Recuperação por Conteúdo em Grandes Coleções de Imagens Heterogêneas

Renato de Oliveira Stehling

Tese de Doutorado

Instituto de Computação
Universidade Estadual de Campinas

Recuperação por Conteúdo em Grandes Coleções de Imagens Heterogêneas

Renato de Oliveira Stehling[1]

Outubro de 2002

Banca Examinadora:

- Prof. Dr. Alexandre Xavier Falcão (Orientador)
- Profa. Dra. Agma Juci Machado Traina
  Instituto de Ciências Matemáticas e de Computação – USP
- Profa. Dra. Ana Carolina Salgado
  Centro de Informática – UFPE
- Profa. Dra. Cláudia Bauzer Medeiros
  Instituto de Computação – UNICAMP
- Prof. Dr. Neucimar J. Leite
  Instituto de Computação – UNICAMP
- Profa. Dra. Maria Beatriz Felgar de Toledo (Suplente)
  Instituto de Computação – UNICAMP
- Prof. Dr. Caetano Traina Júnior (Suplente)
  Instituto de Ciências Matemáticas e de Computação – USP

[1]**Projeto financiado pela FAPESP, processo 98/12899-6**

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP

Stehling, Renato de Oliveira

St32r Recuperação por conteúdo em grandes coleções de imagens heterogêneas / Renato de Oliveira Stehling – Campinas, [S.P. :s.n.], 2002.

Orientador : Alexandre Xavier Falcão

Co-orientador: Mario Antonio do Nascimento

Tese (doutorado) - Universidade Estadual de Campinas, Instituto de Computação.

1. Processamento de imagens. 2. Recuperação da informação. 3. Banco de dados. I. Falcão, Alexandre Xavier. II. Nascimento, Mario Antonio do. III. Universidade Estadual de Campinas. Instituto de Computação. IV. Título.

TERMO DE APROVAÇÃO

Tese defendida e aprovada em 10 de outubro de 2002, pela Banca Examinadora composta pelos Professores Doutores:

Profa. Dra. Ana Carolina Brandão Salgado
UFPe

Profa. Dra. Agma Juci Machado Traina
ICMSC - USP

Profa. Dra. Claudia Maria Bauzer Medeiros
IC - UNICAMP

Prof. Dr. Neucimar Jerônimo Leite
IC - UNICAMP

Prof. Dr. Alexandre Xavier Falcão
IC - UNICAMP

Recuperação por Conteúdo em Grandes Coleções de Imagens Heterogêneas

Este exemplar corresponde à redação final da Tese devidamente corrigida e defendida por Renato de Oliveira Stehling e aprovada pela Banca Examinadora.

Campinas, 10 de outubro de 2002.

Prof. Dr. Alexandre Xavier Falcão
(Orientador)

Prof. Dr. Mário A. Nascimento
(Co-orientador)

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

Para Célia e Rogério

Agradecimentos

A Deus, que nos direcionou e protegeu durante todo esse período.

À minha esposa Eliana e ao meu filho Matheus, para quem e por quem estamos sempre buscando novas conquistas, e de quem temos recebido todo o amor, carinho e incentivo necessários para continuar essa busca.

A toda minha família, sobretudo meus pais (Célia e Rogério), pelo incentivo constante, pelo apoio irrestrito e por seus valiosos conselhos.

Ao Prof. Alexandre Falcão, que nos deu um voto de confiança ao aceitar nos orientar em um projeto já em andamento, e que conduziu esse projeto sempre com muito interesse, coerência e competência.

Ao Prof. Mário Nascimento, com quem iniciamos esse projeto e que, apesar de sua transferência para o Canadá, continuou participando ativamente de todas as atividades realizadas. Agradeço também pelos momentos agradáveis que ele e sua família nos proporcionaram durante o período em que parte desse projeto foi desenvolvido sob sua supervisão na Universidade de Alberta, Canadá.

A todos os amigos que direta ou indiretamente contribuiram para o sucesso desse trabalho.

Aos professores e funcionários do Instituto de Computação, pelos serviços prestados e pela disposição em ajudar.

À FAPESP, pelo apoio financeiro (processo 98/12899-6).

Ao projeto PRONEX SAI, pelo auxílio financeiro recebido.

Resumo

A recuperação de imagens por conteúdo (CBIR) é uma área que vem recebendo crescente atenção por parte da comunidade científica por causa do crescimento exponencial do número de imagens que vêm sendo disponibilizadas, principalmente na WWW. À medida que cresce o volume de imagens armazenadas, cresce também o interesse por sistemas capazes de recuperar eficientemente essas imagens a partir do seu conteúdo visual.

Nosso trabalho concentrou-se em técnicas que pudessem ser aplicadas em grandes coleções de imagens heterogêneas. Nesse tipo de coleção, não se pode assumir nenhum tipo de conhecimento sobre o conteúdo semântico e/ou visual das imagens, e o custo de utilizar técnicas semi-automáticas (com intervenção humana) é alto em virtude do volume e da heterogeneidade das imagens que precisam ser analisadas. Nós nos concentramos na informação de cor presente nas imagens, e enfocamos os três tópicos que consideramos mais importantes para se realizar a recuperação de imagens baseada em cor: (1) como analisar e extrair informação de cor das imagens de forma automática e eficiente; (2) como representar essa informação de forma compacta e efetiva; e (3) como comparar eficientemente as características visuais que descrevem duas imagens.

As principais contribuições do nosso trabalho foram dois algoritmos para a análise automática do conteúdo visual das imagens (*CBC* e *BIC*), duas funções de distância para a comparação das informações extraídas das imagens (*MiCRoM* e *dLog*) e uma representação alternativa para abordagens que decompõem e representam imagens a partir de células de tamanho fixo (*CCH*).

Abstract

Content-based image retrieval (CBIR) is an area that has received increasing attention from the scientific community due to the exponential growing of available images, mainly at the WWW. This has spurred great interest for systems that are able to efficiently retrieve images according to their visual content.

Our work has focused in techniques suitable for broad image domains. In a broad image domain, it is not possible to assume or use any *a priori* knowledge about the visual content and/or semantic content of the images. Moreover, the cost of using semi-automatic image analysis techniques is prohibitive because of the heterogeneity and the amount of images that must be analyzed. We have directed our work to color-based image retrieval, and have focused on the three main issues that should be addressed in order to achieve color-based image retrieval: (1) how to analyze and describe images in an automatic and efficient way; (2) how to represent the image content in a compact and effective way; and (3) how to efficiently compare the visual features extracted from the images.

The main contributions of our work are two algorithms to automatically analyze the visual content of the images (*CBC* and *BIC*), two distance functions to compare the visual features extracted from the images (*MiCRoM* and *dLog*), and an alternative representation for CBIR approaches that decompose and represent images according to a grid of equal-sized cells (*CCH*).

Lista de abreviações

$\theta \times R$ – *Theta vs. Recall*
11P-Precision – *Eleven-Point Precision*
3P-Precision – *Three-Point Precision*
ARG – *Attributed Relational Graph*
BD – Banco de Dados
BDI – Banco de Dados de Imagens
BIC – *Border/Interior Pixel Classification*
CBC – *Color-Based Clustering*
CBIR – *Content-Based Image Retrieval*
CCH – *Cell/Color Histograms*
CCV – *Color-Coherence Vector*
CMM – *Color Moments*
CPU – *Central Processing Unit*
DBMS – *Database Management System*
E/S – Entrada e Saída
FPN – *Floating-Point Number*
GCH – *Global Color Histogram*
HSI – *Hue, Saturation and Intensity*
HSV – *Hue, Saturation and Value*
I/O – *Input and Output*
IR – *Information Retrieval*
IRM – *Integrated Region Matching*
LCH – *Local Color Histogram*
MAM – *Metric Access Method*
MBR – *Minimum Bounding Rectangle*
MST – *Minimal Spanning Tree*
MiCRoM – *Minimum-Cost Region Matching*
NavgR – *Normalized Average Rank*
$P \times R$ – *Precision vs. Recall*

$P(100)$ – *Precision at rank 100*
$P(30)$ – *Precision at rank 30*
$P(r)$ – *Precision at the first point r where recall could be 100%*
QBE – *Query By Example*
RGB – *Red, Green and Blue*
$R(100)$ – *Recall at rank 100*
$R(30)$ – *Recall at rank 30*
RRSet – *Relevant Result Set*
SAM – *Spatial Access Method*
SGBD – Sistema Gerenciador de Banco de Dados
WWW – *World-Wide Web*

Conteúdo

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

Bancos de dados de imagens (BDIs) têm se tornado cada vez mais freqüentes nos mais variados domínios de aplicações, tais como máquinas de busca multimídia [85, 97], bibliotecas digitais [59, 62, 98], bancos de dados geográficos [47, 57, 98] e bancos de dados médicos [49]. A evolução das tecnologias de aquisição, transmissão e armazenamento de imagens tem permitido a construção de BDIs cada vez maiores. À medida que cresce o volume de imagens armazenadas, cresce também o interesse por sistemas capazes de recuperar essas imagens de acordo com o seu conteúdo visual (CBIR - *Content-Based Image Retrieval*) [8, 120]. CBIR é uma área multidisciplinar e envolve, principalmente, técnicas de banco de dados, processamento de imagens, recuperação de informação, reconhecimento de padrões, e interfaces usuário-máquina [65].

A recuperação de imagens baseada em conteúdo baseia-se em descrições compactas das imagens. A descrição das imagens pode acontecer em vários níveis diferentes, e pode ser ou não dependente do domínio das imagens [90]. Em imagens médicas, por exemplo, pode-se extrair informação a respeito das estruturas anatômicas do corpo humano que são conhecidas *a priori* [49, 74]. O mesmo acontece em sistemas de recuperação de faces humanas [9, 77, 116, 117]. Nesses domínios, as características visuais mais relevantes são aquelas que descrevem as relações espaciais entre os objetos que compõem cada imagem.

É possível descrever imagens utilizando-se atributos que são independentes do conteúdo visual das imagens, tais como o nome do arquivo, seu formato gráfico, seu tamanho físico, e as suas dimensões espaciais. Atributos como esses podem ser eficientemente gerenciados por sistemas gerenciadores de banco de dados (SGBDs) [27, 50, 105]. No entanto, a maior dificuldade desse tipo de representação é que as consultas são restritas aos atributos armazenados, e esses atributos não descrevem o conteúdo das imagens.

Uma segunda alternativa para descrever imagens consiste em utilizar palavras-chave e/ou anotações geradas por especialistas acerca do conteúdo das imagens. A descrição textual das imagens pode ser eficientemente gerenciada por sistemas de recuperação de

informação [42, 48, 81, 82, 115, 118]. No entanto, esse tipo de técnica requer intervenção humana para obter as descrições textuais de cada imagem individualmente. Existe ainda o problema da subjetividade e da incompletude na descrição das imagens, já que a interpretação do conteúdo visual de uma imagem varia de acordo com o conhecimento, objetivo, experiência e percepção de cada analisador [34].

Finalmente, é possível utilizar características visuais de baixo nível tais como distribuição de cores [4, 30, 33, 73, 96, 107], textura [63, 70, 86, 95, 112, 121], formas [41], posição e relações topológicas entre regiões da imagem [19, 53, 78] para descrever, representar e comparar imagens. Em geral, essas características visuais são representadas como vetores k-dimensionais. As relações espaciais entre os objetos/regiões (por exemplo, adjacência, sobreposição, e relações de inclusão) têm sido representadas através de *2D-strings* [19, 53] ou ARGs (*Attributed Relational Graphs*) [78].

A descrição de imagens utilizando características visuais de baixo nível é especialmente útil em grandes coleções de imagens heterogêneas [92]. Consideramos heterogêneo um conjunto de imagens que não pertence a um único domínio semântico e/ou visual, ou seja, tanto a semântica associada às imagens quanto as suas características visuais não seguem um padrão preestabelecido. A *World-Wide Web* (WWW) é o melhor exemplo de um gigantesco repositório de imagens heterogêneas. Em coleções desse tipo, não é possível preestabelecer nenhuma característica das imagens armazenadas e são necessárias técnicas automáticas e eficientes para a análise, representação e comparação dessas imagens. O custo de utilizar técnicas semi-automáticas (com intervenção humana) é alto em virtude da heterogeneidade e do volume das imagens que precisam ser analisadas.

Dentre as características visuais de baixo nível que podem ser utilizadas na recuperação de imagens baseada em conteúdo, a informação de cor é uma das mais amplamente utilizadas [13, 61]. Essa preferência pela informação de cor se deve a alguns fatores [103]: (1) a cor é uma característica visual que é imediatamente percebida quando se olha para uma imagem; (2) os conceitos envolvidos são simples de serem entendidos e implementados; (3) a informação de cor está presente na ampla maioria dos domínios de imagens e (4) os resultados obtidos utilizando a informação de cor são satisfatórios em geral; (4) a informação de cor pode ser processada de forma automática.

Apesar da importância de descrever imagens em diferentes níveis e utilizando diferentes características visuais, nosso trabalho enfocou exclusivamente a informação de cor e técnicas para lidar com esse tipo de informação, já que a utilização da informação de cor é importante na maioria dos sistemas de recuperação de imagens por conteúdo.

A Figura 2.1 mostra a representação esquemática de uma imagem sendo armazenada em um sistema que faz uso da informação de cor para descrever, representar, comparar e recuperar imagens. Após uma imagem ser fornecida como entrada, o seu conteúdo visual é analisado e resumido em um espaço de cores preestabelecido. Em seguida, uma repre-

sentação é escolhida para a informação extraída durante a etapa de análise da imagem. Essa representação é o que na prática denominamos uma característica visual da imagem. A característica visual que representa a imagem de entrada é então armazenada no BDI e indexada com o objetivo de reduzir o tempo de busca quando uma consulta visual for processada.

A recuperação de imagens armazenadas em um BDI é um processo interativo que envolve diversos tipos de consulta, em diferentes etapas. Em geral, o objetivo das consultas visuais é encontrar e recuperar imagens que sejam similares a uma imagem/esboço fornecido pelo usuário, ou seja, as consultas não se baseiam em correspondência exata (*matching*) como acontece em bancos de dados tradicionais. Existem basicamente dois tipos de consultas visuais [93]: (1) encontrar as k imagens mais próximas de uma dada imagem e (2) encontrar as imagens que estão acima de um limite de similaridade l de uma dada imagem.

Em nosso trabalho, estamos interessados na informação de cor (característica visual de baixo nível) presente nas imagens. Esse tipo de informação é difícil de ser expressa pelos usuários durante a formulação de uma consulta visual. Por causa disso, nos concentramos no paradigma de consultas-por-exemplo (QBE), isto é, em consultas nas quais uma imagem consulta é fornecida pelo usuário como exemplo das imagens desejadas e o sistema retorna as imagens do BDI classificadas em ordem decrescente de similaridade em relação a essa imagem consulta. Para comparar a (dis)similaridade entre duas imagens, é necessário uma função que calcule a distância entre as características visuais que representam essas imagens. A função de distância também é determinante na escolha de uma estrutura de indexação que acelere o processamento de consultas, já que cada estrutura impõe suas próprias restrições ao tipo de função que é capaz de indexar.

De acordo com o cenário descrito acima, nós consideramos a existência de quatro tópicos chave que precisam ser explorados para que se realize a recuperação automática de imagens baseada em informação de cor: (1) como analisar e extrair informação de cor das imagens de forma automática e eficiente; (2) como representar essa informação de forma compacta e efetiva; (3) como comparar de maneira efetiva e eficiente as características visuais que descrevem duas imagens; e (4) como indexar de forma adequada essas características visuais para reduzir ao máximo o tempo de busca quando uma consulta visual é processada. Em nosso trabalho nós enfocamos os três primeiros tópicos acima. Apesar de não termos explorado formalmente a indexação das características visuais extraídas das imagens, os requisitos para uma indexação eficiente foram uma preocupação constante em todas as técnicas que propusemos.

Finalmente, para avaliar quantitativamente o desempenho de um sistema de recuperação de informação, são necessárias medidas de eficiência e efetividade [48, 115, 118]. As medidas de eficiência estão relacionadas aos custos (em termos de recursos computa-

cionais) para realizar um conjunto de tarefas e são, de certa forma, independentes do tipo de documento que está sendo recuperado. As medidas de efetividade, ao contrário, estão intimamente relacionadas ao tipo de documento que está sendo recuperado e aos critérios de avaliação desses documentos, pois se preocupam em medir a capacidade de um sistema fornecer adequadamente as informações requeridas pelo usuário [48, 82, 118]. Por se tratar de um tipo de recuperação de informação bastante específico, a recuperação de imagens por conteúdo requer uma metodologia de avaliação também específica, de acordo com as restrições desse domínio.

O restante desse capítulo está organizado como se segue. A Seção 1.1 introduz alguns conceitos importantes sobre imagem digital e o processamento de imagens digitais. Na Seção 1.2, são identificadas as principais técnicas para a análise e a representação do conteúdo visual de imagens em sistemas de recuperação de imagens baseados em informação de cor. A Seção 1.3 discute aspectos relacionados ao processamento de consultas visuais tais como a comparação e a indexação de características visuais extraídas das imagens. Os critérios e as metodologias existentes para a avaliação da efetividade de sistemas de recuperação de imagens são identificados na Seção 1.4. Uma classificação para as técnicas e sistemas existentes para CBIR é proposta na Seção 1.5. A Seção 1.6 identifica as principais contribuições de nosso trabalho. Finalmente, a organização dos demais capítulos da tese é detalhada na Seção 1.7.

1.1 Imagem digital

Uma imagem (monocromática) é uma função bidimensional $f(x, y)$, onde x e y são coordenadas espaciais e o valor de f em qualquer ponto (x, y) é proporcional ao brilho (ou nível de cinza) da imagem nesse ponto [37]. Uma imagem digital nada mais é que uma imagem $f(x, y)$ que teve tanto as suas coordenadas espaciais quanto o seu brilho discretizados (digitalizados). Dessa forma, uma imagem digital pode ser interpretada como uma matriz onde cada elemento é identificado pelos índices da linha e da coluna às quais pertence, e o valor do elemento corresponde ao seu brilho ou nível de cinza. Os elementos dessa matriz são conhecidos como pixels (*picture elements*).

A digitalização das coordenadas espaciais é conhecida como amostragem (*image sampling*) e a digitalização do brilho é conhecida como quantização do nível de cinza (*gray-level quantization*) [37]. 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 aproxima o conteúdo da imagem original. No entanto, os custos de armazenamento e de processamento da imagem digital crescem rapidamente com o aumento da resolução.

No caso de imagens digitais coloridas, cada pixel é descrito não apenas pelo seu brilho,

mas também por outras propriedades como matiz e saturação. Em geral, a cor de cada pixel é representada como um ponto em um sistema de coordenadas 3D conhecido como espaço de cores. Alguns espaços de cores são discutidos na Seção 2.1. Um exemplo de espaço de cores 3D é o espaço RGB (*Red, Green, Blue*), onde cada cor é representada como uma combinação de três cores primárias (vermelho, verde e azul).

Um pixel p com coordenadas espacias (x, y) possui quatro vizinhos no espaço (horizontais e verticais) cujas coordenadas são:

$$N_4(p) = \{(x+1, y), (x-1, y), (x, y+1), (x, y-1)\} \tag{1.1}$$

Adicionalmente, é possível definir outros quatro vizinhos (nas diagonais):

$$N_D(p) = \{(x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)\} \tag{1.2}$$

A união dos dois conjuntos anteriores determina um total de oito vizinhos para o pixel p:

$$N_8(p) = N_4(p) \cup N_D(p) \tag{1.3}$$

Um caminho entre dois pixels p e q cujas coordenadas espacias são (x, y) e (s, t) é uma seqüência de pixels distintos com coordenadas:

$$(x_0, y_0), (x_1, y_1), ..., (x_n, y_n) \tag{1.4}$$

onde $(x_0, y_0) = (x, y)$ e $(x_n, y_n) = (s, t)$, (x_i, y_i) é adjacente a (x_{i-1}, y_{i-1}) de acordo com algum critério de adjacência (por exemplo, são vizinhos considerando-se 4 vizinhos por pixel), $0 \leq i \leq n$, e n é o tamanho do caminho [37].

Se p e q são pixels que pertencem a um subconjunto S de pixels da imagem, então p é conexo a q em S se existe um caminho entre p e q formado apenas por pixels que pertencem a S. Para qualquer pixel $p \in S$, o conjunto de todos os pontos que são conexos a p em S é conhecido como uma componente conexa de S. Como conseqüência, dois pixels quaisquer de uma mesma componente conexa são conexos entre si e duas componentes conexas diferentes são disjuntas [37].

1.1.1 Processamento de imagens digitais

O processamento de imagens digitais é uma área que envolve aspectos de *hardware*, *software* e vários conceitos teóricos [37]. O processamento de uma imagem digital pode ser subdividido em cinco passos principais: (1) aquisição da imagem, (2) pré-processamento, (3) segmentação, (4) representação e descrição e (5) reconhecimento e interpretação.

O primeiro passo do processo consiste em capturar a imagem digital. Para isso são necessários sensores (por exemplo, uma câmera de vídeo) e, caso o sinal produzido pelo sensor seja analógico, um conversor analógico-digital para digitalizar esse sinal.

O próximo passo consiste em pré-processar a imagem digital capturada na etapa anterior. O objetivo do pré-processamento é melhorar a qualidade da imagem em aspectos que permitam elevar a chance de sucesso das etapas seguintes.

O terceiro passo refere-se à segmentação da imagem. O objetivo da segmentação é particionar a imagem em suas partes constituintes ou objetos. A qualidade da segmentação é decisiva para o sucesso das etapas posteriores.

Uma vez segmentadas, as regiões da imagem precisam ser representadas em uma forma adequada ao processamento por computador. Nesse caso, a primeira decisão diz respeito à representação ou da borda ou do conteúdo das regiões. Cada representação é adequada a um tipo específico de processamento. Também é necessário fazer uma *descrição* das regiões com o objetivo de destacar características visuais de interesse para as etapas seguintes. A descrição (ou extração de características visuais) refere-se à extração de características que sejam úteis para diferenciar diferentes classes de objetos.

A última etapa do processo refere-se ao reconhecimento e interpretação de objetos. O reconhecimento é um processo que identifica objetos a partir das informações extraídas de seus descritores. A interpretação, por sua vez, consiste em associar significado a um conjunto de objetos previamente identificados.

É importante observar que todas as cinco etapas descritas anteriormente utilizam informações sobre o domínio do problema que está sendo tratado e também fornecem novas informações acerca desse domínio, do processo em andamento, e da imagem sendo processada. Todo esse conhecimento é codificado e armazenamento em uma base de dados que está disponível ao longo de todas as etapas do processamento da imagem.

1.2 Análise e representação do conteúdo visual das imagens

A primeira decisão a ser tomada ao projetar um sistema de recuperação de imagens por conteúdo utilizando informação de cor refere-se à escolha do espaço de cores a ser utilizado. Um espaço de cores é um sistema de coordenadas 3D onde cada cor é representada por um ponto nesse espaço tridimensional [37]. Os espaços de cores existentes podem ser classificados em três grandes categorias [13, 37, 61] (1) orientados ao *hardware*, (2) orientados ao usuário e (3) uniformes. Cada uma dessas categorias tem um contexto de aplicação bastante específico. Os espaços de cores existentes são discutidos em detalhes na Seção 2.1. Em particular, são discutidos os espaços RGB (orientado ao *hardware*),

HSV (orientado ao usuário) e LAB (uniforme).

Uma vez que o espaço de cores foi escolhido, a segunda decisão em qualquer sistema para CBIR consiste em reduzir o número de cores presente nas imagens (quantização) e também o número de posições espaciais que precisam ser consideradas para se descrever a distribuição espacial das cores dentro das imagens (amostragem). As técnicas existentes para esse tipo de redução podem ser classificadas em estáticas ou dinâmicas. As técnicas estáticas utilizam esquemas preestabelecidos que são independentes da imagem sendo analisada. Exemplos de técnicas estáticas são a quantização uniforme do espaço de cores [13, 61] e a decomposição espacial das imagens em células de tamanho fixo [94, 104, 113]. As técnicas dinâmicas fazem uso do conteúdo das imagens para obter uma redução de informação maior e mais robusta que a das técnicas estáticas. As técnicas dinâmicas realizam simultaneamente a redução do número de cores e de posições espaciais através de algoritmos de agrupamento (*clustering*) [5, 31, 45] ou de segmentação de imagens [37]. Tanto as técnicas estáticas quanto as técnicas dinâmicas para a simplificação do conteúdo visual das imagens são discutidas em detalhes na Seção 2.2.

Uma vez que a informação presente nas imagens foi devidamente reduzida, a terceira decisão refere-se à escolha de uma representação adequada para essa informação. As possíveis representações podem ser classificadas em globais, baseadas em particionamento ou regionais. As representações globais descrevem a distribuição de cores da imagem como um todo, desprezando a distribuição espacial das cores dentro das imagens. As representações baseadas em particionamento assumem que as imagens foram espacialmente decompostas em células de acordo com um esquema preestabelecido e então descrevem a informação de cor de cada célula individualmente. As representações regionais assumem que as imagens foram segmentadas em regiões com tamanho, forma e posição variáveis e descrevem cada uma dessas regiões individualmente. Todas essas representações são discutidas em detalhes na Seção 2.3.

1.3 Processamento de consultas visuais

Um dos componentes mais importantes de um sistema de recuperação de imagens por conteúdo é a função de distância utilizada para comparar as características visuais extraídas das imagens. Essa função afeta diretamente o tempo de processamento de uma consulta visual e a qualidade da resposta obtida (efetividade) [13, 61]. Quanto maior a correlação entre a função de distância e a percepção humana de similaridade, maior será a efetividade do sistema em recuperar imagens relevantes de acordo com os requisitos do usuário. A complexidade computacional da função de distância também é importante pois, dependendo dessa complexidade, é possível que o tempo para comparar as características visuais extraídas de duas imagens (tempo de CPU) seja maior que o tempo gasto para

acessar as páginas de disco onde essas características visuais estão armazenadas (tempo de entrada e saída – E/S) [20, 23, 108]. Ao contrário de sistemas convencionais, o tempo de busca passa a ser dominado pelo tempo de CPU ao invés do tempo de E/S.

A função de distância utilizada para comparar imagens também limita o universo de técnicas de filtragem e métodos de acesso que podem ser utilizados para reduzir o espaço de busca quando uma consulta visual é processada. As técnicas de filtragem são baseadas em uma distância simples que é comprovadamente um limite inferior para a distância original. A nova distância é utilizada para eliminar rapidamente imagens não relevantes, gerando uma lista de imagens candidatas que deve ser pós-processada utilizando a distância original para eliminar falsos-positivos [84]. Os métodos de acesso utilizam combinações mais sofisticadas de estrutura de dados e algoritmos para organizar as características visuais das imagens e gerenciar o processo de busca, de forma que as imagens de interesse possam ser localizadas rapidamente [20, 35].

Distâncias geométricas como L_1 e L_2 podem ser utilizadas em conjunto com métodos de acesso espaciais (SAMs) [35] para reduzir o espaço de busca. Funções mais complexas que satisfazem os axiomas métricos (principalmente à propriedade da desigualdade triangular) podem ser utilizadas em conjunto com métodos de acesso métricos (MAMs) [20] para reduzir simultaneamente o espaço de busca e o número de comparações entre imagens ao processar uma consulta visual. Já a indexação de funções não-métricas é um problema em aberto para o qual existem apenas soluções aproximadas [51, 56, 71].

A Seção 2.4 discute em detalhes as funções de distância utilizadas para comparar as características visuais extraídas das imagens. Essas funções são classificadas em geométricas (vetoriais), métricas e não-métricas. Em particular, são discutidas as funções da família L_p e os axiomas métricos da positividade, simetria, reflexividade e desigualdade triangular. Em seguida, a Seção 2.5 discute o conceito de busca por similaridade e a necessidade de técnicas de filtragem e/ou métodos de acesso para reduzir o tempo de busca quando uma consulta visual é processada. São discutidas algumas técnicas de filtragem, métodos de acesso espaciais (SAMs), métodos de acesso métricos (MAMs) e a indexação aproximada baseada em funções não-métricas.

1.4 Avaliação de efetividade

Quando uma nova abordagem para a recuperação de imagens por conteúdo é proposta, é necessário avaliar seu desempenho. Em sistemas de banco de dados tradicionais, o tempo de resposta e o espaço utilizado para representar os dados são os critérios normalmente utilizados nessa avaliação. No contexto de recuperação de informação, é preciso avaliar, adicionalmente, a relevância da informação recuperada (efetividade) em relação aos requisitos do usuário [48, 82, 115, 118].

A avaliação da efetividade de um sistema de recuperação de informação é uma tarefa bastante complexa. No contexto de recuperação de informação textual, existem várias coleções de documentos utilizadas para se realizar esse tipo de avaliação (CACM, ADI, INSPEC, Medilars e ISI) e uma conferência denominada TREC especialmente dedicada a esse tópico [115, 118]. Como resultado, existem experimentos e medidas padronizadas e um fórum para pesquisadores que queiram comparar os seus resultados utilizando um mesmo *framework*.

Infelizmente, na área de recuperação de imagens por conteúdo (CBIR), o cenário é completamente diferente daquele descrito acima. Em geral, são utilizadas coleções de imagens relativamente pequenas e completamente diferentes entre si. Cada grupo de pesquisadores realiza experimentos baseados em critérios completamente distintos e não há um *benchmark* que seja aceito e amplamente utilizado.

Esforços importantes na direção de criar um *benchmark* para CBIR vêm sendo realizados por Gunther & Beretta [39], Leung & Ip [54] e Muller et al. [66]. Dentre os maiores problemas enfrentados para se obter um *benchmark* desse tipo estão a criação de uma coleção de imagens sem restrições de direitos autorais, o julgamento da relevância dessas imagens em relação a um conjunto de imagens consulta (*ground truth*) e um conjunto de medidas de efetividade apropriadas para a avaliação de CBIR.

A Seção 2.7 discute em detalhes o problema de avaliação de efetividade em sistemas de recuperação de imagens por conteúdo. Em particular, são discutidas várias medidas de efetividade que vêm sendo utilizadas nesse tipo de avaliação.

1.5 Abordagens existentes para a recuperação de imagens por conteúdo

As abordagens existentes para a recuperação de imagens baseada na informação de cor podem ser classificadas em (1) globais [3, 29, 88, 106, 122], (2) baseadas em particionamento [38, 55, 64, 87, 113] e (3) regionais [7, 18, 28, 58]. Essa classificação baseia-se no tipo de representação adotada para as características visuais extraídas das imagens.

Cada uma das três categorias identificadas acima oferece um compromisso distinto entre a complexidade dos algoritmos de análise das imagens, a utilização de espaço em disco para representar essas características, a complexidade da função de distância utilizada para comparar as características visuais extraídas e, finalmente, a efetividade do processo de recuperação das imagens. É importante observar que cada categoria possui características desejáveis e também limitações bem conhecidas. Nenhuma delas é ótima em todas as situações. Na prática nós temos observado que em algumas situações específicas, abordagens globais bastante simples são mais efetivas que abordagens regionais

bastante complexas.

1.5.1 Abordagens globais

As abordagens globais para CBIR [3, 29, 88, 106, 122] descrevem a distribuição de cores das imagens como um todo, desprezando a distribuição espacial dessas cores dentro das imagens. Em geral, essas abordagens são as mais eficientes em termos de extração, representação e comparação das características visuais extraídas das imagens.

A abordagem global mais simples consiste em representar a distribuição de cores de uma imagem através de um histograma de cor global – *GCH* [13, 61]. Esse histograma é obtido contando-se, para cada uma das cores possíveis, o número de pixels da imagem com essas cores. O *GCH* pode ser visto como um vetor k-dimensional, onde k é o número de cores representadas. Esses vetores podem ser eficientemente comparados utilizando-se uma distância vetorial como a distância L_1 (*City-Block*) ou a distância L_2 (Euclideana). Adicionalmente, os *GCHs* podem ser, dependendo de sua dimensionalidade k, eficientemente indexados utilizando-se métodos de acesso espaciais - SAMs. As abordagens globais para a recuperação de imagens por conteúdo são discutidas em detalhes na Seção 2.6.1.

1.5.2 Abordagens baseadas em particionamento

As abordagens para CBIR baseadas em particionamento [38, 55, 64, 87, 113] decompõem espacialmente as imagens utilizando uma estratégia de particionamento simples e comum a toda imagem. Por exemplo, cada imagem é particionada em 3×3 regiões retangulares de mesmo tamanho. A distribuição de cores de cada partição é descrita individualmente. O objetivo do particionamento espacial é adicionar informação de como as cores estão espacialmente distribuídas dentro da imagem. Assim como em abordagens globais, a extração das características visuais é bastante eficiente, com a vantagem de que a informação espacial capturada por essas abordagens aumenta a efetividade em relação às abordagens globais. No entanto, a representação e a comparação das imagens ficam computacionalmente bem mais caras, já que o conteúdo de cada partição é representado e comparado individualmente.

O particionamento espacial mais simples consiste em decompor as imagens de acordo com uma grade de células retangulares e que não se sobrepõem. O conteúdo visual de cada célula é representado por um histograma de cor (nesse caso um Histograma de Cor Local – LCH). Assim como nas abordagens globais, esses histogramas são vetores k-dimensionais que podem ser eficientemente comparados utilizando distâncias vetoriais e indexados utilizando métodos de acesso espaciais (SAMs).

A principal limitação das abordagens baseadas em particionamento é que as imagens são decompostas sem levar em consideração o conteúdo visual das imagens. É possível

que um objeto da imagem seja particionado em várias partes e, ao mesmo tempo, que partes de vários objetos distintos sejam representadas conjuntamente. As abordagens baseadas em particionamento para a recuperação de imagens por conteúdo são discutidas em detalhes na Seção 2.6.2.

1.5.3 Abordagens regionais

As abordagens regionais para CBIR [7, 18, 28, 58] utilizam técnicas automáticas de segmentação para decompor as imagens de acordo com o seu conteúdo visual. O número de regiões obtido, assim como o tamanho, a forma e localização espacial de cada região varia de imagem para imagem. Nesse contexto, o objetivo da segmentação não é, necessariamente, segmentar de maneira precisa todos os objetos presentes em uma imagem, mas a decomposição das imagens em regiões cujos pixels possuem um alto grau de similaridade de acordo com alguma propriedade visual preestabelecida. No entanto, quanto melhor as regiões obtidas representarem os objetos que compõem as imagens, mais efetiva será a abordagem na recuperação de imagens visualmente similares de acordo com a percepção humana de similaridade.

As abordagens regionais utilizam algoritmos complexos (computacionalmente caros) para segmentar imagens e também para comparar duas imagens de acordo com o seu conjunto de regiões. Tanto a segmentação automática de imagens quanto a comparação efetiva e eficiente de imagens segmentadas são problemas bastante difíceis que limitam o potencial das abordagens regionais. No entanto, essas abordagens costumam ser bem mais efetivas que as abordagens globais e as abordagens baseadas em particionamento. As abordagens regionais para a recuperação de imagens por conteúdo são discutidas em detalhes na Seção 2.6.3.

1.6 Contribuições

Esta seção descreve as principais contribuições do nosso trabalho. Cada uma dessas contribuições está detalhada em um capítulo da tese, e corresponde a um artigo publicado em conferência, periódico ou livro internacional. Nosso objetivo aqui não é descrever em detalhes cada contribuição, mas enumerá-las e fornecer um indicador da natureza da contribuição e dos resultados obtidos.

Uma lista das abreviações utilizadas e os respectivos significados podem ser encontrados no prefácio da tese. Da mesma forma, a descrição detalhada e as referências bibliográficas relativas às diversas técnicas e métodos citados podem ser encontradas no Capítulo 2.

