base dados, que é o caso do método baseado em TS, do que determinar um bom limiar para cada imagem da base separadamente, que é o caso do método baseado em esqueleto.

A Figura 5.7 mostra a diferença de granularidade entre os dois métodos. Enquanto o método baseado em esqueleto tem uma granularidade mais alta, detectando apenas as doze saliências globais presentes no contorno do objeto, o método baseado em TS detectou mais diferenças abruptas que existem ao longo do contorno.

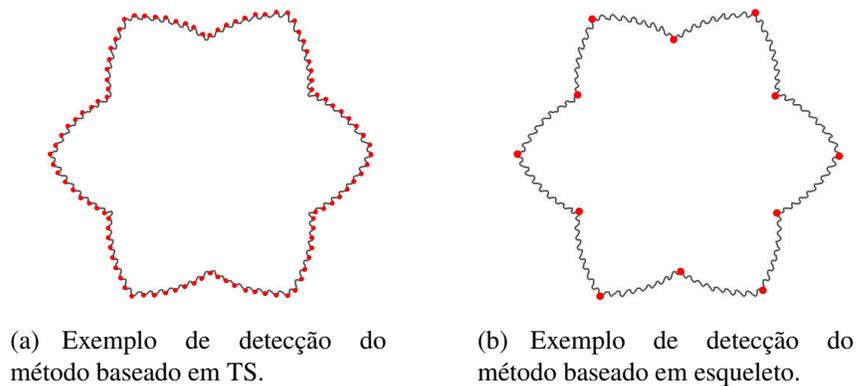


Figura 5.7: Granularidade dos métodos baseados em esqueleto e em *Tensor Scale*.

Além da análise da granularidade, os detectores de pontos de saliência foram testados quanto à sua eficiência e quanto à sua eficácia. Os primeiros experimentos são referentes à comparação da eficiência dos métodos de detecção. Para tanto, mediu-se o tempo de execução de ambos os métodos para todas as imagens da base *Fish-shape*.

O método baseado em TS apresentou tempo de execução duas vezes mais rápido do que o baseado em esqueleto, significando um ganho (*speedup*) de aproximadamente 2.04, em média. Estes experimentos foram realizados em máquina com processador AMD 64 3000+, com 1 GB de memória RAM.

Para análise da eficácia, uma base foi construída com 42 formas da base *Fish-shape* e 112 formas da base MPEG-7 CE-shape-1 part B, resultando em 2835 saliências. As imagens foram escolhidas levando-se em consideração a obviedade da localização dos pontos de saliência, para que nenhum método fosse privilegiado. A partir disso, um conjunto de imagens com as saliências verdadeiras (imagens de *ground-truth*) foi formado, contendo a localização de todas as saliências que os métodos deveriam encontrar. A Figura 5.8 mostra exemplos de duas imagens de *ground-truth*.

O experimento consiste na contagem de saliências verdadeiro-positivas (T_+) e de saliências falso-positivas (F_+) detectadas pelos métodos, em comparação com as imagens de *ground-truth*.

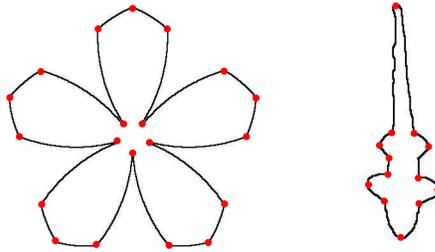


Figura 5.8: Exemplos de imagens de *ground-truth*.

Após esta contagem, as medidas de eficácia – revocação, precisão e acurácia – são calculadas:

$$\text{Revocação} = \frac{T_+}{T_+ + T_-}; \quad (5.2)$$

$$\text{Precisão} = \frac{T_+}{T_+ + F_+}; \quad (5.3)$$

$$\text{Acurácia} = \frac{T_+ + T_-}{T_+ + T_- + F_+ + F_-}; \quad (5.4)$$

onde T_- é o número de verdadeiro-negativos, F_- é o número de falso-negativos e $T_+ + T_-$ representa o número total de pontos (de saliência e não-saliência).

