Pedro Filipe M. V. C. Martins

Licenciado em Engenharia Informática

Pesquisa de Clip Arts Combinando Imagens Raster e Vectoriais

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática

Orientadores : Nuno Manuel Robalo Correia, Prof. Catedrático, Universidade Nova de Lisboa
Rui Manuel Feliciano Jesus, Prof. Adjunto, Instituto Superior de Engenharia de Lisboa

Júri:

Presidente:	Prof. Doutor José Alberto Cardoso e Cunha
Arguente:	Prof. Doutor João Miguel Duarte Ascenso
Vogal:	Prof. Doutor Nuno Manuel Robalo Correia

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Dezembro, 2012

Pesquisa de Clip Arts Combinando Imagens Raster e Vectoriais

Para a minha mãe

Agradecimentos

Em primeiro lugar, gostaria de agradecer aos meus orientadores, ao Professor Nuno Correia e ao Professor Rui, pelo apoio, disponibilidade e a paciência demonstrados ao longo de toda a dissertação. Queria agradecer também aos colegas do Departamento de Informática da Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa(DI - FCT/UNL) que colaboraram comigo nesta dissertação, ao Professor Manuel J. Fonseca e ao Gabriel Barata do Instituto de Engenharia de Sistemas e Computadores, Investigação e Desenvolvimento em Lisboa (INESC ID - IST) e ao projecto CRUSH (Clip art Retrieval using Sketches), no qual a dissertação se insere, com referência PTDC/EIA-EIA/108077/2008, financiado pela Fundação para a Ciência e Tecnologia.

Quero agradecer também a toda a minha família, em especial à minha mãe, por me ter acompanhado ao longo de toda a minha formação académica e pelo carinho e preocupação de mãe que fazem sentir saudades de casa. E também a todos os meus amigos, que estiveram ao meu lado durante a minha vida e percurso académico, Diogo Cabrita, Gonçalo Consiglieri, Daniel Franco, José Esteves, João Santos, Luís Alho, Rui Pimentel, Igor Valente, Joana Anacleto, Carolina Salgueiro, Joana Nobre, Patrícia Lino, João Vitorino, João Moreno, Pedro Rodrigues, ...

Muito Obrigado.

Resumo

Existe actualmente um crescente desenvolvimento de sistemas de armazenamento e pesquisa de imagens. Uma aproximação adoptada nesses sistemas é a recuperação de imagens baseada em conteúdo (CBIR, Content-Based Image Retrieval). No âmbito destas aplicações existem utilizadores que pretendem utilizar imagens clip art para os seus trabalhos e apresentações.

Existem muitas imagens clip art espalhadas por diversas bases de dados em sítios na Internet ou em colecções vendidas em dispositivos ópticos. A pesquisa de imagens nestas bases de dados leva os utilizadores a percorrem várias listas de imagens manualmente ou por métodos de pesquisa por texto, muitas vezes ineficientes. Essas bases de dados de clip arts são representadas por imagens vectoriais e imagens *raster*.

Existem várias tecnologias de pesquisa e recuperação de ambos os tipos de imagens clip art, *raster* e vectoriais, contudo, a investigação tem sido realizada em separado sem retirar partido das duas áreas de investigação em conjunto, no problema de recuperar e explorar colecções de clip arts. O objectivo deste trabalho é implementar um motor de busca para encontrar clip arts em base de dados compostas por imagens vectoriais e imagens *raster*. O trabalho envolve um conversor de imagens *raster* em vectoriais, a extracção de características das imagens *raster* e vectoriais e a avaliação do sistema de recuperação de clip arts.

Palavras-chave: Características *Raster*, Características vectoriais, Simplificação vectorial, Clip art CBIR, Extracção de características

Abstract

There is a growing development of search and storage systems for images today. One approach adopted in these systems is content-based image retrieval (CBIR). In the scope of these applications there are users who wish to use clip art images to include in their presentations and documents.

There are many of these images in databases on Internet sites or sold in optical media. Search and retrieval of images in these databases make the user go though a number of large lists manually or use searching methods based on text, which are somewhat inefficient. Those clip art databases are made of vector images or raster images.

There are technologies for searching and retrieving for both clip art image types, raster and vectorial, however research has been done separately without taking advantages of both research fields to the problem of recovery and exploration of clip art collections. The goal of this thesis is to implement a search engine to retrieve clip art images in databases which include vector and raster images. The solution requires the implementation of a raster to vector image converter, the extraction of features from raster and vector images and the evaluation of the clip art retrieval system.

Keywords: Raster features, Vector features, Vector image simplification, Clip art CBIR, Feature extraction

Conteúdo

Lista de Figuras

Lista de Tabelas

1

Introdução

Clip arts são colecções de desenhos digitais sintéticos que podem ser importadas e utilizadas em diversos tipos de documentos ou programas. Estas imagens pode ser encontradas em formato *raster* e vectorial. Algumas colecções podem ter dezenas de imagens enquanto outras podem chegar a ter milhares. A existência de grande número de colecções de imagens, distribuídas por diversas bases de dados, na Internet ou adquiridas em colecções em dispositivos ópticos ou *pen drives*, em diferentes formatos e com métodos de pesquisa e indexação diferentes, faz com que encontrar uma determinada imagem não seja tarefa fácil.

Em programas de produção de documentos e em *websites* estas colecções são agrupadas em diversas categorias, como pessoas, objectos ou árvores, que identificam os elementos contidos nas imagens. Também costumam ser associadas palavras-chave a cada imagem de modo a facilitar as pesquisas dentro das categorias e para descrever diversos conceitos presentes na imagem. O utilizador pode então percorrer a colecção para encontrar a imagem que pretende. No entanto, a procura manual é morosa e vai contra o objectivo de acelerar a produção dos conteúdos. A utilização de palavras-chave para reduzir as pesquisas também se torna inviável, porque essa informação tem de ser produzida manualmente e o utilizador precisa de conhecer que meta-informação foi utilizada para caracterizar as imagens. A utilização de descrição textual não é adequada para descrever a estrutura, forma e topologia da imagem e a sua classificação também pode ser subjectiva [38]. Os métodos de pesquisa têm de acompanhar este crescente número de imagens e não obrigar o utilizador a percorrer diferentes tipos

de colecções estruturadas de modos diferentes ou a conhecer as classes de meta-informação das imagens na colecção.

De modo a remover a subjectividade nas pesquisas e classificação de imagens deve ser usado um sistema que inclua as características das mesmas, num processo que as extraia automaticamente e não recaia sobre informação produzida manualmente. O utilizador deve poder expressar a sua pesquisa de modo mais natural, por exemplo por esboços do que pretende ou por imagens semelhantes. Dentro deste paradigma de pesquisa de imagens são incluídos os sistemas **CBIR**, *content-based image retireval* [12], que utilizam as características presentes nas imagens de forma a efectuar pesquisas numa colecção de imagens. Dentro destes sistemas de pesquisa podemos encontrar dois grupos, sistemas que utilizam características extraídas imagens *raster* e características extraídas de imagens vectoriais. Neste documento é descrito um sistema CBIR, desenvolvido para recuperação de clip arts que permite a utilização de características extraídas de ambos os grupos de imagens e também combina estas características, utilizando assim características extraídas tanto de imagens *raster* como de imagens vectoriais.

1.1 Descrição do problema, objectivos e contexto

Actualmente existem várias tecnologias de CBIR para imagens digitais que se aplicam a diferentes meios. Existem tecnologias especializadas na recuperação de imagens fotográficas, imagens CAD ou detecção de rostos. Estes sistemas utilizam algoritmos de estado de arte que permitem ao sistema interpretar, anotar e indexar as imagens. Porém, algumas destas tecnologias não se adequam a desenhos do tipo clip art, e o seu desempenho pode não ser reproduzido em colecções deste tipo de imagens.

Embora actualmente também existam sistemas que permitem pesquisar as colecções de clip arts que existem tanto em formato *raster* como em formato vectorial, estes necessitam que o utilizador conheça um conjunto de palavras-chave ou um nome que descreva a imagem. Contudo, a investigação na recuperação de desenhos clip art continua um pouco dividida entre os dois tipos de armazenamento de imagem. Estas soluções procuram resolver problemas nestas áreas separadamente, utilizando as particularidades das imagens definidas em formato *raster* ou vectoriais, quando poderiam utilizar as vantagens de ambos.

No trabalho desenvolvido em [22] é apresentado um sistema de extracção de características que inclui descritores de formas geométricas presentes na imagem

e as relações espaciais entre elas através de um grafo de topologia. Estas características são extraídas de imagens no formato vectorial.

Nesta dissertação apresentamos uma alternativa à pesquisa por palavras-chave (utilizando informação anotada manualmente), utilizando a pesquisa por conteúdo da imagem. O nosso sistema possibilita ainda a combinação de características extraídas, tanto de imagens *raster*, utilizando diversos descritores de cor e textura, como de imagens vectoriais, utilizando os descritores utilizados em [22]. Os objectivos desta dissertação são:

- Implementar um sistema de simplificação de imagens clip art em formato vectorial, utilizando versões *raster* das mesmas;
- Implementar um sistema que incorpora técnicas de extracção de características de imagens clip art *raster*, que representam as suas cores, textura e pontos de interesse. O sistema incorpora também técnicas de extracção de características de imagens clip art vectoriais, que representam a geometria dos elementos que as compõem e a relação entre eles;
- Implementar um sistema de recuperação de imagens clip art utilizando técnicas de pesquisa de imagens *raster* combinadas com técnicas de pesquisa de imagens vectoriais.

Esta dissertação insere-se no projecto denominado CRUSH (Clip art Retrieval using Sketches), que está a ser desenvolvido pelo **INESC ID** (Instituto de Engenharia de Sistemas e Computadores, Investigação e Desenvolvimento em Lisboa), do Instituto Superior Técnico, da Universidade Técnica de Lisboa e pela Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa (**FCT-UNL**). Este projecto conta com o apoio financeiro da Fundação para a Ciência e Tecnologia. O projecto CRUSH tem por objectivo criar uma plataforma de pesquisa de clip arts que tire partido de imagens vectoriais e *raster* numa colecção de grande dimensão.

1.2 Solução apresentada

A solução proposta nesta dissertação é composta por um sistema de recuperação de clip arts baseado nas suas características (CBIR - *Content-Based Image Retrieval*). O sistema pode ser dividido em dois componentes (ver figura 1.1): um módulo que faz a extracção de características e um módulo que permite recuperar clip arts

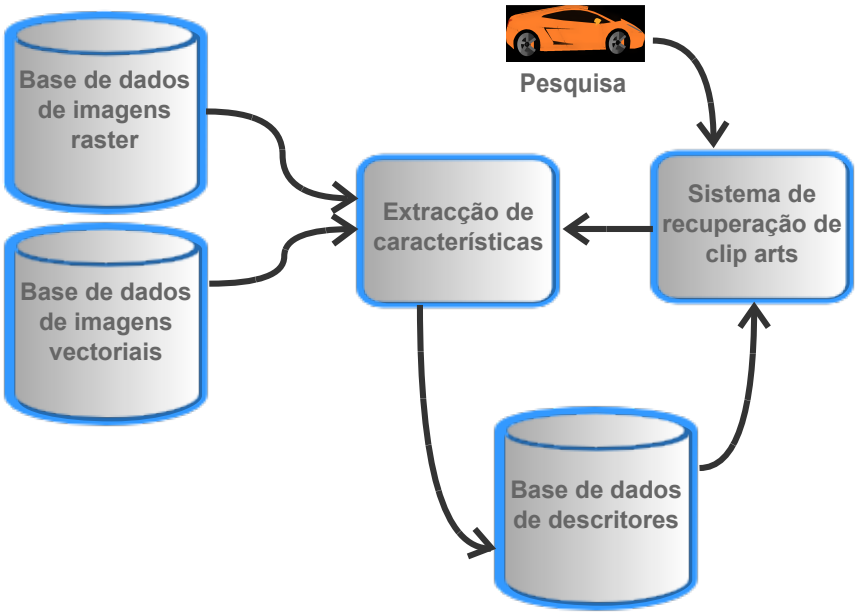


Figura 1.1: Diagrama dos componentes da solução implementada.

em bases de dados vectoriais e *raster*. Este módulo normaliza os vários descritores presentes na base de dados e efectua as interrogações na colecção.

O módulo de extracção permite a extracção de diversas características e o armazenamento de diversos vectores de descritores dessas mesmas características. Nesse módulo foram implementados diversos métodos de extracção de características de imagens *raster*. Para as imagens vectoriais foi implementado um método de vectorização de imagens *raster* de modo a produzir uma simplificação das imagens vectoriais originais. As características das imagens vectoriais são extraídas usando as técnicas desenvolvidas em [22].

No segundo módulo é implementado o sistema de recuperação de clip arts. Para tal, foram implementadas as técnicas de normalização dos descritores dos clip arts da base de dados e das interrogações. Inclui também o sistema recuperação de imagens na colecção utilizando *query-by-example*, e a avaliação dos resultados obtidos. Para a recuperação de imagens foram implementados diversos algoritmos para cada uma das características e para cada combinação de características.

1.2.1 Tecnologias utilizadas

O desenvolvimento da solução proposta utilizou a plataforma OpenCV (Open Source Computer Vision)[1], uma biblioteca *open-source* para aplicação de algoritmos de estado de arte de visão por computador. Esta biblioteca tem interfaces

em C, C++ e Python e possuí variados algoritmos optimizados já implementados que foram utilizados no decorrer do desenvolvimento da solução proposta. Foram também utilizados *scripts* Matlab para extracção de alguns descritores e redução de cores, bem como o sistema EDISON, implementado em C++ e o sistema para extracção de descritores de imagens vectoriais, implementado em JAVA.

1.3 Contribuições

As principais contribuições centram-se na área da combinação de técnicas vectoriais e *raster*. Como principais contributos desta dissertação podemos considerar os seguintes:

- **Extracção de características raster** - Implementação de métodos de extracção de descritores de imagens clip art *raster* e desenvolvimento de um sistema que incorpora os métodos implementados e alguns dos métodos desenvolvidos em [26] e no âmbito do projecto Videoflow [8].

- **Recuperação de Clip Arts** - Implementação de um sistema de recuperação de clip arts que utiliza características extraídas de imagens *raster* e de imagens vectoriais.

- **Simplificação de imagens vectoriais** - Implementação de um algoritmo de simplificação de imagens vectoriais baseado na identificação de polígonos presentes em manchas de cores extraídas de versões *raster* das imagens vectoriais e a sua posterior vectorização.

- **Combinação de características *raster* e vectoriais** - Implementação de algoritmos que permitem diversas combinações de descritores. Combinação de diversos descritores extraídos de imagens *raster* que incluem informação sobre cor e textura e descritores extraídos de imagens vectoriais que incluem informação sobre a topologia e geometria.

- **Estudo de descritores** - Considerando o ponto anterior foi feito um estudo de quais os descritores que produzem melhores resultados isoladamente e em combinação com a informação vectorial.

[1] http://opencv.willowgarage.com/

1.4 Organização da dissertação

Este documento está dividido em seis capítulos. O capítulo 1 faz uma introdução ao trabalho apresentado nesta dissertação. Este capítulo explica o problema, a motivação para o abordar, o contexto em que se insere, a solução apresentada e as principais contribuições.

De modo a melhor entender o estado da arte e as suas aplicações neste projecto, no capítulo 2 foi feito um estudo sobre o trabalho relacionado. Este estudo inclui a conversão de imagens *raster* em vectoriais, a selecção e extracção de descritores e o cálculo do *ranking* obtido por pesquisas de imagens.

No capítulo 3 são descritas as várias técnicas utilizadas na implementação da extracção de características nos clip arts. São abordadas separadamente cada uma das características utilizadas, a extracção dos descritores e a criação dos vectores de descritores.

O sistema implementado e os métodos de recuperação de imagens são descritos no capítulo 4. Neste capítulo é descrita a arquitectura do sistema e os algoritmos utilizados na recuperação de imagens na colecção de clip arts. São descritos os processos de normalização das várias bases de dados de descritores e também os algoritmos aplicados no *matching* de imagens utilizando cada um dos descritores isoladamente e para as diversas combinações de descritores implementadas.

No capítulo 5 é feita uma análise aos resultados experimentais obtidos nos testes efectuados com a solução desenvolvida.

O capítulo final contêm um resumo do trabalho realizado no decorrer desta dissertação e apresenta as conclusões obtidas. É também discutido o trabalho futuro, referindo as extensões possíveis à solução implementada.

2

Trabalho relacionado

A informação contida numa imagem é elevada, o que torna difícil a pesquisa numa base de dados de milhares de imagens, sem que a informação relevante, que a permita descrever, seja extraída [34]. Essa informação é uma simplificação da imagem e deve fazer com que esta possa ser identificada e comparada com outras de modo a encontrar imagens semelhantes. É com esta premissa que têm sido investigados métodos que tiram proveito dessa informação relevante de modo a criar sistemas de pesquisa de imagem baseados no conteúdo.

Nesta secção são discutidos trabalhos de investigação relacionados com o trabalho desenvolvido no âmbito da dissertação. Este estudo permite compreender o estado da arte nas áreas relacionadas com esta dissertação. Assim, foi feito um estudo sobre descritores de imagens e a sua extracção (2.1), um estudo sobre métodos de vectorização de imagens *raster* e simplificação vectorial (2.2.1). Foi também feito um estudo sobre alguns trabalhos relevantes na indexação e pesquisa em CBIR (2.3) e sobre a avaliação de sistemas CBIR (2.4).

