Instituto de Computação
Universidade Estadual de Campinas

Recuperação de Vídeos Comprimidos por Conteúdo

Jurandy Gomes de Almeida Junior

Este exemplar corresponde à redação final da Tese devidamente corrigida e defendida por Jurandy Gomes de Almeida Junior e aprovada pela Banca Examinadora.

Campinas, 28 de dezembro de 2011.

Prof. Dr. Ricardo da Silva Torres
(Orientador)

Prof. Dr. Neucimar Jerônimo Leite
(Coorientador)

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

ERRATA:

A informação de cabeçalho no anverso dessa folha é "*28 de novembro de 2011*", ao invés de "*28 de dezembro de 2011*".

CPG/IC, 11/04/2012

Prof. Dr. Paulo Licio de Geus
Coord. de Pós-Graduação
Instituto de Computação - Unicamp
Matrícula 10.326-8

FICHA CATALOGRÁFICA ELABORADA POR ANA REGINA MACHADO – CRB8/5467
BIBLIOTECA DO INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E COMPUTAÇÃO CIENTÍFICA – UNICAMP

AL64r

Almeida Junior, Jurandy Gomes de, 1983-
Recuperação de vídeos comprimidos por conteúdo / Jurandy Gomes de Almeida Junior. – Campinas, SP : [s.n.], 2011.

Orientador: Ricardo da Silva Torres.
Coorientador: Neucimar Jerônimo Leite.
Tese (doutorado) - Universidade Estadual de Campinas, Instituto de Computação.

1. Processamento de imagens. 2. Banco de dados. 3. Recuperação da informação. 4. Sistemas multimidia. 5. Videodigital. 6. MPEG (Padrão de codificação de vídeo). I. Torres, Ricardo da Silva, 1977-. II. Leite, Neucimar Jerônimo, 1961-. III. Universidade Estadual de Campinas. Instituto de Computação. IV. Título.

Unidade BCCL
T/UNICAMP
Cutter AL64r
V. ____ Ed 94782
Tombo BC
Proc. 16-100-12
C ____ D 1
Preço R$ 11,00
Data 17/04/12
Cód. tit. 846099

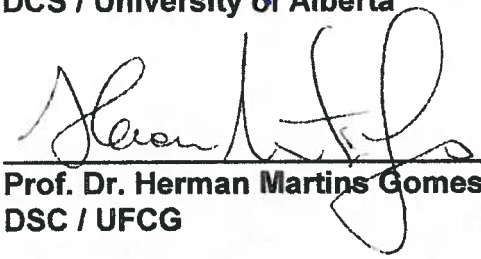
Informações para Biblioteca Digital

Título em inglês: Content-based retrieval of compressed videos
Palavras-chave em inglês:
Image processing
Databases
Information retrieval
Multimedia systems
Digital video
MPEG (Video coding standard)
Área de concentração: Ciência da Computação
Titulação: Doutor em Ciência da Computação
Banca examinadora:
Ricardo da Silva Torres [Orientador]
Mário Antonio do Nascimento
Herman Martins Gomes
Hélio Pedrini
Edson Borin
Data da defesa: 28-11-2011
Programa de Pós-Graduação: Ciência da Computação

TERMO DE APROVAÇÃO

Tese Defendida e Aprovada em 28 de novembro de 2011, pela Banca examinadora composta pelos Professores Doutores:

Prof. Dr. Mario Antonio do Nascimento
DCS / University of Alberta



Prof. Dr. Herman Martins Gomes
DSC / UFCG

Prof. Dr. Hélio Pedrini
IC / UNICAMP

Prof. Dr. Edson Borin
IC / UNICAMP

Prof. Dr. Ricardo da Silva Torres
IC / UNICAMP

Instituto de Computação
Universidade Estadual de Campinas

Recuperação de Vídeos Comprimidos por Conteúdo

Jurandy Gomes de Almeida Junior[1]

14 de outubro de 2011

Banca Examinadora:

- Prof. Dr. Ricardo da Silva Torres
  (Orientador)
- Prof. Dr. Mário Antonio do Nascimento
  Dept. of Computing Science – University of Alberta
- Prof. Dr. Herman Martins Gomes
  DSC – UFCG
- Prof. Dr. Hélio Pedrini
  IC – UNICAMP
- Prof. Dr. Edson Borin
  IC – UNICAMP
- Prof. Dr. Arnaldo de Albuquerque Araújo
  DCC – UFMG (Suplente)
- Profa. Dra. Anamaria Gomide
  IC – UNICAMP (Suplente)
- Prof. Dr. William Robson Schwartz
  IC – UNICAMP (Suplente)

[1]Financiado pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), processo 01P-05866/2007, no período de outubro de 2007 a setembro de 2008; Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processo 2008/50837-6, no período de outubro de 2008 a setembro de 2011; e Fundo de Apoio ao Ensino, à Pesquisa e à Extensão (FAEPEX), processo 1005/11, no período de outubro a novembro de 2011.

À memória da minha bisavó
Luzia e do seu irmão Mauro.

Agradecimentos

Agradeço,
a Deus, por todas as oportunidades que Ele me oferece.
ao meu orientador e co-orientador, Prof. Dr. Ricardo da Silva Torres e Prof. Dr. Neucimar Jerônimo Leite, pela oportunidade e orientação, cujos conselhos e sugestões foram muito valiosos tanto para este trabalho quanto para minha vida.
a toda minha família, em especial a minha bisavó Luzia (*in memorian*) e ao seu irmão Mauro (*in memorian*), e avós, Anésio (*in memorian*), Virgínia (*in memorian*), João (*in memorian*) e Ordália, por me ensinarem a valorizar tudo que a vida nos oferece.
aos meus pais Jurandy e Sônia, pela formação como ser humano, pelo incentivo constante, pelo apoio irrestrito e por seus valiosos conselhos.
à minha namorada, Noemi, pela dedicação, companheirismo, paciência e carinho.
ao meu irmão Tiago, pelos inúmeros conselhos, correções e sugestões.
ao meu irmão Marcos, pelo apoio.
aos meus companheiros do LIV e RECOD, pelas frutíferas discussões.
aos demais colegas de pós-graduação, pela ótima convivência.
aos professores e funcionários do IC, pelos serviços prestados.
à UNICAMP, pela infra-estrutura.
à CAPES, FAPESP e FAEPEX, pelo apoio financeiro.
a todos que de alguma forma contribuíram com o meu progresso.

Resumo

Avanços recentes na tecnologia têm permitido o aumento da disponibilidade de dados de vídeo, criando grandes coleções de vídeo digital. Isso tem despertado grande interesse em sistemas capazes de gerenciar esses dados de forma eficiente.

Fazer uso eficiente de informações de vídeo requer o desenvolvimento de ferramentas poderosas capazes de extrair representações semânticas de alto nível a partir de características de baixo nível do conteúdo de vídeo. Devido à complexidade desse material, existem cinco desafios principais na concepção de tais sistemas: (1) dividir o fluxo de vídeo em trechos manuseáveis de acordo com a sua estrutura de organização, (2) implementar algoritmos para codificar as propriedades de baixo nível de um trecho de vídeo em vetores de características, (3) desenvolver medidas de similaridade para comparar esses trechos a partir de seus vetores, (4) responder rapidamente a consultas por similaridade sobre uma enorme quantidade de sequências de vídeo e (5) apresentar os resultados de forma amigável a um usuário.

Inúmeras técnicas têm sido propostas para atender a tais requisitos. A maioria dos trabalhos existentes envolve algoritmos e métodos computacionalmente custosos, em termos tanto de tempo quanto de espaço, limitando a sua aplicação apenas ao ambiente acadêmico e/ou a grandes empresas.

Contrário a essa tendência, o mercado tem mostrado uma crescente demanda por dispositivos móveis e embutidos. Nesse cenário, é imperativo o desenvolvimento de técnicas tanto eficazes quanto eficientes a fim de permitir que um público maior tenha acesso a tecnologias modernas.

Nesse contexto, este trabalho apresenta cinco abordagens originais voltadas a análise, indexação e recuperação de vídeos digitais. Todas essas contribuições são somadas na construção de um sistema de gestão de vídeos por conteúdo computacionalmente rápido, capaz de atingir a um padrão de qualidade similar, ou até mesmo superior, a soluções atuais.

Abstract

Recent advances in the technology have enabled the increase of the availability of video data, creating large digital video collections. This has spurred great interest in systems that are able to manage those data in a efficient way.

Making efficient use of video information requires the development of powerful tools to extract high-level semantics from low-level features of the video content. Due to the complexity of the video material, there are five main challenges in designing such systems: (1) to divide the video stream into manageable segments according to its organization structure; (2) to implement algorithms for encoding the low-level features of each video segment into feature vectors; (3) to develop similarity measures for comparing these segments by using their feature vectors; (4) to quickly answer similarity queries over a huge amount of video sequences; and (5) to present the list of results in a user-friendly way.

Numerous techniques have been proposed to support such requirements. Most of existing works involve algorithms and methods which are computationally expensive, in terms of both time and space, limiting their application to the academic world and/or big companies.

Contrary to this trend, the market has shown a growing demand for mobile and embedded devices. In this scenario, it is imperative the development of techniques so effective as efficient in order to allow more people have access to modern technologies.

In this context, this work presents five novel approaches for the analysis, indexing, and retrieval of digital videos. All these contributions are combined to create a computationally fast system for content-based video management, which is able to achieve a quality level similar, or even superior, to current solutions.

Sumário

Agradecimentos **xi**

Resumo **xiii**

Abstract **xv**

1 Introdução **1**

2 Conceitos Básicos **7**
2.1 Imagem Digital 7
2.2 Espaços de Cor 8
2.3 Fluxo Ótico 10
2.4 Vídeo Digital 11
2.5 Operações de Câmera 12
2.6 Padrão de Codificação MPEG-1/2/4 13
2.7 Imagem DC 16
2.8 Espaço Métrico 17
2.9 Funções de Distância 18
2.10 Consultas por Similaridade 19
2.11 Medidas de Avaliação de Desempenho 21

3 Segmentação Temporal **25**
3.1 Revisão da Literatura 26
3.2 Abordagem Proposta 26
3.3 Protocolo Experimental 30
3.4 Resultados Experimentais 33
3.5 Conclusões 35

4 Representação do Conteúdo Visual **37**
4.1 Revisão da Literatura 38

4.2 Abordagem Proposta . . . 39
4.2.1 Extração de Características . . . 39
4.2.2 Ajuste do Modelo de Movimento . . . 40
4.2.3 Estimação Robusta dos Parâmetros . . . 42
4.3 Protocolo Experimental . . . 43
4.4 Resultados Experimentais . . . 46
4.5 Conclusões . . . 48

5 Medida de Similaridade 49
5.1 Revisão da Literatura . . . 50
5.2 Abordagem Proposta . . . 52
5.3 Protocolo Experimental . . . 53
5.4 Resultados Experimentais . . . 56
5.5 Conclusões . . . 59

6 Indexação de Dados 61
6.1 Revisão da Literatura . . . 62
6.2 Abordagem Proposta . . . 65
6.2.1 Visão Geral . . . 67
6.2.2 Criação do Índice . . . 70
6.2.3 Consultas por Similaridade . . . 72
6.2.4 Consultas por Abrangência . . . 73
6.2.5 Consultas por Vizinhos mais Próximos . . . 74
6.2.6 Atualização do Índice . . . 77
6.3 Avaliação Experimental . . . 77
6.3.1 Eficiência . . . 78
6.3.2 Escalabilidade . . . 91
6.3.3 Construção Incremental . . . 95
6.4 Conclusões . . . 98

7 Visualização dos Resultados 99
7.1 Revisão da Literatura . . . 100
7.2 Abordagem Proposta . . . 104
7.2.1 Extração de Características . . . 105
7.2.2 Seleção de Conteúdo . . . 105
7.2.3 Filtragem de Ruídos . . . 108
7.2.4 Personalização do Usuário . . . 110
7.3 Protocolo Experimental . . . 112
7.3.1 Avaliação de Resumos Estáticos . . . 112

7.3.2 Avaliação de Resumos Dinâmicos 114
7.4 Resultados Experimentais 115
7.4.1 Efeitos da Função de Distância 115
7.4.2 Escolha dos Limiares de Filtragem 116
7.4.3 Ajuste dos Parâmetros do Método 117
7.4.4 Desempenho em Resumos Estáticos 118
7.4.5 Desempenho em Resumos Dinâmicos 126
7.5 Conclusões 129

8 Conclusões e Perspectivas **131**

Lista de Tabelas

2.1 Tabela de contingência para avaliar a eficácia de recuperação. 21

3.1 As principais características de cada sequência de vídeo do conjunto de testes (adaptada de Bezerra e Leite [28]). 31
3.2 Comparação da precisão (P), revocação (R) e medida-F (F) obtida por abordagens diferentes para cada vídeo do conjunto de testes. 33

4.1 As principais características de cada sequência de vídeo do mundo real (R_i). 46
4.2 Eficácia (ZNCC) obtida por diferentes abordagens no vídeo M_1. 46
4.3 Eficácia (ZNCC) obtida por diferentes abordagens no vídeo M_2. 46
4.4 Eficácia (ZNCC) obtida por diferentes abordagens no vídeo M_3. 46
4.5 Eficácia (ZNCC) obtida por diferentes abordagens no vídeo M_4. 47
4.6 Eficácia (ZNCC) obtida por diferentes abordagens no vídeo R_1. 47
4.7 Eficácia (ZNCC) obtida por diferentes abordagens no vídeo R_2. 47
4.8 Eficácia (ZNCC) obtida por diferentes abordagens no vídeo R_3. 47
4.9 Eficácia (ZNCC) obtida por diferentes abordagens no vídeo R_4. 48

5.1 O custo computacional e os requisitos de espaço de diferentes abordagens. . 58

6.1 Sumário de símbolos e definições. 70
6.2 As principais características das bases de dados usadas nos experimentos. . 82

7.1 A média da medida-F obtida pela melhor combinação de parâmetros (ϵ e λ) usando diferentes funções de distância. 115
7.2 Diferença entre as médias da medida-F de diferentes funções de distância a um nível de confiança de 98%. 116
7.3 A média da medida-F obtida por diferentes abordagens em várias categorias de vídeo. 119
7.4 Diferença entre as médias da medida-F atingidas por diferentes abordagens a um nível de confiança de 98%. 119

7.5 A média da medida-F obtida por VSUMM e VISON em diferentes categorias de vídeo. 122
7.6 Diferença entre as médias da medida-F atingidas por VSUMM e VISON em diferentes categorias de vídeo, a um nível de confiança de 98%. 122
7.7 O custo computacional e os requisitos de espaço de diferentes abordagens. . 125
7.8 Diferenças entre a técnica apresentada e trabalhos anteriores a um nível de confiança de 95%. 127
7.9 O custo computacional e os requisitos de espaço de diferentes abordagens. . 129

Lista de Figuras

1.1 Arquitetura típica de um sistema de gestão de vídeos por conteúdo (adaptada de Torres e Falcão [161]). 2
1.2 Fluxograma de processamento de um sistema de gestão de vídeos por conteúdo. 3

2.1 Representação de uma imagem digital. 7
2.2 O espaço de cor RGB. 8
2.3 O espaço de cor HSV. 9
2.4 O espaço de cor CIE Lab. 10
2.5 Exemplos de uma correspondência ótica (adaptada de Minetto [114]). . . . 10
2.6 Representação de um fluxo ótico a partir da amostragem de um conjunto fixo de pontos, resultando em um conjunto de vetores, cuja direção e magnitude são representadas por um segmento de reta. 11
2.7 Estrutura hierárquica de uma sequência de vídeo. 12
2.8 As operações básicas de uma câmera (adaptada de Park *et al.* [127]). . . . 13
2.9 Estrutura típica de um vídeo MPEG (adaptada de Ghanbari [67]). 14
2.10 Estrutura típica de um GOP em vídeos MPEG (adaptada de Tan *et al.* [159]). 15
2.11 Imagem original com 352×240 pontos e sua imagem DC com 44×30 pontos. 17
2.12 Representação dos objetos situados à distância r de um objeto o (adaptada de Zezula *et al.* [174]). 18
2.13 Exemplo de consultas por similaridade utilizando a função de distância L_2 (adaptada de Zezula *et al.* [174]). 19
2.14 Distribuição da coleção de vídeos de acordo com uma escala binária. 21
2.15 Exemplo de espaço $P \times R$. 22
2.16 Exemplo de espaço ROC. 23

3.1 Diagrama de blocos do algoritmo de detecção de cortes. 27
3.2 Dissimilaridades entre pares de quadros adjacentes do vídeo I do conjunto de testes. 28

3.3 Seis quadros consecutivos de um GOP que inclui uma transição do vídeo A do conjunto de testes e o tipo de codificação de seus respectivos macroblocos: intracodificado (magenta) e intercodificado (azul). 29
3.4 Os comportamentos possíveis para os macroblocos dos quadros B. 29
3.5 Exemplo do efeito planalto em um GOP que inclui uma transição gradual. 30
3.6 A função densidade de probabilidade (PDF) para a distribuição do valor da similaridade entre diferentes tipos de quadro de vídeo. 32

4.1 O fluxo ótico real (acima) e o ideal (abaixo) gerados por panorâmica horizontal, panorâmica vertical, zoom e rotação, respectivamente (da esquerda para a direita). 42
4.2 Cenas do POV-Ray de um modelo realista de um escritório utilizado na criação do conjunto de teste sintético, com a câmera realizando uma panorâmica horizontal. 43
4.3 Função suave e cíclica utilizada para modelar o movimento da câmera. . . 44
4.4 As principais características de cada sequência de vídeo (M_i) do conjunto de teste sintético. 45

5.1 Uma visão geral do método proposto. 52
5.2 Exemplos de resumos do vídeo original e das suas cópias transformadas. . . 55
5.3 Curvas Precisão vs. Revocação. 56
5.4 As curvas ROC. 57

6.1 Efeitos de diferentes paradigmas de divisão do espaço em um conjunto de objetos com uma distribuição uniforme. 66
6.2 Efeitos do uso de partes de tamanho fixo na fragmentação de um conjunto de objetos com uma distribuição esparsa. 67
6.3 Exemplo de como a BP-tree se beneficia e combina as vantagens de ambas as abordagens disjuntas e não disjuntas. 69
6.4 Uma representação da BP-tree. 71
6.5 Limite inferior da distância $d(o_j, o_q)$ entre o objeto de consulta o_q e qualquer objeto o_j da subárvore $T(o_i)$ enraizada pelo objeto o_i usando o objeto de referência o_{ref}. 75
6.6 Histograma de distribuição de distâncias de cada base de dados. 81
6.7 A eficiência de busca (em termos do número médio de cálculos de distância) dos métodos comparados como uma função do raio de consulta (R-NN), em que cada gráfico se refere a uma das coleções. 83

6.8 A eficiência de busca (em termos do número médio de cálculos de distância) dos métodos comparados como uma função do número k de vizinhos mais próximos (k-NN) requisitados para a consulta, em que cada gráfico se refere a uma das coleções. 84

6.9 A eficiência de busca (em termos do número médio de acessos a disco) dos métodos comparados como uma função do raio de consulta (R-NN), em que cada gráfico se refere a uma das coleções. 86

6.10 A eficiência de busca (em termos do número médio de acessos a disco) dos métodos comparados como uma função do número k de vizinhos mais próximos (k-NN) requisitados para a consulta, em que cada gráfico se refere a uma das coleções. 87

6.11 A eficiência de busca (em termos do tempo médio em milissegundos) dos métodos comparados como uma função do raio de consulta (R-NN), em que cada gráfico se refere a uma das coleções. 89

6.12 A eficiência de busca (em termos do tempo médio em milissegundos) dos métodos comparados como uma função do número k de vizinhos mais próximos (k-NN) requisitados para a consulta, em que cada gráfico se refere a uma das coleções. 90

6.13 Histograma de distribuição de distâncias da coleção ETH-80. 91

6.14 Comparação entre o tempo de construção (em segundos) de diferentes índices em relação ao tamanho do conjunto de objetos. 92

6.15 Comparação entre a ocupação de espaço (em termos do número de páginas de disco) de diferentes índices em relação ao tamanho do conjunto de objetos. 92

6.16 A escalabilidade de diferentes métodos em relação ao número de objetos da base de dados. Esses gráficos apresentam resultados para consultas R-NN (coluna da esquerda) e k-NN (coluna da direita). Eles reportam o número médio de cálculos de distância (primeira linha), o número médio de acessos a disco (segunda linha) e o tempo médio (em milissegundos) para realizar uma consulta (terceira linha). 94

6.17 Comparação entre o tempo de construção (em segundos) das versões estática e dinâmica em relação ao tamanho do conjunto de objetos. 95

6.18 Comparação entre a ocupação de espaço (em termos do número de páginas de disco) das versões estática e dinâmica em relação ao tamanho do conjunto de objetos. 96

6.19 Comparação entre a eficiência de busca das versões estática e dinâmica em relação ao número de objetos da base de dados. Esses gráficos apresentam resultados para consultas R-NN (coluna da esquerda) e k-NN (coluna da direita). Eles reportam o número médio de cálculos de distância (primeira linha), o número médio de acessos a disco (segunda linha) e o tempo médio para realizar uma consulta (terceira linha). 97

7.1 Fluxograma do VISON. 104
7.2 Dissimilaridades entre pares de quadros do vídeo *MRS150072* do conjunto de dados do TRECVID 2007. 106
7.3 Quadros selecionados pela abordagem proposta para o vídeo *MRS150072* do conjunto de dados do TRECVID 2007. 107
7.4 O comportamento dos histogramas de cor (segunda coluna) e de orientação (terceira coluna) para quadros normais (primeira linha), monocromáticos (segunda linha) e barras coloridas (terceira linha). 109
7.5 Quadros mantidos pela abordagem proposta para o vídeo *MRS150072* do conjunto de dados do TRECVID 2007. 110
7.6 A interface do VISON, na qual resumos de vídeo são produzidos *online*. Os usuários podem controlar a qualidade dos resumos e especificar o tempo máximo de espera. 111
7.7 O método de avaliação CRU (adaptada de Avila [21]). 113
7.8 A função densidade de probabilidade (PDF) para a distribuição da variância normalizada de histogramas extraídos de diferentes tipos de quadros. . 117
7.9 A média da medida-F obtida a partir da combinação de diferentes níveis para os parâmetros ϵ e λ. 117
7.10 Diferença entre as médias da medida-F de diferentes combinações de parâmetros a um nível de confiança de 98%. 118
7.11 Resumos obtidos por diferentes abordagens para o vídeo *A New Horizon, segment 2* (disponível via Open Video). 120
7.12 Resumos dos usuários para o vídeo *A New Horizon, segment 2*. 121
7.13 Dois resumos automáticos e um resumo de usuário para um vídeo de esportes. 123
7.14 Dois resumos automáticos e um resumo de usuário para um vídeo de notícias. 123
7.15 Dois resumos automáticos e um resumo de usuário para um comercial. . . 124
7.16 Dois resumos automáticos e um resumo de usuário para um vídeo caseiro. . 124
7.17 Exemplo de um gráfico de caixa no modelo de Tukey [124, 125]. 127
7.18 Comparação da qualidade do resumo de vídeo produzido por diferentes abordagens. 128

Lista de Acrônimos

BBC	*British Broadcasting Corporation* (Corporação Britânica de Radiodifusão)
BD	Banco de Dados
BP-tree	*Ball-and-Plane tree* (Árvore de Bola e Plano)
CIE	*Commission Internationale de L'Éclairage* (Comissão Internacional de Iluminação)
CPU	*Central Processing Unit* (Unidade Central de Processamento)
CRU	Comparação com Resumos de Usuários
DBM-tree	*Density-Based Metric tree* (Árvore Métrica Baseada em Densidade)
DCT	*Discrete Cosine Transform* (Transformada Discreta de Cossenos)
DT	*Delaunay Triangulation* (Triangulação de Delaunay)
DVR	*Digital Video Recording* (Gravador de Vídeo Digital)
E/S	Entrada e Saída
EMD	*Earth Mover's Distance* (Distância do Movimento de Terra)
FPF	*Furthest-Point-First* (Ponto mais Distante Primeiro)
GH-tree	*Generalized Hyperplane tree* (Árvore de Hiperplano Generalizado)

GJD	*Generalized Jaccard Distance* (Distância Generalizada de Jaccard)
GNAT	*Geometric Near-Neighbor Access Tree* (Árvore de Acesso Geométrico ao Vizinho Próximo)
GOP	*Group Of Pictures* (Grupo de Imagens)
HI	*Histogram Intersection* (Intersecção de Histogramas)
HSV	*Hue, Saturation, and Value* (Matiz, Saturação e Intensidade)
IACC	*Internet Archive Creative Commons* (Arquivo da Internet de Criações Comuns)
LC	*List of Clusters* (Lista de Agrupamentos)
LCS	*Longest Common Subsequence* (Maior Subsequência Comum)
LMedS	*Least Median-of-Squares* (Mínima Mediana dos Quadrados)
MA	Métodos de Acesso
MAE	Métodos de Acesso Espaciais
MAM	Métodos de Acesso Métricos
MOCA	*Movie Content Analysis Project* (Projeto de Análise de Conteúdo de Filmes)
MPEG	*Moving Picture Experts Group* (Grupo de Especialistas em Imagens com Movimento)
MST	*Minimum Spanning Tree* (Árvore Geradora Mínima)
M-tree	*Metric tree* (Árvore Métrica)
MVP-tree	*Multi-Vantage-Point tree* (Árvore de Múltiplos Pontos de Vantagem)
PAM	*Partitioning Around Medoids* (Partição ao Redor de Medoids)

PCA	*Principal Component Analysis* (Análise de Componentes Principais)
PDF	*Probability Density Function* (Função Densidade de Probabilidade)
RANSAC	*RANdom SAmple Consensus* (Consenso de Amostra Aleatória)
RGB	*Red, Green, and Blue* (Vermelho, Verde e Azul)
ROC	*Receiver Operating Characteristics* (Características de Operação do Receptor)
SAT	*Spatial Approximation Tree* (Árvore de Aproximação Espacial)
SGBD	Sistema Gerenciador de Banco de Dados
SIFT	*Scale Invariant Features Transform* (Transformada de Propriedades Invariantes a Escala)
STIMO	*STIll and MOving Video Storyboard* (Sumário de Vídeo em Movimento e Estático)
SVD	*Singular Value Decomposition* (Decomposição em Valores Singulares)
VISON	*VIdeo Summarization for ONline applications* (Sumarização de Vídeo para aplicações *Online*)
VP-tree	*Vantage-Point tree* (Árvore de Ponto de Vantagem)
VSUMM	*Video SUMMarization* (Sumarização de Vídeo)
ZNCC	*Zero-mean Normalized Cross Correlation* (Correlação Cruzada Normalizada com média Zero)

Capítulo 1

Introdução

A evolução das tecnologias de aquisição, compressão e transmissão de dados tem facilitado a maneira como vídeos digitais são criados, armazenados e distribuídos. O aumento na quantidade de dados de vídeo vem criando grandes bibliotecas de vídeo digital. Isso tem despertado grande interesse em sistemas capazes de gerenciar essas coleções de maneira eficiente [41, 78, 133].

Fazer uso eficiente da informação de vídeo requer um sistema capaz de armazenar os dados de maneira compacta e organizada. É possível gerenciar vídeos utilizando-se atributos independentes do seu conteúdo, tais como o seu formato gráfico, o seu tamanho físico e a sua resolução. Esses atributos podem ser manipulados por sistemas gerenciadores de bancos de dados (SGBDs) [58, 136]. Porém, as operações nesses sistemas ficam restritas aos atributos utilizados, dificultando o processo de recuperação, uma vez que não descrevem o conteúdo armazenado.

Uma alternativa a esses sistemas consiste em utilizar palavras-chave para representar o conteúdo dos vídeos [78, 133]. Isso requer uma anotação prévia de cada vídeo, uma tarefa que consome muito tempo. Além disso, esse processo normalmente é pouco eficaz, já que a interpretação do conteúdo de um vídeo varia de acordo com o conhecimento, o objetivo, a experiência e a percepção de cada usuário [64].

Tais dificuldades vêm sendo resolvidas por meio de sistemas capazes de abstrair representações semânticas de alto nível a partir das informações de baixo nível dos vídeos, conhecidos por sistemas de gestão de vídeos por conteúdo [41, 61]. A Figura 1.1 mostra a arquitetura básica desses sistemas. Duas funcionalidades principais devem ser apoiadas por essa arquitetura: a inserção de dados e o processamento de consultas.

A inserção de dados é responsável por extrair características textuais e visuais apropriadas dos vídeos e armazená-las em um banco de dados (módulos e setas tracejadas). Esse processo é normalmente realizado *offline* [161].

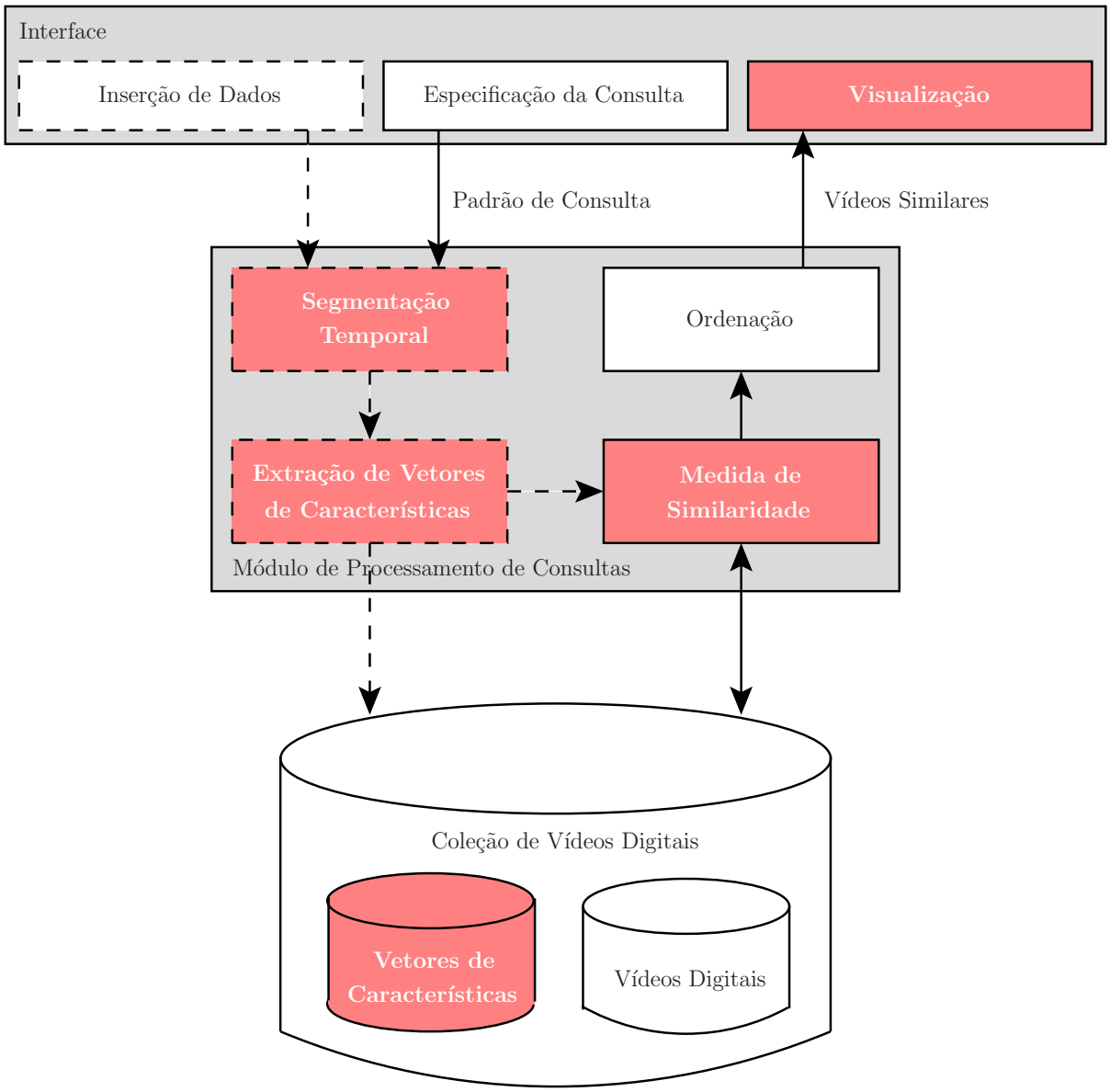


Figura 1.1: Arquitetura típica de um sistema de gestão de vídeos por conteúdo (adaptada de Torres e Falcão [161]).

O processamento de consultas, por outro lado, é organizado como segue: a interface permite a um usuário especificar um padrão de consulta e visualizar os resultados relevantes. O módulo de processamento de consultas extrai um vetor de características desse padrão de consulta e aplica uma função de distância para avaliar a sua similaridade em relação aos dados do banco. A seguir, ordenam-se esses resultados em ordem decrescente de similaridade em relação ao padrão de consulta especificado e enviam-se os resultados mais relevantes para o módulo de interface [161].

A Figura 1.2 apresenta o fluxograma de processamento de sistemas baseados nessa arquitetura. Devido à complexidade desse material, existem cinco desafios principais na construção de tais sistemas: (1) dividir o fluxo de vídeo em trechos manuseáveis de acordo com a sua estrutura de organização, (2) implementar algoritmos para codificar as propriedades de baixo nível de um trecho de vídeo em vetores de características, (3) desenvolver medidas de similaridade para comparar esses trechos a partir de seus vetores, (4) responder rapidamente a consultas por similaridade sobre uma enorme quantidade de sequências de vídeo e (5) apresentar os resultados de forma amigável a um usuário [146].

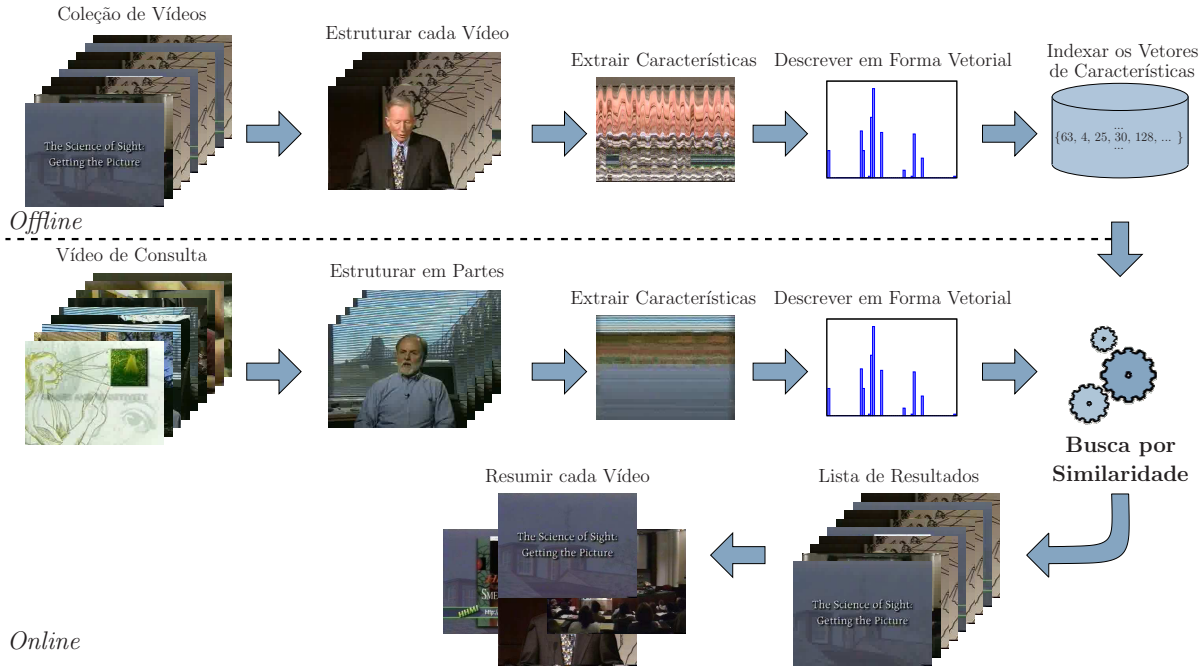


Figura 1.2: Fluxograma de processamento de um sistema de gestão de vídeos por conteúdo.

O primeiro passo para gerenciar o conteúdo de um vídeo é estruturá-lo em suas partes constituintes [43]. Existem muitas formas possíveis para dividir o conteúdo de um vídeo. A divisão em cenas é particularmente útil para vídeos de notícias, nos quais cada cena é normalmente independente das outras cenas do mesmo programa. Por outro lado, uma tomada é uma unidade útil para a indexação e recuperação de vídeos, uma vez que ela é visualmente coerente [113].

Inúmeros algoritmos têm sido propostos para representar o conteúdo do vídeo a partir das suas características de baixo nível, tais como cor, forma, movimento, textura e muitas outras [3, 138, 150, 173]. Contudo, essas técnicas sofrem do problema conhecido como descontinuidade semântica (do inglês, *semantic gap*), que consiste na falta de correspondência entre a informação de baixo nível que pode ser extraída e a interpretação que um usuário tem dessa informação em uma dada situação [147].

Após obter representações compactas, é preciso ainda definir uma medida de similaridade para comparar vídeos a partir dessas representações. Muitos esforços nessa área concentram-se no problema de procurar por um segmento pequeno, tal como um comercial de televisão, dentro de uma sequência longa [79, 105, 117]. Ao estender a medida de similaridade para todo o vídeo, o desafio é definir uma única medida para avaliar a similaridade entre duas sequências de vídeos inteiras [40, 51, 179].

Assim, quando um usuário especifica um padrão de consulta ao sistema, a sua assinatura é extraída e a medida de similaridade é aplicada para identificar todas as assinaturas similares, em um banco de dados (BD) com um grande número de entradas. O algoritmo mais simples para manipular as informações de um BD é uma busca sequencial. Nesse método, todas as entradas devem ser analisadas. Embora simples, esse método é inviável para grandes coleções, uma vez que o tempo gasto para processar uma consulta é proporcional ao tamanho do banco [52].

Para garantir uma resposta rápida, é imperativo o desenvolvimento de algoritmos de busca por similaridade que acelerem esse processo. Elaboradas estruturas de dados, conhecidas como métodos de acesso (MAs), têm sido propostas a fim de acelerar buscas por similaridade [44, 45, 47, 65, 94]. No entanto, a maioria desses métodos sofre de um problema conhecido como maldição da dimensionalidade (do inglês, *curse of dimensionality*): na medida em que aumenta a dimensionalidade dos dados, as estruturas de indexação desses métodos tornam-se lentas e grandes demais [174].

Uma estratégia possível para contornar tal problema consiste em utilizar algoritmos de agrupamento de dados [2, 25, 27, 62, 103, 139, 171]. Esses métodos fornecem uma organização eficiente de grandes coleções, permitindo que os usuários encontrem facilmente uma informação relevante [145].

No final, os usuários são apresentados com uma lista de vídeos similares a um dado padrão de consulta especificado. É inviável esperar que um usuário assista a todo o conteúdo desses vídeos ou uma boa parte dele, a fim de descobrir do que eles realmente se tratam, uma vez que o custo de tempo necessário para a realização desse processo é elevado, ainda mais se observado que um vídeo pode não conter o que era esperado.

Portanto, é imperativo o desenvolvimento de métodos capazes de fornecer aos usuários uma representação concisa do vídeo para dar uma ideia do seu conteúdo, sem ter que vê-lo completamente, de modo que um usuário possa julgar quando um vídeo é relevante ou não. Em geral, humanos julgam mais rapidamente a relevância de segmentos de vídeo interrelacionados. Assim, a abordagem mais simples é agrupar trechos de vídeo com um conteúdo similar e, dessa forma, o julgamento para um dado trecho pode ser aplicado a todos os seus similares [22, 63, 99, 119, 126, 134].

Existem extensivos estudos em técnicas para resolver cada um dos problemas supracitados e atender a cada um dos requisitos envolvidos no desenvolvimento de sistemas de

gestão de vídeos por conteúdo. A maioria desses trabalhos envolve algoritmos e métodos computacionalmente custosos, em termos de tempo e espaço, limitando a sua aplicação apenas ao ambiente acadêmico e/ou a grandes empresas.

Ao contrário a essa vertente, os usuários têm mostrado um crescente interesse por soluções em portabilidade e conectividade, o que tem refletido diretamente na expansão do mercado de dispositivos móveis e embutidos. Nesse cenário, é imperativo o desenvolvimento de técnicas tanto eficazes quanto eficientes a fim de atender a essa demanda e permitir que eles tenham acesso a tecnologias modernas.

Nesse contexto, o objetivo deste trabalho é oferecer soluções para vários problemas em aberto na literatura de processamento de vídeos digitais, mais especificamente, voltados à análise, indexação e recuperação; e contribuir com a superação dos desafios de pesquisa envolvidos na especificação e implementação de sistemas de gestão de vídeos por conteúdo que, além de atender aos diversos requisitos envolvidos em sua concepção, possam ser aplicados a dispositivos com baixo poder computacional e/ou em ambientes que necessitem de uma resposta rápida, mantendo um padrão de qualidade comparável ao estado da arte.

A principal contribuição deste trabalho é a introdução de cinco abordagens originais, uma para cada módulo da arquitetura básica de um sistema de gestão de vídeos por conteúdo, os quais estão destacados em vermelho na Figura 1.1. Todas esses métodos foram projetados para serem, ao mesmo tempo, eficientes e eficazes, a fim de torná-los escaláveis e, portanto, adequados a grandes coleções de vídeos. Tais técnicas foram implementadas em linguagem C, usando a biblioteca FFmpeg[1] para a manipulação de vídeos. Elas são:

1. Uma nova abordagem para a segmentação temporal de sequências de vídeo (Segmentação Temporal). A eficiência computacional dessa técnica a torna adequada para tarefas *online* [16].

2. Uma nova abordagem para representar o movimento da câmera em sequências de vídeo (Extração de Vetores de Características). Essa técnica relaciona os parâmetros de movimento diretamente com as operações físicas da câmera [8, 9].

3. Uma nova abordagem para a comparação de sequências de vídeo (Medida de Similaridade). A eficiência computacional dessa técnica a torna adequada para grandes coleções de vídeo [17].

4. Uma nova estrutura de indexação para a realização de buscas por similaridade em espaços métricos (Vetores de Características). Essa técnica é escalável, o que a torna adequada para grandes volumes de dados [11, 12, 15].

[1]FFmpeg. Disponível em: `http://ffmpeg.org/` (último acesso em 14 de outubro de 2011)

5. Uma nova abordagem para a sumarização de sequências de vídeo, que permite a customização do usuário (Visualização). Essa técnica foi projetada para a produção de resumos de vídeo tanto estáticos quanto dinâmicos em tarefas *online* [13, 19, 20].

Uma outra contribuição é a avaliação empírica dessas técnicas contra métodos clássicos em análise, indexação e recuperação de vídeos digitais. Para isso, diversos experimentos foram conduzidos em várias coleções de vídeo, usando protocolos experimentais imparciais, controlados e reproduzíveis. Todos esses experimentos foram cuidadosamente projetados para assegurar significância estatística, permitindo discernir diferenças genuínas em desempenho de diferenças casuais.

Por fim, todas essas contribuições são somadas na construção de um sistema de gestão de vídeos por conteúdo computacionalmente rápido, capaz de atingir a um padrão de qualidade similar, ou até mesmo superior, a soluções atuais. Isso abre uma série de possibilidades que merecem um estudo mais aprofundado, mas uma consequência imediata é que ele pode ser implementado em dispositivos com capacidade limitada.

A fim de facilitar o entendimento desta tese, optou-se por agrupar tanto a fundação teórica quanto os resultados de pesquisa de acordo com os módulos supracitados, uma vez que eles integram diferentes tópicos de interesse em diferentes áreas da computação, apresentando diferentes problemas em aberto e desafios de pesquisa. Portanto, as abordagens propostas não se limitam somente ao sistema introduzido, podendo ser incorporadas individualmente em diferentes aplicações e métodos.

O restante deste trabalho está organizado como se segue. O Capítulo 2 introduz os principais conceitos básicos necessários para a compreensão deste manuscrito, incluindo vídeo digital, padrões de codificação, entre outros. O Capítulo 3 apresenta uma nova abordagem para a segmentação temporal de sequências de vídeo. O Capítulo 4 analisa a eficácia de um novo método para caracterizar o conteúdo de um vídeo. O Capítulo 5 introduz uma nova abordagem para a comparação de sequências de vídeo a partir de seu conteúdo visual. O Capítulo 6 estuda o desempenho de uma nova estrutura de indexação usada para acelerar o processamento de consultas por similaridade em grandes coleções. O Capítulo 7 apresenta uma nova abordagem para auxiliar na visualização de resultados, a qual é adequada para o uso *online*, permitindo a interação com usuários. Por fim, as conclusões finais sobre o estudo desenvolvido, bem como direções para trabalhos futuros, são discutidas no Capítulo 8.

Capítulo 2

Conceitos Básicos

Neste capítulo, são apresentados os aspectos teóricos necessários para o embasamento do presente trabalho. Esses conceitos incluem: imagens digitais, espaços de cor, fluxo ótico, vídeo digital, operações de câmera, padrões de codificação, entre outros.

2.1 Imagem Digital

Uma imagem (monocromática) é uma função bidimensional $I(x, y)$, na qual x e y são coordenadas espaciais e o valor de I em qualquer ponto (x, y) é proporcional ao brilho (ou nível de cinza) da imagem nesse ponto [70]. Uma imagem digital nada mais é que uma imagem $I(x, y)$ que teve tanto as suas coordenadas espaciais quanto o seu brilho discretizados (digitalizados). Dessa forma, ela pode ser interpretada como uma matriz na qual cada elemento é identificado pelos índices da linha e da coluna às quais pertence, cujo valor corresponde ao seu brilho ou nível de cinza, como ilustra a Figura 2.1.

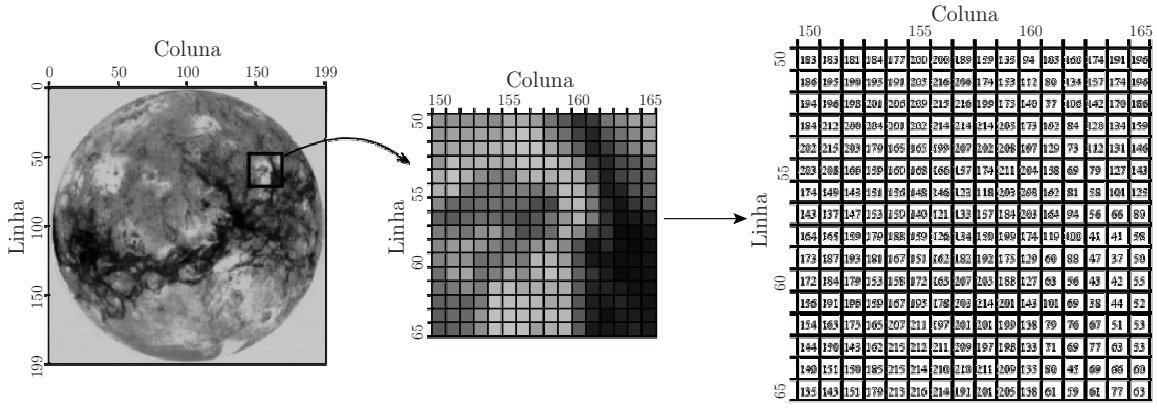


Figura 2.1: Representação de uma imagem digital.

A digitalização das coordenadas espaciais é conhecida como amostragem (do inglês, *sampling*) e a digitalização do brilho é conhecida como quantização do nível de cinza (do inglês, *gray-level quantization*) [70]. A resolução de uma imagem (o grau de detalhes perceptíveis) é fortemente dependente desses dois parâmetros. Quanto mais finas a amostragem e a quantização, melhor a imagem digitalizada se aproxima do conteúdo da imagem original. No entanto, os custos de armazenamento e de processamento da imagem digital crescem rapidamente com o aumento da resolução [70].

No caso de imagens digitais coloridas, cada ponto é descrito não apenas pelo seu brilho, mas também por outras propriedades como matiz e saturação. Em geral, a cor de cada ponto é representada a partir de um modelo tridimensional, chamado de espaço de cor.

2.2 Espaços de Cor

Um espaço de cor é uma especificação de um sistema de coordenadas tridimensionais e um subespaço dentro desse sistema, no qual cada cor é representada por um único ponto [70]. Eles podem ser classificados em três categorias principais [70]: (1) orientados à arquitetura física, (2) orientados ao usuário e (3) espaços de cor uniformes.

Um modelo orientado à arquitetura física é definido de acordo com as propriedades dos dispositivos utilizados para reproduzir as cores (monitores, impressoras, etc) [70]. O espaço de cor mais conhecido e utilizado é um modelo orientado à arquitetura física, chamado RGB (do inglês, *Red, Green, and Blue*) [32]. Nesse modelo, a cor exibida depende não somente dos valores RGB, mas também das especificações do dispositivo que a exibe. A percepção desse espaço não é uniforme, pois as diferenças entre as cores não refletem as diferenças percebidas pelos humanos. O espaço de cor RGB é um cubo (Figura 2.2), no qual a diagonal principal representa valores de cinza do preto ao branco e qualquer ponto (cor) é representado pela soma ponderada de vermelho (R), verde (G) e azul (B).

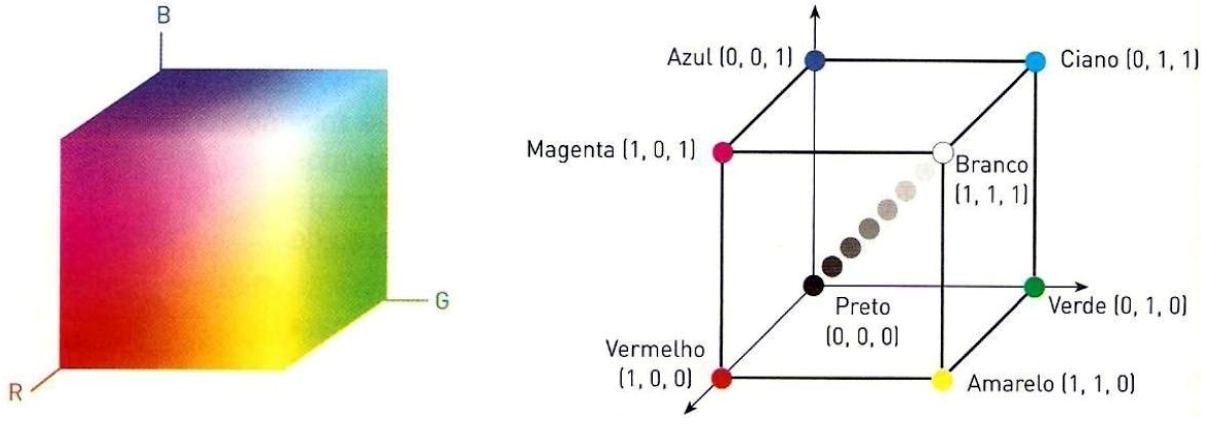


Figura 2.2: O espaço de cor RGB.

Os espaços de cor orientados ao usuário baseiam-se na percepção humana das cores [70]. Eles exploram as características que são utilizadas por humanos para distinguir uma cor de outra, como matiz (o comprimento de onda dominante que produz a sensação visual de vermelho, amarelo, verde, azul, ou a combinação de duas dessas cores), saturação (o grau de pureza da cor, o qual está relacionado ao desvio padrão em relação ao comprimento de onda dominante) e a intensidade (o brilho da cor, que está relacionado à quantidade de branco que ela apresenta) [70]. O espaço de cor HSV (do inglês, *Hue, Saturation, and Value*) [32] é um exemplo de espaço de cor orientado ao usuário. O espaço de cor HSV é representado por um cone (Figura 2.3). O eixo vertical desse cone representa os valores de intensidade (V), o ângulo formado por esse eixo define a matiz (H) e a distância desse eixo fornece a saturação (S).

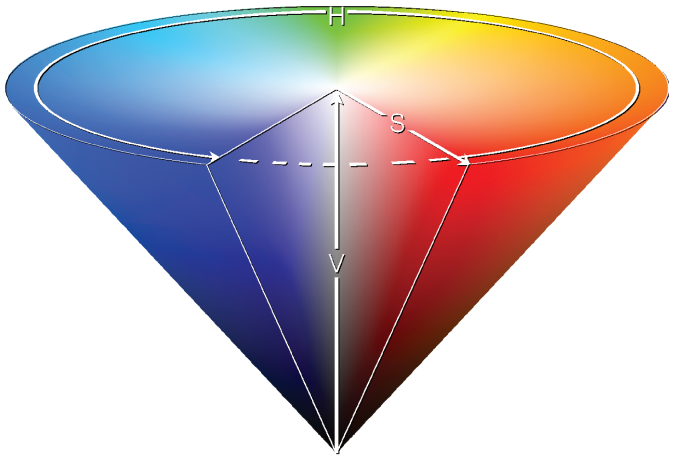


Figura 2.3: O espaço de cor HSV.

Os espaços de cor uniformes são espaços nos quais as diferenças numéricas entre as cores refletem as diferenças percebidas pelos humanos [70]. O espaço de cor Lab da CIE[1] (*Commission Internationale de L'Éclairage*) [32] é um exemplo de espaço de cor uniforme. O modelo CIE Lab representa as diferenças de três pares elementares: vermelho-verde, amarelo-azul e branco-preto. Dessa forma, como mostra a Figura 2.4, o eixo a do espaço de cor CIE Lab se estende do verde $(-a)$ ao vermelho $(+a)$ e o eixo b do azul $(-b)$ ao amarelo $(+b)$. O eixo L varia o brilho do preto $(-L)$ ao branco $(+L)$.

[1] *Commission Internationale de L'Éclairage* (CIE) – `http://www.cie.co.at/cie/` (último acesso em 14 de outubro de 2011).

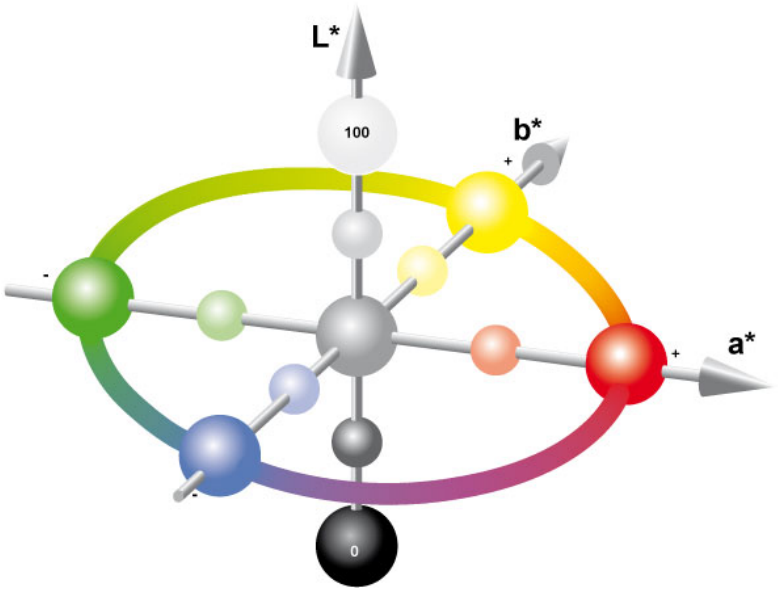


Figura 2.4: O espaço de cor CIE Lab.

2.3 Fluxo Ótico

Dado um ponto (*pixel*) p de uma imagem J, o seu **correspondente ótico** em uma imagem K é definido como sendo um ponto q tal que os valores de intensidade em J na vizinhança de p são maximalmente similares aos valores de K na vizinhança de q, conforme mostra a Figura 2.5.