Os resultados obtidos para vários limiares do método baseado em TS são apresentados na Tabela 5.1. O limiar para o método baseado em esqueleto foi fixado em 5%, que é o valor recomendado em [22].

Medidas	Esqueleto	TS 10 (%)	TS 12 (%)	TS 14 (%)	TS 16 (%)	TS 18 (%)
Revocação	0.956	0.964 (0.84)	0.964 (0.84)	0.963 (0.73)	0.963 (0.73)	0.962 (0.63)
Precisão	0.903	0.889 (-1.55)	0.923 (2.21)	0.946 (4.76)	0.963 (6.64)	0.968 (7.20)
Acurácia	0.718	0.840 (17.00)	0.862 (20.06)	0.874 (21.73)	0.875 (21.87)	0.867 (20.75)

Tabela 5.1: Medidas de eficácia para os métodos de esqueleto e TS. Para cada valor de limiar testado no método TS, a tabela apresenta o valor absoluto das medidas e, em parêntesis, a porcentagem de ganho em relação ao método baseado em esqueleto.

O método baseado em TS é mais acurado que o método baseado em esqueleto, para todos os valores de limiar testados. No método baseado em TS, a acurácia foi maximizada com o limiar 16 e este valor foi adotado nos experimentos descritos na próxima seção.

Descritor de forma

Ambos os descritores (CS e TSCS) foram computados para a base de imagens *Fish-shape* e as curvas de separabilidade obtidas são apresentadas na Figura 5.9. Apenas essa medida foi utilizada para os experimentos comparativos entre os descritores, pois os experimentos originais realizados com o CS, em [22], utilizaram apenas separabilidade multiescala.

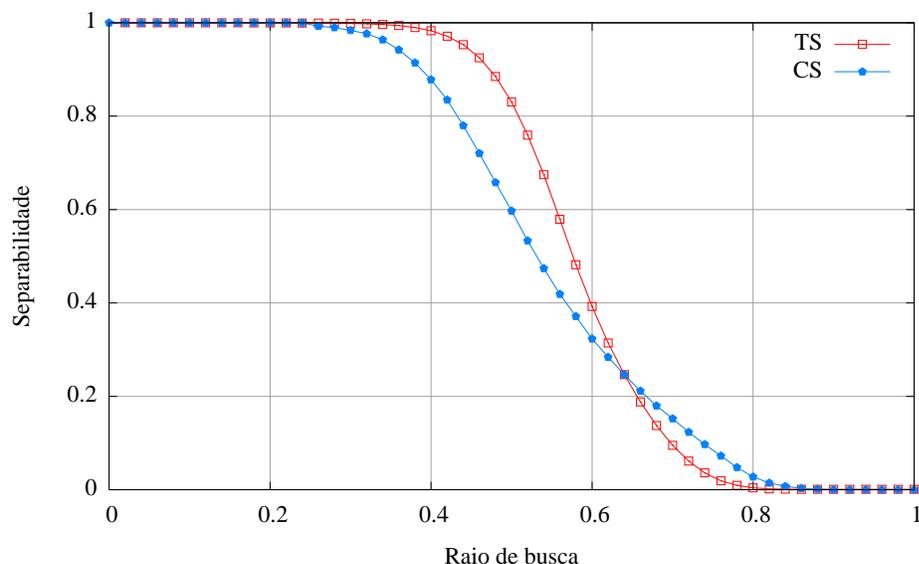


Figura 5.9: Curva de separabilidade multiescala para a base de imagens *Fish-shape*.

Os descritores TSCS e CS possuem performance equivalente para raios de busca inferiores a 25% da máxima distância. Deste ponto até 65% da distância máxima, o TSCS se apresenta mais robusto e eficaz do que o CS. Pela análise do gráfico da Figura 5.9, observamos que o TS é igual ou mais eficaz do que o CS em 80% dos raios de busca.