2.1 Descritores de imagem *raster*

Nas últimas décadas, os sistemas de recuperação de imagens têm usado cor, textura, forma, topologia e pontos de interesse [12]. Em [42] alguns destes descritores são descritos no contexto do standard MPEG-7. A cor permite gerar descritores simples e oferece bons resultados quando temos elementos que possam

ser representados por cores distintas. Histogramas de cor (2.1.1.1), momentos de cor [50] e cores dominantes são os mais usados. Para bases de dados de imagens com texturas distintas são usados descritores obtidos com o banco de filtros de Gabor, histogramas de arestas e descritores baseados em pontos de interesse [37,40] (2.1.2, 2.1.3). Para obter descritores de forma são usadas técnicas de segmentação para se obter os elementos relevantes na imagem, utilizando algoritmos como Mean Shift [11,45,51].

Bons descritores podem ser caracterizados pela sua invariância a diversas variáveis e transformações [46].

2.1.1 Cor

Um dos descritores mais utilizados e mais antigos para caracterizar uma imagem é o histograma de cor. É um descritor robusto e eficiente [52]. A cor é um dos elementos visuais mais reconhecíveis, e existem vários exemplos em que pode até determinar, por si só, certas classes de objectos. A pigmentação concede a vários objectos e seres vivos uma cor distintiva, como a clorofila nas plantas [52]. Os histogramas podem ser construídos usando diversos sistemas de cor como por exemplo, RGB, tons de cinzento, CMYK ou HSV. Outros métodos de descrever a cor foram implementados de modo a aumentar o desempenho dos histogramas de cor, em [50] Stricker e Orengo descrevem os histogramas de cor cumulativos e os momentos de cor. Estes descritores não incluem as relações entre as diversas cores na imagem nem a sua localização, Huang descreve em [24], os correlogramas de cor, que incluem informação sobre a relação de cores ao longo da imagem.

2.1.1.1 Histogramas de cor

O algoritmo de construção de um histograma baseia-se na contagem de quantas ocorrências de uma cor existem. Contudo, estas cores têm de ser agrupadas. Um histograma separa os valores por diversos contentores, mas normalmente não é utilizado o espaço completo de cores. Reduzindo o espaço de cores podemos agrupar cores semelhantes, através de uma função de equivalência que as considera idênticas e utilizar menos contentores. Na figura 2.1 está representado um histograma de cores de uma imagem no formato RGB.

O que torna os histogramas de cor bastante robustos é o facto de ser invariante à translação e rotação em torno do eixo de visão e de sofrer pequenas mudanças com a alteração do ângulo de visão, mas não quanto ao tamanho [52]. De modo a reduzir as variações com a escala pode ser utilizado um histograma de média

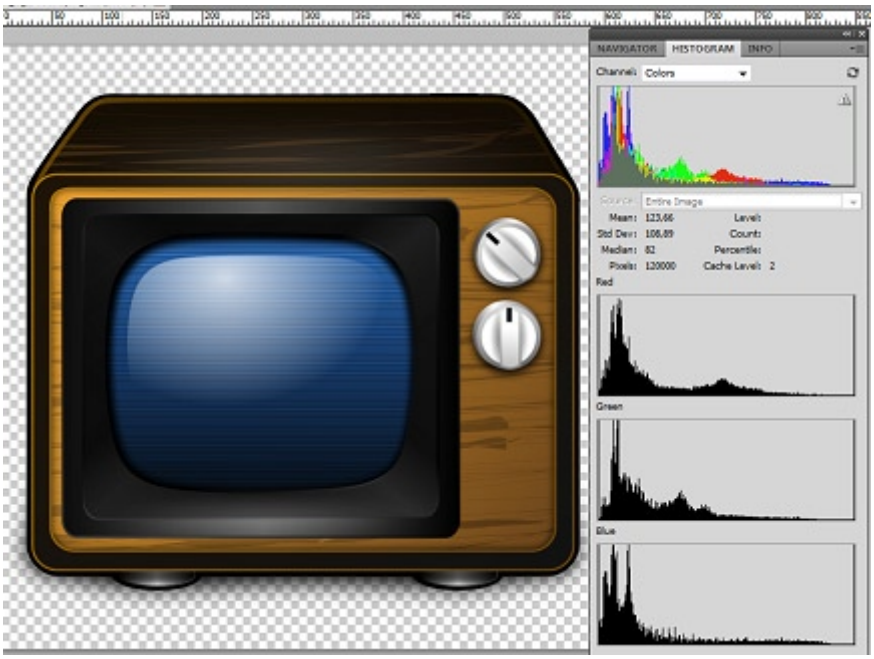


Figura 2.1: Histograma RGB de uma imagem.

de cores, tendo em conta o número total de pixels. Para que possam ser feitas comparações é necessário uma função que calcule a distância entre dois histogramas. O método mais utilizado é a distância Euclidiana [18], mas pode também ser utilizada a métrica de Minkowski [27] ou a distância de Manhattan.

2.1.2 Textura

A textura é um atributo que podemos aplicar a vários contextos na natureza. O ser humano reconhece uma textura como a repetição espacial de um elemento, com várias escalas, orientações ou outras características geométricas ou visuais. Para usarmos a textura em CBIR têm de ser extraídos descritores de textura que possam ser comparados [23].

Alguns sistemas de CBIR utilizam os coeficientes de textura definidos por Tamura [53]. São eles a *coarseness*, o contraste, a *directionality*, a *linelikeness*, a regularidade e a *roughness*. A *coarseness* está relacionada com grandes variações espaciais de níveis de cinzento em relação ao tamanho dos elementos que formam a textura. O contraste mede como os níveis de cinzento variam na imagens e quanto a sua distribuição se aproxima do preto ou do branco. O grau de *directionality* é medido utilizando a frequência da distribuição das arestas orientadas e dos seus ângulos direccionais. A *roughness* pode ser facilmente obtida somando as medidas de contraste e *coarseness*. Na maior parte dos casos, apenas a *coarseness*, o contraste e a *directionality* são utilizados em CBIR. Estas características utilizam a captura de atributos a nível perceptual da imagem. No entanto não

são eficazes para texturas mais complexas [7].

Existem dois métodos principais para extrair descritores de textura. A primeira classe de métodos usa transformações lineares, aplicando filtros à imagem globalmente. As técnicas descritas acima encaixam nesta classe. Estes métodos transformam a imagem original usando um filtro e calculam a energia da imagem transformada em várias regiões. As respostas aos filtros de energia locais representam os valores locais do descritor de textura. No entanto estes método têm uma complexidade de processamento muito alta [23]. Outras técnicas que podem ser incluídas nesta classe são, as máscaras de Laws [30], *Daubechies wavelets* [13], transformadas de Fourier [5] e banco de filtros de Gabor [31] (na figura 2.2 são representados alguns filtros do banco de filtros de Gabor 2D).

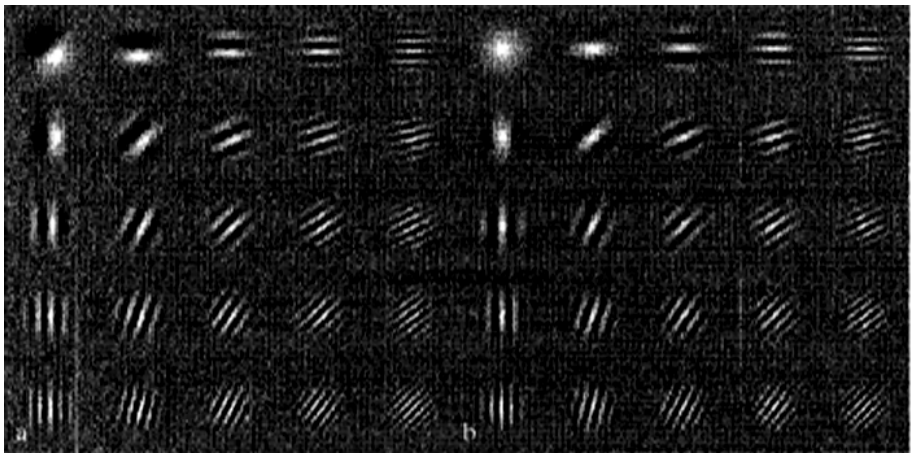


Figura 2.2: Representação de filtros do banco de filtros de Gabor 2D.

Os filtros de Gabor têm sido muito usados em aplicações de análise de imagens, incluindo classificação e segmentação, reconhecimento de imagens, *tracking* de movimento e recuperação de imagem. As técnicas da segunda classe de métodos irão dividir a imagem em blocos de pixels e depois é aplicado ao pixel central desse bloco uma transformada, como a Discrete Fourier Transform (DFT), a Discrete Cosine Transform (DCT) ou Discrete Wavelet Transforms (DWT). As transformadas *wavelet* são semelhantes às transformadas de Fourier, mas além da informação sobre a frequência contém informação em várias resoluções. Para extrair os descritores de textura para cada pixel é aplicada uma janela, com um tamanho predeterminado, a cada pixel. A janela "desliza" sobre todos os pixels e realiza a transformada *wavelet* em cada ponto de modo a determinar os descritores de textura de cada pixel (ver figura 2.3).

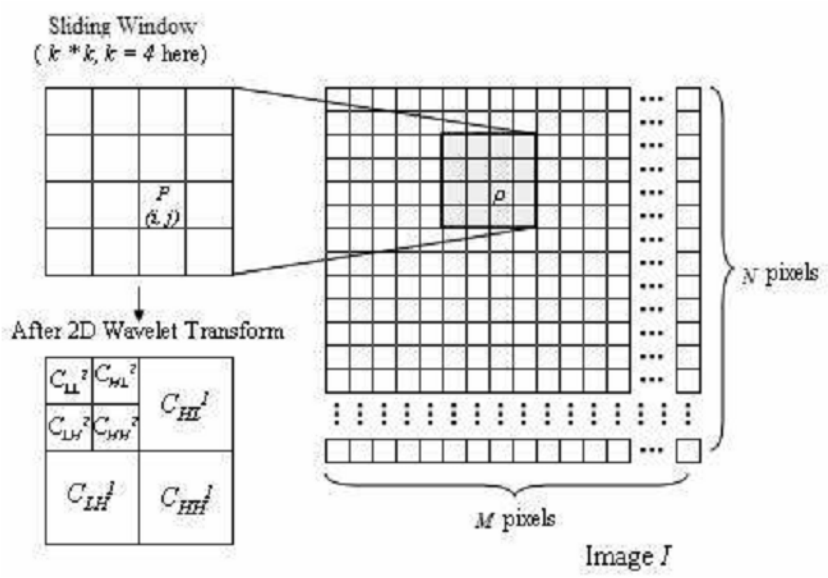


Figura 2.3: Extracção de descritores de textura do pixel(i, j).

As técnicas mais promissoras são descritores multi-resolução obtidos com transformadas de *wavelets* ortogonais [23, 55] ou com o banco de filtros de Gabor. Os seus descritores contêm informação sobre a distribuição espacial das arestas orientadas, na imagem, em diversas escalas. Outra abordagem interessante foi feita por Lam em [29] e utiliza programação genética (GP - Genetic Programming) para criar métodos óptimos de extracção de descritores através de algoritmos genéticos. Na tabela 2.1, podemos ver os resultados de análises de precisão de pesquisas efectuadas com diversas técnicas de extracção de descritores que incluem descritores de textura.

Descritores utilizados	Brodatz [56]	Vistex [56]
Unser	92.6%	81.4%
Galloway	84.7%	70.4%
Laine - *Daubechies wavelets*	92.4%	75.6%
Laws	89.7%	79.8%
Coef. de Fourier	92.7%	80.1%
Chen	93.1%	84.5%
Pikaz e Averbuch	79.4%	74.4%
Gabor	92.2%	75.4%
Markov	83.4%	65,6%
Amadasun	83.4%	65.6%
Mao e Jain	86.3%	73.0%
Amelung	93.0%	82.1%
Haralick	86.1%	75.5%
Descritores GP	81.5%	74.8%
Descritores GP + Haralick	88.2%	83.2%

Tabela 2.1: Desempenho de vários algoritmos de extracção de descritores. Os resultados são uma combinação dos resultados em [56] e [29]. Ambas usaram as as bases de dados de texturas Brodatz e Vistex.

2.1.3 Pontos de interesse

David Lowe apresentou em [37] um método de extrair descritores em pontos de interesse, denominado por SIFT (Scale-invariant Feature Transform). Estes descritores são invariantes quanto à escala e rotação e produzem bons resultados na tarefa de encontrar semelhanças em imagens que contenham ruído, distorções e mudanças de iluminação (como pode ser visto na figura 2.4). Esta abordagem produz um elevado número de descritores em toda a imagem, no entanto, os descritores produzidos são distintos, o que permite com grande probabilidade encontrar um resultado correto em bases de dados com descritores de diversas imagens.

O algoritmo desenhado por Lowe para detecção de pontos de interesse e extracção dos descritores está optimizado utilizando uma *pipeline* de filtros em que as operações mais complexas apenas são aplicadas aos pontos que passem nos filtros. Um exemplo dos resultados da aplicação de alguns destes filtros é apresentado na figura 2.6. Em seguida são descritos os passos principais da *pipeline* que gera um conjunto de descritores.

- **Seleção de picos no espaço de escalamento** - No primeiro passo têm de ser percorridas várias escalas e localizações da imagem, que é implementada usando uma função DoG (*Difference-of-Gaussian*) para identificar potenciais pontos de interesse que sejam invariantes quando à escala e orientação.

Figura 2.4: À esquerda podemos encontrar as imagens de treino que foram usadas para retirar os descritores SIFT. Ao meio temos os objectos inseridos numa outra fotografia com muitos elementos. Os resultados do reconhecimento podem ser vistos à direita.

- **Localização de pontos de interesse** - Para cada possível localização é utilizado um modelo de modo a determinar a localização, escala e contraste. São escolhidos os pontos de interesse com base na sua estabilidade.

- **Atribuição de orientação** - Através das propriedades locais da imagem é atribuída à localização do ponto de interesse, uma ou mais orientações. A partir daqui todas as operações passarão a ser feitas tendo em conta a orientação, escala e localização de cada um destes descritores. Deste modo estes descritores tornam-se invariantes quanto a este tipo de transformações.

- **Geração do descritor de pontos de interesse** - O descritor irá agrupar a informação em redor de uma dada área definida em torno dos pontos de interesse. Primeiro são medidos os gradientes locais da imagem nos pontos recolhidos, como pode ser visto na figura 2.5. São depois transformados numa representação que permite distorção da forma e mudanças de iluminação locais. As medidas calculadas anteriormente são acumuladas num histograma de orientação, representando o conteúdo das regiões.

Uma imagens com muitos elementos, pode conter muitos pontos de interesse. Para resolver o problema no *matching* dessas imagens, Lowe propõe que sejam agrupados vários sub-conjuntos de pontos de interesse, que definam o mesmo objecto e tenham a mesma localização, escala e orientação relativos, em ambas as imagens.

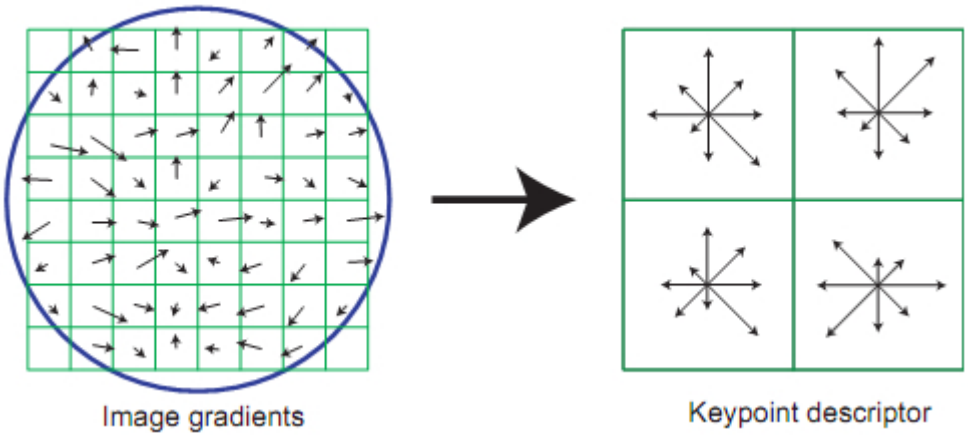


Figura 2.5: Criação do descritor de ponto de interesse a partir dos gradientes da imagem.