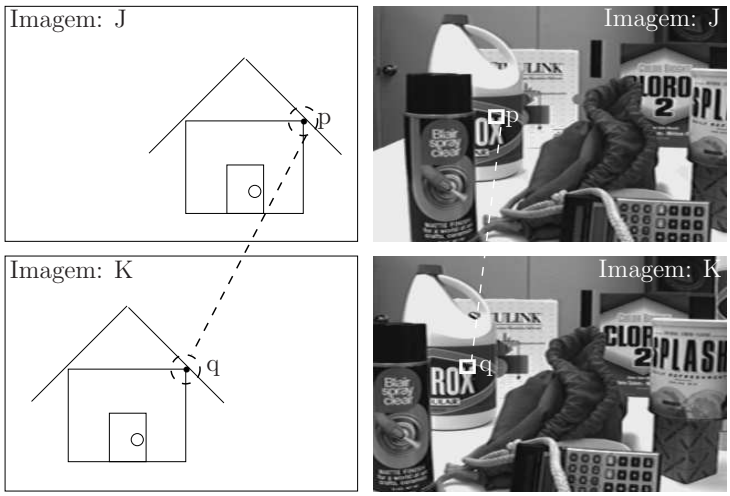


Figura 2.5: Exemplos de uma correspondência ótica (adaptada de Minetto [114]).

O critério de similaridade utilizado para definir uma correspondência ótica é normalmente expresso por uma função de distância, que mede a dissimilaridade entre as imagens J e K na vizinhança dos pontos p e q, respectivamente. O vetor $\vec{f} = q - p$ é conhecido como o **deslocamento ótico** do ponto p entre as imagens J e K. A função $f(p)$ que associa cada ponto p ao seu deslocamento ótico é chamada de **fluxo ótico** [26].

A Figura 2.6 ilustra um fluxo ótico f. Nessa figura, foi amostrado um conjunto fixo de pontos $p_1, p_2, \ldots, p_n$, resultando em um conjunto de vetores $\vec{f_1}, \vec{f_2}, \ldots, \vec{f_n}$. Cada vetor $\vec{f_i}$ é representado por um segmento de reta com origem no ponto p_i, cuja direção é a do vetor $\vec{f_i}$ e o comprimento é proporcional a $|\vec{f_i}|$.

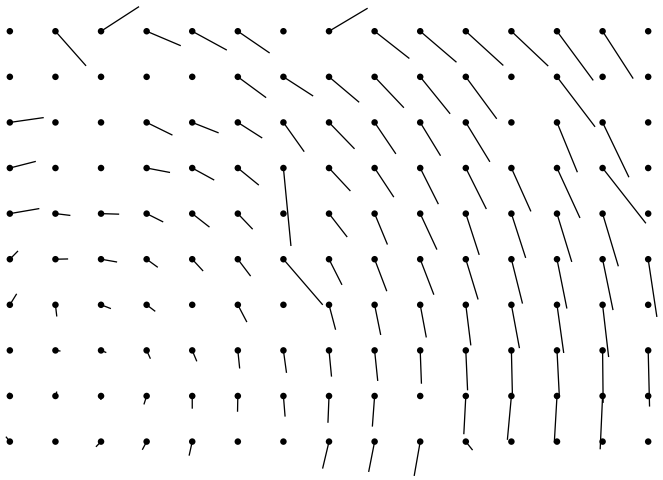


Figura 2.6: Representação de um fluxo ótico a partir da amostragem de um conjunto fixo de pontos, resultando em um conjunto de vetores, cuja direção e magnitude são representadas por um segmento de reta.

2.4 Vídeo Digital

Vídeo digital é uma sequência de quadros (imagens) em formato digital que, quando reproduzidos um a um, em determinada velocidade, apresentam a ilusão de movimento. Tipicamente, são usados 24 quadros por segundo para se obter tal efeito. Um vídeo pode ainda conter um canal de áudio sincronizado à sequência de imagens.

A taxa de quadros, medida em quadros por segundos (do inglês, *frames per second*), define quantos quadros são mostrados a cada segundo do vídeo. Quanto maior for essa taxa, mais suave será a aparência do movimento durante a reprodução do vídeo e, como consequência, maior será a quantidade de quadros para armazenar. Para reduzir este problema, técnicas de compressão de vídeo (por exemplo, MPEG-1/2/4) são normalmente utilizadas.

Um vídeo é quase sempre estruturado em uma hierarquia rígida, conforme mostra a Figura 2.7. Um **programa** é dividido em **cenas**, e uma cena é composta por uma ou mais **tomadas** (do inglês, *shots*) de câmera. Uma cena geralmente corresponde a um evento lógico de um programa, de forma que uma sequência de tomadas pode formar uma cena de diálogo em uma comédia ou uma cena de ação em um filme.

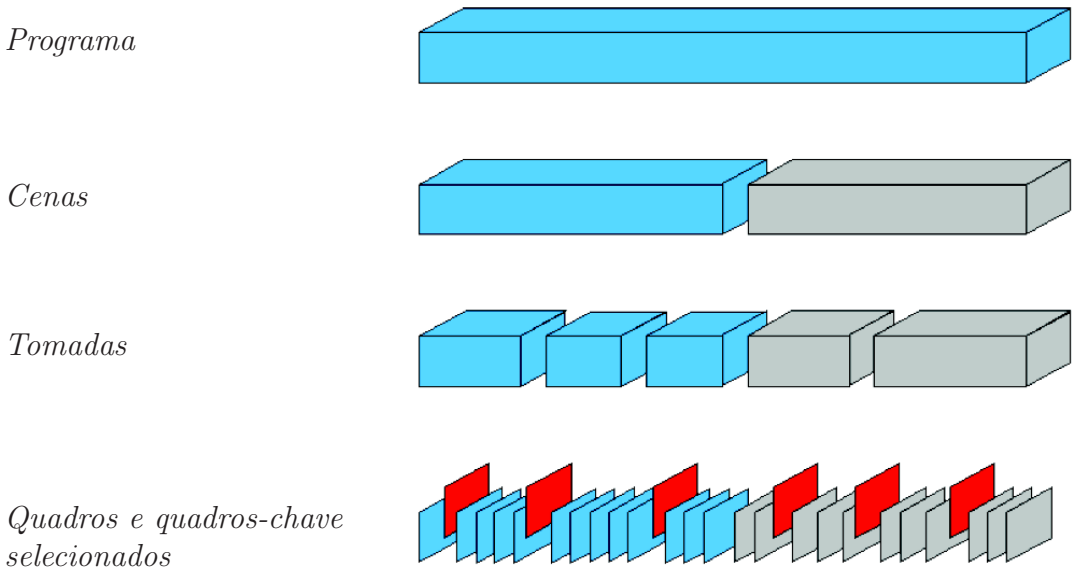


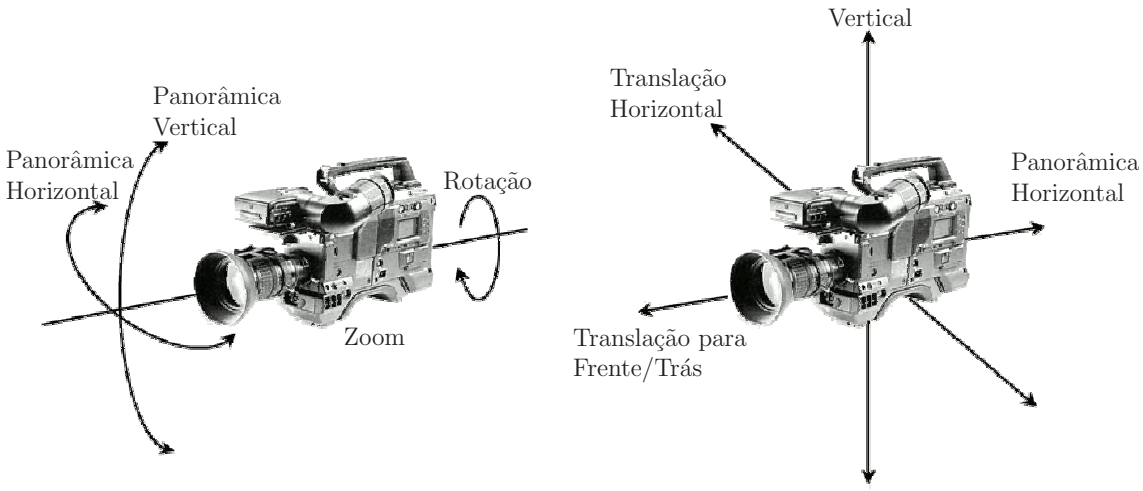
Figura 2.7: Estrutura hierárquica de uma sequência de vídeo.

Uma tomada corresponde a **quadros** (do inglês, *frames*) de vídeo de uma única câmera, podendo envolver tanto movimentos de câmera quanto movimentos de objetos entrando e/ou saindo de cada quadro. Assim, uma única tomada pode conter uma grande quantidade de movimento, como em vídeos de clipes de músicas, ou nenhum movimento, como em quadros estáticos de uma paisagem ao ar livre.

É possível ainda decompor uma tomada em quadros-chave, isto é, um ou mais quadros representativos. Muitos sistemas de gerenciamento de vídeos representam uma tomada por meio da seleção de um único quadro-chave [120, 177].

2.5 Operações de Câmera

Durante a gravação de uma sequência de vídeo, a câmera normalmente tem que ser orientada de acordo com o movimento de um objeto de interesse. Assim, estimar o movimento de uma câmera a partir de um vídeo implica obter informações sobre as operações de câmera envolvidas na criação desse vídeo. Segundo Park *et al.* [127], essas operações podem ser classificadas em sete tipos básicos, conforme mostra a Figura 2.8.



(a) Operações Rotacionais (b) Operações Translacionais

Figura 2.8: As operações básicas de uma câmera (adaptada de Park *et al.* [127]).

As operações da câmera podem ser tratadas em duas situações [127]: uma com a posição da câmera sendo fixa e outra com a câmera sendo móvel. Quando a posição da câmera é fixa (Figura 2.8(a)), as operações possíveis são panorâmica horizontal (do inglês, *panning*), panorâmica vertical (do inglês, *tilting*), zoom (do inglês, *zooming*) e rotação (do inglês, *rolling*). Quando a câmera é móvel (Figura 2.8(b)), as movimentações possíveis são translação horizontal (do inglês, *tracking*), translação vertical (do inglês, *booming*) e translação para frente/trás (do inglês, *dollying*). O movimento da câmera pode ser constituído por uma única movimentação ou qualquer combinação dessas operações.

2.6 Padrão de Codificação MPEG-1/2/4

No padrão de codificação MPEG, um vídeo é composto por três tipos principais de imagens: intracodificada (quadro I), predita (quadro P) e bidirecionalmente predita (quadro B). Essas imagens são organizadas em sequências de grupos de imagens (do inglês, *Group of Pictures* – GOPs), como ilustrado na Figura 2.9.

Cada imagem é dividida em um conjunto de blocos disjuntos de 8×8 pontos. Esses blocos são normalmente agrupados para formar regiões maiores, denominadas macroblocos. Para minimizar os efeitos da propagação de erros no fluxo de vídeo, tais macroblocos são organizados em sequências, chamadas fatias. Elas podem começar e terminar em qualquer macrobloco de uma imagem, mas com a seguinte restrição: a primeira fatia deve começar na parte superior esquerda da imagem (primeiro macrobloco) e o fim da última fatia deve ser o macrobloco inferior direito (último macrobloco) da imagem.

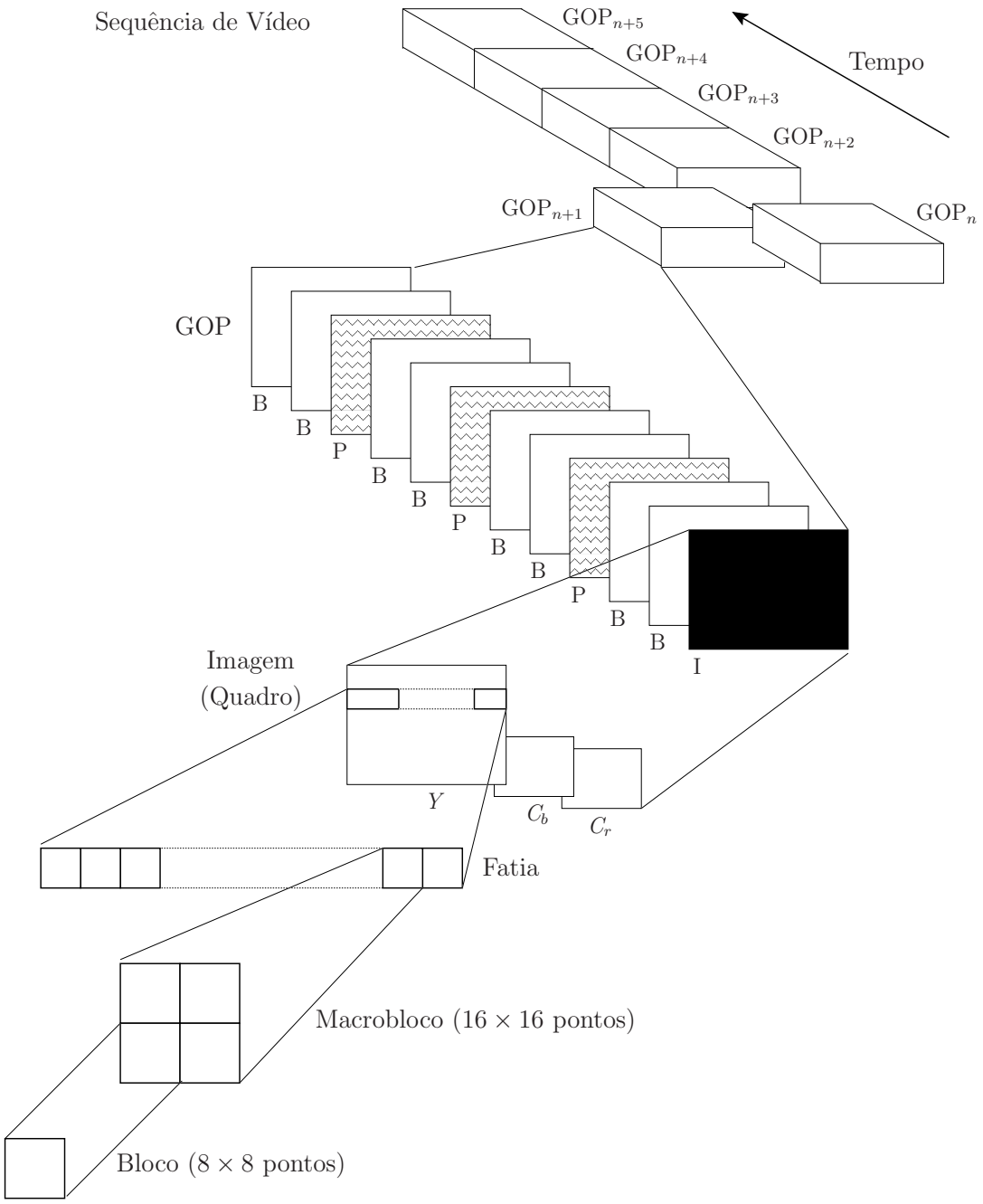


Figura 2.9: Estrutura típica de um vídeo MPEG (adaptada de Ghanbari [67]).

Cada GOP deve começar com um quadro I e pode ser sucedido por um número qualquer de quadros I e/ou P, os quais são considerados como quadros âncora. Entre cada par de quadros âncora consecutivos podem aparecer vários quadros B. A Figura 2.10 mostra uma estrutura típica de um GOP.

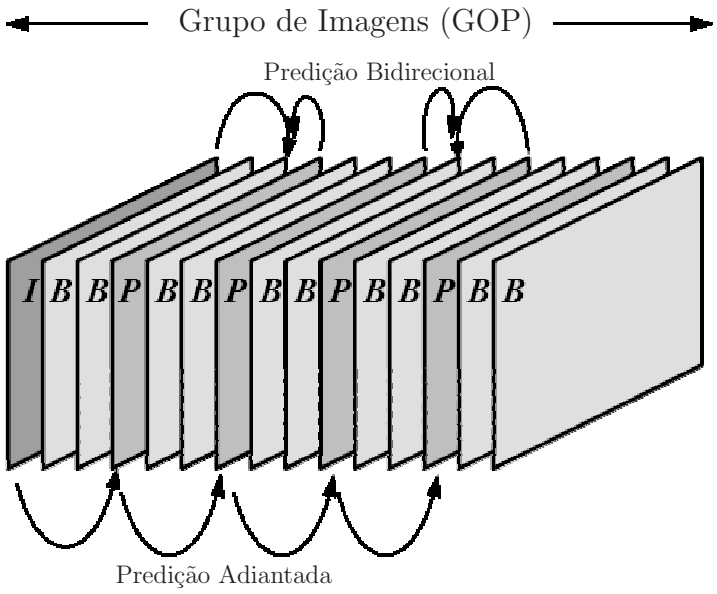


Figura 2.10: Estrutura típica de um GOP em vídeos MPEG (adaptada de Tan *et al.* [159]).

Um quadro I não se refere a nenhum outro quadro de vídeo e, portanto, ele pode ser decodificado de forma independente, fornecendo um ponto de entrada para o acesso aleatório rápido a um vídeo comprimido. Por outro lado, a codificação de um quadro P baseia-se em um quadro âncora anterior, enquanto a codificação de um quadro B pode ser feita com base em dois quadros âncoras, tanto no anterior quanto no subsequente.

Cada quadro I é dividido em uma sequência de macroblocos disjuntos. Para um vídeo codificado usando o espaço de cor YCbCr no formato 4:2:0, cada macrobloco é composto por seis blocos de 8×8 pontos: quatro para a luminância (Y) e dois blocos de crominância (CbCr). Cada macrobloco é totalmente intracodificado, portanto, todos esses blocos são transformados para o domínio da frequência usando a transformada discreta de cossenos (do inglês, *Discrete Cosine Transform* – DCT). Os 64 coeficientes da DCT são então codificados usando quantização (causa perda de energia) e energização (comprimento de execução e Huffman, sem perda de energia) a fim de alcançar compressão.

Cada quadro P é codificado com predição adiantada em relação a seu quadro âncora antecessor (o quadro I ou P anterior). Para cada macrobloco de um quadro P, uma região similar é procurada no quadro âncora a fim de obter um bom casamento entre elas em termos da diferença de intensidade. Se um bom casamento for encontrado, o macrobloco é representado por um vetor de movimento que aponta para a posição dessa região, juntamente com a codificação DCT da diferença (ou resíduo) entre o macrobloco e sua respectiva região. Os coeficientes DCT do resíduo são quantizados e codificados,

enquanto o vetor de movimento é codificado com base na diferença em relação a seus vetores de movimento vizinhos. Tal processo é normalmente conhecido como codificação com compensação de movimento para a frente e esse tipo de macrobloco codificado é chamado de intercodificado. Se um bom casamento não for encontrado, o macrobloco é intracodificado usando o mesmo método de codificação dos macroblocos de quadros I. Normalmente, o resíduo de um macrobloco intercodificado pode ser codificado com menos *bits*. Assim, um macrobloco intercodificado tem melhor compressão do que um macrobloco intracodificado. Espera-se que uma pequena mudança no conteúdo de imagem entre um quadro P e seu quadro âncora irá resultar em mais macroblocos sendo emparelhados no quadro âncora e, portanto, menos macroblocos exigindo intracodificação.

A fim de alcançar uma maior compressão, quadros B são codificados com predição bidirecional usando compensação de movimento para frente e/ou para trás em relação a seu quadro I e/ou P, antecessor e/ou sucessor, mais próximo. Como quadros B não são usados como referência na codificação de outros quadros, eles podem acomodar mais distorção e, portanto, atingir uma compressão mais elevada que quadros P.

A posição de um quadro na sequência de vídeo e o seu tipo (I, P ou B), o número de blocos intracodificados e os coeficientes DC de cada bloco codificado pela DCT, e as posições e os vetores de movimento dos macroblocos intercodificados, podem ser obtidos por meio da decodificação parcial (interpretação e energização) do fluxo de vídeo. Essas operações levam menos que 20% da carga computacional total envolvida no processo de decodificação completa do vídeo [30].

2.7 Imagem DC

A compressão dos quadros I de um vídeo MPEG é realizada por meio da divisão da imagem original em blocos de 8×8 pontos e da transformação dos valores de cada bloco em 64 coeficientes da DCT. O termo DC $c(0,0)$ está relacionado à intensidade dos pontos $f(i,j)$ através da seguinte equação [169]:

$$c(0,0) = \frac{1}{8} \sum_{x=0}^{7} \sum_{y=0}^{7} f(x,y). \tag{2.1}$$

Em outras palavras, o valor do termo DC é 8 vezes a intensidade média dos pontos de um bloco. Ao extrair o termo DC de todos os blocos, pode-se usar esses valores para formar uma versão reduzida da imagem original, como mostra a Figura 2.11. Essa imagem menor é conhecida como **imagem DC** [169].

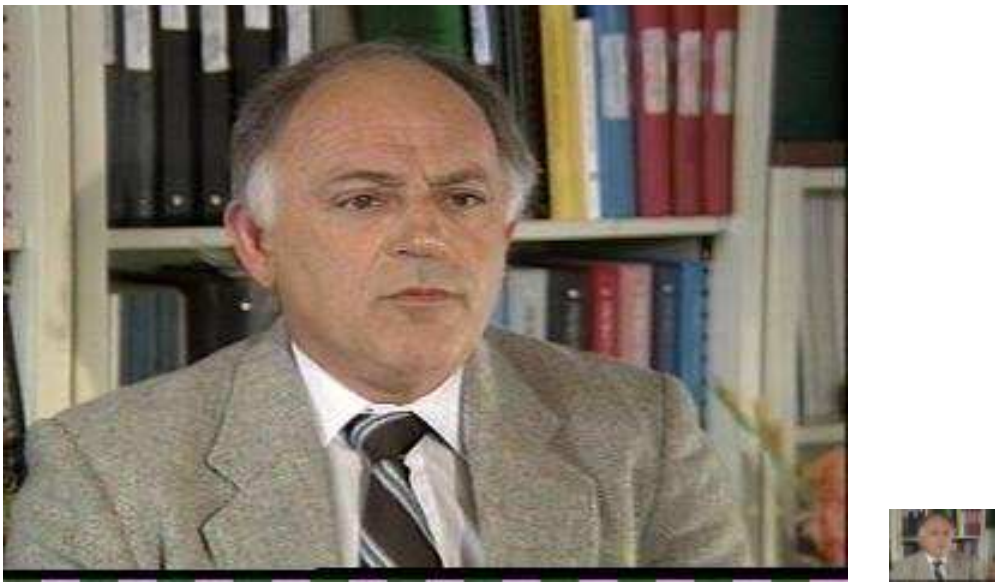


Figura 2.11: Imagem original com 352×240 pontos e sua imagem DC com 44×30 pontos.

2.8 Espaço Métrico

Um **espaço métrico** $\mathcal{M}$ é um par ordenado $\mathcal{M} = (\mathcal{D}, d)$ definido para os objetos de dados de um domínio $\mathcal{D}$ e uma função de distância d, tal que as seguintes propriedades são satisfeitas [47]:

$\forall x, y \in \mathcal{D}, d(x, y) \geq 0$	positividade
$\forall x, y \in \mathcal{D}, d(x, y) = d(y, x)$	simetria
$\forall x, y \in \mathcal{D}, x = y \Leftrightarrow d(x, y) = 0$	identidade
$\forall x, y, z \in \mathcal{D}, d(x, z) \leq d(x, y) + d(y, z)$	desigualdade triangular

Essas propriedades permitem a elaboração de estruturas de indexação capazes de responder a consultas por similaridade de modo eficiente. Se os objetos do domínio $\mathcal{D}$ correspondem a vetores com uma mesma dimensão, então o espaço é chamado de **vetorial**. Caso contrário, o espaço é conhecido como **adimensional**. Uma característica dos espaços métricos é a possibilidade de englobar espaços tanto vetoriais quanto adimensionais.

2.9 Funções de Distância

Uma **função de distância** d de determinado espaço métrico $\mathcal{M}$ representa uma maneira de quantificar a proximidade entre objetos de dados do mesmo domínio $\mathcal{D}$. Essas funções normalmente atendem a aplicações específicas ou classes de aplicações [174].

Tais medidas de distância podem ser divididas em dois grupos dependendo da característica dos valores retornados [174]:

- **discreta** – funções de distância que retornam somente um conjunto pequeno ou predefinido de valores,
- **contínua** – funções de distância na qual a cardinalidade do conjunto de valores retornados é muito grande ou infinita.

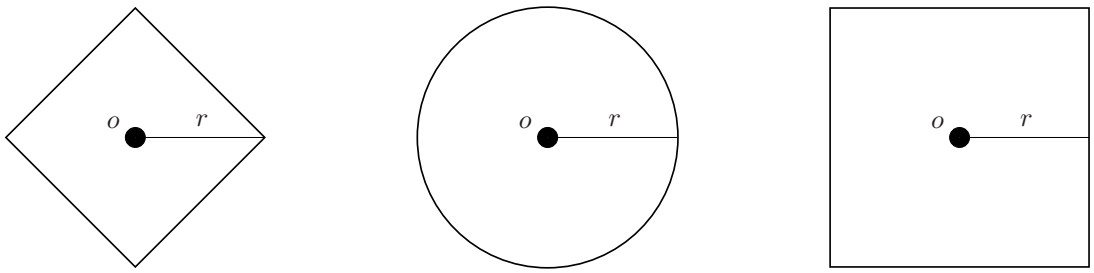
As funções de distância mais conhecidas e utilizadas são as da família L_p (ou distâncias de **Minkowski**), as quais são definidas por [174]:

$$\mathrm{L}_p((x_1,\ldots,x_k),(y_1,\ldots,y_k)) = \left(\sum_{i=1}^{k} |x_i - y_i|^p\right)^{\frac{1}{p}}. \tag{2.2}$$

A Figura 2.12 ilustra, para os diferentes casos da família L_p, o conjunto de objetos que estão à mesma distância r, a partir de um objeto o. Para a função de distância L_1, também conhecida como **Bloco** ou **Manhattan**, o conjunto de objetos de mesma distância r forma um losango. Na função de distância L_2, mais conhecida como distância **Euclidiana**, ele forma uma circunferência. Quando p tende ao infinito, obtém-se a função de distância L_∞, também conhecida como **Chebychev**,

$$\mathrm{L}_\infty((x_1,\ldots,x_k),(y_1,\ldots,y_k)) = \max_{i=1}^{k} |x_i - y_i|, \tag{2.3}$$

na qual o conjunto de objetos com mesma distância forma um quadrado.



(a) L_1 (Manhattan) (b) L_2 (Euclidiana) (c) L_∞ (Chebychev)

Figura 2.12: Representação dos objetos situados à distância r de um objeto o (adaptada de Zezula *et al.* [174]).

Entre outras funções de distância encontradas na literatura estão a intersecção de histogramas (do inglês, *Histogram Intersection* – HI) [158], a distância de Hausdorff [86] e a distância do movimento de terra (do inglês, *Earth Mover's Distance* – EMD) [141].

2.10 Consultas por Similaridade

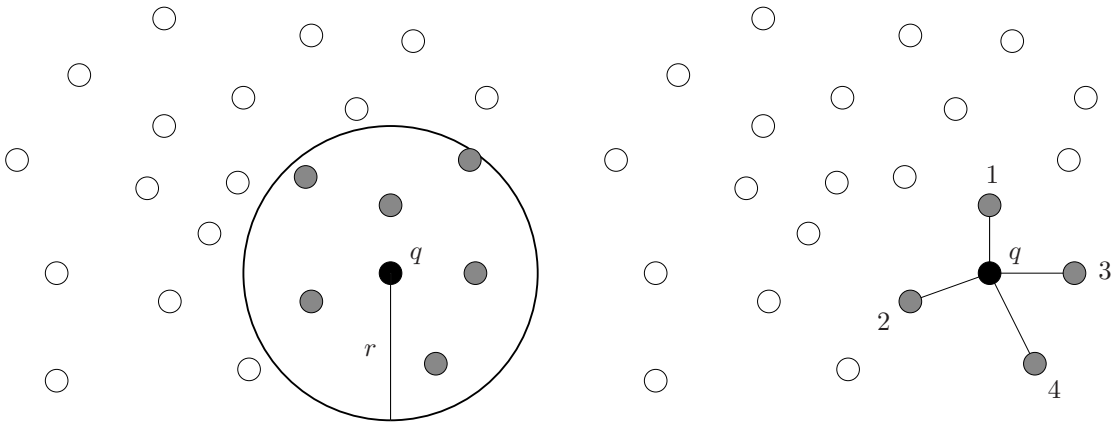
Uma **consulta por similaridade** é definida por um objeto de consulta $q \in \mathcal{D}$ e uma restrição acerca da forma e proximidade necessária, tipicamente expressa como distância. Em resposta, a consulta retorna todos os objetos de dados pertencentes ao domínio $\mathcal{D}$ que satisfazem essas condições, assumindo que esses objetos sejam próximos ao dado objeto de consulta [174].

O tipo de consulta por similaridade mais comum é a **consulta por abrangência** $R(q, r)$. Tal consulta é especificada por um objeto de consulta $q \in \mathcal{D}$ e por um raio de busca r. Esse tipo de consulta recupera todos os objetos de dados encontrados dentro da distância r de q, ou seja [47]:

$$R(q, r) = \{o \in X, d(o, q) \leq r\}. \tag{2.4}$$

Se necessário, os objetos do conjunto de resposta podem ser ordenados de acordo com suas distâncias em relação a q. É importante notar que o objeto de consulta q não precisa necessariamente existir na coleção de pesquisa $X \subseteq \mathcal{D}$. A única restrição é que q deve pertencer ao domínio $\mathcal{D}$.

A Figura 2.13(a) ilustra um exemplo de uma consulta por abrangência $R(q, r)$ utilizando a função de distância L_2. Os objetos (em cinza) contidos na circunferência pertencem ao conjunto de resposta dessa consulta.



(a) Consulta por Abrangência (b) Consulta por Vizinhos mais Próximos

Figura 2.13: Exemplo de consultas por similaridade utilizando a função de distância L_2 (adaptada de Zezula *et al.* [174]).

Quando o raio de busca é zero, a consulta por abrangência $R(q, 0)$ é chamada de **consulta pontual** ou **busca exata**. Nesse caso, o usuário está interessado em cópias idênticas ao objeto de consulta q. Esse tipo de consulta é normalmente utilizado em algoritmos de remoção, quando se deseja localizar um dado objeto para removê-lo do banco [174].

Caso contrário, é preciso especificar uma distância máxima para qualificar os objetos do conjunto de resposta. Entretanto, pode ser difícil especificar um raio de consulta sem algum conhecimento prévio do domínio $\mathcal{D}$ e da função de distância d.

Uma alternativa para buscar objetos similares é utilizar a **consulta por vizinhos mais próximos**. A versão mais básica dessa consulta encontra o objeto mais próximo ao objeto de consulta, que é o vizinho mais próximo de q. Esse conceito pode ser generalizado para o caso em que se procura os k vizinhos mais próximos ao objeto de consulta q. Se a coleção de pesquisa é composta por menos que k objetos de dados, então a consulta retorna todo o banco. Quando diversos objetos de dados apresentam a mesma distância do k-ésimo vizinho mais próximo, então eles podem ser escolhidos arbitrariamente. Formalmente, o conjunto de resposta dessa consulta pode ser definido como [47]:

$$kNN(q) = \{R \subseteq X, |R| = k \wedge \forall x \in R, y \in X - R : d(q, x) \leq d(q, y)\}. \quad (2.5)$$

Um exemplo de uma consulta por vizinhos mais próximos $4NN(q)$ utilizando a função de distância L_2 é ilustrado na Figura 2.13(b). Os objetos conectados por uma linha ao objeto de consulta q pertencem ao conjunto de resposta.

Em muitas situações, é interessante saber o quanto um objeto específico é percebido, em termos de distâncias, por outros objetos na coleção, isto é, quais objetos de dados veem o objeto de consulta q como seu vizinho mais próximo. Isso é conhecido como **consulta por vizinho mais próximo reversa**, na qual o conjunto de resposta é definido por [174]:

$$kRNN(q) = \{R \subseteq X, \forall x \in R : q \in kNN(x) \wedge \forall x \in X - R : q \notin kNN(x)\}. \quad (2.6)$$

É importante notar que mesmo um objeto localizado distante do objeto de consulta q pode pertencer ao conjunto de resposta $kRNN(q)$. Ao mesmo tempo, um objeto próximo de q não necessariamente será membro do conjunto de resposta $kRNN(q)$. Tal propriedade é conhecida como **espalhamento** [174].

Uma extensão dos tipos de consultas definidos acima é a combinação de dois ou mais deles. Por exemplo, pode-se combinar uma consulta por abrangência com uma consulta por vizinho mais próximo, de forma que [174]:

$$kNN(q, r) = \{R \subseteq X, |R| \leq k \wedge \forall x \in R, y \in X - R : d(q, x) \leq d(q, y) \wedge d(q, x) \leq r\}. \quad (2.7)$$

Desse modo, todos os objetos no conjunto de resposta devem apresentar distância inferior ou igual a r em relação ao objeto de consulta q e, se existir mais que k objetos de dados, então apenas os primeiros k objetos mais próximos serão retornados.

2.11 Medidas de Avaliação de Desempenho

O tempo de resposta e o espaço de armazenamento são as medidas normalmente adotadas para avaliar um sistema de recuperação. No domínio de recuperação de informação, entretanto, é necessário avaliar a relevância da informação recuperada (eficácia) [154].

A eficácia de um sistema de recuperação é uma medida relacionada à satisfação de um usuário com o resultado obtido. Para estabelecer uma medida de eficácia, a primeira decisão a ser tomada é definir quais julgamentos são permitidos a um usuário em sua avaliação. A escolha básica é entre uma medida binária e uma medida de múltipla escolha [97].

A medida binária é a mais simples de ser implementada e utilizada, na qual cada vídeo pode ser aceito ou rejeitado. Essas condições estão normalmente ligadas à relevância desse vídeo para um usuário. Entretanto, essas condições apresentam um alto grau de subjetividade, uma vez que diferentes usuários, ou os mesmos usuários em diferentes circunstâncias, podem perceber o conteúdo visual de um vídeo de maneiras diferentes [23].

Quando uma escala binária é utilizada tanto para a relevância quanto para a abrangência, uma tabela de contingência (Tabela 2.1) pode ser estabelecida, mostrando como os vídeos do banco estão divididos de acordo com essas duas classificações (Figura 2.14).

Relevância	**Abrangência**	
	Recuperados	Não recuperados
Relevantes	A	B
Não relevantes	C	D

Tabela 2.1: Tabela de contingência para avaliar a eficácia de recuperação.

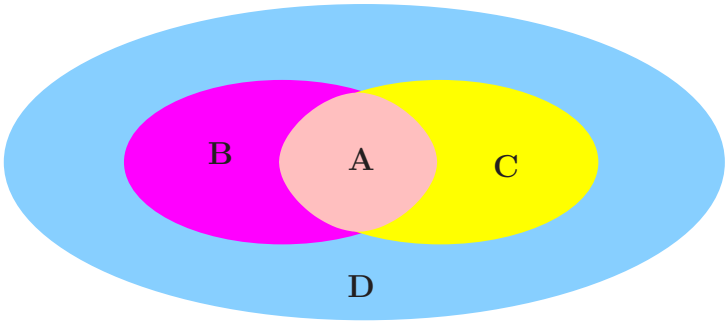


Figura 2.14: Distribuição da coleção de vídeos de acordo com uma escala binária.

Entre as diversas medidas de avaliação de desempenho existentes, a curva **Precisão vs. Revocação** ($P \times R$) [23, 32, 97] é a medida mais conhecida e utilizada na prática. A precisão P_r é a fração dos r vídeos recuperados que são relevantes para a consulta,

$$P_r = \frac{A}{r} = \frac{A}{A + C}, \tag{2.8}$$

enquanto a revocação R_r é a proporção do número total de vídeos relevantes que foram recuperados entre os r vídeos retornados,

$$R_r = \frac{A}{A+B}. \tag{2.9}$$

Cada resultado dessas medidas representa um ponto no espaço $P \times R$ (Figura 2.15). O melhor resultado possível produziria um ponto localizado no canto superior direito, coordenada (1, 1), representando 100% de precisão (nenhum vídeo irrelevante foi retornado) e 100% de revocação (todos os vídeos relevantes foram recuperados).

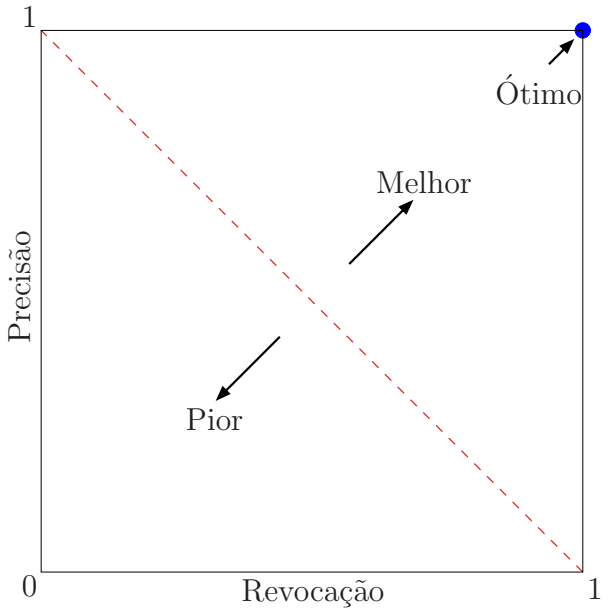


Figura 2.15: Exemplo de espaço $P \times R$.

Outra curva empregada na avaliação de desempenho de sistemas de recuperação ao se adotar uma escala binária é a **ROC** (do inglês, *Receiver Operating Characteristics*). Essa curva permite estudar a variação da sensibilidade e da especificidade para diferentes valores do número r de vídeos recuperados [23, 32, 97].

Trata-se de um gráfico de taxas de acerto Ta_r,

$$Ta_r = R_r = \frac{A}{A+B}, \tag{2.10}$$

por taxas de erro Te_r,

$$Te_r = 1 - P_r = \frac{C}{A+C}, \tag{2.11}$$

representando a relação entre custo (erros) e benefício (acertos).

Cada resultado dessas medidas representa um ponto no espaço ROC (Figura 2.16). O ponto localizado no canto superior esquerdo, coordenada (0, 1), indica o melhor resultado possível, que consiste em uma taxa de acerto igual a 100% (todos os vídeos relevantes foram recuperados) e uma taxa de erro igual a 0% (nenhum vídeo irrelevante foi retornado).

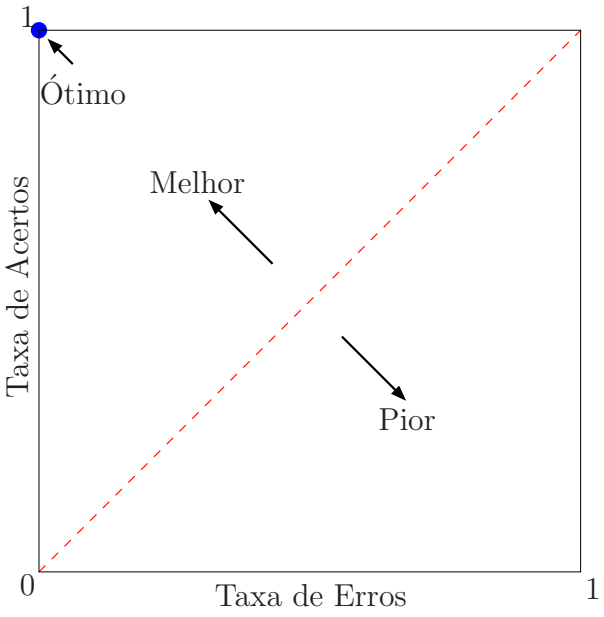


Figura 2.16: Exemplo de espaço ROC.

Alguns pesquisadores acreditam que uma medida de eficácia de recuperação deve ser expressa por meio de um único número, capaz de representar valores relativos e absolutos [23]. Essa medida pode ser obtida utilizando um único ponto da curva $P \times R$, por exemplo [23]: (1) a precisão no ponto mínimo no qual a revocação é igual a 100% (*R-value*), (2) a precisão após o primeiro resultado relevante ser recuperado, (3) a precisão em um ponto fixo de revocação ou (4) a precisão após r vídeos serem recuperados (*top-r*). Outra medida de valor único é a medida-F (do inglês, *F-measure*), que combina a precisão e a revocação em um único número através da média harmônica [23, 32, 97]:

$$F_r = \frac{2 \times P_r \times R_r}{P_r + R_r}. \tag{2.12}$$

Se 100 vídeos são recuperados como resposta a uma consulta e 75 deles são relevantes, a precisão para $r = 100$ é $P_{100} = 75\%$. Nesse caso, a taxa de erros é $Te_{100} = 25\%$. Se, nessa mesma consulta, há 150 vídeos dentro dessa coleção que são relevantes, a revocação e, portanto, a taxa de acertos, para $r = 100$ é $R_{100} = Ta_{100} = 50\%$, uma vez que 75 dos 150 vídeos relevantes foram selecionados dentro dos primeiros 100 vídeos recuperados (*top-100*). Para tal situação, o resultado da medida-F é $F_{100} = 60\%$.

Capítulo 3

Segmentação Temporal

Fazer uso eficiente da informação de vídeo requer que os dados sejam armazenados de maneira organizada. Para isso, um vídeo deve ser dividido em um conjunto de unidades compreensíveis e gerenciáveis, de forma que o seu conteúdo seja consistente em termos de operações da câmera e eventos visuais. Isso tem sido o objetivo de uma área de pesquisa bastante conhecida, denominada segmentação de vídeo.

Diferentes técnicas têm sido propostas na literatura para lidar com a segmentação temporal de sequências de vídeo [16, 28, 74, 101, 129, 130, 167, 169, 175]. Muitos desses trabalhos de pesquisa estão centrados no domínio descomprimido [96, 104]. Embora os métodos existentes proporcionem uma qualidade elevada, a decodificação e a análise de uma sequência de vídeo são duas tarefas extremamente lentas e requerem uma enorme quantidade de espaço.

Neste capítulo, é apresentada uma nova abordagem para a segmentação temporal de sequências de vídeo que opera diretamente no domínio comprimido. Ela baseia-se na exploração de características visuais extraídas do fluxo de vídeo e em um algoritmo simples e rápido para detectar mudanças temporais. A melhoria da eficiência computacional torna essa técnica adequada para tarefas *online* [16].

O algoritmo proposto foi avaliado em um conjunto de testes com diferentes gêneros de vídeo e comparado com abordagens populares na literatura de segmentação temporal. Resultados de uma avaliação experimental sobre vários tipos de transições entre tomadas mostram que esse método é, ao mesmo tempo, bastante preciso e rápido.

O restante deste capítulo está organizado como segue. A Seção 3.1 introduz conceitos básicos e descreve trabalhos relacionados. A Seção 3.2 apresenta a abordagem proposta e mostra como aplicá-la na segmentação de sequências de vídeo. A Seção 3.3 discute o protocolo experimental em termos da coleção de vídeos e medidas de eficácia. A Seção 3.4 reporta os resultados experimentais e compara essa técnica com outros métodos. Por fim, são oferecidas considerações finais e direções para trabalhos futuros na Seção 3.5.

3.1 Revisão da Literatura

Uma tomada de vídeo é uma série de quadros inter-relacionados obtidos de uma única câmera. Na fase de edição de um vídeo, diversas tomadas são unidas para formar a sequência completa. Elas representam uma ação contínua no tempo e no espaço, em que mudanças no conteúdo da cena não podem ser percebidas [80].

Existem dois tipos diferentes de transições que podem ocorrer entre duas tomadas: transições abruptas (descontínuas), também conhecidas como cortes; e transições graduais (contínuas), que incluem movimentos de câmera (por exemplo, panorâmica horizontal, panorâmica vertical e zoom) e efeitos de edição de vídeo (por exemplo, aparecer, desaparecer, dissolver e limpar) [96].

Uma ampla revisão de métodos para resolver o problema de identificar a fronteira entre tomadas consecutivas é apresentada por Koprinska e Carrato [96] e Lienhart [104]. A maioria dos trabalhos de pesquisa existentes está centrada no domínio descomprimido. Embora essas técnicas ofereçam uma alta qualidade, elas gastam muito tempo e espaço para decodificar e analisar uma sequência de vídeo. Por essa razão, tais abordagens são inadequadas para as tarefas *online*.

A abordagem mais comum baseia-se na definição de medidas de similaridade entre quadros consecutivos. Métricas usuais empregam a diferença ponto a ponto [175] e histogramas de cor [130]. Rastreamento de características da imagem (por exemplo, bordas [167]) também pode ser utilizado para detectar o limite da tomada, pois elas tendem a desaparecer em um corte. Em uma abordagem diferente, padrões são detectados em uma subamostra bidimensional do vídeo, chamada fatia de vídeo ou ritmo visual [28, 74].

Uma vez que os dados de vídeo são geralmente disponibilizados na forma comprimida, é desejável processar diretamente o vídeo comprimido, sem decodificação. Isso permite economizar a elevada carga computacional da decodificação plena do fluxo de vídeo.

Diversos métodos para a segmentação de vídeo que manipulam diretamente o vídeo comprimido têm sido propostos para domínios específicos, tais como esportes, música e notícias [101, 129, 169]. Incidir sobre um domínio particular ajuda a reduzir os níveis de ambiguidade quando se analisa o conteúdo de um vídeo por meio da aplicação do conhecimento prévio do domínio durante o processo de análise [96].

Diferente das técnicas anteriores que operam diretamente no domínio comprimido, a abordagem proposta foi projetada para segmentar vídeos genéricos e, portanto, não usa nenhuma informação específica além do próprio conteúdo do vídeo.

3.2 Abordagem Proposta

No padrão de codificação MPEG (Seção 2.6), um vídeo é organizado em uma sequência de GOPs, os quais podem conter três tipos de quadros: intracodificado (quadro I),

predito (quadro P) e bidirecionalmente predito (quadro B). Em geral, o quadro I contém informação suficiente para caracterizar todo o conteúdo de um GOP.

A Figura 3.1 apresenta um diagrama de blocos do algoritmo proposto. Para cada tipo de quadro, é definido um módulo de detecção independente, que implementa critérios de detecção adequados a esse tipo. O módulo de decodificação parcial simplesmente analisa um vídeo de entrada e despacha dados para os três módulos de detecção. Por fim, um módulo de fusão de informações recolhe os resultados e determina as transições de vídeo.

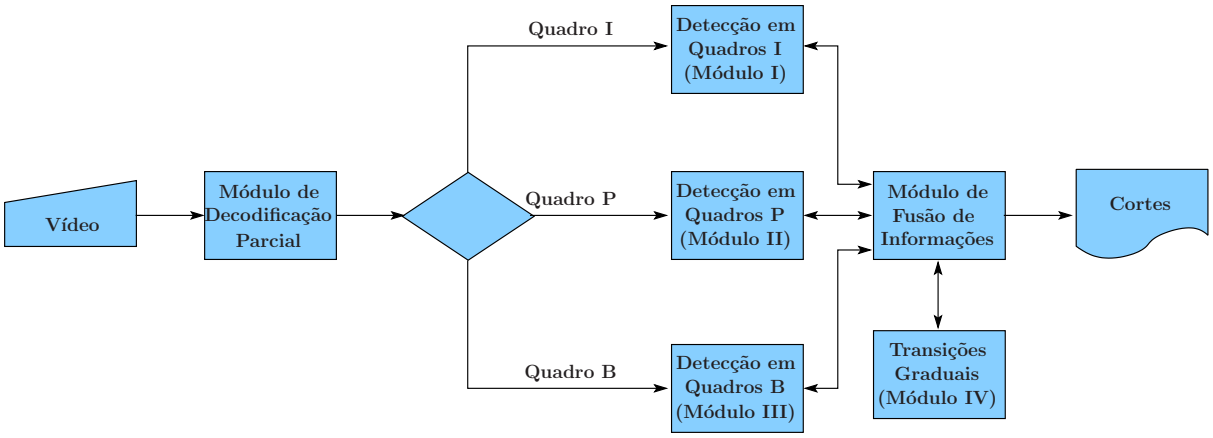


Figura 3.1: Diagrama de blocos do algoritmo de detecção de cortes.

Inicialmente, uma grande quantidade de GOPs é descartada por meio da comparação entre pares consecutivos de quadros I. Para isso, cada imagem DC (Seção 2.7) é convertida em um vetor de características de 256 dimensões calculando-se um histograma de cor. Ele é extraído da seguinte maneira: o espaço de cor YCbCr é dividido em 256 subespaços (cores), usando 16 intervalos do Y, 4 intervalos do Cb e 4 intervalos do Cr. O valor de cada dimensão desse vetor é a densidade de cada cor na imagem DC inteira.

Seja $\mathcal{H}^i$ o valor da i-ésima dimensão do histograma de cor $\mathcal{H}$. A similaridade entre os quadros I é dada pela intersecção de histogramas, uma métrica bastante conhecida, que é definida como [158]:

$$HI(\mathcal{H}_{t_1}, \mathcal{H}_{t_2}) = \frac{\sum_i \min(\mathcal{H}^i_{t_1}, \mathcal{H}^i_{t_2})}{\sum_i \mathcal{H}^i_{t_1}}, \tag{3.1}$$

na qual $\mathcal{H}_{t_1}$ e $\mathcal{H}_{t_2}$ são os histogramas de cor extraídos dos quadros I tomados nos instantes de tempo t_1 e t_2, respectivamente. Essa função retorna um valor real que varia de 0, para situações em que os histogramas são totalmente diferentes; até 1, quando eles são idênticos.

A Figura 3.2 mostra um exemplo de como esses valores estão distribuídos ao longo do tempo. Pode-se observar que há instantes de tempo nos quais o valor de dissimilaridade varia consideravelmente (correspondendo aos picos), enquanto há longos períodos nos

quais a variação é pequena (correspondendo às regiões bastante densas). Normalmente, os picos estão associados a movimentos bruscos na cena ou a fronteiras entre duas tomadas.

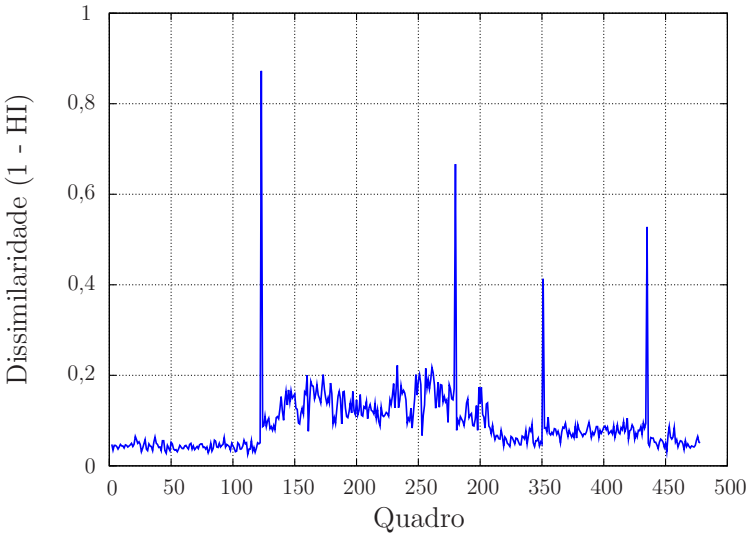


Figura 3.2: Dissimilaridades entre pares de quadros adjacentes do vídeo I do conjunto de testes.

O método proposto analisa somente os GOPs para os quais a intersecção de histogramas é abaixo de 0,85. Se eles são completamente intracodificados (isto é, compostos somente por quadros I), calcula-se a diferença normalizada ponto a ponto da luminância (Y) entre as imagens DC. Em seguida, um corte seco é declarado toda vez que o valor da similaridade for menor que 0,7 (ou seja, dissimilaridade menor que 0,3) e a diferença pontual for maior que 0,1 (Módulo I). A escolha desses valores é detalhada na Seção 3.3.

Caso contrário (isto é, os GOPs contêm quadros P e/ou B), o algoritmo de compensação de movimento é explorado a fim de detectar limites das tomadas. Para isso, examina-se o número de macroblocos intercodificados dentro de cada quadro P ou B. A ideia principal é que o algoritmo de compensação de movimento não consegue encontrar um bom casamento entre macroblocos de quadros I e/ou P vizinhos (anterior e/ou posterior) se o GOP estiver na fronteira entre duas tomadas, como ilustra a Figura 3.3.

Isso faz com que a maioria dos macroblocos dos quadros P sejam intracodificados (magenta) ao invés de intercodificados (azul). Se a razão entre o número de macroblocos intercodificados e o número total de macroblocos for menor que 0,9; há uma alta probabilidade de existir um corte na vizinhança desse quadro P. Nesse caso, analisam-se seus quadros B precedentes a fim de detectar o tipo e a localização dessa transição de vídeo. Para GOPs que não contêm quadros B, um corte seco é declarado se a percentagem de macroblocos intercodificados nesse quadro P for menor que 60% (Módulo II).

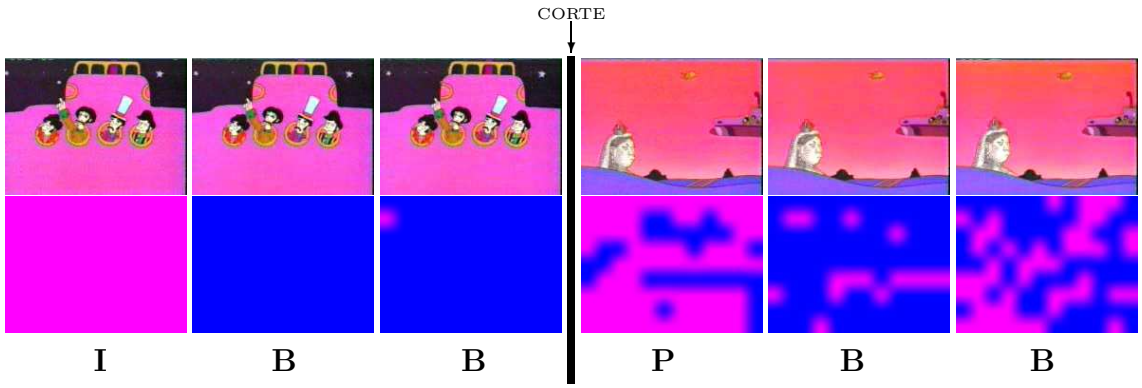


Figura 3.3: Seis quadros consecutivos de um GOP que inclui uma transição do vídeo *A* do conjunto de testes e o tipo de codificação de seus respectivos macroblocos: intracodificado (magenta) e intercodificado (azul).

Três comportamentos possíveis para os macroblocos dos quadros B são mostrados na Figura 3.4. A largura das setas indica a direção dominante para a compensação de movimento. Neste trabalho, um quadro B tem uma direção dominante se o número de macroblocos com compensação de movimento em uma determinada direção for o dobro daquele na direção oposta.