5.4 Resumo

Este capítulo apresentou um detector de saliências do contorno baseado em *Tensor Scale*. Para este propósito, utiliza as diferenças entre orientações *Tensor Scale* mapeadas para o contorno do objeto. Os resultados experimentais mostram que este método é mais rápido do que o detector de saliências proposto em [22] e é mais robusto, considerando-se uma base de imagens constituída por formas das bases *Fish-shape* e MPEG-7 part B.

Também foi apresentado descritor de forma baseado nas saliências detectadas. Os experimentos feitos para a base de imagens *Fish-shape* indicam que este novo método é mais ou igualmente eficaz ao descritor CS em 80% dos raios de busca, considerando-se a medida de separabilidade.

O descritor de forma baseado nas saliências detectadas foi projetado a título de comparação com o descritor CS. Ou seja, ao considerar as mesmas funções de extração do vetor e de similaridade do CS, os experimentos relativos aos descritores comprovaram a qualidade dos pontos de saliências obtidos pelos métodos, já que a única diferença entre eles reside exatamente na fase de detecção destes pontos.

Capítulo 6

Conclusões

Neste trabalho, foram explorados descritores de forma e sua utilização em sistemas de recuperação de imagens por conteúdo. Este estudo se justifica, pois descritores de forma são necessários em vários tipos de aplicação atuais, como sistemas de segurança, sistemas de biodiversidade, análise de componentes na engenharia, sistemas de informação geográfica, sistemas de diagnósticos médicos, entre outros. Neste sentido, este trabalho propôs a utilização de parâmetros morfométricos, chamados *Tensor Scale*, para a descrição da forma de objetos.

O resultado final foi a especificação e implementação de dois descritores de forma baseados em *Tensor Scale*, a partir do estudo de outros descritores de forma e de algoritmos eficazes para o cálculo do *Tensor Scale*.

As principais contribuições deste trabalho são:

- Estudo de descritores de textura e cor e, mais extensivamente, descritores de forma;
- Estudo de algoritmos para o cálculo do *Tensor Scale*;
- Proposta, implementação e validação de detector de saliências de contorno baseado em *Tensor Scale*;
- Proposta e implementação do descritor Saliências do Contorno por *Tensor Scale* (TSCS);
- Proposta e implementação do descritor baseado em *Tensor Scale* por zonas de influência (TSDIZ);
- Validação e comparação, segundo diversas medidas, dos descritores propostos com outros descritores de forma relevantes.

Os resultados obtidos nos Capítulos 4 e 5 foram submetidos para conferências internacionais [4, 5]; e um artigo com as contribuições deste trabalho será submetido para uma revista.

Para trabalhos futuros, sugerem-se alguns tópicos relacionados à implementação dos descritores, à validação dos descritores e às possíveis extensões dos métodos apresentados. Os trabalhos nestas três frentes são detalhados a seguir.

Implementação

- Estudo de método automático para a detecção do melhor limiar a ser utilizado no descritor TSCS. Há vários trabalhos na literatura que apresentam estudos referentes à detecção automática de limiares para imagens em níveis de cinza. Sendo assim, pode-se entender a curva de diferença gerada pelo TSCS como um histograma no qual se deseja separar o que é fundo e o que é objeto (ou, no caso do TSCS, o que é e o que não é ponto de saliência). Fazendo isso, pode-se aplicar qualquer algoritmo de detecção automática de limiar, como o algoritmo de Otsu [56].
- Estudo para viabilizar redução de complexidade de espaço no armazenamento de vetores de características do TSDIZ. O melhor resultado obtido com o descritor TSDIZ foi com a divisão do contorno do objeto em 60 segmentos. Para cada segmento, existe uma orientação associada, resultando em um espaço de armazenamento de 240 *bytes*. Um estudo deve ser feito para diminuir este espaço de armazenamento, identificando eventuais redundâncias que podem estar sendo armazenadas no vetor de características.
- Estudo de algoritmos simplificados para o cálculo do *Tensor Scale*. Para sistemas que exigem menor tempo de computação em detrimento do resultado, heurísticas podem ser usadas para a determinação das elipses. Um método sugerido é utilizar o menor caminho de um *pixel* até o contorno como o semi-eixo menor da elipse. A partir desta escolha, o semi-eixo maior é determinado pela reta que forma 90° com o semi-eixo menor da elipse. Nem sempre a maior elipse centrada no *pixel* é encontrada, porém, utilizando este método, encontra-se uma aproximação da elipse com apenas um cálculo da TDE.