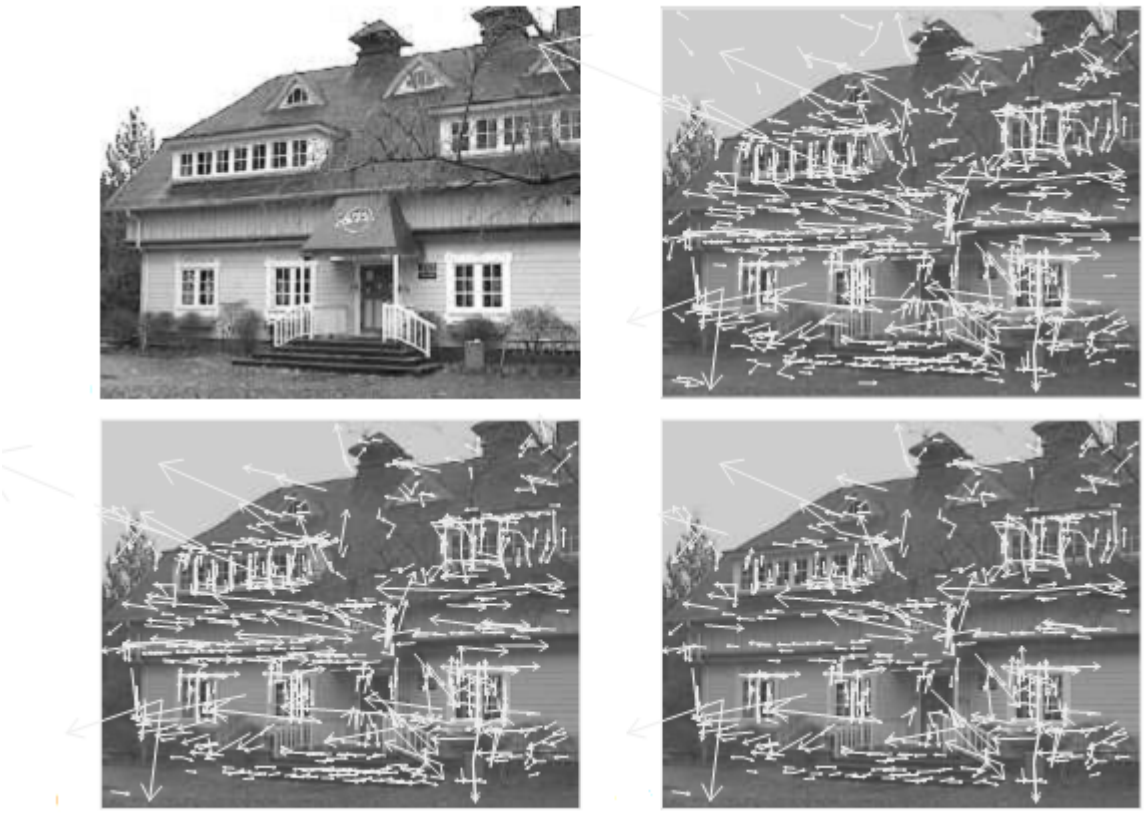


Figura 2.6: Várias fases do processo de obtenção dos descritores de pontos de interesse SIFT. Os pontos de interesse estão representados como vectores que indicam a escala, orientação e localização.

2.2 Descritores de imagem vectoriais

As imagens vectoriais são representadas utilizando os elementos e a estrutura que compõem a imagem. Em sistemas CBIR, devido à sua estrutura, as imagens vectoriais necessitam de uma abordagem diferente das imagens *raster*. Como foi descrito anteriormente as imagens *raster* utilizam essencialmente a cor e a textura para descrever o conteúdo, as imagens vectoriais necessitam de características diferentes para descrever o conteúdo.

No trabalho realizado por Pedro Sousa e Manuel J. Fonseca [49] são utilizadas, como características para descrever o conteúdo das imagens, características relacionadas com a forma dos elementos (geometria) e o seu posicionamento e relação com outros elementos (topologia). Testes realizados [49] mostram que quando os utilizadores pesquisam por imagens clip art usando esboços recorrem mais à forma ou geometria dos objectos que à sua relação espacial (ver fig. 2.7). Estas pesquisas são realizadas com esboços dos objectos mais relevantes na imagem que querem pesquisar. Deste modo, o esboço poderá ser mais semelhante a uma representação vectorial mais simplificada.

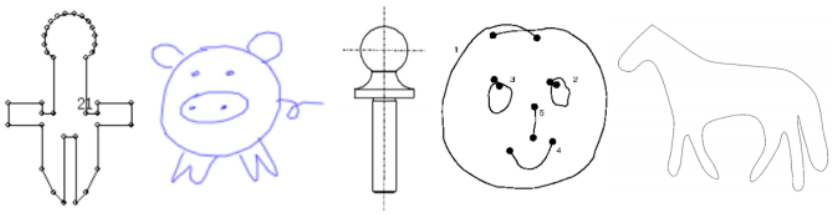


Figura 2.7: Esboços suportados por algumas soluções.

2.2.1 Vectorização de imagens *raster* e simplificação vectorial

Segundo Karl Tombre et al. [54], um dos factores para robustez no processo de vectorização é minimizar o número de parâmetros e *thresholds* necessários. De modo a simplificar o processo de vectorização, primeiro devem ser removidos os elementos desnecessários. Estes elementos costumam ser pequenos objectos ou linhas e devem ser eliminados, pois não são necessários à extracção de características. Para remover estes "artefactos" e o ruído que possa existir na imagem, são aplicados filtros de convolução (como *blur*), normalização de histogramas e *thinning*. Para termos uma imagem mais nítida também podem ser aplicados filtros, que aumentam o contraste nas imagens e que reforçam os contornos [16]. Os

passos do processo de vectorização desse sistema podem ser vistos na figura 2.8. Neste sistema é aplicado um *thresholding* para a binarização da imagem. Para obter os polígonos a partir da imagem binarizada este sistema utiliza uma combinação do algoritmo de chain-coding de primitivas [41] com o algoritmo de Douglas-Peucker [15] para reduzir o número de pontos que representam uma curva. Existem outros métodos que permitem segmentar a imagem e criar sua representação geométrica, como RANSAC [17], mean shift [11], Canny [6]. Para obter uma melhor segmentação da imagem nos seus elementos relevantes os algoritmos usam diversas "guias", como contornos, ou cores [6, 11, 17, 51].

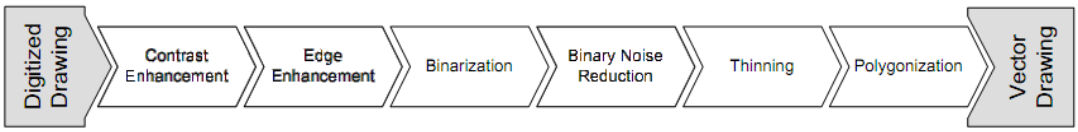


Figura 2.8: Algumas fases do processo de vectorização de uma imagem *raster*.

2.2.2 Métodos de segmentação e detecção de padrões

Existem vários métodos para segmentar imagens e identificar a sua geometria ou extrair a sua representação vectorial como mapas de saliência [44], uso de *patches* curvilíneos para obter uma vectorização de imagens fotográfica sem perder muito detalhe [60] e algoritmos baseados em *flood-fill* [18]. Huang et at. implementou [25] um método que usa os *strokes* feitos nos desenhos *cartoon* como descritor geométrico e como método de segmentação. Vamos focar o estudo nas três técnicas descritas abaixo.

2.2.2.1 Mean Shift *clustering*

O algoritmo Mean Shift, desenvolvido por D. Comaniciu e P. Meer [11], é um método iterativo que aproxima os valores de um conjunto para a média do valor dos seus vizinhos. É baseado numa função *kernel* para determinar os pesos dos vizinhos para os cálculos sucessivos da média. Este processo é aplicado em tarefas de *clustering* em visão por computador e processamento de imagem. A convergência de valores segundo a vizinhança produzida pelo Mean Shift permite que seja um método de *clustering* nas imagens.

É com base neste método de *clustering* que podemos criar a segmentação da imagem. O algoritmo quando aplicado numa imagem usa uma janela para percorrer todos os pixels. Iterativamente vai re-calculando a cor média. O resultado

final após aplicar a cor média de um *cluster* a todos os pontos desse *cluster* será uma imagem constituída por manchas de cor. Estas manchas representam os objectos mais relevantes em termos de cor (ver figura 2.9) e poderão ser extraídos como forma geométrica.

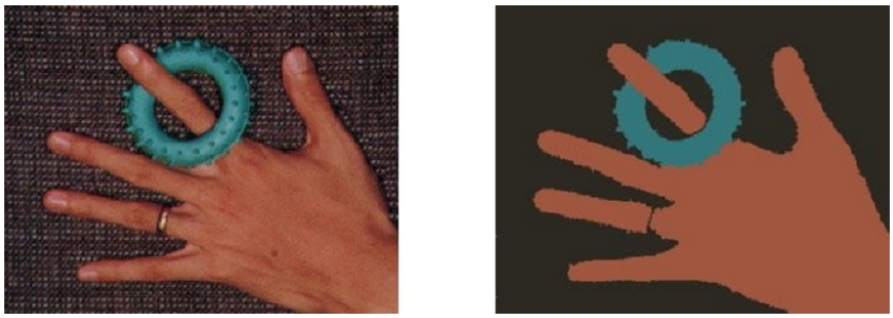
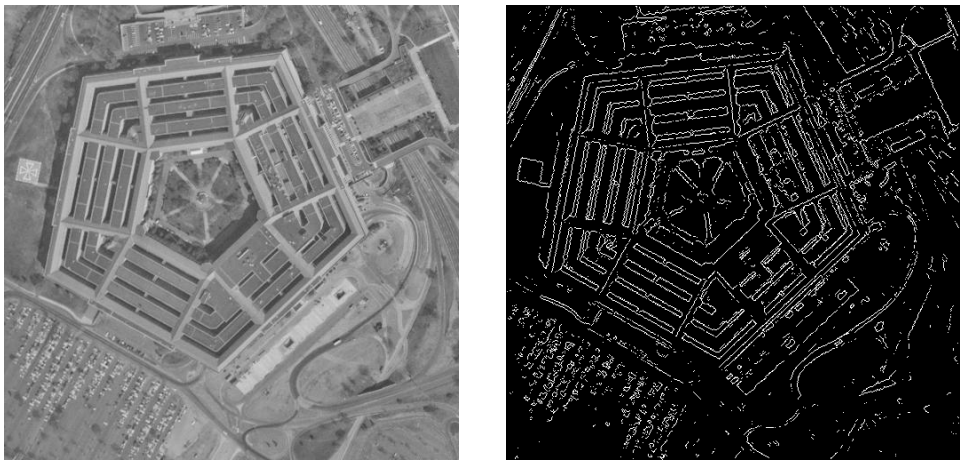


Figura 2.9: Exemplo de segmentação de uma imagem *raster* usando Mean Shift.

2.2.2.2 Canny

Este algoritmo é um detector de contornos, com resistência ao ruído, desenvolvido por John F. Canny [6]. Foi desenvolvido para utilizar o mínimo de pontos para definir uma aresta. São aplicados filtros de convolução de modo a suavizar a imagem e remover o máximo de ruído possível, como um filtro Gaussiano (ainda antes de ser feita a pesquisa). Depois destes filtros a imagem fica com um aspecto *blurred*. De seguida são pesquisados os gradientes de intensidade. Em princípio, onde forem localizados pontos com um valor de intensidade alto é mais provável que se esteja perante uma aresta. Após encontrar os gradientes que correspondam a uma aresta irá continuar o processo até que todos os contornos da imagem sejam obtidos. O resultado será uma imagem bitmap que indica se um ponto pertence aos contornos ou não (ver fig. 2.10). A partir dos contornos podem ser extraídos os objectos geométricos.



(a) (b)

Figura 2.10: Extracção de modelos usando o algoritmo Canny. A figura 2.10a representa a imagem original. Na figura 2.10b podemos ver o resultado da aplicação do algoritmo Canny.

2.2.2.3 Random sample consensus - RANSAC

O RANSAC originalmente proposto por Fishler e Bolles [17] é um método de recuperação de modelos a partir de dados que contenham muito ruído. O RANSAC assume que existem dois tipos de pontos, *inliers* e *outliers*. Os *inliers* são pontos que coincidem com os parâmetros de um dado modelo. Os *outliers* são todos os pontos que não pertencem ao modelo. O objectivo do algoritmo é descobrir, de entre todos os pontos, quais são os pontos que são *inliers* e quais são os pontos que são *outliers*. O resultado final será um conjunto de pontos que foram considerados *inliers* por pertencerem ao modelo e um conjunto de *outliers*.

Ao contrário de outros algoritmos que usam o máximo de dados possível e daí procuram remover o ruído, o RANSAC começa com o menor número possível para descrever o modelo (e.g., dois pontos para representar uma linha) e vai aumentando o conjunto de dados com os pontos que se encaixam no modelo [17](ver fig. 2.11). O algoritmo básico pode ser resumido da seguinte forma:

1. Seleccionar aleatoriamente um número mínimo de pontos necessários para determinar os parâmetros do modelo.

2. Resolver os parâmetros para a função do modelo (e.g., para uma linha $y = ax + b$).

3. Determinar quantos pontos do conjunto total de pontos fazem parte do modelo, com alguma tolerância.

4. Se a razão entre o número de *inliers* e o número total de pontos excederem um *threshold* predefinido, recalcular os novos parâmetros do modelo usando todos os *inliers* identificados e terminar. Caso contrário repetir os passos anteriores. O ciclo continuará até terminar ou até chegar a um valor *n* predefinido.

Este algoritmo permite encontrar modelos mesmo com 50% de ruído. Este facto foi demonstrado por Zuliani [63]. Quando aplicado à representação geométrica de uma imagem permite eliminar o ruído e obter os modelos dos objectos na imagem. A maior desvantagem consiste na dificuldade em implementar todos os modelos que possam aparecer de forma a encontrar os objectos que pretendemos.

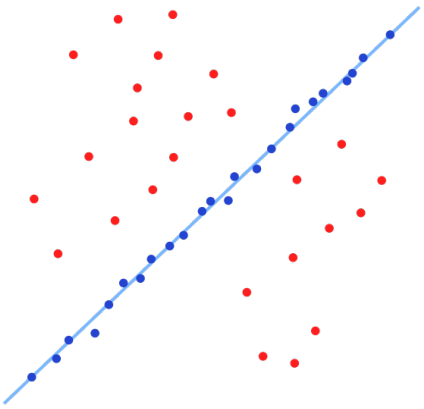


Figura 2.11: Exemplo da aplicação do algoritmo RANSAC. Os pontos mais perto da recta são os *inliers* estimados. Os pontos que ficaram mais afastados e não pertencem ao modelo são os *outliers*.

Após ter sido extraída a geometria da imagem *raster* terão de ser extraídos os descritores de geometria. Para tal será usada a biblioteca CALI [22] (ver figura 2.12). Esta biblioteca foi desenvolvida inicialmente para reconhecimento de caligrafia mas como também tem um bom desempenho a reconhecer desenhos feitos à mão foi expandido para poder gerar vários descritores geométricos. Esses descritores são depois convertidos para descritores *affine-invariant*. Os descritores são armazenados num vector que permite descrever de uma forma independentemente da escala, rotação, translação ou tipo de linha.

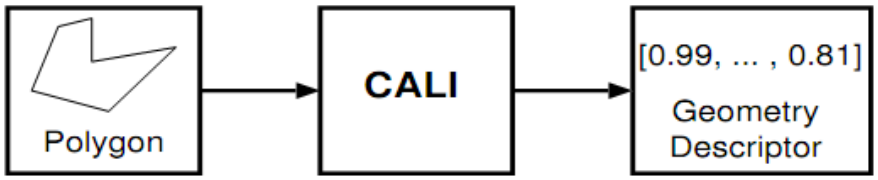


Figura 2.12: Obtenção de descritores a partir da geometria da imagem.

2.3 Pesquisa e indexação

Nesta secção são apresentados métodos de pesquisa de imagens e de indexação utilizados em sistemas de CBIR.

2.3.1 Pesquisa do espaço de descritores

Do ponto de vista da base de dados, a pesquisa de imagem pode ser tratada como um problema de interrogação [62]. O objectivo da pesquisa de imagens é encontrar uma ou mais imagens semelhantes à interrogação, ou seja encontrar os seus vizinhos no espaço descrito pelo vector que representa os descritores. Diferentes tipos de pesquisa podem ser usados para diferentes tipos de aplicações [4, 62]. Para tal é necessário elaborar uma medida de semelhança. Uma medida de semelhança devolve, para dois objectos, um valor positivo que denota a sua semelhança. Um método de calcular esta semelhança é através da distância Euclidiana, mas também pode ser usada a fórmula de Minkowski [27]. O estudo efectuado por Christian Böhm et al [4] em bases de dados multimédia refere dois métodos utilizados quando pretendemos pesquisas por semelhança. Semelhança por gama de valores, quando estamos interessados em todos os objectos que tenham um valor de semelhança abaixo de uma *threshold* ε e semelhança NN (*Neareast Neighbor*), quando estamos interessados nos objectos que são mais semelhantes ao objecto pelo qual estamos a pesquisar. Em **CBIR** são usados, para resolver problemas de semelhança, soluções baseadas em descritores. A ideia é extrair os descritores mais relevantes, como foi descrito na secção 2.1. Esses descritores são mapeados num vector de descritores de alta dimensão. A pesquisa é feita procurando na base de dados de vectores de descritores por objectos com vector de descritores semelhantes [4, 12, 18, 25] (ver fig. 2.13). Com esta aproximação as pesquisas de objectos semelhantes tornam-se em ε-*queries* e *NN-queries*. Tendo em consideração o tamanho das bases de dados e o espaço dimensional é essencial o uso de técnicas de indexação eficientes [4]. Um sistema de indexação, para recuperação de imagens, para ser utilizado em vários tipos de pesquisa em diferentes

aplicações deverá permitir [4, 62]:

1. Interrogações pontuais, para pesquisa de um único objecto, o mais semelhante;

2. Interrogações-NN, para obter os resultados mais próximos da pesquisa;

3. Interrogações por gama de valores, para processar pesquisas em que a distância de semelhança entre o objecto e a interrogação está abaixo de um *threshold e* ou entre dois valores.