1.6.1 Revisão bibliográfica

Em [103], nós identificamos, descrevemos e classificamos diversas técnicas e sistemas para a recuperação de imagens baseada em informação de cor. Essa revisão bibliográfica corresponde ao Capítulo 2 da tese. Nesse capítulo são discutidos os espaços de cores, técnicas para a redução da informação presente nas imagens, representações para as características visuais extraídas das imagens, funções de distância para a comparação dessas características visuais, técnicas de filtragem e métodos de acesso para reduzir o tempo de busca quando uma consulta visual é processada, sistemas existentes para a recuperação de imagens por conteúdo, e métodos e medidas para a avaliação de efetividade.

1.6.2 *CCH* – Cell/Color Histograms

Em [99, 104], nós propusemos e avaliamos uma representação alternativa e mais compacta para abordagens de recuperação de imagens baseada em particionamento denominanda *CCH – Cell/Color Histograms*. Essa representação é formalmente proposta e avaliada no Capítulo 3.

A idéia central da abordagem *CCH* é que a utilização de *Cell/Color histograms* implica em uma representação mais compacta e flexível que a utilização de histogramas de cor locais (LCHs). O ganho em termos de espaço baseia-se no fato de que apenas um subconjunto reduzido de cores está presente na maioria das imagens. A abordagem *CCH* descreve a distribuição espacial de cada cor nas partições da imagem, ao invés de descrever a distribuição de cores em cada partição individualmente. Essa representação alternativa é mais compacta porque evita a representação da distribuição espacial de cores que não estão presentes nas imagens ou de cores que, intencionalmente, não se deseja representar (por exemplo cores presentes em um número "desprezível" de pixels). Adicionalmente, é proposta uma generalização da função de distância L_1 (*City-block*) para comparar os histogramas utilizados na abordagem *CCH*.

Outra contribuição desse trabalho é a metodologia de avaliação de efetividade discutida na Seção 3.3. Nessa seção, são fornecidas algumas diretrizes para se realizar a avaliação de efetividade em sistemas para CBIR. Dentre os requisitos discutidos estão: (1) uma coleção de imagens que seja representativa do universo a ser investigado; (2) um conjunto de imagens consulta que seja representativo da coleção de imagens utilizada; (3) o conjunto de imagens aceitas como relevantes para cada imagem consulta utilizada; (4) uma medida de efetividade coerente com os critérios de avaliação adotados. Em particular, é proposta a medida de efetividade θ_{rel}. Essa medida é uma variação da medida de precisão média [17] que tem por objetivo normalizar os resultados de efetividade de acordo com características implícitas da metodologia de avaliação adotada. A normalização dos resultados baseia-se no uso de uma abordagem de referência.

Os experimentos descritos no Capítulo 3 basearam-se em uma coleção de 20.000 imagens heterogêneas, no espaço de cores RGB quantizado uniformemente em 64 cores e em 15 imagens consulta. Para cada imagem consulta, o conjunto de imagens consideradas relevantes foi determinado *a priori*. A efetividade das abordagens foi medida utilizando-se gráficos de Precisão vs. Revocação (P×R) e também a medida θ_{rel} mencionada acima.

Um experimento preliminar envolvendo a nossa coleção de 20.000 imagens (discutido na Seção 3.1) mostrou que, em média, cada imagem da coleção é formada por 29 das 64 cores possíveis. Também foi observado que 90% do conteúdo de uma imagem poderia ser descrito (em média) utilizando-se apenas 9 das 64 cores possíveis.

O experimento descrito na Seção 3.4.1 comparou a abordagem *CCH* com 3 outras abordagens para a recuperação de imagens baseada na informação de cor discutidas na Seção 2.6 (*GCH*, *CCV* e *Grid*). Duas das abordagens comparadas eram globais (*GCH*, *CCV*) e a outra era uma abordagem baseada em particionamento (*Grid*). Os resultados confirmaram que as abordagens baseadas em particionamento, apesar de utilizarem consideravelmente mais espaço para representar as imagens, também oferecem ganhos em termos de efetividade. Também foi observado que a abordagem *CCH* foi tão efetiva quanto a abordagem *Grid* porém, como esperávamos, com uma substancial redução de 55% no espaço utilizado para representar as características visuais extraídas das imagens.

O experimento descrito na Seção 3.4.2 avaliou o compromisso entre espaço utilizado e efetividade variando-se o número de células do particionamento espacial. Como esperávamos, quanto maior o número de células, maior a efetividade e maior o espaço utilizado. O experimento descrito na Seção 3.4.3 investigou o compromisso entre espaço utilizado e efetividade quando o conteúdo das imagens é parcialmente representado. Foi observado um grande ganho de espaço (sem comprometer sensivelmente a efetividade) quando apenas cerca de 90% do conteúdo das imagens foi representado. Finalmente, o experimento descrito na Seção 3.4.4 combinou algumas configurações dos experimentos anteriores para demonstrar a flexibilidade da abordagem *CCH*. Os resultados mostraram que, utilizando-se uma partição espacial de 8×8 células e representando 100% do conteúdo das imagens, a abordagem *CCH* e tão efetiva quanto a abordagem *Grid*, com a vantagem de utilizar 55% menos espaço. Também foi observado que, utilizando-se uma partição espacial em 3 × 3 células e representando-se 80% do conteúdo das imagens, a abordagem *CCH* utiliza menos espaço que um *GCH* e, ainda assim, conseguiu ser 43% mais efetiva. Outros resultados intermediários também foram analisados.

1.6.3 *CBC* – Color-Based Clustering

Em [100], nós propusemos e avaliamos o *CBC* (*Color-Based Clustering*), uma nova abordagem regional para a recuperação de imagens baseada em informação de cor. O *CBC* é

formalmente apresentado no Capítulo 4.

O *CBC* segmenta imagens automática e eficientemente utilizando uma técnica de agrupamento de pixels baseada em informação de cor. O algoritmo de agrupamento utilizado é automático, tem uma implementação eficiente e é independente do domínio das imagens, permitindo sua aplicação em grandes coleções de imagens heterogêneas. Foi utilizada uma variação do algoritmo de *single-linkage* [31] cuja complexidade computacional é $O(n \log n)$, onde n é o número de pixels da imagem de entrada. Os pixels são agrupados até que a distância intergrupos exceda um limite d_0. As regiões obtidas após a aplicação do algoritmo de agrupamento são disjuntas, conexas e têm um tamanho mínimo definido por um parâmetro s_0. O número de regiões obtido ao final do processo depende dos parâmetros d_0 e s_0, e do conteúdo visual de cada imagem.

Para cada região obtida é extraído um vetor de características que armazena a cor média da região, seu tamanho (número de pixels), e as coordenadas espaciais do seu centro geométrico. O tamanho e as coordenadas do centro de cada região são normalizados em relação ao tamanho e às dimensões da imagem. Dessa forma, uma imagem é representada por um conjunto de vetores, um para cada região segmentada.

A comparação de duas imagens na abordagem *CBC* é realizada região por região. Como o número de regiões de duas imagens pode ser distinto e as regiões obtidas com o algoritmo automático de segmentação são apenas uma aproximação dos objetos que compõem uma imagem, nós idealizamos uma função de distância que contorna essas aproximações. A função de distância proposta é uma composição ponderada da distância entre pares de regiões. As regiões (reais) das imagens são decompostas em regiões virtuais, de forma que as duas imagens comparadas passam a ter o mesmo número de regiões virtuais, existindo um casamento 1 para 1 entre as regiões virtuais das duas imagens. Duas regiões casadas possuem sempre o mesmo tamanho (que é utilizado como peso na obtenção da distância final entre as imagens). Por coincidência, enquanto o artigo que propunha o *CBC* estava sendo avaliado para publicação, foi publicado um outro artigo que propunha uma distância para comparação de imagens segmentadas denominada *IRM* (*Integrated Region Matching*) [58]. Apesar da formulação distinta, a distância *IRM* se mostrou equivalente à distância que estávamos propondo, limitando assim a nossa contribuição.

Os resultados experimentais descritos na Seção 4.3 comparam o *CBC* com cinco outras abordagens para CBIR discutidas na Seção 2.6: três abordagens globais (*GCH*, *CMM* e *CCV*), e duas abordagens baseadas em particionamento (*Grid* e *CCH*). O *CBC* foi investigado com três combinações distintas de parâmetros, cada uma das três resultando em um número diferente de regiões por imagem. As abordagens foram comparadas em termos de espaço utilizado e efetividade. Os experimentos basearam-se em duas coleções de imagens, uma com 1023 imagens e a outra com 20.000 imagens heterogêneas. Foram utilizadas 29

imagens consulta. Para cada imagem consulta, o conjunto de imagens consideradas relevantes foi determinado *a priori*. A efetividade das abordagens foi medida utilizando gráficos de Precisão vs. Revocação ($P \times R$) e também a medida de rank médio normalizado – $NavgR'$, uma variação simples da medida de efetividade proposta no contexto do projeto QBIC da IBM [32].

Os experimentos mostraram que, dentre as abordagens globais, o *GCH* tem os melhores resultados de efetvidade, enquanto o *CMM*, a menor utilização de espaço. Também foi confirmado que o *CCH* que propomos no Capítulo 3 é mais eficiente e utiliza menos espaço que o *Grid*. Apesar de mais efetivo que o *GCH*, o *CCH* utiliza bem mais espaço por representar o conteúdo de cada célula individualmente. As três variações da abordagem *CBC* se mostraram mais efetivas que a abordagem *CCH* (a mais efetiva dentre as abordagens existentes que foram comparadas), mais robustas em relação ao crescimento da coleção de imagens, e também mais compactas em termos de utilização de espaço. A configuração com um número intermediário de regiões mostrou o melhor compromisso entre espaço utilizado e efetividade.

1.6.4 *MiCRoM* – Minimum-Cost Region Matching

Recentemente, diversos sistemas para CBIR baseados em técnicas de segmentação de imagens têm sido propostos. Nesses sistemas, as imagens são segmentadas e representadas por um conjunto de regiões, e a comparação das imagens é feita de acordo com as características visuais extraídas de cada região. Um problema claro nesse tipo de sistema é a função de distância usada para comparar imagens segmentadas. Em geral, as funções existentes são não-métricas, dificultando a utilização de técnicas de filtragem e/ou métodos de acesso para acelerar o processamento das consultas. Com o objetivo de contornar essa limitação, nós propusemos *MiCRoM* (*Minimum-Cost Region Matching*), uma função métrica para a comparação de imagens segmentadas [102]. A função *MiCRoM* é formalmente apresentada no Capítulo 5.

A função *MiCRoM* é uma extensão da função não-métrica que propusemos em [100] e que, na verdade, se mostrou equivalente à função *IRM* utilizada no sistema SIMPLIcity [58]. A função *MiCRoM* fornece a distância ótima entre duas imagens (de acordo com a modelagem do problema adotada) que a abordagem gulosa utilizada na função *IRM* alguns vezes não consegue obter. A função *MiCRoM* modela a comparação de imagens segmentadas como um problema de fluxo de custo mínimo em redes [2]. Mais especificamente, a comparação de imagens é modelada como o problema do transporte.

O problema do transporte é um problema de programação linear com uma estrutura bastante específica [2]. Por causa disso, é possível utilizar algoritmos especializados que encontram a solução para esse tipo de problema de forma muito mais eficiente que al-

goritmos convencionais de programação linear. Existem inúmeros algoritmos eficientes para solucionar o problema do transporte. No nosso caso, nós utilizamos o algoritmo CS2 proposto por Cherkassky e Goldberg[1] [36].

Os experimentos desse trabalho basearam-se em uma coleção com 20.000 imagens heterogêneas e em 18 imagens consulta. Nesse trabalho, nós passamos a determinar o conjunto de imagens relevantes (RRSet) para cada consulta utilizando uma técnica de *pooling* similar àquela utilizada nas conferências TREC [115, 118]. As imagens relevantes para uma consulta são extraídas de um conjunto de imagens candidatas. Esse conjunto de candidatas é composto pelas 30 primeiras imagens retornadas por cada abordagem investigada no estudo comparativo em questão. As imagens candidatas são visualmente inspecionadas para se determinar a relevância de cada uma. O subconjunto de imagens relevantes passa a ser o RRSet da consulta em questão. A efetividade das abordagens foi medida utilizando-se gráficos de Precisão vs. Revocação $P \times R$.

Nos experimentos descritos na Seção 5.2, nós comparamos as distâncias *IRM* e *MiCRoM* utilizando imagens segmentadas com o *CBC* (descrito na seção anterior). Os resultados mostraram que a distância *MiCRoM* é ao menos tão efetiva quanto a distância *IRM*. Esse resultado comprova que a estratégia gulosa adotada pela *IRM* funciona muito bem, pois os resultados de efetividade são quase tão bons quanto os resultados obtidos com a *MiCRoM* (versão ótima da distância *IRM*). A vantagem da *MiCRoM* é ser uma função métrica que permite a utilização da propriedade da desigualdade triangular para acelerar o processamento de consultas.

Os experimentos descritos na Seção 5.3 avaliam a utilização de uma técnica de filtragem baseada na propriedade da desigualdade triangular para acelerar o processamento de consultas. A técnica de filtragem utilizada foi proposta por Santos et al [84]. A utilização da filtragem permitiu reduzir em 2/3 o tempo gasto para se realizar uma busca pelos 100 vizinhos mais próximos de uma imagem.

1.6.5 *BIC* – Border/Interior Pixel Classification

Em [101] nós propusemos *BIC* (*Border/Interior Pixel Classification*), uma nova abordagem para a recuperação de imagens por conteúdo em grandes coleções de imagens heterogêneas. A abordagem *BIC* é formalmente apresentada no Capítulo 6.

O foco da abordagem *BIC* é a simplicidade. Nossa experiência com a abordagem *CBC* e a distância *MiCRoM* nos ensinou que, no contexto de imagens heterogêneas, tanto a segmentação automática quanto a comparação de imagens segmentadas são problemas bastante difíceis. Com o objetivo de manter as soluções para esses problemas tratáveis do ponto de vista computacional, foi necessário introduzir vários tipos de simplificações

[1] http://www.intertrust.com/star/goldberg/soft.html

que, como não poderia deixar de ser, têm impacto direto na efetividade do sistema. Nesse sentido, a abordagem *BIC* representa uma alternativa diferente para o problema de recuperação de imagens por conteúdo. Ao invés de utilizar técnicas sofisticadas cujos resultados precisam ser relaxados para serem tratáveis do ponto de vista computacional, a abordagem *BIC* utiliza técnicas simples (porém poderosas) cujos resultados podem ser preservados (sem simplificações) durante todas as etapas do processo de recuperação de imagens por conteúdo. A abordagem *BIC* tem três componentes principais: (1) um algoritmo simples, eficiente e poderoso para a análise do conteúdo visual das imagens, (2) uma nova função de distância logarítmica para a comparação de histogramas de cores e (3) uma representação compacta para as características visuais extraídas das imagens.

O algoritmo de análise de imagens da abordagem *BIC* utiliza o espaço de cores RGB uniformemente quantizado em $4 \times 4 \times 4 = 64$ cores. Após a quantização do espaço de cores, é feita uma classificação binária dos pixels da imagem de entrada. Cada pixel é classificado em borda ou interior. Um pixel é considerado borda se ao menos um de seus quatro vizinhos (superior, inferior, direito e esquerdo) possui uma cor quantizada diferente da sua. Caso contrário, o pixel é classificado como interior. Após a classificação dos pixels, são calculados dois histogramas de cores: um considerando-se apenas pixels classificados como borda e o outro, considerando-se apenas pixels classificados como interior.

A classificação dos pixels em borda/interior permite analisar o conteúdo das imagens em termos (1) do tamanho das regiões conexas (regiões grandes possuem mais pixels de interior enquanto regiões pequenas possuem mais pixel de borda), (2) da forma das regiões conexas (regiões com forma regular possuem mais pixels de interior enquanto regiões com forma irregular possuem mais pixels de borda) e (3) da homogeneidade das regiões (regiões planas possuem mais pixels de interior enquanto regiões de textura possuem mais pixels de borda). O grau em que cada uma das propriedades acima é verdadeira depende da proporção entre pixels de interior e de borda e também da porção da imagem coberta por cada uma das cores.

Os histogramas que representam as imagens na abordagem *BIC* são comparados utilizando-se uma nova distância à qual denominamos *dLog*. A função *dLog*, ao invés de calcular a diferença entre os elementos do histograma diretamente, calcula a diferença entre o log desses elementos. O objetivo é reduzir o efeito negativo introduzido por um único elemento do histograma com um valor muito alto. Um único elemento do histograma com um valor muito alto domina a diferença entre histogramas mas, em geral, esse elemento está associado ao fundo da imagem (*background*) o qual possui pouca informação semântica e, como conseqüência, possui pouca importância semântica no julgamento de similaridade feito pelo usuário.

A utilização da função *dLog* para comparar histogramas, além de aumentar a efetividade do sistema, permite armazenar os histogramas em metade do espaço originalmente

necessário. Essa redução é possível armazenando-se o log dos elementos do histograma ao invés do valor original. No caso da abordagem *BIC* (tal como propusemos), é possível representar o conteúdo visual de qualquer imagem em apenas 64 bytes de memória. Como conseqüência, é possível manter em memória as características visuais de grandes coleções de imagens, eliminando completamente a necessidade de métodos de acesso a disco para agilizar o processamento de consultas visuais.

Os experimentos desse trabalho basearam-se em uma coleção com 20.000 imagens heterogêneas e em 50 imagens consulta. Assim como discutido na Seção 1.6.4, o conjunto de imagens relevantes (RRSet) para cada consulta foi determinado utilizando-se uma técnica de *pooling* similar àquela utilizada nas conferências TREC [115, 118]. A efetividade das abordagens comparadas foi medida utilizando-se 11 medidas diferentes. Foram utilizadas duas medidas em forma de gráficos ($P \times R$ e $\theta \times R$), e nove outras medidas que resultam, cada uma delas, em um único valor para a efetividade de um sitema ($P(r)$, $P(30)$, $R(30)$, $P(100)$, $R(100)$, *3P-Precision*, e *11P-Precision*). A medida de $\theta \times R$ é uma variação da medida de $P \times R$ que nós propusemos e que acreditamos ser mais adequada ao contexto de recuperação de imagens por conteúdo, além de ser mais facilmente interpretada.

Na Seção 6.3, a abordagem *BIC* é comparada com quatro outras abordagens, o *CBC* descrito na Seção 1.6.3, uma abordagem baseada em particionamento (*Grid 9*) e duas abordagens globais (*GCH* e *CCV*). Os resultados de 11 medidas de efetividade confirmam que a abordagem *BIC* é consideravelmente mais efetiva que as demais, incluindo o *CBC*. Além de ser mais efetiva, a abordagem *BIC* é também mais compacta e mais eficiente.

Um segundo experimento avaliou a utilização da distância *dLog* em várias abordagens baseadas em histogramas de cores. Em todos os casos, houve um ganho sensível de efetividade em comparação com a utilização da função L_1. Além do ganho de efetividade, a utilização da função *dLog* permite reduzir pela metade o espaço necessário para armazenar os histogramas. Nenhuma das abordagens existentes (mesmo utilizando a função *dLog* para comparar histogramas) conseguiu ser melhor que a abordagem *BIC*, sugerindo que, embora a função *dLog* tenha uma contribuição importante na efetividade da abordagem *BIC*, o algoritmo de análise de imagens proposto é capaz de fazer a diferença em relação às abordagens investigadas.

1.7 Organização da tese

O restante desta tese, com exceção do Capítulo 7 (Conclusões e trabalhos futuros), está escrito em inglês. O conteúdo de cada capítulo baseia-se em um artigo publicado em periódico, conferência ou livro internacional. O conteúdo de cada capítulo foi adaptado para evitar redundância em termos de conteúdo com os capítulos anteriores. Quando relevante, foram acrescentadas algumas informações que não estão presentes nos artigos

originais por causa de restrições de espaço. O restante da tese está organizado como se segue.

O Capítulo 2 identifica, classifica e descreve as principais técnicas e sistemas para a recuperação de imagens baseada em informação de cor [103]. São discutidos os espaços de cores (Seção 2.1), técnicas para a redução da informação presente nas imagens (Seção 2.2), representações para a informação de cor (Seção 2.3), funções de distância para a comparação das características visuais extraídas das imagens (Seção 2.4), técnicas de filtragem e métodos de acesso para reduzir o tempo de busca quando uma consulta visual é processada (Seção 2.5), sistemas existentes para a recuperação de imagens por conteúdo (Seção 2.6), e métodos e medidas para a avaliação de efetividade (Seção 2.7).

O Capítulo 3 descreve e avalia uma representação alternativa e mais compacta para abordagens de recuperação de imagens baseada em particionamento denominada *Cell/Color histograms – CCH* [99, 104]. O *CCH* é proposto na Seção 3.1. A Seção 3.2 apresenta uma generalização da função de distância L_1 (*City-block*) para comparar os histogramas utilizados na abordagem *CCH*. Uma nova metodologia para a avaliação de efetividade e também uma nova medida denominada θ_{rel} são discutidas na Seção 3.3. Os resultados experimentais são apresentados e discutidos na Seção 3.4. A Seção 3.5 apresenta as conclusões do capítulo.

O Capítulo 4 descreve e avalia uma nova abordagem regional para a recuperação de imagens baseada em informação de cor denominada *CBC* (*Color-Based Clustering*) [100]. O algoritmo de agrupamento utilizado para segmentar as imagens é descrito na Seção 4.1, e a função de distância utilizada para comparar as características visuais extraídas das imagens é descrita na Seção 4.2. Os experimentos são detalhados na Seção 4.3 e os resultados experimentais são discutidos na Seção 4.4. Finalmente, a Seção 4.5 apresenta as conclusões do capítulo.

O Capítulo 5 descreve e avalia *MiCRoM* (*Minimum-Cost Region Matching*), uma nova função métrica para a comparação de imagens segmentadas [102]. A função *MiCRoM* é formalmente descrita na Seção 5.1. A efetividade da função *MiCRoM* é comparativamente avaliada na Seção 5.2. A utilização da propriedade da desigualdade triangular para acelerar o processamento de consultas visuais é avaliada na Seção 5.3. A Seção 5.4 apresenta as conclusões do capítulo.

O Capítulo 6 descreve e avalia *BIC* (*Border/Interior Pixel Classification*), uma nova abordagem para a recuperação de imagens por conteúdo em grandes coleções de imagens heterogêneas [101]. A abordagem *BIC* tem três componentes principais: (1) um algoritmo simples, eficiente e poderoso para a análise do conteúdo visual das imagens descrito na Seção 6.1.1, (2) uma nova função de distância (denominada *dLog*) para a comparação de histogramas de cores descrita na Seção 6.1.2 e (3) uma representação compacta para as características visuais extraídas das imagens que é descrita na Seção 6.1.3. Os experimen-

tos são detalhados na Seção 6.2 e os resultados experimentais são discutidos na Seção 6.3. Finalmente, a Seção 6.4 apresenta as conclusões do capítulo.

O Capítulo 7 apresenta as conclusões da tese e identifica trabalhos futuros.

Capítulo 2

Color-based Image Retrieval

This chapter[1] discusses techniques for color-based image retrieval, focusing in the five most important issues that have to be addressed in order to achieve color-based image retrieval: (a) what color-space we should use to describe, analyze and compare images; (b) how to describe images based on their color distribution and the spatial distribution of colors; (c) how to represent the image content (i.e., visual features) in an image database; (d) what distance function should be used to measure the similarity between two images based on their visual features; and (e) which access method should be used to speedup query processing. In addition, existing color-based image retrieval approaches are discussed and classified into global, partition-based and regional, according to the representation adopted for the color distribution of the images.

Image databases are becoming more and more common in several distinct application domains, such as (multimedia) search engines, digital libraries, medical and geographic databases and criminal investigation. The evolution of techniques for acquisition, transmission and storage of images has also allowed the construction of very large image databases. All these factors have spurred great interest in image retrieval techniques.

Image retrieval is performed based on short descriptions of the images. Images may be described by a set of content-independent attributes (file name, format, category, size, author's name, input device, date of creation and network/disk location) that can be managed through conventional database management systems - DBMS. The main drawback of this approach is that the allowed queries are limited to those based on the existing attributes. Another alternative is to use keywords or annotations, such that images can be retrieved by traditional information retrieval techniques (IR). This approach is less restrictive than the previous one, but it still has problems like incompleteness, subjectiveness and the drawback of manually annotating each individual image.

[1] The content of this chapter will be published as a chapter in the book entitled *"Multimedia Mining - a Highway to Intelligent Multimedia Document"* [103].

A more adequate alternative to image retrieval consists of using low-level image features like color and texture to represent, compare and retrieve images. This approach is called *content-based image retrieval* – CBIR, nowadays an area of active multidisciplinary research. A typical application for CBIR techniques is a World-Wide-Web (WWW) multimedia search engine. The visual content of the WWW is a good example of a large and heterogeneous image database, in the sense that the images belong to several distinct, non-related semantic and visual domains. In this context, it is not possible to assume or use any *a priori* knowledge about the visual content of the images during the image analysis step. Moreover, the cost of using semi-automatic image analysis techniques is prohibitive. In this scenario, low-level features related to the visual content of the images, such as color, are useful to represent and compare images automatically.

In fact, color is the most commonly used low-level feature in CBIR systems. Some possible reasons for this fact are that (1) color is a feature which is immediately perceived by humans when looking at an image, (2) the concepts involved are easy to understand and to implement, (3) color is an important visual feature in the large majority of image domains and (4) the results obtained by using color information are often satisfactory.

Despite the importance of describing images at different levels, using distinct visual features and retrieval techniques, this chapter is mostly concerned with color-based image retrieval, an important component in many image retrieval systems. As it shall become clear from our discussion, retrieval of images according to color properties is inherently different from, and more complex than, retrieval of well-structured (traditional) data.

Figure 2.1 shows a schematic representation of an image being inserted into an image database. After a new input image is given (in its pictorial form), its visual content (e.g., color distribution and spatial distribution of colors) is analyzed and summarized according to a predefined color-space. Compact representations are chosen for the information obtained during the image analysis step. The representation of the image's visual content is then inserted into an index structure, useful to reduce the search space at query time and, consequently, the query processing time. The index structure is based on the image representation and uses the properties of a distance function (used to measure the similarity of two images) to reduce the search space. As well, the visual features of the input image are stored in the image database.

According to the schema above, we consider the existence of five most important issues that have to be addressed in order to achieve color-based image retrieval, each addressed in a forthcoming section: (a) what color-space we should use to describe, analyze and compare images (Section 2.1); (b) how to describe images based on their color distribution and the spatial distribution of colors (Section 2.2); (c) how to represent the image content (visual features) in an image database (Section 2.3); (d) what distance function should be used to measure the similarity between two images based on their visual features

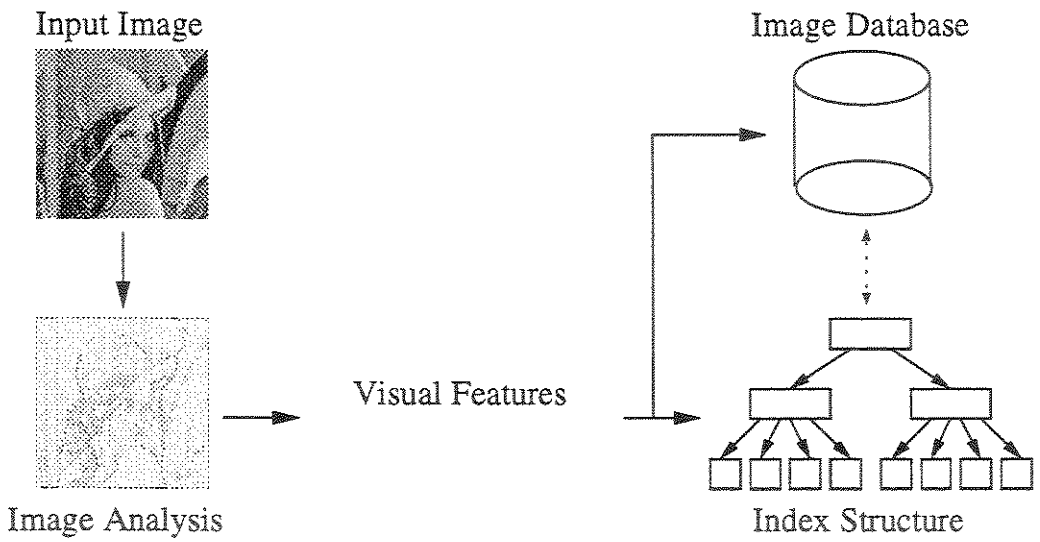

Figure 2.1: Schematic representation of an image being stored in an image database

(Section 2.4); and (e) which access method should be used to index the visual features (Section 2.5).

In addition, Section 2.6 discusses some existing approaches for color-based image retrieval. We use the representation adopted for the color distribution to classify the approaches in global, regional and partition-based. Section 2.7, presents a discussion about retrieval effectiveness evaluation (i.e. how to evaluate the user's satisfaction with the retrieved images), which is a complex problem shared by all kinds of CBIR systems.

2.1 Color-spaces

Color information in digital images is found at pixel level. The color of a pixel is represented by three values, one for each channel of the chosen color-space. In essence, a color-space is a specification of a 3D coordinate system and a subspace within that system where each color is represented by a single point [37]. The choice of a color-space where images will be represented, analyzed and compared is the first step in any color-based image retrieval system. Existing color-spaces can be classified in three main categories: (1) hardware-oriented, (2) user-oriented and (3) uniform color-spaces.

Hardware oriented models are defined according to properties of the devices used to reproduce the colors (computer screen, color printer, TV monitor, etc). The best known and used color-space is a hardware-oriented model known as RGB (Red, Green, Blue) [13, 37, 61]. The RGB color-space is device-dependent, i.e., the displayed color depends not only on the RGB values, but also on the device specifications. It is also not

perceptually uniform, in the sense that the differences between RGB colors do not reflect the differences perceived by humans. The RGB color-space is a cube as shown on the left of Figure 2.2, where the main diagonal represents the gray values from black to white, and any point (color) inside the cube is represented by a weighted sum of red, green, and blue.

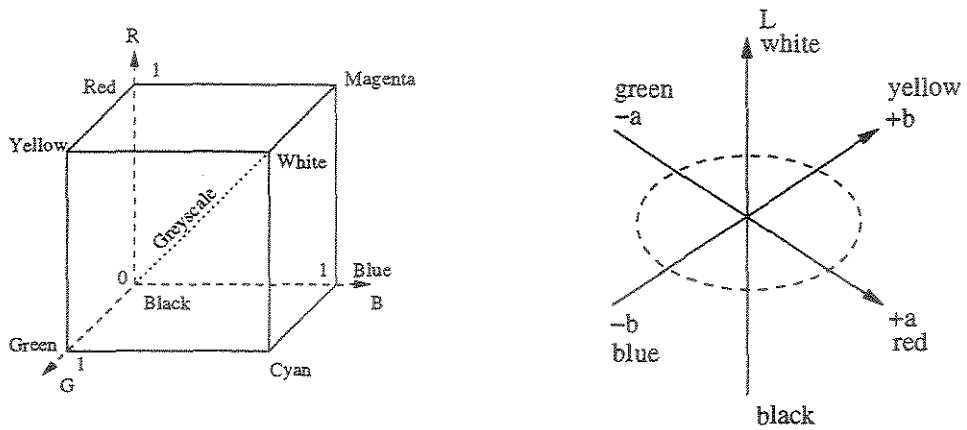

Figure 2.2: The RGB and the CIE Lab color-spaces

Uniform color-spaces are spaces where the numerical differences among colors are consistent with the differences perceived by humans. Examples are the Lab and Luv color-spaces of CIE [13, 61]. CIE Lab model represents the differences of three elementary pairs: red-green, yellow-blue and black-white. Thus, as shown on the right of Figure 2.2, the a axis of the CIE Lab color-space extends from green $(-a)$ to red $(+a)$ and the b axis from blue $(-b)$ to yellow $(+b)$. The brightness (L) increases from the bottom to the top of the three-dimensional model. The most important aspect of the CIE Lab color-space is that it is device independent.

User-oriented color-spaces are based on human perception of colors [13, 61]. They exploit characteristics that are used by humans to distinguish one color from another such as hue (the dominant wavelength that produces the visual sensation of red, yellow, green and blue, or a combination of two of them), saturation (the purity of the color, that is related to the standard deviation around the dominant wavelength) and intensity (the brightness of the color, that is related to the amount of white in the color). Some examples of user-oriented spaces are the HSI and HSV color-spaces. The HSV (Hue, Saturation, Value) color-space, for example, is represented by a hexagonal cone (see Figure 2.3). The vertical axis of this cone represents the gray values (or intensities) from black to white, the angle around the vertical axis defines the hue, and the distance from the vertical axis gives the saturation. The Hue values vary from [0,360] degrees, starting from red (0),

through yellow (60), green (120), cyan (180), blue (240), magenta (300) and back to red (360=0). HSV is also an non-uniform color-space.

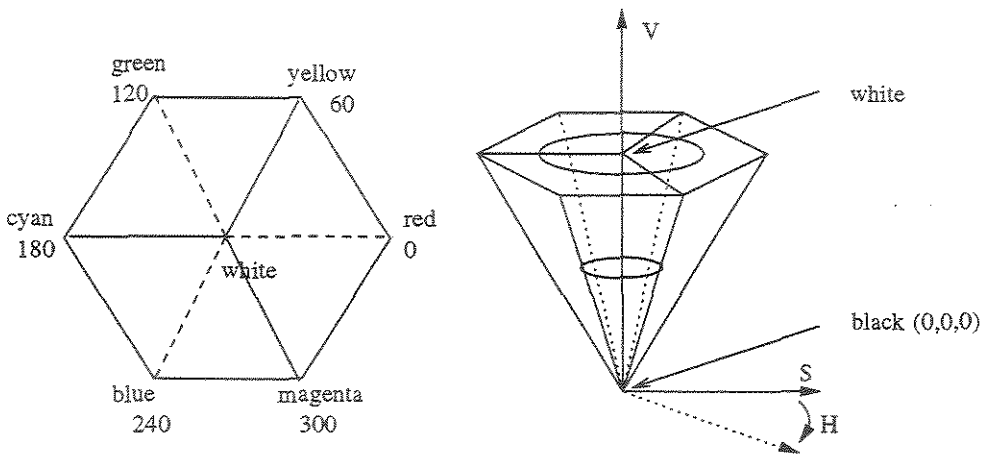

Figure 2.3: The HSV color-space

2.2 Color-based image description

This section discusses techniques to describe the color information present in an image. Color information in digital images could be represented at pixel level, but this approach would make impractical CBIR systems. Suppose for example that each color channel of the RGB color-space is represented using 8 bits, i.e., it is possible to represent $2^8 = 256$ distinct levels for each color component, resulting in $256 \times 256 \times 256 = 16,777,216$ distinct colors. Moreover, consider an image with spatial dimensions 300×300. This means that there are 90,000 absolute spatial locations to be considered in a pixel-by-pixel comparative analysis of two images. These two numbers (distinct colors and spatial locations) are usually sufficiently large to prevent the comparison of images at the pixel level.

Therefore, it is required a shorter description of the color distribution and the spatial distribution of colors that provides efficiency and effectiveness in CBIR systems. The color distribution indicates the percentage of each color in the image, while the spatial distribution of colors indicates in what regions of the image a given color appears. These descriptors can be further reduced in size by static or dynamic reduction methods. Static methods uses a fixed scheme for every image, while dynamic methods exploit the visual content of the image to produce shorter, more flexible and more robust descriptors.