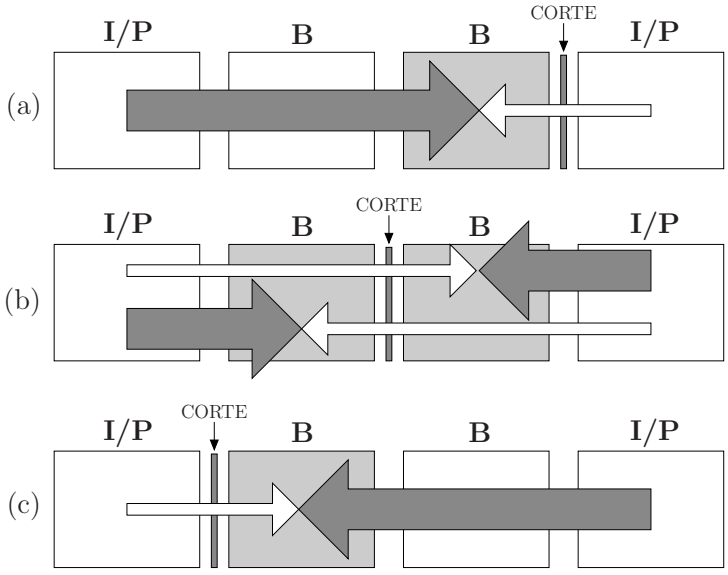


Figura 3.4: Os comportamentos possíveis para os macroblocos dos quadros B.

Se a maioria dos quadros B forem codificados com compensação de movimento para frente, uma transição abrupta é detectada entre o último quadro B e seu quadro âncora subsequente (I ou P) (Figura 3.4(a)). Por outro lado, se a maioria dos quadros B forem codificados com compensação de movimento para trás, uma transição abrupta é detectada entre o primeiro quadro B e o seu quadro âncora precedente (I ou P) (Figura 3.4(c)). Por

fim, se a predição adiantada é dominante na primeira metade dos quadros B e a predição atrasada é dominante na última metade dos quadros B, então um corte seco é detectado no meio dessa sequência (Figura 3.4(b)). A fim de evitar alarmes falsos, uma transição de vídeo é declarada somente se a percentagem de macroblocos intercodificados em tais quadros B for menor que 50% (Módulo III).

Se nenhuma das condições acima for satisfeita, existe a possibilidade de ocorrência de uma transição gradual. Isso pode ser verificado a partir da análise da variação do percentual de macroblocos intracodificados com compensação de movimento para frente ao longo dos quadros do GOP, como ilustra a Figura 3.5. No caso de transições graduais, esses valores formam um **planalto** (ou seja, um trapézio isósceles), como observado pela primeira vez por Yeo e Liu [169]. Uma vez que uma transição gradual tem uma certa duração, pelo menos 7 quadros devem ser envolvidos para se definir tal transição entre duas tomadas (Módulo IV).

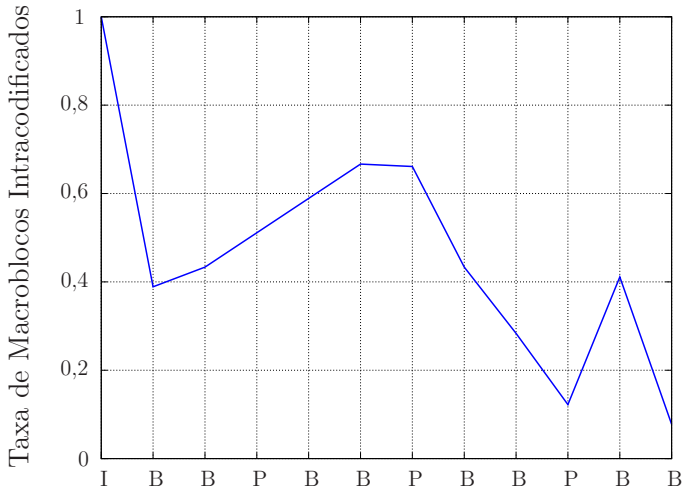


Figura 3.5: Exemplo do efeito planalto em um GOP que inclui uma transição gradual.

3.3 Protocolo Experimental

Experimentos foram realizados em um conjunto de vídeos com um gabarito conhecido. A fim de manter uma referência comum, foi utilizado o conjunto de testes[1] apresentado por Whitehead *et al.* [167]. Tal conjunto contém 10 sequências de vídeo, incluindo movimentos de câmera e objeto, diversos efeitos de edição e várias condições de iluminação. Todos os vídeos estão no formato MPEG-1 e distribuídos entre vários gêneros, como mostrado na Tabela 3.1. Esses vídeos são os mesmos utilizados por Pfeiffer *et al.* [130], Whitehead *et al.* [167], Bezerra e Leite [28] e Guimarães *et al.* [74].

[1]Todas as sequências de vídeo e os gabaritos estão disponíveis em `http://www.site.uottawa.ca/~laganier/videoseg/` (último acesso em 14 de outubro de 2011).

Tabela 3.1: As principais características de cada sequência de vídeo do conjunto de testes (adaptada de Bezerra e Leite [28]).

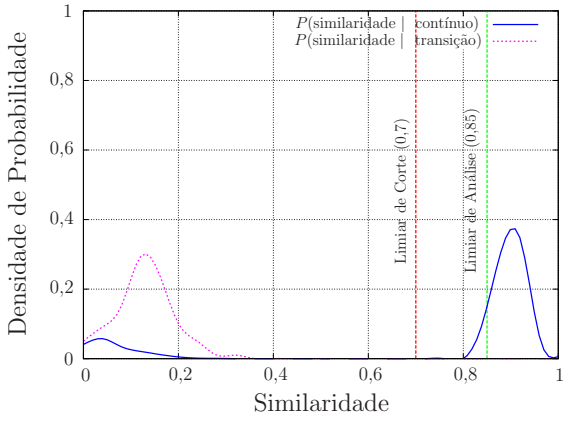
Vídeo	**Gênero**	**Dur. (s)**	**Dim.**	**Tam. GOP**	**Quadros**
A	desenho (baixa qualidade)	21	192 × 144	6	650
B	ação (movimento)	38	320 × 142	6	959
C	terror (preto e branco)	53	384 × 288	2	1619
D	drama (alta qualidade)	105	336 × 272	15	2632
E	ficção científica	17	384 × 288	2	536
F	comercial (efeitos)	7	160 × 112	15	236
G	comercial	16	384 × 288	1	500
H	comédia	205	352 × 240	12	5133
I	notícias	15	384 × 288	1	479
J	ação	36	240 × 180	12	873

Para selecionar os valores dos limiares empregados na detecção de mudanças temporais utilizando a abordagem proposta, todos os pares de quadros adjacentes dos vídeos desse conjunto foram manualmente anotados como contínuo ou de transição. Em seguida, calculou-se o valor da similaridade entre os quadros de cada par, o qual é medido por meio da intersecção de histogramas, no caso de quadros I; ou dado pela razão entre o número de macroblocos intercodificados e o número total de macroblocos, para quadros P ou B; conforme descrito na Seção 3.2.

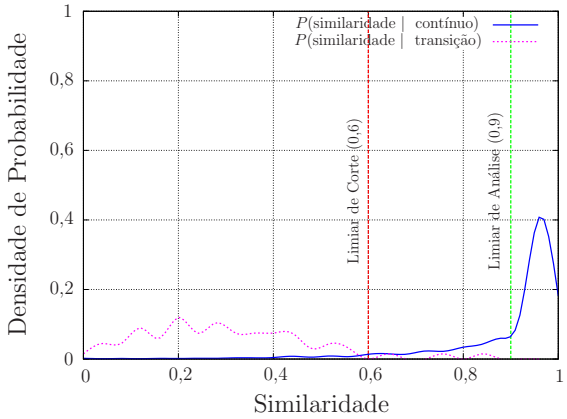
A Figura 3.6 apresenta a função densidade de probabilidade (do inglês, *Probability Density Function* – PDF) para a distribuição do valor da similaridade entre quadros contínuos e de transição. É importante notar que os limiares de análise, empregados em quadros I e P, correpondem ao menor valor de similaridade para o qual o valor da probabilidade acumulada da curva PDF de quadros contínuos é maior que 0,3. Já para a limiarização de corte, foi adotado o maior valor de similaridade para o qual o valor da soma dessas funções, ou seja, a sobreposição das curvas, é mínimo.

A eficácia do método proposto foi avaliada usando-se as métricas de precisão e revocação (Seção 2.11). Precisão (P) é a relação entre o número de posições temporais corretamente identificadas como cortes e o número total de posições temporais identificadas como cortes. Revocação (R) é a relação entre o número de posições temporais corretamente identificadas como cortes e o número total de cortes na sequência de vídeo. Entretanto, há uma relação de compromisso entre precisão e revocação, uma vez que pode-se aumentar a precisão ao custo de reduzir a revocação e vice-versa. Por isso, a medida-F (F) foi também empregada para avaliar a qualidade da segmentação temporal.

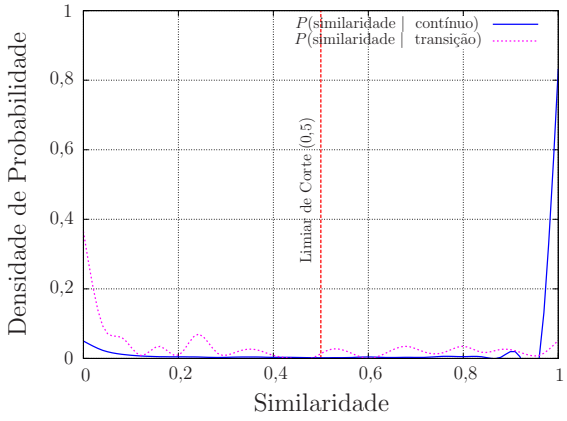
Os experimentos foram realizados em uma máquina equipada com um processador Intel Core 2 Quad Q6600 (4 núcleos de 2,4 GHz) e 2 GBytes de memória DDR3, executando o sistema operacional Ubuntu (núcleo linux 2.6.31) e o sistema de arquivos ext3.



(a) Quadros I



(b) Quadros P



(c) Quadros B

Figura 3.6: A função densidade de probabilidade (PDF) para a distribuição do valor da similaridade entre diferentes tipos de quadro de vídeo.

3.4 Resultados Experimentais

A Tabela 3.2 compara os resultados da técnica proposta com os reportados por quatro abordagens diferentes: (1) método baseado em histograma [130], (2) método baseado no rastreamento de características locais com ajuste automático de parâmetros [167], (3) ritmo visual com maior subsequência comum (do inglês, *Longest Common Subsequence* – LCS) [28] e (4) ritmo visual com agrupamento por k-médias [74]. As taxas médias de cada medida de qualidade correspondem à média ponderada dos resultados individuais, cujos pesos são dados pelo número total de cortes em cada vídeo. Além disso, os desvios padrões ponderados revelam a quantidade de dispersão com relação a esses valores.

Tabela 3.2: Comparação da precisão (P), revocação (R) e medida-F (F) obtida por abordagens diferentes para cada vídeo do conjunto de testes.

Vídeo	Método Proposto (dom. comprimido)			Ritmo Visual com k-médias			Ritmo Visual com LCS		
	P	R	F	P	R	F	P	R	F
A	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	0,857	0,923
B	**1,000**	**1,000**	**1,000**	0,500	**1,000**	0,667	0,096	**1,000**	0,176
C	0,891	**0,907**	**0,899**	0,662	**0,907**	0,766	0,635	0,870	0,734
D	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	0,971	0,985
E	0,815	0,786	0,800	0,828	0,857	0,842	0,676	0,821	0,742
F	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**
G	0,938	0,833	0,882	0,950	**1,000**	**0,974**	**1,000**	0,842	0,914
H	**0,974**	0,949	**0,961**	0,949	**0,974**	**0,961**	0,943	0,868	0,904
I	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	0,667	0,500	0,571
J	0,776	0,506	0,612	0,683	0,869	**0,765**	0,639	0,885	0,742
Avg.	0,882	0,786	0,825	0,793	0,923	**0,848**	0,745	0,878	0,789
Dev.	0,095	0,213	0,164	0,156	0,061	0,111	0,205	0,067	0,153

Vídeo	Método Proposto (dom. comprimido)			Rastreamento de Características			Histograma (MOCA)		
	P	R	F	P	R	F	P	R	F
A	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**
B	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	0,375	0,545
C	0,891	**0,907**	**0,899**	0,595	0,870	0,707	**0,936**	0,536	0,682
D	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	0,941	0,969
E	0,815	0,786	0,800	0,938	**1,000**	**0,968**	**0,955**	0,700	0,808
F	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**
G	0,938	0,833	0,882	0,810	0,944	0,872	**1,000**	0,667	0,800
H	**0,974**	0,949	**0,961**	0,895	0,895	0,895	0,971	0,895	0,932
I	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	0,500	0,667
J	0,776	0,506	0,612	0,497	**0,897**	0,637	**0,850**	0,395	0,540
Avg.	0,882	0,786	0,825	0,730	**0,924**	0,803	**0,932**	0,621	0,730
Dev.	0,095	0,213	0,164	**0,220**	0,054	0,157	0,063	**0,229**	**0,178**

De modo geral, os resultados indicam que o método proposto é robusto a diversas condições (por exemplo, velocidade de exibição, tamanho do quadro, duração total, etc), apresentando um desempenho similar a das técnicas comparadas. Na maioria das sequências de vídeo (7 de 10), a abordagem proposta atingiu o maior valor obtido para a medida-F.

A análise individual das sequências de vídeo mostra que os piores resultados obtidos pelo método proposto foram para os vídeos E e J. O mesmo comportamento também pode ser observado em outras técnicas. Esses vídeos são escuros e contêm pouca variação de luz e, portanto, tendem a apresentar uma baixa variância nos valores de similaridade.

Pela análise dos valores médios, pode-se notar a relação de compromisso entre as métricas de precisão e revocação. Esse efeito é notável para os métodos baseados em histograma e em rastreamento de características. O primeiro, obteve a melhor taxa média de precisão, porém a pior taxa média de revocação. O último, por outro lado, atingiu o mais alto valor médio de revocação, mas o mais baixo de precisão.

Comparando-se a medida-F de todas as estratégias analisadas, parece que a eficácia da abordagem proposta é, em média, inferior a do método baseado em ritmo visual com agrupamento por k-médias, o qual obteve o melhor resultado. Contudo, o desvios padrões mostram que essas técnicas exibem um desempenho similar e, portanto, as diferenças entre elas não são significativas (nível de confiança superior a 99,9%).

A principal vantagem da técnica proposta é a sua eficiência computacional. Uma vez que o tempo necessário para segmentar sequências de vídeo é dependente da arquitetura física (com componentes mais potentes, a velocidade computacional aumenta, reduzindo o tempo de execução) e os códigos-fonte de todos os métodos comparados não estão disponíveis, é impossível realizar uma comparação justa de desempenho em relação à abordagem apresentada.

Para avaliar a eficiência do método proposto, foi analisado o tempo por quadro gasto para executar todas as etapas de seu algoritmo, excluindo-se o tempo de decodificação parcial de cada quadro. A fim de garantir resultados estatisticamente sólidos, foram realizadas 10 replicações para cada vídeo.

Esse estudo mostra que a execução de todas as etapas da técnica proposta leva um tempo médio igual a, aproximadamente, 112 ± 38 microssegundos (nível de confiança superior a 99,9%). Para uso *online*, considerando um tempo máximo de espera igual a 39 segundos [34], esse método pode ser utilizado, em média, em vídeos de até 349.515 quadros (cerca de 194 minutos a uma taxa de 29,97 quadros por segundo). É importante lembrar que esses valores dependem do poder computacional da arquitetura física utilizada.

3.5 Conclusões

Neste capítulo, foi apresentada uma nova abordagem para detecção de cortes em vídeo que trabalha no domínio comprimido. Essa técnica baseia-se na exploração de características visuais extraídas do fluxo de vídeo e no uso de um algoritmo simples e rápido para detectar as transições do vídeo. Tal combinação faz com que o método proposto seja adequado para o uso *online*.

A técnica apresentada foi validada usando um conjunto de testes com diferentes gêneros de vídeo e comparada com abordagens populares na literatura de segmentação temporal. Resultados de uma avaliação experimental sobre vários tipos de transições de vídeo mostram que esse método apresenta alta eficácia e velocidade computacional.

Extensões futuras incluem a avaliação de outras características visuais e medidas de similaridade. Além disso, o método proposto pode ser ampliado para considerar características locais [4–6] e/ou análise de movimento [7–10].

Capítulo 4

Representação do Conteúdo Visual

Fazer uso eficiente da informação de vídeo requer que os dados sejam armazenados de maneira compacta. Para isso, um vídeo deve ser associado com características apropriadas a fim de permitir qualquer recuperação futura. Uma característica importante em sequências de vídeo é a mudança de intensidade temporal entre quadros sucessivos. Essa característica é geralmente atribuída ao movimento causado por objetos ou introduzido por operações de câmera. A estimativa do movimento da câmera é um dos aspectos mais importantes para caracterizar o conteúdo de sequências de vídeo [93].

Diferentes técnicas têm sido propostas na literatura para estimar o movimento da câmera a partir das sequências de vídeo [9, 10, 68, 93, 115, 127, 142, 149, 151]. Essas soluções baseiam-se tipicamente em uma abordagem de duas etapas: primeiro, identificando correspondências (movimento) entre quadros consecutivos e, em seguida, calculando uma forma paramétrica (modelo) que descreve o deslocamento entre esses quadros.

Os modelos mais populares na estimativa do movimento da câmera a partir dos vetores de movimento são os modelos afins de quatro [149] e de seis [93] parâmetros. Porém, esses parâmetros não estão diretamente relacionados às operações físicas da câmera [127].

Neste capítulo, é apresentada uma nova abordagem para estimar o movimento da câmera em sequências de vídeo com base em um modelo de operações de câmera. O método proposto gera o modelo de câmera usando combinações lineares de protótipos de fluxo ótico produzidos por cada operação de câmera [8, 9].

Para a avaliação dessa técnica, utilizou-se um conjunto de teste sintético e sequências de vídeo do mundo real, incluindo todas as operações básicas de câmera e muitas de suas combinações possíveis. Além disso, foram realizados vários experimentos para mostrar que essa técnica é mais eficaz do que as abordagens baseadas no modelo afim.

O restante deste capítulo está organizado como segue. A Seção 4.1 introduz conceitos básicos e descreve trabalhos relacionados. A Seção 4.2 apresenta a abordagem proposta e mostra como aplicá-la para estimar o movimento da câmera. A Seção 4.3 discute o

protocolo experimental em termos da coleção de vídeos e medidas de eficácia. A Seção 4.4 reporta os resultados experimentais e compara essa técnica com outros métodos. Por fim, são oferecidas considerações finais e direções para trabalhos futuros na Seção 4.5.

4.1 Revisão da Literatura

A maneira mais simples de representar o conteúdo de um vídeo consiste em avaliar a diferença ponto a ponto de quadros consecutivos [29, 73, 176]. Embora simples, esses métodos são muito sensíveis, pois capturam qualquer detalhe dos quadros, como mudanças de iluminação e movimentos de câmera e/ou objetos.

Uma alternativa a essas abordagens é utilizar uma estratégia global para codificar as propriedades de baixo nível (e.g., cor, forma e textura) em vetores de características [110, 131, 137]. Essas técnicas baseiam-se na premissa de que quadros com um fundo comum e com a mesma disposição de objetos apresentam uma pequena diferença em seus vetores.

Uma limitação dos métodos globais é que eles são muito sensíveis a pequenas mudanças na distribuição espacial. Por outro lado, as técnicas baseadas em correspondências pontuais sofrem da falta de robustez na presença de movimentos de câmera e objeto. Um compromisso entre essas abordagens consiste na comparação entre regiões (blocos) correspondentes [148, 168]. Todavia, o problema da escolha de uma escala apropriada para comparar as características relacionadas ao conteúdo visual ainda persiste. Utilizando uma escala menor aumenta-se a suscetibilidade do método em relação a movimentos de câmera e objeto, enquanto que uma escala maior reduz a sua sensibilidade a mudanças na distribuição espacial [48].

Para superar o problema do movimento de câmera e objeto, diversos métodos têm sido propostos na intenção de eliminar a diferença entre quadros causada por tais movimentos [56, 115, 127, 135, 142, 151, 178]. Muitas dessas técnicas baseiam-se na análise do fluxo ótico (Seção 2.3) para determinar movimento de câmera e objeto. A premissa dessas abordagens é que o movimento dominante do fluxo ótico de um vídeo pode ser associado à câmera, enquanto que o fluxo ótico localizado pode ser associado a objetos. Entretanto, o cálculo do fluxo ótico, o qual é normalmente obtido a partir do gradiente ou de métodos de comparação de blocos, é computacionalmente caro [68].

Uma vez que os dados de vídeo são geralmente disponibilizados na forma comprimida, é desejável processar diretamente o vídeo comprimido, sem decodificação. Alguns métodos para extrair o movimento da câmera que manipulam diretamente o vídeo comprimido têm sido propostos [68, 93, 149, 160]. Tais abordagens utilizam os vetores de movimento[1] do padrão de compressão dos vídeos como uma alternativa ao fluxo ótico, permitindo

[1]Na compressão do vídeo, um vetor de movimento é um vetor bidimensional que fornece o deslocamento entre pontos de um quadro decodificado e um quadro de referência.

economizar uma alta carga computacional sob duas perspectivas: decodificação completa do vídeo e cálculo do fluxo ótico [93].

Neste trabalho, o foco de interesse são algoritmos que operam no domínio comprimido. A seguir, são apresentadas três técnicas existentes, usadas como referência nos experimentos realizados. Tais métodos foram implementados e a eficácia da sua abordagem foi comparada na Seção 4.4.

Kim *et al.* [93] utilizam um modelo afim bidimensional capaz de detectar seis tipos de operações de câmera: panorâmica horizontal, panorâmica vertical, zoom, rotação, movimentação de objetos e câmera fixa. De antemão, vetores de movimento discrepantes (do inglês, *outliers*) são eliminados por um filtro de suavização. Os parâmetros do modelo são estimados por meio do ajuste dos mínimos quadrados sobre os dados remanescentes.

Smolic *et al.* [149] utilizam um modelo afim simplificado capaz de distinguir entre panorâmica horizontal, panorâmica vertical, zoom e rotação. Os autores usam o método estimador M [140] para lidar com dados corrompidos por vetores atípicos. Trata-se basicamente de uma técnica de mínimos quadrados ponderada, que reduz o efeito das observações aberrantes por meio de uma função de influência.

Gillespie e Nguyen [68] estenderam essa última abordagem a fim de melhorar a sua eficácia pela aplicação de um estimador robusto denominado mínima mediana dos quadrados (do inglês, *Least Median-of-Squares* – LMedS) [140] para obter os parâmetros do modelo da câmera e minimizar a influência dos dados discrepantes.

Em geral, essas soluções simplesmente encontram o modelo afim que melhor se ajusta ao padrão de movimento da câmera usando o método dos mínimos quadrados (do inglês, *least squares*) [140]. Contudo, os parâmetros afins não estão diretamente relacionados com as operações físicas da câmera (Seção 2.5). Diferente das técnicas anteriores que operam diretamente no domínio comprimido, a abordagem proposta estima o movimento da câmera a partir de um modelo de operações de câmera.

4.2 Abordagem Proposta

O método proposto gera o modelo de movimento de câmera a partir da combinação linear de protótipos de fluxo ótico produzidos por cada operação de câmera. Ele consiste de três etapas principais: (1) extração de características, (2) ajuste do modelo de movimento e (3) estimação robusta dos parâmetros. Nas subseções a seguir, cada uma dessas etapas é explicada com mais detalhes.

4.2.1 Extração de Características

Cada quadro de vídeo é dividido em uma sequência de macroblocos disjuntos. Para cada macrobloco, é estimado um vetor de movimento que aponta para um bloco similar

em um quadro âncora. Algoritmos de estimativa de movimento tentam encontrar a melhor correspondência entre blocos em termos de eficiência de compressão. Isso pode levar a vetores de movimento que não representam o movimento da câmera como um todo [59].

Os vetores de movimento são extraídos diretamente do fluxo de vídeo. Apenas os vetores de movimento dos quadros P (Seção 2.6) são processados pela abordagem proposta. Eles foram escolhidos porque, normalmente, quatro a cada quinze quadros de um vídeo são do tipo P e, portanto, a resolução temporal é suficiente para a maioria das aplicações. Além disso, a direção predita e a distância temporal dos vetores de movimento não são singulares em quadros B, aumentando a complexidade computacional.

4.2.2 Ajuste do Modelo de Movimento

Uma câmera projeta um ponto tridimensional do mundo real em um ponto bidimensional de uma imagem. O movimento da câmera pode ser limitado a um único movimento, tal como translação, rotação ou zoom; ou a alguma combinação desses três movimentos. Esses movimentos da câmera podem ser bem modelados utilizando-se poucos parâmetros.

Ao considerar que o campo visual da câmera é estreito, podem-se estabelecer modelos ideais de fluxo ótico, que são livres de ruídos, por meio de uma expressão numérica para o relacionamento dos vetores de movimento, criando um protótipo de fluxo ótico para cada operação de câmera.

Assim, uma aproximação do modelo real de fluxo ótico pode ser obtida a partir da combinação ponderada de modelos ideais, ou seja,

$$f = \mathcal{P} \times p + \mathcal{T} \times t + \mathcal{Z} \times z + \mathcal{R} \times r, \tag{4.1}$$

em que p, t, z e r são os protótipos gerados por cada operação básica de câmera, ou seja, panorâmica horizontal, panorâmica vertical, zoom e rotação, respectivamente.

O problema de identificar movimentos de câmera resume-se agora em obter uma estimativa para os parâmetros $\mathcal{P}$, $\mathcal{T}$, $\mathcal{Z}$ e $\mathcal{R}$ com base em um conjunto de vetores de movimento do modelo real $\{\hat{f}_i\}$ com cardinalidade i. Uma vez que as medições não são precisas, mas contaminadas por erros, não é possível assumir que todos eles vão se encaixar perfeitamente no modelo ideal. Assim, a melhor solução é computar o ajuste de mínimos quadrados dos dados. Para isso, o erro de modelagem é definido como a soma da norma quadrada dos vetores de discrepância entre os vetores de movimento do modelo real $\hat{f}_i$ e os vetores de movimento obtidos com o modelo ideal:

$$E = \sum_i \|(\mathcal{P} \times p_i + \mathcal{T} \times t_i + \mathcal{Z} \times z_i + \mathcal{R} \times r_i) - \hat{f}_i\|^2, \tag{4.2}$$

em que $\mathcal{P}$, $\mathcal{T}$, $\mathcal{Z}$ e $\mathcal{R}$ representam o movimento introduzido pelas operações de câmera, ou seja, panorâmica horizontal (ou translação horizontal), panorâmica vertical (ou translação vertical), zoom (ou translação para frente/trás) e rotação, respectivamente.

Para minimizar o erro de modelagem E, podem-se tomar suas derivadas com relação aos parâmetros do modelo

$$\begin{aligned}
\frac{\partial E}{\partial \mathcal{P}} &= \sum_i 2\ p_i^\top\ (\mathcal{P} \times p_i + \mathcal{T} \times t_i + \mathcal{Z} \times z_i + \mathcal{R} \times r_i - \hat{f}_i),\\
\frac{\partial E}{\partial \mathcal{T}} &= \sum_i 2\ t_i^\top\ (\mathcal{P} \times p_i + \mathcal{T} \times t_i + \mathcal{Z} \times z_i + \mathcal{R} \times r_i - \hat{f}_i),\\
\frac{\partial E}{\partial \mathcal{Z}} &= \sum_i 2\ z_i^\top\ (\mathcal{P} \times p_i + \mathcal{T} \times t_i + \mathcal{Z} \times z_i + \mathcal{R} \times r_i - \hat{f}_i),\\
\frac{\partial E}{\partial \mathcal{R}} &= \sum_i 2\ r_i^\top\ (\mathcal{P} \times p_i + \mathcal{T} \times t_i + \mathcal{Z} \times z_i + \mathcal{R} \times r_i - \hat{f}_i),
\end{aligned}$$

e igualá-las a zero, resultando em

$$\begin{aligned}
\sum_i (\mathcal{P}\ p_i^\top p_i + \mathcal{T}\ p_i^\top t_i + \mathcal{Z}\ p_i^\top z_i + \mathcal{R}\ p_i^\top r_i - p_i^\top \hat{f}_i) &= 0,\\
\sum_i (\mathcal{P}\ t_i^\top p_i + \mathcal{T}\ t_i^\top t_i + \mathcal{Z}\ t_i^\top z_i + \mathcal{R}\ t_i^\top r_i - t_i^\top \hat{f}_i) &= 0,\\
\sum_i (\mathcal{P}\ z_i^\top p_i + \mathcal{T}\ z_i^\top t_i + \mathcal{Z}\ z_i^\top z_i + \mathcal{R}\ z_i^\top r_i - z_i^\top \hat{f}_i) &= 0,\\
\sum_i (\mathcal{P}\ r_i^\top p_i + \mathcal{T}\ r_i^\top t_i + \mathcal{Z}\ r_i^\top z_i + \mathcal{R}\ r_i^\top r_i - r_i^\top \hat{f}_i) &= 0,
\end{aligned}$$

que pode ser escrito na forma matricial como

$$\begin{bmatrix}
\sum_i \langle p_i, p_i\rangle & \sum_i \langle p_i, t_i\rangle & \sum_i \langle p_i, z_i\rangle & \sum_i \langle p_i, r_i\rangle \\
\sum_i \langle t_i, p_i\rangle & \sum_i \langle t_i, t_i\rangle & \sum_i \langle t_i, z_i\rangle & \sum_i \langle t_i, r_i\rangle \\
\sum_i \langle z_i, p_i\rangle & \sum_i \langle z_i, t_i\rangle & \sum_i \langle z_i, z_i\rangle & \sum_i \langle z_i, r_i\rangle \\
\sum_i \langle r_i, p_i\rangle & \sum_i \langle r_i, t_i\rangle & \sum_i \langle r_i, z_i\rangle & \sum_i \langle r_i, r_i\rangle
\end{bmatrix}
\begin{pmatrix} \mathcal{P} \\ \mathcal{T} \\ \mathcal{Z} \\ \mathcal{R} \end{pmatrix}
=
\begin{pmatrix} \sum_i \langle p_i, \hat{f}_i\rangle \\ \sum_i \langle t_i, \hat{f}_i\rangle \\ \sum_i \langle z_i, \hat{f}_i\rangle \\ \sum_i \langle r_i, \hat{f}_i\rangle \end{pmatrix}, \qquad (4.3)$$

na qual $\langle u, v\rangle = u^\top v$ é o produto interno entre os vetores u e v.

Neste trabalho, define-se o modelo ideal de fluxo ótico para panorâmica horizontal (p), panorâmica vertical (t), zoom (z) e rotação (r), respectivamente, por:

$$p(x,y) = \begin{pmatrix} -1 \\ 0 \end{pmatrix},\ t(x,y) = \begin{pmatrix} 0 \\ -1 \end{pmatrix},\ z(x,y) = \begin{pmatrix} -x \\ -y \end{pmatrix},\ r(x,y) = \begin{pmatrix} y \\ -x \end{pmatrix},$$

em que (x, y) é um ponto amostrado da imagem, cujo sistema de coordenadas tem origem em seu centro.

A Figura 4.1 ilustra os protótipos de fluxo ótico gerados por panorâmica horizontal, panorâmica vertical, zoom e rotação, respectivamente. Essa figura expressa o módulo e a

direção dos vetores de movimento do fluxo ótico produzido por cada operação de câmera. É interessante notar as diferenças entre o fluxo ótico real e o ideal. Elas se devem a distorções causadas pela lente da câmera. Quanto maior for o campo visual da câmera, maior é a distorção produzida e, portanto, maior será o erro de modelagem E.

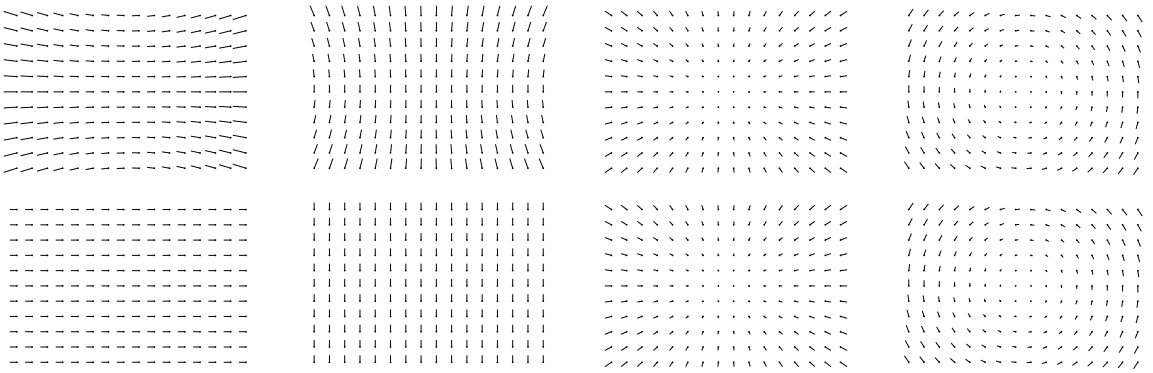


Figura 4.1: O fluxo ótico real (acima) e o ideal (abaixo) gerados por panorâmica horizontal, panorâmica vertical, zoom e rotação, respectivamente (da esquerda para a direita).

4.2.3 Estimação Robusta dos Parâmetros

A aplicação direta do método dos mínimos quadrados para estimar os parâmetros do modelo de câmera funciona bem apenas quando o número de observações aberrantes que não se desviam demais do movimento correto é pequeno. Contudo, o resultado é significativamente distorcido quando o número de vetores de movimento atípicos é grande ou o movimento é muito diferente daquele realizado pelas operações de câmera. Particularmente, se a sequência de vídeo apresenta movimentos de objetos independentes, um ajuste dos mínimos quadrados sobre o conjunto inteiro de dados disponíveis tentaria incluir todos os movimentos dos objetos visíveis em um único modelo de movimento.

Para reduzir a influência dos dados discrepantes, é aplicado um estimador robusto bastante conhecido, denominado RANSAC (do inglês, *RANdom SAmple Consensus*) [60]. A ideia é estimar repetidamente um conjunto de parâmetros do modelo usando pequenos subconjuntos de observações que são sorteados aleatoriamente a partir de todos os dados de entrada. Espera-se, assim, obter um subconjunto com amostras que fazem parte do mesmo modelo de movimento. Após o sorteio de cada subconjunto, seus parâmetros de movimento são determinados e a quantidade de dados de entrada consistente com esses parâmetros é contada. O conjunto de parâmetros do modelo com o maior suporte dos dados de entrada é considerado o modelo de movimento dominante visível na imagem.

4.3 Protocolo Experimental

A abordagem proposta foi avaliada em vídeos sintéticos com um gabarito conhecido. Para isso, um conjunto de teste sintético composto por quatro clipes de vídeo[2], no formato MPEG-4 (com resolução de 640 × 480 pontos e 14,98 quadros por segundo), foi criado a partir de cenas do POV-Ray bem texturizadas de um modelo realista de um escritório (Figura 4.2), incluindo todas as operações básicas de câmera e muitas de suas combinações possíveis. A principal vantagem é que os parâmetros do modelo de câmera podem ser totalmente controlados, permitindo verificar a qualidade de estimação de forma confiável.

Figura 4.2: Cenas do POV-Ray de um modelo realista de um escritório utilizado na criação do conjunto de teste sintético, com a câmera realizando uma panorâmica horizontal.

O primeiro passo na criação de vídeos sintéticos é definir a posição e a orientação da câmera em relação à cena. O mapeamento mundo-câmera é uma transformação rígida que leva coordenadas da cena $p_{\mathrm{w}} = (x_{\mathrm{w}}, y_{\mathrm{w}}, z_{\mathrm{w}})$ de um ponto para suas coordenadas da câmera $p_{\mathrm{c}} = (x_{\mathrm{c}}, y_{\mathrm{c}}, z_{\mathrm{c}})$. Esse mapeamento é dado por [109]

$$p_{\mathrm{c}} = Rp_{\mathrm{w}} + T, \tag{4.4}$$

em que R é a matriz de rotação 3×3 que define a orientação da câmera e T define a sua posição.

A matriz de rotação R é formada pela composição de três matrizes ortogonais especiais (conhecidas como *matrizes de rotação*)

$$R_x = \begin{bmatrix} \cos(\alpha) & 0 & -\sin(\alpha) \\ 0 & 1 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) \end{bmatrix}, R_y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\beta) & \sin(\beta) \\ 0 & -\sin(\beta) & \cos(\beta) \end{bmatrix}, R_z = \begin{bmatrix} \cos(\gamma) & \sin(\gamma) & 0 \\ -\sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

com α, β e γ representando os ângulos de rotação.

[2]Todos os clipes de vídeo e os gabaritos do conjunto de teste sintético estão disponíveis em `http://www.liv.ic.unicamp.br/~minetto/videos/` (último acesso em 14 de outubro de 2011).

A ação de uma câmera se movendo continuamente pode ser considerada como uma trajetória na qual as matrizes R e T mudam de acordo com o tempo t, em representação homogênea,

$$\begin{bmatrix} p_{\mathrm{c}} \\ 1 \end{bmatrix} = \begin{bmatrix} R(t) & T(t) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{\mathrm{w}} \\ 1 \end{bmatrix}. \tag{4.5}$$

Assim, a execução de operações de câmera, como panorâmica horizontal (mudanças graduais em R_x), panorâmica vertical (mudanças graduais em R_y), rotação (mudanças graduais em R_z) e zoom (mudanças graduais na distância focal f) pode ser definida por uma função $F(t)$ que retorna os parâmetros (α, β, γ e f) utilizados para mover a câmera no momento t. Neste trabalho, foi empregada a função suave e cíclica (Figura 4.3) [116]

$$F(t) = \mathcal{M} \times \frac{(1 - \cos(2\pi t/\mathcal{T}))(0.5 - t/\mathcal{T})}{0,263}, \tag{4.6}$$

na qual $\mathcal{M}$ é a magnitude máxima (amplitude da onda) e $\mathcal{T}$ é a duração total (comprimento de onda) do movimento da câmera. Todos os clipes de vídeo foram criados usando um valor de $\mathcal{M}$ igual 8^o de panorâmica horizontal (α), 3^o de panorâmica vertical (β), 90^o de rotação (γ) e $1,5$ de zoom (f).

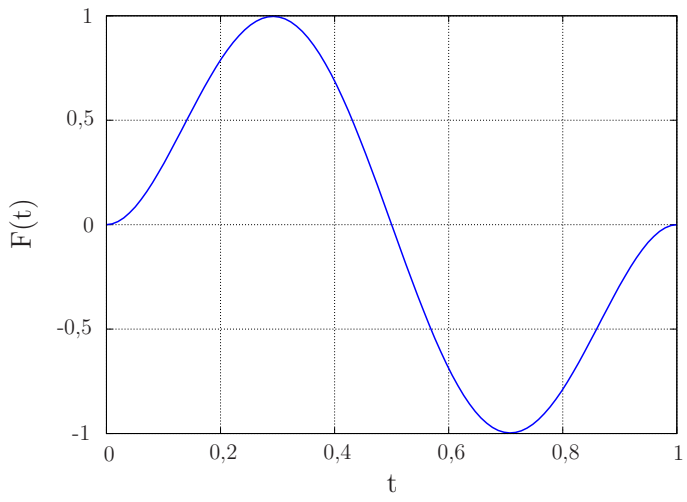


Figura 4.3: Função suave e cíclica utilizada para modelar o movimento da câmera.

A Figura 4.4 mostra as principais características de cada sequência de vídeo resultante (M_i). Os termos P, T, R e Z representam o movimento induzido pelas operações da câmera, ou seja, panorâmica horizontal, panorâmica vertical, zoom e rotação, respectivamente. Os vídeos M_3 e M_4 têm combinações de dois ou três tipos de operações. Para representar um cenário mais realista, os vídeos M_2 e M_4 foram modificados para incluir oclusões devido a movimentos de objetos.

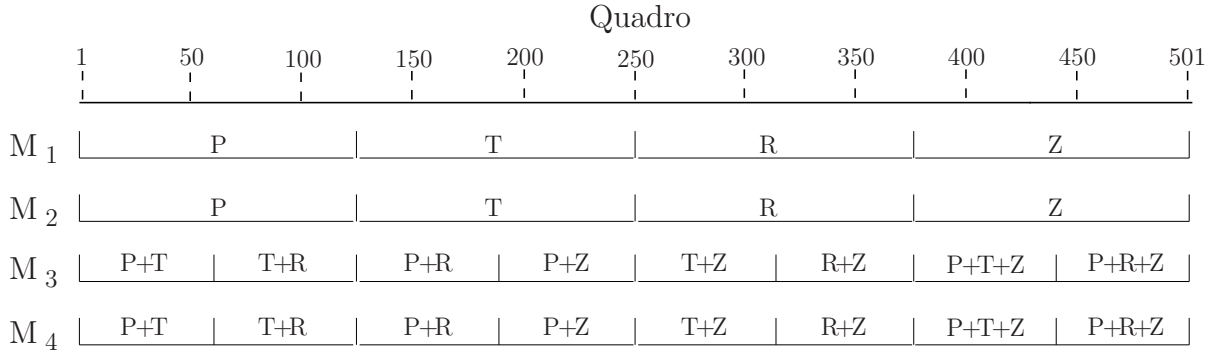


Figura 4.4: As principais características de cada sequência de vídeo (M_i) do conjunto de teste sintético.

Além disso, a iluminação artificial das lâmpadas e a reflexão dos raios de sol foram modificadas em algumas partes da cena de acordo com o movimento da câmera. E mais, todos os objetos da cena têm texturas complexas, que são muito similares às reais.

É importante notar o rigor adotado na intensidade dos movimentos de câmera. Movimentos rápidos da câmera e combinações de vários tipos de operações ao mesmo tempo, são raros de ocorrer. O objetivo desses casos é medir a resposta do algoritmo proposto em condições adversas, e não apenas com as operações básicas da câmera.

A eficácia do método proposto foi avaliada usando-se uma métrica bastante conhecida, denominada ZNCC (do inglês, *Zero-mean Normalized Cross Correlation*) [111], definida por

$$\mathrm{ZNCC}(\mathcal{F},\mathcal{G}) = \frac{\sum_t((\mathcal{F}(t)-\bar{\mathcal{F}})\times(\mathcal{G}(t)-\bar{\mathcal{G}}))}{\sqrt{\sum_t(\mathcal{F}(t)-\bar{\mathcal{F}})^2\times\sum_t(\mathcal{F}(t)-\bar{\mathcal{G}})^2}}, \tag{4.7}$$

na qual $\mathcal{F}(t)$ e $\mathcal{G}(t)$ são os parâmetros de câmera real e estimado, respectivamente, no momento t. Essa função retorna um valor real entre -1 e $+1$. Um valor igual a $+1$ indica uma estimativa perfeita; e -1, uma estimativa inversa.

A técnica proposta também foi avaliada em quatro sequências de vídeo do mundo real[3]. Tais vídeos foram filmados por uma câmera de mão do tipo DVR (Canon Optura 40) com zoom variável. Eles foram gravados no formato MPEG-4, a uma resolução de 320×240 pontos e 14,98 quadros por segundo.

A Tabela 4.1 resume as principais características de cada sequência de vídeo resultante (R_i). Todos os clipes de vídeo foram afetados pelo ruído natural da câmera e do ambiente. Os vídeos R_3 e R_4 apresentam oclusões resultantes de movimentos de objetos.

Devido à falta de um gabarito, experimentos nesse conjunto de vídeos analisaram a eficácia das técnicas baseadas em vetores de movimento em relação ao estimador bastante conhecido e preciso baseado na análise do fluxo ótico, proposto por Srinivasan *et al.* [151].

[3]Todas as sequências de vídeo do mundo real estão disponíveis em `http://www.liv.ic.unicamp.br/~minetto/videos/` (último acesso em 14 de outubro de 2011).

Tabela 4.1: As principais características de cada sequência de vídeo do mundo real (R_i).

Vídeo	**Quadros**	**Operações de Câmera**
R_1	338	P,T,R,Z
R_2	270	P,T,R,Z
R_3	301	P,T,R,Z
R_4	244	P,T,R,Z

Os experimentos foram realizados em uma máquina equipada com um processador Intel Core 2 Quad Q6600 (4 núcleos de 2,4 GHz) e 2 GBytes de memória DDR3, executando o sistema operacional Ubuntu (núcleo linux 2.6.31) e o sistema de arquivos ext3.

4.4 Resultados Experimentais

As Tabelas 4.2, 4.3, 4.4 e 4.5 reportam os resultados da abordagem proposta para o conjunto de teste sintético e compara-os com os resultados das técnicas apresentadas na Seção 4.1. Claramente, o uso de um modelo de operações de câmera para estimar o movimento de câmera é mais eficaz que de um modelo afim.

Tabela 4.2: Eficácia (ZNCC) obtida por diferentes abordagens no vídeo M_1.

Método	*Vertical*	*Horizontal*	*Rotação*	*Zoom*
Abordagem Proposta	**0,981419**	**0,996312**	**0,999905**	**0,964372**
Gillespie e Nguyen	0,970621	0,987444	0,999830	0,958607
Smolic *et al.*	0,950911	0,994171	0,999199	0,949852
Kim *et al.*	0,649087	0,912365	0,994067	0,858090

Tabela 4.3: Eficácia (ZNCC) obtida por diferentes abordagens no vídeo M_2.

Método	*Vertical*	*Horizontal*	*Rotação*	*Zoom*
Abordagem Proposta	**0,981029**	**0,995961**	**0,999913**	**0,965994**
Gillespie e Nguyen	0,972189	0,988062	0,999853	0,959516
Smolic *et al.*	0,936479	0,991438	0,999038	0,949367
Kim *et al.*	0,633559	0,821266	0,986408	0,865052

Tabela 4.4: Eficácia (ZNCC) obtida por diferentes abordagens no vídeo M_3.

Método	*Vertical*	*Horizontal*	*Rotação*	*Zoom*
Abordagem Proposta	**0,587136**	**0,950760**	**0,999624**	**0,956845**
Gillespie e Nguyen	0,575178	0,931957	0,999521	0,954215
Smolic *et al.*	0,559669	0,940782	0,999037	0,951701
Kim *et al.*	0,501764	0,942563	0,997240	0,942588

Tabela 4.5: Eficácia (ZNCC) obtida por diferentes abordagens no vídeo M_4.

Método	*Vertical*	*Horizontal*	*Rotação*	*Zoom*
Abordagem Proposta	**0,592071**	**0,949922**	**0,999659**	**0,956440**
Gillespie e Nguyen	0,577467	0,932568	0,999545	0,954286
Smolic *et al.*	0,557849	0,940886	0,998920	0,951640
Kim *et al.*	0,498081	0,941956	0,997334	0,944102

Vale a pena notar que a eficácia alcançada por todos os métodos é razoavelmente reduzida para movimentos verticais na presença de vários tipos concomitantes de operações de câmera. A principal razão para esses resultados é que, apesar dos vetores de movimento melhorarem o desempenho em relação ao tempo de execução, muitas vezes eles não modelam adequadamente o movimento real [59].

As Tabelas 4.6, 4.7, 4.8 e 4.9 reportam os resultados da abordagem proposta para as sequências de vídeo do mundo real e compara-os com os resultados das técnicas apresentadas na Seção 4.1. De fato, a estratégia baseada no modelo de operações de câmera supera a baseada no modelo afim para estimar o movimento de câmera.

Tabela 4.6: Eficácia (ZNCC) obtida por diferentes abordagens no vídeo R_1.

Método	*Vertical*	*Horizontal*	*Rotação*	*Zoom*
Abordagem Proposta	**0,986287**	**0,986294**	**0,987545**	**0,982227**
Gillespie e Nguyen	0,982345	0,978892	0,980464	0,964398
Smolic *et al.*	0,984085	0,976381	0,977135	0,966526
Kim *et al.*	0,982998	0,884470	0,795713	0,944286

Tabela 4.7: Eficácia (ZNCC) obtida por diferentes abordagens no vídeo R_2.

Método	*Vertical*	*Horizontal*	*Rotação*	*Zoom*
Abordagem Proposta	**0,914379**	**0,954113**	**0,929268**	**0,684219**
Gillespie e Nguyen	0,863166	0,931218	0,899512	0,357249
Smolic *et al.*	0,874244	0,952316	0,919447	0,611227
Kim *et al.*	0,899520	0,901673	0,846316	0,670006

Tabela 4.8: Eficácia (ZNCC) obtida por diferentes abordagens no vídeo R_3.

Método	*Vertical*	*Horizontal*	*Rotação*	*Zoom*
Abordagem Proposta	**0,964425**	**0,960878**	**0,957735**	**0,454204**
Gillespie e Nguyen	0,949270	0,931442	0,927145	0,379836
Smolic *et al.*	0,957662	0,953751	0,956303	0,444741
Kim *et al.*	0,954097	0,912121	0,924798	0,368722

Tabela 4.9: Eficácia (ZNCC) obtida por diferentes abordagens no vídeo R_4.

Método	*Vertical*	*Horizontal*	*Rotação*	*Zoom*
Abordagem Proposta	**0,976519**	**0,958020**	**0,927920**	**0,577974**
Gillespie e Nguyen	0,948314	0,902511	0,851247	0,308588
Smolic *et al.*	0,969314	0,956417	0,903442	0,523507
Kim *et al.*	0,969613	0,938639	0,839906	0,474439

É importante observar que o modelo de operações de câmera identifica as operações de câmera melhor do que os parâmetros do modelo afim. Por exemplo, considerando as operações de zoom no vídeo R_4, o método proposto é mais de 10% (≈ 5 pontos percentuais) mais eficaz que a melhor técnica baseada no modelo afim.

Foi necessário menos de um segundo para processar a sequência inteira de cada vídeo (R_i) usando os métodos baseados em vetores de movimento (Abordagem Proposta, Gillespie e Nguyen [68], Smolic *et al.* [149] e Kim *et al.* [93]). Por outro lado, o método baseado na análise do fluxo ótico (Srinivasan *et al.* [151]) exigiu uma magnitude de quase um segundo por quadro, usando a mesma arquitetura física.

Mais especificamente, o resultado de 10 replicações de um experimento para analisar o tempo por quadro gasto pelo método proposto, no processamento de cada sequência de vídeo do mundo real (R_i), mostrou que a execução de todas as etapas de seu algoritmo, excluindo-se o tempo de decodificação parcial de cada quadro, leva um tempo médio igual a, aproximadamente, 284 ± 23 microssegundos (nível de confiança superior a 99,9%).

4.5 Conclusões

Neste capítulo, foi apresentada uma nova abordagem para a estimativa do movimento da câmera em sequências de vídeo. Essa técnica baseia-se na combinação linear de modelos ideais de fluxo óptico produzidos por cada operação de câmera. Tais modelos identificam as operações de câmera melhor do que os parâmetros do modelo afim.

A técnica apresentada foi validada usando vídeos sintéticos e reais, incluindo todas as operações básicas de câmera e muitas de suas combinações possíveis. Experimentos mostram que o uso de um modelo de operações de câmera para estimar o movimento da câmera em sequências de vídeo é mais eficaz que de um modelo afim.

Extensões futuras incluem a distinção entre operações translacionais e rotacionais (Seção 2.5). Além disso, abordagens baseadas em vetores de movimento normalmente suportam uma quantidade limitada de movimento da cena e, portanto, o método proposto pode ser ampliado para considerar características locais [108], as quais são mais robustas a uma quantidade substancial de distorções afins [7, 10].

Capítulo 5

Medida de Similaridade

Fazer uso eficiente da informação de vídeo requer o desenvolvimento de ferramentas poderosas para identificar rapidamente vídeos semelhantes em um banco de dados com um grande número de entradas. Para isso, um vídeo deve ser associado a uma representação compacta capaz de resumir o seu conteúdo, a qual é geralmente conhecida como assinatura de vídeo. A seguir, é necessário ainda definir uma medida de similaridade para comparar sequências de vídeo a partir de suas assinaturas.

Existem duas questões importantes que devem ser levadas em conta nessa tarefa: robustez e discriminância. Robustez é a quantidade de inconsistência dos dados tolerada pelo sistema antes da ocorrência de um falso positivo. Discriminância é a capacidade do sistema de rejeitar dados irrelevantes e reduzir os falsos positivos [92].

Diferentes técnicas têm sido propostas na literatura para resolver o problema da comparação entre sequências de vídeo [17, 49, 77, 84, 92, 98]. Muitos desses trabalhos de pesquisa estão centrados no domínio descomprimido. Embora os métodos existentes proporcionem uma qualidade elevada em termos de robustez e discriminância, a decodificação e a análise de uma sequência de vídeo são duas tarefas extremamente lentas e requerem uma enorme quantidade de espaço.

Neste capítulo, é apresentada uma nova abordagem para a comparação de sequências de vídeo que atua diretamente no domínio comprimido. Ela baseia-se no reconhecimento de padrões de movimento extraídos do fluxo de vídeo, que são acumulados para formar um histograma normalizado. Essa abordagem computacionalmente simples é robusta a várias distorções e transformações. A melhora da eficiência computacional torna essa técnica adequada para enormes coleções de dados de vídeo [17].

O algoritmo proposto foi avaliado em cerca de 11.500 vídeos (400 horas) de um conjunto de dados do TRECVID 2010 (IACC.1) e comparado com abordagens recentes na literatura de detecção de similaridade entre vídeos. Resultados de uma avaliação experimental sobre vários tipos de transformações de vídeo mostram que esse método apresenta alta eficácia

e velocidade computacional na identificação de vídeos similares.

O restante deste capítulo está organizado como segue. A Seção 5.1 introduz conceitos básicos e descreve trabalhos relacionados. A Seção 5.2 apresenta a abordagem proposta e mostra como aplicá-la na comparação de sequências de vídeo. A Seção 5.3 discute o protocolo experimental em termos da coleção de vídeos e medidas de eficácia. A Seção 5.4 reporta os resultados experimentais e compara essa técnica com outros métodos. Por fim, são oferecidas considerações finais e direções para trabalhos futuros na Seção 5.5.

5.1 Revisão da Literatura

No capítulo anterior, foram apresentados métodos para codificar as propriedades de baixo nível de um quadro em vetores de características. Dessa forma, a similaridade visual entre dois vídeos pode ser medida a partir de uma métrica definida no espaço desses vetores. A premissa básica nesse processo é que o conteúdo visual a ser analisado é homogêneo para uma mesma característica de interesse. Uma vez que um vídeo é tipicamente uma coleção de tomadas com conteúdos muito diferentes, essa tarefa deve ser modelada como um conjunto ou uma série temporal de vetores de características, geralmente com um vetor por tomada [50].

Muitos esforços nessa área têm se concentrado no problema de procurar por um segmento pequeno, tal como um comercial de televisão, dentro de uma sequência longa [79, 105, 117]. Ao estender a medida de similaridade para todo o vídeo, o desafio é definir uma única medida para avaliar a similaridade entre duas sequências inteiras. Entre várias propostas que podem ser encontradas na literatura para tal feito, vale destacar os trabalhos de Adjeroh *et al.* [1], Chang *et al.* [40], Lienhart *et al.* [106] e Naphade *et al.* [121].

Um passo comum partilhado por todas as abordagens anteriores consiste na comparação de características similares entre duas sequências de vídeo, o que geralmente requer uma pesquisa em parte ou em todo o vídeo. Portanto, o cálculo completo dessas medidas exige o armazenamento de todo o vídeo e sua complexidade computacional é ao menos linear no tamanho da sequência. Aplicar esses cálculos na busca de conteúdos semelhantes dentro de um BD com uma enorme quantidade de sequências de vídeo pode ser inviável [50].

Por outro lado, o valor preciso de uma medida de similaridade não é normalmente necessário. Como os vetores de características representam modelos ideais e não capturam completamente o processo de como a similaridade é julgada no sistema visual humano [143], muitas aplicações requerem apenas uma aproximação do valor da similaridade inerente. Consequentemente, é desnecessário manter representações fidedignas dos vetores de características, e esquemas de aproximação podem ser utilizados para aliviar a alta complexidade computacional [51, 71, 87].