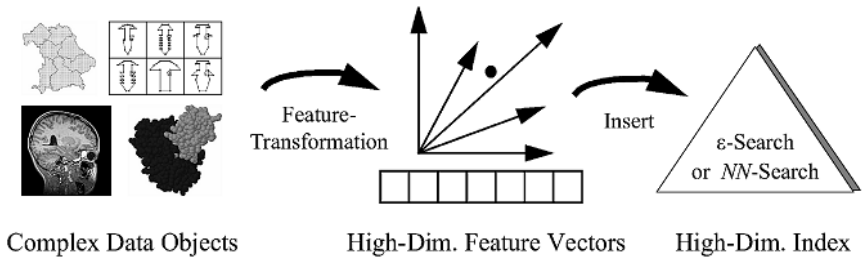


Figura 2.13: Ideia base da pesquisa por semelhança de descritores

Estes tipos de pesquisa, à excepção da pesquisa pontual, não nos permitem saber qual o número de resultados que será obtido. De modo a resolver este problema existe a pesquisa *k*-NN. Esta pesquisa mantém uma lista de *nearest neighbors* como possíveis candidatos, onde apenas são retornados no resultado os *k* objectos mais semelhantes.

2.3.2 Estruturas de indexação

No inicio do desenvolvimento de bases de dados de sistemas multimédia, os objectos eram simplesmente guardados como ficheiros num directório ou entradas numa tabela de uma base de dados SQL. Quanto ao desempenho e eficiência computacional estas estruturas não eram as melhores. Os sistemas de ficheiros utilizam pesquisa linear dentro dos directórios e as bases de dados SQL não eram optimizadas para ficheiros grandes, como ficheiros multimédia. À medida que as colecções multimédia continuaram a crescer não se podia continuar a usar estes meios para armazenamento e pesquisa pois tinham custos computacionais muito altos e tempos de resposta muito demorados [34].

Foi então que os investigadores optaram por bases de dados baseada em semelhança [33]. Estas bases de dados utilizam índices baseado em árvores para

obter complexidade logarítmica, no entanto, na altura, necessitavam de hardware especializado. Como foi referido acima (ver secção 2.3.1) estas estruturas usam um vector de descritores como índice. Métodos de indexação em dimensões elevadas são baseados em métodos de *clustering* hierárquico do espaço de dados [4], sendo semelhantes em termos de estrutura a uma árvore B+. Existem vários métodos baseados em k-d-tree para indexar e pesquisar dados de dimensões elevadas como SS-tree [59], TV-tree [36], X-tree [2], SR-tree [28], NB-tree [21], multi resolution k-d tree [14], CIR-tree [61], sendo este último especialmente desenvolvido para pesquisa de imagens baseada no conteúdo. Na tabela 2.2 podemos visualizar várias características para diversos métodos de indexação.

Nome	**Problemas em dimensões elevadas**	**Tipos de pesquisas suportadas**	**Utilização de espaço de armazenamento**	**Distribuição/ Tamanho das entradas no índice entradas**
R-tree	Mau algoritmo de *split* leva a directórios deteriorados	NN, Região, *Range*	Mau	Mau, linearmente dependente da dimensão
R-tree*		NN, Região, *Range*	Médio	Mau, linearmente dependente da dimensão
X-tree	A probabilidade de as pesquisas fazerem *overlapping* leva a má performance	NN, Região, *Range*	Médio	Mau, linearmente dependente da dimensão
SS-tree	*Overlapping* elevado no directório	NN	Médio	Muito bom, independente da dimensão
TV-tree	Apenas útil para tipos específicos de dados	NN	Médio	Mau, alguma dependência da dimensão
SR-tree	Directórios com tamanho muito grande	NN	Médio	Muito bom, independente da dimensão
Curvas de Peano	Mau particionamento do espaço	NN, Região, *Range*	Médio	Tão bom como árvores B+, independente da dimensão
Pyramid tree	Problemas com interrogações assimétricas	Região, *Range*	Médio	Tão bom como árvores B+, independente da dimensão

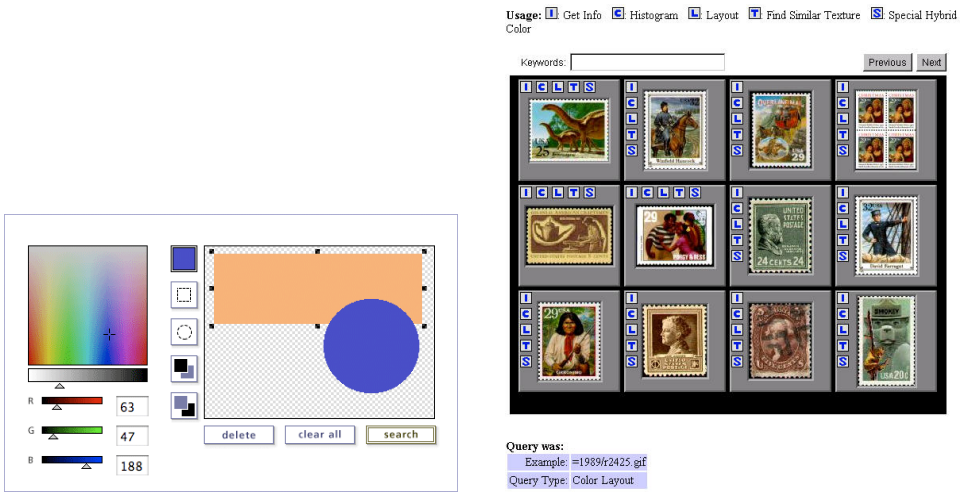
Tabela 2.2: Comparação qualitativa de estruturas de indexação de dimensão elevada [4].

2.4 Aplicações de recuperação de imagens

Para melhor entender o funcionamento de várias destas técnicas e a sua aplicação em sistemas de CBIR foram estudados alguns sistemas. Um dos sistemas mais

utilizados para pesquisa e utilização de clip arts é o incluído em muitos processadores de documentos, como o Microsoft Office, o LibreOffice e o OpenOffice. Estes sistemas utilizam pesquisa por *keywords* e divisão por categorias. São muito limitados pois as imagens necessitam de meta-dados associados de modo poderem ser utilizados os filtros de categorias e o reconhecimento de *keywords*. A recuperação de imagens é feita utilizando informação textual inserida pelo utilizador ou então feita manualmente, percorrendo as diversas colecções disponíveis.

O sistema QBIC [18] foi desenvolvido pela IBM para recuperação de imagens e vídeos. São usadas cores, texturas, formas, movimento de objectos e palavras-chave (ver figura 2.14).



(a) (b)

Figura 2.14: Sistema QBIC.

A pesquisa pode ser feita usando os descritores isoladamente, permitindo selecção de cores, texturas, formas ou *sketching*. Podem também ser feitas pesquisas com combinações de alguns destes descritores, a partir do desenho de um esboço e de escolha de cores dominantes. O sistema possui dois componentes, um componente para popular a base de dados com imagens e outro componente para efectuar interrogações à base de dados. Durante a criação da base de dados são processadas as imagens e extraídos os descritores do conteúdo. O componente de pesquisa aceita as interrogações referidas acima, extraindo os descritores da interrogação, e comparando com os descritores na base de dados são devolvidos os resultados mais semelhantes. Este sistema demonstrou o *query by sketch* e *query by exemple* como técnicas de pesquisa promissoras.

O projecto VisualSEEk [48] foi desenvolvido para pesquisa de descritores visuais em bases de dados de imagens utilizando esboços de zonas de cor. Utiliza

zonas de cor, a sua posição e as relações espaciais entre as zonas.

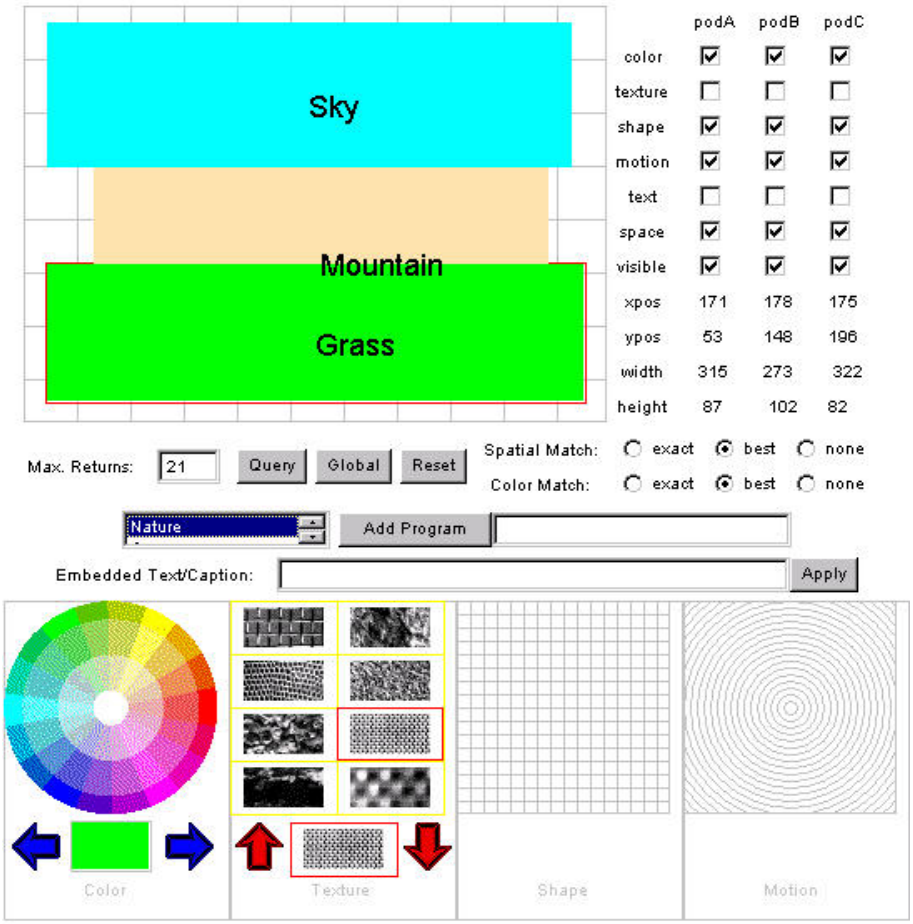


Figura 2.15: Pesquisa com o VisualSEEk.

Na fase de pré-processamento e criação da base de dados o sistema divide a imagem em zonas que tenham a mesma cor dominante. Dessas zonas são retirados os descritores de cor, a relação espacial entre as zonas e o seu tamanho. De modo a extrair a cor é utilizada *back-projection* de conjuntos de cor. É utilizado um conjunto de 166 cores no modelo HSV. São permitidas pesquisas por cor, pesquisa de uma zona e pesquisa de múltiplas zonas. Na fase de pesquisa é feita a pesquisa por cada zona individualmente (ver figura 2.15). Os resultados das pesquisas individuais são combinados e são recuperadas as imagens que possuam o maior número de zonas. As relações espaciais não são utilizadas nesta fase de modo a reduzir a complexidade da pesquisa. Depois são comparadas as relações espaciais de modo a filtrar os resultados.

O sistema Photofinder 2 [43] (ver figura 2.16) utiliza um método de segmentação por Mean Shift *clustering* para classificar e pesquisar imagens fotográficas. Na fase de criação da base de dados o sistema utiliza o Mean Shift *clustering* para

obter uma segmentação por manchas de cor. Da segmentação é obtido o grafo com a topologia da imagem e a distribuição das cores pelas regiões.

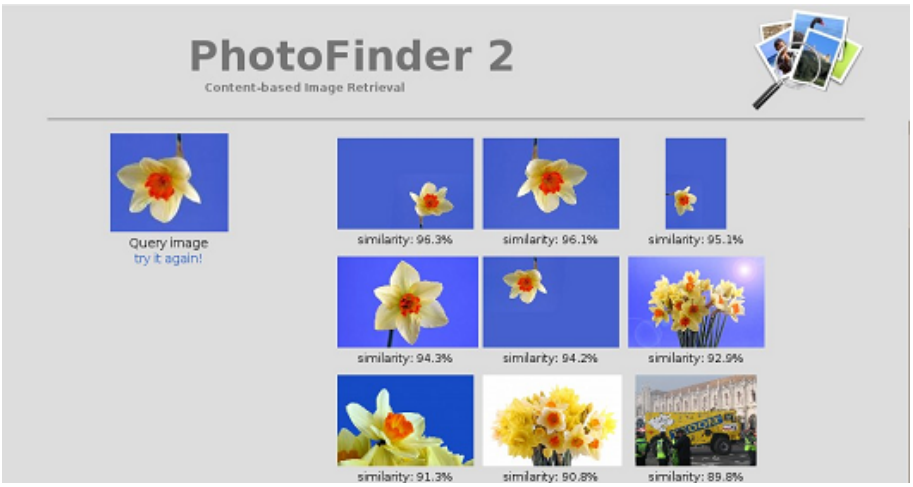


Figura 2.16: Sistema Photofinder 2.

O sistema utiliza a aplicação EDISON, que implementa o Mean Shift *clustering*. Esta aplicação extrai uma representação da imagem por manchas de cor e uma representação dos contornos da imagem. O sistema extrai as imagens de manchas de cor e de contornos utilizando vários níveis de segmentação. O Photofinder utiliza uma implementação do algoritmo de Freeman para efectuar a vectorização, usando a imagem de contornos e a imagem com as manchas de cor. Da imagem vectorial são extraídos os descritores de geometria usando o sistema desenvolvido em [22]. Da imagem vectorial é também formado o grafo da topologia. A partir do grafo são criados descritores de topologia que contêm a informação sobre inclusão e adjacência. Ao ter a informação referente a subgrafos e diversos níveis de segmentação permite que sejam feitas pesquisas totais e também pesquisas por imagens que contenham o objecto.

O sistema Indagare [49] utiliza descritores de geometria e grafos de topologia para efectuar recuperação de imagens clip art (ver figura 2.17). Este sistema foi feito com base em sistemas desenvolvidos anteriormente para recuperação de desenhos CAD baseados em pesquisas por esboços [16, 19, 20]. As imagens *raster* são simplificadas de modo a serem retirados os descritores de geometria [22]. Os grafos de topologia incluem a informação sobre inclusão e adjacência. A informação sobre a cor é excluída e são usados apenas os descritores para a geometria (objectos contidos na imagem) e topologia (relação espacial entre os objecto na imagem). De modo a reduzir a pesquisa primeiro é feita uma filtragem usando a topologia e depois é feita a pesquisa usando a geometria. A pesquisa pode ser efectuada através de esboços de imagens e também imagens de exemplo. O nosso

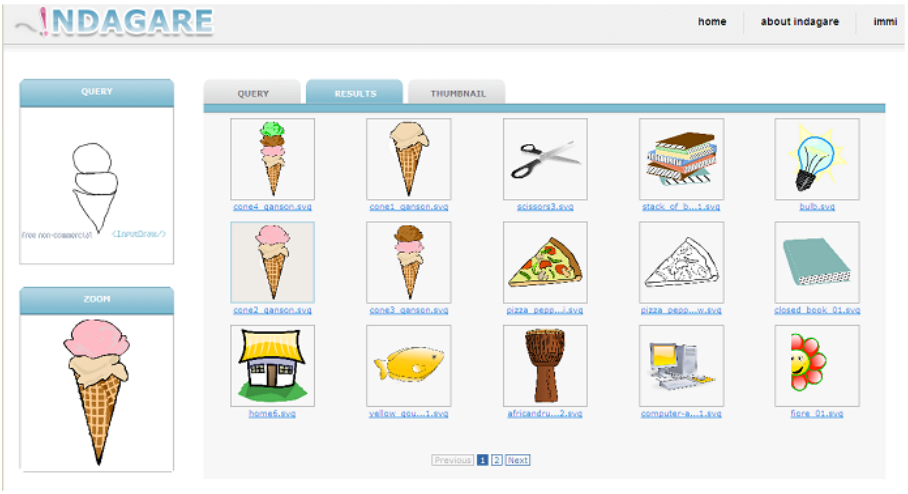


Figura 2.17: Sistema Indagare.

sistema combina os descritores vectoriais utilizados neste sistema, com os descritores de cor, textura e pontos de interesse, mas utilizando apenas pesquisas por exemplo.

3

Extracção de informação em Clip Arts

Neste capítulo são apresentados os conceitos e tecnologias utilizados na extracção de descritores utilizados na realização do projecto proposto. Do estudo realizado na secção anterior foram escolhidas algumas tecnologias que posteriormente foram implementadas. Também é descrito o processo de pré-processamento aplicado às imagens.

Como primeiro passo foi necessário implementar métodos de extracção de descritores das imagens. Para cada um destes métodos implementados é gerado um vector do descritor, que permite efectuar comparações entre imagens de modo a realizar pesquisas.