2.2.1 Static reduction methods

The simplest scheme to reduce the number of colors present in an image is the use of a uniform and coarse quantization of each color channel. For example, if instead of using 8 bits to represent each color channel, it is used the two most significant bits (hence uniformly quantizing each channel in 4 distinct values), it is obtained a total of $4\times4\times4=64$ distinct colors. An advantage of the static quantization is that the obtained colors do not need to be represented explicitly, since they can be derived from the quantization scheme. It makes the comparison of images easier, because the number of colors and the colors themselves are constant for all images.

The static and uniform quantization of a color-space has also well-known disadvantages. One problem is that the colors present in an image are not necessarily uniformly distributed in the color-space. Another problem is that it is difficult to obtain an adequate compromise about the granularity of the quantization. It should be fine enough in such a way that perceptually distinct colors are not classified together, but coarse enough to drastically reduce the number of distinct colors present in the image. Finally, uniform quantization is not appropriate for non-uniform color-spaces such as RGB and HSV, since similar colors may be separated and non-similar colors classified together.

An alternative to avoid the static quantization step is to reduce the color information by computing statistics about the color distribution such as average color. Such methods have the advantage to be computationally simple, to result in very compact descriptors, and to provide an efficient way for image comparison. However, their effectiveness is usually low because images composed by completely different colors might result in identical statistics.

Static quantization schemes can also be used to reduce the color spatial distribution. This corresponds to reduce the image resolution by pixel resampling, or the most common approach, by superimposing a grid of rectangular cells over the image such that the color distribution of each cell is computed individually. Image partitioning is an important factor to determining the functionality and the efficiency of CBIR systems [94]. For instance, by breaking the images into smaller, more manageable units, it usually becomes easier for the systems to compress, store, access and retrieve the image data. However, no single partitioning scheme is known to be optimal for distinct CBIR applications.

2.2.2 Dynamic reduction methods

Dynamic reduction methods exploit the visual content of the images to reduce simultaneously the number of distinct colors and the number of spatial locations in an image. These methods rely on image segmentation techniques that group together neighboring pixels with similar colors. Each group represents an image region whose color is the average

color of its pixels. In this way, the number of distinct colors present in the original image is reduced. Simultaneously, the image is segmented into regions with high degree of color similarity and well-defined spatial location, size and shape. These characteristics are more compact and meaningful than the spatial location of each individual image pixel.

In general, dynamic reduction methods make use of one of the following image segmentation techniques: boundary detection [37], region growing [37], region splitting and merging [37], density estimation [18, 76] and hierarchical clustering [5, 31, 45].

Boundary detection techniques assume that the transition between two regions can be determined on the basis of visual property discontinuities. In general they should be followed by some kind of edge-linking algorithm. Region growing techniques start with a set of *seed points* and, from these points, grows regions by appending to each seed point those neighboring pixels with similar color. Region splitting and merging techniques subdivide an image into a set of arbitrary, disjoint regions, and then merge and/or split the regions depending of the colors present in each region. Density estimation techniques are based on the assumption that the underlying data density is a mixture of g Gaussian densities. The g means and covariances of these Gaussians are estimated and the data are partitioned among them to get regions.

Hierarchical clustering [5, 31, 45] are among the best-known clustering methods. There are basically two types of hierarchical algorithms: *agglomerative* and *divisive*. Agglomerative methods start when all pixels are apart, i.e., they start with n singleton clusters. Then in each step two clusters are merged until a stop criterion is satisfied (for example, a predefined number of regions is obtained). A generic agglomerative clustering algorithm is shown in Figure 2.4.

```
    Aggl-Clustering(k)
1      Consider n singleton clusters, one for each data element
2      Let k' = n.
3      If k' ≤ k, then stop.
4      Find the nearest pair of distinct clusters, say A and B
5      Merge A and B, delete B, and decrement k' by one
6      Go to 3
```

Figure 2.4: Generic agglomerative clustering algorithm

Divisive methods start when all pixels are together and, in each following step, a cluster is split up, until there are n of them. In the literature, hierarchical clustering is usually meant to be agglomerative clustering. The main reason for this appears to be the computational aspect [45]. In the first step of an agglomerative algorithm all possible

fusion of two objects are considered, leading to $C_n^2 = \frac{n(n-1)}{2}$ combinations. This number grows quadratically with n. A divisive algorithm based on the same principle would start by considering all divisions of the data set into two nonempty subsets, which amounts to $2^{n-1}-1$ possibilities. The latter number grows exponentially fast and even for medium-size data sets, such a complete enumeration approach is computationally prohibitive.

When $d_{min}(A,B) = min(d(A_i, B_j))$ is used as distance measure between clusters $A = \{A_1, A_2, ..., A_n\}$ and $B = \{B_1, B_2, ..., B_m\}$, the resulting clustering algorithm is often called *nearest-neighbor* or the *minimum* algorithm. If it is terminated when the distance between nearest clusters exceeds an arbitrary threshold, it is called the *single-linkage* algorithm, which is the oldest and simplest agglomerative clustering algorithm. Figure 2.5 shows an image processed using two of the techniques described above (boundary detection and hierarchical clustering).

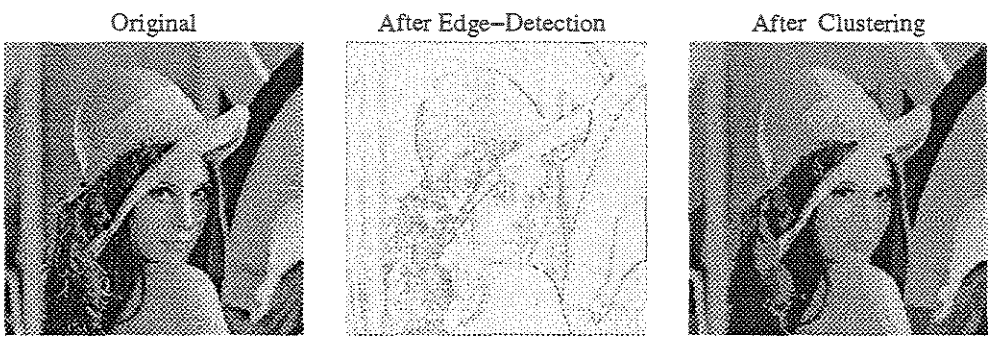

Figure 2.5: An image after edge detection and hierarchical clustering

2.3 Visual features extraction and representation

Once we have chosen a short descriptor for the color information present in an image, the next step in a color-based CBIR system consists of representing this information in the image database. The stored information about the visual content of an image is what we call its *visual features*. In this section, possible representations for the color information are classified in global, partition-based and regional.

2.3.1 Global representations

Global representations describe the color distribution of the whole image, ignoring the spatial distribution of colors. The most used global representation for the color distribution of an image is the *Global Color Histogram – GCH*.

A *GCH* is obtained by counting how many pixels of the image have each of the quantized colors (obtained after a quantization step). It can be viewed as a k-dimensional vector, where k is the number of colors represented by the histogram. Usually the pixel count is normalized to avoid scaling bias. An example of global histogram is shown in Figure 2.6.

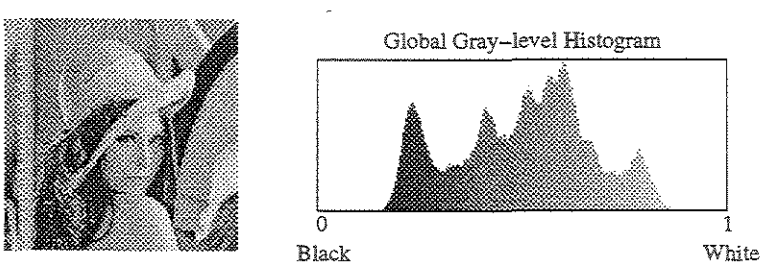

Figure 2.6: An image and its global histogram

When used to represent non-uniformly quantized colors, *GCH*s are always dense vectors, i.e., there is no histogram bins with zero value. Moreover, each quantized color should be represented explicitly, and the dimension of the *GCH*s of two distinct images can be different, as the number of colors obtained using non-uniform quantization schemes depends on the visual content of each image.

When the *GCH* is used to represent uniformly quantized colors, there is no need to represent colors explicitly, as this information can be derived from the quantization scheme adopted. In this context, the existence of null bins within the histogram is common, i.e., it is common that an image be composed only by some of the quantized colors. In this particular case, it is possible to apply compression techniques based on the existence of null bins to reduce the space required to store *GCH*s.

2.3.2 Partition-based representations

Partition-based representations describe the color distribution of each cell of an image individually. In this case, it is assumed that the image was statically partitioned into a set of rectangular cells, according to a predefined scheme. The color distribution of each partition cell is described individually, by means of a *Local Color Histogram* – LCH, as shown in Figure 2.7.

As the partition-scheme is independent of the visual content of the images, it is not possible to assume that the colors of each partition cell are similar. In this case, the local color histogram (LCH) representation seems to be more robust than the use of simple statistics such as average color.

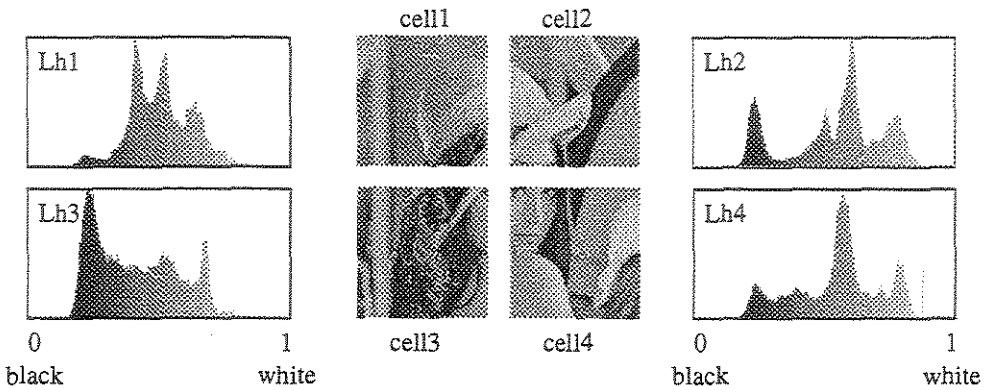

Figure 2.7: An image partitioned in 4 cells and their respective local gray-level histograms

In partition-based representations, there is no need to store spatial properties of the partition cells such as size, shape and spatial location. This information can be easily derived from the partition scheme adopted (which is common for every image).

2.3.3 Regional representations

Regional representations describe the color distribution of each image region individually. The main difference of regional and partition-based representations is that the regions of an image are obtained dynamically, according to the visual content of the image. Ideally, the obtained regions correspond to the high-level concept of objects that an user can easily distinguish when he/she looks at the image. Unlike partition cells, the regions of an image have different size, spatial location and shape. This additional information can be stored and used to increase retrieval effectiveness.

When the segmentation process is based solely on color properties, it is expected a high degree of color-similarity among the colors present in a region. In this case, it is possible to adopt simple statistic information (such as average color) to represent the color distribution of each region.

2.4 Distance functions

An important aspect of any CBIR system is the distance function used to compare the visual features extracted from images. The distance function affects directly the time spent processing a visual query and the quality of the retrieval (effectiveness). The better the distance simulates the human perception of similarity using the available visual features, the more effective is the CBIR system in retrieving images relevant to the user's needs.

The computational complexity of the distance function is also an important factor when processing a visual query. Depending on the distance complexity, it is possible that the processing time to compute distances between images dominates the time needed to access the disk pages where the visual features are stored. The distance function also restricts the universe of filtering techniques and/or access methods that can be used to speedup query processing.

If the visual features of an image are represented by k-dimensional vectors, these vectors can be viewed as points in a k-dimensional space (each vector element corresponds to a spatial coordinate). In this case, it is possible to use geometric distances of the L_p family to compare the visual features of two images. Suppose $a = \{a_1, a_2, ...a_k\}$ and $b = \{b_1, b_2, ..., b_k\}$ are two k-dimensional vectors. The family of L_p distances is defined as:

$$L_p(a, b) = (\sum_{i=1}^{k} |a_i - b_i|^p)^{1/p} \tag{2.1}$$

Some well-known members of the L_p family are the following distances:

- L_1 (City-Block): $L_1(a, b) = \sum_{i=1}^{k} |a_i - b_i|$
- L_2 (Euclidean): $L_2(a, b) = (\sum_{i=1}^{k} |a_i - b_i|^2)^{1/2}$
- L_∞ (Chebyshev): $L_\infty(a, b) = max_{i=1}^{k} |a_i - b_i|$

Figure 2.8 shows the set of points (in a 2D vectorial space) at the same distance r from a center point, according to each of the three geometric distances discussed above.

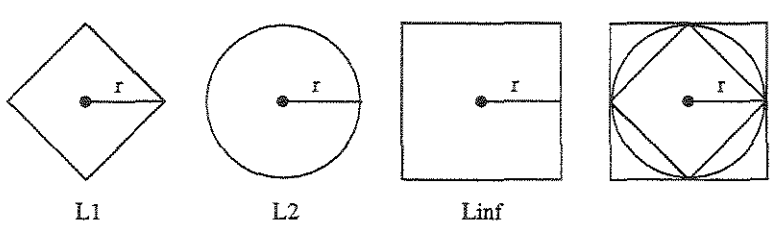

Figure 2.8: Points at the same distance r from a central point according to distinct L_p distances

The advantage of modeling visual features in a vectorial space is that the geometric distances used to compare two vectors are computationally simple. Moreover, as will be discussed in Section 2.5.2, it is possible to use spatial access methods to speedup query processing. However, it is not always possible or effective to model complex CBIR systems in a vectorial space. This is the case in regional CBIR systems, where the number

of regions of two images and their properties are not the same. In this case, a more adequate alternative is to model the system using a metric space.

A metric space is composed by a set of elements (in our case, these elements are the visual features stored in the image database) plus a *metric distance* to compare these elements. In metric spaces, there is no restriction about the representation of the visual features. In this case, what really matter are the properties of the distance used to compare the visual features. A distance d that is used to compare images is considered a metric if, for any images x, y and z, the following properties hold:

- Positiveness or minimality - $d(x, y) \geq 0$, for every x and y
- Symmetry - $d(x, y) = d(y, x)$, for every x and y
- Reflexivity or self-similarity - $d(x, x) = 0$, for every x
- Triangular inequality - $d(x, y) \leq d(x, z) + d(z, y)$, for every x, y and z

A graphical representation of the triangular inequality property in a 2D vectorial space can be viewed in Figure 2.9. It is important to notice that vectorial spaces are particular cases of metric spaces. The triangular inequality property is the most important metric axiom for indexing purposes, as this property is extensively used to reduce the search space at query time, as discussed in the next section.

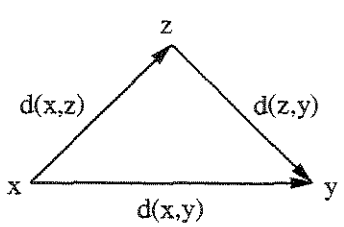

Figure 2.9: A graphical 2D representation of the triangular inequality property

In the psychology literature, it has been found that some measures used to model the human perception of similarity contradict in different ways the metric axioms [83]. It is believed that the metric axioms are too restrictive in the context of similarity search. One of the most criticized metric axioms is the triangular inequality property, coincidentally the most important axiom for indexing purposes [6, 71]. There are some alternatives to deal with the limitations of the metric model such as the three ordinal properties (more flexible than the metric axioms) proposed by Tversky and Gati [110], and a model based on set-theoretic considerations known as *Feature Contrast Model* – FCM [109]. This model

was recently extended by Santini and Jain [83] to include the use of fuzzy predicates. The main disadvantage of using non-metric spaces is that, in this context, the indexing is an open problem in computer science (Section 2.5.4).

2.5 Similarity search

Searching is a fundamental problem in computer science. However, searching for database images which are similar or close to a given visual query is inherent different from the exact-match search in traditional database systems.

The simplest algorithm for searching an image database is *sequential scanning*. In this approach each image of the database is compared against the query image to measure their similarity and select the images that should be returned. Although simple, this approach is not viable for large image databases as the time spent processing a query is proportional to the database size, i.e., the sequential scanning approach does not scale well.

There are basically two alternatives to reduce the complexity of the searching process. One is the use of filtering techniques and the other is the use of access methods. Filtering techniques are based on a simple distance that lower-bounds the original complex distance used to compare images. This distance is used to quickly filter out irrelevant images. Only those images that could not be filtered out (in general a much smaller subset of the complete database) have to be compared using the more complex function. Access methods may use more sophisticated combinations of techniques and data structures to organize the visual features and manage the search process so that visual features relevant to a visual query can be located quickly. In access methods, the aim is to divide the search space into several subspaces in a way that only a few of these subspaces need to be searched when processing a visual query.

2.5.1 Filtering

Since efficient techniques to cope with vector spaces exist, application designers try to give their problems a vector space structure. One of the most common reductions consists of mapping a general metric space into a vector space in such a way that each element of the metric space will be represented as a point in the target vector space [84]. The two spaces will be related by two distances, the original distance $d(x, y)$ and the vectorial distance $d_v(x, y)$ which calculate the distance between points in the vectorial space. Because of the space reduction, some non-relevant elements (false-positives) can be captured in the vectorial space when a query is processed. Thus, the result of a query processed in the vectorial space generates a candidate list, which should be analyzed using the original

distance to eliminate false-hits. If the vectorial distance is a lower-bound for the original distance, then it is guaranteed that the filtering process will not filter out relevant images (false-negatives).

An example of the reduction discussed above is the use of the average color as a filter for color histograms. As the comparison of average colors is much more efficient than the comparison of color histograms, it is possible to quickly eliminate the majority of the non-relevant images using this simple filter. Only the images in the candidate-list should be compared using their color histograms.

More general filtering techniques define k images of the database as reference, compute and store the distances between the database images and the reference images as k-dimensional vectors (which represent points in a vectorial space) and then, use a simple and efficient geometric distance to filter out non-relevant images in the vectorial space, generating a candidate list. It is important to observe that this approach implies in additional overhead to compute the coordinates of the images in the projected vectorial space and also additional overhead to store these new coordinates. Santos et al [84] discuss how to define the best number of reference objects (spatial dimensions in the projected vectorial space) and present an efficient algorithm to find out good reference objects based on the concept of intrinsic dimension [20].

Filtering techniques make extensive use of the triangular inequality property to eliminate non-relevant images without computing the original distance between images, reducing the CPU time required to process a visual query. However, the number of disk accesses (I/Os) remains approximately the same, as the whole database should be compared in the vectorial space. One alternative to reduce the number of I/Os to process a query is to index the vectorial space using a spatial access method (SAM). A SAM reduces the comparison of images only to those near to the query image in the vectorial space, reducing the number of I/Os to process a visual query. SAMs will be discussed in next section.

2.5.2 Spatial access methods – SAMs

Spatial access methods (SAMs) [35] make extensive use of spatial coordinates to group and classify points in the space. These methods are very sensitive to the number of dimensions of the vectorial space. This dependence is called the *curse of dimensionality* [1, 12, 14, 35]. In general, when the vector space dimension is high, the use of dimensionality reduction techniques is common. Some examples of these techniques are mathematical transforms that preserve distance, like Karhunen Loeve, Discrete Fourier or Discrete Cosine [80]. These mathematical transforms map the original vectors into new vectors where the information is more representative at the first coefficients. The indexing of only these

first coefficients of the new vector reduces the dimensionality of the data at the cost of introducing false-positives (because of the little loss of information related to coefficients not indexed). These false-positives should be discarded in a post-processing step. A good survey on SAMs can be found in [35]. Next we will briefly discuss some existing SAMs.

The MD^+-tree [26] is an extension of the traditional B^+-tree structure [27, 50, 105] to support multiple dimensions and also similarity queries. The K-d tree [11] is a generalization of a binary tree that uses k-dimensional vectors instead of a single-valued number. The main problem of this structure is that it is not balanced and thus, its performance depends on the order in which objects are inserted. The Grid-file method statically divides a k-dimensional space into equal-sized hypercubes, and use these hypercubes to reduce the search space [72]. The R-tree [40] and its most well-known variation, the R^*-tree [10], are height-balanced trees that dynamically decompose the space, and represent this decomposed space into an hierarchical structure based on the notion of *minimum bounding rectangles* – MBRs. An example of some MBRs and their organization into an R-tree structure is shown in Figure 2.10. The SS-tree is a variation of a R-tree that uses a sphere (instead of a rectangle) as a minimum bounding region [114]. The SR-tree is another variation of the R-tree that uses a combination of spheres (compact representation and bounding regions with smaller diameter) and rectangles (good in terms of volume at the leaves of the tree) as bounding regions [44]. The TV-tree can also be considered a variation of a R-tree [60]. In this structure, the minimum bounding region can be of any shape, depending on the application. Moreover, the vectors are allowed to contract or extend dynamically.

2.5.3 Metric access methods – MAMs

SAMs use the absolute spatial location of objects to partition and search a vectorial space. However, this information is not available in a general metric space. In this case, the only information available are the relative distances among objects. Because of this, metric access methods (MAMs) aims to partition the data space in regions by choosing representative elements and clustering the other elements around them [20].

MAMs can be classified in two main categories [20]: those based on discrete distance functions and those that deal with continuous distances. They also can be classified as static or dynamic, according to their support for insertion/deletion after the creation of the index. We will focus our discussion in MAMs that support continuous distance functions, as this is the case in color-based image retrieval systems. Next we will briefly describe some existing metric access methods. A good survey on this topic can be found in [20].

The Vantage-point Tree (VPT) is a MAM that recursively builds a binary tree ac-

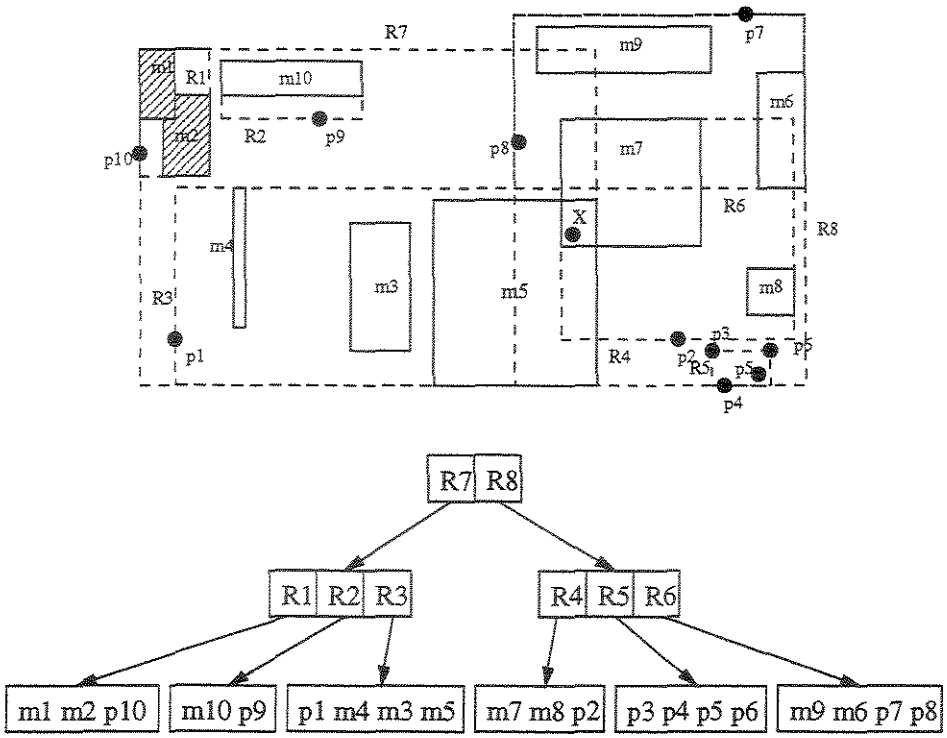

Figure 2.10: An example of R-tree organization

cording to a representative object called vantage point [119]. Elements are put at the left or right subtrees if their distances to the vantage point are smaller/higher than the median of all distances. The Multi-Vantage-Point tree (MVPT) extends the previous idea to m-ary trees using $m-1$ percentiles instead of just the median [15]. The Bisector-tree (BST) is also a binary tree constructed using two points c_1 and c_2 called centers [43]. The elements closer to c_1 are stored at the left subtree and those closer to c_2 are stored at the right subtree. For each of these points, it is also stored its covering radius (the maximum distance between the center and the elements in its associated subtree). The Generalized-Hyperplane Tree (GHT) is similar to the BST. The main difference is that it is used the hyperplane between c_1 and c_2 (instead of the query radius) as the pruning criterion at query time [111]. The Geometric Near-Neighbor Access Tree (GNAT) extends the GHT to an m-ary tree [16]. All MAMs discussed above are static in the sense that they do not support insertions/deletions after the index is created.

The first dynamic MAM proposed was the M-tree [23]. The M-tree is an m-ary height-balanced tree projected to reduce both I/O and distance computations. It aims at combining advantages of balanced and dynamic SAMs with the capabilities of static MAMs to index objects using features and distance functions that do not fit into a vector

space. The SLIM-tree is a variation of the M-tree that uses a new splitting algorithm based on the concept of minimum-spanning tree, and a new algorithm to reorganize a metric-tree in order to reduce the degree of overlap between nodes at the same tree level [108].

2.5.4 Approximate and non-metric methods

In some applications, the precision of a query can be relaxed to reduce the query processing time. The notion of *exact similarity search* is replaced by the notion of *approximate similarity search*, based on approximate or probabilistic algorithms. There are also approximate methods for the indexing of non-metric spaces.

The defenders of approximate similarity search argument that this is just another approximation step introduced into a process where there are several approximations: (1) the visual features used to represent and compare images is an approximation of the visual content of the images; (2) the metric space used to model the similarity between images is an approximation of the human perception of similarity; (3) the retrieval threshold used during a query processing is also an approximation for the similarity of relevant images. In non-critical applications, it is not necessary to pay the high price of an exact search, as it is acceptable to miss a (small) fraction of the target objects introducing one more approximation in a completely approximated process.

Approximate search algorithms make extensive use of clustering techniques to classify similar objects together [51, 56, 71]. Some approaches like [51, 71] use the triangular inequality to reduce the search space even if the distance used to compare two objects does not satisfy this metric property. Other approaches perform only a local search in the disk block where the query objects reside. It is also common to use a traditional access method like M-tree together with a precision parameter which controls the degree of approximation used in the search algorithm [20].

2.6 Existing CBIR approaches

In this section, some existing color-based CBIR approaches are discussed and classified into three main groups: (1) global approaches, (2) partition-based approaches and (3) regional approaches. This classification is based on the representation adopted for the color information present in an image, as discussed in Section 2.3.

The category of global approaches is, in general, the most efficient one in terms of visual features extraction, space overhead, and comparisons of images. However, the absence of spatial and topological information is an important limitation that affects directly retrieval effectiveness. In the other extreme are regional approaches, based on complex

image processing techniques to decompose images into regions of high color-wise similarity. These approaches require complex algorithms to extract visual features, use complex distance functions to compare images and implies high space overhead. Nonetheless, in general, the retrieval effectiveness is improved considerably. In between these two categories are the partition-based approaches. These approaches decompose images using a simple fixed strategy, usually based on a grid of rectangular cells superimposed over the images. In general, both the efficiency and the effectiveness of partition-based approaches are a compromise between global and regional approaches. The main exception to this rule is the space overhead, which is bound to be large for partition-based approaches.

Besides the tradeoff of efficiency and effectiveness, each of these categories has some desirable characteristics and also important limitations. Not a single approach is the best for all applications. For instance, in some specific situations, the use of a simple global color histogram plus an L_p geometric distance can be more effective than the use of complex regional approaches.

2.6.1 Global CBIR approaches

The most simple and well-known approach to color-based image retrieval consists in uniformly quantizing the RGB color-space (typically into 64 colors), representing the color distribution of images by means of global color histograms – *GCH*s, mapping these *GCH*s into a k-dimensional vectorial space, comparing the *GCH*s using the L_1 geometric distance, and indexing the obtained vectorial space using spatial access methods – SAMs.

The main advantages of this basic approach are that it is very efficient in terms of visual features extraction, representation and comparison, and the adopted representation is also invariant to image rotation and translation. This invariance is a necessary condition in some application domains. However, this basic approach has also many well-known limitations.

The most important limitation of the histogram representation is that it does not have any kind of information about the spatial distribution of colors. Images with very different spatial layout may have similar representations, specially in large collections of images. Another problem is that, although a histogram can be viewed as a k-dimensional vector, there are correlations between the spatial coordinates of this vector, as the colors represented by each histogram bin have different degrees of color-similarity (color crosstalk). Thus, it is possible that two images with similar (but not exactly the same) colors have maximum distance according to a geometric distance between their color histograms. There are also problems related to the color-space quantization and the storage requirements.

One alternative to deal with the color crosstalk phenomenon is to compare histograms

using distances that exploit the correlations between histograms bins, such as the weighted Euclidean distance [13, 61]. However, when this distance is adopted to compare histograms, it is not possible to index the visual features using SAMs, as these methods are unable to treat correlations among spatial dimensions. In this case, one alternative to index the visual features is the use of metric access methods – MAMs.

Another possibility to deal with the problem of color crosstalk is to exploit alternative representations for the color distribution. Stricker and Orengo [106] proposed two of such representations. The first one is the cumulative color histogram, which exploits the fact that the color similarity between two nearby histogram bins should be bigger than the color similarity between two further separated bins. The idea of cumulative histograms was improved by Zhang et al [122].

The second approach presented by Stricker and Orengo [106], instead of representing the complete color distribution, represents only its dominant features, via the first three moments of each color channel. This alternative representation also reduces considerably the space overhead when compared to color histograms. Dimai [29] also proposed a compact representation of colors which deal with the problem of colors crosstalk. In his approach, an image is represented by its average color and the covariance matrix of the color channels. The statistic methods discussed above have the advantage that they are computationally simple, and avoid the quantization of the color-space. However, images composed by colors completely distinct can have the same color statistics.

The problem of color-space quantization is related to the fact that the uniform quantization of a non-uniform color-space such as RGB is not the most adequate alternative, as perceptually similar colors can be classified apart and non-similar colors grouped together. One alternative to deal with this problem is the use of perceptually uniform color-spaces such as the CIE Lab. Another possibility is the use of non-uniform quantization schemes such as the ones proposed in [3, 88].

Chitkara [22] proposed a technique to deal with the problem of the high storage requirements of histograms. He observed that, after color quantization, images usually exhibit a low number of colors, and most of those cover less than 10% of the image area. Assuming that the human visual perception of colors follows a log-like scale, he proposed a non-uniform discretization of the *GCH* bins in order to encode each color bin into a bit-string. A careful representation of those bit-strings may reduce significantly the *GCH*'s space overhead.

Finally, the problem of the lack of information about the spatial location of colors was addressed by Pass et al [75]. They proposed classifying each pixel of a histogram bin as either coherent or incoherent depending upon whether the pixel is part of a large, connected, and similarly-colored region. The resulting structure is called Color-Coherence Vector – *CCV*. With the same purpose, Chen and Wong [21] proposed an augmented

color histogram that captures the spatial distribution of pixels in addition to the color distribution. The spatial information is incorporated by computing features from the spatial distance between pixels belonging to the same intensity color. The mean, variance and entropy of the distances are computed to form an augmented image histogram.

2.6.2 Partition-based CBIR approaches

The basic partition-based image retrieval system consists in decomposing images using a simple fixed strategy based on a grid of rectangular cells (3×3, 4×4) superimposed over the images, in such a way that all cells have the same size and do not overlap. The color-space is uniformly quantized as in the basic global approach described in previous section, and the visual content of each partition cell is described by means of a local color histogram – LCH. The distance of two images is computed as an average of the distances between the LCHs of equivalent cells (cells at the same spatial position).

Like global approaches, the partition-based approaches vary in terms of the underlying color-space, the color-space quantization scheme, the chosen representation for the color information an the distance used to compare images. However, in partition-based approaches, there is also the possibility of exploiting alternative partition schemes. Although the decomposed representation of the color information adopted in partition-based approaches has the advantage to spatially locating colors inside the image, at the same time, it introduces some new limitations related to the cell crosstalk phenomenon, to the sensitivity to rotation and/or translation of images, to the sensitivity to the absolute spatial location of image objects and finally to the increasing in space overhead when compared to global approaches.

The main problem of our basic partition-based approach is that the distance between distinct partition cells are not considered when comparing two images. So, it is possible that two similar images whose objects are in different positions have the maximum possible distance. This problem is similar to the color crosstalk phenomenon when comparing color histograms; hence we call this the *cell-crosstalk* phenomenon.

The problems of cell crosstalk, sensitivity to image rotation and/or translation and sensitivity to absolute spatial location of objects are all related and can be addressed in two distinct ways. One alternative is to exploit more complex distance functions that, instead of comparing only the corresponding cells between two images, perform a more elaborate comparison in order to define the best matching of cells. One possible solution to this problem is to compare the content of every pair of cells, weighting the distances by the spatial distance between them. One such approach was presented by Wang [113], where each cell of each image is modeled as a node in a bipartite graph with the edge cost being the color-distance between cells. The best matching of cells is the solution of

the corresponding assignment problem [52].

The second and more commonly used solution for the problem of cell crosstalk consists of adopting a hierarchical representation of the spatial decomposition [38, 55, 64, 87]. In general, this hierarchy is based on a quadtree structure [91]. At the top of the hierarchy, there is a global representation of the image content that does not suffer from spatial limitations. At the second level, the image representation is decomposed in 2×2 cells. At the third level, the representation is decomposed in 4×4 cells, and so on. The cells of distinct levels have different sizes and overlaps, minimizing the spatial problems identified above. The comparison of two images is performed initially at the top of this hierarchy and then refined in subsequent levels. The hierarchical representation of the partition structure implies a great increasing in terms of storage requirements.

2.6.3 Regional CBIR approaches

Regional CBIR systems are based on segmentation techniques to decompose images according to their visual content. The segmentation of the images is more flexible and robust than the fixed scheme adopted in partition-based approaches. However, the comparison of segmented images is a very difficult problem because of inaccurate segmentation [58], an inherent characteristic of fully automatic regional CBIR systems. The most common approach in regional systems is to compare the regions of the images individually, as in Blobworld system [18]. Recently, in order to reduce the influence of inaccurate segmentation, systems like SIMPLIcity [58] start comparing images according to the properties of all segmented images, not only in a region-by-region basis. Next we will discuss the characteristics of some existing regional approaches.

The IBM QBIC system [7] is based on a clustering process where two clusters of pixels are merged if their mutual rank falls bellow a predefined threshold. The mutual rank of clusters P and Q is $n + m$, where Q is the nth closest cluster to P and P is the mth closest cluster to Q. The distance between two clusters is measured as the Euclidean distance between their mean colors. For each color obtained after the clustering process, the connected components of the pixel population having that color are identified and, for each connected component, a bounding rectangle is calculated. The bounding rectangles of a given color are successively clustered into groups of geometrically close rectangles until one rectangle remains. The result is a hierarchical tree structure for each color. The distance between two regions is calculated as a weighted sum of the distance between the colors themselves and the distance between their associated tree. The distance between two images I and J is the average of the distances between each region of I and its closest region in J.

The Netra II system [28] uses a boundary detection algorithm called EdgeFlow to

segment images. A perceptual color quantization scheme is then used to quantize the colors in each image region. A color histogram is computed for the colors obtained after the quantization step. The color histogram of two regions is compared using a distance equivalent to the weighted Euclidean distance [13]. Each quantized color of a region is indexed individually in a 3D space. The corresponding percentage and the label of the region are stored along with the quantized colors. Instead of using a SAM to index each quantized color, it is proposed an alternative structure called Lattice, where a set of reference points (Lattices) in the 3D space are chosen *a priori* and the quantized colors to be indexed are assigned to their nearest lattice point.