Logo, cada sequência de vídeo pode ser primeiro resumida em representações compactas de tamanho fixo. Dessa maneira, a similaridade entre duas sequências de vídeo pode ser aproximada a partir da comparação de suas respectivas representações. Existem dois tipos de técnicas de compactação para a aproximação da similaridade: de primeira ordem e de ordem superior [50].

As técnicas de ordem superior representam todos os vetores de características de um vídeo como uma distribuição estatística. Esses métodos são úteis na recuperação semântica, já que eles são altamente adaptativos e robustos a pequenas perturbações. As técnicas de primeira ordem, por sua vez, representam um vídeo em um pequeno conjunto de vetores de características representativos. Uma abordagem consiste em utilizar algoritmos de agrupamento para identificar um número fixo de vetores de características que minimiza a distância em relação aos demais [50].

Neste trabalho, o foco de interesse são algoritmos que extraem descrições globais do vídeo. Essas abordagens têm alcançado a mesma precisão que os métodos baseados em características locais [98]. A seguir, são apresentadas quatro técnicas recentes na literatura de detecção de similaridade entre vídeos. Tais métodos foram implementados e a eficácia da sua abordagem foi comparada na Seção 5.4.

Hampapur e Bolle [76] introduziram uma assinatura baseada em movimento que combina características de cor e forma. Entretanto, essas características não são robustas a variações não-lineares de intensidade. Para contornar esse problema, Mohan [117] propôs o uso de medidas ordinais para comparar sequências de vídeo. A medida ordinal é extraída a partir de um conjunto discreto ordenado. A relação entre dois valores do conjunto não é importante, somente sua ordem relativa é relevante. A ordem relativa entre esses valores é expressa por suas posições dentro do conjunto. Essas posições são obtidas ordenando-se o conjunto em ordem crescente e rotulando-se seus valores com números inteiros. Tal estratégia foi inicialmente adotada no casamento estéreo entre imagens [31], mostrando resultados promissores. Mais tarde, Hampapur e Bolle [77] compararam várias abordagens para descrever uma sequência de vídeo, incluindo métodos baseados em movimento, ordem e cor. Eles descobriram que as medidas ordinais são superiores às demais.

Hua *et al.* [84] aplicaram uma medida ordinal para comparar sequências de vídeo. Inicialmente, cada quadro é dividido em 3×3 regiões de mesmo tamanho. Em seguida, uma matriz de ordinais é obtida pela ordenação da média dos valores de intensidade em cada região. Por fim, programação dinâmica é usada para comparar essas assinaturas.

Kim e Vasudev [92] mostraram que uma matriz de ordinais em termos de quadros particionados em 2×2 regiões é mais robusta aos diferentes formatos de exibição. Além disso, eles introduziram a informação temporal na medida de similaridade pela simples associação dos valores $-1/0/1$ para representar a mudança de valores de intensidade entre regiões correspondentes em quadros adjacentes de um vídeo.

Chen e Stentiford [49] observaram que medidas ordinais ao longo da série temporal pode descrever melhor as variações temporais das sequências de vídeo. Assim, eles propuseram o uso da ordenação das regiões ao longo do alinhamento temporal, em vez de ao longo do arranjo espacial dos quadros de vídeo.

A principal desvantagem desses algoritmos é que a assinatura é proibitiva em termos de espaço de armazenamento, e sua comparação usando uma medida de similaridade baseada em uma abordagem quadro a quadro é impraticável em bases de dados muito grandes.

5.2 Abordagem Proposta

Dados de vídeo contêm uma grande quantidade de informação redundante. Para economizar tempo computacional, muitos padrões de codificação de vídeo (por exemplo, MPEG-1/2/4) dividem o fluxo de vídeo em unidades básicas, denominadas GOPs. Em geral, o quadro I contém informação suficiente para caracterizar todo o conteúdo do GOP.

Conforme detalhado na Seção 2.6, a compressão dos quadros I é obtida dividindo-se a imagem original em blocos de 8×8 pontos e transformando os valores de intensidade de cada bloco em 64 coeficientes da DCT. Ao se extrair o termo DC dos quatro blocos da luminância (Y), podem-se utilizar esses valores para gerar uma impressão digital de cada macrobloco. Esse processo é ilustrado na Figura 5.1.

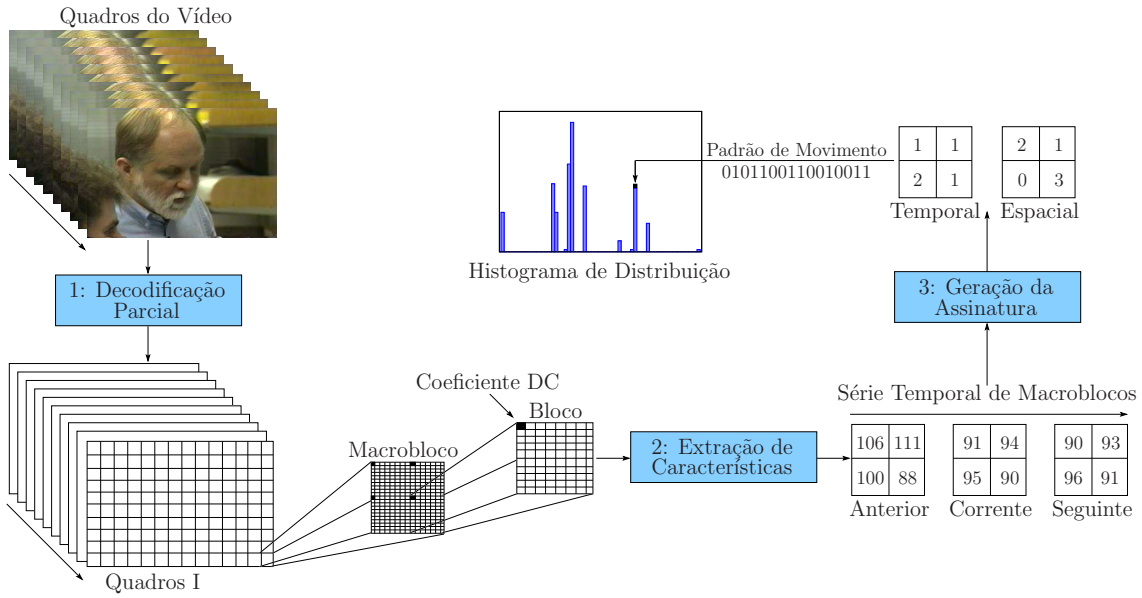


Figura 5.1: Uma visão geral do método proposto.

Inicialmente, cada macrobloco é representado pelo valor médio de intensidade de seus quatro blocos de luminância (etapa 1). A partir dessa descrição, podem-se classificar os macroblocos em diferentes classes: **movendo** ou **parado**. Um macrobloco é classificado como **movendo** se pelo menos um dos blocos de macroblocos correspondentes em três quadros consecutivos (anterior, corrente e seguinte) tem um valor de intensidade diferente. Caso contrário, ele é classificado como **parado**.

Em seguida, uma matriz de ordinais é obtida pela ordenação dos valores de intensidade dos macroblocos (etapa 2). Essa estratégia é empregada para gerar uma assinatura espacial dos quatro blocos de um macrobloco e uma assinatura temporal de blocos correspondentes em três quadros consecutivos (anterior, corrente e seguinte).

Cada combinação possível das medidas ordinais é tratada como um padrão individual de 16-*bits*, ou seja, 2-*bits* para cada elemento das matrizes de ordinais de tamanho 2×2. Por fim, o padrão espaço-temporal de todos os macroblocos da sequência de vídeo classificados como *movendo* são acumulados para formar um histograma normalizado (etapa 3). Dessa forma, pode-se efetivamente combinar abordagens estruturais e estatísticas: os padrões espaço-temporais detectam características de movimento, cuja distribuição inerente é estimada pelo histograma.

Portanto, pode-se representar sequências de vídeo utilizando apenas $2^{16} = 64$ kBytes. Na prática, esse histograma é geralmente bastante esparso, contendo apenas uma pequena percentagem de elementos diferentes de zero (5% em média). Assim, técnicas de compressão podem ser aplicadas para melhorar a utilização das unidades de armazenamento.

A comparação entre histogramas pode ser realizada por qualquer função de distância (Seção 2.9), tais como as distâncias Manhattan (L_1) ou Euclidiana (L_2). A principal vantagem das distâncias vetoriais é a sua eficiência computacional na comparação de histogramas. Além disso, elas permitem o uso de métodos de acesso espaciais ou métricos [12, 14] para acelerar o processamento de consultas.

5.3 Protocolo Experimental

Experimentos foram realizados em cerca de 11.500 vídeos (400 horas) de um conjunto de dados do TRECVID 2010 (IACC.1)[1], projetado para a tarefa de detecção de vídeo-cópias. Esses vídeos estão distribuídos entre vários gêneros (por exemplo, esportes, notícias, shows de TV, vídeos caseiros, etc). Eles encontram-se categorizados em duas classes: sequências curtas, contendo aproximadamente 8.300 vídeos (200 horas) com duração entre 10 segundos e 3,5 minutos; e sequências longas, composta por quase 3.200 vídeos (200 horas) cuja duração varia de 3,6 a 4,1 minutos.

[1] `http://www-nlpir.nist.gov/projects/tv2010/` (último acesso em 14 de outubro de 2011).

A fim de testar falsos alarmes, cada classe foi aleatoriamente dividida em dois subconjuntos: 75% dos vídeos foram usados como uma base de referência e o restante, como dados de ruído. Em seguida, selecionou-se aleatoriamente cerca de 1% dos vídeos de cada subconjunto para serem utilizados como consultas, totalizando 117 sequências de vídeo (80 curtas e 37 longas). Depois disso, 10 transformações foram aplicadas a cada vídeo de consulta. Assim, um total de 1170 cópias (800 de curta duração e 370 de longa duração) foram inseridas como vídeos de consulta.

A Figura 5.2 ilustra as diferentes transformações aplicadas em cada vídeo de consulta, a saber: contraste aumentado em 25%, contraste diminuído em 25%, tamanho diminuído para 80%, tamanho aumentado para 120%, desfoque gaussiano com raio 2, efeito imagem em caixa (do inglês, *letter-box*), corte de 5% com borda preta, zoom ampliado para 120%, zoom reduzido para 80% com borda preta e inserção de um pequeno logotipo. Nessa figura, são apresentados os resumos do vídeo original e das suas cópias transformadas. Esses resumos de vídeo foram produzidos usando a abordagem apresentada no Capítulo 7.

A eficácia do método proposto foi avaliada usando-se curvas $P \times R$ (Seção 2.11). Precisão é a razão entre o número de vídeos corretamente identificados como cópias e o número total de vídeos identificados como cópias. Revocação é a razão entre o número de vídeos corretamente identificados como cópias e o número total de vídeos copiados.

O desempenho da abordagem apresentada também foi avaliado por meio de curvas ROC (Seção 2.11). Esse é um gráfico da taxa de falsos negativos em relação a taxa de falsos positivos. A primeira, é a razão entre o número de vídeos copiados que não foram identificados como cópias e o número total de vídeos copiados; enquanto a última, é a razão entre o número de vídeos legítimos identificados como cópias e o número total de vídeos legítimos.

Uma sequência de vídeo é identificada como uma cópia, se a sua distância para o vídeo de consulta for menor do que um limiar normalizado, variando de 0 até 1. Diferentes valores foram utilizados para esse limiar na produção dessas curvas. Esses valores referem-se a pontos específicos de revocação (0%, 10%, . . . , 90%, 100%) para os quais foram obtidas as demais medidas de eficácia.

Os experimentos foram realizados em uma máquina equipada com um processador Intel Xeon E5620 (8 núcleos de 2,4 GHz) e 16 GBytes de memória DDR2, executando o sistema operacional Gentoo (núcleo linux 2.6.32) e o sistema de arquivos ext3.

(a) vídeo original

(b) contraste aumentado em 25%

(c) contraste diminuído em 25%

(d) tamanho diminuído para 80%

(e) tamanho aumentado para 120%

(f) desfoque gaussiano com raio 2

(g) efeito imagem em caixa

(h) corte de 5% com borda preta

(i) zoom ampliado para 120%

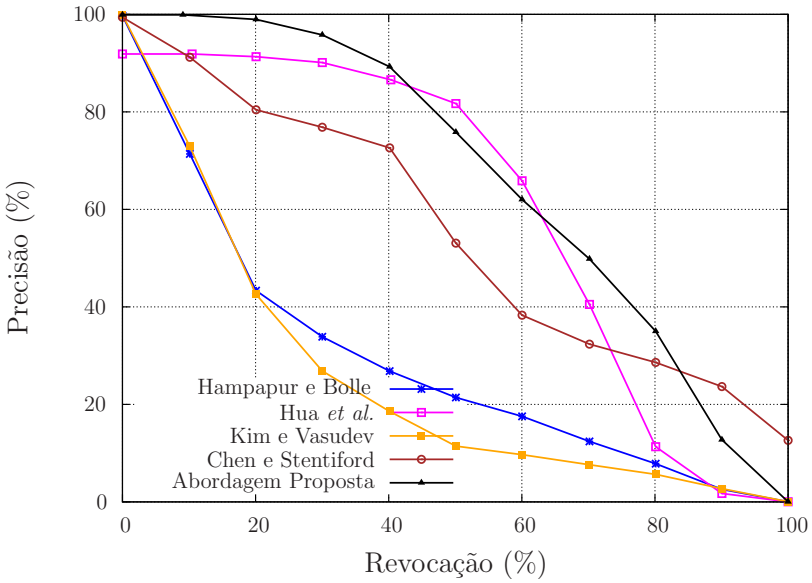
(j) zoom reduzido para 80% com borda preta

(k) inserção de um pequeno logotipo

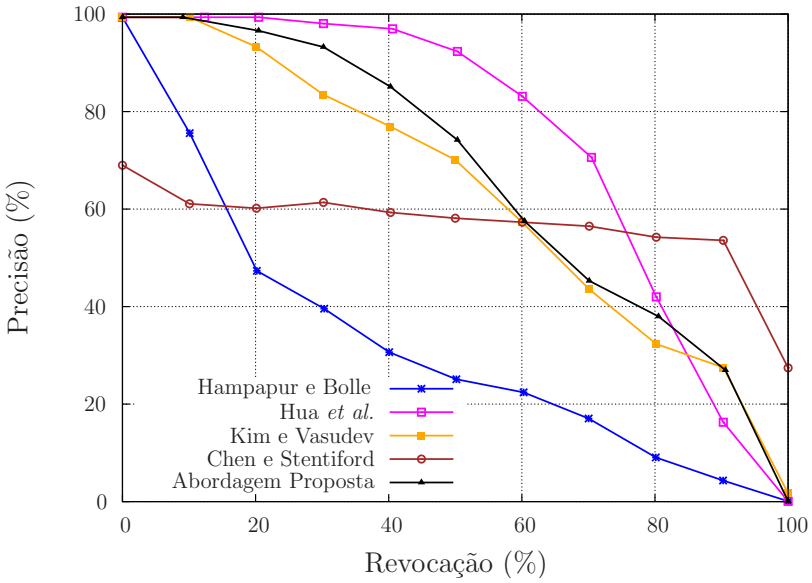
Figura 5.2: Exemplos de resumos do vídeo original e das suas cópias transformadas.

5.4 Resultados Experimentais

A Figura 5.3 compara a abordagem proposta com as técnicas apresentadas na Seção 5.1. De modo geral, resultados indicam que o método proposto é robusto a diversas transformações, mostrando uma precisão comparável a das soluções atuais.



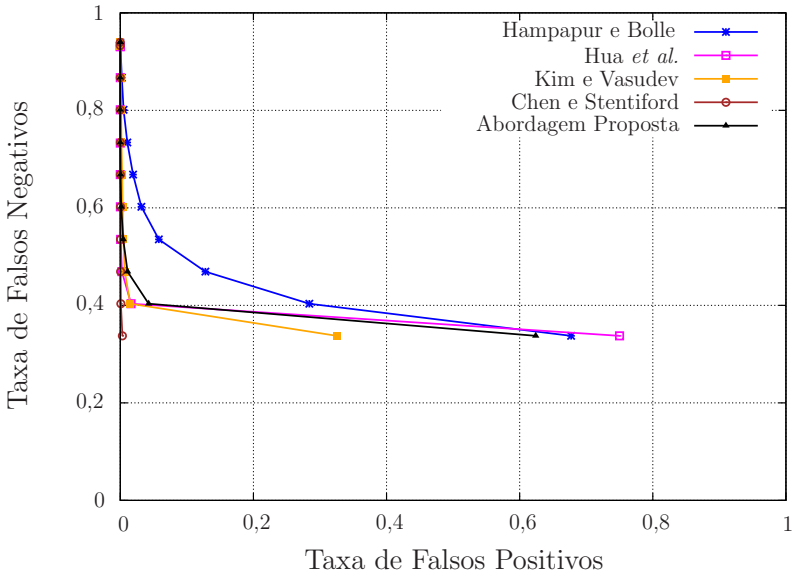
(a) Sequências Curtas



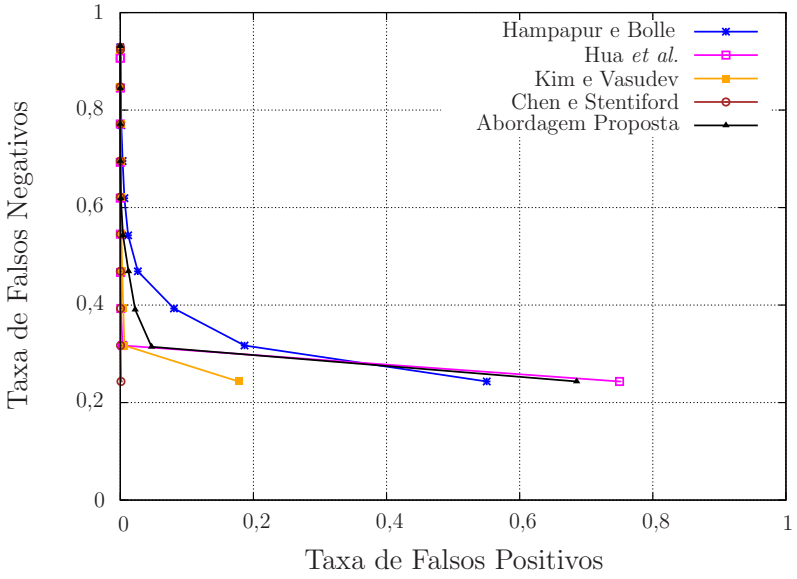
(b) Sequências Longas

Figura 5.3: Curvas Precisão vs. Revocação.

A Figura 5.4 apresenta a curva ROC obtida por cada um desses métodos na variação do limiar de similaridade. A curva ideal deve interceptar a origem (0, 0). Esses gráficos mostram o comportamento de cada abordagem em relação às características de robustez e discriminância.



(a) Sequências Curtas



(b) Sequências Longas

Figura 5.4: As curvas ROC.

Como se pode observar na Figura 5.3(a), a abordagem proposta atinge, em média, uma precisão mais alta que os métodos comparados, na detecção de similaridade entre sequências curtas. É interessante notar que, em níveis intermediários de revocação (entre 50% e 60%), a técnica de Hua *et al.* apresenta um desempenho melhor do que o método proposto. Essa situação se inverte nos extremos (níveis baixos e altos).

Uma das razões para esses resultados é o fato do método proposto ser mais sensível a variação do limiar de similaridade. Tal efeito pode ser observado a partir dos gráficos na Figura 5.4(a). Pode-se notar que a curva obtida pelo método de Hua *et al.* está mais próxima da origem (0, 0) e, portanto, ele é menos sensível que a abordagem proposta em relação a esse limiar.

Em sequências longas, a técnica de Chen e Stentiford se mostrou a mais robusta a variações do limiar de similaridade (Figura 5.4(b)). Por outro lado, a precisão observada em tal método para níveis de revocação menores que 60% é inferior às obtidas nas demais abordagens (Figura 5.4(b)).

Vale a pena notar que a precisão atingida por essa técnica se manteve aproximadamente constante para os diferentes níveis de revocação. Dessa forma, apesar desse método ser capaz de rejeitar uma grande quantidade de falsos positivos (discriminância), ele tolera poucas diferenças entre vídeos similares (robustez).

Isso mostra a relação de compromisso entre robustez e discriminância. Como pode ser observado, a abordagem proposta oferece melhor equilíbrio em termos dessas características que o método de Chen e Stentiford.

A principal vantagem da técnica proposta é a sua eficiência computacional. Uma vez que o tempo necessário para comparar sequências de vídeo é dependente de vários parâmetros (por exemplo, taxa de quadros por segundo, resolução do vídeo, duração total, etc), é muito difícil realizar uma comparação justa do desempenho de diferentes métodos.

A Tabela 5.1 apresenta o custo computacional e os requisitos de espaço (em termos do número de quadros n) de todos os métodos comparados. Dessa forma, pode-se investigar a diferença relativa de desempenho entre diferentes estratégias. Claramente, a abordagem proposta é mais eficiente que as demais técnicas.

Tabela 5.1: O custo computacional e os requisitos de espaço de diferentes abordagens.

Método	*Custo Computacional*		*Requisitos de Espaço*
	Assinatura	*Similaridade*	
Hampapur e Bolle	$O(n)$	$\Omega(n)$	$\Theta(n)$
Hua *et al.*	$O(n)$	$\Omega(n)$	$\Theta(n)$
Kim e Vasudev	$O(n)$	$\Omega(n)$	$\Theta(n)$
Chen e Stentiford	$O(n)$	$\Omega(n)$	$\Theta(n)$
Abordagem Proposta	$\boldsymbol{O(n)}$	$\boldsymbol{O(1)}$	$\boldsymbol{\Theta(1)}$

A fim de avaliar a eficiência do método proposto, foi realizada uma análise de seu desempenho sobre os 1170 vídeos de consulta, pois eles apresentam uma variabilidade bastante abrangente em relação às características supracitadas. Nesse estudo, avaliou-se o tempo por quadro gasto para executar todas as três etapas de seu algoritmo, excluindo-se o tempo de decodificação parcial de cada quadro. Para garantir resultados estatisticamente sólidos, esse experimento foi repetido 10 vezes para cada vídeo.

De acordo com esses experimentos, a execução de todas as etapas da técnica proposta leva um tempo médio igual a, aproximadamente, 700 ± 15 microssegundos por quadro (nível de confiança superior a 99,9%). Para uso *online*, considerando um tempo máximo de espera de 39 segundos [34], esse método pode ser aplicado, em média, a vídeos de até 55.666 quadros (cerca de 37 minutos a uma taxa de 25 quadros por segundo). É importante lembrar que esses valores dependem, naturalmente, do poder computacional da arquitetura física utilizada.

5.5 Conclusões

Neste capítulo, foi apresentada uma nova abordagem para comparação de sequências de vídeo que atua diretamente no domínio comprimido. Essa técnica baseia-se no reconhecimento de padrões de movimento extraídos do fluxo de vídeo e na contagem de sua ocorrência para formar um histograma. Tal combinação faz com que ela seja adequada para enormes coleções de dados de vídeo.

A técnica apresentada foi validada usando um conjunto de dados do TRECVID 2010 (IACC.1), incluindo vários tipos de transformações de vídeo. Resultados de uma comparação do método proposto com soluções recentes em detecção de similaridade entre vídeos mostram que essa técnica apresenta alta eficácia e velocidade computacional na identificação de vídeos similares.

Extensões futuras incluem a avaliação de outras características visuais e medidas de similaridade. Além disso, o método proposto pode ser ampliado para considerar características locais [4–6] e/ou análise de movimento [7–10].

Capítulo 6

Indexação de Dados

Fazer uso eficiente da informação de vídeo requer o desenvolvimento de estruturas diferenciadas para organizar as assinaturas extraídas a partir dela e facilitar a busca por vídeos semelhantes. Por mais de duas décadas, esforços significativos foram gastos tentando melhorar a eficiência no processamento de consultas por similaridade. Apesar disso, muitas questões ainda são consideradas problemas em aberto. Um problema com respeito à criação desses índices é o relacionamento entre a geometria dos dados e a organização da estrutura.

A maioria dos índices existentes empregados para acelerar a recuperação desses dados é construída a partir do particionamento de um conjunto de objetos usando somente a informação de distância entre eles. A fim de manter o balanceamento da estrutura, tal conjunto é fragmentado em partes de mesmo tamanho, ignorando o agrupamento inerente de seus objetos. Em geral, essas técnicas podem ser divididas em duas categorias diferentes. Uma classe de métodos produz partições, portanto, essas partes são disjuntas, o que pode separar objetos próximos, afetando gravemente a eficiência de busca. Outra classe de métodos produz agrupamentos, portanto, essas partes podem se sobrepor, o que pode degradar consideravelmente o tempo das consultas [37, 53, 162, 166].

Neste capítulo, é estudado o desempenho de uma nova estrutura de indexação, chamada de BP-tree (do inglês, *Ball-and-Plane tree*), a qual é construída dividindo-se um conjunto de objetos em grupos compactos. Ela combina vantagens de ambos os paradigmas disjuntos (partições) e não disjuntos (agrupamentos) a fim de obter uma estrutura de aglomerados densos e pouco sobrepostos, melhorando a eficiência no processamento de consultas por similaridade [11, 12, 15].

Essas propriedades da BP-tree são apoiadas por uma extensa avaliação experimental realizada sobre várias bases de dados reais. Os resultados reportados demonstram que essa abordagem supera as soluções tradicionais. Além disso, ela é escalável, exibindo um comportamento sublinear em relação ao número de objetos indexados, que a torna

adequada para indexar grandes coleções.

O restante deste capítulo está organizado como segue. A Seção 6.1 introduz alguns conceitos básicos e descreve trabalhos relacionados. A Seção 6.2 apresenta a abordagem proposta e mostra como aplicá-la para acelerar uma consulta por similaridade. A Seção 6.3 discute o protocolo experimental, reporta resultados de experimentos e compara essa técnica com outros métodos. Por fim, são oferecidas considerações finais e direções para trabalhos futuros na Seção 6.4.

6.1 Revisão da Literatura

Sistemas gerenciadores de banco de dados (SGBDs) tradicionais [58, 136] são capazes de lidar eficientemente com registros estruturados usando o modelo de correspondência exata. Entretanto, tipos de dados complexos, tais como dados multimídia (imagem, áudio e vídeo), dados biológicos (sequências genômicas e proteínas), entre outros, não podem ser representados, de forma eficaz, como registros estruturados [174].

Nesses casos, a busca por similaridade [88] foi estabelecida como o paradigma fundamental. Essencialmente, ela consiste em encontrar, dentro de um conjunto de objetos, aqueles que são os mais semelhantes a um dado objeto de consulta. A semelhança entre pares de objetos quaisquer é calculada por meio de uma função de distância e, portanto, fica subentendido que baixos valores de distância correspondem a um elevado grau de similaridade [174].

Conforme detalhado na Seção 2.10, os tipos mais comuns de consultas por similaridade incluem (1) consultas por abrangência (R-NN), nas quais requisitam-se todos os objetos cuja distância ao objeto de consulta não exceda um dado limite R, e (2) consultas pelos k vizinhos mais próximos (do inglês, *k-nearest neighbors* – k-NN), em que um número especificado k de objetos, que estão mais próximos à consulta, é solicitado [174].

Elaboradas estruturas de dados, conhecidas como métodos de acesso (MAs), têm sido propostas para acelerar consultas por similaridade [12, 14, 47, 65, 139]. Elas podem ser amplamente classificadas, dependendo de seu campo de aplicabilidade, como métodos de acesso espaciais (MAEs) ou métodos de acesso métricos (MAMs), sendo que os primeiros só se aplicam quando o espaço de dados for um espaço vetorial (Seção 2.8).

Algoritmos de busca em espaços métricos podem ser divididos em duas grandes classes: métodos baseados em pivôs e métodos baseados em agrupamento [47]. A primeira, seleciona alguns objetos da coleção como pivôs e, em seguida, calcula e armazena as distâncias entre eles e os demais objetos do banco de dados. Durante uma busca, essas distâncias são usadas para descartar objetos sem compará-los com o objeto de consulta. Um estratégia baseada em agrupamento consiste em dividir o espaço em regiões as mais compactas possíveis, normalmente de uma forma recursiva, e armazenar um objeto repre-

sentativo ("centro") para cada região, e mais algumas informações extras que permitem rapidamente descartá-la no momento da consulta. Em uma busca, regiões completas são descartadas usando as distâncias de seus objetos representativos ao objeto de consulta [47].

Dois critérios podem ser empregados para delimitar uma região em abordagens baseadas em agrupamento. O primeiro elege um subconjunto de objetos como representativos e coloca os demais objetos do conjunto de entrada dentro da região de seu representativo mais próximo, portanto, eles são delimitados por hiperplanos. O segundo critério é o raio de cobertura, que é a distância máxima entre um objeto representativo e qualquer objeto na sua região [47].

Uma ampla revisão de métodos de acesso métricos pode ser encontrada no trabalho de Chávez *et al.* [47]. Algumas das principais características das técnicas já publicadas são brevemente discutidas a seguir.

O trabalho pioneiro de Burkhard e Keller [38] apresenta duas técnicas para fragmentar um conjunto de objetos de forma recursiva. A primeira abordagem particiona o espaço de dados com base na escolha de um objeto representativo e no agrupamento dos demais objetos usando suas distâncias a ele. A segunda abordagem divide o conjunto de entrada em um número fixo de grupos e escolhe um objeto de cada grupo para ser o seu representativo.

Uhlmann [165] introduziu dois métodos hierárquicos, chamados Árvores Métricas. O primeiro, referenciado por GH-tree (do inglês, *Generalized Hyperplane tree*), é uma estrutura em forma de árvore binária que corresponde à segunda técnica de Burkhard e Keller [38]. Essa abordagem divide recursivamente o espaço de dados em uma hierarquia de partições a partir da seleção de dois objetos representativos e da subsequente atribuição dos objetos remanescentes ao objeto representativo mais próximo. O segundo método é referenciado por Decomposição por Bolas (do inglês, *Ball Decomposition*), sendo similar à primeira técnica de Burkhard e Keller [38]. Ele estabelece cortes ao redor de um único objeto representativo, particionando, assim, o espaço de dados.

O VP-tree (do inglês, *Vantage-Point tree*) [170] é uma extensão do método de decomposição por bolas proposto por Uhlmann [165], que reparte recursivamente um conjunto de objetos em uma hierarquia de partições a partir de cortes sobre um objeto representativo diferente a cada nível, o qual também é chamado de "ponto de vantagem" (do inglês, *vantage point*), gerando, assim, uma árvore binária. A fim de reduzir o número de cálculos de distância para responder a consultas por similaridade usando o VP-tree, Baeza-Yates *et al.* [24] sugeriram utilizar um mesmo objeto representativo em todas as partições pertencentes a um mesmo nível. Dessa forma, a árvore binária se degenera em uma simples lista de representativos. Bozkaya e Özsoyoglu [35] propuseram uma generalização do VP-tree chamado de MVP-tree (do inglês, *Multi-Vantage-Point tree*), que escolhe múltiplos objetos representativos para cada nível da estrutura, produzindo uma árvore multivias.

O GNAT (do inglês, *Geometric Near-Neighbor Access Tree*) [37] pode ser visto como uma versão refinada da segunda técnica de Burkhard e Keller [38], que funciona de forma semelhante a GH-tree, com a diferença de escolher mais que dois representativos por nível. Ele particiona o espaço com base nos chamados domínios de Dirichlet, os quais são considerados uma generalização do particionamento em células de Voronoi. Além do objeto representativo e do raio de cobertura, ele armazena distâncias entre pares de objetos representativos. Tais distâncias podem ser exploradas na redução do espaço de busca usando desigualdade triangular.

O SAT (do inglês, *Spatial Approximation Tree*) [122] também se baseia em diagramas de Voronoi, mas em contraste com o GNAT, ele tenta criar uma estrutura parecida com a do grafo de Delaunay. Os autores comprovaram ser impossível construir uma estrutura ideal sobre espaços métricos. Portanto, SAT é uma simplificação que obriga o retrocesso (do inglês, *backtracking*) na árvore de pesquisa.

O LC (do inglês, *List of Clusters*) [46] divide o espaço em grupos de acordo com a proximidade para escolher objetos representativos, armazenando também o raio de cobertura de cada grupo. Os autores mostraram que sua abordagem é bastante adequada para a busca em espaços de alta dimensionalidade, nos quais ela superou, de longe, diversos índices métricos conhecidos.

A M-tree (do inglês, *Metric tree*) [53] é uma árvore balanceada que também se baseia na segunda técnica de Burkhard e Keller [38], na qual os dados são armazenados nas folhas e uma hierarquia de grupos apropriada é construída em cima delas, permitindo operações dinâmicas. Traina Jr. *et al.* [162] propuseram uma extensão do M-tree, denominada Slim-tree. Três novos recursos foram introduzidos: (1) uma estratégia de divisão de nós baseada no algoritmo MST (do inglês, *Minimum Spanning Tree*), (2) uma política de inserção de objetos baseada na taxa de ocupação dos nós e (3) um algoritmo de pós-otimização para reduzir o grau de sobreposição entre os nós da árvore, chamado *Slim-down*. Vieira *et al.* [166] sugeriram flexibilizar a restrição de balanceamento da árvore, originando o DBM-tree (do inglês, *Density-Based Metric tree*). Essa flexibilização é rigidamente controlada a partir do equilíbrio entre o número de nós acessados na execução de buscas em largura e em profundidade. Tal estratégia minimiza a sobreposição entre nós em regiões de alta densidade e, consequentemente, melhora a eficiência de busca nessas regiões.

O uso de múltiplos objetos representativos foi proposto por Santos Filho *et al.* [144]. A abordagem deles elege uma parcela dos objetos do conjunto de entrada como representativos globais, chamados "omni-focos". As distâncias de todos os outros objetos a cada "omni-foco" são usadas para produzir um sistema de coordenadas do espaço de dados. Cada coordenada pode ser indexada usando qualquer método de acesso espacial ou métrico, ou mesmo uma busca sequencial, gerando, assim, toda uma nova família de métodos de acesso, denominada de "família Omni".

O iDistance [89] é um índice eficiente na realização de consultas k-NN em espaços vetoriais de alta dimensionalidade. Ele divide o espaço em grupos e seleciona um objeto representativo de cada um deles. Em seguida, cada objeto é atribuído a uma chave unidimensional de acordo com suas distâncias ao representativo de cada grupo. Essa chave é indexada usando uma árvore B^+ [58, 136], na qual todas as operações subsequentes são realizadas. Os autores mostraram que o iDistance é superior à M-tree e à "família Omni". Todavia, a sua eficiência de busca é significativamente afetada caso a escolha do esquema de agrupamento e/ou dos objetos representativos não for apropriada.

Em geral, todos os trabalhos anteriores seguem dois paradigmas básicos: disjuntos (partições) ou não disjuntos (agrupamentos). O primeiro, divide o espaço de dados em partições, o que pode separar objetos próximos, afetando gravemente a eficiência de busca. O último, divide o conjunto de objetos em grupos, o que pode degradar consideravelmente o tempo das consultas caso eles se sobreponham. A fim de manter o balanceamento da estrutura, eles fragmentam o conjunto de entrada em partes de tamanhos iguais, ignorando o agrupamento natural de seus objetos.

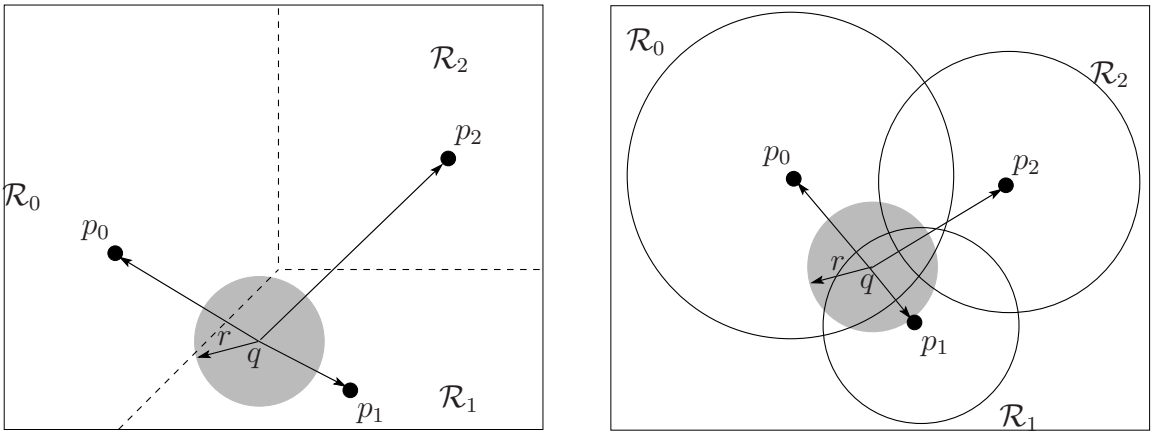
Diferente de todas as técnicas anteriores, o método proposto não divide o conjunto de objetos em partições (disjuntas) ou agrupamentos (não disjuntos). Em vez disso, ela é uma estrutura de indexação que combina vantagens de ambas estratégias.

6.2 Abordagem Proposta

Métodos de acesso métricos tradicionais dividem o espaço em regiões e escolhem objetos representativos para representá-las. Esse processo é normalmente realizado de forma recursiva, permitindo que a estrutura seja organizada hierarquicamente, resultando em uma árvore. Em geral, os nós dessa árvore contêm informações sobre objetos representativos, objetos em sua região de cobertura e distâncias aos seus representativos. Cada nó é geralmente armazenado em disco usando registros de tamanho fixo. Isso é essencial para garantir a persistência de dados e permitir o manuseio de um número qualquer de objetos. Uma vez que os acessos ao disco são lentos, é importante minimizar o número de acessos a disco (custo de E/S) necessários para responder a consultas.

Quando uma consulta é realizada, o objeto de consulta é primeiro comparado com os objetos representativos do nó raiz. A desigualdade triangular é então utilizada para podar subárvores, evitando cálculos de distância entre o objeto de consulta e objetos nas subárvores descartadas. Cálculos de distância entre tipos complexos de dados podem ser computacionalmente caros. Portanto, para obter um bom desempenho em métodos de acesso métricos, também é vital minimizar o número de cálculos de distância (custo de CPU) em operações de consulta.

Se os objetos do conjunto de entrada estão distribuídos de maneira uniforme, existe uma alta probabilidade da região de consulta intersectar a região de cobertura de vários nós, afetando gravemente a eficiência de busca. Essa situação é ilustrada na Figura 6.1, em que é requisitada uma consulta por abrangência (região cinza) usando um objeto de consulta q e um raio de busca r. As linhas tracejadas (a) e as circunferências (b) delimitam três regiões $\mathcal{R}_0$, $\mathcal{R}_1$ e $\mathcal{R}_2$. Os objetos representativos de cada região são denotados pelos pontos p_0, p_1 e p_2, respectivamente.



(a) Partições (disjuntas). (b) Agrupamentos (não disjuntos).

Figura 6.1: Efeitos de diferentes paradigmas de divisão do espaço em um conjunto de objetos com uma distribuição uniforme.

A Figura 6.1(a) ilustra os efeitos do uso de partições (disjuntas). Nesse caso, a região de consulta cruza ambas as regiões $\mathcal{R}_0$ e $\mathcal{R}_1$, portanto, esses dois nós devem ser acessados a fim de responder à consulta. Como se pode observar, o nó que cobre a região $\mathcal{R}_0$ deve ser analisado mesmo que nenhum objeto qualificado possa ser encontrado. Contudo, se o raio de busca r for pequeno o suficiente, toda a consulta pode ser respondida por um único nó.

Os efeitos do uso de agrupamentos (não disjuntos) são ilustrados na Figura 6.1(b). Nessa figura, a região de resposta à consulta cobre as regiões $\mathcal{R}_0$, $\mathcal{R}_1$ e $\mathcal{R}_2$, portanto, esses três nós devem ser acessados. Além disso, vale ressaltar que o algoritmo deve analisar vários nós independente do quão pequeno for o raio de busca r.

Quando os objetos do conjunto de entrada estão distribuídos de maneira esparsa, ambos os paradigmas disjuntos e não disjuntos podem gerar grupos compactos. Entretanto, a fragmentação em partes de tamanhos iguais a fim de manter o balanceamento da estrutura pode dividir grupos inerentes de objetos, como ilustrado na Figura 6.2.

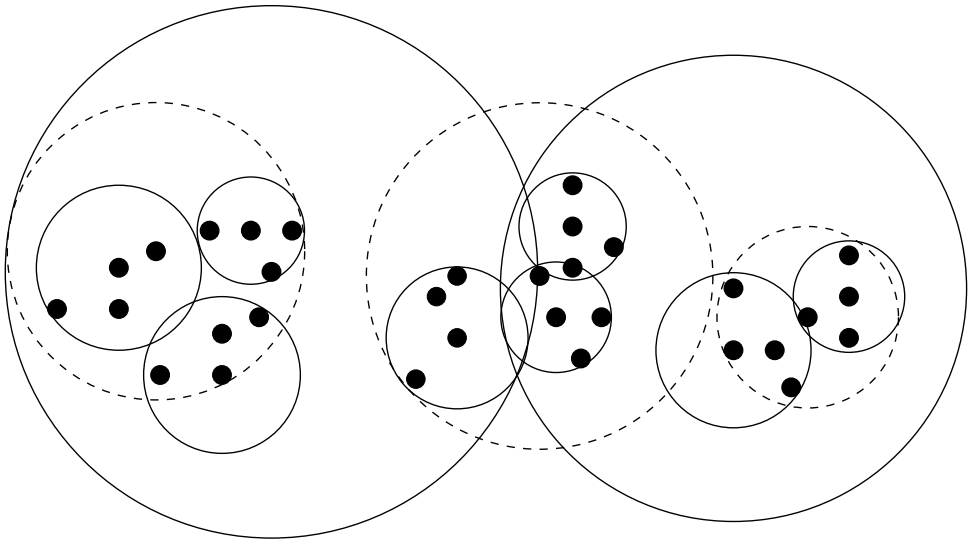


Figura 6.2: Efeitos do uso de partes de tamanho fixo na fragmentação de um conjunto de objetos com uma distribuição esparsa.

Nesse exemplo, há três grupos naturais no espaço de dados, delimitados por circunferências em linha tracejada. Devido às restrições de balanceamento, cada grupo pode ser dividido em grupos menores, demarcados por circunferências em linha sólida. Quando isso acontece, os problemas discutidos acima podem emergir e, consequentemente, afetar a eficiência de busca.

A novidade da BP-tree é combinar as vantagens de ambas as abordagens disjuntas e não disjuntas. Nesse método, o espaço de dados é dividido em grupos compactos, respeitando a sua distribuição. Essa estratégia supera as deficiências supracitadas.

6.2.1 Visão Geral

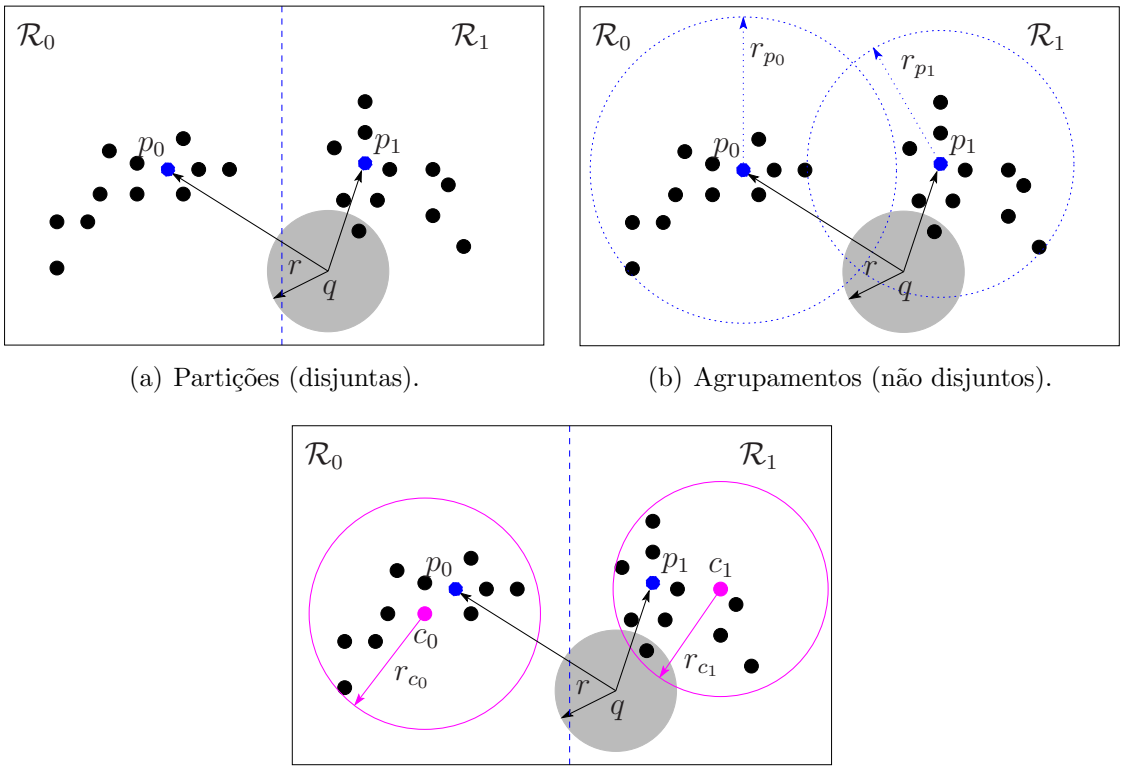
Para entender como a abordagem proposta trabalha, é preciso considerar um conjunto inicial de objetos. Inicialmente, ele é dividido em grupos de acordo com a sua distribuição global. Pode-se refinar ainda mais esse particionamento ao dividir cada grupo existente com base na distribuição local de um subconjunto de objetos. Esse processo pode ser repetido tomando-se um subconjunto menor a cada iteração até obter grupos tão compactos que não faça mais sentido dividi-los novamente. Por fim, tem-se um conjunto hierárquico de grupos. Essa é a ideia geral da BP-tree, que adapta tal modelo ao disco. Em outras palavras, dado um objeto de consulta, pode-se reduzir gradualmente o espaço de busca através da pesquisa em um subconjunto de objetos com uma distribuição mais relevante.

Em cada nível, o conjunto de objetos é dividido em subconjuntos por meio do agrupamento ao redor de objetos representativos mais próximos. Cada subconjunto é uma

partição no sentido de que seus objetos têm distâncias similares ao seu objeto representativo. Assim, o espaço de dados é particionado em subespaços, definindo hiperplanos entre objetos representativos. Esse processo capta a distribuição de dados no sentido de que objetos distantes são inseridos em subconjuntos diferentes. Em seguida, para cada subconjunto, estabelece-se uma região limítrofe a partir da escolha de um objeto de referência que defina o menor raio de cobertura. O raio de cobertura é a distância máxima entre o objeto de referência e qualquer objeto em seu subconjunto. Assim, o espaço de dados é dividido em agrupamentos, definindo hiperesferas em torno de objetos de referência. A introdução dessas regiões delimitadoras aumenta a taxa de poda do espaço de busca. Depois disso, aplica-se esse processo recursivamente até que cada subconjunto de objetos não possa mais ser dividido. No final, tem-se uma coleção de agrupamentos, que formam um particionamento hierárquico de objetos.

A fim de realçar a novidade da BP-tree, são apresentados a seguir mais detalhes sobre como esse método se beneficia e combina as vantagens de ambos os paradigmas disjuntos e não disjuntos. Por um lado, BP-tree divide o conjunto de objetos em subconjuntos com base em distâncias a objetos representativos. Existe uma ordem total sobre distâncias a um mesmo objeto representativo e, portanto, esses subconjuntos são disjuntos (partições). Por outro lado, BP-tree divide o conjunto de objetos em subconjuntos com base em distâncias a seus objetos de referência. Um mesmo objeto pode estar dentro da intersecção entre regiões definidas por objetos de referência de diferentes subconjuntos e, portanto, tais subconjuntos podem se sobrepor. Essa estratégia produz uma estrutura de grupos compactos e pouco sobrepostos, melhorando a eficiência no processamento de consultas por similaridade, especialmente em espaços de alta dimensionalidade.

A Figura 6.3 ilustra como a técnica proposta e os dois tipos diferentes de paradigmas de divisão do espaço de dados lidam com um conjunto de objetos e com uma consulta. As abordagens disjuntas particionam o espaço de dados por meio de um corte (linha tracejada) entre objetos representativos (Figura 6.3(a)). Nesse exemplo, a região de consulta cruza as partições $\mathcal{R}_0$ e $\mathcal{R}_1$, logo, dois nós devem ser acessados a fim de responder à consulta. As abordagens não disjuntas agrupam objetos em torno de representativos e usam o raio de cobertura (circunferência em linha pontilhada) para definir regiões que delimitam cada grupo (Figura 6.3(b)). Normalmente, essas regiões se sobrepõem. Nessa figura, a região de resposta à consulta cobre duas regiões e, assim, ambos os nós $\mathcal{R}_0$ e $\mathcal{R}_1$ têm que ser acessados. BP-tree particiona o espaço de dados a partir da definição de um corte (linha tracejada) entre objetos representativos e, para cada partição, um objeto de referência com o menor raio de cobertura é utilizado para delimitar uma região mínima o possível (circunferência em linha sólida), na qual se pode encontrar objetos (Figura 6.3(c)). Nesse caso, a desigualdade triangular é usada para podar uma subárvore ($\mathcal{R}_0$) e, portanto, apenas um nó precisa ser acessado ($\mathcal{R}_1$).



(a) Partições (disjuntas).

(b) Agrupamentos (não disjuntos).

(c) BP-tree.

Figura 6.3: Exemplo de como a BP-tree se beneficia e combina as vantagens de ambas as abordagens disjuntas e não disjuntas.

Por fim, é notável que a BP-tree pode ser desbalanceada. Contudo, na busca por similaridade em espaços de alta de dimensionalidade, árvores desbalanceadas podem proporcionar um desempenho melhor do que árvores balanceadas, como foi demonstrado por Chávez e Navarro [46]. Em tal estudo, os autores mostraram que, em espaços de alta dimensionalidade, o custo de busca é determinado pela taxa de poda do espaço de busca, e não pela altura da árvore. Essa taxa está diretamente relacionada à forma como o espaço de dados é organizado. Métodos que produzem árvores balanceadas fragmentam o conjunto de objetos em partes de tamanho fixo, ignorando a sua distribuição intrínseca. A BP-tree divide o conjunto de objetos de acordo com a sua distribuição natural e, assim, ela pode organizar melhor o espaço de dados. Uma discussão mais detalhada sobre os benefícios de empregar árvores desbalanceadas em buscas por similaridade é apresentada no trabalho de Chávez e Navarro [46].

6.2.2 Criação do Índice

Em geral, a BP-tree é uma árvore desbalanceada obtida a partir do particionamento hierárquico de um conjunto de objetos. Tal como em outras árvores métricas, os objetos do conjunto de entrada são armazenados em páginas de disco de tamanho fixo. A Tabela 6.1 sumariza os símbolos utilizados no restante deste capítulo.

Tabela 6.1: Sumário de símbolos e definições.

Símbolo	**Definição**
$d(x, y)$	a distância entre dois objetos quaisquer x e y
C	a capacidade de uma página de disco
D	o tamanho de uma página de disco
E	a dimensionalidade embutida do espaço de dados
K	o número de partições geradas a partir de um conjunto
M	a taxa de ocupação mínima de um nó
N	o número de objetos em um conjunto
O	um conjunto de objetos
P	um conjunto de partições
T	o tipo (adimensional/vetorial) do espaço métrico

BP-tree tem dois tipos de nós: nós folhas e nós índices. Cada nó índice corresponde a uma única página de disco e contém informações sobre um particionamento. Em contraste, cada nó folha consiste em uma lista de páginas de disco e, portanto, pode ter uma capacidade ilimitada. Essa estratégia autoriza a criação de "folhas grandes" a fim de evitar a quebra de grupos naturais de objetos. Dessa forma, mantém-se um equilíbrio entre acessos sequenciais e aleatórios ao disco, permitindo a aceleração do processo de busca.

Os objetos são armazenados tanto em nós índices quanto folhas. A estrutura de um nó folha é

$$leafnode\ [\ \text{vetor de}\ \langle d(o_i, o_{rep}), o_i \rangle\ ],$$

em que o_i é um objeto e $d(o_i, o_{rep})$ é a sua distância ao objeto representativo o_{rep} desse nó. A estrutura de um nó índice é

$$indexnode\ [\ \text{vetor de}\ \langle o_i, r(o_i), d(o_i, o_{rep}), r(o_{ref}), d(o_i, o_{ref}), ptr(T(o_i)) \rangle\ ],$$

em que o_i mantém o objeto representativo da subárvore $T(o_i)$ apontada por $ptr(T(o_i))$ e $r(o_i)$ é o raio de cobertura que delimita a região definida por o_i. A distância entre o_i e o objeto representativo o_{rep} desse nó é mantida em $d(o_i, o_{rep})$. A distância entre o_i e o objeto de referência o_{ref} que estabelece a região mínima definida pelo raio de cobertura $r(o_{ref})$ é mantida em $d(o_i, o_{ref})$. O ponteiro $ptr(T(o_i))$ aponta para o nó raiz da subárvore $T(o_i)$ enraizada por o_i.

A construção da árvore é feita de cima para baixo. Esse processo é ilustrado na Figura 6.4. No início, o conjunto de objetos $O = \{o_1, o_2, \ldots, o_N\}$ é considerado como parte de uma única partição. Uma vez que o tamanho da página de disco D é fixo, ela suporta um número predefinido máximo de objetos C, tal que

$$C = \frac{D}{\arg\max\limits_{o_i \in O} sizeof(o_i)}. \tag{6.1}$$

Assim, esses objetos são primeiramente divididos em $K \leq C$ subpartições $P = \{p_1, p_2, \ldots, p_K\}$, com pelo menos $\lceil M \times C \rceil$ objetos em cada uma. Informações sobre todas essas subpartições formam o nó índice do primeiro nível da árvore. Para cada partição $p_i \in P$, um subconjunto O_{p_i} é criado tomando-se os objetos de p_i. Para construir níveis subsequentes da árvore, esse processo é repetido para todos os novos subconjuntos de objetos em cada nível, criando uma hierarquia de nós índices. Tal processo termina quando o número de objetos em um subconjunto for menor ou igual à capacidade da página ou o número de partições geradas a partir de um subconjunto for menor que 2. Então, os objetos no subconjunto são gravados em um nó folha no disco.