Validação

- Utilização de mais descritores de forma na realização do estudo comparativo. Alguns descritores muito utilizados, como o *Curvature Scale Space* [3], devem ser incorporados aos testes comparativos. Para a realização deste estudo, também é desejável que outras bases de imagens sejam testadas.
- Realização de testes específicos para avaliar o comportamento dos descritores mediante a presença de transformações geométricas. O experimento MPEG-7 CE-shape-1 part B, por exemplo, realiza etapas específicas para testar quão robusto o descritor é quanto à rotação, escala e deformações não-rígidas e de movimento [40].

Extensões

- Extensão do algoritmo do cálculo do *Tensor Scale* via IFT para imagens em níveis de cinza. Uma forma de realizar tal extensão é considerar cada camada (nível de cinza) da imagem como uma imagem binária e aplicar o mesmo algoritmo. Utilizando-se este raciocínio, o cálculo do *Tensor Scale* também pode ser estendido para imagens coloridas.
- Extensão do algoritmo de cálculo do *Tensor Scale* via IFT para formas tridimensionais. Neste caso, o *Tensor Scale* seria representado por elipsóides e não por elipses.

Referências Bibliográficas

- [1] Fish-shape database. <http://www.ee.surrey.ac.uk/research/vssp/imagedb/demo.html>, January 2007.
- [2] The MPEG project. <http://www.chiariglione.org/mpeg>, January 2007.
- [3] S. Abbasi, F. Mokhtarian, and J. V. Kittler. Enhancing CSS-based shape retrieval for objects with shallow concavities. *Image and Vision Computing*, 18(3):199–211, February 2000.
- [4] F. A. Andaló, P. A. V. Miranda, R. da S. Torres, and A. X. Falcão. Detecting contour saliencies using Tensor Scale. In *Proceedings of IEEE International Conference on Image Processing*, 2007. Submitted.
- [5] F. A. Andaló, P. A. V. Miranda, R. da S. Torres, and A. X. Falcão. A new shape descriptor based on Tensor Scale. In *Proceedings of the International Symposium on Mathematical Morphology*, 2007. Submitted.
- [6] N. Arica and F. T. Y. Vural. BAS: a perceptual shape descriptor based on the beam angle statistics. *Pattern Recognition Letters*, 24(9-10):1627–1639, June 2003.
- [7] F. Attneave. Some informational aspects of visual perception. *Psychological Review*, 61(3):183–193, 1954.
- [8] R. Baeza-Yates and B. Ribeiro-Neto. *Modern information retrieval*. Addison-Wesley Longman Publishing Co. Inc., Boston, MA, USA, 1999.
- [9] M. Bober. MPEG-7 visual shape descriptors. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(6):716–719, June 2001.
- [10] A. C. Bovik, M. Clark, and W. S. Geisler. Multichannel texture analysis using localized spatial filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):55–73, January 1990.