3.1 Pré-processamento

Antes de procedermos à extracção de informação de qualquer imagem há alguns aspectos a considerar. De modo a aumentar o desempenho do sistema, existem várias transformações que podem ser previamente aplicadas às imagens. Uma característica que tem influência em alguns algoritmos de processamento de imagem é o tamanho da imagem pois o número de pixels pode ser directamente relacionado com o tempo de processamento da imagem em vários dos algoritmos utilizados. Com base em testes efectuados [43], que utilizam algoritmos de segmentação semelhantes aos que foram utilizados neste projecto, decidimos utilizar

como altura e largura máxima 400 pixels, que oferecem um bom compromisso entre a qualidade da imagem e o desempenho. Outro factor levado em consideração foram os algoritmos desenvolvidos por José Amante [1], do qual foi utilizado o algoritmo de redução de cores, referido mais à frente, e que também utiliza como dimensão máxima 400 pixels e redimensiona imagens com dimensões superiores. De modo a reduzir a altura e largura das imagens sem perder qualidade foi feito um recorte dos desenhos nas imagens. Contudo, algumas imagens possuem *backgrounds* grandes, em que os clip arts não estão centrados. Assim, como primeiro passo os clip arts são recortados de modo a conter apenas o conteúdo do clip art e remover o *background* irrelevante. Se mesmo assim a imagem possuir uma das dimensões superior a 400 pixels é feito o redimensionamento.

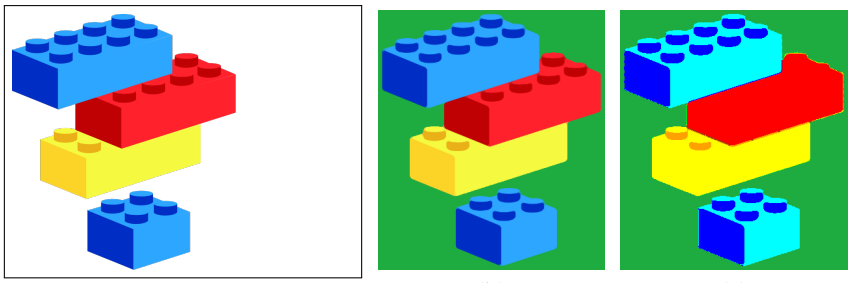
As imagens, na sua maioria, possuem elementos de pequenas dimensões que não representam qualquer informação útil e que apenas contribuem para a complexidade da imagem. De modo a tentar remover alguns destes artefactos foi aplicado um filtro *blur* à imagem. De modo a não incluir novas cores na imagem foi utilizado um filtro de mediana com uma janela de 5x5 pixels. O resultado são imagens com contornos mais suaves e com menos artefactos, o que resulta em menos pontos com informação indesejada, o que por sua vez acelera o processamento das imagens.

	Color	R	G	B	H	S	V
Red		255	0	0	0	100	100
Brown ('saddlebrown')		139	69	19	24	86	54
Orange		255	165	0	38	100	100
Yellow		255	255	0	60	100	100
Green ('lime')		0	255	0	120	100	100
Cyan		0	255	255	180	100	100
Blue		0	0	255	240	100	100
Purple		128	0	128	300	100	50
Pink ('deeppink')		255	20	147	327	92	100
Black		0	0	0	0	0	0
Gray		128	128	128	0	0	50
White		255	255	255	0	0	100

Figura 3.1: Paleta de cores JNS.

Outra característica importante para muitos dos descritores utilizados é a cor. As imagens podem ser compostas de variadíssimas cores, dependendo do modelo de cor utilizado, segundo o modelo HSV a imagem é definida pela sua tonalidade, luminosidade e saturação. De modo a reduzir a amostra de cores nas imagens e aumentar a probabilidade de *match* nas pesquisas decidimos implementar um algoritmo de redução do número de cores numa imagem. Como foi estudado anteriormente [3,9,58] existem aproximadamente doze cores que todas as culturas consideram diferentes ou JNS ('Just Not the Same'), apresentados na imagem 3.1. Para tal foi implementada uma variante do algoritmo de *fuzzy quantization* desenvolvida em [1] para converter a *pallete* de cores da imagem para uma

pallete com apenas doze cores utilizando o modelo de cor HSV.



(a) (b) (c)

Figura 3.2: Imagens após o pré-processamento. 3.2a Imagem original. 3.2b paleta de cores completa. 3.2c paleta de cores JNS

3.2 Descritores de geometria extraídos de imagens vectoriais

Um dos principais objectivos no âmbito desta dissertação é a comparar o desempenho do uso de descritores extraídos de imagens vectoriais com descritores de versões de imagens vectoriais simplificadas. As imagens vectoriais são compostas pela representação vectorial de vários polígonos. Através da forma e da relação entre os vários polígonos é possível extrair descritores que representam informação sobre a topologia e os elementos geométricos que constituem a imagem. Como já foi referido, utilizamos trabalho desenvolvido anteriormente [49] para extrair os descritores das imagens vectoriais.

3.2.1 Simplificação de imagens vectoriais

Antes da extracção dos descritores é necessário simplificar as imagens em elementos geométricos. Nesta secção apresentamos algumas das técnicas estudadas de modo a atingir o nível de segmentação e informação necessária.

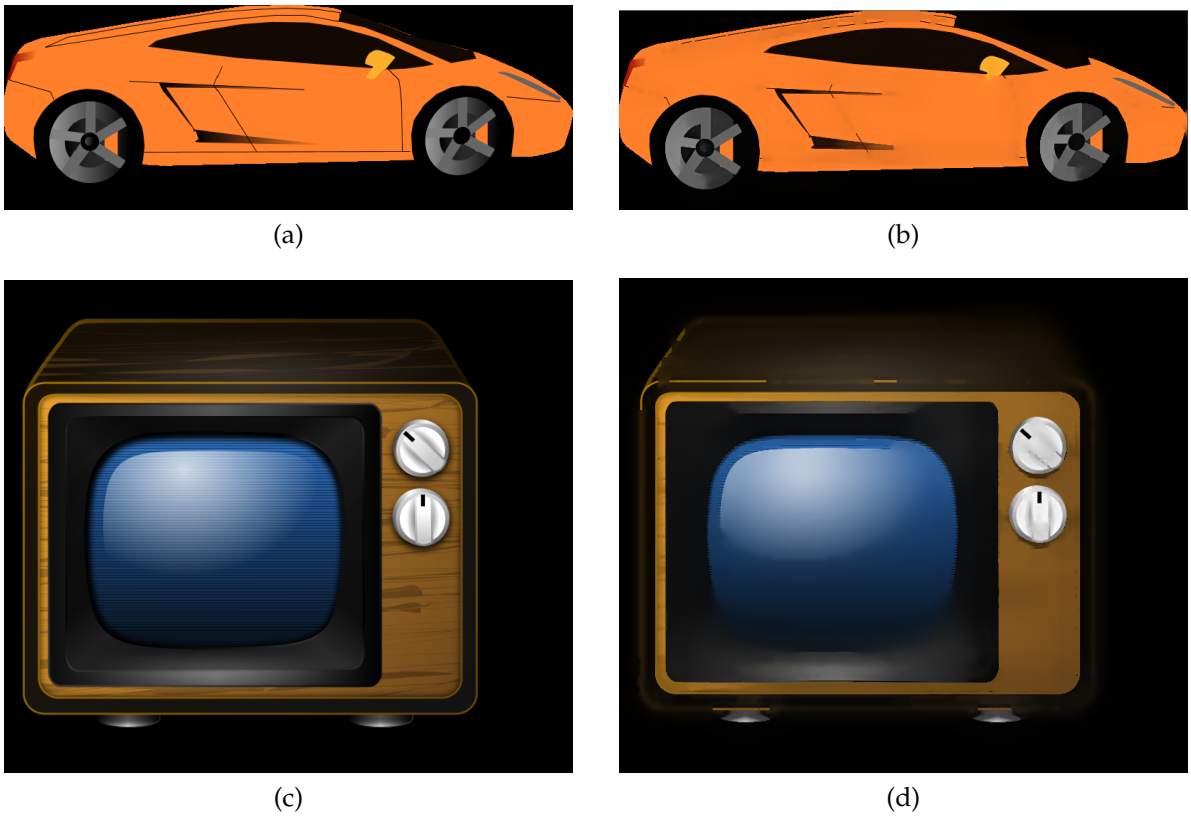
3.2.1.1 Simplificação de imagens clip art vectoriais através de imagens *raster*

Na representação mais complexa de uma imagem vectorial existem vários elementos que se podem sobrepor, o que produz informação redundante sobre pontos que não são visíveis. De maneira a remover essa informação redundante foram geradas imagens *raster* a partir das imagens vectoriais originais. O processo de simplificação passa por simplificar a imagem *raster* sem perda de elementos

importantes e então utilizar os elementos geométricos que formam a imagem para voltar a converter para o formato vectorial. Os elementos geométricos são processados para identificar os elementos mais relevantes na imagem e obter a sua representação vectorial, que é utilizada para extracção de descritores. De forma a obter os elementos geométricos, é necessário segmentar a imagem nesses elementos.

3.2.1.2 Segmentação utilizando o algoritmo Mean Shift da biblioteca OpenCV

Uma primeira abordagem à segmentação foi feita utilizando um algoritmo muitas vezes utilizado com sucesso na segmentação de imagens, o Mean Shift [11]. Apesar de ser promissor as imagens clip art produziam muitos artefactos (ver figura 3.3).



(a) (b) (c) (d)

Figura 3.3: Aplicação do algoritmo Mean Shift implementado pela biblioteca OpenCV. As figuras 3.3a e 3.3c são as imagens antes da aplicação do algoritmo. Após a aplicação do algoritmo podemos ver o resultado nas figuras 3.3b e 3.3d.

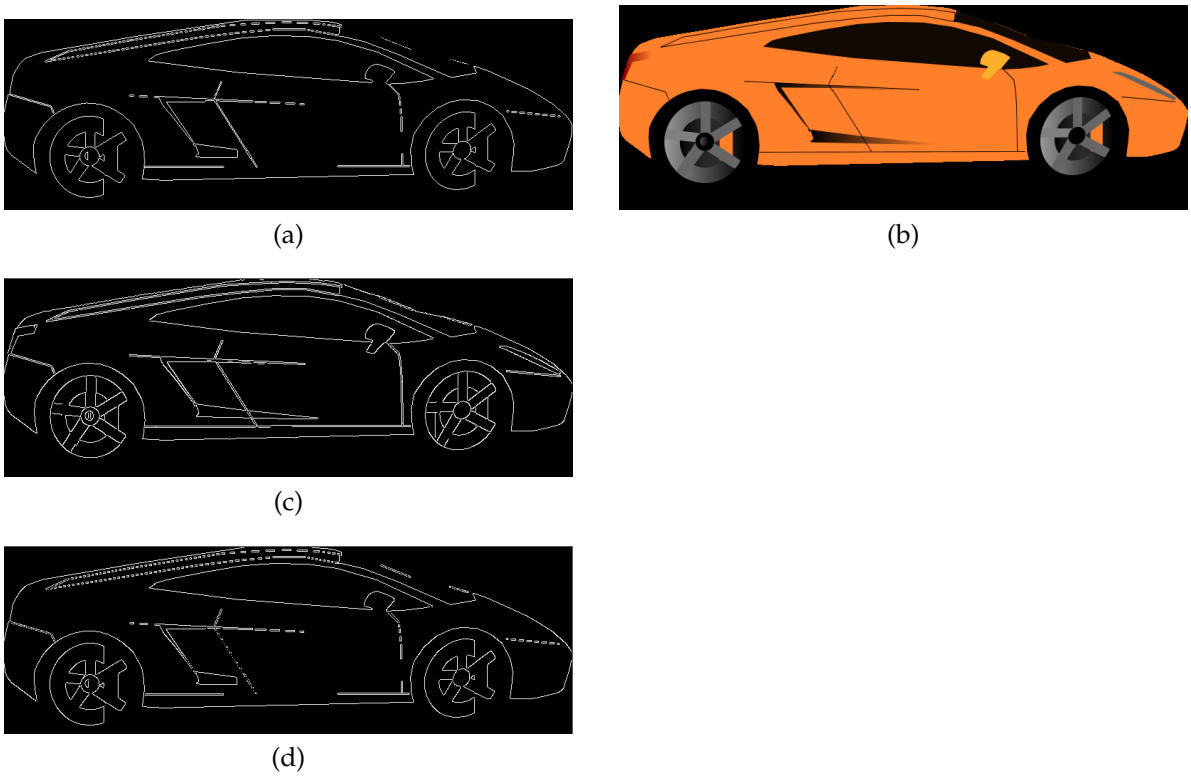
Devido a estes artefactos não havia vantagens em usar este método para extrair informação geométrica, nem uma segmentação que simplificasse a representação vectorial através dos contornos. É de notar que não foram utilizados filtros

de modo a realçar contrastes e nem outras transformações para melhorar os resultados do algoritmo. Este algoritmo está implementado na biblioteca OpenCV, no método *cvPyrMeanShiftFiltering*, e permite a alteração de vários parâmetros do algoritmo. Foram testadas várias combinações de parâmetros e nenhuma se mostrou satisfatória.

3.2.1.3 Segmentação utilizando o combinações do algoritmo Canny e filtros de *threshold*

Uma segunda aproximação foi a utilização do algoritmo de pesquisa de contornos disponível no OpenCV, *cvFindContours*. A implementação fornecida permite alterar diversos tipos de parâmetros e pode ter como entrada uma imagem de contornos ou uma imagem binária. Nesta implementação as imagens não binárias com pixels com valores superiores a 0 são consideradas como 1's, o equivalente a uma imagem binária. Foram então analisados os métodos de obter ambos os tipos de imagens. Uma maneira de obter uma imagem de contornos é utilizar o algoritmo Canny [6] para detecção de contornos, que vem implementado no método *cvCanny* do OpenCV. A criação de uma representação *bitmap* de uma imagem pode ser conseguida através de diversos meios. Foi decidido utilizar um filtro de *threshold* de modo a realçar os contornos e efectuar a criação da imagem binária. Também este filtro está implementado no OpenCV, através do método *cvThreshold*. Foram então analisados os resultados produzidos por ambos os métodos bem como a combinação de ambos, aplicando o algoritmo Canny no imagem binária criada pelo do filtro de *threshold*. Os resultados podem ser vistos na figura 3.4.

Destes resultados o que produz contornos mais definidos é a utilização do filtro de *threshold* seguida do *cvFindContours*, do OpenCV. Apesar dos contornos serem nítidos não foi produzida uma simplificação significativa dos elementos presentes na imagem. A combinação do filtro com o algoritmo Canny seguido da pesquisa de contornos foi o que produziu maior simplificação sem sacrificar os elementos essenciais da imagem, apesar de serem mantidos alguns artefactos.



(a) (b)

(c)

(d)

Figura 3.4: Extracção de contornos. Do lado direito, a figura 3.4b representa a imagem original. Do lado esquerdo na figura 3.4a está o resultado da aplicação do filtro de *threshold*, na figura 3.4c o resultado da aplicação do algoritmo Canny e na figura 3.4d o resultado de aplicar o filtro de *threshold* seguido do algoritmo Canny.

3.2.1.4 Segmentação utilizando o sistema EDISON

Decidimos utilizar outra implementação do algoritmo Mean Shift, pois esse método fornece informações referentes à cor predominante de cada mancha que são importantes, como por exemplo nas regiões de cor 3.3.2. A implementação escolhida foi o Edge Detection and Image Segmentation (EDISON) System [10]. Este sistema implementa os algoritmos definidos em [10, 11, 39] para segmentação e detecção de contornos. As imagens são segmentadas com base na cor e são criadas duas imagens. Uma com segmentação em manchas de cor e uma com contornos, como se pode ver na figura 3.5. É possível ainda definir o tamanho máximo de cada mancha e a distância a partir da qual duas cores são consideradas distintas. O EDISON apresenta todas as características que procurámos aliadas ao benefício de ter o código fonte (c++) também disponível on-line, o que nos permitiu fazer algumas alterações e extrair mais informação da imagem. A informação adicional extraída contém uma matriz que indica a que mancha pertence cada pixel da imagem, a área de cada macha e o descritor baseado na moda

da cor média de cada região da imagem produzido pelo EDISON. Na matriz que indica os pixels de cada mancha, é atribuído, a cada pixel, um identificador com número da mancha a que esse pixel pertence. O descritor de modas é utilizado para construir os descritores de regiões de cor.

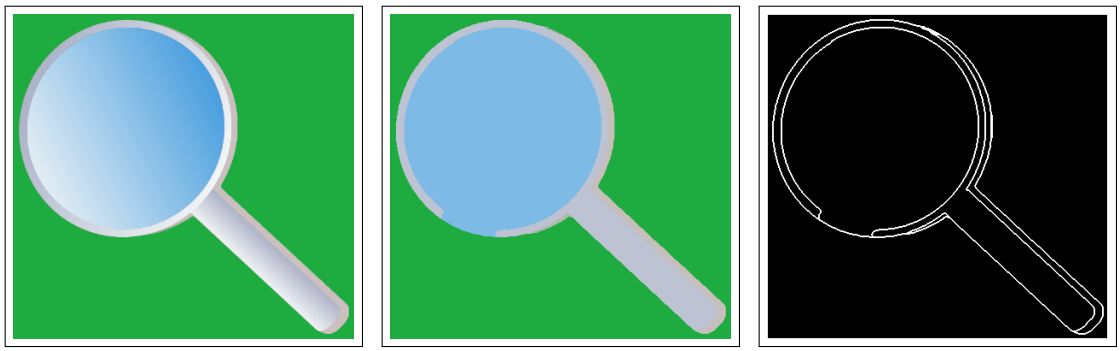
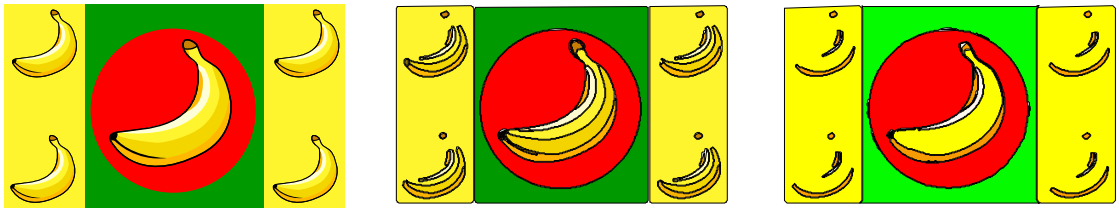


Figura 3.5: Imagem de manchas de cor e de contornos obtidas com a segmentação utilizando o sistema EDISON.