The Blobworld system [18] clusters pixels in a joint color-texture-position 8D space modeled as a mixture of Gaussians. The color of each region is represented as a 500 bins local color histogram (in Lab color-space). To compare the color of two regions, it is used the weighted Euclidean distance [13]. Images are compared based on individual regions. Although querying based on a limited number of regions is allowed, the query is performed by merging single-region query results.

The SIMPLIcity system [58] segments images based on color and frequency features using the *k-means* algorithm to cluster the feature vectors into classes. Each class corresponds to a region in the segmented image. Images are compared using the properties of all segmented regions, according to the *IRM* (Integrated Region Matching) similarity measure. Initially, the *IRM* similarity measure matches regions of the two images. The match process allows one region of an image to be matched to several regions of another image. After regions are matched, the similarity measure is computed as a weighted sum of the similarity between region pairs, with weights defined by a significance matrix.

2.7 Evaluation of retrieval effectiveness

Once a new CBIR approach is conceived, it is necessary to evaluate its performance. In data retrieval systems, the response time and the space required are usually the metrics adopted for evaluating a new system. In the domain of information retrieval, however, there is the additional issue of evaluating the relevance of the information retrieved (effectiveness). Effectiveness evaluation is a very complex task. For the purpose of effectiveness evaluation in text-based retrieval, there are several reference collections available (e.g., CACM, ADI, INSPEC, Medilars and ISI) and even a full conference (TREC) dedicated to the issue [118]. Thus, there is a wealth of reference experiments, uniform scoring procedures, and forums for researchers interested in comparing their results using a common framework.

Unfortunately, in the domain of CBIR systems the situation is quite different. The CBIR community has not been nearly as active in this regard and has used relatively small

and widely different test collections in its experiments. Comparisons between various CBIR systems are difficult to make because distinct groups conduct experiments focused on distinct aspects of retrieval (even when the same test collection is used) and there are no widely accepted benchmarks. Some efforts toward a benchmark for CBIR have been made by Muller et al [66], Gunther and Beretta [39] and Leung and Ip [54]. The main problems to obtain a benchmark for CBIR are (1) the construction of a reference database of images without copyright restrictions, (2) judging the relevance of the database images for a set of reference visual queries (ground truth) and (3) the evaluation of retrieval effectiveness.

The effectiveness of a retrieval system is a measure related to the user's satisfaction with the system output. In establishing measures of effectiveness, the first decision to make is the number of levels of judgment allowed for the user in this evaluation [48]. The basic choice is between a *binary* and an *n-ary* measure. A binary measure is the simplest to implement and to use. Each image is either accepted or rejected. This acceptance or rejection is usually couched in terms of the *relevance* of the image to the user. However, relevance is itself an ill defined term, as it has a degree of psychological subjectiveness. Different users, or even the same user under other circumstances, may perceive the visual content of an image in a different way. A system is judged to be effective if satisfactory evaluation results are obtained using an external relevance criteria [82]. Moving beyond a binary measure to a n-ary one allows the user to consider levels or degrees of relevance. While the choice of a scale for relevance is open, the scale for retrieval is closed: either a document is retrieved or it is not [48].

Almost all effectiveness measures used in CBIR systems were originally designed to evaluate textual information retrieval systems. The use of these measures in CBIR systems is acceptable, as the main purpose in both kinds of systems is to evaluate the ranking algorithm according to an external judgment of relevance. The external judgment of relevance is inherently different for images an textual documents, however this judgment is assumed to be correct, and is not the issue under evaluation.

2.7.1 Precision and Recall

Among the large variety of existing retrieval effectiveness measures, Precision vs. Recall ($P \times R$) curves [118, 48, 82, 115] are the most well-known and used measure in practice. Although they are not the most adequate measure in the context of ranked output [32, 48, 82], they have been widely used also in evaluation of CBIR systems. The main problem with these curves is that they do not characterize adequately the ranked output of CBIR systems. They are more adequate to systems that produces an unordered set of documents which are either relevant or not.

When binary scales are used for both relevance and retrieval, a 2×2 contingency table (Table 2.1) can be established showing how the image collection is divided by these two classifications [48].

According to this contingency table, the precision P_r is the fraction of the r retrieved images that are relevant to the query, that is:

$$P_r = \frac{A}{r} = \frac{A}{A+C} \tag{2.2}$$

Table 2.1: Contingency table for evaluating retrieval effectiveness

	Retrieved	**Not Retrieved**
Relevant	A	B
Not relevant	C	D

While precision measures the accuracy of the search, recall measures the extent to which the retrieval is exhaustive. The recall R_r of a method is the proportion of the total number of relevant images that were retrieved among the r returned images, namely:

$$R_r = \frac{A}{A+B} \tag{2.3}$$

The recall measurement requires knowledge about the total number of relevant images within the collection. By definition, it is a non-decreasing function of the rank of the retrieved images [115]. If 50 images are retrieved as the answer to some query, and 35 of them are relevant, the precision at $r = 50$ is $P_{50} = 70\%$. If, on the same query as before, there are 70 relevant images within the image collection, the recall at $r = 50$ is $R_{50} = 50\%$, since 35 out of 70 of the relevant images were selected within the top 50 retrieved images.

Some retrieval systems can produce varying amounts of output, and a recall-precision pair can be computed for each retrieved image. Given a set of recall-precision pairs, a recall-precision curve can be constructed by plotting the precision against the recall. In general, the curve closest to the upper right-hand corner of the curve (where recall and precision are both maximized) indicates the best performance. If interpolated precision values are used, the curve is non-increasing. The interpolated precision at a given point is the maximum precision at this and at all previous recall levels [115]. The interpolated curve is a smoothed version of the original curve that represents the best performance a user can achieve [46].

2.7.2 Single-valued measures

Some observers postulate that a retrieval effectiveness measure should be expressible as a single number (instead of two values such as $P \times R$) that can be put on a scale to give absolute and relative values [82]. Measures such as E measure [81] and the MZ metric [89] combine aspects of precision and recall into a single effectiveness value. Other single-value measures uses a single point sampled from the $P \times R$ curve as an effectiveness descriptor, such as [115]: (1) the precision at the minimum point at which recall could be 100% (R-value), or (2) the precision when the first relevant image is retrieved, or (3) the precision at a fixed recall level such as 10% or 20%, or (4) the precision at specific rank values such as after 30 or 100 images are retrieved. It is also common to compute a single number that characterizes the effectiveness at all recall levels, such as 3-point or 11-point average precision [115]. All variations of the $P \times R$ curves discussed above provide very specific effectiveness information and thus, have a limited context of application. However, the use of several of those measures in addition to a $P \times R$ graph gives a clear characterization of the retrieval process according to different viewpoints.

Ranking algorithms are at the core of CBIR systems and attempt to establish a simple ordering of the retrieved images. Images appearing at the top of this ordering are considered to be more likely to be relevant. Images are presented and examined sequentially by the user in order to decide about their relevance. One of the first measures especially designed to the context of ranked output was the *normalized recall* - R_{norm} [82]. The R_{norm} value reflects the number of nonrelevant images that have to be retrieved in order to reach a recall value of 100%. The main problem of this measure is that the effectiveness result is dependent of the size of the collection. The larger the collection, the smaller the numerical difference between the effectiveness of two systems, even when using the same set of query images.

A variation of the R_{norm} measure proposed in QBIC project [32] uses the ratio between the average rank of the relevant images and an ideal average rank (where all relevant images appear ahead of the non-relevant ones) to measure the effectiveness of CBIR systems. This ratio shows how close to the top of the ranked output the set of relevant images appear. The main problem with this measure is that the average rank of relevant images is very sensitive to the rank of the last relevant images retrieved (large numerical values). Another problem is that the process of averaging the results of several distinct queries is also sensitive to the worse queries, since they are bound to have much larger numerical values.

There are also other single-valued measures to evaluate ranking algorithms as, for example, the *expected search length* [24], the *sliding ratio* [79] and the *satisfaction/frustration* [48] measures. The expected search length is a measure that assumes that the images are presented to the user in a weakly ordered sequence. The sliding ratio measure is based on

the comparison of two ranked lists of items. One list is the output of an actual retrieval system, and the other represents an ideal system in which the items are ranked in decreasing relevance order. This model is more complex than the ones previously described because it allows the assignment of numeric relevance weights to the images. The sliding ratio measure has been further refined by Myaeng and Korfhage [67]. They separated out the relevant and irrelevant images and defined two measures: (1) *satisfaction*, which considers only the relevant images and (2) *frustration*, which considers only the non-relevant images. They also proposed a weighted combination of satisfaction and frustration.

Capítulo 3

CCH – Cell/Color Histograms

Color is a commonly used feature for realizing content-based image retrieval (CBIR). In this context, this chapter presents a new approach for CBIR that is based on the well known and widely used color histograms[1]. Previous approaches have used a single global color histogram (*GCH*) for the whole image, or local color histograms (LCHs) for cells within a grid of fixed size. Our approach is also based on a grid of cells, but unlike the latter it uses a cell histogram for each of the colors actually present in the images, representing how that color is distributed among the image cells – thus the name *Cell/Color Histograms*. Our experiments have shown that the actual number of colors present in images is often low. Thus we are able to achieve performance comparable to using LCHs within a grid, but with a much smaller space overhead. Furthermore, the proposed approach is very flexible in the sense that the user has alternative ways to calibrate the trade-off between space overhead and retrieval effectiveness. In fact, we have been able to outperform *GCH*s (typically a compact representation) in terms of effectiveness, requiring less storage space.

The remainder of this chapter is organized as follows. Section 3.1 presents our approach for CBIR, named Cell/Color Histograms – *CCH*, which is more compact, robust and flexible than those discussed in Sections 2.6.1 and 2.6.2. Section 3.2 presents a generalization of the L_1 distance that is used to compare images in all approaches included in our comparative analysis. Section 3.3 discusses how we evaluate retrieval effectiveness in our experiments and Section 3.4 presents our experimental results. Finally, Section 3.5 presents the chapter conclusions.

[1]This chapter will be published in the *"Knowledge and Information Systems International Journal"* [104].

3.1 Cell/Color Histograms – CCH

Our main contribution in this chapter is a compact and flexible representation for partition-based CBIR approaches. Our motivation is to reduce the space overhead of these approaches taking advantage of the fact that only a relatively low number of distinct values of a particular visual feature are present in most images. In particular, we adopt the color feature represented by means of histograms to describe images. However, it is possible to encode any other visual feature of an image under the same principle.

Our proposal contrasts with those that exploit only alternative representations for the visual features in order to reduce space overhead. Consider a technique that yields a very compact representation for the color distribution of an image. Such a technique can be applied directly to represent the content of the whole image or the content of each image cell individually, after an appropriate spatial partitioning. However, if the global representation requires n bytes, the partition-based representation will require $O(mn)$ bytes, where m is the number of partition cells in which the image is decomposed. The increasing factor in space overhead is constant, independently of the chosen compact representation. In our approach, the space overhead reduction is not obtained via compact color representation, but through exploiting a more elegant representation for the spatial partition structure as a whole. Thus, it can be applied in any partition-based CBIR approach, independently of the chosen color-space, quantization scheme, distance function, or color representation. These issues will be discussed in more details at the end of this section.

In our approach, we exploit the fact that only a relatively low number of distinct quantized colors are present in most images. The graph shown in Figure 3.1 confirms our intuition. This graph was obtained using a collection of 20,000 heterogeneous JPEG images and the RGB color-space uniformly quantized in 64 colors (a typical quantization scheme). This graph shows how much of an image is covered by a given number of colors. In the average, there were only 28.71 colors per image from a total of 64 quantized colors. Moreover, about 90% of the image content corresponds to only 9 colors. These values show that at least 55% of a color histogram has null bins and that one is able to describe 90% of the color distribution of the image by using only 14% of such bins. Observe also that, if a grid of cells is adopted to spatially decompose an image and a local color histogram is used to describe the content of each cell individually, the amount of null bins grows proportionally to the number of cells. Hereafter, we refer to such generic partition-based CBIR approach as the *Grid* approach.

In order to avoid the explicit representation of quantized colors not present in an image, we propose to represent the partition structure in an alternative perspective. Assuming that the number of cells is fixed for a partition scheme and that the number of quantized colors present in a given image is not, we propose to describe the spatial distribution

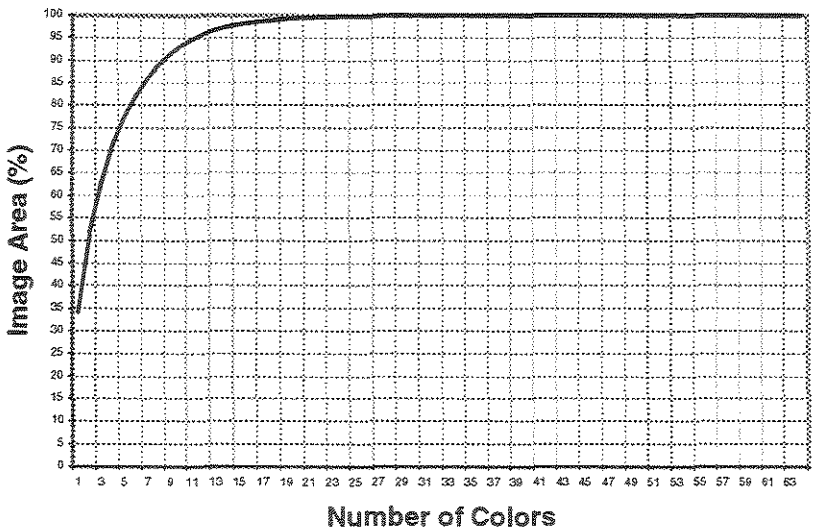

Figure 3.1: Color contribution for image content in our collection of heterogeneous images.

of each individual color through the partition cells, instead of representing the color distribution of each individual cell. Thus, we propose to represent the partition structure by a set of *cell* histograms instead of a set of (local) *color* histograms.

A cell histogram is formally defined as follows. Consider an image partitioned into $I \times J$ non-overlapping cells and a color-space uniformly quantized in C colors. A *cell histogram* for a given color c, $0 \leq c < C$, is a set of $I \times J$ bins (one for each partition cell). The value of each histogram bin is given by the function $p(cell_k) = n_k/n$. In this function, $cell_k$ is the kth partition cell ($0 \leq k < I \times J$), n_k is the number of pixels in $cell_k$ with color c, and n is the number of image pixels. An image composed by m quantized colors is thus described by m cell histograms, each one describing the spatial distribution of one color.

We call the set of cell histograms used to describe an image *Cell/Color Histograms* or *CCH* for short. It is important to notice that, when using the *CCH* representation, if a color is not present in the image, there is no cell histogram associated to it, hence we save storage space. Compare this to the *Grid* approach representation, where one would need necessarily to store $I \times J$ histograms, each with C bins, regardless of how many colors are actually present in the image. Figure 3.2 illustrates this discussion by showing an image composed by two colors with a 2×2 grid of cells superimposed, and its *CCH* representation.

The *CCH* approach combines in a compact, flexible and elegant way the color dis-

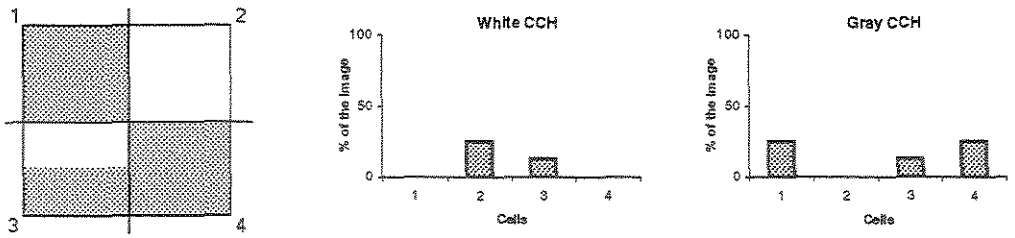

Figure 3.2: An image partitioned using a 2×2 grid of cells and its *CCH* representation.

tribution of an image with the spatial distribution of each color. Our approach is more compact in the sense that only colors present in the image are represented. It is flexible because the task of performing color exclusion (Section 3.4, experiment III) becomes more natural in the global representation of cell histograms than in the local representation of local color histograms. Moreover, *CCH* allows the adoption of alternative types of similarity metrics to compare images, as discussed in Section 3.2. Finally, it offers various opportunities for trade-off between space overhead and retrieval effectiveness, as shown in Section 3.4, experiment IV.

The cell histograms have all desirable characteristics and limitations of traditional histograms, as discussed in Section 2.6.1. All techniques so far discussed to overcome histogram limitations can be equally applied to our cell histograms. It is possible to use more compact representations, represent the cell histograms as cumulative histograms, or use compression techniques to reduce their space overhead. Compression techniques are equally useful for cell histograms because the probability that all colors are present in all partition cells is as small as the probability that all quantized colors are present in the image.

It is possible to use our *CCH* approach in conjunction with any existing color-space and quantization scheme. When using dynamic color-space quantization schemes, the *CCH* approach becomes even more compact, because there is no need to replicate the explicit representation of the quantized colors in each individual cell. Moreover, the *CCH* representation can be used in conjunction with more robust partition-based techniques. It is possible to adopt alternative partition schemes, a hierarchical representation for *CCH*, and also more complex and robust distance functions to compare two *CCH* representations.

3.2 Similarity metric

In this section, we discuss how to extend a traditional similarity metric used to compare images to the variable-size representation of *CCH*. As we will see, all that is required is a

generalization of the metric definition.

We will adopt the L_1 metric distance to demonstrate the generalization process. In Section 3.3 this metric is used to compare images in all compared CBIR approaches. The L_1 distance definition for a multi-histogram representation is shown in Equation 3.1.

$$D(h_q[i], h_d[i]) = \sum_{j=1}^{m} |h_q[i][j] - h_d[i][j]| \tag{3.1}$$

where m represents the number of histogram bins. The values $h_q[i][j]$ and $h_d[i][j]$ represent the normalized value of the jth bin of the ith histogram used to describe the query image (h_q) and the database image (h_d), respectively. In the case of a *GCH*, there is only one histogram to be compared. In traditional partition-based approaches such as the *Grid* approach discussed in the previous section, there are a fixed number of local color histograms, one for each partition cell. In *CCH* approach, there are a variable number of histograms per image, depending on the number of quantized colors actually present in the image.

We are assuming that the histogram bins are normalized with respect to the image size, i.e. to the number of image pixels. In this way, the sum of the histogram bins is at most 1. This limit occurs when the area represented by the histogram equals the image size, i.e., when we are using a global histogram. Moreover, the distance between two histograms $D(h_q[i], h_d[i])$ is at most 2. The distance limit may occur only when two completely distinct global histograms are compared.

A stricter limit for the distance between any pair of histograms can be established in the following way. Let $a_q[i]$ be the image area described by the $h_q[i]$ histogram and $a_d[i]$ be the image area described by the $h_d[i]$ histogram. Consider that these values are also normalized according to the image size. When these two histograms are compared, we obtain $D(h_q[i], h_d[i]) \leq a_q[i] + a_d[i]$. Thus, in order to normalize the distance between two histograms $D(h_q[i], h_d[i])$, we divide this distance by $a_q[i] + a_d[i]$, as shown in Equation 3.2.

$$D_n(h_q[i], h_d[i]) = \frac{D(h_q[i], h_d[i])}{a_q[i] + a_d[i]} \tag{3.2}$$

So far, D_n measures the normalized distance between two histograms. The similarity between two histograms is then the complement of the distance D_n. Finally, the similarity S between two images (Equation 3.3) is the weighted sum of the similarity between the histograms that describe each image:

$$S(h_q, h_d) = \sum_{i=1}^{n} w[i] \times (1 - D_n(h_q[i], h_d[i]) \tag{3.3}$$

The weight values are used to normalize the similarity between two images and are related to the image area described by each pair of compared histograms. Three possibilities for the weight function are:

- $w_1[i] = \frac{1}{n}$, where n is the number of histograms used to represent an image
- $w_2[i] = min(a_q[i], a_d[i])$
- $w_3[i] = a_q[i]$

If $a_q[i] = a_d[i]$ for every i, these three weight functions are identical. This is the case when we compare *GCHs*, where $n = 1$ and $a_q[i] = a_d[i] = 1$, and also when we compare traditional partition-based approaches such as *Grid*, where $a_q[i] = a_d[i] = 1/n$ for every i. However, for the *CCH* approach, these three weight functions are distinct, since the number of histograms used to represent an image is variable, as well as the image area covered by each histogram. Both depend on the actual color distribution of the compared images. The w_1 function results in a simple arithmetic mean of the distance between histograms. The w_3 function is not symmetric and thus, the distance between two images does not satisfy the symmetry axiom for a metric. Because of this, we choose to work with the function w_2 in our experiments. It is a symmetric function and we have observed in practice that the effectiveness results obtained with this function are better than the w_1 results.

The two images in Figure 3.3 will be used to exemplify the application of the similarity metric in three histogram-based CBIR approaches: *GCH*, *Grid* and *CCH*. For simplicity, we divide the images only into 4 cells (2×2 grid) in order to spatially locate colors. The cells are compared from top to bottom, left to right, and the color space has only three colors: black, gray and white, represented by the numbers 1, 2 and 3, respectively. In Figure 3.3, q is the query image, and d is the database image to be compared against q. Figure 3.4 shows the visual features (histograms) obtained from the query image by each CBIR approach. The top row shows the single *GCH*. The second row depicts the set of four LCHs of the *Grid* approach, one for each partition cell. The bottom of the figure shows the three cell histogram, one for each quantized color present in q. Similarly, Figure 3.5 shows the respective histograms for the database image (d).

The *GCH* for q could be represented as $h_q = [0.5, 0.25, 0.25]$ meaning that it has 50% of black, 25% of gray and 25% of white pixels, respectively. Similarly, $h_d = [0.5, 0.0, 0.5]$. Using Equation 3.3 we have:

$$S_{GCH}(q, d) = 1 \times (1 - \frac{|0.5 - 0.5| + |0.25 - 0.0| + |0.25 - 0.5|}{2}) = 0.75$$

In the *Grid* approach, the normalized distance for the first cell is $D_n^1 = 0$, because both cells have only black pixels. For the other three cells, the distances are $D_n^2 = 1, D_n^3 = 1$

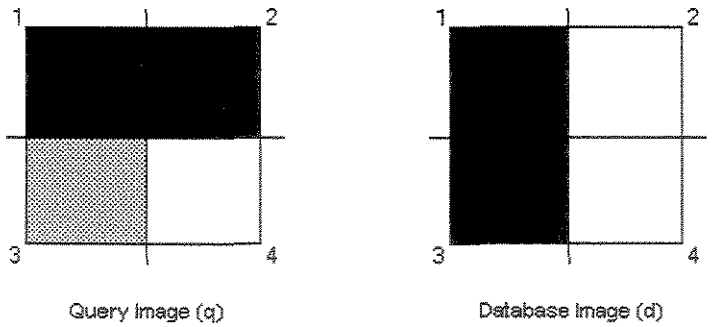

Figure 3.3: Sample images partitioned in 2×2 cells.

and $D_n^4 = 0$, respectively. In addition, the weights $w[i]$ are all the same, because all cells have the same relative area. Thus, we have:

$$S_{Grid}(q, d) = 0.25 \times (4 - (D_n^1 + D_n^2 + D_n^3 + D_n^4)) = 0.5$$

Next we compare the three *CCH* cell histograms (for black, gray and white colors) using Equation 3.2. We have 25% of black pixels in cells 1 and 2 of image q, and in cells 1 and 3 of image d (recall that the quantity of pixels in a cell is normalized with respect to the image size), hence:

$$D_n^{black} = \frac{|0.25 - 0.25| + |0.25 - 0| + |0 - 0.25| + |0 - 0|}{0.5 + 0.5} = 0.5$$

Likewise, we obtain $D_n^{gray} = 1$, and $D_n^{white} = 0.33$. Lastly, the normalized cell-histogram distances are complemented and weighted according to Equation 3.3 (notice that, unlike the *GCH* and the *Grid* approaches, the weights $w[i]$ are now variable, depending on the areas occupied by each color). The similarity between the two images according to CCH approach is:

$$S_{CCH}(q, d) = 0.5 \times (1 - 0.5) + 0 \times (1 - 1) + 0.25 \times (1 - 0.33) = 0.42$$

3.3 Evaluation of retrieval effectiveness

Once a new CBIR approach is conceived, it is necessary to evaluate its performance. In data retrieval systems, the response time and the space required are usually the parameters adopted for evaluating a new system. In the domain of information retrieval, however, there is the additional issue of evaluating the relevance of the retrieved information (effectiveness). In this paper, our focus is the evaluation of effectiveness vs. space

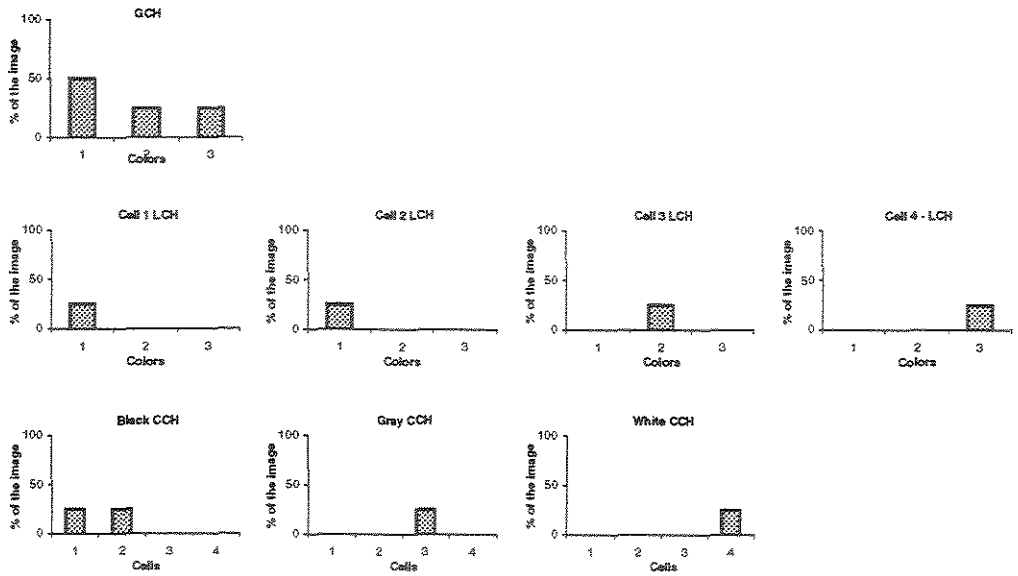

Figure 3.4: Histograms of image q (Figure 3.3) in different CBIR approaches.

overhead. The efficiency of the retrieval is an aspect that is tightly related to indexing structures/techniques, which is subject of further research.

Effectiveness evaluation is a very complex task. In order to evaluate CBIR effectiveness, it is necessary at least a reference collection of images, a set of query images, a set of relevant images (chosen *a priori*) for each query and an adequate retrieval effectiveness measure. Next we discuss how we deal with these requirements in our experiments.

Reference collection – In the context of large and heterogeneous image collections, a good reference collection to evaluate retrieval effectiveness should clearly be large enough to be heterogeneous, i.e., to contain several semantically and/or visually distinct image domains. Ideally, each of such domains should also be composed by clusters of images with similar visual characteristics (visual clusters). Corel Corp.[2] is a well-known manufacturer of image collections that follow such an approach. Incidentally their images are often used, though in an *ad-hoc* manner, to test new CBIR approaches. In our experiments we are using as reference a heterogeneous collection of 20,000 JPEG images from a Corel stock CD[3]. This collection is formed by approximately 200 distinct image domains, each one composed of approximately 100 images. We believe that is a sufficiently large number of distinct domains and images per domain for the purpose of our evaluation study.

[2] `http://www.corel.com`

[3] Corel GALLERY Magic 65,000 - Stock Photo Library 2

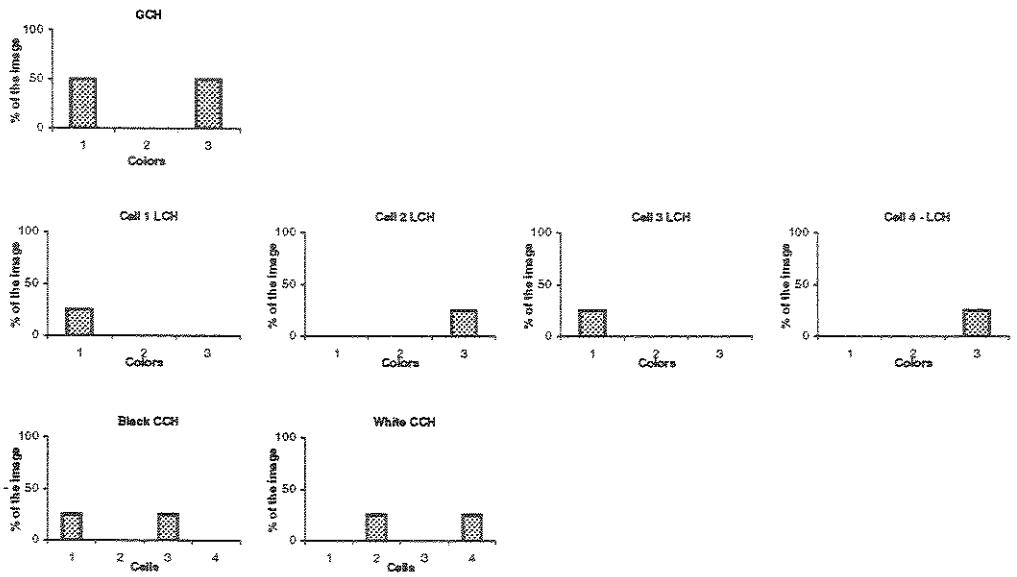

Figure 3.5: Histograms of image d (Figure 3.3) in different CBIR approaches.

Query images – Out of the reference collection, we selected 15 images of distinct domains to be used as query images. The set of query images is a subset of the images shown in Figure 7.1.

Relevant result sets (RRSets) – Once the query images are selected, the next step is to establish the set of images inside the reference collection that we accept as relevant for each query image. We call this set of relevant images the *relevant result set* (RRSet) of a query image. Given a query image, an ideal CBIR approach retrieves the images of its RRSet ahead of any other image within the reference collection. We selected the RRSet of a query image by visually analyzing the other images that belong to the same semantic domain of the query image. All images that, in addition to the semantic similarity, had also similar visual properties were chosen to compose the RRSet. Some examples of RRSets[4] are shown in Figures 7.2, 7.3 and 7.4.

3.3.1 Retrieval effectiveness measures

We evaluate retrieval effectiveness using a variation of the QBIC measure [32] discussed in Section 2.7.2 that we call θ_{abs}. The θ_{abs} measure is defined as follows:

[4]The sets of query images and respective RRSets can be seen at `http://www.cs.ualberta.ca/~mn/CBIRone`.

$$\theta_{abs} = \frac{\sum_{i=1}^{|RRSet|} \frac{i}{rank(i)}}{|RRSet|} \tag{3.4}$$

where $|RRSet|$ is the number of relevant images within the RRSet of a given query. The summation is performed following the order established by the ranking algorithm. Thus, i represents the ideal rank of the ith relevant image that was retrieved, and $rank(i)$ is a function that returns the actual rank of this i^{th} relevant image. The $rank(i)$ value varies in the interval $[1,|DB|]$, where $|DB|$ is the size (number of images) of the reference collection. As we can observe, $0 < \theta_{abs} \leq 1$ and $\theta_{abs} = 1$ represents an ideal system, i.e., a system where all relevant images are retrieved ahead of the non-relevant ones.

When performing a CBIR experiment, the effectiveness results of several distinct queries must be averaged in order to derive a single value that adequately describes the system effectiveness. During this averaging process, there are many implicit factors that must be considered in order to derive a coherent effectiveness value, such as the complexity of each query image, the proportion of relevant/non-relevant images inside the reference collection per query image, and the discriminatory power of the visual features relative to the chosen query image (assuming we are working with a collection of heterogeneous images). For example, searching for simple images composed just by one distinctive object and a homogeneous background is generally much more effective than searching for more complex images.

The implicit characteristics of a CBIR experiment are very hard to extract and to explicitly use during the measurement process, during the analysis of the results, or during the averaging of the results of multiple queries. In practice, we need an indirect way to normalize the results of individual queries according to these implicit characteristics. In order to do this, we propose the use of a well-known (and ideally effective) CBIR approach as a reference to derive a more robust measure. Assuming that the implicit characteristics of the CBIR experiment are the same for both, the approach being analyzed and for the reference approach, a relative result becomes more robust than an absolute effectiveness value obtained with the θ_{abs} measure. Thus, we propose the θ_{rel} measure, based on a reference CBIR approach. Namely, for a given query q and a reference approach denoted as *ref*, we have:

$$\theta_{rel}(q) = \frac{\theta_{abs}(q) - \theta_{abs}^{ref}(q)}{\theta_{abs}^{ref}(q)} \times 100 \tag{3.5}$$

The θ_{rel} value represents the percentage of gain (positive values) or loss (negative values) relative to the reference approach. The fact that the effectiveness results obtained with the θ_{rel} measure are normalized according to the experiment characteristics allows a comparison of results even when the reference collection, query images and RRSets are

not the same. All that is required is the adoption of a similar evaluation methodology and the use of the same reference CBIR approach.

In our experiments we adopted the *GCH* as our reference because it is simple, well known, widely used and, in general, effective for color-based image retrieval.

3.4 Experimental results

This section presents the results of four experiments we performed in order to demonstrate the compactness, robustness and flexibility of the *CCH* approach in terms of space overhead and retrieval effectiveness.

The first experiment compares *CCH* with three other color-based CBIR approaches. Our goal is to contextualize the results obtained with *CCH*. The second and third experiments exploit the flexibility of *CCH* representation. As argued earlier, one of our main goals with the *CCH* is to reduce the space overhead of partition-based CBIR approaches. We exploit this issue in these two experiments, analyzing how the spatial partition (number of cells) and a partial representation of the visual content of the images affect both the space overhead and the retrieval effectiveness. The last experiment combines the other three, in order to demonstrate how one can finely tune the *CCH* approach, analyzing the various opportunities for trade-off between space overhead and effectiveness it offers.

3.4.1 Experiment I – Comparison with traditional approaches

This first experiment compares, in terms of retrieval effectiveness and space overhead, *CCH* against three other color-based CBIR approaches discussed in Section 2.6: (1) *GCH*– our reference approach, (2) *CCV* and (3) *Grid*.

The four compared approaches (*CCH*, *CCV*, *Grid* and *GCH*) are color-based and adopt a histogram representation for the visual features extracted from images. As discussed in Section 2.6.1, the histogram representation has some limitations, and there are many techniques that deal with these limitations in CBIR literature. These techniques can be applied with success to all four compared approaches, even the *CCH*. In this paper however, it is not our goal to analyze the effectiveness of these techniques and the effects of adopting these techniques in distinct CBIR approaches. Although this topic is part of our future research, some discussion was already taken at the end of Section 3.1.

The goal of this first experiment is simply to contextualize the results obtained with *CCH* by showing that, in a common scenario (in terms of color-space, quantization schemes, distance function, partition scheme, color representation, etc.), our approach is more effective than similar global approaches. Moreover, it is as effective as equivalent partition-based approaches, with a considerable gain in space overhead obtained without

the help of any other technique (such as histogram compression) than our alternative representation of the partition structure.

In order to perform a fair and robust comparison, we adopted (where applicable) the same set of simple and widely used parameters. In this way, the source of the space overhead and of the retrieval effectiveness in each approach becomes evident. We adopted the RGB color-space uniformly quantized in 64 colors. The spatial partition of the images was obtained using a fixed 8×8 grid of equal-size, non-overlapping cells. In all approaches, images were compared using the distance function discussed in Section 3.2, based on the w_2 weight function.