- [11] Jr. C. Traina, A. Traina, C. Faloutsos, and B. Seeger. Fast indexing and visualization of metric datasets using slim-trees. *IEEE Transactions on Knowledge and Data Engineering*, 14(2):244–260, March 2002.
- [12] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: image segmentation using expectation-maximization and its application to image querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1026–1038, August 2002.
- [13] N. S. Chang and K. S. Fu. Query-by-pictorial-example. *IEEE Transactions on Software Engineering*, 6(6):519–524, 1980.
- [14] C. A. Christopoulos, D. Berg, and A. N. Skodras. The colour in the upcoming MPEG-7 standard. In *Proceedings of X European Signal Processing Conference - EUSIPCO-2000*, pages 1369–1372, September 2000.
- [15] P. Ciaccia, M. Patella, and P. Zezula. M-Tree: an efficient access method for similarity search in metric spaces. In *Proceedings of 23rd International Conference on Very Large Data Bases*, pages 426–435, 1997.
- [16] I. J. Cox, M. L. Miller, T. P. Minka, T. V. Papatomas, and P. N. Yianilos. The bayesian image retrieval system, PicHunter: theory, implementation, and psychophysical experiments. *IEEE Transactions on Image Processing*, 9(1):20–37, January 2000.
- [17] B. S. da Cunha. Projeto de Operadores de Processamento e Análise de Imagens Baseados na Transformada Imagem-Floresta. Master’s thesis, Institute of Computing, Unicamp, 2001.
- [18] L. da F. Costa, A. G. Campos, and E. T. M. Manoel. An integrated approach to shape analysis: results and perspectives. In *Proceedings of the International Conference on Quality Control by Artificial Vision - QCAV2001*, volume 1, pages 23–24, May 2001.
- [19] L. da F. Costa and Jr. R. M. Cesar. *Shape analysis and classification: theory and practice*. CRC Press, Florida, USA, 2001.
- [20] D. da S. Andrade. Testes de significância estatísticos e avaliação de um modelo de recuperação de imagens por conteúdo. Master’s thesis, Institute of Computing, Unicamp, 2004.
- [21] R. da S. Torres and A. X. Falcão. Content-based image retrieval: theory and applications. *Revista de Informática Teórica e Aplicada*, 13(2):161–185, 2006.
- [22] R. da S. Torres and A. X. Falcão. Contour salience descriptors for effective image retrieval and analysis. *Image and Vision Computing*, 25(1):3–13, January 2007.

- [23] R. da S. Torres, A. X. Falcão, and L. da F. Costa. A graph-based approach for multiscale shape analysis. *Pattern Recognition*, 37(6):1163–1174, June 2004.
- [24] R. da S. Torres, C. B. Medeiros, M. A. Gonçalves, and E. A. Fox. A digital library framework for biodiversity information systems. *International Journal on Digital Libraries*, 6(1):3–17, February 2006.
- [25] A. del Bimbo. *Visual information retrieval*. Morgan Kaufmann Publishers, San Francisco, CA, USA, 1999.
- [26] L. M. del V. Cura. *Content-based image retrieval in geographical information systems*. PhD thesis, Institute of Computing, Unicamp, 2000.
- [27] S. A. Dudani, K. J. Breeding, and R. B. McGhee. Aircraft identification by moment invariants. *IEEE Transactions on Computers*, 26(1):39–45, 1977.
- [28] J. H. Elder and S. W. Zucker. Local scale control for edge detection and blur estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(7):699–716, July 1998.
- [29] A. X. Falcão. Introdução ao processamento de imagem digital. <http://www.ic.unicamp.br/~afalcao>, January 2007.
- [30] A. X. Falcão, J. Stolfi, and R. A. Lotufo. The image foresting transform: Theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):19–29, January 2004.
- [31] M. Ferraro, G. Boccignone, and T. Caelli. On the representation of image structures via scale space entropy conditions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(11):1199–1203, November 1999.
- [32] N. I. Fisher. *Statistical analysis of circular data*. Cambridge University Press, Cambridge, UK, 1993.
- [33] M. Flickner, H. Sawhney, W. Niblack, Q. Huang, J. Ashley, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: the QBIC system. *IEEE Computer*, 28(9):23–32, September 1995.
- [34] R. C. Gonzalez and R. E. Woods. *Digital image processing*. Electrical and Computer Engineering Series. Addison-Wesley Longman Publishing Co. Inc., 2nd edition, November 2001.