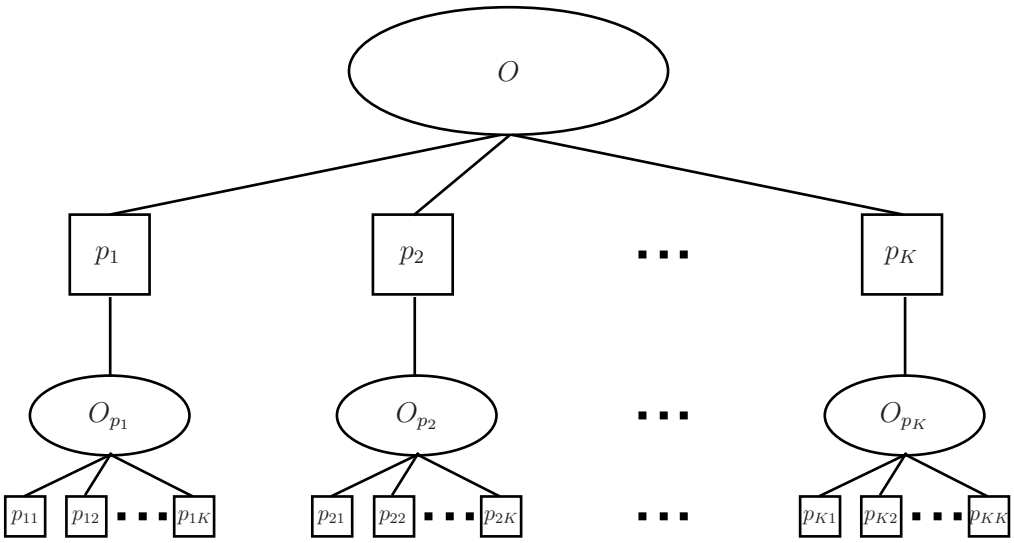


Figura 6.4: Uma representação da BP-tree.

O Algoritmo 1 formaliza o procedimento acima. Ele começa verificando a cardinalidade do conjunto de objetos O (linha 3). Se o conjunto inteiro puder ser inserido em uma única página de disco, a função Create-leafnode é usada para criar um nó folha (linha 4). Caso contrário, chama-se a função Split para dividir esse conjunto em $K \leq C$ partições com uma ocupação mínima igual a $\lceil M \times C \rceil$ (linha 6). A função Split pode usar qualquer método de agrupamento particional, como k-medoids [33]. O algoritmo particional é responsável por encontrar os objetos representativos e os objetos de referência

de cada nível. A taxa de ocupação mínima M é ajustada quando a estrutura é criada, e esse valor deve pertencer ao intervalo $[0, 1]$. Em seguida, verifica-se se tal conjunto deve ser particionado (linha 8). Em caso afirmativo, chama-se a função INITIALIZE-INDEXNODE, que cria um nó índice que armazena informações sobre esse particionamento (linha 11). Então, para cada partição, a função CREATE-SUBSET é usada para criar um subconjunto de objetos (linha 13). Essa função é responsável por selecionar os objetos de uma determinada partição. Depois disso, repete-se esse processo para todos os novos subconjunto de objetos (linha 15). Por fim, a função UPDATE-INDEXNODE é usada para atualizar as informações em cada entrada do nó índice (linha 16).

Algoritmo 1 A construção da BP-tree.

```
1: função BuildTree(d, D, M, N, O)
2:     C ← D / (arg max_{o_i ∈ O} sizeof(o_i)).
3:     se N ≤ C então
4:         return Create-leafnode(N, O);
5:     senão
6:         P ← Split(d, C, M, N, O);
7:         K ← cardinality(P);
8:         se K < 2 então
9:             return Create-leafnode(N, O);
10:        senão
11:            Parent ← Initialize-indexnode(K, P);
12:            para todo p ∈ P faça
13:                O_p ← Create-subset(p, K, P, N, O);
14:                N_p ← cardinality(O_p);
15:                Child ← BuildTree(d, D, M, N_p, O_p);
16:                Update-indexnode(p, Parent, Child);
17:            fim para
18:            return Parent
19:        fim se
20:    fim se
21: fim função
```

6.2.3 Consultas por Similaridade

BP-tree pode responder aos dois tipos mais comuns de consultas por similaridade: consulta por abrangência $R(o_q, r(o_q))$, que recupera todos os objetos encontrados a uma distância requisitada $r(o_q)$ de um dado objeto de consulta o_q; e consulta por vizinhos mais próximos $kNN(o_q)$, que recupera os k vizinhos mais próximos de um dado objeto de consulta o_q. A seguir, cada um desses procedimentos é explicado em mais detalhes.

6.2.4 Consultas por Abrangência

O algoritmo de busca por abrangência inicia-se no nó raiz e percorre a árvore usando uma busca em profundidade, visitando todas as subárvores não descartáveis. Durante a busca, todas as distâncias armazenadas são usadas para podar subárvores.

Esse procedimento é descrito no Algoritmo 2. Se o nó atual é um nó índice, todas as entradas não vazias são processadas como se segue. Primeiro, se $|d(o_q, o_{rep}) - d(o_i, o_{rep})| - r(o_i) > r(o_q)$, a subárvore apontada por $ptr(T(o_i))$ é descartada (linha 6). Caso contrário, calcula-se a distância $d(o_q, o_i)$ e, se $d(o_q, o_i) \leq r(o_q)$, o objeto o_i é retornado no resultado da

Algoritmo 2 Consulta por abrangência.

```
 1: procedimento RangeSearch(o_q, r(o_q), d(o_q, o_rep), ptr(T(o_rep)))
 2:     se T(o_rep) não é uma folha então
 3:         L ← ∅                                   ▷ Lista de todas as subárvores não descartadas.
 4:         d_min ← ∞
 5:         para todo ⟨o_i, r(o_i), d(o_i, o_rep), r(o_ref), d(o_i, o_ref), ptr(T(o_i))⟩ ∈ T(o_rep) faça
 6:             se |d(o_q, o_rep) − d(o_i, o_rep)| − r(o_i) ≤ r(o_q) então
 7:                 Calcule d(o_q, o_i)
 8:                 se d(o_q, o_i) ≤ r(o_q) então
 9:                     Adicione ⟨d(o_q, o_i), o_i⟩ no conjunto de resposta
10:                 fim se
11:                 se d(o_q, o_i) − r(o_i) ≤ r(o_q) então
12:                     se |d(o_q, o_i) − d(o_i, o_ref)| − r(o_ref) ≤ r(o_q) então
13:                         L ← L ∪ {⟨d(o_q, o_i), ptr(T(o_i))⟩}
14:                         se d_min > d(o_q, o_i) então
15:                             d_min ← d(o_q, o_i)
16:                         fim se
17:                     fim se
18:                 fim se
19:             fim se
20:         fim para
21:         para todo ⟨d(o_q, o_i), ptr(T(o_i))⟩ ∈ L faça
22:             se (d(o_q, o_i) − d_min)/2 ≤ r(o_q) então
23:                 RangeSearch(o_q, r(o_q), d(o_q, o_i), ptr(T(o_i)))
24:             fim se
25:         fim para
26:     senão
27:         para todo ⟨d(o_i, o_rep), o_i⟩ ∈ T(o_rep) faça
28:             se |d(o_q, o_rep) − d(o_i, o_rep)| ≤ r(o_q) então
29:                 Calcule d(o_q, o_i)
30:                 se d(o_q, o_i) ≤ r(o_q) então
31:                     Adicione ⟨d(o_q, o_i), o_i⟩ no conjunto de resposta
32:                 fim se
33:             fim se
34:         fim para
35:     fim se
36: fim procedimento
```

consulta (linha 7 a 10). Então, descartam-se todas as subárvores que satisfazem o critério $d(o_q, o_i) - r(o_i) > r(o_q)$ (linha 11). Em seguida, usa-se a desigualdade triangular e evita-se o acesso a subárvores, verificando-se se $|d(o_q, o_i) - d(o_i, o_{ref})| - r(o_{ref}) > r(o_q)$ (linha 12). Esse critério de poda é baseado no fato de que a expressão $|d(o_q, o_i) - d(o_i, o_{ref})| - r(o_{ref})$ forma um limite inferior para as distâncias entre o objeto de consulta o_q e qualquer objeto da subárvore apontada por $ptr(T(o_i))$ (Figura 6.5). Todas as subárvores não descartadas $ptr(T(o_i))$ e suas distâncias $d(o_q, o_i)$ são armazenadas em uma lista (linha 13). Depois disso, procura-se pela entrada cujo objeto representativo o_i é o mais próximo do objeto de consulta o_q (ou seja, $d_{min} \leftarrow \arg\min\ d(o_q, o_i)$) e, então, retiram-se da lista todas as entradas tais que $(d(o_q, o_i) - d_{min})/2 > r(o_q)$ (linha 22). Por fim, o algoritmo prossegue pesquisando recursivamente nas subárvores restantes nessa lista (linha 23).

Nós folhas são processados de maneira similar. Cada entrada é analisada usando o critério de poda $|d(o_q, o_{rep}) - d(o_i, o_{rep})| > r(o_q)$ (linha 28). Se ele for satisfeito, uma entrada pode ser ignorada com segurança. Caso contrário, a distância $d(o_q, o_i)$ é avaliada e o objeto o_i é reportado caso $d(o_q, o_i) \leq r(o_q)$ (linha 29 a 32).

A principal vantagem da BP-tree é aplicar limites inferiores projetados tanto para abordagens disjuntas ($(d(o_q, o_i) - d_{min})/2$) quanto não disjuntas ($|d(o_q, o_{rep}) - d(o_i, o_{rep})| - r(o_i)$ e $d(o_q, o_i) - r(o_i)$). É importante notar que em todas as quatro etapas (linhas 6, 11, 22 e 28) nas quais aplicam-se tais critérios de poda, descartam-se algumas entradas sem calcular distâncias a seus objetos correspondentes. Dessa forma, o processo de busca é realizado de maneira mais rápida e eficiente. A prova formal desses critérios de poda é apresentada no trabalho de Zezula *et al.* [174].

Uma característica diferenciada em relação a esse processo é impor um limite inferior para a distância entre o objeto de consulta o_q e qualquer objeto o_j pertencente a subárvore apontada por $ptr(T(o_i))$ usando o objeto de referência o_{ref}. Essa situação é demonstrada na Figura 6.5. Por desigualdade triangular, tem-se que $d(o_j, o_q) \geq |d(o_q, o_{ref}) - d(o_j, o_{ref})|$ e $d(o_q, o_{ref}) \geq |d(o_q, o_i) - d(o_i, o_{ref})|$. Somando-se essas desigualdades e tendo em mente que $d(o_j, o_{ref}) \leq r(o_{ref})$ (definição de cobertura de raio), obtém-se que $d(o_j, o_q) \geq |d(o_q, o_i) - d(o_i, o_{ref})| - r(o_{ref})$. Portanto, o objeto de consulta o_q é comparado apenas com objetos para os quais não se pode provar que eles estejam longe o suficiente de o_q.

6.2.5 Consultas por Vizinhos mais Próximos

O algoritmo para consultas por k vizinhos mais próximos consiste na simulação de uma busca por abrangência dinâmica com um raio de consulta variável r_k igual à distância entre o objeto de consulta o_q e o k-ésimo vizinho mais próximo. No início, r_k é um valor infinito e, durante o processo de busca, ele é atualizado (reduzido) sempre que um novo vizinho mais próximo for encontrado (∞ caso ainda tiver menos que k candidatos).

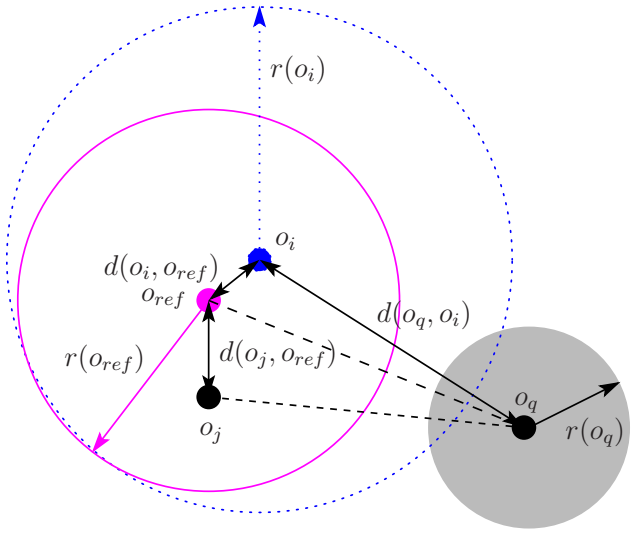


Figura 6.5: Limite inferior da distância $d(o_j, o_q)$ entre o objeto de consulta o_q e qualquer objeto o_j da subárvore $T(o_i)$ enraizada pelo objeto o_i usando o objeto de referência o_{ref}.

Em uma busca por abrangência comum com um raio de consulta fixo r_k, a ordem em que os nós da árvore são visitados não é importante. Esse não é o caso agora, uma vez que é desejável encontrar rapidamente objetos próximos ao objeto de consulta o_q a fim de reduzir o raio de consulta r_k o mais breve possível. A ideia geral proposta por Uhlmann [165] pode ser adaptada para o caso em questão. Porém, as habilidades de poda e a eficiência global dessa abordagem são consideravelmente melhoradas pela BP-tree. Esse procedimento é descrito no Algoritmo 3.

Considerar uma fila de prioridade de nós, sendo o mais promissor o primeiro. Inicialmente, o nó raiz é inserido na fila de prioridade (linha 3). Iterativamente, o nó mais promissor é extraído (linha 5), processado (linhas 8 a 27) e todas as subárvores não descartadas são inseridas na fila (linhas 28 a 33). Esse processo é repetido até que a fila de prioridade torne-se vazia.

A distância $d(o_q, o_i)$ é usada para mensurar o quão promissora é uma subárvore $T(o_i)$ e, portanto, subárvores inseridas na fila de prioridade são visitadas em ordem crescente de proximidade em relação ao objeto de consulta o_q. Como o raio de consulta r_k é reduzido durante o processo de busca, uma subárvore pode parecer útil no momento em que ela for inserida na fila e se tornar inútil mais tarde, quando ela for extraída para ser processada. Assim, os limites inferiores $d(o_q, o_i) - r(o_i)$, $|d(o_q, o_i) - d(o_i, o_{ref})| - r(o_{ref})$ e $(d(o_q, o_i) - d_{min})/2$ são armazenados juntos a subárvore $T(o_i)$ enraizada pelo objeto o_i e a distância $d(o_q, o_i)$. À medida que extrai-se uma entrada da fila de prioridade, verifica-se se esses limites inferiores ultrapassam o raio de consulta r_k, caso em que o nó extraído é descartado, uma vez que ele não pode conter objetos qualificados.

Algoritmo 3 Consulta por vizinhos mais próximos.

procedimento NeighborSearch($o_q, k, ptr(T(o_{rep}))$)
  $r_k \leftarrow \infty$
  $Q \leftarrow \{\langle 0, ptr(T(o_{rep})), 0, 0, 0\rangle\}$ ▷ Fila de prioridade de nós.
  enquanto Q não é vazia **faça**
    Retire $\langle d(o_q, o_{rep}), ptr(T(o_{rep})), l_1, l_2, l_3\rangle$ do topo da fila Q
    se os limites inferiores l_1, l_2 e l_3 não ultrapassarem r_k **então**
      se $T(o_{rep})$ não é uma folha **então**
        $L \leftarrow \emptyset$ ▷ Lista de todas as subárvores não descartadas.
        $d_{min} \leftarrow \infty$
        para todo $\langle o_i, r(o_i), d(o_i, o_{rep}), r(o_{ref}), d(o_i, o_{ref}), ptr(T(o_i))\rangle \in T(o_{rep})$ **faça**
          se $|d(o_q, o_{rep}) - d(o_i, o_{rep})| - r(o_i) \leq r_k$ **então**
            Calcule $d(o_q, o_i)$
            se $d(o_q, o_i) \leq r_k$ **então**
              Atualize o conjunto de resposta e o raio de consulta r_k por meio da inserção de $\langle d(o_q, o_i), o_i\rangle$ e da remoção do objeto mais distante do objeto de consulta o_q
            fim se
            $l_1 \leftarrow d(o_q, o_i) - r(o_i)$
            se $l_1 \leq r_k$ **então**
              $l_2 \leftarrow |d(o_q, o_i) - d(o_i, o_{ref})| - r(o_{ref})$
              se $l_2 \leq r_k$ **então**
                $L \leftarrow L \cup \{\langle d(o_q, o_i), ptr(T(o_i)), l_1, l_2\rangle\}$
                se $d_{min} > d(o_q, o_i)$ **então**
                  $d_{min} \leftarrow d(o_q, o_i)$
                fim se
              fim se
            fim se
          fim se
        fim para
        para todo $\langle d(o_q, o_i), ptr(T(o_i)), l_1, l_2\rangle \in L$ **faça**
          $l_3 \leftarrow (d(o_q, o_i) - d_{min})/2$
          se $l_3 \leq r_k$ **então**
            Insira $\langle d(o_q, o_i), ptr(T(o_i)), l_1, l_2, l_3\rangle$ na fila Q
          fim se
        fim para
      senão
        para todo $\langle d(o_i, o_{rep}), o_i\rangle \in T(o_{rep})$ **faça**
          se $|d(o_q, o_{rep}) - d(o_i, o_{rep})| \leq r_k$ **então**
            Calcule $d(o_q, o_i)$
            se $d(o_q, o_i) \leq r_k$ **então**
              Atualize o conjunto de resposta e o raio de consulta r_k por meio da inserção de $\langle d(o_q, o_i), o_i\rangle$ e da remoção do objeto mais distante do objeto de consulta o_q
            fim se
          fim se
        fim para
      fim se
    fim se
  fim enquanto
fim procedimento

6.2.6 Atualização do Índice

A BP-tree é uma estrutura cujo algoritmo de construção precisa conhecer todos os objetos com antecedência. A seguir, é apresentada uma estratégia que permite atualizar o índice após a sua criação inicial. Em geral, é difícil adicionar novos objetos sem degradar a árvore, uma vez que eles podem impor uma nova geometria para o espaço de dados.

Considerar o processo de inserir um novo objeto na BP-tree. A inserção é feita percorrendo-se a hierarquia de nós índices usando uma busca em profundidade. O algoritmo segue um caminho para baixo na árvore a fim de localizar o nó folha mais adequado, descendo em cada etapa para a subárvore que minimiza a distância do seu objeto representativo ao novo objeto. Nesse ponto, insere-se o novo objeto.

É importante lembrar que nós folhas podem ter uma capacidade ilimitada. Essa liberdade abre uma série de possibilidades que merecem um estudo mais aprofundado, mas uma consequência imediata é que sempre podem-se inserir novos objetos nas folhas da árvore. Dessa forma, a árvore é imutável em sua parte superior (com exceção dos raios de cobertura que precisam ser atualizados) e alterável somente na parte inferior.

Idealmente, é necessário permitir a reconstrução de pequenas subárvores a fim de evitar que elas se transformem em uma espécie de lista ligada. Assim, tal objeto é inserido somente se o tamanho do nó folha for pequeno o suficiente. Caso contrário, uma reconstrução da subárvore pode ser benéfica ao desempenho. Isso leva a uma relação de compromisso entre o custo de inserção e a qualidade da árvore.

A remoção de objetos pode ser feita trivialmente, exceto quando se tratar de objetos representativos. Nesses casos, eles são apenas marcados como objetos removidos, sem liberar o espaço ocupado.

6.3 Avaliação Experimental

Nesta seção, a abordagem proposta é comparada com trabalhos anteriores em termos de eficiência e escalabilidade no processamento de consultas por similaridade. BP-tree foi implementada em C++ usando Linux. Nessa implementação, os algoritmos de particionamento de dados na função Split foram: **k-medoids**, que usa o algoritmo bastante conhecido, denominado PAM (do inglês, *Partitioning Around Medoids*) [33]; e **random**, que seleciona objetos representativos aleatoriamente (ou seja, a etapa de inicialização do algoritmo PAM, que é a realização mais comum do agrupamento em k-medoids). O primeiro tem um desempenho melhor do que o último, no entanto, ambos têm um comportamento similar. No restante deste capítulo, ao se referir à BP-tree, fica subentendido o processo aleatório, salvo indicação em contrário. A estratégia aleatória constitui uma linha de base sobre a eficiência do método proposto.

BP-tree foi comparada com MVP-tree [35], SAT [122], LC [46], M-tree [53], Slim-tree [162], DBM-tree [166] e iDistance [89], que são as abordagens mais populares na busca por similaridade em espaços métricos. Na avaliação experimental, adotou-se a implementação dos métodos MVP-tree, SAT e LC da biblioteca *Metric Space*[1] e a implementação dos métodos M-tree, Slim-tree e DBM-tree da biblioteca *GDBI Arboretum*[2]. A implementação do iDistance foi obtida diretamente da página pessoal do autor[3]. A fim de garantir uma comparação justa, todos esses métodos foram ajustados usando sua melhor configuração recomendada.

Os experimentos foram realizados em uma máquina equipada com um processador Intel Xeon E5620 (8 núcleos de 2,4 GHz) e 16 GBytes de memória DDR2, executando o sistema operacional Gentoo (núcleo linux 2.6.32) e o sistema de arquivos ext3.

Esses experimentos têm o intuito de responder às seguintes questões:

- Como é a eficiência da BP-tree comparada às soluções vigentes?
- Quão escalável é a BP-tree em relação a trabalhos anteriores?
- Como o custo de construção da BP-tree se compara ao dos índices existentes?
- Quão viável é a construção incremental da BP-tree?

6.3.1 Eficiência

Esta seção reporta resultados experimentais sobre a eficiência da técnica apresentada, usando uma grande variedade de bases de dados reais com propriedades diferentes, as quais afetam diretamente o comportamento de uma estrutura de indexação, como a dimensionalidade intrínseca do espaço de dados, o tamanho do conjunto de objetos, a distribuição das distâncias no espaço métrico, entre outros.

BP-tree e trabalhos anteriores foram avaliados em seis bases de dados descritas na literatura e amplamente utilizadas pela comunidade científica, listadas a seguir:

[1] *The Metric Space Library.* Disponível em: `http://www.sisap.org/Metric_Space_Library.html` (último acesso em 14 de outubro de 2011)

[2] *The GBDI Arboretum Library.* Disponível em: `http://www.gbdi.icmc.usp.br/arboretum/` (último acesso em 14 de outubro de 2011)

[3] `http://www.cs.mu.oz.au/~rui/code.htm` (último acesso em 14 de outubro de 2011)

- ALOI[4] — Trata-se de uma coleção de 72.000 imagens, introduzida por Geusebroek *et al.* [66]. Nos experimentos, cada imagem foi caracterizada por um vetor de características de 256 dimensões, que representa um Correlograma de Cor [85]. Cada correlograma de cor é uma tabela indexada por pares de cor, em que a k-ésima entrada para um par $< i, j >$ especifica a probabilidade de encontrar um ponto de cor j a uma distância k de um ponto de cor i na imagem. A distância de Manhattan (L_1) foi usada para comparar esses vetores de características.

- Caltech-256[5] — É um conjunto de 28.123 imagens, apresentado por Griffin *et al.* [72]. Cada imagem foi convertida em um conjunto de vetores de 6 dimensões por meio do método CBC (do inglês, *Color-Based Clustering*) [153]. Esse método decompõe uma imagem em um conjunto de regiões conexas disjuntas, respeitando um tamanho mínimo e uma dissimilaridade máxima definidas a priori. Cada região obtida é caracterizada por um vetor de características (L, a, b, h, v, s), onde L, a e b representam a sua cor média em um espaço CIE Lab (Seção 2.2), h e v são as coordenadas normalizadas horizontal e vertical do seu centro geométrico e s é o seu tamanho em relação a imagem original. A função de distância usada para comparar esses conjuntos é a EMD (Seção 2.9).

- Corel[6] — Ortega *et al.* [123] criaram um conjunto de 68.040 imagens extraídas do *Corel Gallery*. Nos experimentos, cada imagem corresponde a um vetor de características de 32 dimensões, que representa um Histograma de Cor [158]. Esses histogramas foram obtidos da seguinte maneira: o espaço de cor HSV foi dividido em 32 subespaços (cores), usando 8 intervalos do H e 4 intervalos do S. O valor de cada dimensão do vetor de características é a densidade de cada cor na imagem inteira. A distância de Manhattan (L_1) foi usada para comparar esses vetores.

- DNA[7] — Um conjunto de 20.660 sequências genômicas da *Listeria monocytogenes*. Cada sequência é uma cadeia de caracteres de tamanho variável. Essas sequências foram comparadas por meio da distância de Levenshtein [75], que calcula o número mínimo de operações necessárias para transformar uma sequência em outra. Tais operações consistem na inserção, remoção e substituição de um único caractere.

[4] *The Amsterdam Library of Object Images*. Disponível em: `http://staff.science.uva.nl/~aloi/` (último acesso em 14 de outubro de 2011)

[5] *The Caltech-256 Object Category Data Set*. Disponível em: `http://www.vision.caltech.edu/Image_Datasets/Caltech256/` (último acesso em 14 de outubro de 2011)

[6] *The Corel Image Features*. Disponível em: `http://kdd.ics.uci.edu/databases/CorelFeatures/CorelFeatures.html` (último acesso em 14 de outubro de 2011)

[7] *Broad Institute of MIT and Harvard*. Disponível em: `http://www.broad.mit.edu/annotation/genome/listeriagroup/MultiDownloads.html.` (último acesso em 14 de outubro de 2011)

- MNIST[8] — Contém 60.000 imagens de dígitos numéricos escritos a mão, estudadas por Lecun *et al.* [100]. Cada imagem foi convertida em um vetor de características de 128 dimensões usando um Histograma de Orientações do Gradiente [54]. Esses histogramas foram extraídos como se segue: a imagem foi decomposta em 4×4 partições de mesmo tamanho que foram individualmente descritas usando um histograma de orientação das magnitudes do gradiente. Cada orientação tem 8 células, cobrindo a faixa de 360 graus de orientações possíveis. O vetor de características é formado por um vetor contendo os valores de todas as entradas desses 16 histogramas, o qual foi normalizado para o tamanho unitário. A função de distância usada para comparar esses vetores é a distância Euclidiana (L_2).

- TREC — Trata-se da coleção de 11.500 vídeos do conjunto de dados do TRECVID 2010 (IACC.1), utilizada nos experimentos do capítulo anterior. Cada vídeo foi convertido em um vetor de características de 65.536 dimensões usando a abordagem apresentada no Capítulo 5. Esses vetores foram comparados por meio da distância de Manhattan (L_1).

A escolha dessas coleções se deve apenas à conveniência de obter grandes conjuntos de dados métricos nos quais um gabarito razoável pode ser estabelecido. Independentemente disso, o esquema proposto em si não usa qualquer propriedade relacionada com a natureza desses dados.

O real interesse se concentra no espaço métrico gerado por vetores de características usados para codificar as suas propriedades. Diferentes espaços métricos podem ser obtidos a partir de um mesmo conjunto de dados, dependendo do algoritmo empregado para extrair tais vetores e da função de distância usada para medir a dissimilaridade entre eles.

Uma vez que essas bases de dados frequentemente produzem espaços métricos que são muito difíceis de indexar, devido a sua alta dimensionalidade intrínseca, elas formam um excelente ambiente de testes. Porém, vale ressaltar que o método proposto pode ser aplicado a qualquer tipo de dado, uma vez que se trata de um método de acesso métrico e, portanto, apenas distâncias são usadas na indexação.

A Figura 6.6 apresenta os histogramas de distribuição de distâncias de cada base de dados. Essas curvas indicam a probabilidade de que a distância entre dois objetos quaisquer seja menor ou igual a um certo valor [174]. A partir delas é possível se ter uma ideia da dimensionalidade intrínseca do espaço de dados. Espaços métricos de alta dimensionalidade produzem um histograma de distribuição de distâncias cujos valores estão mais concentrados em torno de sua média, enquanto os de baixa dimensionalidade apresentam uma distribuição mais uniforme [46].

[8] *The MNIST database*. Disponível em: `http://yann.lecun.com/exdb/mnist/` (último acesso em 14 de outubro de 2011)

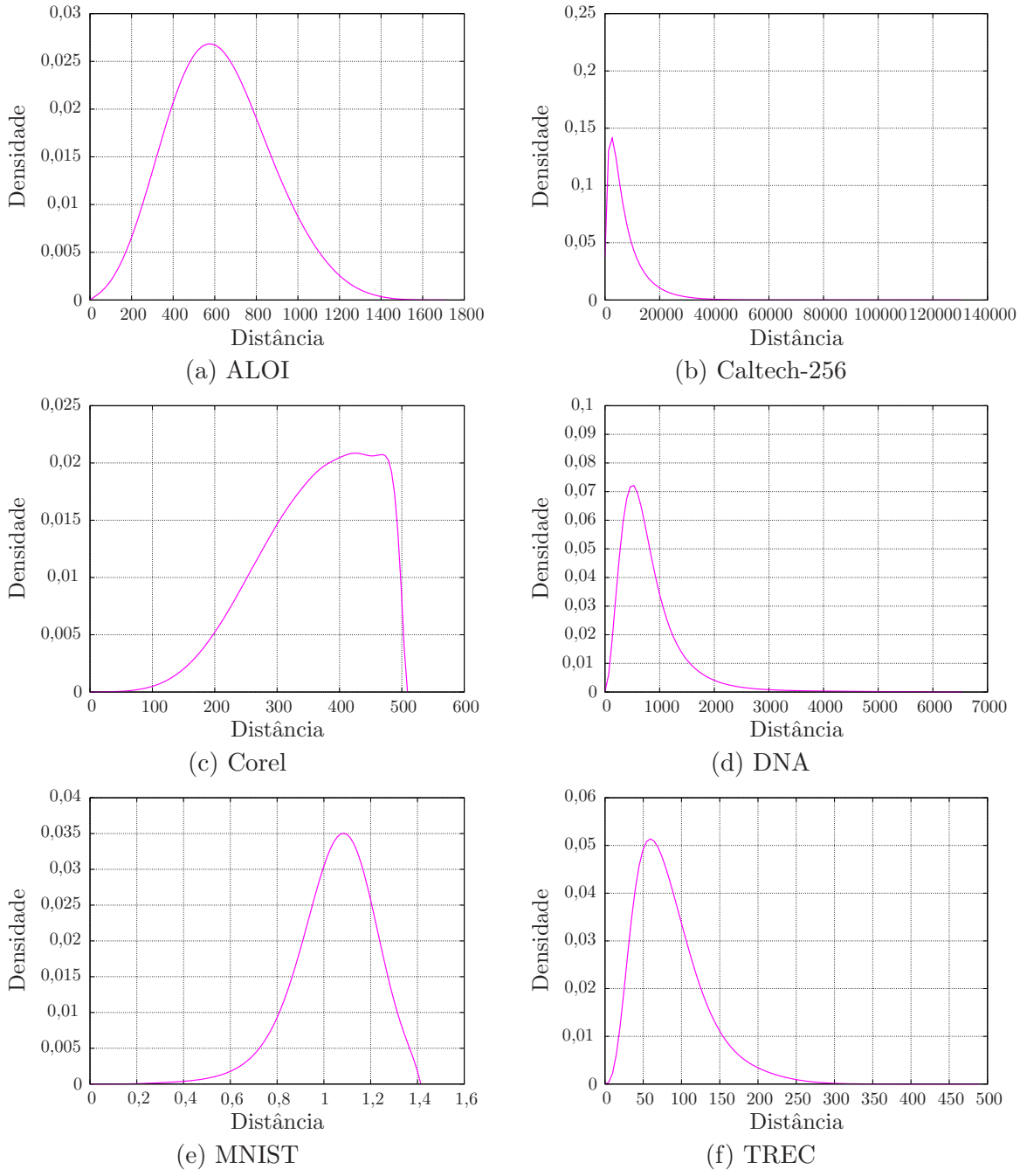


Figura 6.6: Histograma de distribuição de distâncias de cada base de dados.

A Tabela 6.2 sumariza as principais características de cada coleção, tais como o número total de objetos (N), o tipo do espaço métrico (T), a dimensionalidade embutida (E), a função de distância usada para comparar objetos ($d(x, y)$) e o tamanho da página em KBytes (D), o qual foi ajustado para suportar uma capacidade média (C) de 64 objetos.

Tabela 6.2: As principais características das bases de dados usadas nos experimentos.

Coleção	N	T	E	$d(x, y)$	D
ALOI	72.000	vetorial	256	L_1	16
Caltech-256	28.123	adimensional	-	EMD	128
Corel	68.040	vetorial	32	L_1	4
DNA	20.660	adimensional	-	Levenshtein	64
MNIST	60.000	vetorial	128	L_2	32
TREC	11.524	vetorial	65.536	L_1	4096

Para cada coleção, foram selecionados aleatoriamente cerca de 5% do número total de objetos para serem usados como objetos de consulta. Cinco replicações foram realizadas a fim de assegurar resultados estatisticamente sólidos. Em seguida, foram realizadas consultas por abrangência (R-NN) e por k vizinhos mais próximos (k-NN). O raio de busca foi variado de 0,01% a 10% da maior distância entre pares de objetos (raio de cobertura do conjunto inteiro), pois essa é a faixa de abrangência solicitada mais significativa na realização de consultas por similaridade. O número k de vizinhos mais próximos requisitados para a consulta foi variado de 2 a 20.

As medidas tomadas em cada experimento foram o número médio de cálculos de distância, o número médio de acessos a disco e o tempo médio (em milissegundos) necessários para realizar uma consulta. Para cada coleção, intervalos de confiança (a um nível de confiança de 99,9%) para cada medida foram calculados com base na média e no desvio padrão de cada replicação. Nos gráficos a seguir, reportam-se a média e seus intervalos de confiança como uma função do intervalo de consulta.

As Figuras 6.7 e 6.8 comparam a eficiência da BP-tree e trabalhos anteriores para responder a consultas R-NN e k-NN, respectivamente. Esses gráficos mostram o número médio de cálculos de distância para as seis bases de dados.

De modo geral, os resultados indicam que BP-tree é robusta a diferentes tipos de espaços métricos, mostrando uma alta eficiência em comparação com as soluções tradicionais. Em espaços vetoriais (ALOI, Corel, MNIST e TREC), BP-tree supera todos os índices comparados em relação ao número de cálculos de distância (nível de confiança superior a 99,9%). Em espaços adimensionais (Caltech-256 e DNA), ele realiza, em média, menos cálculos de distância que quase todos esses métodos, exceto MVP-tree e SAT.

Ao aumentar o raio de busca das consultas por abrangência, BP-tree tende a apresentar um desempenho melhor que MVP-tree e SAT, reduzindo em até 70% o número médio de cálculos de distância. Para consultas k-NN, BP-tree economiza, em média, 50% de cálculos de distância em relação ao seu melhor concorrente. A exceção em ambos os tipos de consultas por similaridade é para as coleções Caltech-256 e DNA.

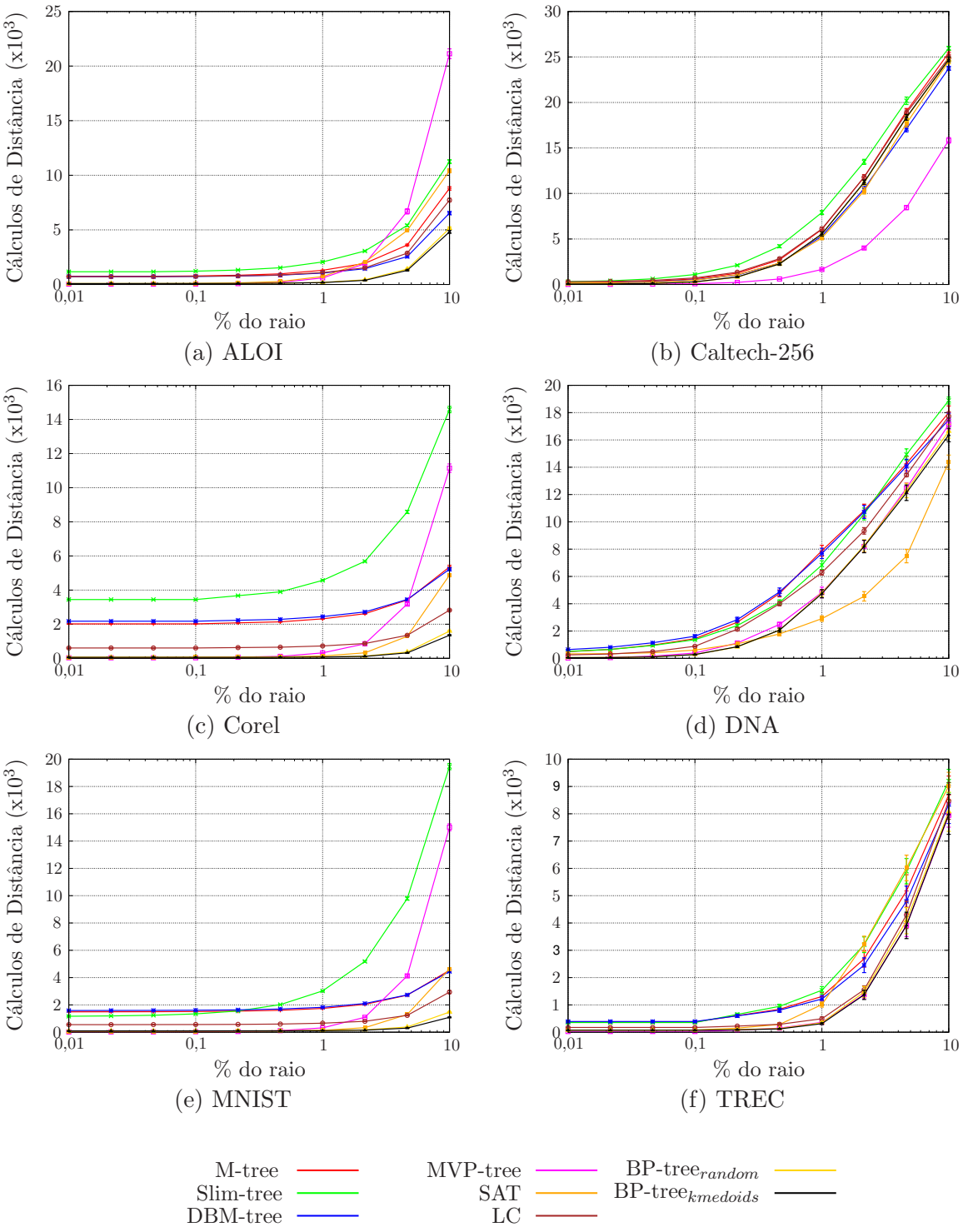


Figura 6.7: A eficiência de busca (em termos do número médio de cálculos de distância) dos métodos comparados como uma função do raio de consulta (R-NN), em que cada gráfico se refere a uma das coleções.

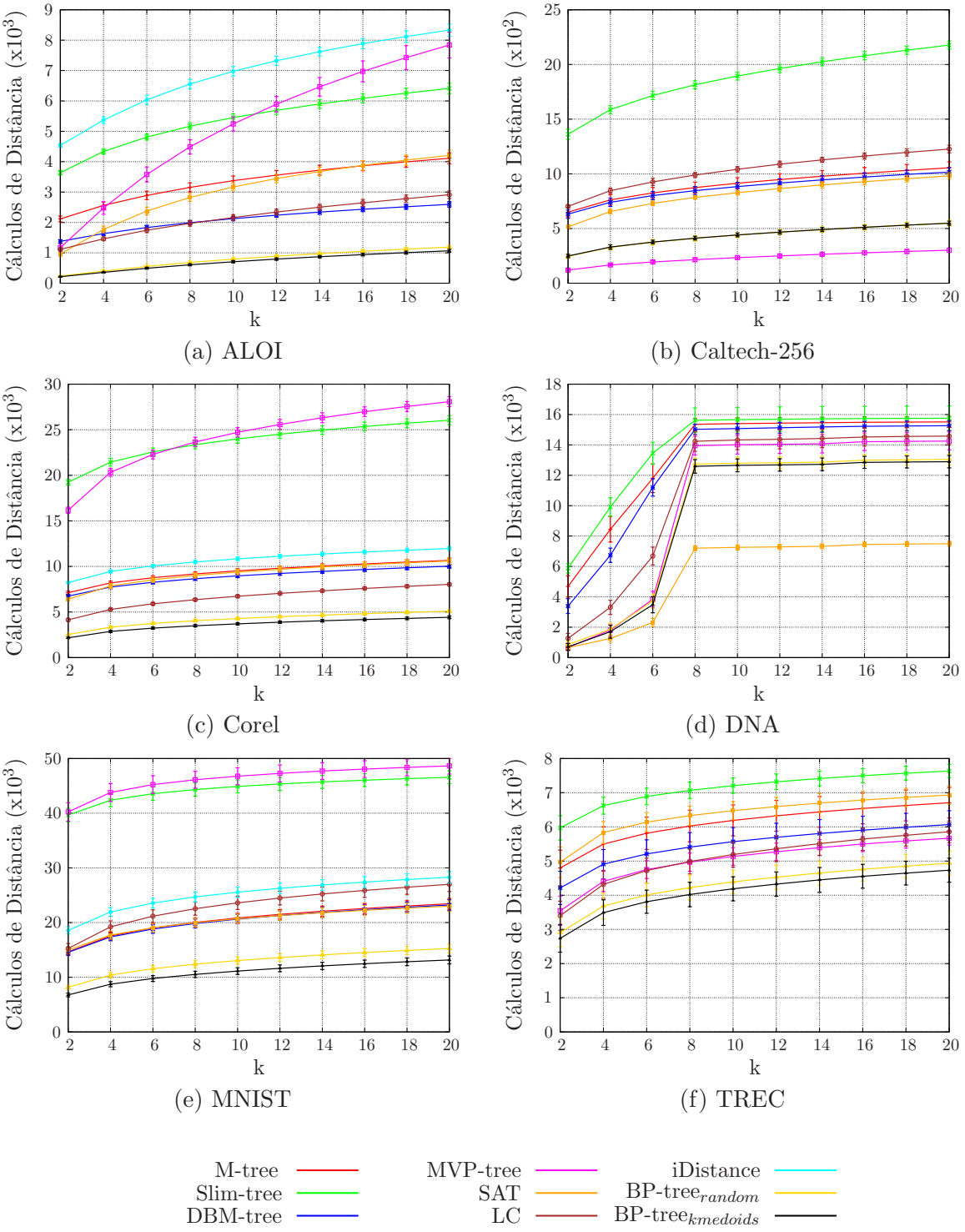


Figura 6.8: A eficiência de busca (em termos do número médio de cálculos de distância) dos métodos comparados como uma função do número k de vizinhos mais próximos (k-NN) requisitados para a consulta, em que cada gráfico se refere a uma das coleções.

Uma das razões para esses resultados é o fato dos métodos MVP-tree e SAT, ao contrário das demais técnicas, serem projetados para o uso em memória principal. Dessa forma, eles não são limitados por diversas restrições envolvidas ao se trabalhar em disco, por exemplo, o uso de páginas de tamanho fixo. Conforme discutido na Seção 6.2, essa liberdade evita a quebra de grupos naturais de objetos. Tal situação ocorre com frequência em índices voltados ao disco, especialmente para espaços adimensionais. Essa característica é explorada nas folhas da BP-tree, melhorando a sua eficiência de busca.

É importante notar que BP-tree apresenta um desempenho melhor que MVP-tree e SAT para responder a consultas k-NN em espaços vetoriais (nível de confiança superior a 99,9%). Em geral, a classe de índices baseados em memória é bastante otimizada para reduzir o número de cálculos de distância.

A fim de avaliar o comportamento da abordagem proposta em relação ao número de acessos a disco para responder consultas por similaridade, BP-tree foi comparada com M-tree, Slim-tree, DBM-tree e iDistance, pois eles são os únicos que consideram custos de E/S em suas análises. Para garantir uma comparação justa, BP-tree foi implementada na biblioteca Arboretum, sob uma mesma otimização de código. Essa biblioteca oferece uma plataforma robusta e uniforme na qual pode-se estabelecer uma análise comparativa confiável entre diferentes MAMs.

As Figuras 6.9 e 6.10 comparam o número médio de acessos a disco de cada abordagem na realização de consultas R-NN e k-NN, respectivamente. Com exceção da coleção Caltech-256, BP-tree supera todas as técnicas supracitadas em relação ao número de acessos a disco para responder a consultas por abrangência (nível de confiança superior a 99,9%), quando o raio de busca é inferior a 1%.

É importante perceber que, para consultas R-NN com um raio muito grande, pode ser mais eficiente carregar a coleção inteira na memória (usando acessos sequenciais). Contudo, é comum esperar que o raio de busca seja bastante pequeno na maioria das consultas, de modo que o uso do índice seja benéfico.

Em espaços vetoriais, BP-tree economiza, em média, 70% de acessos a disco em relação ao seu melhor concorrente. Já em espaços adimensionais, à medida que se aumenta o intervalo de busca, BP-tree tende a realizar mais acesso a disco que M-tree e Slim-tree. Esse efeito é visível para a coleção Caltech-256.

Como pode ser visto nos gráficos da Figura 6.10, BP-tree requer, em média, um número maior de acessos a disco que M-tree e Slim-tree para responder a consultas k-NN nas coleções DNA, MNIST e TREC. Nas demais coleções, BP-tree apresenta um desempenho melhor que essas abordagens (nível de confiança superior a 99,9%). Utilizando a estratégia **kmedoids**, BP-tree reduz em até 10% o número médio de acessos a disco, quando comparado com M-tree. Esse ganho é maior em relação a Slim-tree, chegando a ser 40% mesmo se comparado à estratégia **random**.

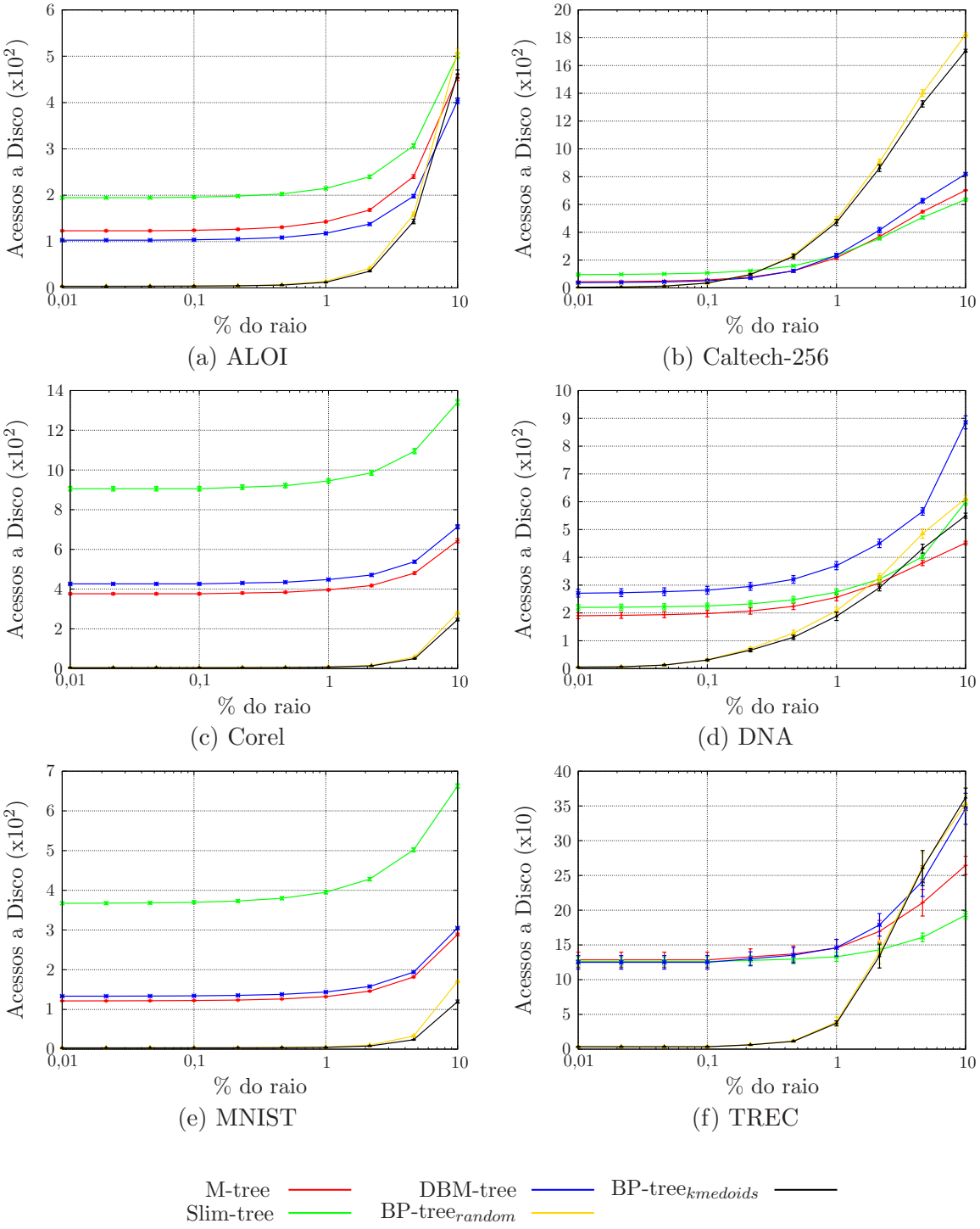


Figura 6.9: A eficiência de busca (em termos do número médio de acessos a disco) dos métodos comparados como uma função do raio de consulta (R-NN), em que cada gráfico se refere a uma das coleções.

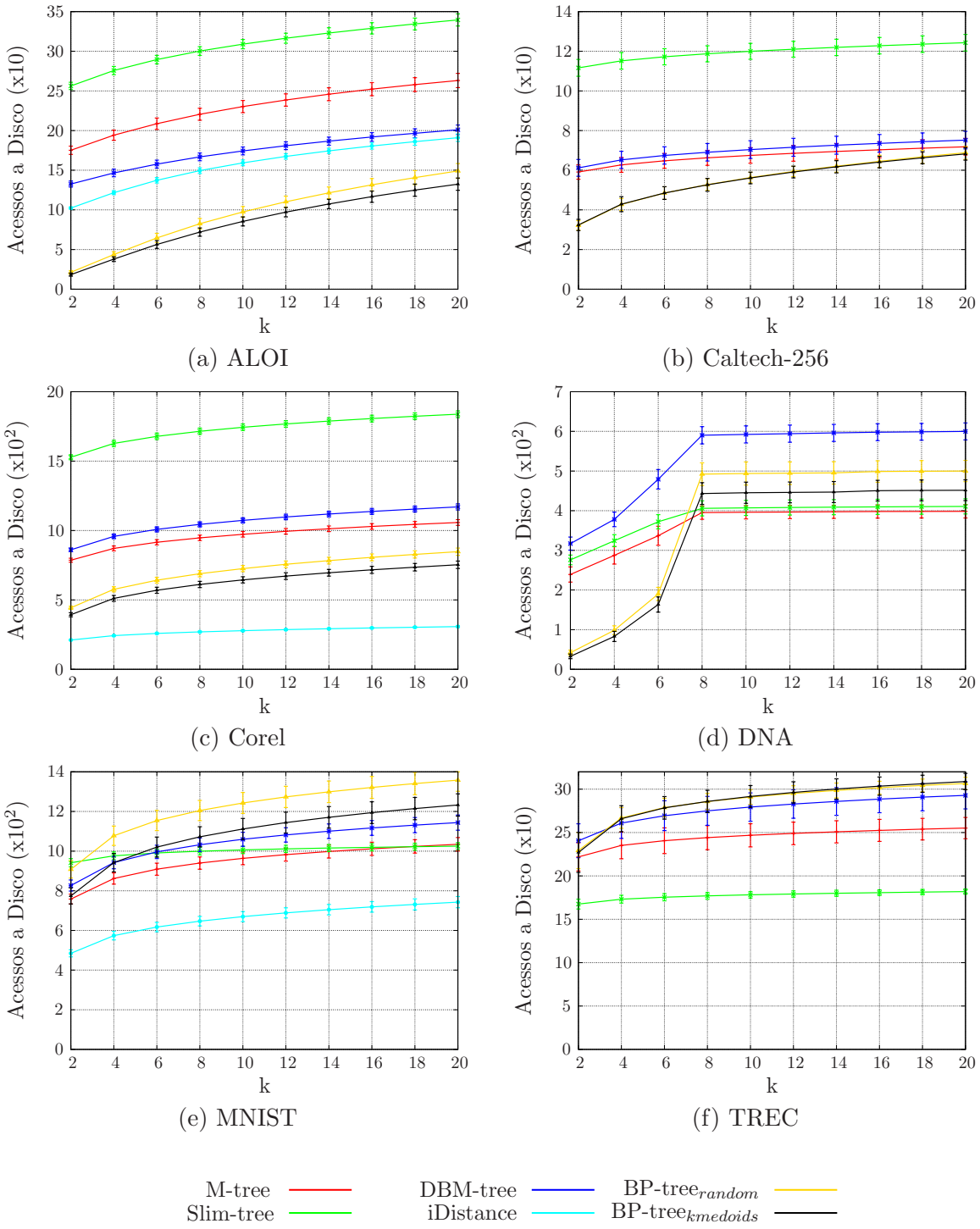


Figura 6.10: A eficiência de busca (em termos do número médio de acessos a disco) dos métodos comparados como uma função do número k de vizinhos mais próximos (k-NN) requisitados para a consulta, em que cada gráfico se refere a uma das coleções.

Isso ocorre devido ao desbalanceamento da BP-tree. Como se pode observar, um comportamento similar é realizado por DBM-tree, que também produz uma árvore desbalanceada. Porém, com exceção das coleções MNIST e TREC, BP-tree consistentemente supera DBM-tree (nível de confiança superior a 99,9%), reduzindo em até 10% o número médio de acessos a disco para responder a consultas k-NN.

Ao contrário dos resultados obtidos no experimento anterior, em que o iDistance executou um elevado número de cálculos de distância, tal índice requer menos acessos a disco que os demais métodos para responder a consultas k-NN nas coleções Corel e MNIST.

Em geral, o custo de E/S é considerado como a medida de eficiência mais importante em grandes bases de dados. Entretanto, algumas funções de distância são tão caras de se calcular, em termos do tempo de CPU, que o tempo total de busca é dominado pelo número total de cálculos de distância realizados e não pelo número total de páginas de disco acessadas, especialmente em espaços de alta dimensionalidade.

Esse efeito pode ser observado a partir da comparação do tempo de consulta desses índices, uma vez que ele reflete a complexidade total do algoritmo além do número de cálculos de distância e do número de acessos a disco.

O tempo médio (em milissegundos) exigido por essas abordagens para realizar consultas R-NN e k-NN é apresentado nas Figuras 6.11 e 6.12, respectivamente. É interessante notar que não há diferenças significativas de desempenho entre M-tree, Slim-tree, DBM-tree e BP-tree para realizar consultas R-NN na coleção Caltech-256.

Por outro lado, em todos os demais casos, BP-tree é mais rápida que esses métodos para a realização de consultas por similaridade (nível de confiança superior a 99,9%). Pode-se observar que BP-tree é, pelo menos, 20% mais rápida que seu melhor concorrente, em ambos os tipos de consulta.

Além disso, BP-tree executa muito bem em diferentes tipos de espaços métricos, mesmo para uma escolha não tão eficaz do esquema de particionamento e seleção de objetos representativos (isto é, a linha de base formada pela estratégia aleatória).

A principal razão para esses resultados é a forte interação entre as restrições geométricas e da estrutura de dados. Se elas não forem compatíveis, o índice como um todo sofre. Os métodos existentes geralmente ignoram propriedades de distribuição e, portanto, produzem regiões com um alto grau de sobreposição devido à alta dimensionalidade.

BP-tree visa preencher essa lacuna, criando uma estrutura de indexação que melhor se adapta à distribuição do espaço de dados. Essa estratégia permite minimizar o número de cálculos de distância e de acessos a disco para responder a ambos os tipos de consultas por similaridade. Tais benefícios são somados para reduzir o tempo de consulta.