3.2.2 Conversão das imagens *raster* para imagens vectoriais e extracção dos descritores

Após gerarmos as imagens segmentadas pelas suas manchas de cor procedemos à sua conversão para um formato vectorial. Para representar as imagens vectoriais foi escolhido o formato Scalable Vector Graphics (SVG) [1]desenvolvido pelo W3C (World Wide Web Consortium). Para extrair as manchas de cor é aplicado à matriz que indica a que mancha pertence cada pixel da imagem um filtro de *threshold*. É utilizado o número de cada mancha para criar uma imagem binária para cada uma das manchas, excepto para o *background* que possui o identificador 0. Os polígonos presentes na imagem são obtidos utilizando as técnicas de detecção de contornos da biblioteca do OpenCV na imagem binária de cada mancha de cor. Cada contorno possui um determinado número de pontos que o definem. É através desses pontos e da imagem segmentada em manchas de cor que é feita a escrita dos polígonos correspondentes a cada mancha no formato SVG. Na figura 3.6 podemos ver o resultado da simplificação da imagem vectorial utilizando diferentes versões de imagens *raster*.

[2]http://www.w3.org/Graphics/SVG/



(a) (b) (c)

Figura 3.6: A figura 3.6a representa a imagem vectorial original. As figuras 3.6b e 3.6c representam a versão vectorial simplificada obtida através de uma imagem *raster* com palete JNS e de uma imagem com a palete de cores completa respectivamente.

A extracção dos descritores das imagens vectoriais, tanto as simplificadas como as imagens vectoriais originais, é feita através do sistema desenvolvido em [49]. O sistema lê imagens no formato SVG e efectua algumas simplificações aplicadas a imagens vectoriais. Depois gera um vector de descritores baseado na topologia e geometria da imagem. Na figura 3.7 são representadas as etapas do processo de extracção dos descritores de topologia e geometria. Os descritores gerados pelo sistema já se encontram normalizados e o processo de normalização pode ser consultado em [49].

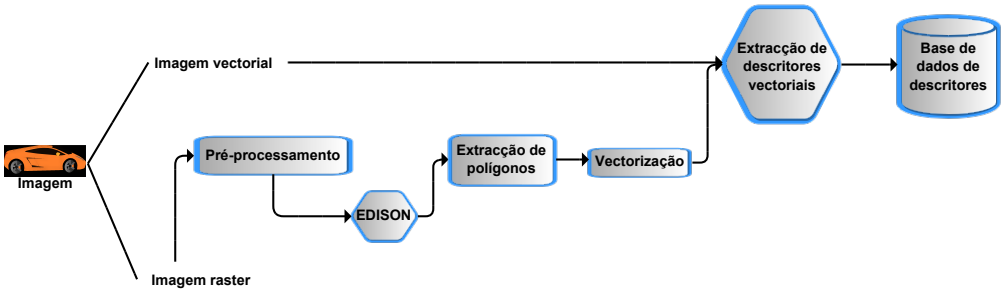


Figura 3.7: *Pipeline* de extracção de descritores de topologia e geometria de imagens vectoriais.

3.3 Extracção de descritores a partir de imagens *Raster*

Na secção anterior foram abordadas as técnicas utilizadas para transformar imagens *raster* em representações vectoriais mais simples de modo a obter descritores baseados em imagens vectoriais. Nesta secção analisamos as técnicas utilizadas

para obter descritores de imagens baseados nas características das imagens do tipo *raster*. Como foi referido na secção 2.1 os descritores de imagens *raster* podem ser baseados na cor, textura, forma e pontos de interesse. Dentro das várias técnicas de cálculo e extracção de descritores escolhemos algumas das mais utilizadas [12]:

- Momentos de cor no espaço HSV;
- Regiões de cor no espaço LUV;
- Descritores de textura obtidos através de filtros de Gabor;
- Descritor SIFT.

Com os descritores extraídos com estas técnicas poderemos testar a sua precisão quando usados individualmente e quando são usados juntamente com os outros descritores de imagens *raster* e com os descritores extraídos das imagens vectoriais. Deste modo podemos determinar a sua precisão e influência da combinação de descritores na precisão das pesquisas. A biblioteca do OpenCV tem alguns dos algoritmos de extracção destas características já implementados. Para os métodos que não estavam incluídos na biblioteca, foram implementados algoritmos para a sua extracção.

3.3.1 Descritores de momentos de cor

Este método tira partido da distribuição de cores na imagem, que pode ser caracterizada pelos seus momentos, como a média e o desvio padrão, de forma a identificar imagens. Esses valores são guardados, para cada uma das componentes de cor, por exemplo a *hue*, *saturation* e *value* para o formato HSV, como um vector de descritores [50]. Estes vectores podem ser comparados no espaço euclidiano. Desta forma duas imagens com distribuição semelhante terão uma distância pequena. No entanto duas imagens, com apenas sub-regiões semelhantes, podem não ter distribuições semelhantes, logo uma distância maior [47]. Uma aproximação é dividir a imagens em vários rectângulos de tamanhos iguais. A implementação utilizada na nossa solução [8] utiliza essa aproximação. É calculada a média e o desvio padrão, para cada componente de cor, em cada rectângulo. Cada imagem é dividida em N_{block}. O número total de blocos é obtido da seguinte forma,

$$N_{block} = factor^2, \tag{3.1}$$

onde $factor$ é um parâmetro que define o número de blocos na vertical ou na horizontal. O o número de linhas e colunas de cada bloco de uma imagem Img com altura Img_{height} e largura Img_{width} é calculado usando as formulas,

$$N_{lcount} = Img_{height}/factor, \tag{3.2}$$

$$N_{ccount} = Img_{width}/factor. \tag{3.3}$$

Obtemos a média de um bloco *n* para a componente de cor *c* com,

$$\mu_{n,c} = \frac{\sum_{i=1}^{N_{lcount}} \sum_{j=1}^{N_{ccount}} Img_{n,c}(i,j)}{N_{lcount} N_{ccount}}, \tag{3.4}$$

e o desvio padrão,

$$\sigma_{n,c} = \sqrt{\frac{\sum_{i=1}^{N_{lcount}} \sum_{j=1}^{N_{ccount}} (Img_{n,c}(i,j) - \mu_{n,c})^2}{N_{lcount} N_{ccount}}}. \tag{3.5}$$

A imagem Img passa a ser representada por um vector, $\upsilon = [\mu_{1,1}, \sigma_{1,1}, ..., \mu_{1,3}, \sigma_{1,3}, ..., \mu_{N_{block},3}, \sigma_{N_{block},3}]$. A figura 3.9 representa o diagrama das várias etapas do processo de extracção de descritores de momentos de cor.

Após alguns testes (ver tabela 7.1) decidimos utilizar $factor = 4$ ou seja 16 blocos (ver imagem 3.8).

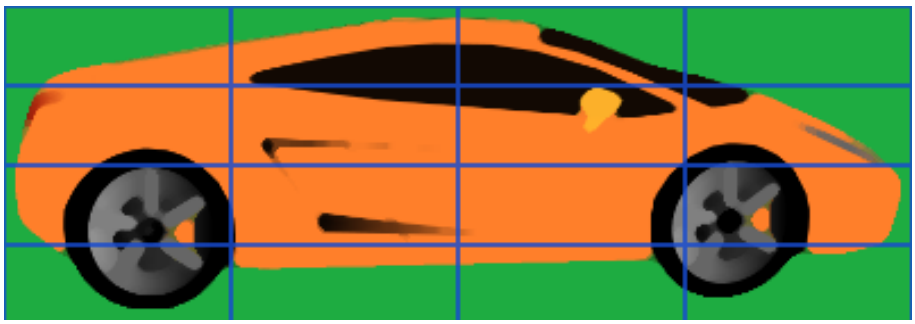


Figura 3.8: Divisão da imagem em rectângulos para o cálculo dos momentos de cor

Figura 3.9: Diagrama das etapas do processo de extracção de descritores de momentos de cor

3.3.2 Descritores de regiões de cor

Os descritores de regiões de cor fornecem informação sobre as manchas de cor existentes numa imagem e a sua posição e para os obter decidimos voltar a recorrer ao sistema EDISON. Como já foi referido o EDISON produz uma versão da imagem original segmentada pelas suas manchas de cor e informação sobre os pixels que pertencem a cada mancha. O processo de extracção das manchas de cor é semelhante ao utilizado na vectorização de imagens *raster*. Também são aplicados filtros de *thresholding* utilizando o identificador de cada mancha de cor, excepto o *background*, para produzir uma imagem binária de cada mancha. A imagem segmentada em manchas de cor é processada de modo a calcular a média μ_c e desvio padrão σ_c de cada componente de cor, o tamanho relativo da região na imagem $relsize = RegPixelsCount/ImgTotalPixels$ e a média μ_x, μ_y e desvio padrão σ_x, σ_y das coordenadas dos pixels da região. A imagem binária de cada mancha de cor é utilizada como uma máscara para processar essa mancha separadamente. O modelo de cor usado nestas operações é o espaço LUV, também usado pelo sistema EDISON. Estes valores irão ser usados para construir um vector de descritores por cada região detectada na imagem. Uma região passa a ser representada por um vector $rv_k = [\mu_l, \mu_u, \mu_v, relsize, \sigma_l, \sigma_u, \sigma_v, \mu_x, \mu_y, \sigma_x, \sigma_y]$. Na figura 3.10 pode ser visto um diagrama com as várias etapas do processo de extracção deste descritor.

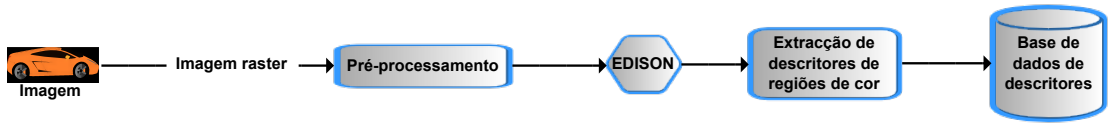


Figura 3.10: Diagrama das etapas do processo de extracção dos descritores de regiões de cor.

3.3.3 Descritores de textura utilizando filtros de Gabor

Decidimos utilizar o banco de filtros de Gabor como um método de captura de texturas. O banco de filtros de Gabor permite analisar imagens com várias escalas e rotações. Para tal utilizámos uma implementação previamente desenvolvida [8]. Esta implementação calcula os filtros de Gabor utilizando 6 orientações θ e 4 escalas s, o que faz com que sejam utilizados 24 filtros do banco de filtros de Gabor. Depois utiliza uma implementação do OpenCV para aplicar o filtro (ver figura 3.11). É utilizada a média $\mu_{s,\theta}$ e o desvio padrão $\sigma_{s,\theta}$ do valor absoluto da imagem filtrada por cada um dos filtros como vector de descritor. Uma imagem Img passa a ser representada por um vector $\upsilon = [\mu_{1,1}, \sigma_{1,1}, ..., \mu_{s,\theta}, \sigma_{s,\theta}]$.

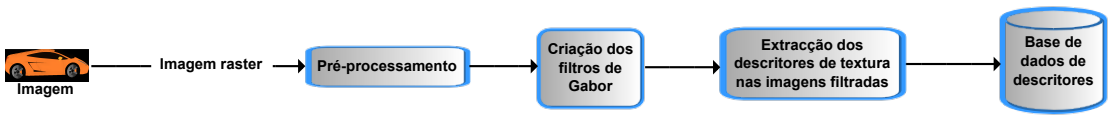


Figura 3.11: Diagrama das etapas do processo de extracção de descritores de textura utilizando o banco de filtros de Gabor.

3.3.4 Descritores SIFT

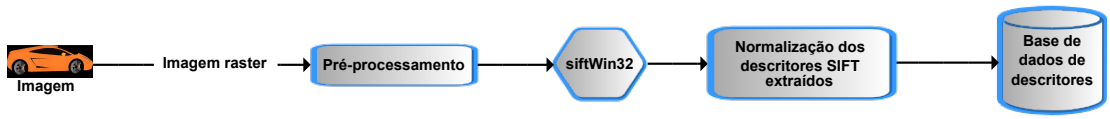


Figura 3.12: Diagrama das etapas do processo de extracção de descritores de textura utilizando o banco de filtros de Gabor.

Como foi discutido na secção 2.1.3, o SIFT desenvolvido por David Lowe [37] é um método capaz de produzir identificadores de várias partes da imagem e criar descritores em pontos representativos dessas áreas. Os passos de identificação de pontos de interesse garantem a invariância à luminosidade, escala e rotação. No último passo de criação dos descritores, descrito anteriormente, é referido que é agrupada a informação em redor da área onde é detectado o ponto de interesse. Essa área é definida numa matriz de 16×16 pixels e posteriormente dividida em blocos de 4x4 pixels. Para cada um desses blocos é calculado um

histograma com 8 orientações. Desta forma é criado um vector descritor com 128 valores ($4 \times 4 \times 8$), um para cada uma das orientações em cada um dos blocos. Cada imagem Img passa a ser representada por um conjunto de k descritores, sendo k o número de pontos de interesse detectados. Na figura 3.12, podemos ver as diferentes etapas no processo de extracção de descritores SIFT. Foi representada a aplicação distribuída por David Lowe (siftwin32) pois é usada para a detecção dos pontos de interesses e criação dos vectores de descritores.

4

Sistema de recuperação de Clip Arts

Neste capitulo são analisados os métodos e tecnologias utilizados de forma a realizar pesquisas e o *matching* de imagens. Como foi descrito no final do capítulo anterior, cada imagem pode ser representada por um conjunto de vectores de descritores. Estes descritores representam diversas características presentes nas imagens. No capítulo anterior analisamos os métodos de extracção de características e construção dos vectores de descritores. Aqui iremos discutir numa primeira parte as técnicas para comparar duas imagens e estabelecer um grau de similaridade entre elas. De seguida, os métodos utilizados de forma normalizar os elementos pertencentes à base de dados. No final, iremos analisar o método proposto pela nossa solução.

Os seguintes símbolos são usados neste capítulo para descrever alguns conceitos utilizados:

- DB - colecção de imagens armazenadas;
- DB_i - *i-nésima* imagem na colecção;
- Q - imagem utilizada como *query*;
- $Norm(Elem)$ - função de normalização de um elemento num vector de descritores;
- BoF - simboliza o vector de descritores *Bag of Features*;
- v_m - vector de descritores de momentos de cor;

- v_g - vector de descritores de textura extraídos com os filtros de Gabor;
- v_s - vector de descritores SIFT;
- v_v - vector de descritores de topologia e geometria;
- $dEu(v_1, v_2)$ - distância Euclidiana entre os vectores de descritores v_1 e v_2;
- $dMan(v_1, v_2)$ - distância de Manhattan entre os vectores de descritores v_1 e v_2;
- $dCos(v_1, v_2)$ - distância de cosenos entre os vectores de descritores v_1 e v_2;
- $dist(Q, Img)$ - função para o cálculo da distância entre a imagem de pesquisa Q e a imagem $Img \in DB$;

4.1 Arquitectura

Como já foi referido a solução foi implementada em C++, recorrendo à biblioteca *open source* OpenCV para tratamento e manipulação de imagens. A implementação foi divida em duas secções, um módulo que disponibiliza as funções de extracção de características e criação dos diversos vectores de descritores e um módulo que disponibiliza a as funções de pesquisa numa base de dados e métodos de avaliação de desempenho das pesquisas.

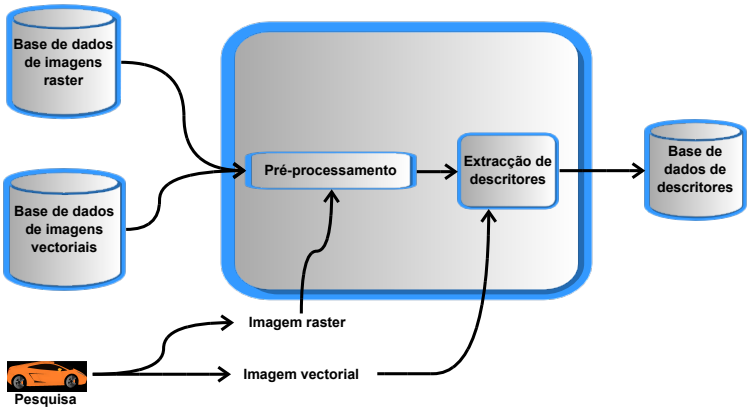


Figura 4.1: Módulo de extracção de características.

O módulo de extracção de características (ver figura 4.1) fornece diversos mecanismos para extracção de descritores e criação de vários vectores de descritores, referidos no capítulo 3. Cada imagem é processada por um *pipeline* de algoritmos semelhante. É efectuado o pré-processamento de imagem, que inclui a criação da imagem com palete de 12 cores em no HSV, depois é aplicado um dos algoritmos

de extracção de descritores e finalmente é guardado em memória permanente o vector de descritores da característica extraída.