- [35] V. N. Gudivada and V. V. Raghavan. Content-based image retrieval systems. *IEEE Computer*, 28(9):18–22, September 1995.
- [36] E. L. Hall. *Computer image processing and recognition*. Computer Science and Applied Mathematics. Academic Press, New York, NY, 1979.
- [37] M. K. Hu. Visual pattern recognition by moment invariants. *IEEE Transactions on Information Theory*, 8(2):179–187, 1962.
- [38] P. M. Kelly, M. Cannon, and D. R. Hush. Query by image example: the CANDID approach. In *SPIE Storage and Retrieval for Image and Video Databases III*, pages 238–248, 1995.
- [39] J. J. Koenderink. The structure of images. *Biological Cybernetics*, 50(5):363–370, 1984.
- [40] L. J. Latecki, R. Lakämper, and U. Eckhardt. Shape descriptors for non-rigid shapes with a single closed contour. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 424–429, June 2000.
- [41] G. M. G. Leal. Análise de resíduos em modelos de regressão von Mises. Master’s thesis, Centro de Ciências e Tecnologia - UFCG, April 2006.
- [42] Y. Leung, J. S. Zhang, and Z. B. Xu. Clustering by scale-space filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1396–1410, December 2000.
- [43] M. Leyton. Symmetry-curvature duality. *Computer Vision, Graphics, and Image Processing*, 38(3):327–341, 1987.
- [44] S. Loncaric. A survey of shape analysis techniques. *Pattern Recognition*, 31(8):983–1001, August 1998.
- [45] B. C. Lovell and A. P. Bradley. The multiscale classifier. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(2):124–137, February 1996.
- [46] W. Y. Ma and B. S. Manjunath. Netra: A toolbox for navigating large image databases. In *IEEE International Conference on Image Processing*, pages 256–268, 1997.
- [47] B. S. Manjunath, J. R. Ohm, V. V. Vasudevan, and A. Yamada. Color and texture descriptors. *IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on MPEG-7*, 11(6):703–715, June 2001.
- [48] K. V. Mardia and P. E. Jupp. *Directional statistics*. Wiley Series in Probability and Statistics. John Wiley and Sons, November 1999.

- [49] D. Marr and E. Hildreth. Theory of edge detection. In *Proceedings of Royal Society London*, 207, pages 187–217, 1980.
- [50] B. M. Mehre, M. S. Kankanhalli, and W. F. Lee. Shape measures for content based image retrieval: a comparison. *Information Processing and Management: an International Journal*, 33(3):319–337, May 1997.
- [51] D. S. Messing, P. van Beek, and J. H. Errico. The MPEG-7 colour structure descriptor: image description using colour and local spatial information. In *Proceedings of International Conference on Image Processing*, volume 1, pages 670–673, 2001.
- [52] P. A. V. Miranda, R. da S. Torres, and A. X. Falcão. TSD: a shape descriptor based on a distribution of tensor scale local orientation. In *Proceedings of XVIII Brazilian Symposium on Computer Graphics and Image Processing*, pages 139–146, October 2005.
- [53] F. Mokhtarian and R. Suomela. Robust image corner detection through curvature scale space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1376–1381, December 1998.
- [54] H. Müller, N. Michoux, D. Bandon, and A. Geissbuhler. A review of content-based image retrieval systems in medical applications - clinical benefits and future directions. *International Journal of Medical Informatics*, 73(1):1–23, February 2004.
- [55] H. Müller, W. Müller, D. McG. Squire, S. Marchand-Maillet, and T. Pun. Performance evaluation in content-based image retrieval: overview and proposals. *Pattern Recognition Letters*, 22(5):593–601, April 2001.
- [56] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, January 1979.
- [57] D. K. Park, Y. S. Jeon, and C. S. Won. Efficient use of local edge histogram descriptor. In *Proceedings of the 2000 ACM workshops on Multimedia - MULTIMEDIA '00*, pages 51–54, 2000.
- [58] A. Pentland, R. W. Picard, and S. Sclaroff. Photobook: tools for content-based manipulation of image databases. *International Journal of Computer Vision*, 18(3):233–254, 1996.
- [59] J. Richefeu. Morphological dominant points detection for motion analysis on programmable retina. In *Proceedings of the 7th International Symposium on Signal Processing and its Applications*, volume 2, pages 627–628, July 2003.