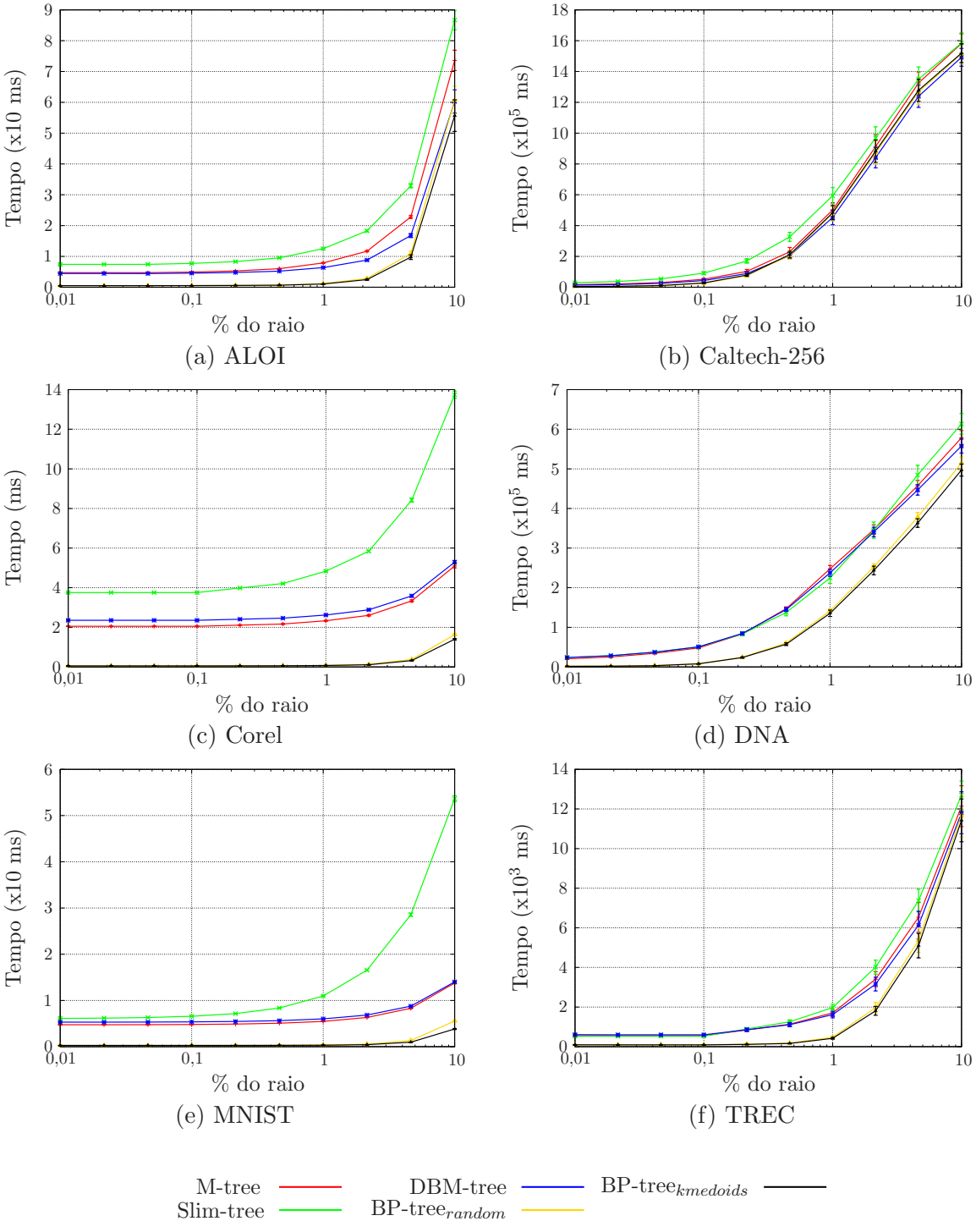


Figura 6.11: A eficiência de busca (em termos do tempo médio em milissegundos) dos métodos comparados como uma função do raio de consulta (R-NN), em que cada gráfico se refere a uma das coleções.

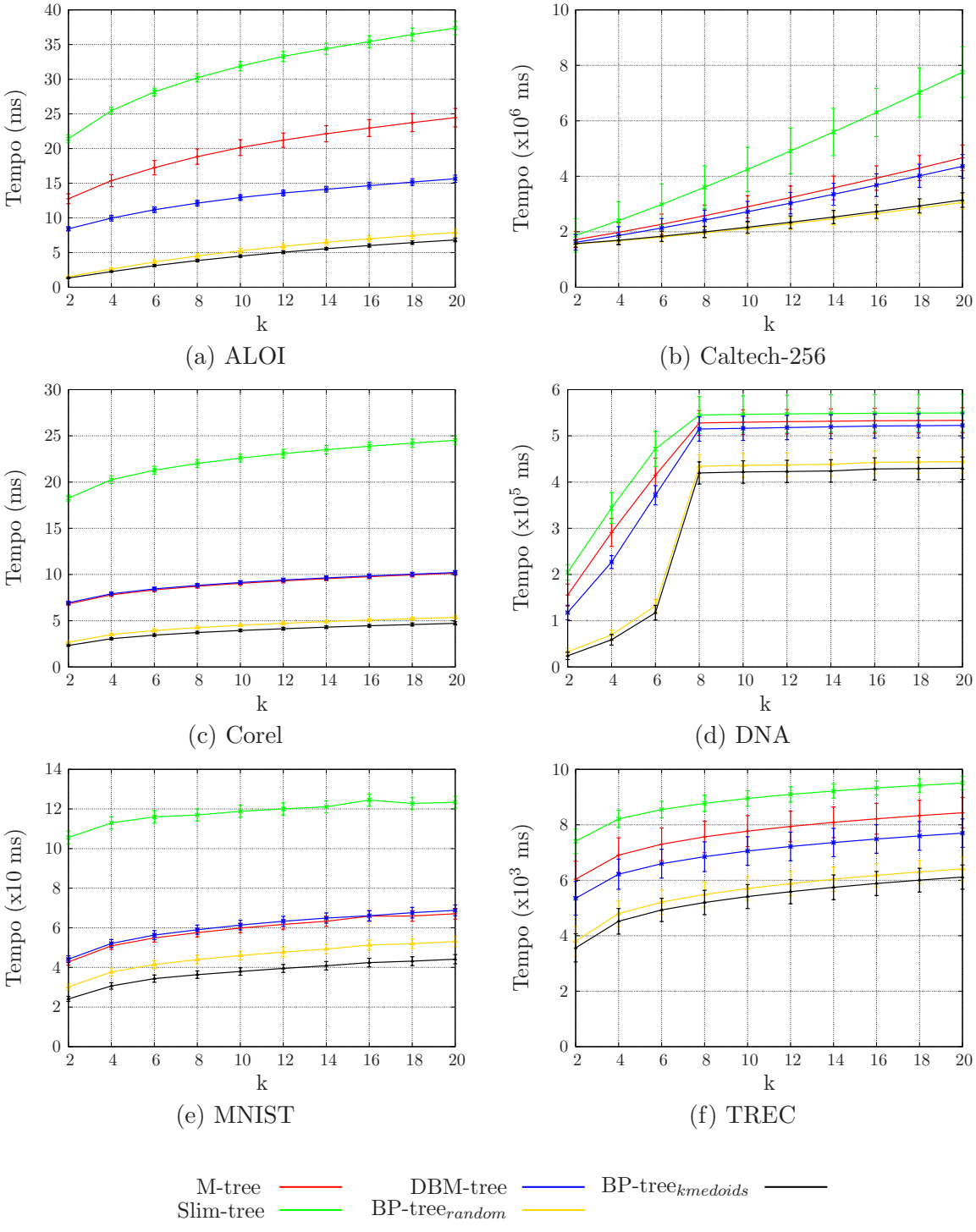


Figura 6.12: A eficiência de busca (em termos do tempo médio em milissegundos) dos métodos comparados como uma função do número k de vizinhos mais próximos (k-NN) requisitados para a consulta, em que cada gráfico se refere a uma das coleções.

6.3.2 Escalabilidade

Nesta seção, examina-se o comportamento da BP-tree com relação ao número de objetos indexados de um conjunto de entrada. A base de dados utilizada nesse experimento foi obtida a partir da extração de características locais de um conjunto de 3.280 imagens da coleção ETH-80[9] [102].

A extração de características locais dessas imagens foi realizada por meio de um método bastante conhecido, denominado SIFT (do inglês, *Scale Invariant Features Transform*) [108]. A coleção resultante contém um total de 134.173 descritores SIFT. Cada descritor SIFT é composto por um vetor de características de 128 dimensões. A função de distância usada para comparar esses vetores é a distância Euclidiana (L_2). O tamanho da página de disco adotado na construção de todos os índices comparados foi 16 KBytes, suportando uma capacidade média de 64 objetos. A Figura 6.13 mostra o histograma de distribuição de distâncias desse conjunto.

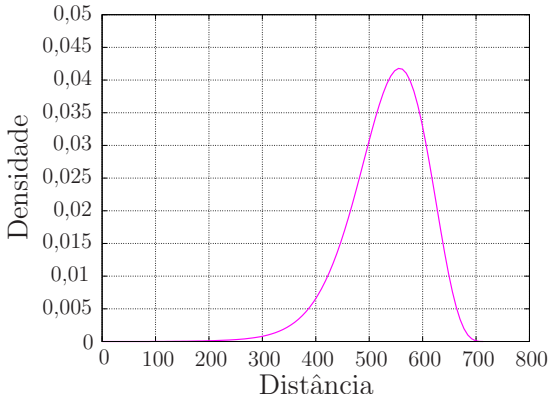


Figura 6.13: Histograma de distribuição de distâncias da coleção ETH-80.

Em um primeiro experimento, foi avaliado o custo de construção da BP-tree. Para esse propósito, a base de dados foi dividida aleatoriamente em dez partes de mesmo tamanho. Inicialmente, o experimento foi realizado usando uma das dez partes como o conjunto de entrada e, gradualmente, as nove partes restantes foram adicionadas, uma por uma, até que todas elas fossem inseridas.

A Figura 6.14 mostra como o tempo de construção (em segundos) da BP-tree cresce com o aumento do número de objetos indexados. Para comparação, apresenta-se também os resultados para M-tree, Slim-tree e DBM-tree. A escala logarítmica foi usada para destacar o comportamento de cada abordagem. Tal como os outros métodos, BP-tree

[9] `http://tahiti.mis.informatik.tu-darmstadt.de/oldmis/Research/Projects/categorization/eth80-db.html` (último acesso em 14 de outubro de 2011)

exibe um crescimento moderado com o aumento do tamanho do conjunto de objetos, indicando uma boa escalabilidade.

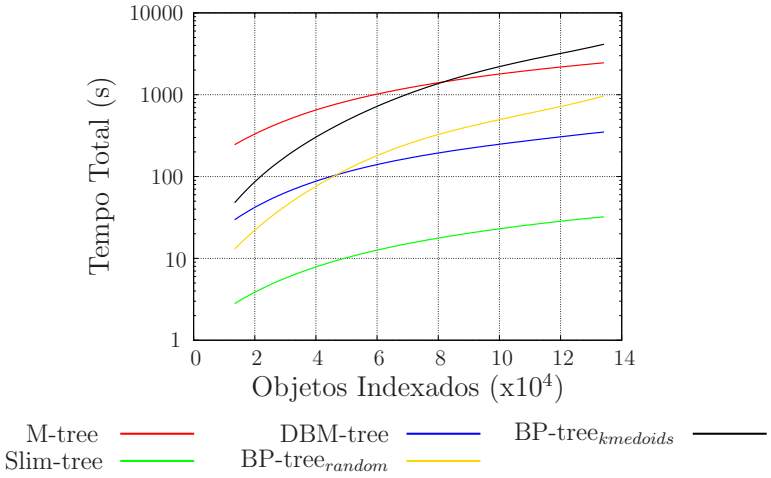


Figura 6.14: Comparação entre o tempo de construção (em segundos) de diferentes índices em relação ao tamanho do conjunto de objetos.

A ocupação de espaço (em termos do número de páginas de disco) desses índices é mostrada na Figura 6.15. Pode-se notar que os requisitos de espaço da BP-tree assemelham-se aos da DBM-tree. Por outro lado, as estruturas de indexação M-tree e Slim-tree requerem menos páginas de disco. Uma das razões é a melhor ocupação de nós em árvores balanceadas. Apesar disso, a utilização do dispositivo de armazenamento pela BP-tree escala muito bem com relação ao tamanho do conjunto de objetos.

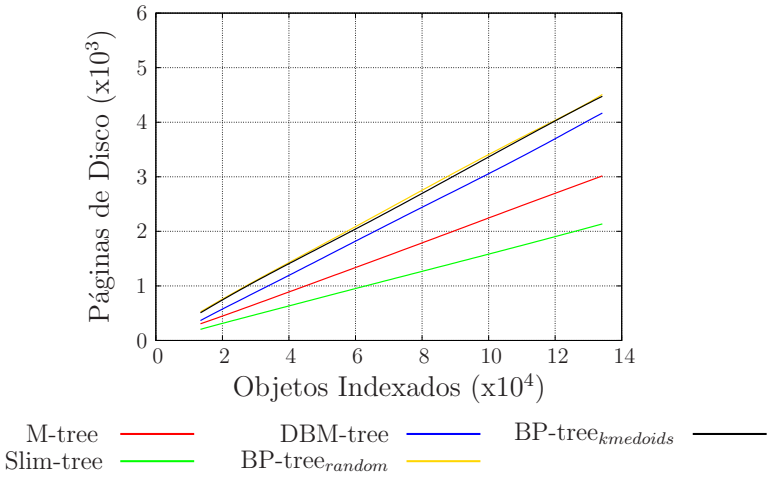


Figura 6.15: Comparação entre a ocupação de espaço (em termos do número de páginas de disco) de diferentes índices em relação ao tamanho do conjunto de objetos.

Para avaliar o custo de buscar um objeto no índice, foi selecionado aleatoriamente cerca de 1% (1300 descritores SIFT) do número total de objetos para serem usados como objetos de consulta. Cinco replicações foram realizadas a fim de assegurar resultados estatisticamente sólidos. O mesmo conjunto de consultas foi usado em cada uma das dez etapas do experimento.

O comportamento em cada etapa foi equivalente para consultas R-NN e k-NN com diferentes valores de R e k e, por isso, escolheu-se reportar apenas os resultados para os valores máximos, ou seja, o raio de busca ajustado para 10% do raio de cobertura do conjunto inteiro e o número k de vizinhos mais próximos igual a 20.

A Figura 6.16 indica o comportamento da BP-tree e de trabalhos anteriores mediante o aumento do número de objetos indexados. Os gráficos apresentam resultados para consultas por abrangência (coluna da esquerda) e por k vizinhos mais próximos (coluna da direita). Eles reportam o número médio de cálculos de distância (primeira linha), o número médio de acessos a disco (segunda linha) e o tempo médio (em milissegundos) para realizar uma consulta (terceira linha).

Esses gráficos mostram que, com o crescimento do conjunto de objetos, o número médio de cálculos de distância realizados pelas abordagens tradicionais aumenta mais rápido que aquele da BP-tree, em ambos os tipos de consultas por similaridade.

Como pode ser visto, aumentando-se o número de objetos indexados, BP-tree requer menos acessos a disco que seus concorrentes para consultas por abrangência. Por outro lado, BP-tree executa um pouco mais de acessos a disco que os demais índices para responder a consultas por k vizinhos mais próximos.

Não surpreendentemente, BP-tree apresenta um crescimento mais lento do tempo de busca que todos os métodos comparados. É importante observar que ela exibe um comportamento sublinear mediante o aumento do número de objetos indexados, tornando-o adequado para indexar grandes conjuntos.

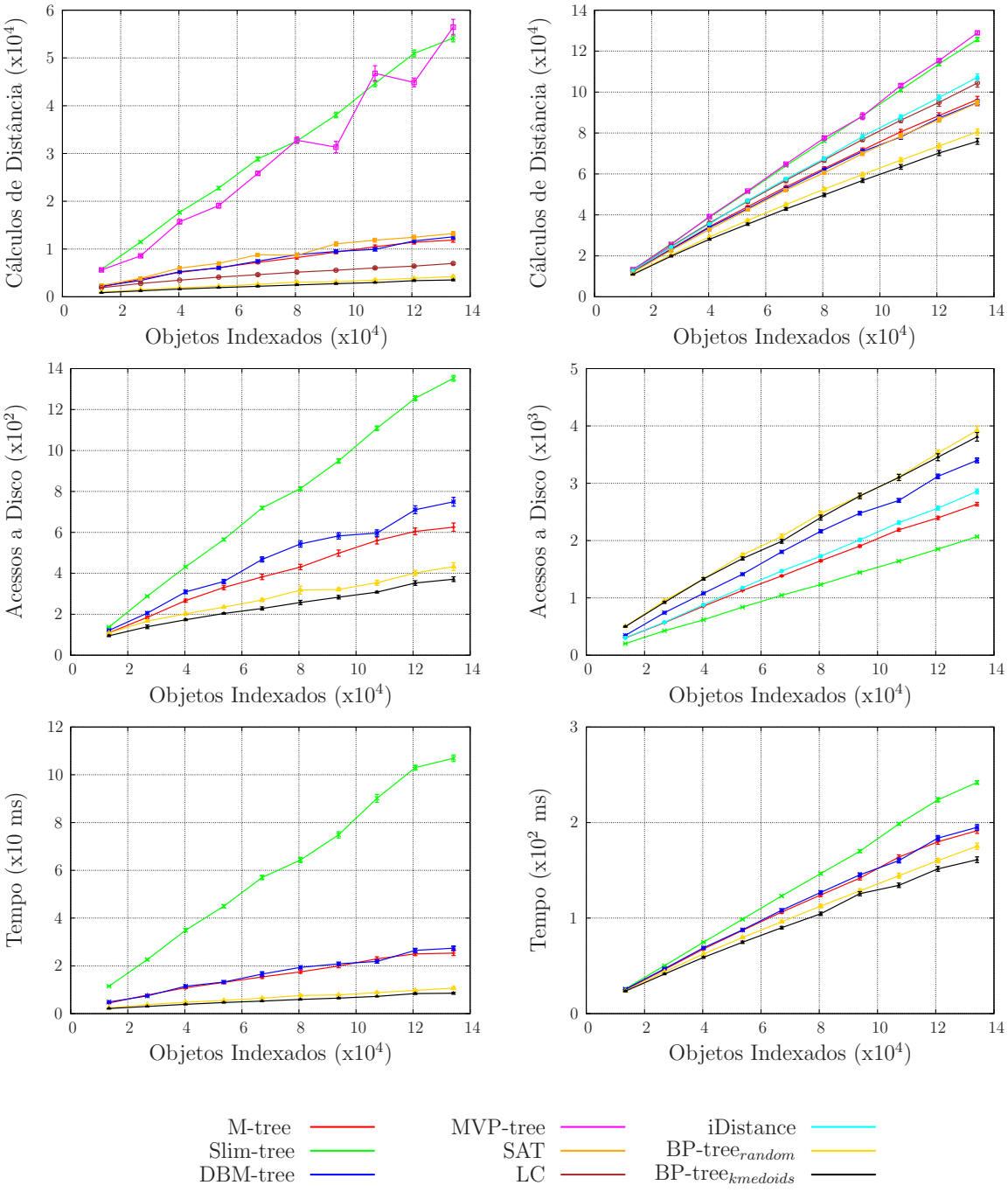


Figura 6.16: A escalabilidade de diferentes métodos em relação ao número de objetos da base de dados. Esses gráficos apresentam resultados para consultas R-NN (coluna da esquerda) e k-NN (coluna da direita). Eles reportam o número médio de cálculos de distância (primeira linha), o número médio de acessos a disco (segunda linha) e o tempo médio (em milissegundos) para realizar uma consulta (terceira linha).

6.3.3 Construção Incremental

Esta seção analisa a eficiência da estratégia proposta para a atualização da BP-tree (Seção 6.2.6). Seguindo o protocolo anterior, avaliou-se o custo de construção estática e incremental, bem como a eficiência de busca das estruturas construídas. Esses experimentos têm o intuito de demonstrar a viabilidade da construção incremental.

Inicialmente, o experimento foi realizado usando cinco das dez partes como o conjunto de entrada e, gradualmente, as cinco partes restantes foram adicionadas, uma por uma, até que todas elas fossem inseridas.

Nos gráficos que se seguem, **static** refere-se à construção estática da árvore, que é realizada pelo Algoritmo 1, e **dynamic** diz respeito ao índice obtido a partir de inserções sucessivas em uma estrutura inicial, empregando a abordagem discutida na Seção 6.2.6.

Utilizando a estratégia estática, o índice foi inteiramente reconstruído a cada iteração, usando todo o conjunto de dados disponível. Por outro lado, na estratégia dinâmica, ele foi construído uma única vez, durante a primeira iteração, e a partir de então, novos objetos de dados foram incrementalmente adicionados à estrutura criada.

A Figura 6.17 mostra como o tempo de construção (em segundos) das versões estática e dinâmica cresce com o aumento do número de objetos indexados. A escala logarítmica foi usada para destacar o comportamento de cada abordagem.

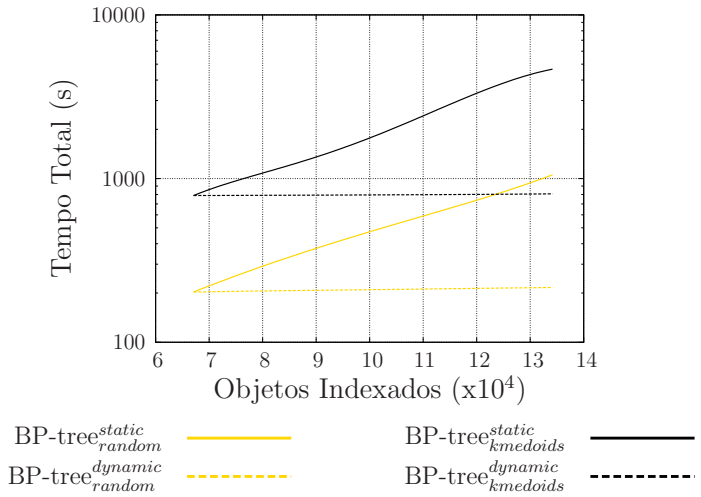


Figura 6.17: Comparação entre o tempo de construção (em segundos) das versões estática e dinâmica em relação ao tamanho do conjunto de objetos.

Como se pode observar, com o aumento do tamanho do conjunto de objetos, o tempo total para construir o índice usando a estratégia estática cresce rapidamente, enquanto na dinâmica, ele se mantém praticamente constante, indicando uma melhor escalabilidade.

A ocupação de espaço (em termos do número de páginas de disco) de ambas as versões é mostrada na Figura 6.15. É interessante notar que, ao aumentar o número de objetos indexados, a estratégia dinâmica requer menos páginas de disco a estática.

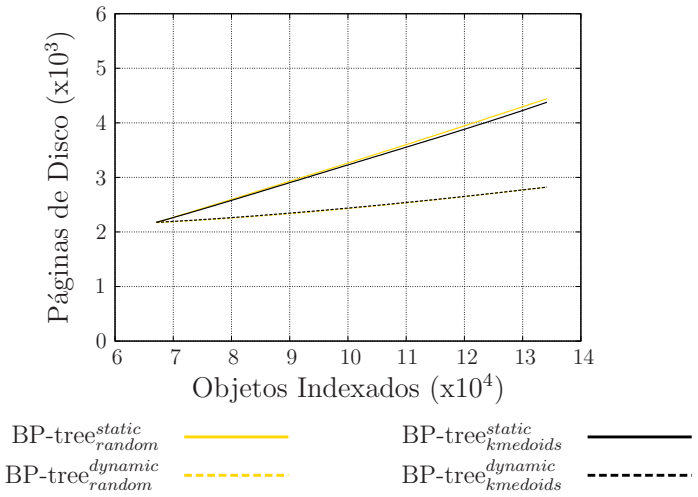


Figura 6.18: Comparação entre a ocupação de espaço (em termos do número de páginas de disco) das versões estática e dinâmica em relação ao tamanho do conjunto de objetos.

A Figura 6.19 compara a eficiência de busca das versões estática e dinâmica mediante o aumento do número de objetos indexados. Os gráficos apresentam resultados para consultas R-NN (coluna da esquerda) e k-NN (coluna da direita). Eles reportam o número médio de cálculos de distância (primeira linha), o número médio de acessos a disco (segunda linha) e o tempo médio para realizar uma consulta (terceira linha).

Esses gráficos mostram que, aumentando-se o número de objetos indexados, a estratégia estática requer menos cálculos de distância que a dinâmica. Porém, o número de acessos a disco realizados pela abordagem estática aumenta significativamente mais rápido que da dinâmica. Tal comportamento pode ser observado em ambos os tipos de consulta.

A principal razão para esses resultados é a melhor ocupação das folhas em árvores obtidas pelo método dinâmico. Isso leva à redução da quantidade de acessos aleatórios a disco, ao custo do aumento da quantidade de objetos analisados de maneira sequencial.

Apesar dessas diferenças, ambas as versões exibem um comportamento similar em relação ao tempo de busca. Isso mostra a capacidade da BP-tree em se adaptar à distribuição do espaço de dados, de modo que, enquanto as características geométricas do espaço são mantidas, a inserção de novos objetos não afeta a eficiência do índice.

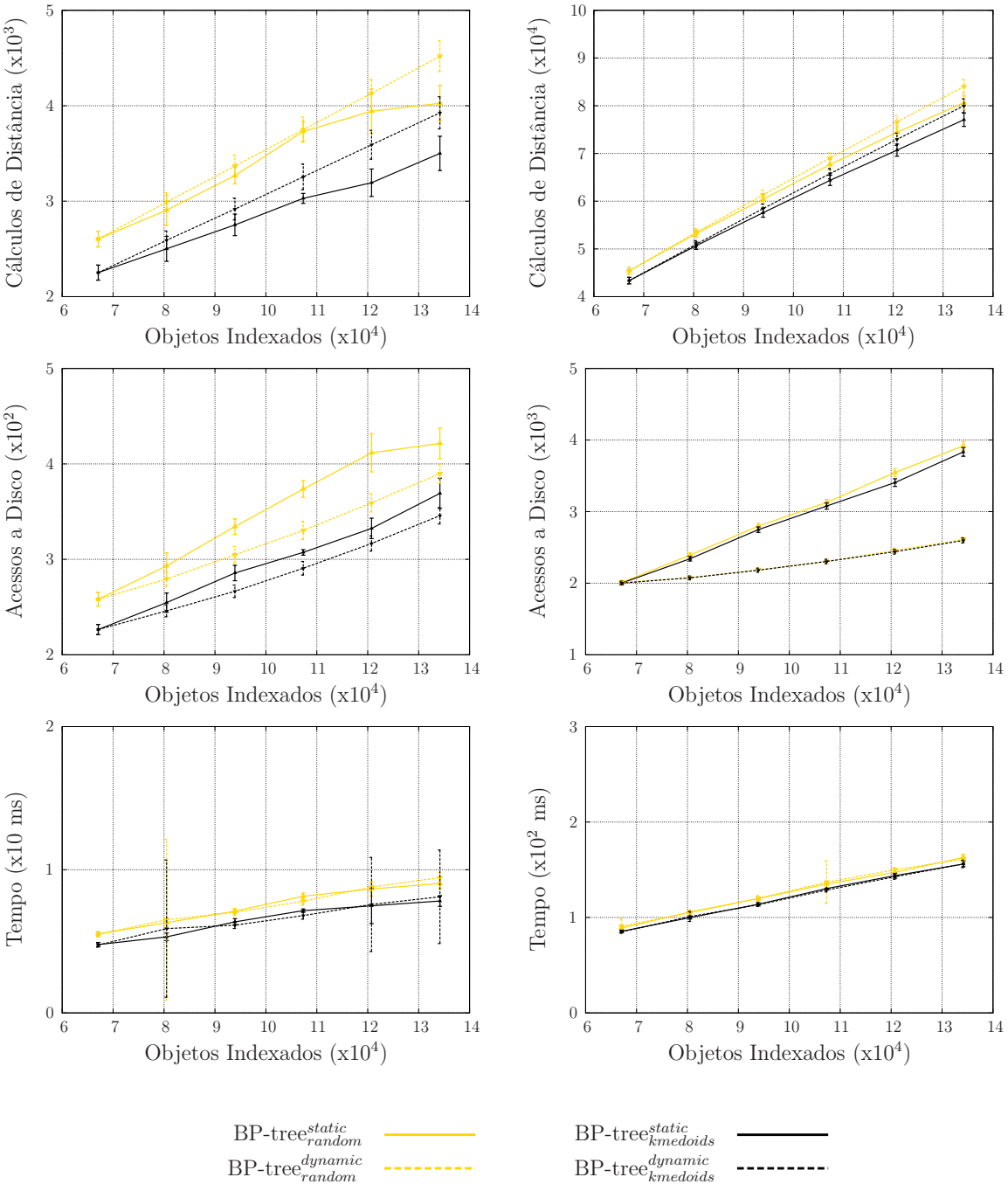


Figura 6.19: Comparação entre a eficiência de busca das versões estática e dinâmica em relação ao número de objetos da base de dados. Esses gráficos apresentam resultados para consultas R-NN (coluna da esquerda) e k-NN (coluna da direita). Eles reportam o número médio de cálculos de distância (primeira linha), o número médio de acessos a disco (segunda linha) e o tempo médio para realizar uma consulta (terceira linha).

6.4 Conclusões

Neste capítulo, foi apresentada BP-tree, uma nova abordagem para a realização de buscas por similaridade em espaços métricos. Nesse método, um conjunto de objetos é dividido em grupos compactos, respeitando-se a sua distribuição natural. Ele combina vantagens de ambas as estratégias disjuntas (partições) e não disjuntas (agrupamentos) a fim de obter uma estrutura de aglomerados densos e pouco sobrepostos, melhorando significativamente a eficiência no processamento de consultas por similaridade, especialmente em espaços de alta dimensionalidade.

Experimentos realizados sobre bases de dados reais têm mostrado que a BP-tree consistentemente supera os índices populares na literatura de busca por similaridade em espaços métricos. BP-tree é, em média, 50% mais rápido que as abordagens tradicionais na realização de consultas por similaridade. Além disso, ele escala muito bem com respeito ao número de objetos indexados, apresentando um comportamento sublinear, que o torna bastante adequado para indexar grandes coleções.

Extensões futuras incluem a melhoria do processo de construção da BP-tree, possivelmente através da produção de várias subárvores em paralelo, pois elas são totalmente independentes umas das outras. Pretende-se também ampliar BP-tree para executar o reparticionamento regional de subárvores a fim de melhorar o suporte à construção incremental. Além disso, seria interessante desenvolver algoritmos aproximados ou probabilísticos com base nessa estrutura, pois eles provaram ser de grande interesse em espaços métricos extremamente difíceis usando outras estruturas de dados, as quais tipicamente funcionam bem apenas em espaços mais fáceis [4, 5, 14, 139].

Capítulo 7

Visualização dos Resultados

Fazer uso eficiente da informação de vídeo requer que ela seja acessada de forma amigável. Para isso, é importante fornecer aos usuários uma representação concisa do vídeo, dando uma ideia do seu conteúdo, sem ter que vê-lo completamente, de modo que um usuário possa decidir se quer ou não assistir ao vídeo todo. Isso tem sido a meta de uma área de investigação conhecida como sumarização de vídeo [63].

Diferentes técnicas têm sido propostas na literatura para resolver o problema de resumir sequências de vídeo [22, 36, 42, 63, 69, 81, 82, 95, 99, 119, 126, 134, 155, 172, 180]. Muitos desses trabalhos de pesquisa estão centrados no domínio descomprimido. Embora as abordagens existentes elaborem resumos com qualidade aceitável, a decodificação e a análise de uma sequência de vídeo são duas tarefas extremamente lentas e requerem uma enorme quantidade de espaço. Por essas razões, tais métodos não são adequados para uso *online* e, portanto, os resumos de vídeo são muitas vezes produzidos completamente *offline*, armazenados e entregues a um usuário quando solicitado. O inconveniente desse sistema é a completa falta de personalização pelo usuário.

Neste capítulo, é apresentado VISON (do inglês, *VIdeo Summarization for ONline applications*), uma nova abordagem para a sumarização de sequências de vídeo que atua diretamente no domínio comprimido. Ela baseia-se na exploração de características visuais extraídas do fluxo de vídeo e em um algoritmo simples e rápido para resumir o seu conteúdo. A melhoria da eficiência computacional torna essa técnica adequada para tarefas *online*. O método proposto foi projetado para oferecer customização: os usuários podem controlar a qualidade dos resumos e também especificar o tempo que eles estão dispostos a esperar. Tal interação do usuário é cada vez mais importante no cenário atual [13, 19, 20].

O algoritmo proposto foi avaliado em um conjunto de dados do TRECVID 2007 (BBC Rushes) e também em vídeos do Open Video e do YouTube, e comparado com abordagens recentes na literatura de sumarização de sequências de vídeo. Os experimentos foram

cuidadosamente projetados a fim de assegurar significância estatística. Resultados de uma avaliação subjetiva com usuários mostram que esse método produz resumos de vídeo com alta qualidade e velocidade computacional.

O restante deste capítulo está organizado como segue. A Seção 7.1 introduz conceitos básicos e descreve trabalhos relacionados. A Seção 7.2 apresenta a abordagem proposta e mostra como aplicá-la na sumarização de sequências de vídeo. A Seção 7.3 discute o protocolo experimental em termos da coleção de vídeos e medidas de eficácia. A Seção 7.4 reporta os resultados experimentais e compara essa técnica com outros métodos. Por fim, são oferecidas considerações finais e direções para trabalhos futuros na Seção 7.5.

7.1 Revisão da Literatura

Um resumo de vídeo é uma versão reduzida de uma sequência de vídeo. Existem dois tipos diferentes de resumos de vídeo [22]: história em quadrinhos (do inglês, *storyboard*), que é uma coleção de quadros-chave extraídos do vídeo original, resultando em resumos estáticos; e trailer de vídeo (do inglês, *video skim*), que é um conjunto de tomadas curtas, unidas em uma sequência e exibidas como um vídeo, resultando em resumos dinâmicos.

Uma vantagem dos resumos dinâmicos em relação aos estáticos é a possibilidade de incluir elementos de áudio e movimentos que, potencialmente, aumentam a expressividade e a informatividade do resumo. Além disso, muitas vezes é mais divertido e interessante assistir ao trailer de um vídeo do que a uma apresentação de *slides*. Por outro lado, como os resumos estáticos não são limitados por qualquer característica de tempo ou sincronização, uma vez que os quadros são extraídos, existem outras possibilidades de organizá-los para fins de navegação além da exibição sequencial dos trailers de vídeo [39, 107].

Uma ampla revisão de métodos para lidar com a sumarização de sequências de vídeo é apresentada por Money e Agius [118] e Truong e Venkatesh [163]. Algumas das principais características das técnicas já publicadas são brevemente discutidas a seguir.

A maioria dos métodos existentes para a sumarização de vídeo está centrada no domínio descomprimido. Devido ao longo tempo gasto para decodificar e analisar uma sequência de vídeo, os resumos são muitas vezes produzidos completamente *offline*, armazenados e entregues a um usuário quando solicitado. A produção *offline* (total ou parcial) visa reduzir o tempo computacional para avaliar todos os quadros de vídeo. Normalmente, esses quadros são convertidos em uma matriz enorme que contém todas as características extraídas de cada um deles. Por essas razões, tais métodos são inadequados para a produção de resumos de vídeo em tarefas *online*.

Zhuang *et al.* [180] propuseram um algoritmo para agrupar quadros com cores similares. Inicialmente, a sequência de vídeo é dividida em tomadas por meio de um algoritmo de agrupamento baseado em histogramas de cor. Para cada tomada, o quadro mais pró-

ximo do centro do grupo é escolhido como o quadro representativo. Apenas um quadro por tomada é selecionado para o resumo, independente da duração ou atividade do vídeo. O número de grupos é definido a priori usando um limiar de densidade, o que pode comprometer a qualidade do resultado.

Hanjalic e Zhang [81] desenvolveram uma abordagem semelhante ao método anterior, a qual encontra o agrupamento ideal a partir da análise de qualidade dos grupos. Para isso, um algoritmo de agrupamento particional é aplicado várias vezes em todos os quadros do vídeo de entrada. O número de grupos começa em um e é aumentado de um a cada iteração. Uma vez que o valor ideal é encontrado, cada um dos grupos é resumido usando o quadro mais representativo.

Gong e Lin [69] apresentaram uma técnica de sumarização que se baseia na decomposição em valores singulares (do inglês, *Singular Value Decomposition* – SVD). A princípio, muita informação redundante é descartada por meio de amostragem temporal e, assim, ao invés de considerar todos os quadros, apenas um subconjunto é tomado. Em seguida, histogramas de cor são usados para representar os quadros restantes. A fim de incorporar informações espaciais, cada quadro é dividido em 3×3 blocos e um histograma é obtido para cada um deles. Esses nove histogramas são concatenados para formar um vetor de características. Depois disso, a SVD é realizada em uma matriz formada por todos os vetores, obtendo-se uma outra matriz, na qual cada coluna representa um quadro em um espaço de características mais refinado. No final, o valor singular do quadro mais próximo da origem de tal espaço é usado como limiar para agrupar os quadros restantes.

Mundur *et al.* [119] propuseram uma abordagem similar à técnica acima, que usa a análise de componentes principais (do inglês, *Principal Component Analysis* – PCA) para reduzir a dimensionalidade do espaço de caraterísticas. Além disso, este método utiliza um algoritmo de agrupamento baseado na triangulação de Delaunay (do inglês, *Delaunay Triangulation* – DT). Ele inicia-se com uma amostragem temporal dos quadros do vídeo original. Esses quadros são convertidos em histogramas de cor e armazenados como uma matriz. Em seguida, a PCA é aplicada sobre tal matriz, reduzindo a sua dimensão. Depois disso, o diagrama de Delaunay é construído. Por fim, os grupos são obtidos a partir da separação de bordas nesse diagrama.

Furini *et al.* [63] introduziram STIMO (do inglês, *STIll and MOving Video Storyboard*), uma técnica de sumarização projetada para produzir resumos em tempo de execução. Inicialmente, é aplicada uma etapa de amostragem temporal, descartando quadros redundantes. Os quadros remanescentes são convertidos em histogramas de cor e armazenados em uma matriz. Em seguida, quadros semelhantes são agrupados por um método que se baseia em uma versão melhorada do algoritmo FPF (do inglês, *Furthest-Point-First*). Os autores exploraram a desigualdade triangular com o intuito de reduzir cálculos de distância desnecessários. Para a obtenção do número de grupos, a dissimilaridade entre pares de

quadros consecutivos é calculada de acordo com a distância generalizada de Jaccard (do inglês, *Generalized Jaccard Distance* – GJD). Normalmente os picos de dissimilaridade estão associados a movimentos bruscos na cena ou a transições entre tomadas. No final, uma etapa de pós-processamento é realizada para a remoção de quadros insignificantes do resumo produzido. Apesar de tal esquema ser projetado para permitir o uso em tempo de execução e a personalização do usuário, ele requer o cálculo a priori da matriz de características, tornando-o inviável em tarefas *online*.

Avila *et al.* [22] apresentaram VSUMM (do inglês, *Video SUMMarization*), um método semelhante a essa última abordagem, no qual a etapa de agrupamento é realizada por uma versão melhorada do algoritmo de k-médias [33]. Os autores exploraram o alinhamento temporal dos quadros de vídeo a fim de acelerar a convergência do método. Para isso, os quadros são inicialmente agrupados em ordem sequencial ao invés de distribuídos aleatoriamente entre os grupos. A seguir, eles são agrupados pelo algoritmo de k-médias tradicional. Por fim, um quadro por grupo é selecionado para compor o resumo.

Todos esses métodos foram projetados para obter uma coleção de quadros a partir de uma sequência de vídeo, isto é, um resumo estático. Uma série de pesquisas em abordagens dinâmicas têm sido feita no âmbito da tarefa de sumarização de vídeos da BBC (do inglês, *British Broadcasting Corporation*), organizada pela equipe do TRECVID [124, 125]. Essa tarefa tem como objetivo criar automaticamente um trailer com tamanho inferior a 4% do vídeo completo, incluindo os principais objetos e eventos. A seguir, são discutidas algumas das principais ideias e resultados entre os trabalhos desenvolvidos nessa tarefa.

Pan *et al.* [126] apresentaram uma técnica de sumarização que se baseia em um algoritmo de detecção de transições. A princípio, os limites das tomadas são detectados a partir da comparação de quadros consecutivos usando histogramas de cor. Depois disso, um algoritmo de agrupamento é usado para unir tomadas similares e uma tomada representativa é selecionada para cada grupo, removendo trechos duplicados. No final, esses segmentos são comprimidos em um resumo de acordo com sua importância para o grupo.

Kleban *et al.* [95] introduziram um método que emprega uma fusão de características de alto nível. Inicialmente, uma etapa de amostragem temporal é aplicada com o intuito de descartar informações redundantes, mantendo apenas um subconjunto de quadros de vídeo. A seguir, cinco características de alto nível são extraídas dos quadros restantes. Então, os pesos ótimos para combinar esses recursos são obtidos pelo método de gradiente. Por fim, um algoritmo de k-médias ponderadas é usado para identificar os segmentos mais importantes que constituem o resumo final.

Le e Satoh [99] propuseram um algoritmo para agrupar quadros com cores semelhantes. Ele inicia-se decompondo o vídeo de entrada em fragmentos por meio do agrupamento de quadros com base em momentos de cor. Em seguida, o método GreedyRSC [83] é empregado para unir esses fragmentos. Depois disso, eles combinam fragmentos consecutivos de

um mesmo grupo em segmentos. Então, segmentos adjacentes são fundidos, reduzindo a redundância entre eles. No final, um subconjunto de quadros dos segmentos mais longos é selecionado para compor o resumo de vídeo.

Putpuek *et al.* [134] desenvolveram uma abordagem similar à técnica anterior, que segmenta o vídeo de entrada em tomadas usando histogramas de cor. Para isso, o algoritmo de Pan *et al.* [126] é empregado para determinar transições entre tomadas. A seguir, os autores aplicam o método GreedyRSC [83] para fundir tomadas adjacentes e, consequentemente, reduzir a redundância. Por fim, apenas a porção com o maior movimento de cada grupo é incluída no resumo final.

Bredin *et al.* [36] introduziram uma técnica de sumarização com base na análise de componentes principais (PCA). A princípio, a segmentação temporal do vídeo de entrada é realizada a partir de uma limiarização adaptativa da distância entre histogramas de cor de quadros consecutivos. Em seguida, a PCA é aplicada em uma matriz formada por todos os histogramas, obtendo outra matriz, na qual cada coluna representa um quadro em um espaço de características mais refinado. Depois disso, um mapa de conteúdo é criado usando as componentes mais representativas desse espaço. Assim, cada tomada é representada por uma espécie de marca em tal mapa, revelando a importância do seu conteúdo. No final, as tomadas mais relevantes são selecionadas para compor o resumo.

Chasanis *et al.* [42] propuseram um método semelhante a essa última abordagem, o qual utiliza agrupamento espectral e alinhamento de sequências. Inicialmente, o vídeo de entrada é segmentado em tomadas comparando-se histogramas de cor de quadros consecutivos. Em seguida, um algoritmo de agrupamento espectral melhorado é utilizado para selecionar o quadro mais representativo de cada tomada. Então, tomadas similares são agrupadas usando um algoritmo de alinhamento de sequências. Cada grupo é representado pela tomada com maior duração. Por fim, o resumo de vídeo é produzido tomando-se quadros adjacentes ao quadro representativo de cada tomada.

Uma vez que os dados de vídeo são geralmente disponibilizados na forma comprimida, é desejável processar diretamente o vídeo comprimido, sem decodificação. Isso permite economizar a elevada carga computacional da decodificação plena do fluxo de vídeo.

Alguns métodos para a sumarização de vídeo que manipulam diretamente o vídeo comprimido têm sido propostos para domínios específicos, tais como esportes, música e notícias [82, 155, 172]. Incidir sobre um domínio particular ajuda a reduzir os níveis de ambiguidade quando se analisa o conteúdo de um vídeo por meio da aplicação do conhecimento prévio do domínio durante o processo de análise [118].

VISON foi projetado para preencher as lacunas acima. A novidade do VISON é que ele permite o uso *online* e personalização do usuário. Diferente de todas as técnicas anteriores, esse método atua diretamente no domínio comprimido e sumariza vídeos genéricos e, portanto, não emprega nenhuma informação específica além do próprio conteúdo do vídeo.

7.2 Abordagem Proposta

O método proposto foi projetado com os seguintes objetivos: (1) gestão de vídeos genéricos, (2) personalização de usuário experiente (a qualidade dos resumos e o tempo máximo de espera) e (3) produção de resumos em um tempo razoável e com uma qualidade aceitável, de modo a permitir o uso *online*. Ele consiste em três etapas principais: (1) extração de características, (2) seleção de conteúdo e (3) filtragem de ruídos[1].

Um fluxograma do VISON é apresentado na Figura 7.1. Para cada quadro do vídeo de entrada, características visuais são extraídas do fluxo de vídeo para descrever o seu conteúdo visual. Em seguida, um algoritmo simples e rápido é usado para detectar grupos de quadros com um conteúdo semelhante e selecionar um quadro representativo de cada grupo. Por fim, os quadros selecionados são filtrados a fim de evitar possíveis quadros redundantes ou sem sentido no resumo de vídeo. Nas subseções a seguir, cada uma dessas etapas é explicada mais detalhadamente.

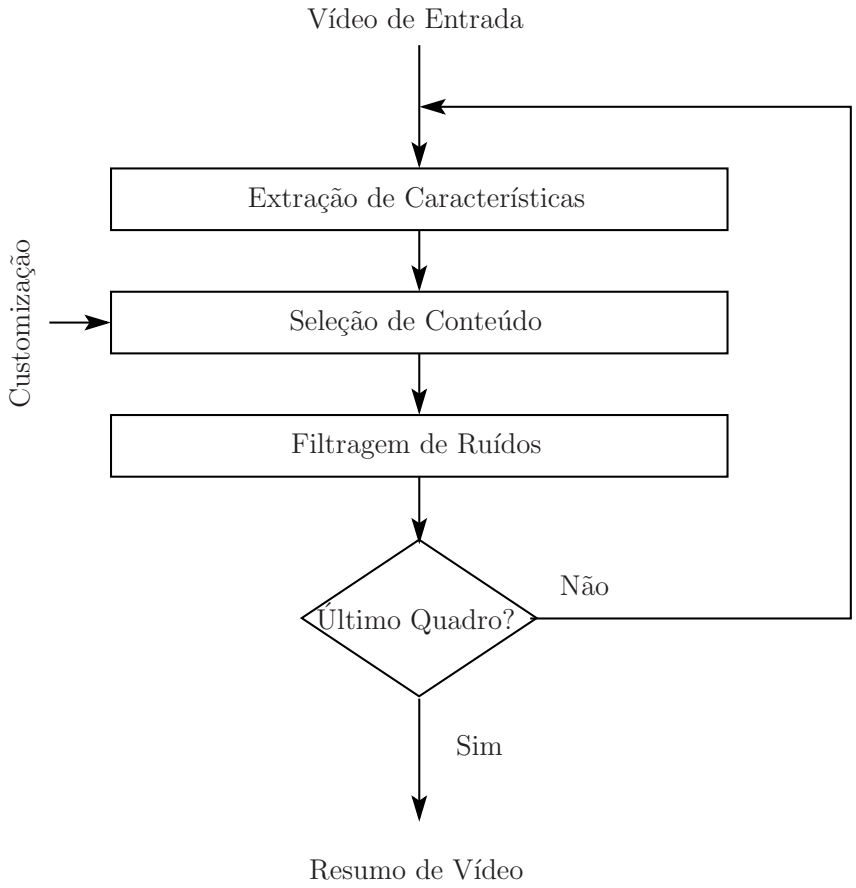


Figura 7.1: Fluxograma do VISON.

[1]No escopo desta tese, o conceito de filtragem de ruídos, diferente da definição adotada em processamento de sinais, refere-se ao processo de eliminar possíveis quadros redundantes ou sem sentido.

7.2.1 Extração de Características

O primeiro passo na sumarização de vídeos é dividir o fluxo de vídeo em um conjunto de unidades significativas e gerenciáveis. Muitos padrões de codificação (por exemplo, MPEG-1/2/4) dividem o fluxo de vídeo em unidades básicas, denominadas GOPs. Em geral, o quadro I contém informação suficiente para caracterizar todo o conteúdo do GOP.

A partir dos quadros I, é possível obter versões menores das imagens originais, denominadas imagens DC (Seção 2.7). Apesar de seu tamanho reduzido, elas estão em formato cru (do inglês, *raw*) e, portanto, necessitam de uma quantidade substancial de espaço. A fim de economizar a quantidade de dados a ser armazenada, cada imagem DC é convertida em um vetor de características de 256 dimensões calculando-se um histograma de cor. Essa técnica é computacionalmente trivial e também robusta a pequenas mudanças de posição da câmera.

A dúvida principal em relação a métodos baseados em histogramas de cor diz respeito à escolha de um espaço de cor adequado. No caso em questão, é importante que o modelo de cor reflita a percepção humana das cores. Essa tarefa é simplificada pelo uso de espaços de cor orientados ao usuário (Seção 2.2), que exploram as características utilizadas por seres humanos para distinguir uma cor da outra [154].

Por essa razão, histogramas de cor são obtidos do espaço de cor HSV, que tem se mostrado mais resistente a ruídos [128]. Eles são extraídos da seguinte maneira: o espaço de cor HSV é dividido em 256 subespaços (cores), usando 32 intervalos do H, 4 intervalos do S e 2 intervalos do V. O valor de cada dimensão do vetor de características é a densidade de cada cor na imagem DC inteira.

7.2.2 Seleção de Conteúdo

O próximo passo da abordagem proposta faz uso dos histogramas de cor extraídos do vídeo comprimido para detectar grupos de quadros com um conteúdo semelhante e selecionar um quadro representativo de cada grupo, de modo a produzir o resumo.

A eficácia de agrupar quadros similares depende da escolha adequada da medida de similaridade utilizada para compará-los. Existem várias opções para mensurar as diferenças entre esses quadros, como intersecção de histogramas, distâncias Minkowski, entre outras. Na Seção 7.4.1, é realizado um estudo que mostra como diferentes medidas de similaridade podem afetar o desempenho da técnica apresentada.

De acordo com tal análise, adotou-se a métrica ZNCC (do inglês, *Zero-mean Normalized Cross Correlation*) [111] como a função de distância utilizada pelo método proposto. Ela foi escolhida devido a sua invariância a transformações fotométricas afins, por exemplo, mudanças lineares e uniformes em brilho ou contraste [112]. Por essa razão, tal medida é amplamente utilizada na correspondência entre modelos [152], análise de movimento [157], visão estéreo [156] e inspeção industrial [164].

Seja $\mathcal{H}^i$, o valor da i-ésima dimensão do histograma de cor $\mathcal{H}$, e $\bar{\mathcal{H}}$, a média dos valores de todas as entradas de $\mathcal{H}$. A similaridade entre quadros de vídeo é dada por

$$\mathrm{ZNCC}(\mathcal{H}_{t_1}, \mathcal{H}_{t_2}) = \frac{\sum_i ((\mathcal{H}^i_{t_1} - \bar{\mathcal{H}}_{t_1}) \times (\mathcal{H}^i_{t_2} - \bar{\mathcal{H}}_{t_2}))}{\sqrt{\sum_i (\mathcal{H}^i_{t_1} - \bar{\mathcal{H}}_{t_1})^2 \times \sum_i (\mathcal{H}^i_{t_2} - \bar{\mathcal{H}}_{t_2})^2}}, \tag{7.1}$$

em que $\mathcal{H}_{t_1}$ e $\mathcal{H}_{t_2}$ são os histogramas de cor extraídos dos quadros I tomados nos instantes de tempo t_1 e t_2, respectivamente. Essa função retorna um valor real que varia de 0, para situações em que os histogramas são totalmente diferentes; até 1, quando eles são idênticos.

Para detectar grupos de quadros, calcula-se a dissimilaridade entre pares de quadros consecutivos. A Figura 7.2 mostra um exemplo de como esses valores estão distribuídos ao longo do tempo. Pode-se observar que há instantes de tempo nos quais o valor de dissimilaridade varia consideravelmente (correspondendo aos picos), enquanto há longos períodos nos quais a variação é pequena (correspondendo às regiões bastante densas). Normalmente, os picos estão associados a movimentos bruscos na cena ou a fronteiras entre duas tomadas, enquanto regiões densas caracterizam uma sequência de quadros similares entre si. Assim, quadros entre dois picos podem ser considerados como um grupo de quadros com um mesmo conteúdo.

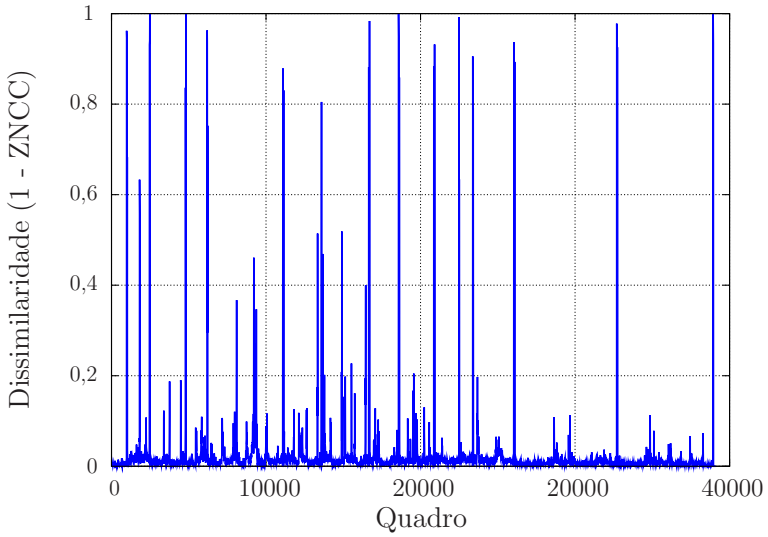


Figura 7.2: Dissimilaridades entre pares de quadros do vídeo *MRS150072* do conjunto de dados do TRECVID 2007.

Neste trabalho, considera-se como um pico somente os instantes nos quais

$$\frac{\mathrm{ZNCC}(\mathcal{H}_t, \mathcal{H}_{t-1})}{\mathrm{ZNCC}(\mathcal{H}_{t-1}, \mathcal{H}_{t-2})} + \frac{\mathrm{ZNCC}(\mathcal{H}_t, \mathcal{H}_{t-1})}{\mathrm{ZNCC}(\mathcal{H}_{t+1}, \mathcal{H}_t)} > 2 \times (1 - \epsilon), \tag{7.2}$$

em que ϵ é um limiar para a dissimilaridade entre quadros semelhantes. A partir de testes experimentais, verificou-se que valores entre 0,05 e 0,15 são, frequentemente, boas escolhas para esse limiar, conforme demonstrado na Seção 7.4.3.

Se a duração de uma sequência de quadros similares estiver abaixo de um mínimo, todos esses quadros são descartados. Caso contrário, um fragmento dessa sequência é selecionado para representar esse grupo de quadros. Em outras palavras, um grupo de quadros é considerado para o resumo somente se seu tamanho for maior do que um limiar λ. Empiricamente, os valores entre 0,5 e 2% da duração total do vídeo têm se mostrado boas escolhas, conforme demonstrado na Seção 7.4.3.

Por fim, um fragmento do grupo de quadros é extraído por amostragem temporal a uma taxa definida pelo usuário, a qual pode ser usada para controlar a duração (ou comprimento) dos resumos. Dependendo do tipo de resumo, apenas quadros I (estático) ou GOPs inteiros (dinâmico) são selecionados. Neste trabalho, apenas 15 por cento de um grupo de quadros é incluído no resumo de vídeo. A Figura 7.3 apresenta os quadros selecionados por essa abordagem para o vídeo *MRS150072* do conjunto de dados do TRECVID 2007, tomando-se o quadro I do meio de cada segmento.