With the parameters above, the *GCH* of an image has 64 bins, one for each quantized color. The *CCV* representation results in 128 bins, two bins for each of the quantized colors. Using *Grid*, each image is described by 8×8=64 local color histograms, each with 64 bins. In *CCH*, each cell histogram has 8×8=64 bins, one for each partition cell. The maximum number of cell histograms is 64, one for each quantized color. However, for our reference collection of images, less than 29 cell histograms (in average) were needed per image (Figure 3.1). Therefore, the *CCH* representation requires in average 55% less space than the *Grid* representation, our representative of traditional partition-based approaches.

As we can see in Table 3.1, the *CCV* effectiveness is 10% (θ_{rel} value) higher than the *GCH* (reference approach) effectiveness, whereas the *Grid* effectiveness is 95% higher. The effectiveness gain obtained with *CCH* is slightly worse than the gain obtained with *Grid*. The $P \times R$ curves of the first experiment can be viewed in Figure 3.6. As one can see, these curves are consistent with the values obtained with the θ_{rel} effectiveness measure (Table 3.1).

Table 3.1: Effectiveness and space overhead values for the first experiment.

Approach	Number of bins	θ_{abs}	θ_{rel}
GCH	64	0.36	—
CCV	128	0.40	10%
Grid	4096	0.59	95%
CCH	1856*	0.59	89%

*Average value

3.4.2 Experiment II – Reducing the number of partition cells

In this experiment, we exploit how the reduction in the number of partition cells affects the effectiveness of *CCH* approach. We compared the use of 8×8, 6×6, 4×4, 3×3 and

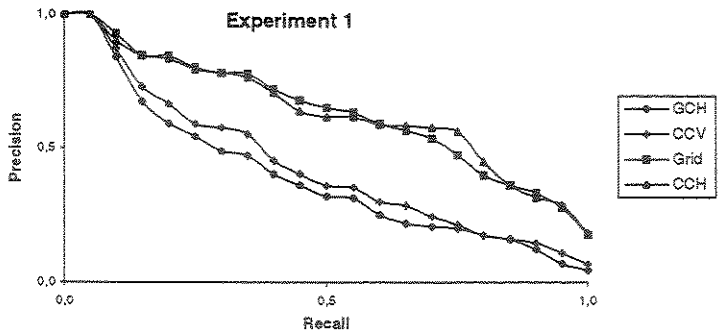

Figure 3.6: Precision vs. Recall curves for Experiment I.

2×2 grids of equal-sized non-overlapping cells. Obviously, the coarser the grid of cells the smaller the space overhead, but the trade-off is evident in terms of retrieval effectiveness. The results of Table 3.2 show that, as the number of cells decreases, the retrieval effectiveness also decreases due to the loss of spatial information. There is less information to distinguish images, increasing the number of false-hits, and thus decreasing the retrieval effectiveness. The $P \times R$ curves showed in Figure 3.7 confirm these results, but also show that, for smaller values of recall, the precision does not vary as much, which is an interesting fact especially for applications where one is mostly interested in the first few retrieved images.

Table 3.2: Effects of the number of cells reduction in *CCH* effectiveness.

Number of cells	Number of bins*	θ_{abs}	θ_{rel}
8×8	1856	0.59	89%
6×6	1044	0.58	84%
4×4	464	0.55	69%
3×3	261	0.55	68%
2×2	116	0.45	35%

*Average values

3.4.3 Experiment III – Partial representation of image's content

This experiment exploits a partial representation for the color distribution of an image. We are interested in investigating how the retrieval effectiveness is affected if the content of the images is partially represented.

As showed in Figure 3.1 and discussed in Section 3.1, on average only about 29 colors

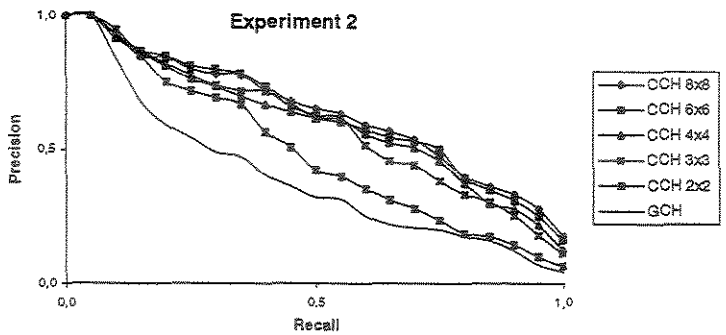

Figure 3.7: Precision vs. Recall curves for Experiment II.

(out of 64 possible) are present in each image of our reference collection. The same figure showed that only a small number of colors are responsible for the majority of the image content. For example, in average, 95% of the image content is composed by only 12 colors, 90% is composed by 9 colors, 80% is composed by 6 colors and 70% of the image content is composed by only 4 colors.

Based on the numbers above, we have two alternatives to partially represent the color distribution of an image. The first one consists in fixing a number, say k, of dominant colors and to represent the percentage of the images covered by these k colors. The second alternative consists in fixing the percentage of the image we want to represent, and to use as many (dominant) colors as necessary to cover that percentage. While in the former alternative, the percentage of the image represented depends on its visual content, in the latter, it is the number of used colors that depends on the visual content of the image.

The main disadvantage of using a fixed number k of dominant colors to represent images is that the portion of the images covered by these colors can vary largely. In some cases, k dominant colors could be sufficient to represent 100% of the image content (the image has exactly k distinct colors). For colorful images where all colors cover approximately the same percentage of the image, k dominant colors (assuming that k is a small value) covers only a small fraction of the image. It should be clear that the representation obtained using this approach is not robust, as simple images can be represented completely, while more complex images are poorly represented. Therefore, in our experiments we represent images partially by fixing the percentage of the images we want to represent (the second alternative discussed above). Thus, we have the guarantee that the same portion of the images is represented, independently of the relative complexity of their visual content.

It is also important to notice that, when using *CCH*, the dominant colors that cover a predefined percentage of the images are chosen globally, while in approaches like *Grid*, this decision is local (the set of dominant colors is determined for each individual cell). We

argue that, in the context of query-by-image-example, the global decision of *CCH* is more robust than the local decision of *Grid*. It is possible that colors dominant in the whole image are not dominant in some cells and also that non-dominant colors in the whole image are dominant in some cells. As a result, the local decision of approaches like *Grid* results in a non-optimal representation for the purpose of query-by-image-example, where the user is interested in the properties of the images as a whole, not in a particular portion of the images. If the idea is to have a more robust representation for the purpose of query-by-region-example, regional approaches (Section 2.6.3) are a much better alternative than partition-based approaches.

The experiment summarized in Table 3.3 exploits the partial representation of image's content in conjunction with the *CCH* approach. We fixed the color-space quantization scheme (64 colors) and the spatial partition scheme (8×8=64 cells). The smaller the number of represented colors, the smaller the number of cell histograms needed to describe an image. As expected, both the storage requirements and the retrieval effectiveness decrease with the partial representation of the image content. However, the range between 90% and 100% of image content results in small effectiveness decreasing when compared to the reduction of the space overhead. For instance, in order to represent 90% of the image content, we need on average 9 colors, resulting in 9 cell histograms. This number is 69% smaller than the number of cell histograms required to represent 100% of the image content (29 colors in average). The effectiveness results (θ_{rel}) decreases 7%, but the gain in storage space seems to offset well the loss in retrieval effectiveness. The corresponding Precision vs. Recall curves can be viewed in Figure 3.8. As in previous experiments, these curves follow the effectiveness results obtained with the θ_{rel} measure.

Table 3.3: Effectiveness results for partial representation of the image content using *CCH*.

% of the image	Number of Colors*	Number of bins*	θ_{abs}	θ_{rel}
100%	29	1856	0.59	89%
95%	12	768	0.58	84%
90%	9	576	0.57	81%
80%	6	384	0.53	67%
70%	4	256	0.43	34%

*Average values

3.4.4 Experiment IV – Fine tuning of *CCH*

In the last round of experiments, we compare the *GCH*, *Grid* and *CCV* against some variations of the *CCH* approach, using different number of cells and representing different

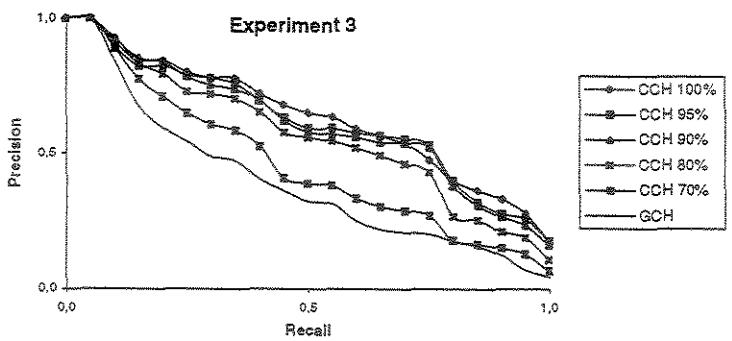

Figure 3.8: Precision vs. Recall curves for Experiment III.

percentages of the image content. Our goal is to demonstrate how one can finely tune the *CCH* approach, by analyzing the various opportunities for trade-off between space overhead and effectiveness it offers. This also serves to summarize the previous experiments, comparing simultaneously the space overhead and the retrieval effectiveness of all investigated approaches.

The results of the last experiment are shown in Table 3.4. By representing 100% of the image content and partitioning the images in 8×8 cells, *CCH* offers retrieval effectiveness similar to *Grid*, but with a considerable reduction of 55% in space overhead. Alternatively, by representing 80% of the image's content and adopting a partition of 3×3 cells, *CCH* requires the smallest space overhead, 15% smaller than the *GCH* (our reference approach), yet yielding an effectiveness (θ_{rel}) 43% higher.

It is also possible to obtain intermediate results between the configurations discussed above. For instance, by representing 90% of image content and by adopting a partition of 4×4 cells, *CCH* results in a respectable reduction of 96.5% in space overhead when compared to the *Grid* approach. As well, using twice the space overhead of *GCH*, it is possible to be 61% more effective. Other intermediate results may be obtained by choosing an adequate compromise between the number of cells and the percentage of the image content (number of colors) being represented. It is possible to emphasize retrieval effectiveness or space overhead reduction. These results confirm the potential and the flexibility of the *CCH* approach in comparison to traditional global and partition-based approaches. The $P \times R$ curves of this experiment are shown in Figure 3.9.

3.5 Chapter conclusion

Our main contribution in this chapter is a simple, compact, flexible and yet very effective variation of partition-based techniques called Cell/Color Histograms – *CCH*. To the best

Table 3.4: Comparing the effectiveness of some distinct *CCH* configurations.

Approach	Parameters			Results	
Used	Grid Size	% of the image	Number of bins	θ_{abs}	θ_{rel}
GCH	1×1	100% (64 colors)	64	0.36	0%
CCV	1×1	100% (64 colors)	128	0.40	10%
Grid	8×8	100% (64 colors)	4096	0.59	95%
CCH	8×8	100% (29* colors)	1856*	0.59	89%
CCH	6×6	95% (12* colors)	432*	0.56	78%
CCH	4×4	90% (9* colors)	144*	0.53	61%
CCH	3×3	80% (6* colors)	54*	0.46	43%

*Average values

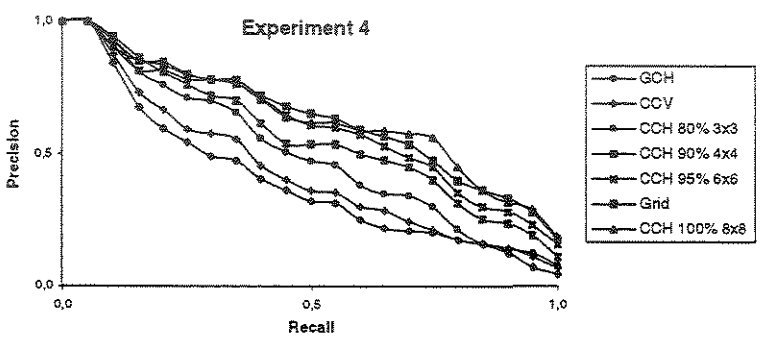

Figure 3.9: Precision vs. Recall curves for Experiment IV.

of our knowledge, it is original in the way that visual features are encoded. Our motivation was to reduce the space overhead of partition-based approaches taking advantage of the fact that only a relatively low number of colors is present in most images. In particular, we used color features to verify the idea.

The experimental results we obtained confirm the compactness and the flexibility of *CCH* in comparison to traditional global and partition-based approaches. It is possible to emphasize retrieval effectiveness or space overhead reduction. For instance, if warranted, one could use the *CCH* and obtain the smallest space overhead, 15% smaller than a *GCH* while still being 43% more effective.

The experiments discussed in this chapter show that, under "usual" (e.g., using default parameters) circumstances, *CCH* performs consistently well. Although we exploit our approach using simple and well-known parameters, it is important to observe that our alternative representation can be applied with success in more sophisticated configurations such as the ones that exploit hierarchical spatial decomposition of images, more complex

distance functions, dynamic color-space quantization, histogram compression techniques, etc.

Capítulo 4

CBC – Color-Based Clustering

In this chapter we present a new Content-Based Image Retrieval (CBIR) approach based on cluster analysis[1]. CBIR relies on compact representations for the visual content of images (visual features). In order to produce such visual features, we propose an efficient and adaptive clustering algorithm to segment the images into regions of high color-similarity. This approach contrasts with those that describe images using a single color histogram for the whole image (global approaches), or local color histograms for a fixed number of image cells (partition-based approaches). Our experimental results show that our clustering approach offers high retrieval effectiveness with low space overhead. For example, using a database of 20,000 images we obtained higher retrieval effectiveness than partition-based methods with about the same space overhead of global methods, which are typically regarded to as storage-wise compact.

This chapter is organized as follows. Section 4.1 describes *CBC*, our clustering-based approach to CBIR. The distance function used to compare the *CBC* representation of two images is discussed in Section 4.2. Section 4.3 presents our experimental setup, and Section 4.4 reports our experimental results comparing the retrieval effectiveness among three variations of our approach and five other existing CBIR approaches. Finally, Section 4.5 presents the chapter conclusions.

4.1 Color-based clustering – Our approach to CBIR

The *CBC* (Color-Based Clustering) approach is based on a fully automatic clustering algorithm that has an efficient implementation. In *CBC*, each image is decomposed into a set of disjoint, connected regions. Each region is larger (in number of pixels) than a threshold size s_0. Additionally, all pixels of a region have a predefined degree of color

[1] This chapter was published in the *"Proceedings of the IDEAS'2001 International Symposium"* [100].

similarity, according to a threshold color-distance d_0. We denote the procedure to obtain these regions $CBC(d_0, s_0)$, where the thresholds d_0 and s_0 are parameters defined by the user. These parameters have a direct impact on the number of regions in which each image is decomposed. We adopt the convention that parameters d_0 and s_0 are percentages of the maximum value allowed. Thus, $s_0 = 10$ means a threshold size equivalent to 10% of the image size (the maximum possible size of a region) and $d_0 = 5$ means a threshold distance equivalent to 5% of the maximum distance between two points in the chosen color-space. The $CBC(d_0, s_0)$ procedure can be conceptually divided into four main steps:

1. Convert an input image I into a weighted and non-oriented graph $G(V, E)$, where the pixels in I are the vertices of V and each pair (p, q) of 4-adjacent pixels in I define an edge of E whose weight is the Euclidean distance between the colors of p and q in the CIE Lab color-space[2].

2. Compute an adaptive and agglomerative clustering of pixels on G using d_0 as parameter. This algorithm outputs a graph partition where each part is a tree whose nodes form a connected region of pixels, and the least similarity between distinct regions is greater or equal to d_0.

3. Systematically merge all regions whose area is less than s_0 with their most similar neighbor, until all remaining regions have area greater or equal to s_0.

4. Characterize each remaining region by a 6D feature vector (L, a, b, s, h, v), where L, a, b are the mean values of the Lab color components in the region, s is the size (number of pixels) of the region normalized by the image size, and h, v are the normalized horizontal and vertical coordinates of the geometric center of the region in the image.

Notice that the feature vectors of all regions that compose the image I represent its visual content, and so, they are stored in the database. To complete the description of the CBC procedure, we explain in the following subsection the clustering algorithm used in step 2.

4.1.1 Clustering algorithm

Cluster analysis [31, 45] is one of the most well-developed and commonly used forms of combinatorial data analysis, and hierarchical clustering are among the best-known

[2]The Lab space has been defined in order to make easier the evaluation of perceptual distance between colors. In fact, they are defined according to transformations that approximate a tri-stimulus color-space into an Euclidean space [13].

clustering methods. Our clustering algorithm consists of a simple and effective variation of the agglomerative *single-linkage* clustering algorithm discussed in Section 2.2.2.

The single-linkage algorithm can be described under the language and concepts of graph theory [31]. Consider the data elements that we want to cluster (image pixels) as being nodes of a graph. The merging of two (initially singleton) clusters $A = \{A_i\}$ and $B = \{B_j\}$ corresponds to adding an edge between the nearest pair of nodes $A_i \in A$ and $B_j \in B$. Since edges are added always between distinct nodes (clusters), the resulting graph never has any closed circuit. In fact, the resulting graph is a *tree*. If this process continues until all clusters are linked together, it can be shown that the resulting graph is a *minimal spanning tree* – MST [31]. Based on the relation between the single-linkage algorithm and minimal spanning trees, we chose to implement our single-linkage algorithm with a implementation equivalent to that of the well known Kruskal's greedy algorithm to generate a MST [25, Section 24.2]. This is the asymptotically fastest implementation known to us. The procedure that describes our variation of the single-linkage algorithm is shown in Figure 4.1.

```
   Single-Linkage(G(V,E), d0)
1     for each vertex v ∈ V do Make-Set(v)
2     sort the edges of E by nondecreasing weight w
3     for each edge (u,v) ∈ E, ordered by nondecreasing weight
4        if d_mean(Find-Set(u), Find-Set(v)) < d0
5           if Find-Set(u) ≠ Find-Set(v), then Union(u,v)
6        else break
```

Figure 4.1: Our implementation of the Single-linkage clustering algorithm

The function `Make-Set(.)` creates a cluster with only one element: the node passed as parameter. The function `Find-Set(.)` returns the identifier of the cluster that contains the element passed as parameter. Our clustering algorithm works as follows. Line 1 creates $|V|$ clusters, each with one node of G. The edges in E are sorted by non-decreasing weight in line 2. The `for` loop in lines 3-6 checks if the distance between the clusters of u and v is smaller than the threshold distance d_0 (the stop criterion) for each edge (u, v). If so, and if the clusters are distinct, they are merged. In order to diminish the effects of the single-linkage chaining effect[3], we have introduced a new heuristic into the stop

[3]The single-linkage algorithm is not able to keep two clusters clearly apart when they come very close to each other, because a single link between the two clusters is sufficient to connect them. This characteristic leads to a notorious *chaining effect* [31, 45], by which poorly separated clusters are chained together.

criterion. Instead of comparing the weight of the edge being analyzed directly with the threshold distance d_0, we compare the distance of the mean vector of each cluster with d_0. To do this, we assign a mean vector to each cluster and update this vector whenever a `Union(.,.)` operation is performed.

As we can see, we use the d_{min} distance (discussed in Section 2.2.2) to decide the ordering and which clusters are candidates to be merged, and d_{mean} distance[4] to decide whether or not the candidates should be merged. This solution has the advantage to be as efficient as the traditional single-linkage procedure and, at the same time, to reduce the chaining effect, the main problem of the single-linkage approach.

Our algorithm differs from Kruskal's algorithm in two ways. First, we do not store the edges that compose the MST. Second, we add a stop criterion (line 4) to finish the process before we obtain only one cluster (the MST). Clearly, the running time of our algorithm is the same as in Kruskal's algorithm: $O(E \log E)$ [25]. However, in our graph model $E = O(V)$ (because of the connectivity restriction). Thus, our clustering procedure runs in time $O(n \log n)$, where $n = |V|$ is the number of pixels in the image. Examples of images automatically segmented with the *CBC* clustering algorithm are shown in Figure 7.7.

4.2 Distance function

In this section, we describe the distance function used to compare two images in *CBC* approach. The distance between two images A and B, $d(A, B, \alpha)$, is a weighted composition of the distances between the regions that compose each image – $Rd(A_i, B_j, \alpha)$. Here, A_i and B_j are regions from images A and B obtained via the clustering algorithm detailed before. The α parameter defines the weights used to combine the distance between the color of the regions with the distance between the spatial position of the regions. The distance function $Rd(A_i, B_j, \alpha)$ between regions A_i and B_j is defined as:

$$\begin{aligned} Rd(A_i, B_j, \alpha) = \quad & \alpha \times L_2(A_i.color, B_j.color) + \\ & (1-\alpha) \times L_2(A_i.center, B_j.center) \end{aligned}$$

where $L_2(.,.)$ represents the Euclidean distance between its arguments. The computation of the distance between two images $d(A, B, \alpha)$ is algorithmically described in Figure 4.2, and works as follows.

Initially, we compute all possible distances between pairs of regions (one of each input image), according to the function $Rd(A_i, B_j, \alpha)$. Additionally, all regions are initialized as "non-matched" (status=0). The initialization steps (lines 1-4) take time $O(nm)$, where

[4] $d_{mean}(A, B) = d(A_{mean}, B_{mean})$, where A and B are clusters, and A_{mean} and B_{mean} are A's and B's median vectors.

```
    d(A, B, α)
1     for each pair of regions A_i ∈ A and B_j ∈ B
2        A_i.status = 0
3        B_j.status = 0
4        D_{A_iB_j} = Rd(A_i, B_j, α)
5     sort the computed distances D_{A_iB_j} in non-decreasing order
6     β = 0
7     for each distance D_{A_iB_j} in non-decreasing order
8        if A_i.status = B_j.status = 0
9           if A_i.size < B_j.size
10             w = A_i.size
11             B_j.size = B_j.size − A_i.size
12             A_i.status = 1
13          else
14             w = B_j.size
15             A_i.size = A_i.size − B_j.size
16             B_j.status = 1
17             if A_i.size = 0 then A_i.status = 1
18          β = β + w × D_{A_iB_j}
19    return β
```

Figure 4.2: Distance function algorithm

n and m are the number of regions in the images A and B, respectively. In line 5, the computed distances are ordered in non-decreasing order. In this way, the first distance value corresponds to the best possible match between a region of image A and a region of image B. The second value corresponds to the second best match and so on. The ordering of nm values takes time $O(nm\log(nm))$. For each distance $D_{A_iB_j}$ in non-decreasing order, we compare the size of the related regions A_i and B_j. The smallest region determines the weight value w that will multiply the value $D_{A_iB_j}$ in order to obtain the distance β between the images. The weight w represents the percentage of the two images that match with distance $D_{A_iB_j}$.

In each iteration of the `for` loop in line 7, the smallest region related to $D_{A_iB_j}$ is marked as "matched" (status=1). This means that the smaller region has matched completely, and any other distance related to this region in the next iterations should be discarded. Since at each iteration the largest region may not be completely matched, we subtract the size of the smallest region (the percentage of the region actually matched) from the size of the largest one. After this operation, if the size of the region equals 0 (occurs when the two compared regions have the same size), then both regions match completely and

we mark them accordingly. If the size of the largest region remains greater than 0, then we continue analyzing distances involving this region until the size of the region equals 0. Observe that the next distances involving a previous analyzed region will always be higher than previous distances because we are analyzing these values in non-decreasing order. Observe also that, at the end of the process, the sum of the weight values w equals 1, meaning that the entire image content has been compared. The `for` loop in line 7 is executed nm times, one for each distance $D_{A_iB_j}$. Thus, the execution of lines [7..18] takes time $O(nm)$.

It is important to notice that an actual region A_i of an image may be virtually "broken" into smaller units to find the best possible match for it. The best possible match may involve only one greater region, a set of several smaller regions, or also a combination of partial content of several regions of the other image. In any way, at the end of the process, the "real" regions of the two images were decomposed in possibly smaller "virtual" regions in such a way that: (1) the number of "virtual" regions of the two images is the same, (2) there is a one-to-one correspondence between the virtual regions of the two images and (3) each pair of corresponding regions has the same size. Thus, the distance between images becomes a weighted average of the distance between their corresponding virtual regions where the weights are their sizes.

We will exemplify the application of our distance function using the images in Figure 4.3. For the sake of simplicity, assume the use of gray-scale images and the distance between two regions given only by the difference of their gray levels. In Figure 4.3, we have two images A and B that we want to compare. Image A has three regions (obtained via a suitable algorithm, e.g., our proposed *CBC*): $A = \{A_1 \cup A_2 \cup A_3\}$. Likewise, image B has only two regions: $B = \{B_1 \cup B_2\}$. The best possible match between regions of A and B is (A_3, B_1) because this pair of regions has exactly the same gray level. Since B_1 is larger than A_3, we split B_1 into two virtual regions $B_1 = \{B_{11} \cup B_{12}\}$, in a way that one of the resulting regions (B_{12}) has exactly the same normalized size of region A_3. Thus, we obtain (A_3, B_{12}) and this match will contribute to the distance between A and B with weight 0.25 (the normalized size of the matched regions).

The second best match between real regions of images A and B is (A_1, B_1) but, as the region B_1 was split before, and one of the resulting regions (B_{12}) was matched in a previous step, the only possible choice is (A_1, B_{11}). Again, as region A_1 is larger than region B_{11}, the algorithm decomposes A_1 in two virtual regions $A_1 = \{A_{11} \cup A_{12}\}$ in a way that the virtual region A_{12} has the same size of B_{11}. Thus, our second match is (A_{12}, B_{11}) also with weight 0.25. Now, let us assume that the third best match is (A_2, B_2). As B_2 is larger than A_2, the algorithm decomposes B_2 in two virtual regions $B_2 = \{B_{21} \cup B_{22}\}$. The third match is then (A_2, B_{22}) and the final match is (A_{11}, B_{21}). Since all the virtual regions have the same size, the four virtual matches (evaluated according to their gray

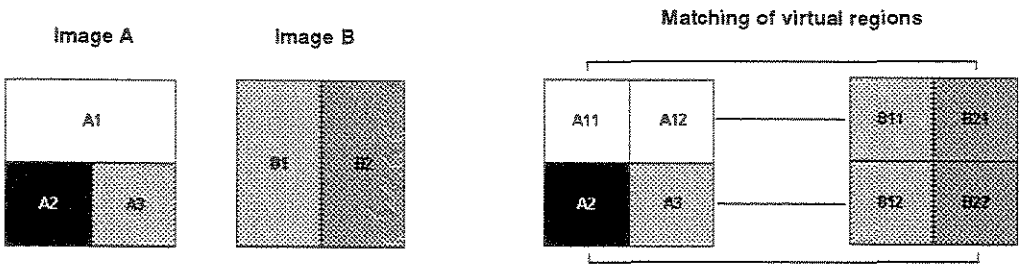

Figure 4.3: An example of how the distance function decomposes real regions into virtual regions

level distance) are averaged with the same weight $w = 0.25$ in order to obtain the distance between A and B.

4.3 Experimental setup

We compared the effectiveness of *CBC* with the effectiveness of five other approaches discussed in Section 2.6: (1) Global Color Histogram – *GCH*, (2) Color-Coherence Vector – *CCV*, (3) Color-Moments – *CMM*, (4) *Grid* and (5) Cell/Color Histograms – *CCH*. The first three approaches are global methods and the last two are partition-based methods.

We compared the *CBC* approach with three distinct combinations of threshold color-distance (d_0) and threshold size (s_0). Each combination resulted in a different number of regions per image. Since our distance function depends of a parameter α, we experimentally determined that $\alpha = 0.875$ corresponds to the most adequate compromise. Confirming our intuition, this value suggests that the distance between the colors of the regions is more relevant than the position of such regions when we compare two images. When we used $\alpha = 1$ (eliminating completely the contribution of the spatial location to the distance value), the effectiveness of our approach did become slightly smaller. A summary of the most important characteristics of the eight compared approaches is shown in Table 4.1.

We evaluated the effectiveness of the approaches in processing image-based queries using a controlled environment. Since we did not count with a population of users, the evaluation study was based on objective criteria. For example, the set of relevant images relative to a given query (RRSet) was determined *a priori*, using an objective relevance criterion in which the relevance of a image relates to how well the image responds to the query that was posed. The subjective view of relevance considers not only the content of an image but also the state of knowledge of the user at the time of the search.

We used a dataset of 20,000 JPEG images from a stock CD by Corel Corp.. This

Table 4.1: Summary of the compared approaches in *CBC*'s effectiveness evaluation.

Approach	Color-Space	Cells/regions ♯	Space♭	Complexity♯	Metric
GCH	RGB	1	64	$O(n)$	L_1
CMM	HSV	1	9	$O(n)$	L_1
CCV	RGB	1	128	$O(n)$	L_1
Grid	RGB	64	4096	$O(n)$	L_1
CCH	RGB	64	1856	$O(n)$	L_1
CBC(5, 0.3)	Lab	11	66	$O(n \log n)$	$d(A, B, \alpha)^\dagger$
CBC(3, 0.1)	Lab	40	240	$O(n \log n)$	$d(A, B, \alpha)^\dagger$
CBC(2, 0.05)	Lab	155	930	$O(n \log n)$	$d(A, B, \alpha)^\dagger$

♯ Average values for the 20,000 images we used
♭ Average number of real numbers needed to represent the images in our database
♯ n is the size (number of pixels) of an image
† $\alpha = 0.875$

database is composed of images of several different domains and, for this reason is called a *heterogeneous database*. In each domain, the images are semantically related, allowing to distinguish one domain from the others easily. Since the images belonging to the same domain may not have a very similar visual content, we called these domains *heterogeneous domains*. We chose 29 of such domains to be used in our experiments. Out of each domain, we selected an image to be used as query image and a set of images visually similar to the chosen query image called *Relevant Result Set* – RRSet. Examples of RRSets are shown in Figures 7.2, 7.3 and 7.4.

In any CBIR approach, we expect to retrieve the RRSets as soon as possible, since they correspond to what we consider relevant for each query image. In the average, we have 30 relevant images per query image. We have performed two distinct experiments with databases of different sizes. In one experiment, we used the 20,000 images of the whole dataset as a large and heterogeneous database. In the other experiment, we used the union of the RRSets of the first experiment as a small and less heterogeneous database with 1,023 images. We used the same queries and RRSets in both experiments. Our goal was to analyze if the relative performance between the analyzed approaches remains the same when we change the size of database, enlarging/diminishing the proportion of relevant/non-relevant images.

We compared the approaches' effectiveness using two distinct measures: (1) Precision vs. Recall [82] and (2) *Normalized Average Rank – NavgR'* [99] (a derivative of the measure presented in [32]).

The Normalized Average Rank (NavgR) measure was first used in the QBIC project [32], and after that in some other CBIR approaches [86, 99]. NavgR measures how close

the set of relevant items appears to the top of the ranked result. This value is normalized considering an ideal retrieval in which the relevant images appear ahead of any non-relevant one in the ranked result. The relative ordering of the RRSet elements is irrelevant because we are using a binary judgment of relevance. The measure used in QBIC project was defined as $NavgR = A/I$, where A and I are, respectively, the actual average rank and the ideal average rank of the RRSet. In this paper, we propose a slightly different definition of the NavgR measure, inverting the original ratio as shown in Equation 4.1.

$$NavgR' = \frac{I}{A} = \frac{\sum_{i=1}^{|RRSet|}(i-1)}{\sum_{i=1}^{|RRSet|} rank(i)} \tag{4.1}$$

In that equation, $rank(i)$ is a function that returns the rank of the ith relevant image. The rank values vary in the range $[0, |DB| - 1]$, where $|DB|$ denotes the cardinality of the image database. With this alternative definition, the $NavgR'$ value ranges from 0 to 1 and equals 1 in the best case, as occurs in precision and recall measures. Higher values of $NavgR'$ are associated to good effectiveness results, instead of vice-versa. This fact reduces the dependence of the measure on the worst results during the process of averaging results from distinct queries.

A drawback with the $NavgR'$ measure, is that, the value obtained for a single query is still very sensitive to the rank of the last relevant document retrieved. In this paper, we deal with this problem considering only a predefined portion of the RRSets for the purpose of $NavgR'$ computation. We consider only the ranks of the first 80% portion of an RRSet. The last 20% of the RRSets (with the highest numerical ranks) are discarded, since any possible misleading judgment of relevance (that will strongly affect the $NavgR'$ value) could materialize itself as a large numerical rank.

4.4 Experimental results

Table 4.2 shows a comparative analysis of the approaches, based on the mean values of $NavgR'$ obtained using the results of the 29 visual queries. This table shows relative values, obtained using the *GCH* approach as reference. We also included the space overhead (in terms of floating-point numbers) and the computational complexity of each approach. Figure 4.4 on the other hand, shows a summary of the results obtained in terms of $P \times R$ curves. Due to the limited space only the curves comparing *GCH* (typical benchmark), *CCV* (runner-up among the global approaches), *CCH* (the best performer among the partitioning approaches) and *CBC*(3,0.1) (a "good compromise" version of the proposed approach) are shown. Nevertheless, in general terms, the obtained values of $NavgR'$ are consistent with the $P \times R$ curves – which further supports the use of $NavgR'$ as a

single measure for retrieval effectiveness. Next we discuss the results in terms of global, partition-based and clustering approaches.

Table 4.2: Single-value *CBC*'s effectiveness using *GCH* results as reference

Approach	Space*	Complexity	Experiment 1	Experiment 2
GCH	64	$O(n)$	—	—
CMM	9	$O(n)$	-48.44%	-61.00%
CCV	128	$O(n)$	-4.07%	-10.60%
Grid	4096	$O(n)$	9.00%	19.16%
CCH	1856	$O(n)$	16.97%	38.01%
CBC(5, 0.3)	66	$O(n \log n)$	10.56%	53.48%
CBC(3, 0.1)	240	$O(n \log n)$	21.67%	98.09%
CBC(2, 0.05)	930	$O(n \log n)$	33.26%	112.93%

* Average number of floating-point numbers needed to represent each image

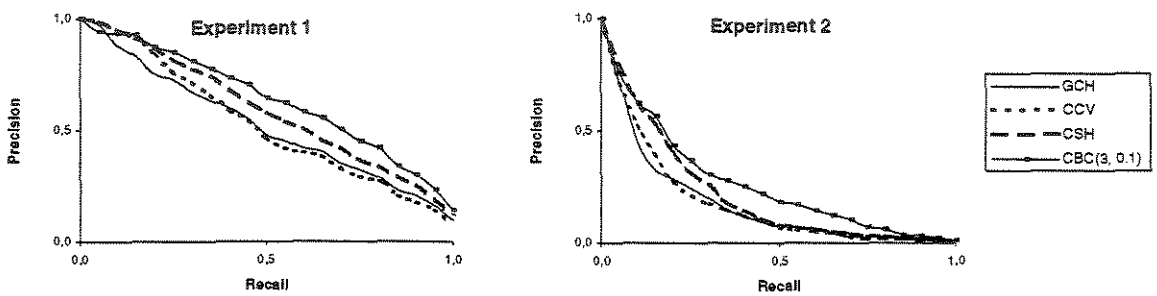

Figure 4.4: Precision vs. Recall curves for some of the investigated approaches

Among the global approaches (*GCH*, *CMM* and *CCV*), the traditional *GCH* has the best performance in both experiments. Clearly, *CMM* is the worst approach in terms of effectiveness, about 50% worse than *GCH*'s. However, one must notice that *CMM* yields the smallest space overhead, seven times smaller than *GCH*'s overhead. *CMM*'s and *CCV*'s relative performance were also sensitive to the growth, in size and heterogeneity, of the image database.