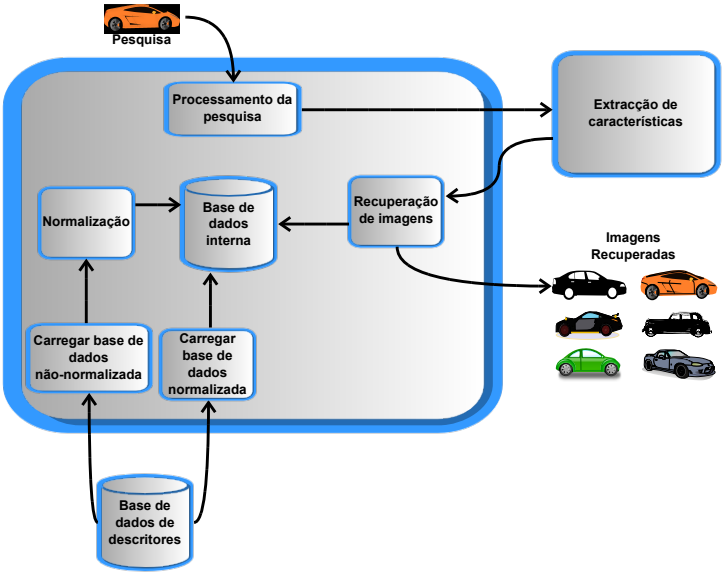


Figura 4.2: Módulo de recuperação de imagens clip art.

O módulo de recuperação de clip arts (ver figura 4.2), que inclui o nosso sistema de recuperação, carrega a base de dados dos vários vectores de descritores para a memória do sistema e então permite a execução de pesquisas, utilizando uma imagem para uma *query-by-example*. Os vectores de descritores podem ser carregados de duas formas, no estado normalizado e no estado não normalizado. Quando carregados no estado não normalizado é aplicado um algoritmo de normalização que varia dependendo do tipo de vector de descritores a ser normalizado. Uma imagem utilizada como interrogação utiliza o *pipeline* de algoritmos para extracção de características utilizado no módulo de extracção. Os vectores de descritores extraídos da interrogação são então comparados com os vectores de descritores na bases de dados e é retornada uma lista de clip arts recuperados ordenados pela sua semelhança com a interrogação.

4.2 Normalização de descritores

Para reduzir erros estatísticos de modo a que um elemento de um vector de descritores não possa influenciar o conjunto de dados, e posteriormente as imagens recuperadas, os vectores têm de ser normalizados para que as características presentes nos vectores de descritores possam ser comparadas.

Para cada uma das técnicas estudadas no capítulo anterior é aplicada uma

operação de normalização aos vectores de descritores de todas as imagens na base de dados (ver figura 4.3). Como as várias características possuem propriedades diferentes o seu processo de normalização também difere. De seguida descrevemos quais as operações de normalização aplicadas a cada um dos tipos de descritores que foram estudados.

Os descritores de topologia são uma excepção nesta etapa de normalização. Na secção 3.2.2 analisamos estes descritores e referimos que os descritores são automaticamente normalizados pelo sistema utilizado para extracção das características. Este processo é descrito em [49].

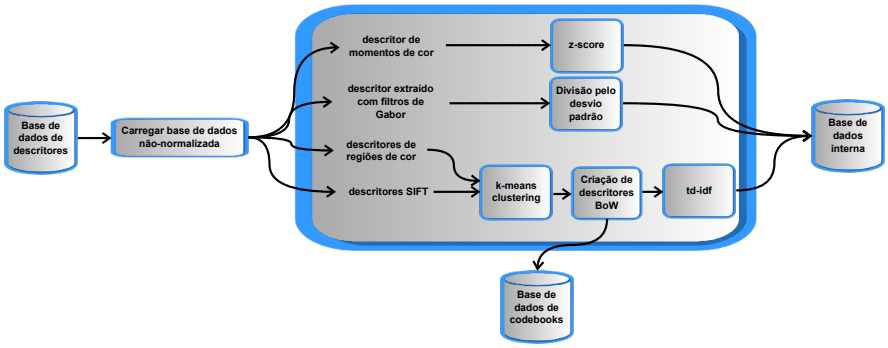


Figura 4.3: Diagrama do processo de normalização dos descritores.

O processo de normalização dos descritores de **momentos de cor** é feito calculando os *z-scores* de cada elemento do vector de descritores. Agrupamos todos os vectores de descritores em DB numa matriz M. Cada linha M_i contém o vector de descritores de momentos de cor da imagem Im_i e M_j simboliza todos os descritores na coluna j. O *z-score* de um elemento é calculado subtraindo a média $\mu(M_j)$ a esse elemento e dividindo o resultados pelo desvio padrão $\sigma(M_j)$. A fórmula utilizada para obter o *z-score* de cada elemento do vector de descritor é a seguinte,

$$Norm(M(i,j)) = (M(i,j) - \mu(M_j))/\sigma(M_j). \tag{4.1}$$

Os valores $\mu(M_j)$ e $\sigma(M_j)$ de cada M_j são guardados de forma a normalizar os descritores de uma imagem de pesquisa.

A normalização aplicada aos descritores de **textura utilizando o banco de filtros de Gabor** é mais simples. É utilizado o mesmo processo que nos momentos de cor, todos os vectores em DB são agrupados numa matriz M. Mas neste caso cada elemento será normalizado dividindo o seu valor por $\sigma(M_j)$. Ou seja em analogia com 4.1, a fórmula para normalizar um elemento do vector de descritores de textura usando os filtros de gabor é a seguinte,

$$Norm(M(i,j)) = M(i,j)/\sigma(M_j). \tag{4.2}$$

Tal como na normalização dos momentos de cor, são guardados os valores de desvio padrão como parâmetros de normalização de modo a que posteriormente possam ser aplicados para normalizar as imagens de pesquisa.

Os restantes vectores de descritores, as **regiões de cor** e os **SIFT**, antes da sua normalização são convertidos em descritores to tipo *bag-of-features* (BoF). O modelo de descritores *bag-of-words* é usado em pesquisas em documentos com texto, mas vários sistemas [32, 35, 57] já o adaptaram para pesquisa de imagens, utilizando características de regiões ou *keypoints*, como 'palavras visuais'. Se pensarmos nos descritores de cada região, para as regiões de cor, e em cada descritor SIFT como 'Palavras', então com este paradigma podemos encarar as imagens como 'documentos'. Fizemos uma analogia com processo usado em pesquisas textuais e consideramos os vectores de descritores de regiões de cor e SIFT, as palavras, como *codewords*, e o dicionário com o conjunto de todas as palavras como um *codebook*.

O primeiro passo neste processo é a criação do *codebook*. O número de descritores obtido para cada imagem através da extracção dos descritores de regiões de cores e dos SIFT é elevado, em média aproximadamente 47, para as regiões de cor e 253 para os SIFT, para a colecção de imagens descrita na secção 5.2. O número total de *codewords* irá aumentar quanto maior for a colecção de imagens. Logo existe a necessidade de um método para representar os conjuntos de vários descritores semelhantes, e utilizar essa representação como *codewords*. Foi usado o algoritmo de *k-means clustering* para agrupar os conjuntos de descritores semelhantes, e assim o conjunto de todos os centroides detectados pelo algoritmo irá constituir o nosso *codebook*.

No segundo passo as imagens passam a ser representadas por um novo vector de descritores de tamanho $CBsize$, o número de *codewords* diferentes, no *codebook*. O descritor *BoF* é um vector com a seguinte forma,

$$v = [N_1, N_2, ..., N_n, .., n_{CBsize}], \tag{4.3}$$

em que cada elemento representa o número de vezes que a *codeword* N_n aparece na imagem.

De seguida os descritores de *BoF* têm de ser normalizados. Então decidimos utilizar um tipo de normalização normalmente aplicado a pesquisas de documentos com texto, o *tf-idf* (*term frequency - inverse document frequency*). O peso

tf-idf representa a importância de uma palavra num documento, numa colecção de documentos. Este processo pode ser dividido em duas partes o cálculo do *tf* e o cálculo do *idf*.

Podemos calcular a parcela *tf* para o elemento n do vector de descritores *BoF* da seguinte forma,

$$tf(BoF, n) = BoF(n) / \sum_{i=0}^{i=CBsize} BoF(i). \tag{4.4}$$

A divisão pelo somatório dos elementos no descritor é utilizada para normalizar o descritor para que não haja influência do tamanho do documento. A parcela *idf* é calculada utilizando todos os descritores na base de dados pois refere-se à raridade de uma *codeword* em DB. Sendo $codeword_n$ a enésima *codeword* no *codebook* temos a fórmula,

$$idf(DB, codeword_n) = \log \frac{|DB|}{|BoF \in DB : codeword_n \in BoF|}. \tag{4.5}$$

• $|DB|$ representa o número de imagens na base de dados;

• e $|BoF \in DB : codeword_n \in BoF|$ representa o número de imagens que a $codeword_n$ existe.

Assim podemos obter o *tf-idf* para cada elemento de um descritor *BoF* da seguinte forma,

$$tf - idf(DB, BoF, n) = tf(BoF, n) * idf(DB, codeword_n). \tag{4.6}$$

De modo a normalizar uma pesquisa são guardados os centroides obtidos através do processo de *k-means clustering*, o *codebook*, e o número de imagens em que cada *codeword* aparece. Estes dados permitem-nos criar o descritor de *BoF* usando o *codebook* e depois obter a normalização td-idf. Na normalização de uma pesquisa a parcela 4.4 é obtida directamente. Para calcular 4.5 é incrementado o número de imagens ($|DB| + 1$) e também os contadores de imagens que contêm uma dada *codeword*, para cada *codeword* detectada na imagem de pesquisa.

4.3 Recuperação de imagens

De modo a que duas imagens possam ser comparadas utilizando as características extraídas e normalizadas, tivemos de definir uma função que nos desse o grau de similaridade entre duas imagens. Quando comparámos duas imagens utilizando uma função de distância e quanto menor for a distância obtida pela função

mais semelhantes são as duas imagens. As várias características possuem propriedades diferentes, então foram definidas algumas funções de distância para os diversos casos implementados. As funções de distância entre imagens foram separadas em dois conjuntos, comparação de imagens usando apenas uma das características e comparação de imagens usando múltiplas características.

As imagens são posteriormente ordenadas pela sua distância à imagem de pesquisa. As lista de imagens recuperadas é obtida retornando as N primeiras imagens da lista de imagens ordenada.

4.3.1 Recuperação utilizando uma única característica

No caso dos descritores de textura extraídos utilizando os filtros de Gabor, foi utilizada a distância Euclidiana [12] seguida de uma normalização das distâncias obtidas (importante na combinação de várias características). A função de similaridade entre uma imagem Q e uma qualquer imagem Img na colecção é a seguinte,

$$dist(Q, Img) = dEu(Q_{v_g}, Img_{v_g})/maxDistance, \tag{4.7}$$

em que $maxDistance$ simboliza a maior distância encontrada entre a imagem Q e todas as imagens em DB, ou seja a imagem que tem a maior distância Euclidiana à imagem Q é utilizada para normalizar as distâncias.

Para a comparação de imagens através dos seus descritores de momentos de cor é utilizada a uma função equivalente à referida anteriormente, mas utilizando a distância de Manhattan [12] para calcular a distância entre os vectores de descritores. Ou seja o valor $maxDistance$ nesta função refere-se à distância de Manhattan entre a imagem Q e a imagem mais distante de si em DB. A função é apresentada de seguida,

$$dist(Q, Img) = dMan(Q_{v_m}, Img_{v_m})/maxDistance. \tag{4.8}$$

A comparação de imagens que utilizam os descritores *BoF*, baseados nas regiões de cor e SIFT utiliza a distância de cosenos. A função de distância é a seguinte,

$$dist(Q, Img) = 1 - dCos(Q_{BoF}, Img_{BoF}). \tag{4.9}$$

Para comparar imagens utilizando as suas características extraídas da sua

forma vectorial, como descrito em 3.2, também é utilizada a distância Euclidiana e por sua vez a função de similaridade utilizada é a apresentada em 4.7, mas em vez de utilizar o vector v_g, utiliza v_v.

4.3.2 Recuperação utilizando várias características

A utilização de apenas uma característica extraída de uma imagem tem limitações pois as características não conseguem isoladamente separar as diversas classes no espaço de características. Para tentar minimizar estas desvantagens foram implementadas as combinações de várias características que obtiveram melhores resultados. Dividimos as características extraídas em 3 classes:

1. Descritores de cores nas imagens;
2. Descritores de textura ou pontos de interesse;
3. Descritores de características extraídas de imagens em formato vectorial

As primeiras duas classes incluem-se nas características extraídas de imagens em formato *raster*. Quando são utilizados descritores de imagens *raster* e descritores de imagens vectoriais utilizando as imagens vectoriais originais ao mesmo tempo é necessário existir uma cópia da imagem em ambos os formatos, tanto na base de dados como para uma pesquisa. Quando são utilizadas as imagens vectoriais simplificadas apenas é necessária a imagem *raster*, pois a versão vectorial simplificada é obtida através dessa imagem.

Na comparação de imagens com vários descritores a função de distância é mais complexa e consiste na combinação de funções apresentadas na secção anterior (4.3.1). O cálculo da distância pode ser dividido em dois passos:

1. O cálculo da distância combinada (ver equação 4.10);
2. A normalização do valor da distância devolvido (ver equação 4.11).

É utilizado $maxDistComb$, o valor máximo de distância combinada, obtido pela equação 4.10, entre Q e todas as $Img \in DB$, para normalizar a distância combinada. Outro factor importante são as n características utilizadas em conjunto em que o sistema implementa combinações de 2 ou 3 características. No caso do uso da função de distância definida na equação 4.9, o resultado é normalizado dividindo o resultado da função pela distância entre Q e a imagem $Img \in DB$ mais distante, à semelhança das equações 4.8 e 4.7.

$$distComb(Q, Img) = \sum_{i=0}^{i=n} sim_i(Q, Img) * \frac{1}{n}. \tag{4.10}$$

$$normDistComb(Q, Img) = distComb * \frac{1}{maxDistComb}. \tag{4.11}$$

Foram implementadas as seguintes técnicas que utilizam várias características:

- Combinação de descritores de **momentos de cor** e descritores de **textura extraídos com o banco de filtros de Gabor**. Na equação 4.10 são utilizadas as funções de distância definidas pelas equações 4.8 e 4.7;
- Combinação de descritores de **momentos de cor** e descritores de *BoF* usando os descritores **SIFT**. Na equação 4.10 são utilizadas as funções de distância definidas pelas equações 4.8 e 4.9.
- Combinação de descritores de **momentos de cor** e descritores de **topologia e geometria**. Na equação 4.10 são utilizadas as funções de distância definidas nas equações 4.8 e 4.7. Na equação 4.7 é utilizado o descritor de topologia e geometria;
- Combinação de descritores de *BoF* usando os descritores **regiões de cor** e descritores de **textura extraídos com o banco de filtros de Gabor**. Na equação 4.10 são utilizadas as funções de distância definidas nas equações 4.9 e 4.9;
- Combinação de descritores de **momentos de cor**, descritores de *BoF* usando os descritores **SIFT** e descritores de **topologia e geometria**. Na equação 4.10 são utilizadas as funções de distância definidas pelas equações 4.8, 4.9 e 4.7 respectivamente.
- Combinação de descritores de **momentos de cor**, descritores de **textura extraídos com o banco de filtros de Gabor** e descritores de **topologia e geometria**. Na equação 4.10 são utilizadas das funções de distância definidas pelas equações 4.8 e 4.7. A função da distância Euclidiana, definida na equação 4.7, é utilizada duas vezes, uma para os descritores de textura e outra para os descritores de topologia e geometria.

5

Resultados experimentais

Neste capítulo analisamos os aspectos mais importantes dos testes realizados de modo a avaliar os algoritmos implementados. São descritas as base de dados utilizadas para a realização dos testes bem como alguns problemas com que nos deparamos no decorrer dos mesmos. Os algoritmos foram avaliados de duas formas: a primeira consistiu na utilização de uma base de dados controlada de 100 imagens, divididas por 10 categorias e a segunda teve como objectivo verificar o desempenho do algoritmo numa base de dados de grandes dimensões. Esta segunda colecção incluí aproximadamente 13000 imagens, divididas em 11 categorias. São apresentados os resultados obtidos nos testes e é feita uma análise sobre os esses mesmos resultados.