Figura 7.3: Quadros selecionados pela abordagem proposta para o vídeo *MRS150072* do conjunto de dados do TRECVID 2007.

7.2.3 Filtragem de Ruídos

O objetivo desta etapa é evitar possíveis quadros redundantes ou sem sentido no resumo de vídeo. Na verdade, o algoritmo de seleção pode incluir um quadro totalmente preto (ou de outra cor), devido a efeitos de edição de vídeo (por exemplo, aparecer, desaparecer, dissolver e limpar) ou ao uso de flashes (muito comum em vídeos de esportes ou notícias, por exemplo). Essa investigação consome um tempo quase insignificante, uma vez que o número de quadros selecionados é geralmente muito pequeno.

Basicamente, existem dois tipos de ruído que podem afetar a qualidade de um resumo de vídeo: quadros inúteis e segmentos redundantes. Os primeiros geralmente aparecem no início e/ou no final de cada tomada, como padrões de teste, placas de sincronização (do inglês, *clapboards*), quadros monocromáticos, entre outros. Esses últimos são trechos repetitivos originados, por exemplo, a partir de tomadas diferentes de uma mesma cena.

Quando um novo quadro é selecionado para compor o resumo de vídeo, ele é analisado e, caso seja considerado indesejável, ele é descartado. Isso pode ser realizado da seguinte forma. Inicialmente, a imagem DC extraída desse quadro é uniformemente quantizada em 256 cores no espaço de cor HSV usando 32 intervalos do H, 4 intervalos do S e 2 intervalos do V. É importante lembrar que tal processo é realizado na primeira etapa (Seção 7.2.1).

Em seguida, verifica-se se o quadro analisado é inútil ou não. Para isso, dois histogramas são obtidos a partir de sua imagem quantizada: um para a distribuição das cores e outro para as orientações do gradiente. O histograma de orientação tem 36 células, cobrindo a faixa de 360 graus de orientações possíveis. A Figura 7.4 ilustra o comportamento desses histogramas para diferentes tipos de quadros de vídeo. Como se pode observar, os quadros inúteis têm uma distribuição homogênea, o que leva a uma alta variação entre as células dos histogramas. Assim, um quadro de entrada é descartado se um de seus histogramas tiver uma variância normalizada maior que 0,5. Caso contrário, ele é comparado com todos os quadros do resumo de vídeo, conforme explicado seguir. A escolha desse limiar é detalhada na Seção 7.4.2.

A fim de lidar com sequências repetitivas, o novo quadro é comparado com todos os quadros anteriores do resumo de vídeo. O algoritmo para a comparação desses quadros baseia-se no casamento pontual entre suas imagens quantizadas. Dois pontos correspondentes são considerados dissimilares se seus valores de intensidade forem diferentes ou se, pelo menos, um de seus 4 vizinhos correspondentes (superior, inferior, esquerda e direita) tiver uma cor quantizada diferente. Caso contrário, esses pontos são considerados similares. Assim, a semelhança entre esses quadros é mensurada a partir da relação entre o número de pontos similares e o número total de pontos. Dois quadros são considerados iguais se o valor da semelhança entre eles for maior que 25%, no caso de resumos estáticos; ou 5%, em resumos dinâmicos.

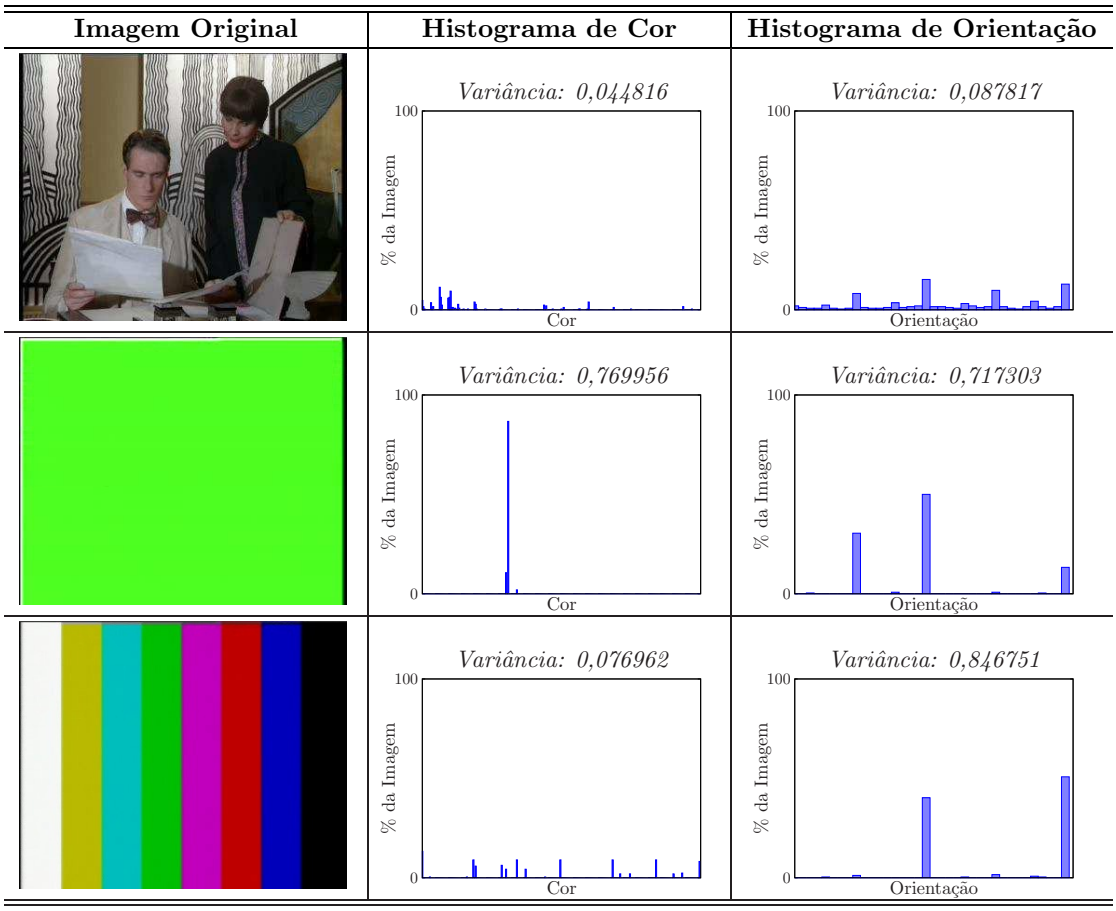


Figura 7.4: O comportamento dos histogramas de cor (segunda coluna) e de orientação (terceira coluna) para quadros normais (primeira linha), monocromáticos (segunda linha) e barras coloridas (terceira linha).

Por fim, se a razão entre o número de quadros iguais e o número total de quadros de um fragmento selecionado for abaixo de um mínimo, então todos os quadros desse fragmento são incluídos no resumo de vídeo. Caso contrário, assume-se que todos eles pertencem a um segmento redundante e, portanto, podem ser descartados. Em outras palavras, um fragmento extraído de um grupo de quadros é considerado para o resumo somente se a sua percentagem de quadros duplicados em relação aos quadros já selecionados for maior do que um limiar κ. Em resumos estáticos, o valor desse limiar deve ser fixado em zero, implicando ausência de quadros iguais. Empiricamente, valores inferiores a 50% do número total de quadros do fragmento de entrada mostraram-se boas escolhas para a produção de resumos dinâmicos. A Figura 7.5 apresenta os quadros mantidos pela abordagem proposta após a filtragem de ruídos do vídeo *MRS150072* do conjunto de dados do TRECVID 2007.

Figura 7.5: Quadros mantidos pela abordagem proposta para o vídeo *MRS150072* do conjunto de dados do TRECVID 2007.

7.2.4 Personalização do Usuário

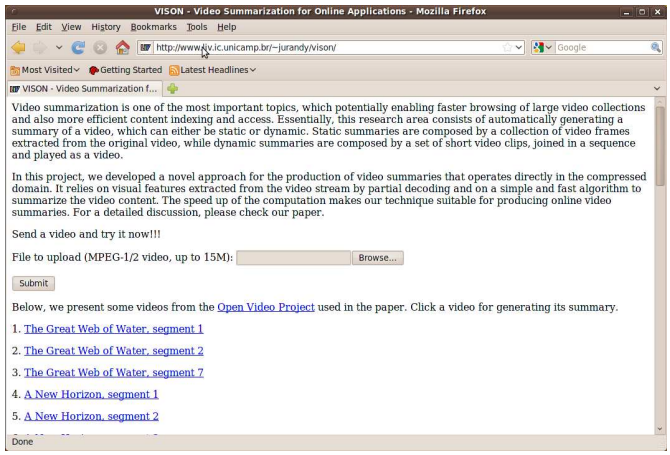
VISON permite que os usuários ajustem os limiares ϵ, λ e κ. Esses valores, juntos, controlam a qualidade dos resumos de vídeo. Além disso, os usuários podem especificar o tempo que estão dispostos a esperar para obter o resumo. Esse recurso é oferecido porque o tempo de produção do resumo de um vídeo depende da sua duração: quanto maior for a duração, maior será o tempo de produção.

Vários estudos observacionais sobre o comportamento dos usuários têm mostrado que o tempo de espera é crítico [91]. Por exemplo, até 5 segundos para obter uma página *web* completa é considerado um bom tempo de espera, enquanto mais de 10 segundos é ruim. Todavia, se a página *web* for carregada de forma incremental, o tempo de espera de até 39 segundos é considerado aceitável [34].

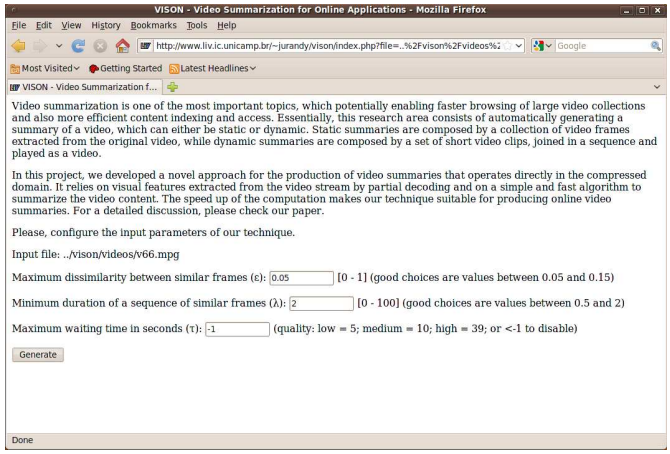
Usando um tempo máximo de espera τ, é possível explorar a redundância espaço-temporal de uma sequência de vídeo a fim de reduzir o número total de quadros processados. Para isso, sincroniza-se o quadro a ser processado com o tempo de processamento. Após o tratamento de um quadro, verifica-se o número de quadros já processados contra o relógio do sistema para ver quantos quadros devem ser descartados. Essa abordagem funciona de forma similar à técnica de amostragem temporal, que é amplamente utilizada pela comunidade de processamento de imagem e vídeo. Apesar de satisfazer a restrição de tempo solicitado, há uma questão que deve ser enfatizada: quanto menor o tempo de espera, pior a qualidade do resumo, uma vez que muitos quadros precisam ser descartados.

A Figura 7.6 mostra a interface do VISON[2]. Primeiro, um usuário pode carregar uma sequência de vídeo em formato MPEG-1/2 ou selecionar um vídeo da coleção disponível (Figura 7.6(a)). Em seguida, ele pode ajustar os parâmetros a serem utilizados na produção do resumo e também configurar o tempo máximo que está disposto a esperar para obtê-lo (Figura 7.6(b)). No final, VISON aplica esses parâmetros para analisar o vídeo selecionado e apresenta o resumo resultante (Figura 7.6(c)).

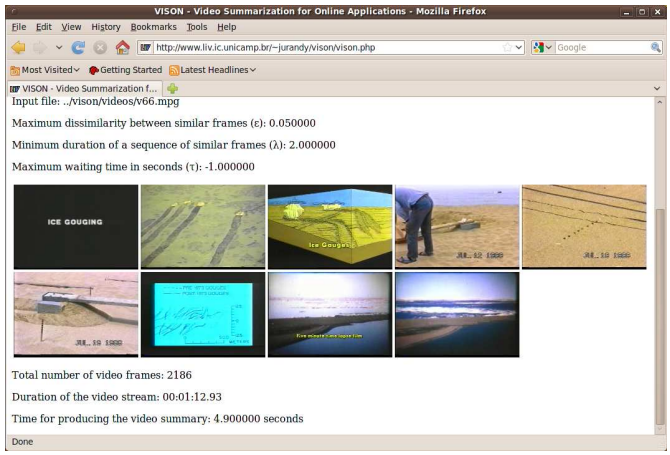
[2] `http://www.liv.ic.unicamp.br/~jurandy/vison/` (último acesso em 14 de outubro de 2011).



(a) Tela inicial.



(b) Customização por usuários.



(c) Apresentação dos resumos.

Figura 7.6: A interface do VISON, na qual resumos de vídeo são produzidos *online*. Os usuários podem controlar a qualidade dos resumos e especificar o tempo máximo de espera.

7.3 Protocolo Experimental

Os experimentos foram realizados em uma máquina equipada com um processador Intel Core 2 Quad Q6600 (4 núcleos de 2,4 GHz) e 2 GBytes de memória DDR3, executando o sistema operacional Ubuntu (núcleo linux 2.6.31) e o sistema de arquivos ext3.

Ao contrário de outras áreas de pesquisa, avaliar um resumo de vídeo não é uma tarefa simples devido à inexistência de um gabarito. A ausência de um modelo de avaliação consolidado é uma deficiência séria para a pesquisa em sumarização de vídeos. Atualmente, cada trabalho da literatura tem sua própria metodologia de avaliação, muitas vezes introduzidas sem qualquer análise de eficiência [118, 163].

Além disso, diferentes tipos de resumo de vídeo apresentam características distintas, como mencionado na Seção 7.1. Isso requer que diferentes estratégias de avaliação sejam empregadas. Nas subseções a seguir, o protocolo experimental adotado em cada um deles é explicado com mais detalhes.

7.3.1 Avaliação de Resumos Estáticos

A fim de manter uma referência comum, a qualidade de resumos estáticos foi avaliada por meio da metodologia subjetiva conhecida como CRU (Comparação com Resumos de Usuários) [21]. Ela incorpora o julgamento do usuário na avaliação da qualidade de um resumo. Em tal método, resumos de vídeo são criados manualmente por vários sujeitos e comparados com os resumos obtidos por diferentes métodos. Os objetivos dessa abordagem são [21]: (1) reduzir a subjetividade da tarefa de avaliação, (2) determinar a qualidade dos resumos de vídeo e (3) permitir a comparação entre diferentes técnicas.

A Figura 7.7 ilustra esse método de avaliação. Inicialmente, os sujeitos são convidados a assistir ao vídeo inteiro. Depois disso, eles são orientados a selecionar um subconjunto de quadros que, na opinião particular, seja capaz de resumir o respectivo conteúdo. Cada indivíduo é livre para escolher qualquer número de quadros em seu resumo. Por fim, tais resumos são comparados com aqueles fornecidos por vários algoritmos.

Para comparar quadros de diferentes resumos, foi aplicado o método descrito na Seção 7.2.3. Se dois quadros forem considerados iguais, eles são removidos da próxima iteração do procedimento de comparação. Assim, a semelhança entre o resumo do usuário e o resumo automático é dada pelo número de quadros casados.

As medidas padrões de precisão e revocação (Seção 2.11) podem então ser utilizadas para avaliar o resumo automático. Precisão é a razão entre o número de quadros iguais e o número total de quadros no resumo automático. Revocação é a razão entre o número de quadros iguais e o número total de quadros no resumo do usuário. Entretanto, há uma relação de compromisso entre precisão e revocação, uma vez que pode-se aumentar a precisão ao custo de reduzir a revocação e vice-versa. Por isso, foi escolhida a medida-F para avaliar a qualidade de resumos automáticos.

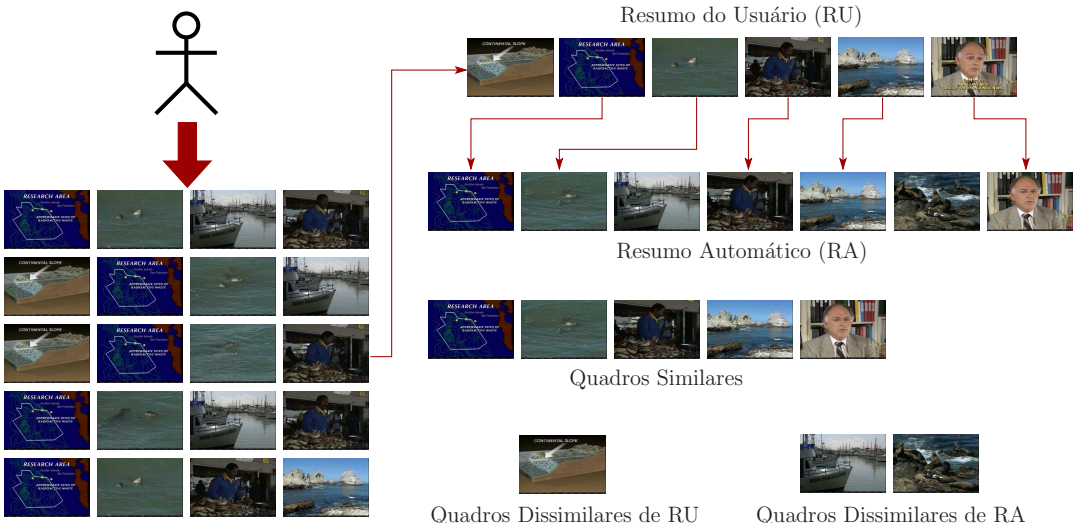


Figura 7.7: O método de avaliação CRU (adaptada de Avila [21]).

A fim de assegurar significância estatística, intervalos de confiança de 98% foram utilizados para analisar as diferenças entre valores da medida-F obtidos em resumos automáticos produzidos por diferentes abordagens. Esse nível de confiança é o mesmo adotado por Avila [21].

Experimentos foram conduzidos em uma amostra de 50 vídeos selecionados aleatoriamente do Open Video[3]. Todos os vídeos estão no formato MPEG-1 (com resolução de 352×240 pontos e 29,97 quadros por segundo), incluindo cor e som. Eles estão distribuídos entre vários gêneros (por exemplo, documentários, educativos e palestras) e sua duração varia de 1 a 4 minutos. O fato de não escolher vídeos longos está ligado à duração da anotação por um sujeito. Esses vídeos são os mesmos utilizados por Mundur *et al.* [119], Furini *et al.* [63] e Avila *et al.* [22].

Os resumos de usuário foram criados por 50 indivíduos, cada um lidando com 5 vídeos, o que significa que cada vídeo tem 5 resumos criados por 5 sujeitos diferentes. Em outras palavras, 250 resumos de vídeo foram criados à mão. Indivíduos distintos podem criar resumos diferentes em termos de aparência e comprimento. Para medir tal efeito, os intervalos de confiança para as semelhanças (medida-F) entre pares de resumos foram computados para comparar cada par de usuários. De acordo com esse experimento, o acordo (média da medida-F) entre os resumos de usuários diferentes é igual a $0,765 \pm 0,050$ (nível de confiança superior a 99,9%).

A eficácia da técnica proposta também foi avaliada sobre 40 vídeos coletados do YouTube[4]. Esses vídeos estão distribuídos entre vários gêneros (e.g., esportes, notícias, pro-

[3]Todos os vídeos e os resumos dos usuários da coleção do Open Video estão disponíveis em `http://sites.google.com/site/vsummsite/` (último acesso em 14 de outubro de 2011).

[4]Todos os vídeos e os resumos dos usuários da coleção do YouTube estão disponíveis em `http://sites.google.com/site/vsummsite/` (último acesso em 14 de outubro de 2011).

gramas de TV, comerciais e vídeos caseiros) e sua duração varia de 1 a 10 minutos.

Seguindo o protocolo anterior, 50 indivíduos foram convidados a criar manualmente resumos para os vídeos dessa coleção. Cada vídeo tem 5 resumos criados por 5 sujeitos diferentes. O acordo (média de medida-F) entre os resumos de usuários diferentes é igual a $0,651 \pm 0,078$ (nível de confiança superior a 99,9%).

7.3.2 Avaliação de Resumos Dinâmicos

Para avaliar a abordagem proposta mediante a produção de resumos dinâmicos, adotou-se a metodologia experimental utilizada pela equipe do TRECVID no âmbito da tarefa de sumarização de vídeos da BBC [124, 125]. Ela incorpora o julgamento de um humano na avaliação da qualidade de um resumo de vídeo. Em tal método, o gabarito é uma lista de segmentos importantes que ocorrem no vídeo original. Cada segmento é identificado por objetos ou eventos distintos, que são caracterizados pelo ângulo da câmera, distância ou alguma outra informação capaz de dar autenticidade à sua descrição [124, 125].

Cada resumo de vídeo foi julgado por um único ser humano com respeito a quanto do gabarito foi incluído nele (IN) e a quanto de repetição foi contida no mesmo (RE). Medidas objetivas adicionais também foram avaliadas: a duração do resumo em relação à duração do vídeo original (DU) e a fração de quadros inúteis retidos no resumo (JU). Para cada medida, intervalos de confiança de 95% foram utilizados para analisar as diferenças entre resumos de vídeo produzidos por diferentes abordagens. Esse nível de confiança é o mesmo adotado por Over *et al.* [124, 125].

Experimentos foram conduzidos em cerca de 5 horas de vídeo (409.630 quadros) de um conjunto de dados do TRECVID 2007 (BBC Rushes)[5]. Todos os vídeos estão no formato MPEG-1 (com resolução de 352×288 pontos e 25 quadros por segundo), incluindo cor e som. Eles consistem de tomadas de vídeo não editadas, que foram filmadas, em especial, para cinco séries de programas de drama da BBC, e sua duração varia de 6 a 34 minutos.

Esses vídeos são a matéria bruta de uma câmera, gravados durante a produção de um filme ou documentário. Eles estão organizados em cenas, que representam uma determinada parte da ação. Devido a mudanças na apresentação de uma ação ou erros inesperados na filmagem de uma cena, várias tomadas de uma mesma cena são gravadas na sequência de vídeo. Além disso, esses vídeos também contêm dados auxiliares, tais como padrões de teste, para calibrar as cores da câmera, ou trechos com placas de sincronização, que identificam o número da tomada e da cena na gravação e facilitam o alinhamento da banda sonora com a visual [57].

[5]Todos os vídeos e os gabaritos do conjunto de dados do TRECVID 2007 estão disponíveis em `http://satoh-lab.ex.nii.ac.jp/users/ledduy/Demo-BBCRushes/` (último acesso em 14 de outubro de 2011).

7.4 Resultados Experimentais

Esta seção reporta resultados experimentais sobre o desempenho da técnica apresentada. Todos os experimentos foram cuidadosamente projetados a fim de garantir significância estatística. Eles têm o intuito de responder às seguintes questões:

- Qual é a função de distância mais adequada na produção de resumos de vídeo?
- Qual é a melhor configuração para os limiares usados na filtragem de ruídos?
- Como os parâmetros de controle do método podem afetar a qualidade dos resumos?
- Como é o desempenho da técnica proposta comparado às soluções vigentes?

7.4.1 Efeitos da Função de Distância

Em geral, a eficácia de agrupar quadros semelhantes é melhor se as diferenças intragrupo forem pequenas e as diferenças intergrupo forem grandes. Esse efeito está relacionado com a medida de similaridade usada para comparar quadros de vídeo e impacta diretamente na qualidade dos resumos. No entanto, a escolha da métrica utilizada para gerar um resumo de vídeo é normalmente dependente dos dados.

Para identificar a função de distância mais adequada para medir as diferenças entre quadros de vídeo, quatro opções foram analisadas (Seção 2.9): intersecção de histogramas (HI), distância de Manhattan (L_1), distância Euclidiana (L_2) e correlação cruzada normalizada de média zero (ZNCC). O experimento consistiu em testar exaustivamente combinações entre medidas de similaridade e diferentes níveis para os parâmetros ϵ e λ.

A Tabela 7.1 apresenta a média da medida-F atingida por diferentes funções de distância na produção de resumos para os vídeos do Open Video e do YouTube. Essa tabela reporta e compara apenas os resultados da melhor combinação de parâmetros (ϵ e λ) para cada medida de similaridade.

Tabela 7.1: A média da medida-F obtida pela melhor combinação de parâmetros (ϵ e λ) usando diferentes funções de distância.

Coleção	*Função*	ϵ	λ	*medida-F*
Open Video	HI	0,15	2,5	0,746
	L_1	0,20	1,5	0,728
	L_2	0,05	1,0	0,618
	ZNCC	**0,05**	**2,0**	**0,755**
YouTube	HI	0,25	2,0	0,449
	L_1	0,20	2,0	0,436
	L_2	0,05	0,5	0,362
	ZNCC	**0,15**	**1,0**	**0,492**

Esses resultados indicam que o desempenho da ZNCC é melhor que das demais métricas em ambas as coleções. A fim de verificar a significância estatística desses resultados, intervalos de confiança para as diferenças entre pares de médias foram computados para comparar cada par de métodos. Se o intervalo de confiança inclui zero, a diferença não é significativa nesse nível de confiança. Por outro lado, se o intervalo de confiança não incluir zero, então o sinal da média das diferenças indica qual alternativa é melhor [90].

A Tabela 7.2 apresenta os intervalos de confiança (a um nível de confiança de 98%) para a diferença entre a média da medida-F alcançada por cada função de distância nas coleções de vídeo do Open Video e do YouTube.

Tabela 7.2: Diferença entre as médias da medida-F de diferentes funções de distância a um nível de confiança de 98%.

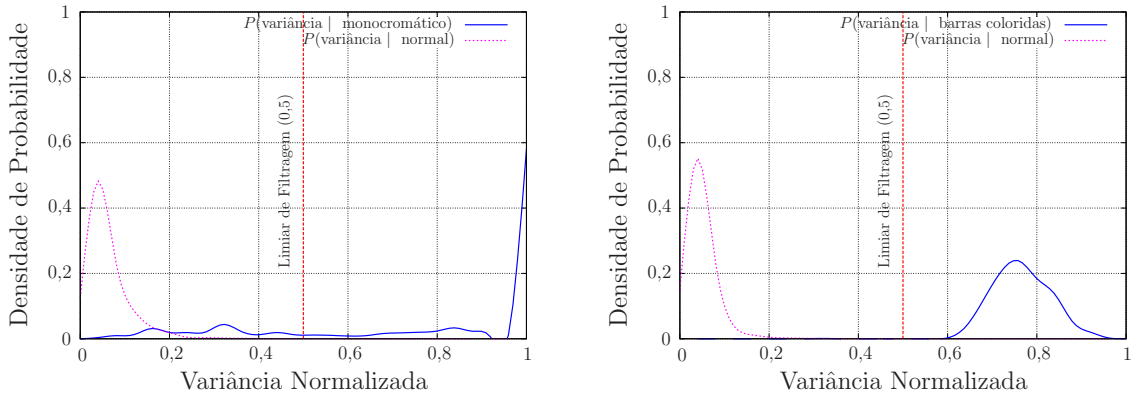
Coleção	*Função*	*Média*	*Intervalo de Confiança (98%)*	
			mínimo	*máximo*
Open Video	ZNCC - HI	0,009	-0,013	0,034
	ZNCC - L_1	0,027	0,003	0,051
	ZNCC - L_2	0,137	0,095	0,178
YouTube	ZNCC - HI	0,043	0,009	0,077
	ZNCC - L_1	0,057	0,023	0,091
	ZNCC - L_2	0,130	0,085	0,174

Para os vídeos do Open Video, ZNCC e HI exibem desempenho similar, ao passo que o desempenho da ZNCC é melhor que da HI na coleção do YouTube. Em contraste, ZNCC consistentemente supera L_1 e L_2 em ambos os conjuntos de dados. Isso acontece porque a ZNCC é invariante a transformações fotométricas afins (por exemplo, mudanças lineares e uniformes em brilho e contraste) [112]. Por essas razões, a ZNCC foi adotada como a função de distância a ser utilizada na produção de resumos de vídeo.

7.4.2 Escolha dos Limiares de Filtragem

Esta seção mostra como foram selecionados os valores dos limiares utilizados para filtrar quadros inúteis. Para isso, todos os quadros dos vídeos do conjunto de dados do TRECVID 2007 foram manualmente anotados como normal, monocromático ou barras coloridas. Em seguida, calculou-se a variância normalizada dos histogramas de cor e de orientação para cada quadro, conforme descrito na Seção 7.2.3.

A Figura 7.8 apresenta a função densidade de probabilidade (PDF) para a distribuição da variância normalizada de cada histograma. Pode-se reparar que as curvas PDF para os diferentes tipos de quadros de vídeo são totalmente separáveis, evidenciando a alta capacidade discriminativa da estratégia proposta. Por essa razão, um valor igual a 0,5 foi utilizado para limiarização de quadros inúteis, em ambas as características.



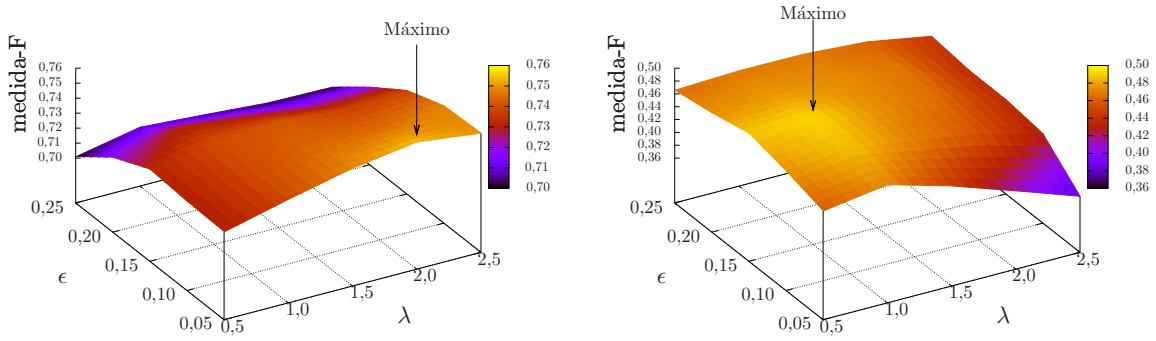
(a) Histograma de Cor (b) Histograma de Orientação

Figura 7.8: A função densidade de probabilidade (PDF) para a distribuição da variância normalizada de histogramas extraídos de diferentes tipos de quadros.

7.4.3 Ajuste dos Parâmetros do Método

Nesta seção, estuda-se como os parâmetros de controle do VISON podem afetar a qualidade dos resumos de vídeo. Para isso, foi avaliada a interação entre seus dois parâmetros mais importantes: ϵ e λ. O experimento consistiu em testar exaustivamente todas as combinações de diferentes níveis para cada parâmetro.

Os resultados são apresentados na Figura 7.9, na qual os eixos x e y representam a variação dos parâmetros ϵ e λ, respectivamente. Os valores do eixo z indicam a média da medida-F obtida por cada combinação desses parâmetros. As setas apontam a melhor combinação de parâmetros para cada conjunto de dados, ou seja, os valores para ϵ e λ que maximizam a média da medida-F. Eles foram: ϵ ajustado em 0,05 e λ igual a 2% da duração do vídeo, para os vídeos do Open Video; e ϵ igual a 0,15 e λ ajustado em 1%, para a coleção do YouTube.

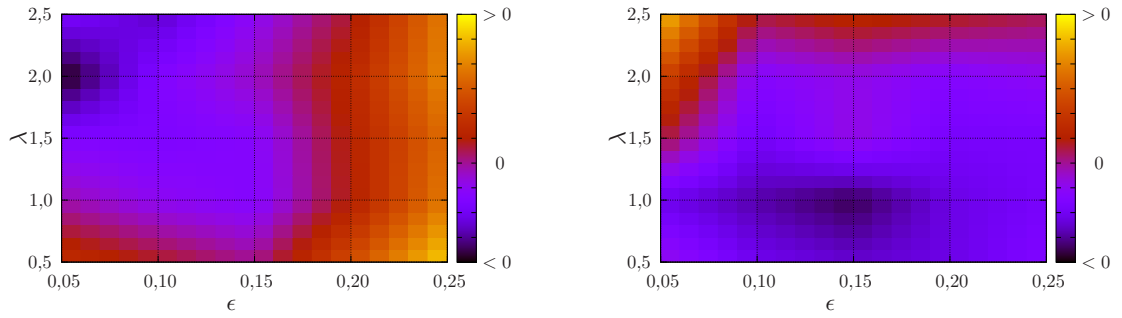


(a) Open Video (b) YouTube

Figura 7.9: A média da medida-F obtida a partir da combinação de diferentes níveis para os parâmetros ϵ e λ.

Esses resultados demonstram a forte interação entre os parâmetros ϵ e λ. A fim de verificar a sensibilidade do VISON em relação a esses parâmetros, foram calculados intervalos de confiança (a um nível de confiança de 98%) para as diferenças entre a média da medida-F obtida por cada combinação de parâmetros com respeito à melhor delas.

Como pode ser visto nos gráficos da Figura 7.10, resumos produzidos para os vídeos do Open Video são mais sensíveis a variações no parâmetro ϵ. Em contraste, mudanças no parâmetro λ são as maiores responsáveis pelas variações observadas na coleção do YouTube.



(a) Open Video (b) YouTube

Figura 7.10: Diferença entre as médias da medida-F de diferentes combinações de parâmetros a um nível de confiança de 98%.

A principal razão para esses resultados é a sensibilidade dos histogramas de cor a movimentos de câmera (por exemplo, panorâmica horizontal, panorâmica vertical ou zoom), o que é típico nos vídeos de documentários do Open Video. Por outro lado, a maioria dos vídeos do YouTube é composta por várias tomadas curtas, o que é muito comum em vídeos de esportes ou de notícias.

Embora os valores adequados para os parâmetros de controle do VISON sejam dependentes dos dados, o parâmetro ϵ ajustado em um valor entre 0,05 e 0,15 e os valores de λ entre 0,5 e 2% da duração do vídeo têm se mostrado boas escolhas,

7.4.4 Desempenho em Resumos Estáticos

A eficácia do método proposto mediante a produção de resumos estáticos foi primeiro avaliada na coleção de vídeos do Open Video[6]. Conforme detalhado na Seção 7.4.3, os parâmetros utilizados pelo VISON para produzir resumos de vídeo nessa coleção foram: ϵ ajustado em 0,05 e λ igual a 2% da duração do vídeo. Para efeito de comparação,

[6]Os resumos produzidos pelo VISON para os 50 vídeos do Open Video podem ser vistos em `http://www.liv.ic.unicamp.br/~jurandy/summaries/` (último acesso em 14 de outubro de 2011).

foram utilizados os resultados reportados por três técnicas[7]: DT [119], STIMO [63] e VSUMM [22]. Além disso, os resumos produzidos pelo VISON foram comparados com aqueles exibidos no Open Video (OV), os quais foram produzidos a partir do algoritmo de DeMenthon *et al.* [55] e adicionados de intervenção manual para refinar os resultados.

A Tabela 7.3 apresenta a média da medida-F obtida por diferentes abordagens em várias categorias de vídeo. Os intervalos de confiança (a um nível de confiança de 98%) para a diferença entre a média da medida-F atingida por cada método são apresentados na Tabela 7.4

Tabela 7.3: A média da medida-F obtida por diferentes abordagens em várias categorias de vídeo.

Categoria	Vídeos	OV	DT	STIMO	VSUMM	VISON
Documentário	44	0,626	0,563	0,627	0,732	**0,765**
Educativo	2	0,707	0,584	0,587	**0,768**	0,729
Palestra	4	**0,714**	0,535	0,655	0,639	0,660
Média Ponderada	50	0,636	0,562	0,627	0,726	**0,755**

Tabela 7.4: Diferença entre as médias da medida-F atingidas por diferentes abordagens a um nível de confiança de 98%.

Método	*Média*	*Intervalo de Confiança (98%)*	
		mínimo	*máximo*
VISON - OV	0,119	0,074	0,164
VISON - DT	0,193	0,134	0,252
VISON - STIMO	0,127	0,086	0,169
VISON - VSUMM	0,029	0,002	0,056

Uma vez que os intervalos de confiança não incluem zero em nenhum caso, os resultados apresentados na Tabela 7.4 confirmam que VISON produz resumos com qualidade superior (medida-F mais alta) em relação às demais técnicas. Considerando essas observações, é possível concluir que a abordagem proposta supera todos os métodos comparados para essa coleção de vídeos.

A Figura 7.11 apresenta os resumos produzidos por diferentes abordagens para o vídeo *A New Horizon, segment 2*. Os resumos dos usuários para o mesmo vídeo são apresentados na Figura 7.12. Nota-se que o resumo produzido pela DT contém quadros que são muito semelhantes entre si. Em contraste, os outros métodos fornecem um resumo mais conciso para esse vídeo. O resumo com a maior qualidade é conseguido com o uso do método proposto, o que pode ser confirmado por meio de uma comparação visual com os resumos dos usuários.

[7]Todos os resumos produzidos pelos métodos comparados para a coleção do Open Video estão disponíveis em `http://sites.google.com/site/vsummsite/` (último acesso em 14 de outubro de 2011).

(a) OV (provedores dos dados): medida-F = 0,417

(b) DT (Mundur *et al.* [119]).: medida-F = 0,268

(c) STIMO (Furini *et al.* [63]): medida-F = 0,526

(d) VSUMM (Avila *et al.* [22]): medida-F = 0,715

(e) VISON (abordagem proposta): medida-F = 0,873

Figura 7.11: Resumos obtidos por diferentes abordagens para o vídeo *A New Horizon, segment 2* (disponível via Open Video).

(a) Usuário 1

(b) Usuário 2

(c) Usuário 3

(d) Usuário 4

(e) Usuário 5

Figura 7.12: Resumos dos usuários para o vídeo *A New Horizon, segment 2*.

Uma característica que se sobressai é o comprimento dos resumos gerados pela abordagem proposta. Como ela produz resumos maiores, ela tende a acertar mais, mas também, a errar mais. Todavia, os usuários geralmente preferem resumos mais extensos que representam todos os diversos segmentos de um vídeo, independentemente de seu tamanho, como pode ser observado na Figura 7.12.

Seguindo o mesmo protocolo experimental, a técnica proposta foi também avaliada em vídeos do YouTube[8]. Nessa coleção, os parâmetros do VISON foram (Seção 7.4.3): ϵ ajustado em 0,15 e λ igual a 1% da duração do vídeo. Nesses experimentos, os resultados do VISON foram comparados com aqueles reportados pelo VSUMM [22][9].

A Tabela 7.5 apresenta os resultados de uma comparação entre VISON e VSUMM para diferentes categorias de vídeo. Os intervalos de confiança (a um nível de confiança de 98%) para a diferença entre a média da medida-F obtida por cada abordagem são apresentados na Tabela 7.6. As Figuras 7.13– 7.16 mostram os resumos produzidos por esses métodos e um resumo de usuário para as categorias de vídeo consideradas.

Tabela 7.5: A média da medida-F obtida por VSUMM e VISON em diferentes categorias de vídeo.

Categoria	Vídeos	VSUMM	VISON
Esportes	17	**0,556**	0,441
Notícias	15	0,630	**0,648**
Programas de TV	5	0,262	**0,264**
Comerciais	2	**0,722**	0,337
Vídeos Caseiros	1	0,240	**0,476**
Média Ponderada	40	**0,548**	0,492

Tabela 7.6: Diferença entre as médias da medida-F atingidas por VSUMM e VISON em diferentes categorias de vídeo, a um nível de confiança de 98%.

Método	*Média*	*Intervalo de Confiança (98%)*	
		mínimo	*máximo*
Esportes	-0,115	-0,179	-0,052
Notícias	0,018	-0,046	0,081
Programas de TV	0,002	-0,102	0,105
Comerciais	-0,385	-0,808	0,038
Vídeos Caseiros	0,236	0,236	0,236
Média Ponderada	-0,055	-0,112	0,001

[8]Os resumos produzidos pelo VISON para os 40 vídeos do YouTube estão disponíveis em `http://www.liv.ic.unicamp.br/~jurandy/vison/VISONSummary.zip` (último acesso em 14 de outubro de 2011).

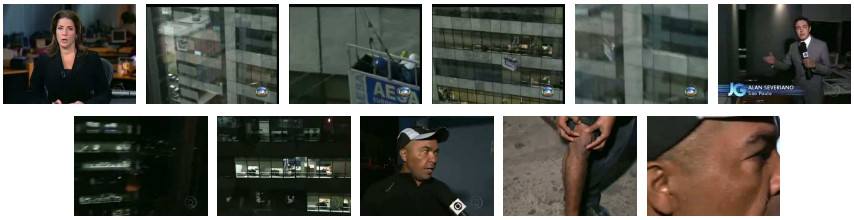
[9]Os resumos produzidos pelo VSUMM para a coleção do YouTube estão disponíveis em `http://sites.google.com/site/vsummsite/` (último acesso em 14 de outubro de 2011).

(a) VSUMM

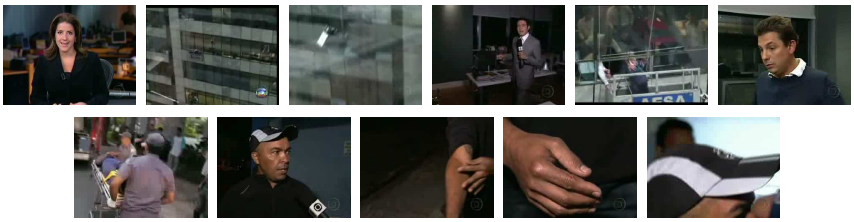
(b) VISON

(c) Usuário 5

Figura 7.13: Dois resumos automáticos e um resumo de usuário para um vídeo de esportes.



(a) VSUMM



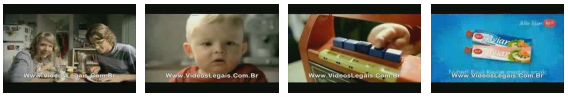
(b) VISON

(c) Usuário 4

Figura 7.14: Dois resumos automáticos e um resumo de usuário para um vídeo de notícias.

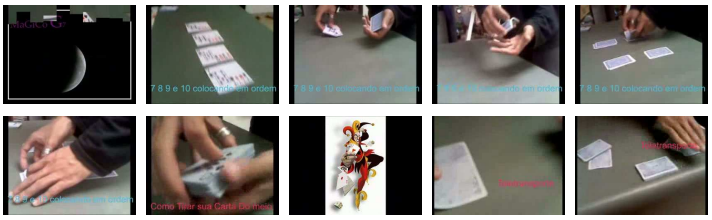
(a) VSUMM

(b) VISON



(c) Usuário 4

Figura 7.15: Dois resumos automáticos e um resumo de usuário para um comercial.



(a) VSUMM

(b) VISON

(c) Usuário 5

Figura 7.16: Dois resumos automáticos e um resumo de usuário para um vídeo caseiro.

A análise desse experimento mostra que VISON e VSUMM exibem um desempenho similar, uma vez que os intervalos de confiança incluem zero e, portanto, as diferenças entre eles não são significativas nesse nível de confiança. Em particular, VISON e VSUMM atingem uma medida-F comparável para vídeos de notícias, programas de TV e comerciais.

Para vídeos de esportes (sobretudo vídeos de futebol), VISON obtém uma medida-F baixa. Isso acontece porque a técnica proposta visa à produção de resumos concisos. Ao contrário do que se espera, os usuários preferem criar resumos que representam sequências inteiras de movimentos. Nesses casos, métodos voltados a domínios específicos podem ser uma escolha melhor.

Pelos valores apresentados na Tabela 7.5, parece que ambas as abordagens têm um baixo desempenho. No entanto, a principal razão para esses valores é o desacordo entre os usuários com relação aos resumos. É importante perceber que os melhores resultados alcançáveis estão dentro dos intervalos de confiança para as semelhanças entre os resumos dos usuários.

A principal vantagem da abordagem proposta é a sua eficiência computacional. Uma vez que o tempo necessário para elaborar resumos de vídeo é dependente da arquitetura física (com componentes mais potentes, a velocidade computacional aumenta, reduzindo o tempo de execução) e os códigos-fonte de todos os métodos comparados não estão disponíveis, é impossível realizar uma comparação justa de desempenho em relação a técnica apresentada.

A Tabela 7.7 apresenta o custo computacional e os requisitos de espaço (em termos do número de quadros n, do comprimento do resumo k e da dimensionalidade do vetor de características d) de todos os métodos comparados. Dessa forma, pode-se investigar a diferença relativa de desempenho entre diferentes abordagens.

Tabela 7.7: O custo computacional e os requisitos de espaço de diferentes abordagens.

Método	*Custo Computacional*	*Requisitos de Espaço*
DT	$O(n \log n)$	$O(nd)$
STIMO	$O(nk)$	$O(nd)$
VSUMM	$O(nk)$	$O(nd)$
VISON	$\boldsymbol{O(n)}$	$\boldsymbol{O(d)}$

Para avaliar a eficiência do método proposto mediante a produção de resumos estáticos, foi analisado o tempo por quadro gasto para executar todas as três etapas de seu algoritmo, excluindo-se o tempo de decodificação parcial de cada quadro. Esse estudo foi realizado sobre os 50 vídeos do Open Video, pois eles têm a mesma resolução e taxa de quadros por segundo. Para cada vídeo, 10 replicações foram realizadas a fim de garantir resultados estatisticamente sólidos.

De acordo com esses experimentos, a execução de todas as etapas da técnica proposta leva um tempo médio igual a, aproximadamente, 189 ± 18 microssegundos por quadro (nível de confiança superior a 99,9%). Para uso *online*, considerando um tempo máximo de espera de 39 segundos, como sugerido por Bouch *et al.* [34], esse método pode ser utilizado, em média, em vídeos de até 205.826 quadros (cerca de 115 minutos a uma taxa de 29,97 quadros por segundo). É importante lembrar que esses valores dependem do poder computacional da arquitetura física utilizada.

7.4.5 Desempenho em Resumos Dinâmicos

A abordagem proposta foi comparada a duas técnicas[10]: Le e Satoh [99] e Putpuek *et al.* [134]. A primeira ficou em 2^o lugar entre os 22 participantes da tarefa de sumarização de vídeos da BBC do TRECVID 2007 [124]. Os parâmetros utilizados para produzir os resumos de vídeo usando a abordagem proposta foram[11]: ϵ ajustado em 0,075, λ igual a 0,5% da duração do vídeo e κ igual a 25% dos quadros de cada fragmento.

A Figura 7.18 compara a qualidade do resumo de vídeo produzido por diferentes abordagens. Os resultados globais de cada medida de eficácia são apresentados como gráficos de caixa (do inglês, *boxplots*). A Figura 7.17 apresenta as convenções usadas em gráficos de caixa no modelo de Tukey [124, 125]. Esses gráficos estão ordenados pela fração da mediana da taxa de inclusão (IN), o que torna a comparação mais fácil.

Os resultados indicam que a abordagem proposta apresenta uma qualidade similar às demais estratégias no que diz respeito à fração do gabarito incluída no resumo de vídeo (IN) e a sua duração relativa sobre o vídeo original (DU). Por outro lado, os resumos produzidos pelo método proposto são melhores que aqueles das outras técnicas em termos de ausência de redundância (RE) e ausência de detritos (JU) e, portanto, eles são mais fáceis de entender.

A Tabela 7.8 apresenta os intervalos de confiança (a um nível de confiança de 95%) para as diferenças entre a técnica apresentada e os trabalhos anteriores. A análise desse experimento mostra que as inclusões (IN) e a duração (DU) dos resumos produzidos por essas abordagens são semelhantes. Uma vez que os intervalos de confiança não incluem zero para as outras medidas, os resultados confirmam que o método proposto produz resumos com qualidade superior (isto é, RE e JU mais altos) em relação a seus concorrentes.

[10] Todos os resumos produzidos pelos métodos comparados para o conjunto de dados do TRECVID 2007 estão disponíveis em `http://satoh-lab.ex.nii.ac.jp/users/ledduy/Demo-BBCRushes/` (último acesso em 14 de outubro de 2011).

[11] Os resumos produzidos pelo VISON para os vídeos do conjunto de dados do TRECVID 2007 estão disponíveis em `http://www.liv.ic.unicamp.br/~jurandy/videoskims/` (último acesso em 14 de outubro de 2011).

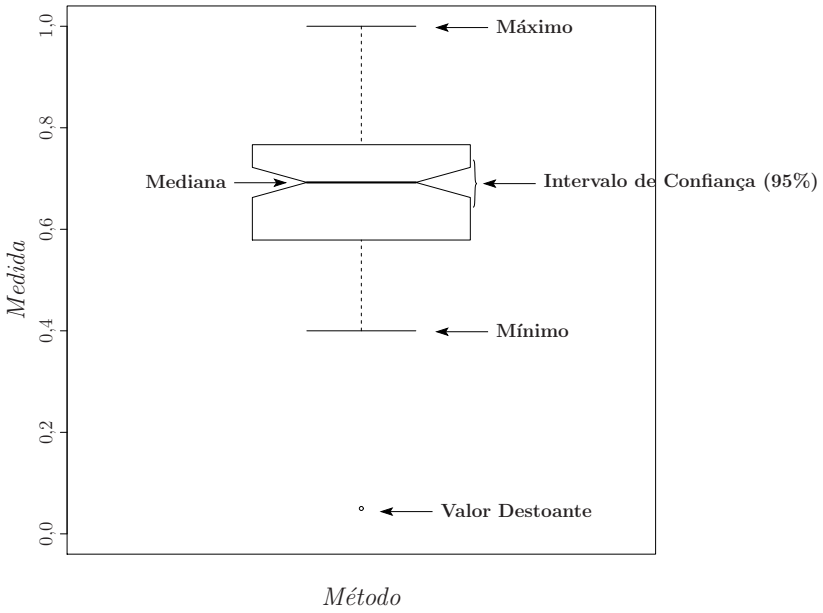
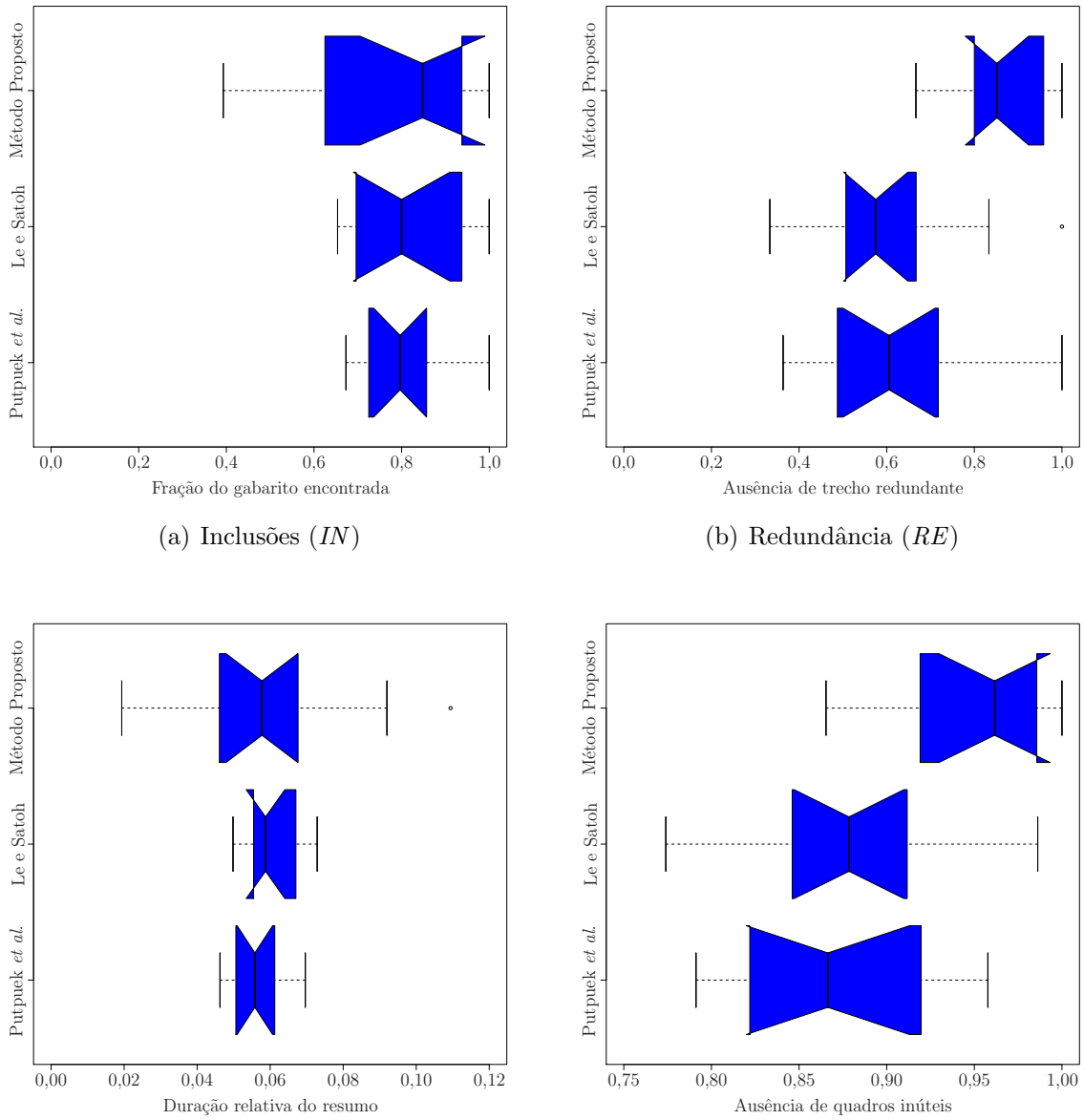


Figura 7.17: Exemplo de um gráfico de caixa no modelo de Tukey [124, 125].

Tabela 7.8: Diferenças entre a técnica apresentada e trabalhos anteriores a um nível de confiança de 95%.

Abordagem	*Medida*	*Mediana*	*Intervalo de Confiança (95%)*	
			mínimo	*máximo*
Método Proposto - Le e Satoh	*IN*	0,017	-0,152	0,254
	RE	0,239	0,069	0,423
	DU	0,005	-0,013	0,017
	JU	0,081	0,017	0,173
Método Proposto - Putpuek *et al.*	*IN*	0,000	-0,205	0,300
	RE	0,243	0,131	0,422
	DU	0,002	-0,016	0,014
	JU	0,050	0,020	0,146



(a) Inclusões (IN)

(b) Redundância (RE)

(c) Duração (DU)

(d) Detritos (JU)

Figura 7.18: Comparação da qualidade do resumo de vídeo produzido por diferentes abordagens.

A Tabela 7.9 apresenta o custo computacional e os requisitos de espaço (em termos do número de quadros n e da dimensionalidade do vetor de características d) de todos os métodos comparados. Claramente, a estratégia proposta é mais eficiente que as soluções existentes.

Tabela 7.9: O custo computacional e os requisitos de espaço de diferentes abordagens.

Método	*Custo Computacional*	*Requisitos de Espaço*
Le e Satoh[12]	$\Omega(n \log n)$	$O(nd)$
Putpuek *et al.*[12]	$\Omega(n \log n)$	$O(nd)$
Abordagem Proposta	$\boldsymbol{O(n)}$	$\boldsymbol{O(d)}$

Seguindo o protocolo anterior, a eficiência do método proposto mediante a produção de resumos dinâmicos foi avaliada usando todos os vídeos do conjunto de dados do TRECVID 2007. A fim de garantir resultados estatisticamente sólidos, foram realizadas 10 replicações para cada vídeo.