Among the partition-based approaches (*Grid* and *CCH*), *CCH* is the best approach in both experiments, with a space overhead 50% smaller than *Grid*'s. The effectiveness of *CCH* is about 38% better than *GCH*'s (at the second experiment of Table 4.2), at the cost of a space overhead 30 times larger.

Finally, the results in Table 4.2 show that the three distinct configurations of our clustering approach are more effective than *CCH*– the best of the compared approaches.

The results also show that our approach is more robust to the growth of the database. Considering the *CBC*(5,0.3) configuration, we have simultaneously the space overhead of the *GCH* approach and an effectiveness higher than *CCH*'s effectiveness (at the second experiment), with more robustness since the relative effectiveness of the *CCH* approach doubled when the database growth, while the effectiveness of *CBC*(5,0.3) becomes 5 times higher. Using a configuration that results in more regions (*CBC*(2,0.05)), we can achieve gains of the order of 110% relative to the *GCH* approach, with more robustness and with a space overhead 50% smaller than *CCH*'s. The robustness is an important factor since it implies that, the larger the database, the larger the differences in effectiveness between our approach and the other compared approaches.

The *CBC*(3,0.1) configuration represents an interesting compromise among the size of the number of regions, effectiveness and robustness. With this configuration, we obtain in average 40 regions per image, resulting in a space overhead 4 times larger than *GCH*'s, but 90% smaller than *CCH*'s. The effectiveness of *CBC*(3,0.1) is 98% higher than *GCH*'s while *CCH*'s gain is only of 38%. When the database grows from 1.023 to 20.000 images, the advantage of *CCH* in relation to *GCH* becomes approximately two times larger, while with *CBC*(3,0.1) this advantage becomes five times larger, suggesting that *CBC*(3,0.1) is also considerably more robust than the other compared approaches. Figure 7.5 shows an example of the top 30 images returned in response to a visual query using the *CBC*(3,0.1) configuration. The query image in this example is the first image retrieved.

4.5 Chapter conclusion

In this chapter, we presented *CBC*, a new *content-based image retrieval* approach based on cluster analysis. Overall, our contribution within the ever-growing area of image databases is an efficient process to obtain an image's representation (visual features), and an effective way to assess similarity between images. We have used a simple variation of the single-linkage clustering algorithm to find out disjoint regions of the images composed by pixels with a predefined degree of color similarity. This approach has the advantage of avoiding the notorious problem of color-space quantization, and of being adaptive in the sense that the segmentation process depends mostly on the image itself rather than on "artificial" parameters, such as the number of clusters or the like. Using *CBC*, images are represented and compared based on the set of regions in which they were decomposed. We have also proposed a new distance function to compare the *CBC* representation of two images.

The effectiveness of three configurations of *CBC* were compared against the effectiveness of five other CBIR approaches, in a controlled environment. Given the results discussed along the paper, we believe that the main advantages of our approach are: (1)

flexibility – using different thresholds for region's size and color's distance, it is possible to obtain different compromises between space overhead and effectiveness; (2) *configurability* – both the clustering algorithm and the distance function can be tuned to work in specific domains, i.e., to consider additionally domain-dependent visual features; (3) *effectiveness* – the three configurations of *CBC* yielded better retrieval effectiveness than the other five compared approaches; and (4) *robustness* – our experiments have shown that *CBC* is more robust with respect to the database growth. In fact, the larger the database size, the larger the relative advantage of *CBC*.

Capítulo 5

MiCRoM – Minimum-Cost Region Matching

Recently, several content-based image retrieval (CBIR) systems that make use of segmented images have been proposed. In these systems, images are segmented and represented as a set of regions, and the distance between images is computed according to the visual features of their regions. A major problem of existing distance functions used to compare segmented images is that they are not metrics. Hence, it is not possible to exploit filtering techniques and/or access methods to speedup query processing, as both techniques make extensive use of the triangular inequality property – one of the metric axioms. In this work, we propose *MiCRoM* (Minimum-Cost Region Matching), an effective metric distance that models the comparison of segmented images as a *minimum-cost network flow* problem[1]. To our knowledge, this is the first time a true metric distance function is proposed to evaluate the distance between segmented images. Our experiments show that *MiCRoM* is at least as effective as existing non-metric distances. Moreover, we have been able to use the recently proposed Omni-sequential filtering technique, and have achieved nearly 2/3 savings in retrieval/query processing time.

The main problem to model a regional CBIR approach in a metric space is related to the distance function used to compare segmented images. To the best of our knowledge, there are only a few works dedicated to this topic. In general, the most common approach is to perform comparisons based on individual regions, as in Blobworld system [18]. In this system, although querying based on a limited number of regions is allowed, the query is performed by merging single-region query results. Even if it was possible to combine the results obtained with each individual region of an image, there is no guarantee that the full content of the images is compared. It is possible that most of the regions in an image matches with the same region of the other. Moreover, if the comparison is performed in

[1]This chapter was published in the *"Proceedings of the VISUAL'2002 International Conference"* [102].

the opposite direction, it is possible to obtain a completely different distance.

In order to reduce the influence of inaccurate segmentation, and to guarantee the comparison of the full content of the images, systems like SIMPLIcity [58] and *CBC* [100] compare images according to the properties of all segmented regions simultaneously, not only in a region-by-region basis. SIMPLIcity compares images according to the *IRM* (Integrated Region Matching) distance. An equivalent distance function is used in *CBC*. The main difference is that the visual features used to compare individual regions in *CBC* and SIMPLIcity are not the same.

The *IRM* distance between two images X and Y is algorithmically described in Figure 4.2. The main problem of the *IRM* distance function is that it does not satisfy the triangular inequality property. This problem is related to the greedy approach of choosing first the most similar regions to be matched. The greedy algorithm in this case does not guarantee that the obtained distance is the best (smallest) one.

Figure 5.1 shows a counterexample where the results obtained with the *IRM* greedy distance do not satisfy the triangular inequality property. In this example, images X, Y and Z are compared two-by-two, according to their regions. Each image has exactly two regions of the same size (0.5). For illustrative purpose only, each region has its visual feature represented by a single numerical value. This number could be, for example, the average gray level of the region. The size and also the visual feature of the regions are normalized between 0 and 1. The distance between two regions (d_{reg}) is given by the module of the difference of their visual features. The edges between images show the matched regions according to the *IRM* distance. On the right of Figure 5.1, there is also the result of the comparisons, organized in a triangular shape.

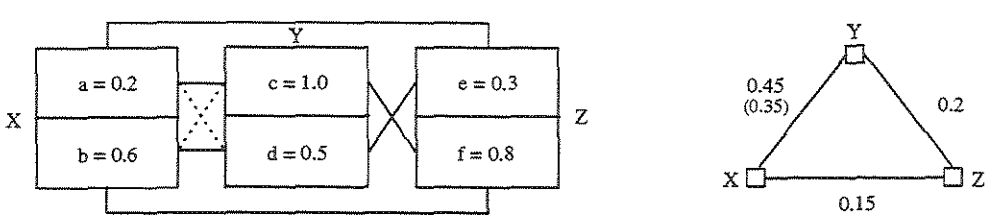

Figure 5.1: An example to show that the *IRM* distance does not satisfy the triangular inequality property

As we can observe, the triangular comparison of the images give us the inequality $0.45 \geq 0.2 + 0.15$, which contradicts the triangular inequality property. The problem in this example is in the distance between images X and Y. The greedy approach adopted in *IRM* results in a non-optimal distance when X and Y are compared, because there is another match which reduces the distance between them.

The optimal comparison that minimizes the distance between images X and Y is shown in dotted lines, and gives the result $optimal(X, Y) = 0.5 \times |0.2 - 0.5| + 0.5 \times |0.6 - 1.0| = 0.35$. The result of this optimal comparison is shown between brackets in the triangular representation of the distances among the three images. If the optimal distance is used, we have $0.35 \leq 0.2 + 0.15$, which satisfies the triangular inequality property.

The remainder of this chapter is organized as follows. In Section 5.1, we propose *MiCRoM*, our new metric distance to compare segmented images. The effectiveness of *MiCRoM* is evaluated in Section 5.2. Experimental results related to the use of filtering techniques based on the *MiCRoM* metric properties are presented in Section 5.3. Finally, Section 5.4 states the chapter conclusions.

5.1 The *MiCRoM* metric distance

This section proposes *MiCRoM* (Minimum-Cost Region Matching), a new *metric* distance function to compare the visual content of segmented images. As it will be shown in Section 5.2, *MiCRoM* is at least as effective as *IRM*, the distance function used in SIMPLIcity and *CBC* systems, and has the advantage that it can be adequately indexed using existing MAMs [20] such as the M-tree [23]. It is also possible to use a combination of filtering techniques and SAMs [35] to speedup the query processing, as it will be shown in Section 5.3.

The main idea of *MiCRoM* consists of modeling the comparison of segmented images as a *minimum-cost network flow problem* [2]. More specifically, the comparison of images is modeled as a *transportation problem*. The transportation problem is an optimization problem that can be informally expressed as follows. Assume that we have a number of consumers with certain demand for a product. This product is made by a number of producers with certain production capacities. The system is balanced in the sense that the total demand equals the total production capacity. The production should be transported from the producers to the consumers, such that every consumer gets exactly as much product as it needs, and the transportation costs from all producers to all consumers are known in advance. The transportation problem is to find the optimal (cheapest) way to bring the products from the producers to the consumers. Next, a formal definition for the transportation problem is given.

A *network* is a directed graph $G = (V, E)$ composed by a set V of n nodes and a set E of m arcs. Each node represents either a producer or a consumer. Assuming that there are p producers and c consumers, we have: $n = p + c$. Each node has an associated number pd that represents its production (positive values) or its demand (negative values) depending on whether the node is a producer or a consumer. The system is balanced, so $\sum_{i=1}^{p} pd_i + \sum_{j=1}^{c} pd_j = 0$. There is a directed arc (i, j) for every pair of producer

i and consumer j. Thus, $m = p \times c$. Each arc (i, j) has two associated values: its transportation capacity cap_{ij}, and its transportation cost $cost_{ij}$. The arc capacity is given by $cap_{ij} = min(|pd_i|, |pd_j|)$. The decision variable in the transportation problem is the flow $flow_{ij}$ in each arc (i, j). These flows should satisfy $0 \leq flow_{ij} \leq cap_{ij}$, and should minimize the function $\sum_{i=1}^{p} \sum_{j=1}^{c} (cost_{ij} \times flow_{ij})$.

The minimum value of the function above corresponds to the *MiCRoM* distance (μ) between the two images, that is, $\mu = min(\sum_{i=1}^{p} \sum_{j=1}^{c} (cost_{ij} \times flow_{ij}))$. Despite of the differences in the modeling of the problem, *MiCRoM* gives the optimal solution for the comparison of segmented images that the greedy approach adopted in *IRM* sometimes fails to obtain. In fact, the *IRM* distance can be thought as a greedy function to solve the transportation problem (as defined above) that gives as much flow as possible to the arcs with the smallest cost.

The minimum-cost network flow problem is a linear program with a very special structure [2]. As such, specialized algorithms can find solutions much faster than plain linear programming algorithms. A large number of efficient algorithms for this specialized instance of the problem are available. In our case, we used the CS2 code developed by Cherkassky and Goldberg[2]. CS2 is a an efficient implementation of a scaling push-relabel algorithm for the minimum-cost flow/transportation problem [36].

An example of two images and the modeling of their comparison as a transportation problem can be viewed in Figure 5.2. Image X is composed by three regions a, b and c, and image Y is composed by regions d and e. A single number represents the visual feature of each region. This number and also the size of the regions are normalized between [0,1]. For example, $size(a) = 0.5$ and $size(b) = 0.25$. The comparison of images X and Y is modeled as a transportation problem in the following way.

Each region of image X is modeled as a producer node, where the production is given by the normalized size of the region. Similarly, each region of image Y is modeled as a consumer node, with a demand given by its size (remember that a demand is represented by a negative value). Each arc between pairs of producer/consumer nodes has a cost given by the distance (d_{reg}) between the corresponding regions. In this example, this distance is given by the absolute difference of the numerical properties of the regions.

A solution for the transportation problem modeled on top of Figure 5.2 can be viewed on the bottom part of the same figure. As can be seen, half of node a's production (0.25) was transported to node d with cost 0.2. The other half (0.25) was transported to node e with cost 0.7. All production of node b (0.25) was transported to node e with cost 0.3, filling the demand of that node. Finally, the total production of node c (0.25) was transported to node d with cost 0. The minimum transportation cost in this network is thus $(0.25 \times 0.2) + (0.25 \times 0.7) + (0.25 \times 0.3) + (0.25 \times 0.0) = 0.3$. The bottom-right part

[2]http://www.intertrust.com/star/goldberg/soft.html

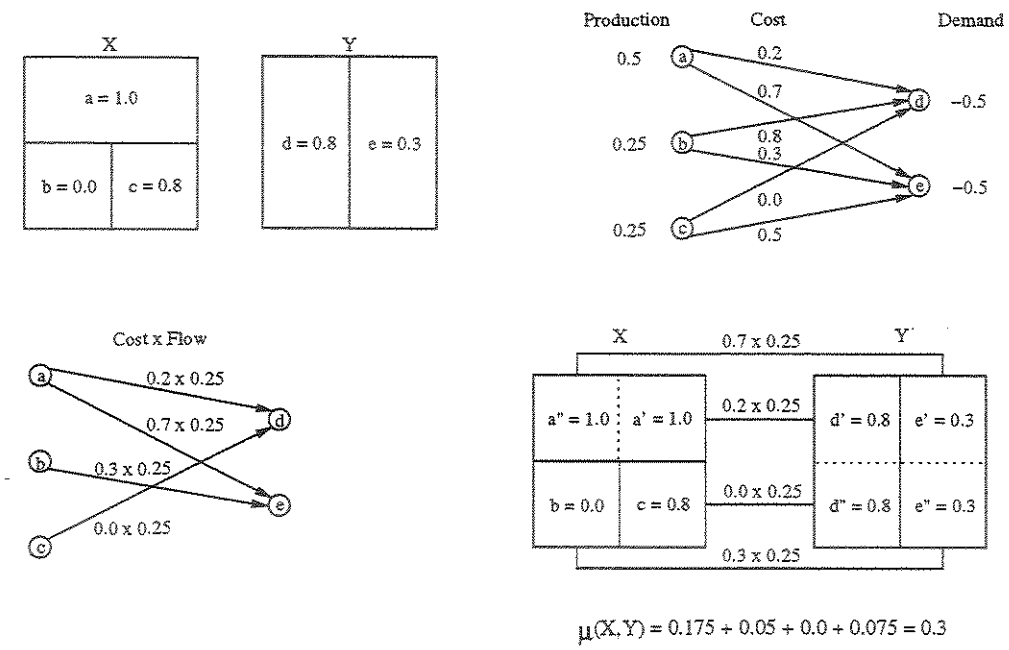

Figure 5.2: Modeling the comparison of segmented images as a transportation problem

of Figure 5.2 shows how the solution of the transportation problem maps back on the compared images. In this particular example, the *IRM* distance is exactly the same as *MiCRoM*, i.e., $\mu(X,Y) = irm(X,Y)$. However, as it was shown in the previous section, this is not always the case.

5.1.1 *MiCRoM* metric properties

The *MiCRoM* distance decomposes the "real" regions of the images in "virtual" subregions to compute the minimum distance between them. The regions obtained after the virtual decomposition have very interesting properties:

- The number of regions of the compared images becomes the same.
- The obtained regions are the ones that minimize the distance between the two images, according to the model adopted (transportation problem).
- There is a one-to-one match between regions of the two images.
- Matched regions have the same size.

The above properties ensure that the distance between images is optimal and that the full content of the images is compared. These properties are also useful to show that the *MiCRoM* distance is a metric. By construction, it is clear that the *MiCRoM* distance satisfies the axioms of positiveness, symmetry and reflexivity. Next, it will be shown that this distance also satisfies the triangular inequality property. The demonstration assumes that the distance d_{reg} (used to compare individual regions of images) is a metric.

Consider the triangular comparison of three images (X, Y and Z) at the level of virtual regions. Assume that a virtual region X_i of image X matches with a virtual region Y_j of image Y. Similarly, assume that the virtual region Y_j matches with a virtual Z_k of image Z, and the virtual region Z_k matches with a virtual region X_l of image X, closing a triangular match for a particular virtual region. As shown in Figure 5.3, there are two possible relations between the virtual regions X_i and X_l of image X: either $X_i = X_l$ or $X_i \neq X_l$. We call the first case a cyclic match, because the virtual region that started the triangular match is the same that ends the process. The second case is called an acyclic match, as the regions that started and ends the triangular match are different.

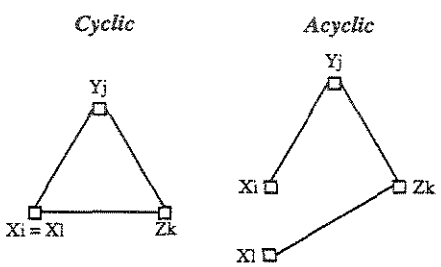

Figure 5.3: Two alternatives for the triangular comparison of virtual regions

Initially, let us suppose that the application of the *MiCRoM* distance to compare images X, Y and Z, results only in cyclic matches ($X_l = X_i$) at the level of virtual regions. As we are assuming the cyclic property only when images X, Z are compared (closing the triangular comparison of the images), this specific *MiCRoM* distance (with the additional restriction of cyclic matches) is represented as $\mu_{cyclic}(X, Z)$.

We know that in the case of cyclic matches, $d_{reg}(X_i, Z_k) \leq d_{reg}(X_i, Y_j) + d_{reg}(Y_j, Z_k)$ for any regions X_i, Y_j and Z_k, as we assumed that d_{reg} is a metric. We also know that the *MiCRoM* distance is only a linear combination of d_{reg} distances. As the linear combination of metric distances is also a metric, we have that, for the case of cyclic matches of virtual regions, $\mu_{cyclic}(X, Z) \leq \mu(X, Y) + \mu(Y, Z)$.

The assumption of cyclic matches at the level of virtual regions does not guarantee that the obtained distance is optimal, because this is not a restriction of our model. However, as the *MiCRoM* distance is optimal, we have that $\mu(X, Z) \leq \mu_{cyclic}(X, Z) \leq$

$\mu(X,Y) + \mu(Y,Z)$, i.e., independently of the use of acyclic matches of virtual regions, the optimality of the *MiCRoM* distance always guarantee that the triangular inequality property holds.

5.2 Effectiveness evaluation

This section presents our experimental results related to the effectiveness of the *MiCRoM* metric distance. We have compared *MiCRoM* with the *IRM* distance, under the same segmentation scheme. In order to have a reference, we have also included the results obtained when images are represented by their global color histograms (*GCH*), and compared with the L_1 vectorial distance. We have used histograms with 64 uniformly quantized colors in RGB color-space.

The experiments used a collection of 20,000 heterogeneous images[3], composed by 200 distinct image domains, each one with 100 JPEG images. The *MiCRoM* and *IRM* distances were used to compare regions obtained with the $CBC(3, 0.1)$ algorithm as described in Section 4.5. This configuration offers an intermediate compromise between the number of obtained regions (which affects the space overhead and the query processing time) and the retrieval effectiveness. With this configuration, each image within our reference collection was segmented (in average) in 40 connected regions. Each region of an image is represented by its average color in the Lab color-space (3 values), its size (1 value), and the spatial coordinates of its geometric center (2 values). Thus, each region of an image is represented by 6 float-point numbers (fpns), and an image is represented by $6 \times 40 = 240$ fpns in average. The distance between regions of two images (d_{reg}) is a weighted composition of the distances between the average color and between the spatial position of the regions.

Since it is generally difficult to express low-level features of images, the Query-By-Example (QBE) paradigm was adopted, where an image is given as example and the system retrieves the most similar matches for this image. The effectiveness of the approaches was evaluated using a set of 18 query images, selected from our reference collection of images. The set of images accepted as relevant for each query image (RRSet) was determined a priori, using a technique similar to the *pooling method* adopted in TREC conferences [118, 115]. We extracted the set of relevant images (for a given query) from a pool of possible relevant images. This pool is created by taking the top 30 images retrieved by each compared approach. The pool of candidate images was then visually analyzed to ultimately decide on the relevance of each image. The subset of relevant images in the pool is the RRSet of the query image. We evaluated the effectiveness of the approaches

[3]Corel GALLERY Magic 65,000 - Stock Photo Library 2.

using $P \times R$ curves (Section 2.7.1).

The results of the effectiveness comparison can be viewed in Figure 5.4. The best overall results were obtained with the *MiCRoM* metric distance, followed by the *IRM* distance. In both cases, the comparison was based on the regions obtained with the *CBC* clustering algorithm. As can be seen, both results are better than the use of a *GCH* to represent images plus a geometric distance (L_1) to compare these histograms. The advantage of *MiCRoM* over *IRM* is evident, but not very large. This means that the *IRM* distance, although not a metric, is a good approximation for the *MiCRoM* metric distance in terms of effectiveness and also in terms of efficiency, as it is a less expensive distance in computational terms. However, the *MiCRoM* metric distance, besides being a little better in terms of effectiveness, has the advantage that its metric properties can be used to speedup the query processing using filtering techniques and/or access methods.

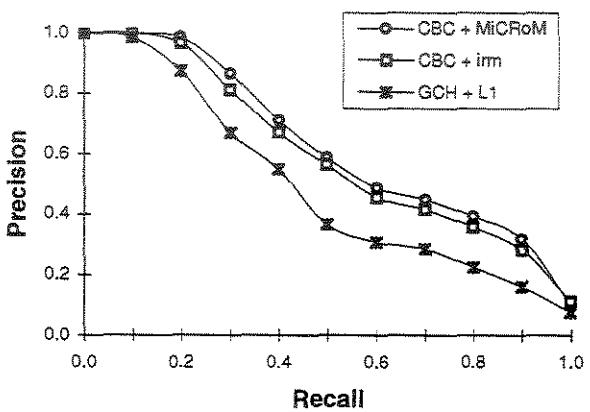

Figure 5.4: *MiCRoM* effectiveness results

For small collections, the combination of a efficient distance like *IRM* and a linear scan of the image database is an interesting approach. However, for large databases, independently of its computational complexity, the use of a metric distance like *MiCRoM* becomes more attractive as it is possible to reduce the query time making extensive use of the triangular inequality property. In the next section, we will investigate a filtering technique that reduces the CPU time to process a visual query when complex distances like *MiCRoM* are used to compare images.

5.3 Filtering based on metric distances

Since there are efficient techniques to cope with vectorial spaces, application designers try to give their problems a vectorial space structure. A common reduction consists of mapping a general metric space into a projected vectorial space. A query processed in

the vectorial space generates a *candidate list* of images that should be analyzed in the original metric space in order to eliminate false-positives.

The space reduction as discussed above is obtained by defining k images of the database as reference, computing the *MiCRoM* distance between the database images and the reference images, storing these distances as k-dimensional vectors, and using a simple and efficient geometric distance to filter out non-relevant images in the vectorial space (at query time). Santos et al called this space reduction *Omni-concept* [84]. They proposed the *HF-algorithm* to define the k reference images (*foci*) used to generate the k-dimensional vectorial space (*omni-space*). The sequential scan of the omni-space was called *Omni-sequential.*

The omni-sequential algorithm makes extensive use of the triangular inequality property to eliminate non-relevant images at query time. In order to illustrate this process, let us consider Q a query image, D a database image, F_i the ith focus used to generate the k-dimensional omni-space ($1 \leq i \leq k$), and a query radius r. The database image D is a candidate image only if the following inequality holds:

$$max_{1 \leq i \leq k} |\mu(Q, F_i) - \mu(F_i, D)| \leq r \tag{5.1}$$

Notice that the distances $\mu(Q, F_i)$ and $\mu(F_i, D)$ are known at query time, as they correspond to the ith omni-coordinate (in the omni-space) of images Q and D, respectively.

In our filtering experiments, we adopted the omni-sequential algorithm. As discussed in previous section, our reference collection of images has 20,000 images. The results presented are relative to the 18 query images used in the effectiveness evaluation discussed in previous section.

The proportion of the database filtered out using the omni-sequential algorithm was evaluated by varying the number of foci between 1 and 10. The foci images were selected according to the HF-algorithm. We used query radius varying between 0.005 and 0.1 (as the distances are normalized, the maximum distance between two images is 1.0). On the left of Figure 5.5, it is shown the relation between the query radius and the average number of images retrieved, i.e., the number of images with a *MiCRoM* distance to the query images smaller than the query radius.

As can be seen, in order to retrieve the top 100 most similar images to a query image, in average, a query radius of 0.045 is enough. A query radius of 0.1 (not shown in the Figure) is sufficient to retrieve, in average, the top 9039 most similar images to the query image. This is approximately half of the database size.

On the right of Figure 5.5, it is shown the degree of filtering using query-radius between 0.05 and 0.045, according to the number of foci used. As can be seen, independently of the query radius used, the ideal number of foci seems to be 4. After this point, the proportion of the database filtered out does not increase substantially. For example, for a query radius

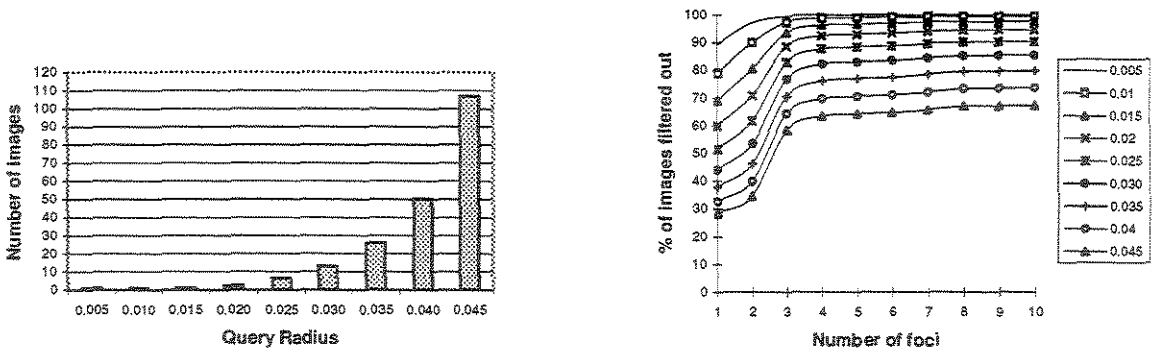

Figure 5.5: *MiCRoM* filtering results

of 0.045, 63.45% of the image database was filtered out using only 4 foci. This means that 2/3 of the database was pruned without computing the *MiCRoM* distance, but only using the L_1 distance in the 4-dimensional omni-space. This proportion grows to only 67.34% when 10 foci are used. This behavior is the same for all query radius tested.

As the time to compare two 4-dimensional vectors using the L_1 distance is much smaller than the comparison of the regions of two images using the *MiCRoM* distance, we can say that the gain in CPU time using omni-sequential (for a query radius of 0.045) is almost of 2/3 when compared to a linear scan of the image database.

In order to reduce the I/O time to process a visual query, it is possible to index the generated 4-dimensional vectorial space using a spatial access method (SAM) such as the R*-tree [10]. SAMs reduce the comparison of images only to those near the query image. In this way, only a portion of the omni-space need to be read from the disk, further reducing the number of I/O operations to process a visual query.

5.4 Chapter conclusion

This chapter presented *MiCRoM* (Minimum-Cost Region Matching), an effective metric distance to compare the visual content of segmented images. *MiCRoM* models the comparison of the regions of two images as a minimum-cost network flow problem [2]. Our experimental results show that the *MiCRoM* metric is at least as effective as the *IRM* distance [58, 100]. This result shows that the greedy approach adopted in *IRM*, although not optimal, gives results very close to the results obtained with *MiCRoM* metric, with the advantage of being less complex. However, the main disadvantage of *IRM* is that it is not a metric distance and so, it is useful only when the image database is relatively small. The *MiCRoM* metric, although computationally more complex than *IRM*, is not only slightly more effective, but more importantly, it has the great advantage that it allows the use

of the triangular inequality property in filtering techniques [84] and/or access methods [20, 35]. This yields substantially reductions in query processing time and a much broader context of application than *IRM*.

Capítulo 6

BIC – Border/Interior Pixel Classification

This chapter presents *BIC* (Border/Interior pixel Classification), a compact and efficient CBIR approach suitable for broad image domains[1]. The *BIC* approach has three main components: (1) a simple and powerful image analysis algorithm that classifies image pixels as border or interior; (2) a new distance to compare histograms – *dLog*; (3) a compact representation for the visual features extracted from images. Our experimental results show that the *BIC* approach is consistently more compact, more efficient and more effective than state-of-the-art CBIR approaches based on sophisticated image analysis algorithms and complex distance functions. The *BIC* image analysis algorithm runs in linear time on the image size, and the obtained visual features can be stored in mere 64 bytes of memory. Our experimental results also show that the *dLog* distance function has two main advantages over vectorial distances (e.g., L_1): it is able to increase substantially the effectiveness of histogram-based CBIR approaches and at the same time to reduce by 50% the space requirement to represent a histogram.

The remainder of this chapter is organized as follows. In Section 6.1, we discuss limitations and drawbacks of the use of regional CBIR approaches in broad image domains, and introduce the *BIC* approach. Section 6.2 presents our experimental setup in terms of reference collection of images, query images, set of relevant images and retrieval effectiveness measures. Our experimental results are discussed in Section 6.3. Finally, Section 6.4 presents the chapter conclusions.

[1] This chapter will be published in the *"Proceedings of the ACM CIKM'2002 International Conference"* [101].

6.1 The *BIC* approach

During the last years, we have worked with regional CBIR approaches. In this period, we could observe the potential and also the limitations of these approaches when applied to large collections of heterogeneous images (broad image domains). A *narrow image domain* has a limited and predictable variability in all relevant aspects of its appearance. Collections of fingerprints, faces recorded over a clear background, and X-rays of the human brain are examples of narrow image domains. A *broad image domain*, on the other hand, has an unlimited and unpredictable variability of the image's content. In general, the interpretation of the image's content is not unique, and the collection of images is very large. As a consequence, it is not possible to use semi-automatic techniques and domain-dependent knowledge during the analysis and comparison of images. Moreover, the image analysis algorithm and also the distance function used to compare segmented images should be as general as possible.

Our experience taught us that, even using very general image properties and automatic segmentation algorithms, it is possible to obtain very good segmentation results in the sense that the obtained regions match very much with the visual properties observed by users. The main drawback of these algorithms is that sometimes the obtained regions are only part of a real object, i.e., an object a user would likely identify by looking at the image. Thus, it does not have a semantic by itself and should be combined with some neighbor regions in order to represent a meaningful object. This problem is treated in general at query time, by using complex distance functions to compare weakly segmented images.

A second drawback of the automatic image segmentation algorithms is that the criterion of homogeneous visual properties usually leads to a super segmentation of the image. As a result, a precise representation of the obtained regions is prohibitive in terms of storage space, and their comparison using a complex distance function is impractical. The aforementioned problems become even more critical if one recalls that the number of regions per image is variable and the obtained regions are also variable in size, shape and spatial location.

In order to keep the problem of representing and comparing segmented images tractable, the output of the segmentation algorithm is usually simplified, relaxing properties in order to preserve only a few regions, and also representing approximately the remaining regions. As it is not possible to use additional knowledge about the content of the images (domain-dependent knowledge) to perform this simplification, the consequence is that the effectiveness of the approach is reduced in the same proportion in which the problem is simplified. If the result of the image analysis algorithm must be relaxed in order to keep the problem tractable in computational terms, it is very likely that the algorithm used is

not the most adequate one for the problem in hand.

As we show in forthcoming sections, the key to reach efficient and effective CBIR systems in broad image domains is the use of simple and robust image analysis algorithms whose result can be preserved (without approximations) during the representation and the comparison of the visual features. There is no point to use complex image analysis algorithms if the properties of these algorithms must be relaxed and sometimes discarded in order to make the representation and the comparison of the images a tractable problem.

Next we present *BIC* (Border/Interior pixel Classification), a new CBIR approach suitable for broad image domains. The *BIC* approach has three main components: (1) a simple and powerful image analysis algorithm that classifies image pixels as border or interior; (2) a new logarithmic distance to compare histograms; (3) a compact representation for the visual features extracted from images. Each of these components is explained in details in the following subsections.

6.1.1 Image analysis

The algorithm for image analysis in *BIC* approach relies on the RGB color-space uniformly quantized in 4×4×4=64 colors. It is important to notice that any other color-space and quantization scheme could be used as well. We chose this configuration because it is widely used and it is effective, as discussed in Section 4.5. Another reason is to have fair comparisons with other histogram-based CBIR approaches we have implemented that also rely on the same scheme (RGB uniformly quantized in 64 colors). We normalize the pixel count of each histogram bin between 0 and 255. This normalization is helpful because if we approximate the pixel count to integer values in the interval [0,255], we are able to represent a histogram bin using only one byte of memory. We have also observed in practice that there is no clear advantage in using more than 255 distinct values per histogram bin.

After the quantization step, image pixels are classified in border or interior pixels. A pixel is classified as border if it is at the border of the image itself or if at least one of its 4-neighbors (top, bottom, left and right) has a different quantized color. A pixel is classified as interior if its 4-neighbors have the same quantized color. It is important to observe that this classification is mutually exclusive (either a pixel is border or it is interior) and it is based on a inherently binary visual property of the images. We choose 4-neighbors instead of 8-neighbors because, given the simplicity and generality of the problem, the use of 4-neighbors is able to reduce the image analysis complexity without perceptual losses in terms of retrieval effectiveness.

After the image pixels are classified, one color histogram is computed considering only border pixels, and another color histogram is computed considering only interior pix-

els. In this way, we have the border/interior classification represented for each quantized color. A binary classification of image pixels was also proposed within the *CCV* approach (Section 2.6.1). However, the *CCV* binary classification is based on a non-binary visual property of the images – the size of the connected components. In order to have a binary classification in *CCV*, an empirical size threshold was introduced and most of the useful information about the size of the connected components was lost in this reduction. Moreover, the approach may be very sensitive to the chosen threshold that, in practice, should vary according to the visual content of the images. The consequence is that the *CCV* approach is only a little bit more effective than a simple *GCH*, as shown in Section 6.3. The implication of the approximation introduced in *CCV* in terms of effectiveness follows the discussion presented in Section 6.1.

The classification of the pixels in border/interior for each quantized color is much more discriminative than a simple *GCH* or *CCV*, as shown in the experimental results of Section 6.3. This discriminative power can be analyzed for each individual color in terms of shape, texture and connected components. If the number of interior pixels for a given color is smaller than the number of border pixels for the same color, than at least one of the following visual properties is true: (1) the color is distributed in relatively large regions with very irregular shape; (2) the color is distributed in small connected regions such that the border of each region is larger than its interior; (3) the color is part of an image region that is rich in texture information. Similarly, if the opposite situation is true, i.e., the number of border pixels for a given color is smaller than the number of interior pixels for the same color, than we can conclude that (4) the color is distributed in relatively large and homogeneous regions with regular shape. The degree to which each of the four aforementioned visual properties is true depends on the portion of the image covered by and also on the proportion between border/interior pixels for each quantized color. Figure 7.8 shows examples of images analyzed in terms of border and interior pixels[2].

6.1.2 *dLog* Distance function

As discussed in previous section, each image is described within *BIC* by means of two color histograms with 64 bins each (one for each quantized color). In fact, these two histograms can be stored and compared as a single histogram with 128 bins. As such, we are able to use any vectorial distance function like L_1 or L_2 to compare the *BIC* visual features. The main advantage of vectorial distances is their efficiency in comparing histograms. Moreover, they allow the use of spatial or metric access methods to speedup

[2]A set of 50 images analyzed in terms of border/interior pixels can be viewed in color at: `http://www.ic.unicamp.br/~973250/cbir/bic.html`.

query processing [103]. The use of access methods is important for large collections of images, as the query processing time should not increase in the same proportion that the image collection increases.