- [60] Y. M. Ro, M. Kim, H. K. Kang, B. S. Manjunath, and J. Kim. MPEG-7 homogeneous texture descriptor. *Electronics and Telecommunications Research Institute (ETRI) Journal*, 23(2):41–51, June 2001.
- [61] P. K. Saha. Tensor Scale: A local morphometric parameter with applications to computer vision and image processing. *Computer Vision and Image Understanding*, 99(3):384–413, September 2005.
- [62] P. K. Saha, J. K. Udupa, and D. Odhner. Scale-based fuzzy connected image segmentation: theory, algorithms, and validation. *Computer Vision and Image Understanding*, 77(2):145–174, 2000.
- [63] M. Shearer and J. J. Zou. Detection of dominant points based on noise suppression and error minimisation. In *ICITA '05: Proceedings of the Third International Conference on Information Technology and Applications*, volume 2, pages 772–775, July 2005.
- [64] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, December 2000.
- [65] R. O. Stehling, M. A. Nascimento, and A. X. Falcão. A compact and efficient image retrieval approach based on border/interior pixel classification. In *CIKM '02: Proceedings of the eleventh international conference on information and knowledge management*, pages 102–109, 2002.
- [66] M. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, November 1991.
- [67] M. Tabb and N. Ahuja. Multiscale image segmentation by integrated edge and region detection. *IEEE Transactions on Image Processing*, 6(5):642–655, May 1997.
- [68] A. Tofilski. DrawWing, a program for numerical description of insect wings. *Journal of Insect Science*, 4(17):1–5, May 2004.
- [69] R. C. Veltkamp and M. Tanase. Content-based image retrieval systems: a survey. Technical Report revised and extended version of UU-CS-2000-34, Department of Computing Science, Utrecht University, October 2002.
- [70] K. L. Vincken, A. S. E. Koster, and M. A. Viergever. Probabilistic multiscale image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):109–120, February 1997.

- [71] J. Vleugels and R. C. Veltkamp. Efficient image retrieval through vantage objects. *Pattern Recognition*, 35(1):69–80, January 2002.
- [72] Y. P. Wang and T. Pavlidis. Optimal Correspondence of String Subsequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(11):1080–1087, Dec 1990.
- [73] P. J. D. Weeks, M. A. O’Neill, K. J. Gaston, and I. D. Gauld. Automating insect identification: exploring the limitations of a prototype system. *Journal of Applied Entomology*, 123(1):1–8, 1999.
- [74] A. P. Witkin. Scale-space filtering. In *Proceedings of 8th International Joint Conference on Artificial Intelligence*, pages 1019–1022, 1983.
- [75] P. Wu, B. S. Manjunath, S. D. Newsam, and H. D. Shin. A texture descriptor for image retrieval and browsing. In *CBAIVL ’99: Proceedings of the IEEE Workshop on Content-Based Access of Image and Video Libraries*, pages 3–7, June 1999.
- [76] T. Zaharia, F. Preteux, and M. Preda. 3d shape spectrum descriptor. Technical Report ISO/IEC MPEG/M5242, Melbourne, Australia, October 1999.
- [77] D. Zhang and G. Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37(1):1–19, January 2004.