Esse estudo mostra que a execução de todas as etapas da técnica proposta leva um tempo médio igual a, aproximadamente, 347±48 microssegundos por quadro para produzir um trailer de vídeo (nível de confiança superior a 99,9%). Para uso *online*, considerando um tempo máximo de espera de 39 segundos [34], esse método pode ser aplicado, em média, a vídeos de até 112.460 quadros (cerca de 75 minutos a uma taxa de 25 quadros por segundo).

7.5 Conclusões

Neste capítulo, foi apresentado VISON, uma nova abordagem para a sumarização de sequências de vídeo que atua diretamente no domínio comprimido. Essa técnica baseia-se em características visuais extraídas do fluxo de vídeo e em um algoritmo simples e rápido para resumir o seu conteúdo. Tal combinação faz com que ela seja adequada à produção *online* de resumos de vídeo.

VISON permite que os usuários customizem a qualidade dos resumos de vídeo e também especifiquem o tempo que eles estão dispostos a esperar. A personalização do usuário é muito importante no cenário atual, pois os usuários frequentemente têm diferentes necessidades e recursos.

Um projeto experimental rigoroso foi conduzido em um conjunto de dados do TRECVID 2007 (BBC Rushes) e também em vídeos do Open Video e do YouTube, com o intuito de explorar o espaço de parâmetros da abordagem proposta e compará-lo com as soluções atuais. Resultados de uma avaliação subjetiva com usuários mostram que esse método apresenta alta eficácia e velocidade computacional.

Extensões futuras incluem a avaliação de outras características visuais e medidas de similaridade. Além disso, o método proposto pode ser ampliado para considerar características locais [4–6] e/ou análise de movimento [7–10].

[12] O tempo de execução é dominado pelo método GreedyRSC, cujo custo total é, aproximadamente, $O(b^2\ n \log \frac{k}{b} + b\ n \log n \log \frac{k}{b})$, em que k é o tamanho do maior grupo e b é o tamanho da vizinhança [83].

Capítulo 8

Conclusões e Perspectivas

A pesquisa em sistemas capazes de gerir vídeos digitais a partir de seu conteúdo vem recebendo crescente atenção por parte da comunidade científica. Esse interesse pode ser explicado por vários fatores, como por exemplo, (1) a redução do custo de equipamentos de captura, transmissão e armazenamento de vídeos, (2) o crescimento exponencial do número de vídeos disponíveis via Internet, (3) os desafios científicos envolvidos na gestão eficiente desse complexo material, (4) as inúmeras aplicações práticas como bibliotecas digitais e sistemas de segurança e (5) a inadequação de técnicas tradicionais para descrever, representar e realizar buscas em grandes coleções.

Apesar dos avanços significativos em métodos voltados à idealização de tais sistemas, a maioria dos trabalhos existentes envolve soluções computacionalmente caras, em termos tanto de tempo quanto espaço, limitando a sua aplicação apenas ao ambiente acadêmico e/ou a grandes empresas.

No cenário atual, que tem sido caracterizado por uma busca crescente por soluções que integrem portabilidade e conectividade usando dispositivos com capacidade limitada, tais como celulares, *smartphones* e *tablets*, é imperativo o desenvolvimento de técnicas tanto eficazes quanto eficientes, a fim de permitir que um público maior tenha acesso a tecnologias vigentes.

Neste trabalho, foi realizado um estudo abrangente em sistemas de gestão de vídeos por conteúdo, abordando temas que englobam desde a caracterização de vídeos usando suas propriedades visuais, como cor, forma, textura e movimento, até aspectos de indexação e armazenamento dessas informações.

Mais especificamente, este trabalho apresentou cinco abordagens originais voltadas à análise, indexação e recuperação de vídeos digitais. O resultado final da combinação dos métodos propostos é a especificação e implementação de um protótipo de ambiente computacional para a gestão de vídeos que é, ao mesmo tempo, eficiente e eficaz, podendo ser aplicado com sucesso em dispositivos com baixo poder computacional e/ou em ambientes

que necessitem de uma resposta rápida, mantendo um padrão de qualidade comparável ao estado da arte.

Em síntese, as principais contribuições deste trabalho são:

1. *Estudo abrangente em soluções para a gestão de vídeos por análise de conteúdo.*
2. *Proposta de cinco abordagens originais voltadas à análise, indexação e recuperação de vídeos [8, 9, 11–13, 15–17, 19, 20].*
3. *Avaliação experimental bem fundamentada das técnicas propostas em comparação ao estado da arte.*
4. *Especificação de um sistema de gestão de vídeos digitais para ambientes que exigem resposta rápida.*

Há várias extensões previstas, tanto do ponto de vista teórico quanto de implementação para os diferentes módulos da arquitetura. Algumas dessas extensões, inclusive, já estão sendo analisadas. Elas incluem:

- **Extensão para outros formatos de vídeo.** Para aumentar a eficiência computacional, o sistema proposto explora o padrão de compressão de vídeo a fim de processar vídeos diretamente em sua forma comprimida. Devido à sua estabilidade e simplicidade, o padrão de codificação MPEG, e mais especificamente, o formato MPEG-1/2, foi escolhido como base sobre a qual as abordagens propostas foram fundamentadas. Em geral, diferentes padrões de codificação de vídeo seguem um mesmo paradigma básico. Portanto, podem-se estender essas soluções a diferentes padrões de codificação, por exemplo, H.263 e H.264.

- **Extensão para outras características.** Este trabalho tem como foco a informação de movimento. Porém, outras propriedades podem ser exploradas na análise, indexação e recuperação de vídeos. Uma característica em potencial nessa tarefa é a consistência temporal de dados de vídeo. Consistência temporal refere-se à observação de que tomadas de vídeo temporalmente adjacentes têm conteúdo visual e semântico similar. Isso implica que tomadas relevantes que correspondam a um conceito semântico específico ou a um tópico de consulta tendem a se reunir em vizinhanças temporais ou mesmo aparecer uma ao lado da outra. Dessa forma, pode-se fazer previsão mais apurada quanto à relevância de uma tomada, considerando a relevância de suas tomadas vizinhas.

- **Suporte a operações espaço-temporais.** Uma particularidade que distingue os dados de vídeo é o conteúdo temporal. Na arquitetura apresentada, o processamento

de consultas é feito apenas com base no relacionamento espacial entre assinaturas extraídas do conteúdo semântico dos dados de vídeo. A definição e implementação de mecanismos para o suporte a operações baseadas em tempo e espaço precisam ser investigadas. Uma possível solução para esse problema é modelar o relacionamento espaço-temporal entre assinaturas extraídas de tomadas de um mesmo vídeo como uma cadeia de caracteres. Dessa forma, o processamento de consultas poderia ser feito com base em algoritmos de alinhamento de sequências.

- **Integração de diferentes tipos de dados.** O processamento de consultas envolvendo o conteúdo de vídeos pode englobar diferentes tipos de dados, tais como texto, som e imagem. Uma extensão nesse sentido consiste em investigar a utilização de mecanismos para a combinação de resultados provenientes de métodos que processam diferentes recursos. Uma forma de integrar esses dados seria estender o sistema atual como uma composição de subsistemas menores, um para cada tipo de dado. Isso poderia ser feito de forma hierárquica (a saída de um sistema é utilizado como a entrada de outro, refinando o resultado a cada iteração) ou paralela (cada sistema é tratado de forma independente e o resultado final é dado pela composição da resposta gerada por cada um).

- **Interação com usuários.** No sistema atual, após o processamento de uma consulta, um usuário é apresentado a uma lista de vídeos similares, na qual cada vídeo é representado pelo seu resumo. Embora simples, esse modelo requer muito tempo para identificar resultados relevantes. Uma alternativa nesse sentido seria agrupar resumos similares em uma estrutura hierárquica [18, 132]. Nesse caso, seria necessário estender o módulo de visualização como uma ferramenta de navegação intuitiva na qual os usuários possam buscar interativamente por um vídeo, sem ter que ver através de muitos resultados possíveis ao mesmo tempo.

As contribuições supracitadas e outras derivadas deste trabalho foram publicadas em:

Artigos em Periódicos

ALMEIDA, J.; VALLE, E.; TORRES, R. S.; LEITE, N. J. DAHC-tree: An Effective Index for Approximate Search in High-Dimensional Metric Spaces. *Journal of Information and Data Management*, v. 1, n. 3, p. 375-390, 2010.

PINTO CÁCERES, S. M.; ALMEIDA, J.; NERIS, V. P. A.; BARANAUSKAS, M. C. C.; TORRES, R. S.; LEITE, N. J. Navigating through Video Stories using Clustering Sets. *International Journal of Multimedia Data Engineering and Management*, v. 2, n. 3, p. 1-20, 2012. DOI: 10.4018/jmdem.2011070101

ALMEIDA, J.; LEITE, N. J.; TORRES, R. S. VISON: Video Summarization for Online Applications. *Pattern Recognition Letters*, v. 33, n. 4, p. 397-409, 2012. DOI: 10.1016/j.patrec.2011.08.007

ALMEIDA, J.; LEITE, N. J.; TORRES, R. S. Online Video Summarization on Compressed Domain. *Journal of Visual Communication and Image Representation*, 2012. DOI: 10.1016/j.jvcir.2012.01.009

Trabalhos Completos

ALMEIDA, J.; MINETTO, R.; ALMEIDA, T. A.; TORRES, R. S.; LEITE, N. J. Robust Estimation of Camera Motion using Optical Flow Models. Em *Proceedings of the 5th International Symposium on Visual Computing (ISVC'09)*, v. 5875, *Lecture Notes in Computer Science*, p. 435-446, Las Vegas, NV, USA, 2009. DOI: 10.1007/978-3-642-10331-5_41

ALMEIDA, J.; MINETTO, R.; ALMEIDA, T. A.; TORRES, R. S.; LEITE, N. J. Estimation of Camera Parameters in Video Sequences with a Large Amount of Scene Motion. Em *Proceedings of the 17th International Conference on Systems, Signals and Image Processing (IWSSIP'10)*, p. 348-351, Rio de Janeiro, RJ, Brasil, 2010.

ALMEIDA, J.; TORRES, R. S.; LEITE, N. J. Rapid Video Summarization on Compressed Video. Em *Proceedings of the 12th IEEE International Symposium on Multimedia (ISM'10)*, p. 113-120, Taichung, Taiwan, 2010. DOI: 10.1109/ISM.2010.25

KOZIEVITCH, N. P.; ALMEIDA, J.; TORRES, R. S.; SANTANCHE, A; LEITE, N. J. Reusing a Compound-Based Infrastructure for Searching Video Stories. Em *Proceedings of the 12th IEEE International Conference on Information Reuse and Integration (IEEE IRI'11)*, p. 222-227, Las Vegas, NV, USA, 2011. DOI: 10.1109/IRI.2011.6009550

ALMEIDA, J.; LEITE, N. J.; TORRES, R. S. Comparison of Video Sequences with Histograms of Motion Patterns. Em *Proceedings of the 18th IEEE International Conference on Image Processing (ICIP'11)*, p. 3673-3676, Bruxelas, Bélgica, 2011. DOI: 10.1109/ICIP.2011.6116516

ALMEIDA, J.; LEITE, N. J.; TORRES, R. S. Rapid Cut Detection on Compressed Video. Em *Proceedings of the 16th Iberoamerican Congress on Pattern Recognition (CIARP'11)*, v. 5875, *Lecture Notes in Computer Science*, p. 71-78, Pucón, Chile, 2011. DOI: 10.1007/978-3-642-25085-9_8

Resumos Expandidos

ALMEIDA, J.; TORRES, R. S.; LEITE, N. J. BP-tree: An Efficient Index for Similarity Search in High-Dimensional Metric Spaces. Em *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM'10)*, p. 1365-1368, Toronto, ON, Canadá, 2010. DOI: 10.1145/1871437.1871622

LI, L. T.; ALMEIDA, J.; LEITE, N. J.; TORRES, R. S. RECOD Working Notes for Placing Task MediaEval 2011. Em *Working Notes Proceedings of the MediaEval 2011 Workshop (MEDIAEVAL'11)*, v. 807, Pisa, Itália, 2011.

Relatórios Técnicos

ALMEIDA, J.; MINETTO, R.; ALMEIDA, T. A.; TORRES, R. S.; LEITE, N. J. Robust Estimation of Camera Motion using Local Invariant Features. Relatório Técnico, Instituto de Computação, Universidade Estadual de Campinas, IC-09-12, 2009.

ALMEIDA, J.; MINETTO, R.; ALMEIDA, T. A.; TORRES, R. S.; LEITE, N. J. Robust Estimation of Camera Motion using Optical Flow Models. Relatório Técnico, Instituto de Computação, Universidade Estadual de Campinas, IC-09-24, 2009.

ALMEIDA, J.; TORRES, R. S.; LEITE, N. J. BP-tree: An Efficient Index for Similarity Search in High-Dimensional Metric Spaces. Relatório Técnico, Instituto de Computação, Universidade Estadual de Campinas, IC-10-27, 2010.

ALMEIDA, J.; PINTO CÁCERES, S. M.; TORRES, R. S.; LEITE, N. J. Intuitive Video Browsing Along Hierarchical Trees. Relatório Técnico, Instituto de Computação, Universidade Estadual de Campinas, IC-11-06, 2011.

ALMEIDA, J.; LEITE, N. J.; TORRES, R. S. Searching in High-Dimensional Metric Spaces using BP-Trees. Relatório Técnico, Instituto de Computação, Universidade Estadual de Campinas, IC-11-08, 2011.

Referências Bibliográficas

[1] D. A. Adjeroh, M.-C. Lee e I. King. A distance measure for video sequence similarity matching. Em *Proceedings of the IEEE International Workshop on Multi-Media Database Management Systems*, p. 72–79, Dayton, OH, EUA, Agosto 5–7 1998. IEEE Computer Society.

[2] C. C. Aggarwal e P. S. Yu. The IGrid index: reversing the dimensionality curse for similarity indexing in high dimensional space. Em *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, p. 119–129, Boston, MA, EUA, Agosto 20–23 2000. ACM Press.

[3] F. A. Al-Omari e M. A. Al-Jarrah. Query by image and video content: a colored-based stochastic model approach. *Data and Knowledge Engineering*, 52(3):313–332, Março 2005.

[4] J. Almeida. Recuperação de imagens por cor utilizando análise de distribuição discreta de características. Dissertação de Mestrado, Instituto de Computação, UNICAMP, Campinas, SP, Brasil, Agosto 2007.

[5] J. Almeida, A. Rocha, R. da S. Torres e S. Goldenstein. Image retrieval based on color and scale representative image regions (CSIR). Relatório Técnico IC-07-28, Instituto de Computação, UNICAMP, 2007.

[6] J. Almeida, A. Rocha, R. da S. Torres e S. Goldenstein. Making colors worth more than a thousand words. Em R. L. Wainwright e H. Haddad (eds.), *Proceedings of the ACM International Symposium on Applied Computing*, p. 1180–1186, Fortaleza, CE, Brasil, Março 16–20 2008. ACM Press.

[7] J. Almeida, R. Minetto, T. A. Almeida, R. da S. Torres e N. J. Leite. Robust estimation of camera motion using local invariant features. Relatório Técnico IC-09-12, Instituto de Computação, UNICAMP, 2009.

[8] J. Almeida, R. Minetto, T. A. Almeida, R. da S. Torres e N. J. Leite. Robust estimation of camera motion using optical flow models. Relatório Técnico IC-09-24, Instituto de Computação, UNICAMP, 2009.

[9] J. Almeida, R. Minetto, T. A. Almeida, R. da S. Torres e N. J. Leite. Robust estimation of camera motion using optical flow models. Em G. Bebis, R. D. Boyle, B. Parvin, D. Koracin, Y. Kuno, J. Wang, R. Pajarola, P. Lindstrom, A. Hinkenjann, M. L. Encarnacao, C. Silva e D. Coming (eds.), *Proceedings of the International Symposium on Advances in Visual Computing*, v. 5875, *Lecture Notes in Computer Science*, p. 435–446, Las Vegas, NV, EUA, Novembro 30 – Dezembro 2 2009. Springer.

[10] J. Almeida, R. Minetto, T. A. Almeida, R. da S. Torres e N. J. Leite. Estimation of camera parameters in video sequences with a large amount of scene motion. Em *Proceedings of the IEEE International Conference on Systems, Signals and Image Processing*, p. 348–351, Rio de Janeiro, RJ, Brasil, Junho 17–19 2010. IEEE Computer Society.

[11] J. Almeida, R. da S. Torres e N. J. Leite. BP-tree: An efficient index for similarity search in high-dimensional spaces. Relatório Técnico IC-10-27, Instituto de Computação, UNICAMP, 2010.

[12] J. Almeida, R. da S. Torres e N. J. Leite. BP-tree: An efficient index for similarity search in high-dimensional metric spaces. Em *Proceedings of the ACM International Conference on Information and Knowledge Management*, p. 1365–1368, Toronto, ON, Canadá, Outubro 26–30 2010. ACM Press.

[13] J. Almeida, R. da S. Torres e N. J. Leite. Rapid video summarization on compressed video. Em *Proceedings of the IEEE International Symposium on Multimedia*, p. 113–120, Taichung, Taiwan, Dezembro 13–15 2010. IEEE Computer Society.

[14] J. Almeida, E. Valle, R. da S. Torres e N. J. Leite. DAHC-tree: An effective index for approximate search in high-dimensional metric spaces. *Journal of Information and Data Management*, 1(3):375–390, Outubro 2010.

[15] J. Almeida, N. J. Leite e R. da S. Torres. Searching in high-dimensional metric spaces using BP-trees. Relatório Técnico IC-11-08, Instituto de Computação, UNICAMP, 2011.

[16] J. Almeida, N. J. Leite e R. da S. Torres. Rapid cut detection on compressed video. Em C. S. Martín e S.-W. Kim (eds.), *Proceedings of the Iberoamerican Congress on*

Pattern Recognition, Lecture Notes in Computer Science, Pucón, Chile, Novembro 15–18 2011. Springer.

[17] J. Almeida, N. J. Leite e R. da S. Torres. Comparison of video sequences with histograms of motion patterns. Em B. Macq e P. Schelkens (eds.), *Proceedings of the IEEE International Conference on Image Processing*, p. 3673–3676, Bruxelas, Bélgica, Setembro 11–14 2011. IEEE Computer Society.

[18] J. Almeida, S. M. Pinto-Cáceres, R. da S. Torres e N. J. Leite. Intuitive video browsing along hierarchical trees. Relatório Técnico IC-11-06, Instituto de Computação, UNICAMP, 2011.

[19] J. Almeida, N. J. Leite e R. da S. Torres. VISON: VIdeo Summarization for ONline applications. *Pattern Recognition Letters*, 33(4):397–409, Março 2012.

[20] J. Almeida, N. J. Leite e R. da S. Torres. Online video summarization on compressed domain. *Journal of Visual Communication and Image Representation*, 2012.

[21] S. E. F. Avila. Uma abordagem baseada em características de cor para a elaboração automática e avaliação subjetiva de resumos estáticos de vídeos. Dissertação de Mestrado, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil, Setembro 2008.

[22] S. E. F. Avila, A. P. B. Lopes, A. Luz Jr. e A. A. Araújo. VSUMM: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters*, 32(1):56–68, Janeiro 2011.

[23] R. A. Baeza-Yates e B. A. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, EUA, 1999.

[24] R. A. Baeza-Yates, W. Cunto, U. Manber e S. Wu. Proximity matching using fixed-queries trees. Em M. Crochemore e D. Gusfield (eds.), *Proceedings of the Annual Symposium on Combinatorial Pattern Matching*, v. 807, *Lecture Notes in Computer Science*, p. 198–212, Asilomar, CA, EUA, Junho 5–8 1994. Springer.

[25] M. C. N. Barioni, H. L. Razente, A. J. M. Traina e C. Traina Jr. Accelerating k-medoid-based algorithms through metric access methods. *Journal of Systems and Software*, 81(3):343–355, Março 2008.

[26] J. L. Barron, D. J. Fleet e S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, Fevereiro 1994.

[27] K. P. Bennett, U. M. Fayyad e D. Geiger. Density-based indexing for approximate nearest-neighbor queries. Em *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, p. 233–243, São Diego, CA, EUA, Agosto 15–18 1999. ACM Press.

[28] F. N. Bezerra e N. J. Leite. Using string matching to detect video transitions. *Pattern Analysis and Applications*, 10(1):45–54, Fevereiro 2007.

[29] F. N. Bezerra e E. Lima. Low cost soccer video summaries based on visual rhythm. Em J. Z. Wang, N. Boujemaa e Y. Chen (eds.), *Proceedings of the ACM International Workshop on Multimedia Information Retrieval*, pages 71–78, Santa Bárbara, CA, EUA, Outubro 26–27 2006. ACM Press.

[30] V. Bhaskaran e K. Konstantinides. *Image and Video Compression Standards: Algorithms and Architectures*. Kluwer Academic Publishers, Norwell, MA, EUA, 2ª edição, 1997.

[31] D. N. Bhat e S. K. Nayar. Ordinal measures for image correspondence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):415–423, Abril 1998.

[32] A. Bimbo. *Visual information retrieval*. Morgan Kaufmann Publishers Inc., São Francisco, CA, EUA, 1999.

[33] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Inc., Secaucus, NJ, EUA, 2006.

[34] A. Bouch, A. Kuchinsky e N. T. Bhatti. Quality is in the eye of the beholder: meeting users' requirements for internet quality of service. Em *Proceedings of the ACM International Conference on Human Factors in Computing Systems*, p. 297–304, The Hague, Holanda, Abril 1–6 2000. ACM Press.

[35] T. Bozkaya e Z. M. Özsoyoglu. Indexing large metric spaces for similarity search queries. *ACM Transactions on Database Systems*, 24(3):361–404, Setembro 1999.

[36] H. Bredin, D. Byrne, H. Lee, N. E. O'Connor e G. J. F. Jones. Dublin city university at the TRECVid 2008 BBC rushes summarisation task. Em P. Over e A. F. Smeaton (eds.), *Proceedings of the ACM International Workshop on Video Summarization*, p. 45–49, Vancouver, BC, Canadá, Outubro 31 2008. ACM Press.

[37] S. Brin. Near neighbor search in large metric spaces. Em U. Dayal, P. M. D. Gray e S. Nishio (eds.), *Proceedings of the International Conference on Very Large Data Bases*, p. 574–584, Zurique, Suiça, Setembro 11–15 1995. Morgan Kaufmann.

[38] W. A. Burkhard e R. M. Keller. Some approaches to best-match file searching. *Communications of the ACM*, 16(4):230–236, Abril 1973.

[39] J. Calic, D. P. Gibson e N. W. Campbell. Efficient layout of comic-like video summaries. *IEEE Transactions on Circuits Systems and Video Technology*, 17(7): 931–936, Julho 2007.

[40] H. S. Chang, S. Sull e S. U. Lee. Efficient video indexing scheme for content-based retrieval. *IEEE Transactions on Circuits Systems and Video Technology*, 9(8):1269–1279, Dezembro 1999.

[41] S.-F. Chang, W. Chen, H. J. Meng, H. Sundaram e D. Zhong. A fully automated content-based video search engine supporting spatio-temporal queries. *IEEE Transactions on Circuits Systems and Video Technology*, 8(5):602–615, Setembro 1998.

[42] V. Chasanis, A. Likas e N. P. Galatsanos. Video rushes summarization using spectral clustering and sequence alignment. Em P. Over e A. F. Smeaton (eds.), *Proceedings of the ACM International Workshop on Video Summarization*, p. 75–79, Vancouver, BC, Canadá, Outubro 31 2008. ACM Press.

[43] V. T. Chasanis, A. C. Likas e N. P. Galatsanos. Scene detection in videos using shot clustering and sequence alignment. *IEEE Transactions on Multimedia*, 11(1): 89–100, Janeiro 2009.

[44] K. Chatterjee e S.-C. Chen. Hierarchical affinity hybrid tree: A multidimensional index structure to organize videos and support content-based retrievals. Em *Proceedings of the IEEE International Conference on Information Reuse and Integration*, p. 435–440, Las Vegas, NV, EUA, Julho 13–15 2008. IEEE Systems, Man e Cybernetics Society.

[45] K. Chatterjee e S.-C. Chen. GeM-tree: Towards a generalized multidimensional index structure supporting image and video retrieval. Em *Proceedings of the IEEE International Symposium on Multimedia*, p. 631–636, Berkley, CA, EUA, Dezembro 15–17 2008. IEEE Computer Society.

[46] E. Chávez e G. Navarro. A compact space decomposition for effective metric indexing. *Pattern Recognition Letters*, 26(9):1363–1376, Julho 2005.

[47] E. Chávez, G. Navarro, R. A. Baeza-Yates e J. L. Marroquín. Searching in metric spaces. *ACM Computing Surveys*, 33(3):273–321, Setembro 2001.

[48] G. C. Chavez. *Análise de Conteúdo de Vídeo por Meio do Aprendizado Ativo*. Tese de Doutorado, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil, Julho 2007.

[49] L. Chen e F. W. M. Stentiford. Video sequence matching based on temporal ordinal measurement. *Pattern Recognition Letters*, 29(13):1824–1831, Outubro 2008.

[50] S.-C. Cheung. *Efficient Video Similarity Measurement and Search*. Tese de Doutorado, University of California, Berkeley, CA, EUA, Dezembro 2002.

[51] S.-C. S. Cheung e A. Zakhor. Efficient video similarity measurement with video signature. *IEEE Transactions on Circuits Systems and Video Technology*, 13(1): 59–74, Janeiro 2003.

[52] S.-C. S. Cheung e A. Zakhor. Fast similarity search and clustering of video sequences on the world-wide-web. *IEEE Transactions on Multimedia*, 7(3):524–537, Junho 2005.

[53] P. Ciaccia, M. Patella e P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. Em M. Jarke, M. J. Carey, K. R. Dittrich, F. H. Lochovsky, P. Loucopoulos e M. A. Jeusfeld (eds.), *Proceedings of the International Conference on Very Large Data Bases*, p. 426–435, Atenas, Grécia, Agosto 25–29 1997. Morgan Kaufmann.

[54] N. Dalal e B. Triggs. Histograms of oriented gradients for human detection. Em *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, p. 886–893, São Diego, CA, EUA, Junho 20–26 2005. IEEE Computer Society.

[55] D. DeMenthon, V. Kobla e D. S. Doermann. Video summarization by curve simplification. Em *Proceedings of the ACM International Conference on Multimedia*, p. 211–218, Bristol, Inglaterra, Setembro 12–16 1998. ACM Press.

[56] F. Dufaux e J. Konrad. Efficient, robust, and fast global motion estimation for video coding. *IEEE Transactions on Image Processing*, 9(3):497–501, Março 2000.

[57] E. Dumont e B. Mérialdo. Rushes video summarization and evaluation. *Multimedia Tools and Applications*, 48(1):51–68, Maio 2010.

[58] R. A. Elmasri e S. B. Navathe. *Fundamentals of Database Systems*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, EUA, 2005.

[59] R. Ewerth, M. Schwalb, P. Tessmann e B. Freisleben. Estimation of arbitrary camera motion in MPEG videos. Em *Proceedings of the IEEE International Conference on Pattern Recognition*, p. 512–515, Cambridge, Reino Unido, Agosto 23–26 2004. IEEE Computer Society.

[60] M. A. Fischler e R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, Junho 1981.

[61] M. Flickner, H. S. Sawhney, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele e P. Yanker. Query by image and video content: The QBIC system. *IEEE Computer*, 28(9):23–32, Setembro 1995.

[62] Y. Fu, Z. Li, T. S. Huang e A. K. Katsaggelos. Locally adaptive subspace and similarity metric learning for visual data clustering and retrieval. *Computer Vision and Image Understanding*, 110(3):390–402, Junho 2008.

[63] M. Furini, F. Geraci, M. Montangero e M. Pellegrini. STIMO: STIll and MOving video storyboard for the web scenario. *Multimedia Tools and Applications*, 46(1): 47–69, Janeiro 2010.

[64] G. W. Furnas, T. K. Landauer, L. M. Gomez e S. T. Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11): 964–971, Novembro 1987.

[65] V. Gaede e O. Günther. Multidimensional access methods. *ACM Computing Surveys*, 30(2):170–231, Setembro 1998.

[66] J.-M. Geusebroek, G. J. Burghouts e A. W. M. Smeulders. The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112, Janeiro 2005.

[67] M. Ghanbari. *Standard Codecs: Image Compression to Advanced Video Coding*. Institution of Engineering and Technology, Londres, Reino Unido, 2003.

[68] W. J. Gillespie e D. T. Nguyen. Robust estimation of camera motion in MPEG domain. Em *Proceedings of the IEEE Region 10 Conference on Analog and Digital Techniques in Electrical Engineering*, p. 395–398, Chiang Mai, Tailândia, Novembro 21–24 2004. IEEE Computer Society.

[69] Y. Gong e X. Lin. Video summarization using singular value decomposition. Em *Proceedings of the IEEE International Conference on Computer Vision and Pat-*

tern Recognition, p. 2174–2180, Hilton Head, SC, EUA, Junho 13–15 2000. IEEE Computer Society.

[70] R. C. Gonzalez e R. E. Woods. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, EUA, 2 edition, 2001.

[71] H. Greenspan, J. Goldberger e A. Mayer. A probabilistic framework for spatio-temporal video representation & indexing. Em A. Heyden, G. Sparr, M. Nielsen e P. Johansen (eds.), *Proceedings of the European Conference on Computer Vision*, v. 2353, *Lecture Notes in Computer Science*, p. 461–475, Copenhague, Dinamarca, Maio 28–31 2002. Springer.

[72] G. Griffin, A. Holub e P. Perona. Caltech-256 object category dataset. Relatório Técnico 7694, California Institute of Technology, 2007.

[73] S. J. F. Guimarães, M. Couprie, A. A. Araújo e N. J. Leite. Video segmentation based on 2D image analysis. *Pattern Recognition Letters*, 24(7):947–957, Abril 2003.

[74] S. J. F. Guimarães, Z. K. G. Patrocínio Jr., H. B. Paula e H. B. Silva. A new dissimilarity measure for cut detection using bipartite graph matching. *International Journal of Semantic Computing*, 3(2):155–181, Junho 2009.

[75] D. Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, Nova Iorque, NY, EUA, 1997.

[76] A. Hampapur e R. M. Bolle. Feature based indexing for media tracking. Em *Proceedings of the IEEE International Conference on Multimedia and Expo*, p. 1709–1712, Nova Iorque, NY, EUA, Julho 30 – Agosto 2 2000. IEEE Computer Society.

[77] A. Hampapur e R. M. Bolle. Comparison of distance measures for video copy detection. Em *Proceedings of the IEEE International Conference on Multimedia and Expo*, p. 737–740, Tóquio, Japão, Agosto 22–25 2001. IEEE Computer Society.

[78] A. Hampapur, A. Gupta, B. Horowitz, C.-F. Shu, C. Fuller, J. R. Bach, M. Gorkani e R. Jain. Virage video engine. Em I. K. Sethi e R. Jain (eds.), *Proceedings of the SPIE International Conference on Storage and Retrieval for Image and Video Databases*, p. 188–198, São José, CA, EUA, Fevereiro 8–14 1997. SPIE.

[79] A. Hampapur, K. Hyun e R. M. Bolle. Comparison of sequence matching techniques for video copy detection. Em M. M. Yeung, C.-S. Li e R. W. Lienhart (eds.), *Proceedings of the SPIE International Conference on Storage and Retrieval for Media Databases*, v. 4676, p. 194–201, São José, CA, EUA, Janeiro 23 2002. SPIE.

[80] A. Hanjalic. Shot-boundary detection: Unraveled and resolved? *IEEE Transactions on Circuits Systems and Video Technology*, 12(2):90–105, Fevereiro 2002.

[81] A. Hanjalic e H. Zhang. An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis. *IEEE Transactions on Circuits Systems and Video Technology*, 9(8):1280–1289, Dezembro 1999.

[82] L. Herranz e J. M. Martínez. An efficient summarization algorithm based on clustering and bitstream extraction. Em *Proceedings of the IEEE International Conference on Multimedia and Expo*, p. 654–657, Nova Iorque, NY, EUA, Junho 28 – Julho 2 2009. IEEE Computer Society.

[83] M. E. Houle. The relevant-set correlation model for data clustering. *Statistical Analysis and Data Mining*, 1(3):157–176, Novembro 2008.

[84] X.-S. Hua, X. Chen e H. Zhang. Robust video signature based on ordinal measure. Em *Proceedings of the IEEE International Conference on Image Processing*, p. 685–688, Cingapura, Outubro 24–27 2004. IEEE Computer Society.

[85] J. Huang, R. Kumar, M. Mitra, W.-J. Zhu e R. Zabih. Image indexing using color correlograms. Em *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, p. 762–768, San Juan, Porto Rico, Junho 17–19 1997. IEEE Computer Society.

[86] D. P. Huttenlocher, G. A. Klanderman e W. Rucklidge. Comparing images using the hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, Setembro 1993.

[87] G. Iyengar e A. Lippman. Distributional clustering for efficient content-based retrieval of images and video. Em *Proceedings of the IEEE International Conference on Image Processing*, p. 81–84, Vancouver, BC, Canadá, Setembro 10–13 2000. IEEE Computer Society.

[88] H. V. Jagadish, A. O. Mendelzon e T. Milo. Similarity-based queries. Em *Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, p. 36–45, São José, CA, EUA, Maio 22–25 1995. ACM Press.

[89] H. V. Jagadish, B. C. Ooi, K.-L. Tan, C. Yu e R. Zhang. idistance: An adaptive b^+-tree based indexing method for nearest neighbor search. *ACM Transactions on Database Systems*, 30(2):364–397, Junho 2005.

[90] R. Jain. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation e Modeling*. John Wiley and Sons, Inc., Nova Iorque, NY, EUA, 1991.

[91] C. W. Johnson e M. D. Dunlop. Subjectivity and notions of time and value in interactive information retrieval. *Interacting with Computers*, 10(1):67–75, Março 1998.

[92] C. Kim e B. Vasudev. Spatiotemporal sequence matching for efficient video copy detection. *IEEE Transactions on Circuits Systems and Video Technology*, 15(1): 127–132, Janeiro 2005.

[93] J.-G. Kim, H. S. Chang, J. Kim e H.-M. Kim. Efficient camera motion characterization for mpeg video indexing. Em *Proceedings of the IEEE International Conference on Multimedia and Expo*, p. 1171–1174, Nova Iorque, NY, EUA, Julho 30 – Agosto 2 2000. IEEE Computer Society.

[94] S. Kiranyaz e M. Gabbouj. Hierarchical cellular tree: An efficient indexing scheme for content-based retrieval on multimedia databases. *IEEE Transactions on Multimedia*, 9(1):102–119, Janeiro 2007.

[95] J. Kleban, A. Sarkar, E. Moxley, S. Mangiat, S. Joshi, T. Kuo e B. S. Manjunath. Feature fusion and redundancy pruning for rush video summarization. Em P. Over e A. F. Smeaton (eds.), *Proceedings of the ACM International Workshop on Video Summarization*, p. 84–88, Bavaria, Alemanha, Setembro 28 2007. ACM Press.

[96] I. Koprinska e S. Carrato. Temporal video segmentation: A survey. *Signal Processing: Image Communication*, 16(5):477–500, Janeiro 2001.

[97] R. R. Korfhage. *Information Storage and Retrieval*. John Wiley and Sons Inc., Nova Iorque, NY, EUA, 1997.

[98] J. Law-To, L. Chen, A. Joly, I. Laptev, O. Buisson, V. Gouet-Brunet, N. Boujemaa e F. Stentiford. Video copy detection: A comparative study. Em N. Sebe e M. Worring (eds.), *Proceedings of the ACM International Conference on Image and Video Retrieval*, p. 371–378, Amsterdam, Holanda, Julho 9–11 2007. ACM Press.

[99] D.-D. Le e S. Satoh. National institute of informatics, japan at TRECVID 2007: BBC rushes summarization. Em P. Over e A. F. Smeaton (eds.), *Proceedings of the ACM International Workshop on Video Summarization*, p. 70–73, Bavaria, Alemanha, Setembro 28 2007. ACM Press.

[100] Y. Lecun, L. Bottou, Y. Bengio e P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Novembro 1998.

[101] S.-W. Lee, Y.-M. Kim e S. W. Choi. Fast scene change detection using direct feature extraction from MPEG compressed videos. *IEEE Transactions on Multimedia*, 2 (4):240–254, Dezembro 2000.

[102] B. Leibe e B. Schiele. Analyzing appearance and contour based methods for object categorization. Em *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, p. 409–415, Madison, WI, EUA, Junho 16–22 2003. IEEE Computer Society.

[103] C. Li, E. Y. Chang, H. Garcia-Molina e G. Wiederhold. Clustering for approximate similarity search in high-dimensional spaces. *IEEE Transactions on Knowledge and Data Engineering*, 14(4):792–808, Julho 2002.

[104] R. Lienhart. Reliable transition detection in videos: A survey and practitioner's guide. *International Journal of Image and Graphics*, 1(3):469–486, Julho 2001.

[105] R. Lienhart, C. Kuhmünch e W. Effelsberg. On the detection and recognition of television commercials. Em *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, p. 509–516, Ottawa, ON, Canadá, Junho 3–6 1997. IEEE Computer Society.

[106] R. Lienhart, W. Effelsberg e R. Jain. VisualGREP: A systematic method to compare and retrieve video sequences. *Multimedia Tools and Applications*, 10(1):47–72, Janeiro 2000.

[107] X. Liu, T. Mei, X.-S. Hua, B. Yang e H.-Q. Zhou. Video collage. Em R. Lienhart, A. R. Prasad, A. Hanjalic, S. Choi, B. P. Bailey e N. Sebe (eds.), *Proceedings of the ACM International Conference on Multimedia*, p. 461–462, Augsburgo, Alemanha, Setembro 24–29 2007. ACM Press.

[108] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Novembro 2004.

[109] Y. Ma, S. Soatto, J. Kosecka e S. S. Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models*. Springer-Verlag, Inc., Nova Iorque, NY, EUA, 2003.

[110] B. S. Manjunath, J.-R. Ohm, V. V. Vasudevan e A. Yamada. Color and texture descriptors. *IEEE Transactions on Circuits Systems and Video Technology*, 11(6): 703–715, Junho 2001.

[111] J. Martin e J. L. Crowley. Comparison of correlation techniques. Em U Rembold, R. Dillmann, L. O. Hertzberger e T.Kanade (eds.), *Proceedings of the International Conference on Intelligent Autonomous Systems*, p. 86–93, Karlsruhe, Alemanha, Março 27–30, 1995. IOS Press.

[112] S. Mattoccia, F. Tombari e L. Stefano. Reliable rejection of mismatching candidates for efficient ZNCC template matching. Em *Proceedings of the IEEE International Conference on Image Processing*, p. 849–852, São Diego, CA, EUA, Outubro 12–15 2008. IEEE Computer Society.

[113] K. Mc Donald. *Discrete Language Models for Video Retrieval*. Tese de Doutorado, School of Computing, Dublin City University, Dublin, Irlanda, Setembro 2005.

[114] R. Minetto. Detecção robusta de movimento de câmera em vídeos por análise de fluxo ótico ponderado. Dissertação de Mestrado, Instituto de Computação, UNICAMP, Campinas, SP, Brasil, Agosto 2007.

[115] R. Minetto, N. J. Leite e J. Stolfi. Reliable detection of camera motion based on weighted optical flow fitting. Em A. Ranchordas, H. Araújo e J. Vitrià (eds.), *Proceedings of the International Conference on Computer Vision Theory and Applications*, p. 435–440, Barcelona, Espanha, Março 8–11 2007. Institute for Systems and Technologies of Information, Control and Communication.

[116] R. Minetto, N. J. Leite e J. Stolfi. AffTrack: Robust tracking of features in variable-zoom videos. Em *Proceedings of the IEEE International Conference on Image Processing*, p. 4285–4288, Cairo, Egito, Novembro 7–10 2009. IEEE Computer Society.

[117] R. Mohan. Video sequence matching. Em *Proceedings of the IEEE International Conference on Acoustics, Speech e Signal Processing*, p. 3697–3700, Seattle, WA, EUA, Maio 12–15 1998. IEEE Computer Society.

[118] A. G. Money e H. W. Agius. Video summarization: A conceptual framework and survey of the state of the art. *Journal of Visual Communication and Image Representation*, 19(2):121–143, Fevereiro 2008.

[119] P. Mundur, Y. Rao e Y. Yesha. Keyframe-based video summarization using delaunay clustering. *International Journal on Digital Libraries*, 6(2):219–232, Abril 2006.

[120] A. Nagasaka e Y. Tanaka. Automatic video indexing and full-video search for object appearances. Em E. Knuth e L. M. Wegner (eds.), *Proceedings of the IFIP TC2/WG 2.6 Second Working Conference on Visual Database Systems*, v. A-7, *IFIP Transactions*, p. 113–127, Budapeste, Hungria, Setembro 30 – Outubro 3 1991. North-Holland.

[121] M. R. Naphade, R. Wang e T. S. Huang. Multimodal pattern matching for audio-visual query and retrieval. Em M. M. Yeung, C.-S. Li e R. W. Lienhart (eds.), *Proceedings of the SPIE International Conference on Storage and Retrieval for Media Databases*, v. 4676, p. 188–195, São José, CA, EUA, Janeiro 23 2002. SPIE.

[122] G. Navarro. Searching in metric spaces by spatial approximation. *VLDB Journal*, 11(1):28–46, Abril 2002.

[123] M. Ortega, Y. Rui, K. Chakrabarti, K. Porkaew, S. Mehrotra e T. S. Huang. Supporting ranked boolean similarity queries in MARS. *IEEE Transactions on Knowledge and Data Engineering*, 10(6):905–925, Novembro 1998.

[124] P. Over, A. F. Smeaton e P. Kelly. The TRECVID 2007 BBC rushes summarization evaluation pilot. Em P. Over e A. F. Smeaton (eds.), *Proceedings of the ACM International Workshop on Video Summarization*, p. 1–15, Bavaria, Alemanha, Setembro 28 2007. ACM Press.

[125] P. Over, A. F. Smeaton e G. Awad. The TRECVID 2008 BBC rushes summarization evaluation. Em P. Over e A. F. Smeaton (eds.), *Proceedings of the ACM International Workshop on Video Summarization*, p. 1–20, Vancouver, BC, Canadá, Outubro 31 2008. ACM Press.

[126] C.-M. Pan, Y.-Y. Chuang e W. H. Hsu. NTU TRECVID-2007 fast rushes summarization system. Em P. Over e A. F. Smeaton (eds.), *Proceedings of the ACM International Workshop on Video Summarization*, p. 74–78, Bavaria, Alemanha, Setembro 28 2007. ACM Press.

[127] S.-C. Park, H.-S. Lee e S.-W. Lee. Qualitative estimation of camera motion parameters from the linear composition of optical flow. *Pattern Recognition*, 37(4): 767–779, Abril 2004.

[128] G. Paschos. Perceptually uniform color spaces for color texture analysis: an empirical evaluation. *IEEE Transactions on Image Processing*, 10(6):932–937, Junho 2001.

[129] S.-C. Pei e Y.-Z. Chou. Efficient MPEG compressed video analysis using macroblock type information. *IEEE Transactions on Multimedia*, 1(4):321–333, Dezembro 1999.

[130] S. Pfeiffer, R. Lienhart, G. Kühne e W. Effelsberg. The MoCA project - movie content analysis research at the university of mannheim. Em J. Dassow e R. Kruse (eds.), *GI Jahrestagung*, p. 329–338, Magdeburgo, Alemanha, Setembro 21–25 1998. Springer.

[131] M. J. Pickering, D. Heesch, S. M. Rüger, R. O'Callaghan e D. R. Bull. Video retrieval using global features in keyframes. Em *Proceedings of the Text REtrieval Conference*. National Institute of Standards and Technology, 2002. URL `http://trec.nist.gov/pubs/trec11/papers/imperial.pickering.pdf`.

[132] S. M. Pinto-Cáceres, J. Almeida, V. P. A. Neris, M. C. C. Baranauskas, N. J. Leite e R. da S. Torres. Navigating through video stories using clustering sets. *International Journal of Multimedia Data Engineering and Management*, 2(3):1–20, Setembro 2011.

[133] D. B. Ponceleon, S. Srinivasan, A. Amir, D. Petkovic e D. Diklic. Key to effective video retrieval: Effective cataloging and browsing. Em *Proceedings of the ACM International Conference on Multimedia*, p. 99–107, Bristol, Inglaterra, Setembro 12–16 1998. ACM Press.

[134] N. Putpuek, D.-D. Le, N. Cooharojananone, S. Satoh e C. Lursinsap. Rushes summarization using different redundancy elimination approaches. Em P. Over e A. F. Smeaton (eds.), *Proceedings of the ACM International Workshop on Video Summarization*, p. 100–104, Vancouver, BC, Canadá, Outubro 31 2008. ACM Press.

[135] B. Qi, M. Ghazal e A. Amer. Robust global motion estimation oriented to video object segmentation. *IEEE Transactions on Image Processing*, 17(6):958–967, Junho 2008.

[136] R. Ramakrishnan e J. Gehkre. *Database Management Systems*. McGraw-Hill Co., Inc., Nova Iorque, NY, EUA, 2003.

[137] M. Rautiainen e D. S. Doermann. Temporal color correlograms for video retrieval. Em *Proceedings of the IEEE International Conference on Pattern Recognition*, p. 267–270, Quebec, QC, Canadá, Agosto 11–15 2002. IEEE Computer Society.

[138] W. Ren, S. Singh, M. Singh e Y. S. Zhu. State-of-the-art on spatio-temporal information-based video retrieval. *Pattern Recognition*, 42(2):267–282, Fevereiro 2009.

[139] A. Rocha, J. Almeida, M. A. Nascimento, R. da S. Torres e S. Goldenstein. Efficient and flexible cluster-and-search approach for cbir. Em J. Blanc-Talon, S. Bourennane, W. Philips, D. Popescu e P. Scheunders (eds.), *Proceedings of the International Conference on Advanced Concepts for Intelligent Vision Systems*, v. 5259, *Lecture Notes in Computer Science*, p. 77–88, Juan-les-Pins, França, Outubro 20–24 2008. Springer.

[140] P. J. Rousseeuw e A. M. Leroy. *Robust Regression and Outlier Detection*. John Wiley and Sons, Inc., Nova Iorque, NY, EUA, 1987.

[141] Y. Rubner, C. Tomasi e L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, Novembro 2000.

[142] P. Sand e S. J. Teller. Particle video: Long-range motion estimation using point trajectories. *International Journal of Computer Vision*, 80(1):72–91, Outubro 2008.

[143] S. Santini e R. Jain. Similarity measures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(9):871–883, Setembro 1999.

[144] R. F. Santos Filho, C. Traina Jr., A. J. M. Traina, M. R. Vieira e C. Faloutsos. The omni-family of all-purpose access methods: a simple and effective way to make similarity search more efficient. *VLDB Journal*, 16(4):483–505, Outubro 2007.

[145] M. Shyu, S. Chen, M. Chen e C. Zhang. A unified framework for image database clustering and content-based retrieval. Em S. Chen e M. Shyu (eds.), *Proceedings of the ACM International Workshop on Multimedia Databases*, p. 19–27, Washington, DC, EUA, Novembro 13 2004. ACM Press.

[146] A. F. Smeaton. Techniques used and open challenges to the analysis, indexing and retrieval of digital video. *Information Systems*, 32(4):545–559, Junho 2007.

[147] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta e 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, Dezembro 2000.

[148] J. R. Smith, S. Srinivasan, A. Amir, S. Basu, G. Iyengar, C.-Y. Lin, M. R. Naphade, D. B. Ponceleon e B. L. Tseng. Integrating features, models, and semantics for trec video retrieval. Em *Proceedings of the Text REtrieval Conference*. National Institute of Standards and Technology, 2001. URL `http://trec.nist.gov/pubs/trec10/papers/IBM-TREC-VIDEO-2001.pdf`.

[149] A. Smolic, M. Hoeynck e J.-R. Ohm. Low-complexity global motion estimation from p-frame motion vectors for mpeg-7 applications. Em *Proceedings of the IEEE International Conference on Image Processing*, p. 271–274, Vancouver, BC, Canadá, Setembro 10–13 2000. IEEE Computer Society.

[150] C. Snoek e M. Worring. Multimodal video indexing: A review of the state-of-the-art. *Multimedia Tools and Applications*, 25(1):5–35, Janeiro 2005.

[151] M. V. Srinivasan, S. Venkatesh e R. Hosie. Qualitative estimation of camera motion parameters from video sequences. *Pattern Recognition*, 30(4):593–606, Abril 1997.

[152] L. Stefano, S. Mattoccia e F. Tombari. ZNCC-based template matching using bounded partial correlation. *Pattern Recognition Letters*, 26(14):2129–2134, Outubro 2005.

[153] R. O. Stehling. *Recuperação por Conteúdo em Grandes Coleções de Imagens Heterogêneas*. Tese de Doutorado, Instituto de Computação, UNICAMP, Campinas, SP, Brasil, Outubro 2002.

[154] R. O. Stehling, M. A. Nascimento e A. X. Falcão. Techniques for color-based image retrieval. Em C. Djeraba (ed.), *Multimedia Mining - A Highway to Intelligent Multimedia Document*, v. 22 *Multimedia Systems and Applications*, cap. 4, p. 61–80. Kluwer Academic Publishers, Norwell, MA, EUA, 2002.

[155] M. Sugano, Y. Nakajima e H. Yanagihara. Automated MPEG audio-video summarization and description. Em *Proceedings of the IEEE International Conference on Image Processing*, p. 956–959, Rochester, NY, EUA, Setembro 22–25 2002. IEEE Computer Society.

[156] C. Sun. Fast stereo matching using rectangular subregioning and 3D maximum-surface techniques. *International Journal of Computer Vision*, 47(1-3):99–117, - Junho 2002.

[157] C. Sun. Fast optical flow using 3D shortest path techniques. *Image and Vision Computing*, 20(13-14):981–991, Dezembro 2002.

[158] M. J. Swain e B. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, Novembro 1991.

[159] Y.-P. Tan, D. D. Saur, S. R. Kulkarni e P. J. Ramadge. Rapid estimation of camera motion from compressed video with application to video annotation. *IEEE Transactions on Circuits Systems and Video Technology*, 10(1):133–146, Fevereiro 2000.

[160] F. Tiburzi e J. Bescos. Camera motion analysis in on-line MPEG sequences. Em *Proceedings of the IEEE International Workshop on Image Analysis for Multimedia Interactive Services*, p. 42–45, Santorini, Grécia, Junho 6–8 2007. IEEE Computer Society.

[161] R. da S. Torres e A. X. Falcão. Content-based image retrieval: Theory and applications. *Revista de Inforática Teórica e Aplicada*, 13(2):161–185, Edição Especial 2006.

[162] C. Traina Jr., A. J. M. Traina, C. Faloutsos e B. Seeger. Fast indexing and visualization of metric data sets using slim-trees. *IEEE Transactions on Knowledge and Data Engineering*, 14(2):244–260, Março 2002.

[163] B. T. Truong e S. Venkatesh. Video abstraction: A systematic review and classification. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 3(1):1–37, Fevereiro 2007.

[164] D.-M. Tsai, C.-T. Lin e J.-F. Chen. The evaluation of normalized cross correlations for defect detection. *Pattern Recognition Letters*, 24(15):2525–2535, Novembro 2003.

[165] J. K. Uhlmann. Satisfying general proximity/similarity queries with metric trees. *Information Processing Letters*, 40(4):175–179, Novembro 1991.

[166] M. R. Vieira, C. Traina Jr., F. J. T. Chino e A. J. M. Traina. DBM-tree: Trading height-balancing for performance in metric access methods. *Journal of the Brasilian Computer Society*, 11(3):37–52, Abril 2006.

[167] A. Whitehead, P. Bose e R. Laganière. Feature based cut detection with automatic threshold selection. Em *Proceedings of the ACM International Conference on Image and Video Retrieval*, v. 3115, *Lecture Notes in Computer Science*, p. 410–418, Dublin, Irlanda, Julho 21–23 2004. Springer.

[168] L. Wu, Y. Guo, X. Qiu, Z. Feng, J. Rong, W. Jin, D. Zhou, R. Wang e M. Jin. Fudan university at TRECVID 2003. Em *Proceedings of TRECVid*. National Institute of Standards and Technology, 2003. URL `http://www-nlpir.nist.gov/projects/tvpubs/tvpapers03/fudan.final2.paper.pdf`.

[169] B.-L. Yeo e B. Liu. Rapid scene analysis on compressed video. *IEEE Transactions on Circuits Systems and Video Technology*, 5(6):533–544, Dezembro 1995.

[170] P. N. Yianilos. Data structures and algorithms for nearest neighbor search in general metric spaces. Em *Proceedings of the ACM/SIGACT-SIAM International Symposium on Discrete Algorithms*, p. 311–321, Austin, TX, EUA, Janeiro 25–27 1993. ACM/SIAM.

[171] D. Yu e A. Zhang. Clustertree: Integration of cluster representation and nearest-neighbor search for large data sets with high dimensions. *IEEE Transactions on Knowledge and Data Engineering*, 15(5):1316–1337, Setembro 2003.

[172] J. C. S. Yu, M. S. Kankanhalli e P. Mulhen. Semantic video summarization in compressed domain MPEG video. Em *Proceedings of the IEEE International Conference on Multimedia and Expo*, p. 329–332, Baltimore, MD, EUA, Julho 6–9 2003. IEEE Computer Society.

[173] W. Zeng, W. Gao e D. Zhao. Video indexing by motion activity maps. Em *Proceedings of the IEEE International Conference on Image Processing*, p. 912–915, Rochester, NY, EUA, Setembro 22–25 2002. IEEE Computer Society.

[174] P. Zezula, G. Amato, V. Dohnal e M. Batko. *Similarity Search: The Metric Space Approach*. Springer-Verlag, Inc., Secaucus, NJ, EUA, 2005.

[175] H. Zhang, A. Kankanhalli e S. W. Smoliar. Automatic partitioning of full-motion video. *Multimedia Systems*, 1(1):10–28, Junho 1993.

[176] H. Zhang, C. Y. Low e S. W. Smoliar. Video parsing and browsing using compressed data. *Multimedia Tools and Applications*, 1(1):89–111, Março 1995.

[177] H. Zhang, J. Wu, D. Zhong e S. W. Smoliar. An integrated system for content-based video retrieval and browsing. *Pattern Recognition*, 30(4):643–658, Abril 1997.

[178] T. Zhang e C. Tomasi. Fast, robust, and consistent camera motion estimation. Em *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, p. 1164–1170, Ft. Collins, CO, EUA, Junho 23–25 1999. IEEE Computer Society.

[179] J. Zhou e X.-P. Zhang. Automatic identification of digital video based on shot-level sequence matching. Em H. Zhang, T.-S. Chua, R. Steinmetz, M. S. Kankanhalli e L. Wilcox (eds.), *Proceedings of the ACM International Conference on Multimedia*, p. 515–518, Cingapura, Novembro 6–11 2005. ACM Press.

[180] Y. Zhuang, Y. Rui, T. S. Huang e S. Mehrotra. Adaptive key frame extraction using unsupervised clustering. Em *Proceedings of the IEEE International Conference on Image Processing*, p. 866–870, Chicago, IL, EUA, Outubro 4–7 1998. IEEE Computer Society.