Vectorial distances have also well-known limitations. One of such limitations is that a high value in a single histogram bin dominates the distance between two histograms, no matter the relative importance of this single value [61, 68]. If we think about an image in terms of background/foreground regions, in general it is true that the foreground determines the semantic of the image and as such, it is more important in determining the similarity among images. It is equally true that, in general, the background covers the majority of the image area. Thus, the regions that compose the background are usually larger than the regions that compose the foreground. For instance, consider a set of images where the background covers 60% of the image's content and this background is homogeneous in the sense that it can be represented in just one histogram bin. Now imagine that we perform a similarity search using one of such images as example. What does happen when a vectorial distance is used to compare these histograms? Images having a background with the same color but a different foreground are retrieved ahead of any other image having the same foreground (i.e., a high degree of semantic similarity) but a background with a different color.

In order to deal with this distortion using only the information available within the histogram representation, we propose the *dLog* distance function. The *dLog* function compares histograms in a logarithmic scale, and is defined as:

$$dLog(q,d) = \sum_{i=0}^{i<M} |f(q[i]) - f(d[i])| \tag{6.1}$$

$$f(x) = \begin{cases} 0, & \text{if } x = 0 \\ 1, & \text{if } 0 < x \leq 1 \\ \lceil \log_2 x \rceil + 1, & \text{otherwise} \end{cases} \tag{6.2}$$

In the previous equation, q and d are two histograms with M bins each. The value $q[i]$ represents the ith bin of histogram q and $d[i]$ represents the ith bin of histogram d. The histogram bins are normalized between 0 and 255, as discussed in Section 6.1.1. A similar but experimentally defined encoding function $f(.)$ was also used in [68].

The comparison of histograms with the *dLog* function does not solve the problem of histogram bins with very high values, but diminishes its effects in most of the situations. In a log-scale, the difference between the largest and the smallest distances between histogram bins becomes smaller than in the original scale. In the original scale, the smallest distance between histogram bins is zero (both images have the same amount of a particular color) and the largest distance is 255 (when the images have just one color and

they are different). In our log-scale, the smallest distance is 0 and the largest distance is just 9. The range of distances in the original scale is thus $255/9 = 28$ times larger than in the proposed log-scale.

In Section 6.3 we apply the *dLog* distance function to compare histograms in different histogram-based CBIR approaches. In all cases, the use of the *dLog* function (instead of L_1) increases substantially the effectiveness of the approaches, making simple approaches such as *GCH* almost as effective as a regional approach such as *CBC*.

6.1.3 Representation of visual features

When histograms are compared using the *dLog* distance function, it is possible to store the result of the $f(x)$ function (Equation 6.2) instead of the normalized pixel count. The advantages of this log-based representation for histograms are: (1) the comparison of the histograms according to the *dLog* distance becomes computationally simpler; (2) the histogram can be stored in half of the space of the original representation; (3) as in [69], we can interpret, represent, index and compare histograms as binary signatures.

If the log-based representation is adopted, we can compare histograms using simply the L_1 distance. A careful look at Equation 6.1 reveals that the *dLog* distance is in fact an L_1 distance of the log of the pixel count – $f(x)$. If $f(x)$ is already computed and stored, all we have to do is just compare the log-based represented histograms using the L_1 vectorial distance. Moreover, observing Equations 6.1 and 6.2, and remembering that $0 \leq x \leq 255$, we perceive that $0 \leq f(x) \leq 9$. Thus, $f(x)$ can assume only 10 distinct values and these values can be stored in just 4 bits ($10 < 2^4$). This means that the log-based representation of histograms requires half of the space necessary to store the normalized pixel count (original representation).

The log-based representation allows a reduction of 50% in the required storage space for any histogram-based CBIR approach. In the particular case of the *BIC* approach, each *BIC* histogram has 128 bins (64 for border pixels and 64 for interior pixels). Thus, it is possible to store a *BIC* histogram in just 64 bytes of memory. This is a very compact representation for the visual features of an image. As an example, it is possible to store 16,000 *BIC* histograms in just 1Mbyte of memory. Considering a single desktop PC with 1Gbyte of free RAM memory, it is possible to keep in main memory (for the purpose of similarity search) the *BIC* representation of approximately 16 millions of images. High-end workstations can thus maintain fairly large collections of images in memory, completely avoiding the necessity of disk-based access methods to speedup query processing.

6.2 Experimental setup

In our experiments we adopted the widely used query-by-example (QBE) paradigm, as it seems to be the most adequate way to submit queries in CBIR systems based on low-level visual features. In QBE, an image is given as a visual example of the information needed. This image is analyzed and visual features are extracted. These features are used to measure the distance between the query image and the images stored in the image database. The stored images are retrieved in increasing order of their distance to the query image (similarity-search).

The purpose of our experiments is to evaluate the effectiveness of the similarity-search of different CBIR approaches in retrieving relevant images ahead of non-relevant ones. Effectiveness evaluation is a very complex task. While in textual information retrieval there are several reference collections of documents available (e.g., CACM, ADI, INSPEC, Medilars and ISI) and even a full conference (TREC) dedicated to the issue of effectiveness evaluation [118], in the domain of CBIR the situation is quite different. The CBIR community has not been nearly as active in this respect, though some work has begun to appear recently (e.g. [54, 66]).

In order to evaluate CBIR effectiveness, it is necessary at least a reference collection of images, a set of query images, a set of relevant images for each query image (ground truth), and adequate retrieval effectiveness measures. Next we discuss how we dealt with these requirements in our experiments.

We are using as reference a heterogeneous collection of 20,000 JPEG images from a Corel stock CD[3]. This collection has approximately 200 distinct image domains, each one composed of approximately 100 images. We believe this is a sufficiently large number of distinct domains (and also images per domain) for the purpose of our evaluation study.

Out of the reference collection, we selected 50 images of distinct domains to be used as query images. These images are shown in Figure 7.1. Once the query images were selected, the next step was to establish the set of images inside the reference collection that we accept as relevant for each query image. We call this set of relevant images the *relevant result set* (RRSet) of a query image. Given a query image, an ideal CBIR approach retrieves the images of its RRSet ahead of any other image within the reference collection. We selected the RRSets using a technique similar to the *pooling method* adopted in TREC conferences [118, Ch. 3], which is detailed next.

We extract the RRset for a given query from a pool of possible relevant images. This pool consists of the top 30 images retrieved by each compared CBIR approach. The pool of candidate images is visually analyzed to ultimately decide on the relevance of each image. The subset of relevant images in the pool is the RRSet of the query image.

[3]Corel GALLERY Magic 65,000 - Stock Photo Library 2.

The decision about the relevance of a given image was based on its visual properties, its domain properties and its semantics. Three examples of RRSet[4] are shown in Figures 7.2, 7.3 and 7.4.

In our experiments, we adopted a total of 11 different measures of retrieval effectiveness. We used two graphical measures (Precision vs. Recall and θ vs. Recall), and nine single value measures ($P(r)$, $P(30)$, $R(30)$, $P(100)$, $R(100)$, *3P-Precision*, and *11P-Precision*). Each of these measures evaluates a different aspect of the retrieval algorithm, and their combination gives a clear characterization of effectiveness according to several distinct criteria. Next, these retrieval effectiveness measures are discussed in details.

Precision vs. Recall ($P \times R$) curves [118] are well-known and widely used to evaluate retrieval effectiveness. Precision is defined as the fraction of the retrieved images that are relevant to the query. In contrast, recall measures the proportion of relevant images among the retrieved images. As recall is a non-decreasing function of rank, precision can be regarded as a function of recall rather than of rank. In general, the curve closest to the top of the chart indicates the best performance.

A variation of the $P \times R$ curve we propose is the θ vs. Recall curve ($\theta \times R$). We define θ as the average of the precision values measured whenever a relevant image is retrieved. For 100% of recall, the θ value is equivalent to the average precision used in [17]. The main difference between θ and precision is that, unlike precision, the θ value is accumulative, i.e., its computation considers not only the precision at a specific recall level, but also the precision at previous recall levels. This accumulative computation is more consistent with the ranking imposed by CBIR algorithms. While precision rely on a simple binary property of the retrieved images (relevant or not), the θ value takes into account additionally the ordering of the retrieved images in its computation.

We have also used single-value retrieval effectiveness measures that can be put on a scale to give absolute and relative values. One of such measures corresponds to measure the precision when the number of retrieved images is just sufficient to include all the relevant images for a query. This value is known as *R-value* [118], and we call the precision at this point $P(r)$. We also measure the values $P(30)$, $R(30)$, $P(100)$ and $R(100)$. The first two measures correspond to the precision and the recall after 30 images are retrieved. The choice of the value 30 was based on the fact that it corresponds to the retrieval cutoff point used to determine the RRSets of the query images, as discussed at the beginning of this section. Similarly, we compute the precision and the recall after 100 images are retrieved. This value is an estimative of the number of retrieved images an average user would accept to inspect in order to determine their relevance to his/her needs. Finally, the two other single value measures are the 3-point and the 11-point average precision [118]. The 3-point

[4]The 50 query images used in our experiments and the corresponding RRSets can be viewed in color at: `http:/www.ic.unicamp.br/~973250/cbir/query.html`.

average precision (*3P-Precision*) is computed by averaging the precision taking at three predefined recall levels, typically, 20%, 50% and 80%. The 11-point average precision (*11P-Precision*) is computed by averaging the precision taking at eleven predefined recall levels: 0%, 10%, ... , 90%, 100%.

We have compared the *BIC* approach with four other CBIR approaches, namely *GCH*, *CCV*, *Grid* 9 and *CBC*. The *GCH* and the *CCV* approaches were reviewed in Section 2.6.1. The *Grid* 9 approach is a variation of the basic partition-based approach discussed in Section 2.6.2 that decomposes images using a grid of 3×3=9 equal-sized cells. *CBC* is a regional CBIR approach proposed in Chapter 4. We have adopted the suggested *CBC*(3, 0.1) configuration.

6.3 Experimental results

This section discusses our experimental results relative to the effectiveness of the proposed *BIC* approach. Initially, we compare *BIC* and some CBIR approaches reviewed in Section 2.6, showing that *BIC* outperforms all of them. After that, we evaluate the effectiveness of the *dLog* distance function when used with other histogram-based approaches, and show that it indeed improves the effectiveness of all investigated approaches. We conclude showing that *BIC* still prevails, outperforming all *dLog*-improved approaches.

In Figure 6.1 and in the first lines of Table 6.1, we compare *BIC* with existing CBIR approaches. The results of the eleven measures confirm the general belief that partition-based approaches are more effective than global approaches (*Grid* 9 is better than *GCH*), and that regional CBIR approaches are more effective than partition-based approaches (*CBC* is better than *Grid* 9). The comparison of *CCV* and *GCH* reveals that the pixel classification of *CCV* becomes effective only after 20% of recall. However, the gain in terms of effectiveness obtained with *CCV* approach is not very expressive, especially if one considers its storage overhead. More important, however, is the fact that the proposed *BIC* approach is clearly more effective than all investigated CBIR approaches, including *CBC*.

Besides being more effective than *CBC*, the *BIC* approach is also more compact and efficient. The *BIC* approach is based on a very simple (but powerful) image analysis algorithm that runs in time $O(n)$, where n is the size (in pixels) of the image being analyzed. Moreover, as discussed in Section 6.1.3, the *BIC* visual features can be stored in just 64 bytes of memory, and the comparison of these visual features is based on the very efficient and effective *dLog* distance function. The *dLog* distance function is several orders of magnitude more efficient than the *MiCRoM* metric adopted in *CBC* approach. While a visual query in our reference collection of images takes only a small fraction of a second using the *BIC* approach, in *CBC* this same visual query takes several minutes to

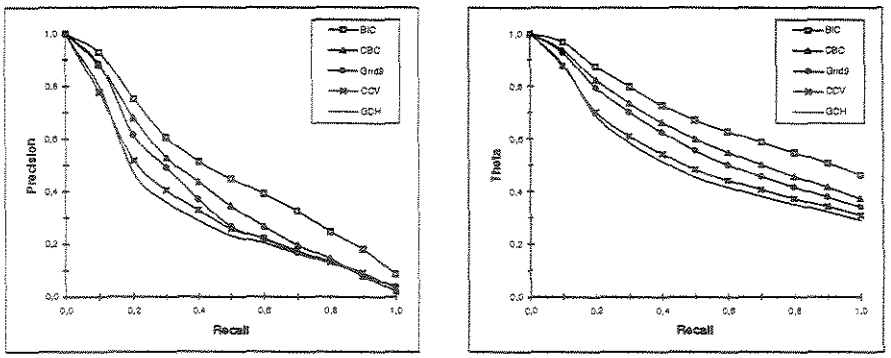

Figure 6.1: *BIC* versus existing CBIR approaches

Table 6.1: Single-valued effectiveness results of *BIC* approach

Approach	3P-Precision	11P-Precision	P(30)	R(30)	P(100)	R(100)	P(r)
BIC	0.48	0.50	0.46	0.44	0.22	0.70	0.44
L_1 *BIC*	0.35	0.39	0.35	0.33	0.19	0.57	0.34
GCH	0.28	0.34	0.30	0.30	0.17	0.52	0.31
CCV	0.30	0.36	0.33	0.32	0.17	0.52	0.32
Grid 9	0.34	0.39	0.35	0.35	0.17	0.56	0.34
dLog GCH	0.38	0.43	0.39	0.37	0.20	0.64	0.39
dLog CCV	0.41	0.44	0.42	0.40	0.20	0.63	0.40
dLog Grid 9	0.40	0.43	0.40	0.40	0.19	0.61	0.39
CBC	0.39	0.42	0.40	0.39	0.18	0.58	0.39

be processed.

The second part of our experiments evaluated the effect of using the *dLog* distance instead of L_1 in existing histogram-based approaches. The effectiveness results of this experiment, which are also supported by the many measures used in Table 6.1, can be observed in Figure 6.2. In that figure, each column is related to a CBIR approach. We have plotted two graphs per approach, comparing its original effectiveness (using L_1 distance) with the effectiveness when *dLog* is used instead of L_1. An exception is the last column where we show how the use of L_1 would adversely affect *BIC* (recall that *dLog* is the "native" distance designed for *BIC*). The top row shows the $P \times R$ graphs while the bottom row shows the $\theta \times R$ graphs.

Observing Figure 6.2 and Table 6.1, one can conclude that the *dLog* distance function clearly increases the effectiveness of all histogram-based approaches tested. This increase in effectiveness is more accentuated in *GCH* and *CCV* than in *Grid* 9. We have observed that, when the *dLog* function is used, the spatial information of *Grid* 9 becomes less

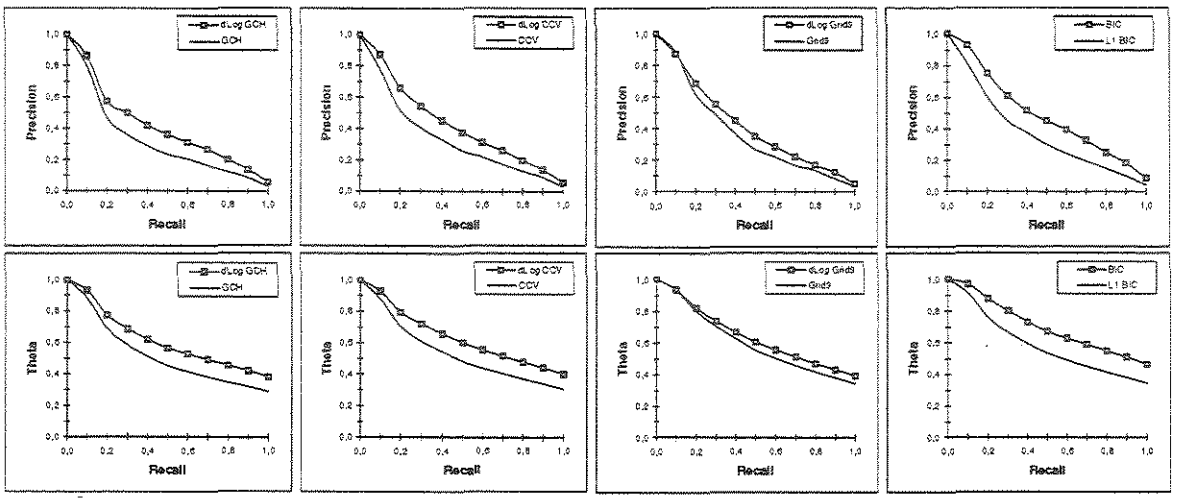

Figure 6.2: Effectiveness results of the *dLog* distance function

important as it is unable to make the *dLog Grid* 9 more effective than *dLog CCV*. We have observed a similar behavior also in the context of the proposed *BIC* approach. We have tried several ways to add spatial information into the *BIC* visual features. However, none of these attempts were successful as they were unable to increase the effectiveness of the *BIC* approach as it was proposed. We explain this behavior in the following way. When the comparison of the visual features is based on less effective distances like L_1, the approaches are able to retrieve only a small fraction of the relevant images for a given query image in the top T retrieved images, where T is a retrieval threshold. In this context, the addition of spatial information is useful because it adds to the set of retrieved images relevant images with similar spatial distribution of colors (that were not originally retrieved). However, if the visual features are compared using more robust and effective distances like the *dLog* distance, the approaches are able to retrieve most of the relevant images for a given query. In this context, if we add restrictions about the spatial distribution of colors, we not only do not include more relevant images to the set of retrieved images (relevant images with similar spatial layout were already retrieved) but, in fact, we eliminate from the set of retrieved images those relevant images that are not similar to the query image in terms of spatial layout of colors.

Finally, observing Figure 6.3, again supported by Table 6.1, we can conclude that the *BIC* approach is clearly more effective than any of the *dLog*-improved histogram-based approaches, including *dLog CCV*. As the *dLog CCV* uses the same representation and distance function used in *BIC*, we can conclude that this gain in effectiveness is due solely to the *BIC* image analysis algorithm. As discussed in Section 6.1.1, the binary classification of image pixels in border/interior adopted in *BIC* is more robust and effective

than the classification adopted in *CCV*, that makes a binary classification of pixels based on a non-binary image property – the size of the connected regions. Figure 7.6 shows an example of the top 30 images returned in response to a visual query using the *BIC* approach[5]. The query image in this example is the first image retrieved.

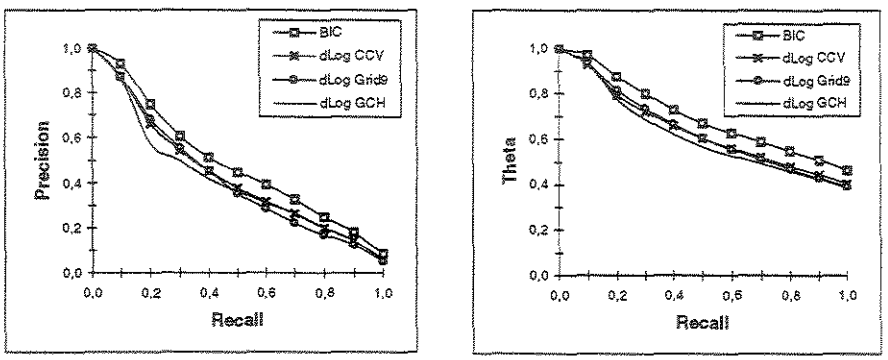

Figure 6.3: *BIC* versus the *dLog*-improved CBIR approaches

6.4 Chapter conclusion

This paper presented *BIC* (Border/Interior pixel Classification), a compact and efficient CBIR approach for broad image domains. The *BIC* approach has three main components: (1) a simple and powerful image analysis algorithm that classifies image pixels as border or interior; (2) a new logarithmic distance to compare histograms; (3) a compact representation for the visual features extracted from images.

The *BIC* image analysis algorithm makes a binary classification of image pixels in border or interior. Our experimental results show that the *BIC* approach is consistently more effective than state-of-the-art regional CBIR approaches based on very sophisticated image analysis algorithms, but that introduces several post-processing simplification steps in order to maintain the representation and the comparison of segmented images a manageable problem (in computational terms).

The second component of the *BIC* approach is the *dLog* metric distance function. This function compares two histograms according to a log scale, diminishing distortions in the measured distance generated by histogram bins with very high values. As our experimental results show, the use of the *dLog* function has two major advantages over vectorial distances like L_1. First, the *dLog* function clearly increases the effectiveness of any histogram-based CBIR approach. Second, the use of this function allows a log-based

[5]The top 30 images retrieved by the *BIC* approach for all 50 query images used in our experiments can be viewed in color at: `http://www.ic.unicamp.br/~973250/cbir/bic30.html`.

representation for the histograms that makes possible to store a histogram bin in just 4 bits of memory. This log-based representation reduces the space required to store a histogram in any histogram-based CBIR approach. In the particular case of *BIC* approach, each *BIC* histogram has 128 bins. Thus, it is possible to store a *BIC* histogram in just 64 bytes of memory. This is a very compact representation for the visual features of an image.

Capítulo 7

Conclusões e Trabalhos Futuros

A recuperação de imagens por conteúdo (CBIR) é uma área que 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 dos equipamentos de captura, transmissão e armazenamento de imagens; (2) o crescimento exponencial do número de imagens e vídeos publicados na internet; (3) os desafios científicos envolvidos e as inúmeras aplicações práticas em sistemas como máquinas de busca, bibliotecas digitais, sistemas de segurança e em bancos de dados médicos e bancos de dados geográficos; (4) a necessidade de integração de técnicas de reconhecimentos de padrões, análise e interpretação de imagens, banco de dados, recuperação de informação, interfaces homem-máquina dentre outras; (5) a inadequação de técnicas tradicionais de banco de dados e recuperação de informação, bem como de técnicas semi-automáticas (com intervenção humana) para descrever, representar e realizar buscas em grandes coleções de imagens.

Nosso trabalho concentrou-se em técnicas de CBIR que pudessem ser aplicadas em grandes coleções de imagens heterogêneas. Nesse tipo de coleção, não se pode assumir nenhum tipo de conhecimento sobre o conteúdo semântico e/ou visual das imagens, e o custo de utilizar técnicas semi-automáticas (com intervenção humana) é alto em virtude da heterogeneidade e do volume das imagens que precisam ser analisadas. O exemplo clássico desse tipo de repositório é o conteúdo visual da *World-Wide Web* – WWW.

Mais especificamente, nós nos concentramos na informação de cor presente nas imagens. A cor é uma das características visuais mais amplamente utilizadas em técnicas de CBIR por ser simples, intuitiva, estar presente na maioria das imagens e fornecer excelentes resultados. Nosso trabalho enfocou três tópicos que consideramos importantes para se realizar a recuperação de imagens por conteúdo utilizando informação de cor: (1) como analisar e extrair informação de cor das imagens de forma automática e eficiente; (2) como representar essa informação de forma compacta e efetiva; (3) como comparar de maneira efetiva e eficiente as características visuais que descrevem duas imagens. Adicionalmente,

foram investigadas técnicas e medidas para avaliar a efetividade de sistema de recuperação de imagens por conteúdo. Apesar de não termos explorado formalmente a indexação das características visuais extraídas das imagens, os requisitos para uma indexação eficiente foram uma preocupação constante em todas as técnicas que propusemos.

No Capítulo 3, nós propusemos e avaliamos uma representação alternativa e mais compacta para abordagens de recuperação de imagens baseada em particionamento denominada *CCH* (*Cell/Color Histograms*). Adicionalmente, foi proposta uma generalização da função de distância L_1 (*City-block*) para comparar os histogramas utilizados na abordagem *CCH*. Nós também propusemos uma metodologia de avaliação de efetividade baseada em uma nova medida denominada θ_{rel}.

Nesse capítulo foi realizado um experimento que mostrou que, considerando-se o espaço de cores RGB uniformemente quantizado em $4 \times 4 \times 4 = 64$ cores e a nossa coleção de 20.000 imagens heterogêneas, em média cada imagem era composta por apenas 29 das 64 cores possíveis. Também foi observado que 90% do conteúdo de uma imagem pode ser descrito (em média) utilizando apenas 9 das 64 cores possíveis. A comparação com abordagens existentes confirmou que as abordagens baseadas em particionamento, apesar de utilizarem consideravelmente mais espaço para representar as imagens, também oferecem ganhos em termos de efetividade. Nesse sentido, a abordagem *CCH* permitiu uma redução de 55% no espaço utilizado quando comparada com abordagens tradicionais de particionamento, sem implicar em perda de efetividade. Adicionalmente, foi demonstrado experimentalmente que quanto maior o número de células do particionamento, maior a efetividade das abordagens e maior o espaço utilizado. Também foi investigada a possibilidade de representar parcialmente o conteúdo das imagens. Foi observado um grande ganho de espaço (sem comprometer sensivelmente a efetividade) quando apenas cerca de 90% do conteúdo das imagens foi representado.

O Capítulo 4 apresentou o *CBC* (*Color-Based Clustering*), uma nova abordagem regional para a recuperação de imagens baseada em informação de cor. O *CBC* segmenta imagens automaticamente, tem uma implementação eficiente e é independente do domínio das imagens, permitindo sua aplicação em grandes coleções de imagens heterogêneas. O algoritmo de análise das imagens tem complexidade computacional $O(n \log n)$, onde n é o número de pixels da imagem de entrada. Os resultados experimentais mostraram que as três variações da abordagem *CBC* que testamos foram mais efetivas que 5 outras abordagens comparadas, incluindo o *CCH*. As variações do *CBC* mostraram-se mais robustas em relação ao crescimento da coleção de imagens, e também mais compactas em termos de utilização de espaço.

O Capítulo 5 apresentou *MiCRoM* (*Minimum-Cost Region Matching*), uma função métrica para a comparação de imagens segmentadas. A função *MiCRoM* é uma extensão da função *IRM* (não-métrica) proposta em [58]. A função *MiCRoM* fornece a distância

ótima entre duas imagens (de acordo com a modelagem do problema adotada) que a abordagem gulosa utilizada na função *IRM* algumas vezes não consegue obter. Esse trabalho também introduziu a idéia de determinar o conjunto de imagens relevantes (RRSet) das imagens consulta utilizando uma técnica de *pooling* similar àquela utilizada nas conferências TREC [115, 118].

Os resultados experimentais mostraram que a distância *MiCRoM* é ao menos tão efetiva quanto a distância *IRM*. Esse resultado comprova que a estratégia gulosa adotada pela *IRM* na prática funciona muito bem, pois os resultados de efetividade são quase tão bons quanto os resultados obtidos com a *MiCRoM* (versão ótima da distância *IRM*). A vantagem da *MiCRoM* é ser uma função métrica que permite a utilização da propriedade da desigualdade triangular para acelerar o processamento de consultas. Com base nisso, foi demonstrado experimentalmente que a utilização de uma técnica de filtragem baseada na propriedade da desigualdade triangular reduziu em 2/3 o tempo gasto para se realizar uma busca pelos 100 vizinhos mais próximos de uma imagem.

O Capítulo 6 apresentou *BIC* (*Border/Interior Pixel Classification*), uma nova abordagem para a recuperação de imagens por conteúdo em grandes coleções de imagens heterogêneas. A abordagem *BIC* tem três componentes principais: (1) um algoritmo simples, eficiente e poderoso para a análise do conteúdo visual das imagens, (2) uma nova função de distância para a comparação de histogramas de cores denominada *dLog*, e (3) uma representação compacta para as características visuais extraídas das imagens.

Nesse trabalho foram utilizadas 50 imagens consulta e um total de 11 medidas diferentes para se avaliar a efetividade da abordagem *BIC*. Dentre essas medidas, está uma nova medida gráfica a qual denominamos $\theta \times R$. Essa medida é uma variação da medida de $P \times R$ que se mostrou mais adequada ao contexto de recuperação de imagens por conteúdo e também mais fácil de ser interpretada. A comparação com abordagens existentes (incluindo o *CBC*) confirmou que a abordagem *BIC* é consideravelmente mais efetiva que as demais. Além de ser mais efetiva, a abordagem *BIC* é também mais compacta e mais eficiente. Um segundo experimento avaliou a utilização da distância *dLog* em várias abordagens baseadas em histogramas de cores. Em todos os casos, houve um ganho sensível de efetividade em comparação com a utilização da função L_1. Além do ganho de efetividade, a utilização da função *dLog* permitiu reduzir pela metade o espaço necessário para armazenar os histogramas.

O nosso trabalho pode ser estendido de várias formas diferentes. Algumas extensões imediatas seriam: (1) técnicas para tratar consultas baseadas em regiões da imagem ao invés de utilizar a imagem inteira como exemplo; (2) utilização de características visuais relacionadas à informação de textura, forma, posição e relações topológicas entre regiões da imagem; (3) investigação de funções de distância para comparar essas novas características visuais; (4) utilização de características visuais e funções de distância

dependentes do domínio das imagens (em aplicações específicas); (5) projeto de uma linguagem de consulta visual; (6) projeto de uma interface de consulta visual e interativa; (7) implementação de uma aplicação protótipo, por exemplo, uma máquina de busca para pesquisar o conteúdo visual da WWW; (8) aplicar as técnicas propostas no contexto de recuperação de vídeo baseada em conteúdo, considerando adicionalmente o aspecto temporal desse domínio; (9) utilização de técnicas de *relevance feedback* para introduzir um caráter semântico ao processo de recuperação de imagens por conteúdo, em particular quando são utilizadas características visuais de baixo nível como distribuição de cores; (10) investigar a possibilidade de realizar a indexação aproximada de funções de distância não-métricas como a *IRM*; (11) investigar a utilização de métodos de acesso para acelerar o processamento de consultas visuais em coleções que sejam compostas por um número extremamente elevado de imagens, ou em abordagens onde a representação das imagens tenha um tamanho não-trivial (por exemplo, na abordagem *CBC*) e/ou a função de distância tenha uma elevada complexidade computacional (por exemplo, a função *MiCRoM*).

Referências Bibliográficas

[1] C.C. Aggarwal, A. Hinneburg, and D.A. Keim. On the surprising behavior of distance metrics in high dimensional space. In *Proc. of the ICDT Intl. Conf*, pages 420–434, 2001.

[2] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[3] D. Androutsos, K.N. Plataniotis, and A.N. Venetsanopoulos. Vector angular distance measure for indexing and retrieval of color. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases VII*, pages 604–613, 1999.

[4] A.R. Appas, A.M. Darwish, A.I. El-Desouki, et al. Image indexing using composite regional color channels features. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases VII*, pages 492–500, 1999.

[5] P. Arabie, L.J. Hubert, and G. De Soete. *Clustering and Classification*. World Scientific, 1996.

[6] F.G. Ashby and N.A. Perrin. Toward a unified theory of similarity and recognition. *Psychological Review*, 95(1):124–150, 1988.

[7] J. Ashley, R. Barber, M. Flickner, et al. Automatic and semi-automatic methods for image annotation and retrieval in QBIC. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases III*, pages 24–35, 1995.

[8] Y.A. Aslandogan and C.T. Yu. Techniques and systems for image and video retrieval. *IEEE TKDE*, 11(1):56–63, 1999.

[9] J.R. Bach, S. Paul, and R. Jam. A visual information management system for the interactive retrieval of faces. *IEEE TKDE*, 5(4):619–628, 1993.

[10] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: An efficient and robust access method for points and rectangles. In *Proc.of ACM SIGMOD Intl. Conf.*, pages 322–331, 1990.

[11] J.L. Bentley. Multidimensional binary search trees in database applications. *IEEE TSE*, 5(4):333–340, 1979.

[12] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is "nearest neighbor" meaningful? In *Proc. of the ICDT Intl. Conf*, pages 217–235, 1999.

[13] A. Del Bimbo. *Visual Information Retrieval.* Morgan Kaufmann, 1999.

[14] C. Bohm, S. Berchtold, and D.A. Keim. Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. *ACM Computing Surveys*, 33:322–373, 2001.

[15] T. Bozkaya and M. Ozsoyoglu. Distance-based indexing for high-dimensional metric spaces. In *Proc. of ACM SIGMOD Intl. Conf.*, pages 357–368, 1997.

[16] S. Brin. Near neighbor search in large metric spaces. In *Proc. of VLDB Intl. Conf.*, pages 574–584, 1995.

[17] C. Buckley and E.M. Voorhees. Evaluating evaluation measure stability. In *Proc. of the ACM SIGIR Intl. Conf.*, pages 33–40, 2000.

[18] C. Carson, M. Thomas, S. Belongie, et al. Blobworld: A system for region-based image indexing and retrieval. In *Proc. of VISUAL Intl. Conf.*, pages 509–516, 1999.

[19] S.-K. Chang, Q.-Y. Shi, and C.-W. Yan. Iconic indexing by 2-d strings. *IEEE PAMI*, 9(3):413–428, 1987.

[20] E. Chavez, G. Navarro, R.B. Yates, et al. Searching in metric spaces. *ACM Computing Surveys*, 33(3):273–321, 2001.

[21] Y. Chen and E.K. Wong. Augmented image histogram for image and video similarity search. In *Proc. of SPIE - Storage and Retrieval for Image and Video Databases VII*, pages 523–429, 1999.

[22] V. Chitkara. Color-based image retrieval using compact binary signatures. Master's thesis, Dept. of Computing Science, University of Alberta, 2001.

[23] P. Ciaccia, M. Partella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *Proc. of VLDB Intl. Conf.*, pages 426–435, 1997.

[24] W.S. Cooper. Expected search length: A single measure of retrieval effectiveness based on the weak ordering action of retrieval systems. *American Documentation*, 19(1):30–41, 1968.

[25] T.H. Cormem, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms.* McGraw-Hill, 1990.

[26] S. Dao, Q. Yang, and A. Vellaikal. Mb$^+$-tree: An index structure for content-based retrieval. In *Multimedia Database Systems: Design and Implementation Strategies.* Kluwer Academic Publishers, 1996.

[27] C.J. Date. *An Introduction to Database Systems.* Addison Wesley, 1995.

[28] Y. Deng and B.S. Manjunath. An efficient low-dimensional color indexing scheme for region-based image retrieval. In *Proc. of IEEE ICASSP Intl. Conf.*, pages 3017–3020, 1999.

[29] A. Dimai. Spatial encoding using differences of global features. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases IV*, pages 352–360, 1997.

[30] M.S. Drew, J. Wei, and Z.N. Li. Illumination-invariant color object recognition via compressed chromaticity histograms of color-channel-normalized images. In *Proc. of ICCCV Intl. Conf.*, pages 533–540, 1998.

[31] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* Wiley-Interscience, 1973.

[32] C. Faloutsos, W. Equitz, M. Flickner, et al. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3(3/4):231–262, 1994.

[33] B. Funt and G. Finlayson. Color constant color indexing. *IEEE PAMI*, 17(5):522–529, 1995.

[34] G. Furnas, T.K. Landauer, L.M. Gomez, et al. The vocabulary problem in huma-system communications. *Communications of the ACM*, 30:964–971, 1987.

[35] V. Gaede and O. Guenther. Multidimensional access methods. *ACM Computing Surveys*, 30(2):123–169, 1998.

[36] A.V. Goldberg. An efficient implementation of a scaling minimum-cost flow algorithm. *Journal of Algorithms*, 22:01–29, 1997.

[37] R.C. Gonzalez and R.E. Woods. *Digital Image Processing.* Addison-Wesley, 1992.

[38] L.J. Guibas, B. Rogoff, and C. Tomasi. Fixed-window image descriptors for image retrieval. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases III*, pages 352–362, 1995.

[39] N.J. Gunther and G. Beretta. A benchmark for image retrieval using distributed systems over the internet: BIRDS-I. In *Proc. of SPIE - Internet Imaging II*, pages 252–267, 2001.

[40] O. Guttman. R-tree: A dynamic index structure for spatial searching. In *Proc. of ACM SIGMOD Intl. Conf.*, pages 47–57, 1984.

[41] H.V. Jagadish. A retrieval technique for similar shapes. In *Proc. of ACM SIGMOD Intl. Conf.*, pages 208–217, 1991.

[42] K.S. Jones and P. Willet. *Readings in Information Retrieval.* Morgan Kaufmann, 1997.

[43] I. Kalantari and G. McDonald. A data structure and an algorithm for the nearest point problem. *IEEE TSE*, 9(5):631–634, 1983.

[44] N. Katayama and S. Satoh. The SR-tree: An index structure for high-dimensional nearest neighbor queries. In *Proc. of ACM SIGMOD Intl. Conf.*, pages 369–380, 1997.

[45] L. Kaufman and P.J. Rousseuw. *Finding Groups in Data - An Introduction to Cluster Analysis.* Wiley-Interscience, 1990.

[46] E.M. Keen. Evaluation parameters. In *The SMART Retrieval System — Experiments in Automatic Document Processing*, chapter 5. Prentice-Hall, 1971.

[47] T. Kirste. SPACEPICTURE – an interactive hypermedia satellite image archival system. *Computers and Graphics*, 17(3):251–260, 1993.

[48] R.R. Korfhage. *Information Storage and Retrieval.* John Wiley and Sons, 1997.

[49] F. Korn, N. Sidiropoulos, C. Faloutsos, et al. Fast nearest-neighbor search in medical image databases. In *Proc. of VLDB Intl. Conf.*, pages 215–226, 1996.

[50] H.F. Korth and A. Silberschatz. *Database System Concepts.* McGraw-Hill, 1991.

[51] S. Krishnamachari. Hierarchical clustering for fast image retrieval. In *Proc. of SPIE - Storage and Retrieval for Image and Video Databases VII*, pages 427–435, 1999.

[52] H.W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quart.*, 2:83–97, 1955.

[53] S.-Y. Lee and F.-J. Hsu. 2Dc-string – a new spatial knowledge representation for image database systems. *Pattern Recognition*, 23(10):1077–1087, 1990.

[54] C.H.C. Leung and H.H.S. Ip. Benchmarking for content-based visual information search. In *Proc.of VISUAL Intl. Conf.*, pages 442–456, 2000.

[55] K.-S. Leung and R. Ng. Multiresolution subimage similarity matching for large image databases. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases VI*, pages 259–270, 1998.

[56] C. Li, E. Chang, H. Garcia-Molina, et al. Clindex: Clustering for similarity queries in high-dimensional spaces. Technical Report 1999-37, Database Group, Stanford University, 1999.

[57] C.-S. Li and J. Turek. Content-based indexing of earth observing satellite image database with fuzzy attributes. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases IV*, pages 438–449, 1996.

[58] J. Li, J.Z. Wang, and G. Wiederhold. IRM: Integrated region matching for image retrieval. In *Proc.of ACM Multimedia Intl. Conf.*, pages 147–156, 2000.

[59] Z. Li, O.R. Zaane, and B. Yan. C-bird: Content-based image retrieval from digital libraries using illumination invariance and recognition kernel. Technical Report 1998-03, Simon Fraser University, Canada, Feb 1998.

[60] K.I. Lin, H.V. Jagadish, and C. Faloutsos. The TV-tree: An index structure for high-dimensional data. *VLDB Journal*, 3(4):517–549, 1994.

[61] G. Lu. *Multimedia Database Management Systems.* Artech House, 1999.

[62] W.Y. Ma, Y. Deng, and B.S. Manjunath. Tools for texture/color based search of images. In *Proc. of SPIE - Human Vision and Electronic Imaging II*, pages 395–406, 1997.

[63] W.Y. Ma and B.S. Manjunath. Texture-based pattern retrieval from image databases. *Multimedia Tools and Applications*, 1(3):35–51, 1996.

[64] J. Malki, N. Boujemaa, C. Nastar, et al. Region queries without segmentation for image retrieval by content. In *Proc. of VISUAL Intl. Conf.*, pages 115–122, 1999.

[65] M.T. Maybury. *Intelligent Multimedia Information Retrieval.* AAAI Press / The MIT Press, 1997.

[66] H. Muller, W. Muller, D. McG. Squire, et al. Performance evaluation in content-based image retrieval: Overview and proposals. *Pattern Recognition Letters*, 22:593–601, 2001.

[67] S.H. Myaeng and R.R. Korfhage. Integration of user profiles: Models and experiments in information retrieval. *Information Processing and Management*, 26(6):719–738, 1990.

[68] M.A. Nascimento and V. Chitkara. Color-based image retrieval using binary signatures. In *Proc. of ACM SAC Intl. Conf.*, pages 687–692, 2002.

[69] M.A. Nascimento, E. Tousidou, V. Chitkara, et al. Image indexing and retrieval using signature trees. *Data and Knowledge Engineering Journal*, 2002. To appear.

[70] A. Natsev, R. Rastogi, and K. Shim. WALRUS: A similarity retrieval algorithm for image databases. In *Proc. of ACM SIGMOD Intl. Conf.*, pages 395–406, 1999.

[71] W. Niblack, X. Zhu, J.L. Hafner, et al. Updates to the QBIC system. In *Proc. of SPIE - Storage and Retrieval for Image and Video Databases VI*, pages 150–161, 1998.

[72] J. Nievergelt, H. Hinterberger, and K.C. Sevcik. The grid file: An adaptive, symmetric multikey file structure. *ACM Transactions on Database Systems*, 9(1):38–71, 1984.

[73] B.C. Ooi, K. Tan, T.S. Chua, et al. Fast image retrieval using color-spatial information. *VLDB Journal*, 7(2):115–128, 1998.

[74] S. C. Orphanoudakis, C. Chronaki, and S. Kostomanolakis. I2C: A system for the indexing, storage and retrieval of medical images by content. Technical Report 113, Foundation for Research and Technology, Jan 1994.

[75] G. Pass, R. Zabih, and J. Miller. Comparing images using color coherence vectors. In *Proc. of ACM Multimedia Intl. Conf.*, pages 65–73, 1996.

[76] E.J. Pauwels and G. Frederix. Finding regions of interest for content-extraction. In *Proc. of SPIE - Storage and Retrieval for Image and Video Databases VII*, pages 501–510, 1999.

[77] A. Pentland, R.W. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. *International Journal of Computer Vision*, 18(3):233–254, 1996.

[78] E.G.M. Petrakis and C. Faloutsos. Similarity searching in large image databases. *IEEE TKDE*, 9(3):435–447, 1997.

[79] S.M. Pollock. Measures for the comparison of information retrieval systems. *American Documentation*, 19(4):387–397, 1968.

[80] W.K. Pratt. *Fast Digital Image Processing.* John Wiley and Sons, 1991.

[81] V. Rijsbergen. *Information Retrieval.* Butterworths, 1979.

[82] G. Salton and M.J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

[83] S. Santini and R. Jain. Similarity measures. *IEEE PAMI*, 21(9):871–883, 1999.

[84] R.F. Santos, A. Traina, C. Traina, et al. Similarity search without tears: The OMNI-family of all-purpose access methods. In *Proc. of ICDE Intl. Conf.*, pages 623–630, 2001.

[85] E. Di Sciascio, G. Mingolla, and M. Mongiello. Content-based image retrieval over the web using query by sketch and relevance feedback. In *Proc. of VISUAL Intl. Conf.*, pages 123–130, 1999.

[86] S. Sclaroff. Distance to deformable prototypes: Encoding shape categories for efficient search. In *Image Databases and Multi-Media Search.* World Scientific, 1997.

[87] N. Sebe, M.S. Lew, and D.P. Huijsmans. Multi-scale sub-image search. In *Proc. of ACM Multimedia Intl. Conf.*, pages 79–82, 1999.

[88] I.K. Sethi, I. Coman, B. Day, et al. Color-wise: A system for image similarity retrieval using color. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases IV*, pages 140–149, 1998.

[89] W. Shaw. On the foundation of evaluation. *Journal of the American Society for Information Science*, 37(5):346–348, 1986.

[90] A. Sheth and W. Klas. *Multimedia Data Management.* McGraw-Hill, 1998.

[91] E. Shusterman and M. Feder. Image compression via improved quadtree decomposition algorithms. *IEEE TIP*, 3(2):207–215, 1994.

[92] A.W.M. Smeulders, M. Worring, S. Santini, et al. Content-based image retrieval at the end of the early years. *IEEE PAMI*, 22(12):1349–1380, 2000.

[93] J.R. Smith. *Integrated Spatial and Feature Image Systems: Retrieval, Compression and Analysis.* PhD thesis, Columbia University, Feb 1997.

[94] J.R. Smith, V. Castelli, and C.-S. Li. Adaptive storage and retrieval for large compressed images. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases VII*, pages 467–478, 1999.

[95] J.R. Smith and S.-F. Chang. Automated binary texture feature sets for image retrieval. In *Proc. of the Intl. Conf. Acoustic, Speech and Signal Processing*, 1996.

[96] J.R. Smith and S.-F. Chang. Tools and techniques for color image retrieval. In *Proc. of SPIE - Storage and Retrieval for Image and Video Databases IV*, pages 426–437, 1996.

[97] J.R. Smith and S.-F. Chang. Visually searching the web for content. *IEEE Multi-Media*, 4(3):12–20, 1997.

[98] T.R. Smith. A digital library for geographically referenced materials. *IEEE Computer*, 29(5):54–60, 1996.

[99] R.O. Stehling, M.A. Nascimento, and A.X. Falcão. On 'shapes' of colors for content-based image retrieval. In *Proc. of ACM MIR Intl. Workshop*, pages 171–174, 2000.

[100] R.O. Stehling, M.A. Nascimento, and A.X. Falcão. An adaptive and efficient clustering-based approach for content based retrieval in image databases. In *Proc. of IDEAS Intl. Symposium*, pages 356–365, 2001.

[101] R.O. Stehling, M.A. Nascimento, and A.X. Falcão. A compact and efficient image retrieval approach based on border/interior pixel classification. In *Proc. of the ACM CIKM Intl. Conf.*, 2002.

[102] R.O. Stehling, M.A. Nascimento, and A.X. Falcão. MiCRoM: A metric distance to compare segmented images. In *Proc. of VISUAL Intl. Conf.*, pages 12–23, 2002.

[103] R.O. Stehling, M.A. Nascimento, and A.X. Falcão. Techniques for color-based image retrieval. In *Multimedia Mining - a Highway to Intelligent Multimedia Document*, chapter 4. Kluwer Academic Publishers, 2002.

[104] R.O. Stehling, M.A. Nascimento, and A.X. Falcão. Cell histograms versus color histograms for image representation and retrieval. *Knowledge and Information Systems Intl. Journal*, 2003. To appear.

[105] M. Stonebraker. *Readings in Database Systems*. Morgan Kaufmann, 1994.

[106] M. Stricker and M. Orengo. Similarity of color images. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases III*, pages 381–392, 1995.

[107] M.J. Swain and D.H. Ballard. Color indexing. *Intl. Journal of Computer Vision*, 7(1):11–32, 1991.

[108] C. Traina, A. Traina, C. Faloutsos, and B. Seeger. Fast indexing and visualization of metric data sets using slim-trees. *IEEE TKDE*, 14(2):244–260, 2002.

[109] A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977.

[110] A. Tversky and I. Gati. Similarity, separability, and the triangle inequality. *Psychological Review*, 89(1):123–154, 1982.

[111] J. Uhlmann. Satisfying general proximity/similarity queries with metric trees. *Information Processing Letters*, 40:175–179, 1991.

[112] J.Z. Wang, G. Wiederhold, O. Firschein, et al. Content-based image indexing and searching using daubechies' wavelets. *IJODL*, 1(4):311–328, 1998.

[113] S. Wang. A robust CBIR approach using local color histograms. Master's thesis, Dept. of Computing Science, University of Alberta, 2001.

[114] D.A. White and R. Jain. Similarity indexing with the SS-tree. In *Proc. of ICDE Intl. Conf.*, pages 516–523, 1996.

[115] I.H. Witten, A. Moffat, and T.C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann, 1999.

[116] J.K. Wu and A.D. Narasimhalu. Identifying faces using mutiple retrievals. *IEEE Multimedia*, 1(2):27–38, 1994.

[117] M.-H. Yang and N. Ahuja. Gaussian mixture model for human skin color and its applicattions in image and video databases. In *Proc. of SPIE - Storage and Retrieval for Image and Video Databases VII*, pages 458–466, 1999.

[118] R.B. Yates and B.R. Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

[119] P. Yianilos. Data structures and algorithms for nearest neighbor search in general metric spaces. In *Proc. of ACM Symposium on Discrete Algorithms*, pages 311–321, 1993.

[120] A. Yoshitaka and T. Ichikawa. A survey on content-based retrieval for multimedia databases. *IEEE TKDE*, 11(1):81–93, 1999.

[121] A. Zhang, B. Cheng, R. Acharya, et al. Comparison of wavelet transforms and fractal coding in texture-based image retrieval. In *Proc. of SPIE - Visual Data Exploration and Analysis III*, pages 116–125, 1996.

[122] Y.J. Zhang, Z.W. Liu, and Y. He. Comparison and improvement of color-based image retrieval techniques. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases VI*, pages 371–382, 1998.

Anexos

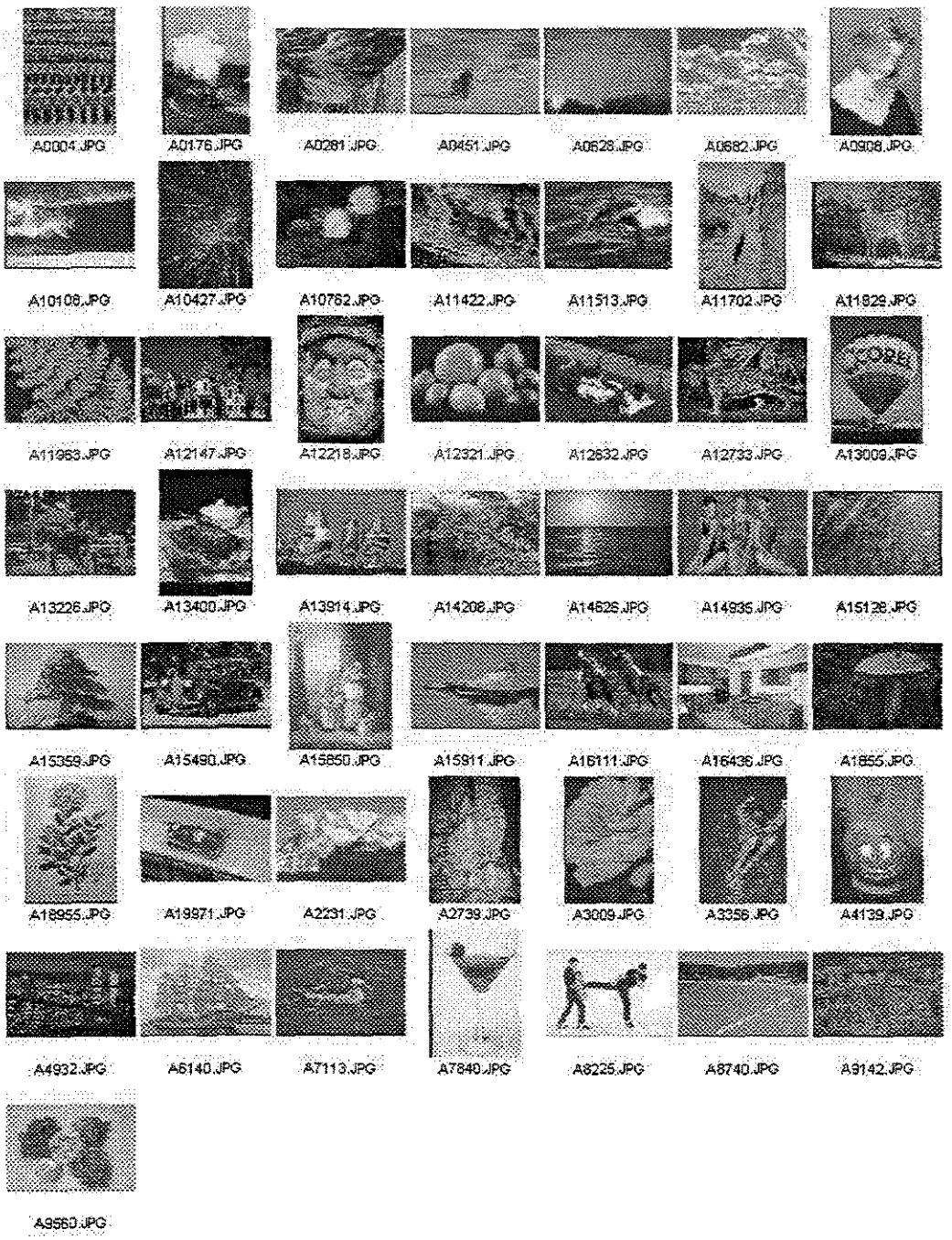

Figure 7.1: Imagens consulta utilizadas em nossos experimentos

Figure 7.2: Primeiro exemplo de RRSet

Figure 7.3: Segundo exemplo de RRSet

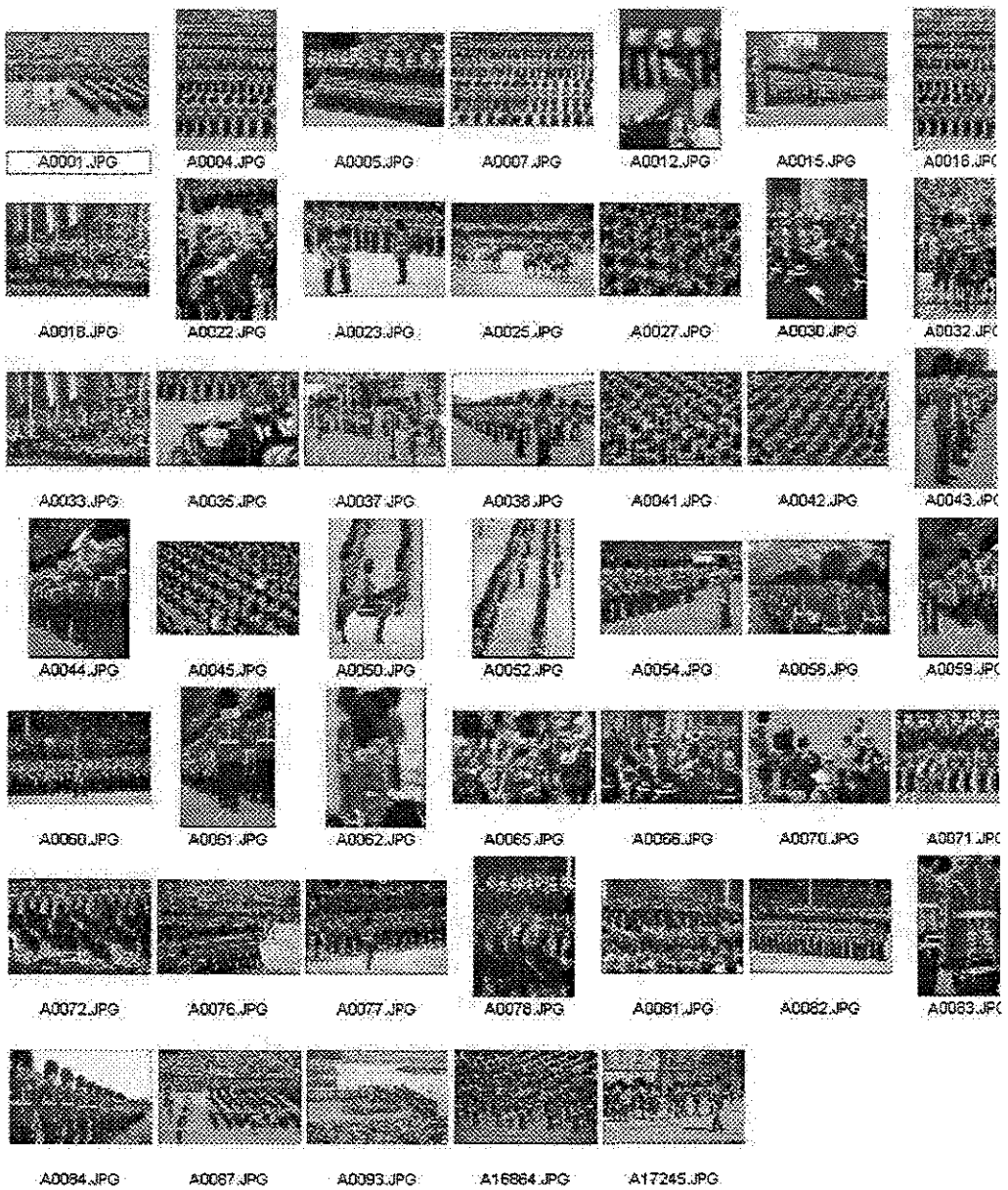

Figure 7.4: Terceiro exemplo de RRSet

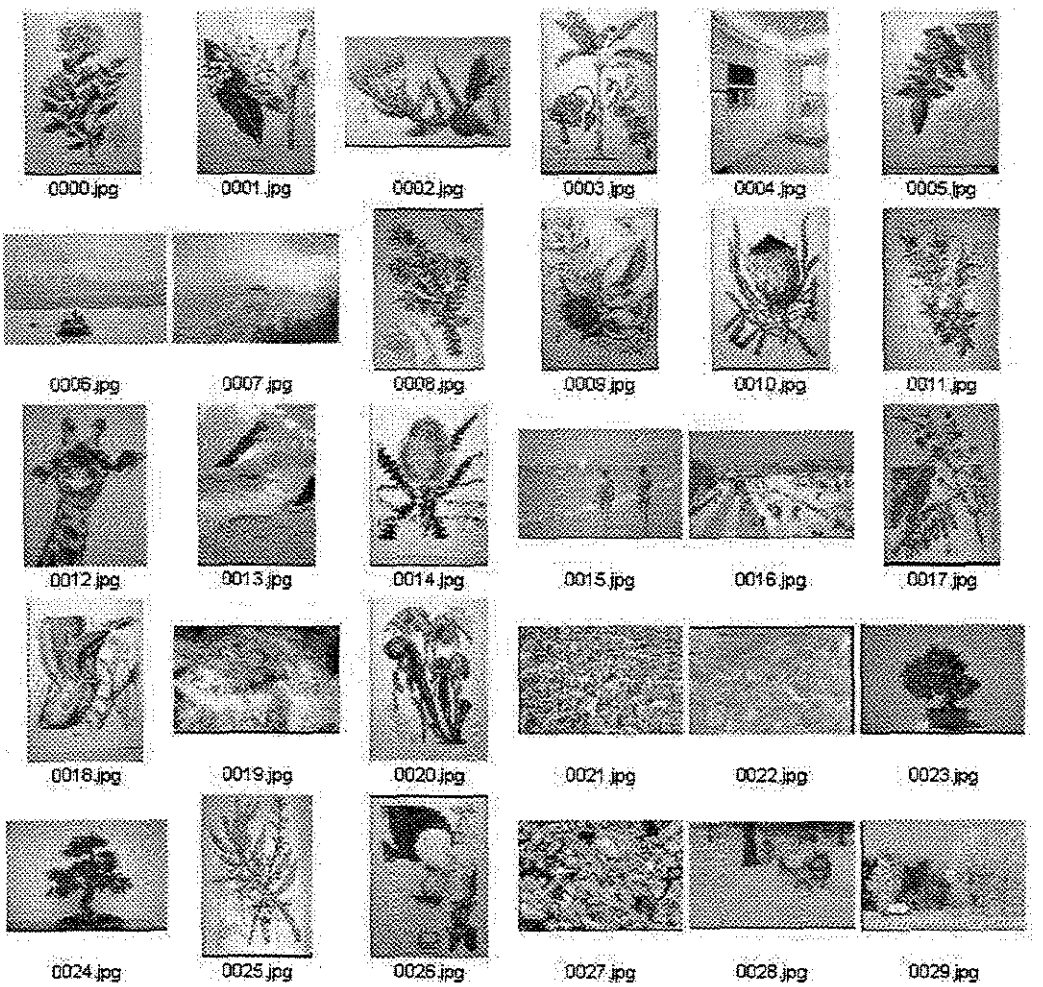

Figure 7.5: Exemplo do resultado de uma busca pelos 30 vizinhos mais próximos de uma imagem utilizando o *CBC*

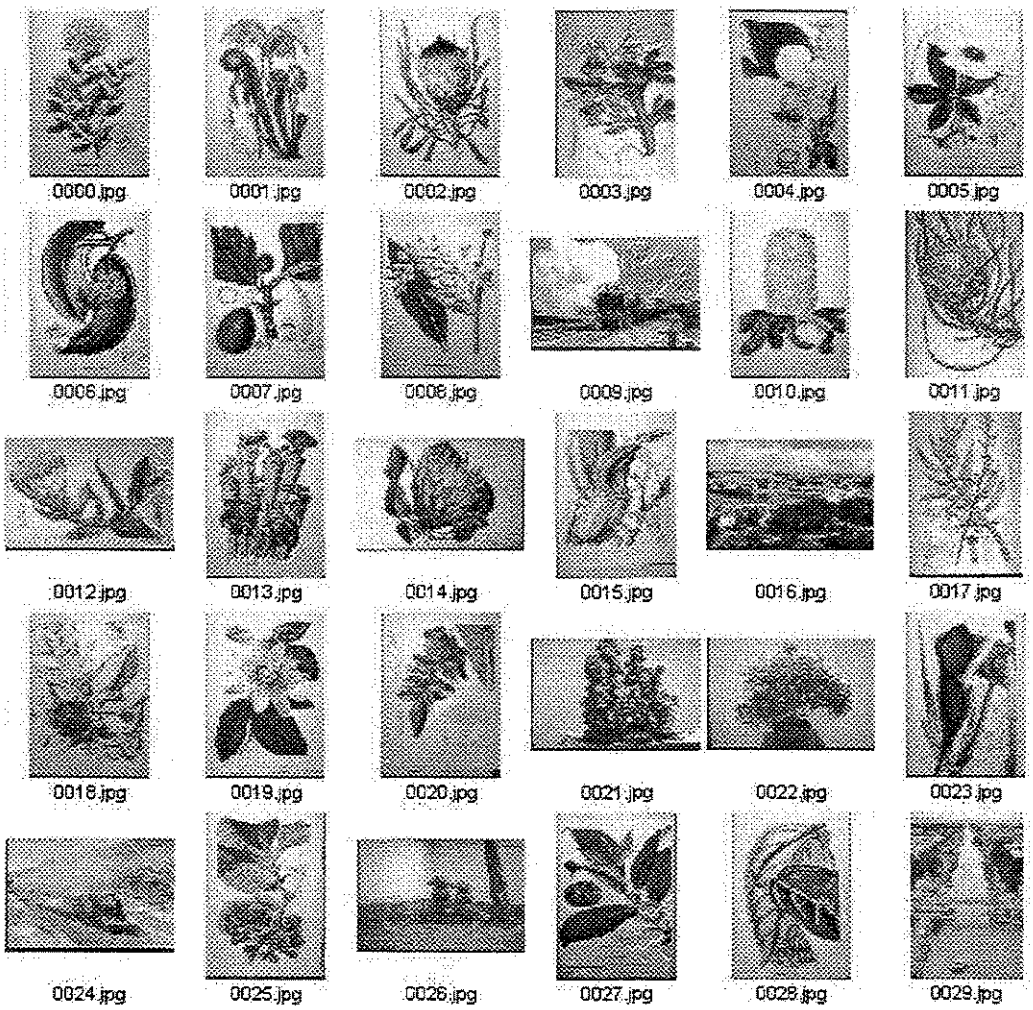

Figure 7.6: Exemplo do resultado de uma busca pelos 30 vizinhos mais próximos de uma imagem utilizando o *BIC*

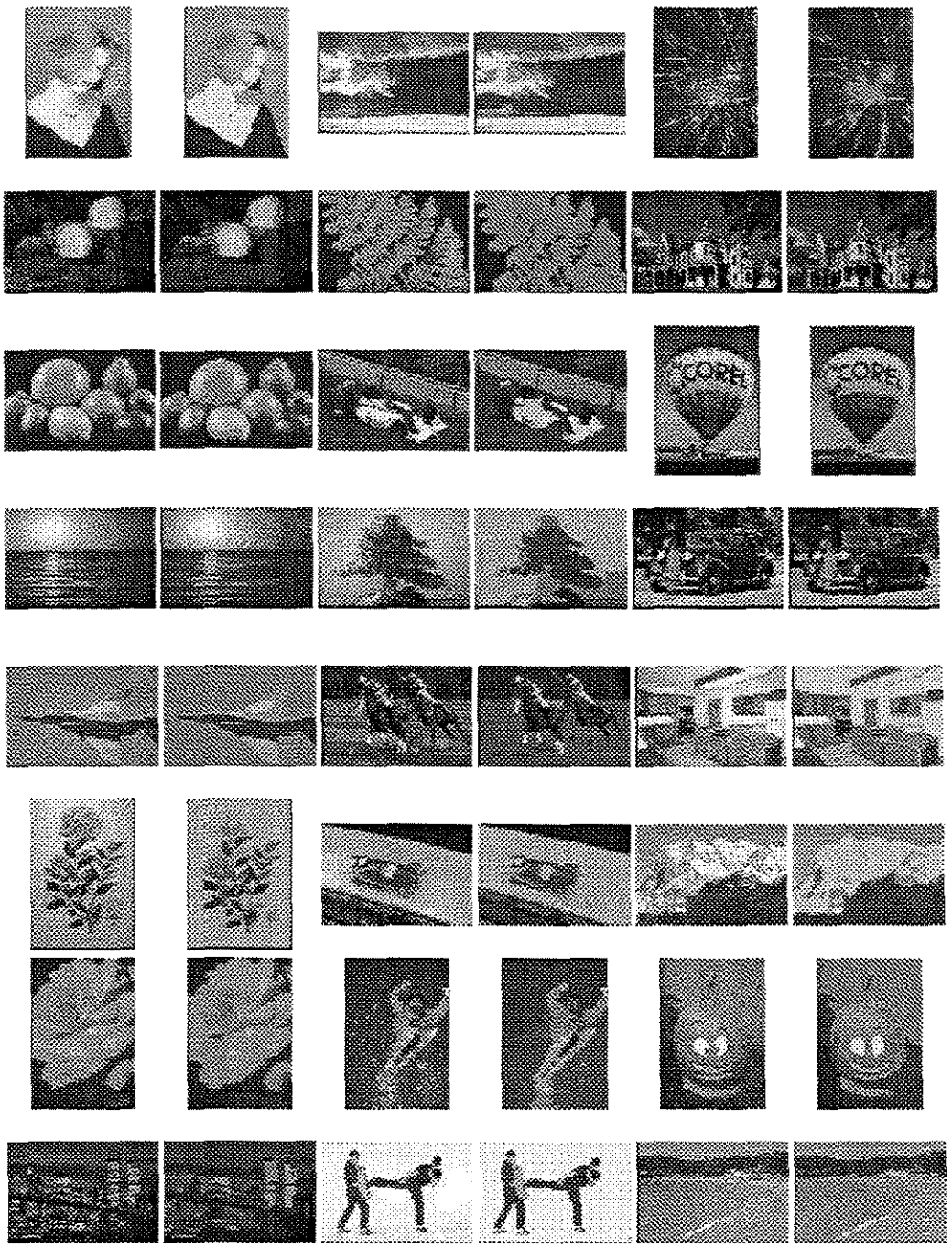

Figure 7.7: Exemplos de imagens automaticamente segmentadas com o algoritmo *CBC*

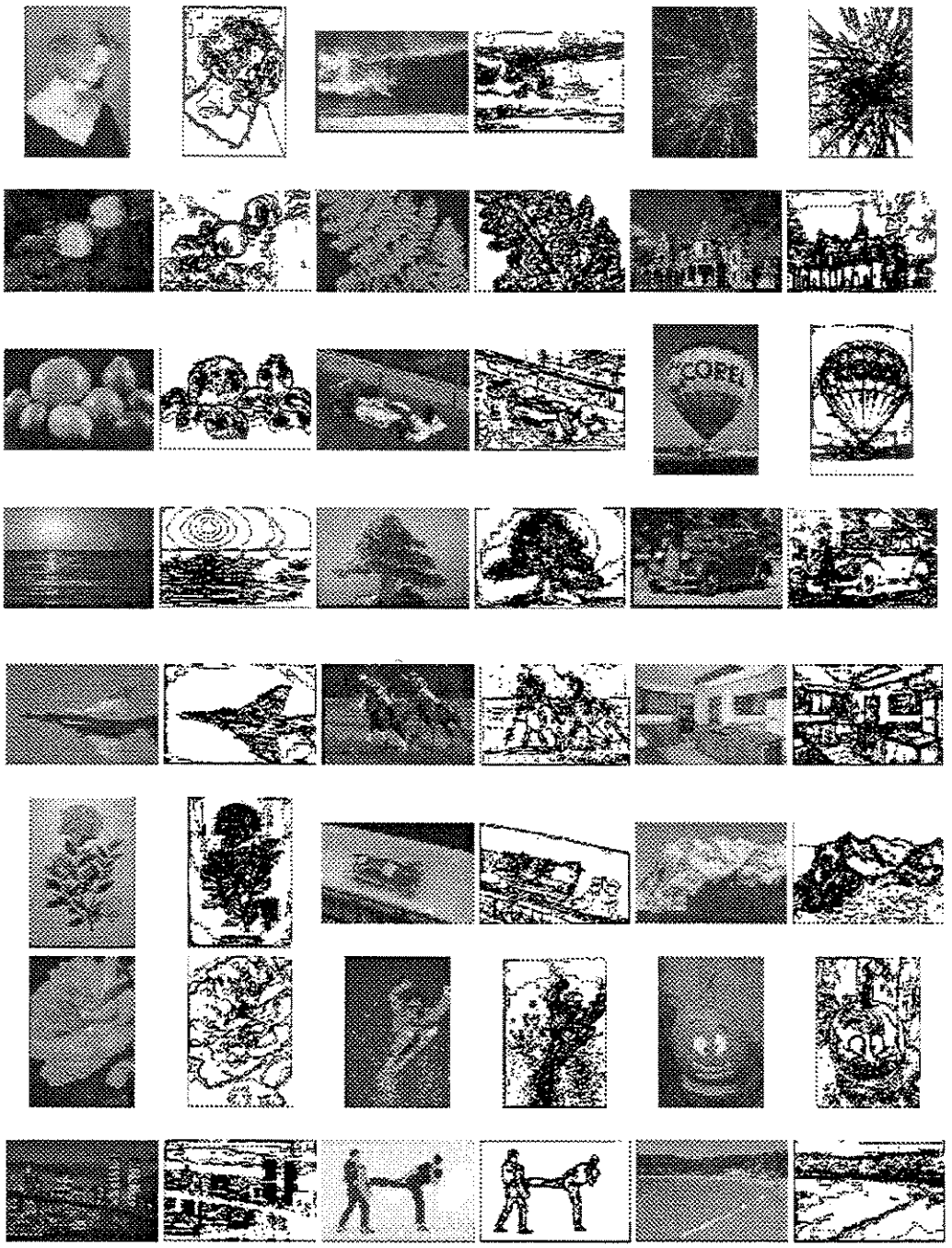

Figure 7.8: Exemplos da classificação binária dos pixels de uma imagem em borda (preto) e interior (branco)