5.1 Métodos de avaliação dos resultados das pesquisas e dos algoritmos

O desempenho dos algoritmos foi avaliado segundo o número de imagens relevantes e não relevantes que são recuperadas e também segundo a posição em que se encontram. Para avaliar sistemas de recuperação de informação são utilizadas medidas como *precision* e *recall*. Estas medidas avaliam a qualidade da informação retornada, a *precision* o rácio entre os resultados correctos e incorrectos, e o *recall* a completude dos resultados correctos retornados, ou seja quantos resultados correctos foram retornados do total de resultados correctos possíveis. Estas

medidas podem ser calculadas segundo as equações 5.1 e 5.2,

$$precision = \frac{|resultados\ relevantes|}{|resultados\ relevantes \cup resultados\ irrelevantes|}, \tag{5.1}$$

$$precision = \frac{|resultados\ relevantes|}{|total\ de\ imagens\ relevantes|}. \tag{5.2}$$

Para termos uma avaliação onde as posições das imagens nos resultados seja relevantes, devemos referir outra duas medidas, a *precision at n* ($P@N$) e a *Average Precision* ($AvgP$). A $P@N$ provém directamente do conceito de *precision* e é simplesmente o valor de *precision* contabilizando apenas os resultados até à posição N. A $AvgP$ combina os conceitos de *precision*, *relevance ranking* e *recall* e pode ser calculada usando a fórmula 5.3, onde $P@(n)$ é uma função de que retorna a $P@N$ para $N = n$. Img_r simboliza a imagem retornada na posição r e $rel(I)$ é uma função que retorna 1 ou 0, conforme I é uma imagem relevante a para a pesquisa ou não respectivamente.

$$AvgP = \frac{\sum_{r=1}^{n} [P@(r) \times rel(Img_r)]}{nmero\ total\ de\ imagens\ relevantes}. \tag{5.3}$$

Para avaliarmos o desempenho dos algoritmos analisamos várias pesquisas e utilizámos o *Mean Average Precision* (MAP), que não é nada mais que o valor médio das $AvgP$ das pesquisas realizadas. Calculamos o MAP para avaliar o desempenho dos algoritmos, mas também o desempenho dos algoritmos para uma categoria de imagens.

No caso da base de dados de grandes dimensões utilizámos a $R - Precision$. Semelhante ao conceito de $P@N$, a $R - Precision$ ($R - P$) não calcula a *precision* até a uma dada posição, e sim até serem encontrados n imagens relevantes. De modo a avaliar o desempenhos dos algoritmos também foi calculada a média de $R - P$ de todas as pesquisas e de cada categoria de imagens.

5.2 Análise experimental com base de dados controlada

De modo a avaliar o desempenho dos algoritmos implementados foram realizadas várias pesquisas utilizando o método de pesquisa de *query-by-example*, numa base de dados controlada em que todas as imagens foram previamente classificadas manualmente para permitir a comparação com os resultados dos algoritmos.

A base de dados utilizada contém 10 categorias: *Bulbs*, *Car*, *Fish*, *Flower*, *Hammer*, *Magnifying Glass*, *Phone*, *Sword*, *Tree* e *TV*. Cada categoria inclui 10 imagens, retiradas da biblioteca OpenClipart, um total de 100 imagens na base de dados. Foram feitas pesquisas com cada uma das imagens na base de dados, num total de 100 pesquisas.

Um dos parâmetros que foi estudado em todos os algoritmos foi a paleta de cores, como foi referido na secção 3.1. Utilizamos descritores extraídos de imagens com as suas cores completas (*AC*) e imagens em que foi feito um *clustering* das cores para apenas 12 cores no espaço HSV (*12C*). Os descritores de textura e pontos de interesse utilizam imagens em tons de cinza. Decidimos utilizar as diferentes paletas com estes algoritmos também pois a redução de cores altera certas zonas das imagens, fundindo até algumas como se pode ver na figura 3.2.

5.2.1 Resultados dos momentos de cor

Foram estudados dois parâmetros na extracção dos descritores de momentos de cor. O factor N (fact$N$) que divide a imagem em $N \times N$ rectângulos e a utilização (CM) ou não (SM) de uma máscara que remove o *background* do cálculos dos momentos de cor.

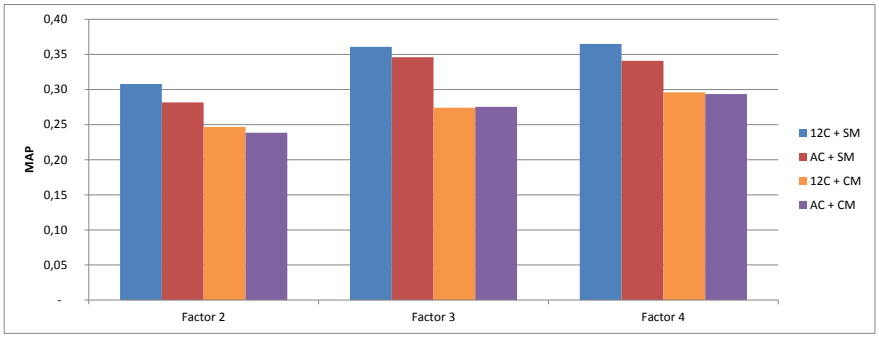


Figura 5.1: Gráfico dos *Mean Average Precision* da pesquisa usando descritores de momentos de cor.

Como pode ser visto na figura 5.1 o uso da máscara produziu piores resultados do que a inclusão do *background* influenciando os descritores. Os momentos de cor baseiam-se na média e nas mudanças de cor presentes na imagem. O uso da paleta de 12 cores aumentou o *matching* entre imagens com cores semelhantes, pois muitas áreas com cores semelhantes passam a ter a mesma cor. A utilização deste parâmetro pode produzir alguns falsos positivos nos resultados, devido a uma das desvantagens dos momentos de cor, o facto de não ser discriminativo

quanto à forma das imagens, a não ser de um modo parcial no desvio padrão. Os testes realizados sobre o conjunto de 100 imagens produziram (ver tabelas 5.1 e 7.3) resultados variados nas diversas categorias. Das várias categorias a que obteve melhores resultados, com quase todos os parâmetros utilizados, foi a categoria *Sword*, que obteve precisão média entre 25,26% e 64% no melhor caso. Esta categoria serviu para demonstrar tanto as vantagens como as desvantagens da utilização deste descritor. Nesta categoria todas as imagens têm pouca largura, o que produz rectângulos mais finos e com menos pixels. Por esta razão, conseguimos obter uma boa precisão nos resultados. Quando existem mais variações de cores entre imagens da mesma categoria são obtidos resultados mais fracos, como na categoria *Fish* em que a maioria das imagens têm formatos semelhantes mas cores variadas, provocando uma dispersão dos resultados no *ranking* das imagens. O uso da paleta de 12 cores melhorou significativamente os resultados desta categoria. Outro factor que influencia também os resultados dos momentos de cor é a posição das imagens. Os momentos de cor não são totalmente invariantes quanto à posição, a extracção dos descritores é feita numa grelha de $N \times N$ independentemente dos objectos que existem em cada parcela. Este foi o descritor que produziu melhores resultados obtendo valores de MAP perto dos 37%. O uso da paleta de 12 cores produziu um aumento de aproximadamente 2% no MAP. A divisão da imagem em 9 (factor 3) e 16 (factor 4) rectângulos produziram resultados semelhantes, com uma diferença menor que 1%. No entanto, o factor 3 apenas produziu melhores resultados em 4 categorias enquanto o factor 4 produziu melhores resultados em 6. A decisão de manter o factor 4 na combinação de descritores foi feita descartando o tempo de pesquisa, pois o desempenho temporal não é considerado no âmbito desta dissertação (ver tabela 7.2).

5.2.2 Resultados das regiões de cor

O primeiro passo foi estabelecer os parâmetros a utilizar pelo sistema EDISON para a captura das manchas de cor. No caso de imagens *12C* os parâmetros do EDISON têm menos impacto, visto que as variações de cor são mínimas pois as manchas de cor já estão formadas e apenas são identificadas. Nas restantes imagens as variações de parâmetros produziram apenas pequenas mudanças em relação aos valores pré-definidos que já tinham sido usados noutros testes que efectuamos. Por isso, decidimos como parâmetros para a extracção das manchas de cor, *spatial bandwidth* $= 10$, *range bandwidth* $= 8$ e *minimum area* $= 20$. Outro parâmetro utilizado na criação dos descritores de *BoF* foi o numero de *clusters k*,

que determina o número de *codewords* no *codebook*, usado no algoritmo *k-means clustering*.

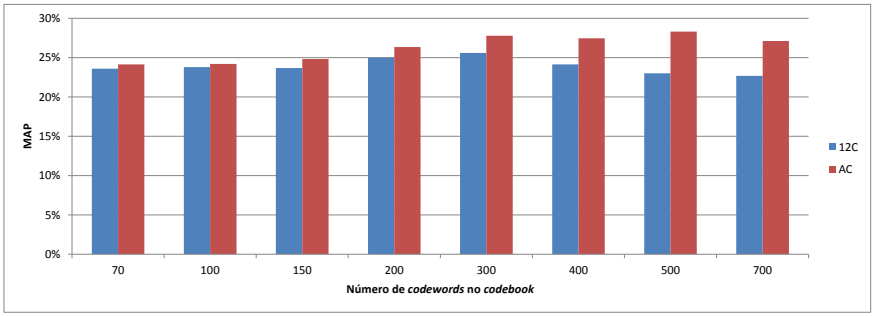


Figura 5.2: Gráfico dos *Mean Average Precision* da pesquisa usando descritores de regiões de cor.

Em grande parte dos resultados as imagens com todo o espaço de cores produziram melhores resultados. Isto deve-se ao facto de uma imagem poder produzir mais regiões com mais informação sobre elas. O número médio de descritores foi de 622 para as imagem *AC* e 253 para as imagens *12C*. Os resultados das pesquisas foram variados, mas pudemos verificar que o aumento do número de *codewords* acima de 500 produz resultados menos favoráveis. Quantas mais *codewords* temos, diminui a probabilidade de duas regiões de cor serem reconhecidas como a mesma *codeword*. Os melhores resultados foram obtidos para 300 e 500 *codewords* com um MAP de 27,8% e 28,3% (ver a figura 5.2 e a tabela 7.7). As categorias que tiveram melhor desempenhos foram as categorias *Phone* e *Sword*. A categoria *Phone* obteve MAP entre 45,7% e 54% para 100 e 200 *codewords*. A categoria *Sword* obteve entre 27% e 40,6% para 70 e 500 *codewords*. Podemos ver na tabela 5.1 que comparativamente com os momentos de cor, o outro algoritmo que extrai características referentes à cor, foram conseguidos melhores resultados nas categorias *Fish* e *TV*, mas os descritores de momentos de cor têm melhores resultados no seu conjunto.

5.2.3 Resultados da textura utilizando descritores extraídos com o banco de filtros de Gabor

Os descritores de imagens criados usando os filtros de Gabor foram a primeira técnica de identificação de texturas utilizada neste trabalho. Como parâmetros para os filtros de Gabor foram usada 4 escalas e 6 orientações como descrito na secção 3.3.3, o que corresponde a 24 filtros do banco de filtros de Gabor.

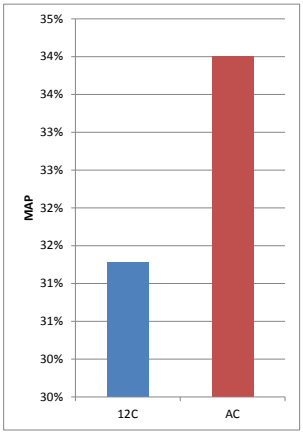


Figura 5.3: Gráfico dos *Mean Average Precision* da pesquisa usando descritores extraídos com os filtros de Gabor.

Os dois algoritmos tiveram uma diferença de 3% (ver figura 5.3). Sendo um algoritmo de reconhecimento de texturas a redução do número de cores, que provocou a fusão de certas áreas das imagens com cores semelhantes, prejudicou o desempenho. Como se pode ver nas figuras 3.2 a figura 3.2c perdeu informação quanto aos contornos. O mesmo acontece com outras imagens reduzindo a informação extraída com os filtros de Gabor. Quando são utilizadas todas as cores os filtros têm um desempenho superior com um MAP de 34%, muito próximo dos resultados obtidos pelos momentos de cor. Com estes descritores obtivemos alguns dos melhores resultados dentro da categoria *Car*, um MAP de aproximadamente 57%, obtendo também bons resultados em categorias que já tinham tidos bons resultados, as categoria *Phone*, *TV* e *Sword*, todas com MAP acima de 40% como podemos ver na tabela 5.1.

5.2.4 Resultados dos descritores SIFT

Tal como nos descritores de regiões de cor, os descritores SIFT são convertidos para descritores *BoF*. Nos testes com estes descritores o parâmetro utilizado foi novamente o número de *codewords* que integram o *codebook*.

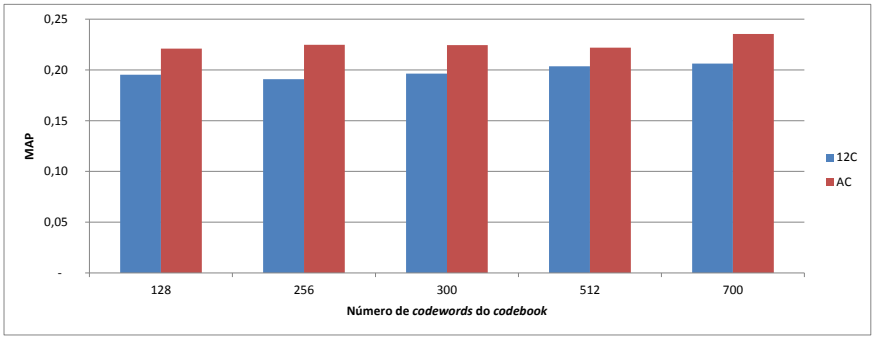


Figura 5.4: Gráfico dos *Mean Average Precision* da pesquisa usando descritores SIFT.

Os pontos de interesse detectados pelo algoritmo SIFT, tal como os filtros de Gabor, identificam a textura dos objectos numa imagem, mas através de pontos relevantes na imagem. Verificou-se o mesmo comportamento, que já se tinha observado com os filtros de Gabor, quanto ao número de cores utilizado. A redução da paleta de cores modifica a estrutura das imagem. Alguns pontos que ajudam a identificar uma imagem são removidos. Estas alterações mantêm-se na imagem em tons de cinzento criada antes de ser aplicado o algoritmo de detecção de pontos de interesse. Com a paleta de cores completa foram obtidos melhores resultados (ver figura 5.4). No entanto, os descritores BoF dos descritores SIFT mostraram um desempenho pior que os descritores de textura extraídos com o banco de filtros de Gabor (ver tabela 5.1).

5.2.5 Resultados dos descritores de topologia e geometria

Um dos objectivos desta dissertação foi um estudo sobre o desempenho dos descritores vectoriais desenvolvidos em [49] e se poderiam ser obtidos melhores resultados utilizando uma simplificação vectorial obtida através de uma imagem *raster*. É utilizada a mesma colecção de imagens que nos restantes testes recorrendo a representações vectoriais dessas mesmas imagens. Testámos o desempenho dos algoritmos de pesquisa utilizando os descritores extraídos das imagens vectoriais na sua forma original e com imagens vectoriais simplificadas, geradas a partir da versão *raster* através da determinação dos polígonos visíveis na imagem *raster*. Como nos testes anteriores, foram efectuados testes em que são usadas imagens *raster 12C* e *AC* utilizadas para gerar as imagens vectoriais simplificadas, pois apesar da cor não ser incluída nestes descritores, a paleta utilizada produz zonas diferentes no processo de segmentação das imagens.

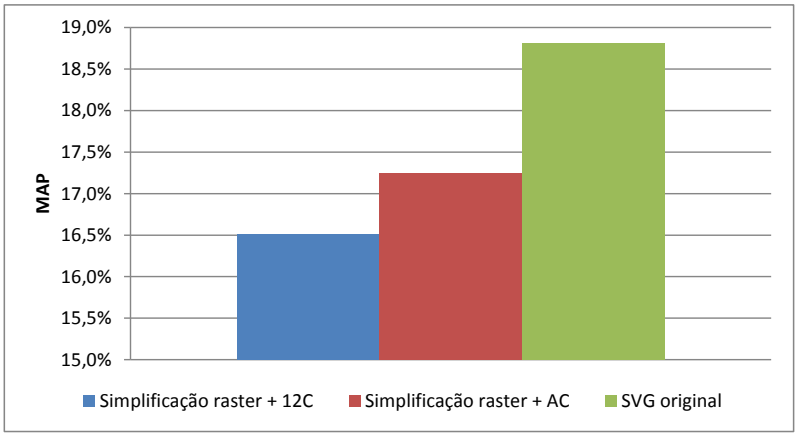


Figura 5.5: Gráfico dos *Mean Average Precision* da pesquisa usando descritores topologia e geometria.

Os resultados obtidos não foram tão bons como os obtidos com os descritores *raster*, como podemos ver na tabela 5.1. Este algoritmo mostrou um melhor desempenho com os descritores extraídos das imagens vectoriais originais (ver figura 5.5). Quanto às imagens simplificadas, foram obtidos piores resultados, menos 1,7%, nos descritores extraídos das imagens *12C*. O algoritmo de simplificação é similar à detecção de regiões de cor, utilizando o sistema EDISON. Tal como nas regiões de cor os polígonos extraídos das imagens *12C* sofrem algumas deformações da junção ou corrupção de certas cores.

A diferença de resultados deve-se à técnica utilizada na extracção dos descritores vectoriais. Uma parte do vector de descritores é baseada nas formas geométricas identificadas na imagem vectorial. As imagens vectoriais completas contêm a representação completa no formato vectorial, que inclui elementos sobrepostos. Um dos objectivos da simplificação é a remoção de informação redundante produzida por estas sobreposições criando uma versão simplificada da imagem utilizando apenas os elementos visíveis. A simplificação acaba por ter um efeito negativo pois perde-se informação de elementos geométricos quando são sobrepostos por outros elementos. A sobreposição causa também um aumento das formas geométricas detectadas, pois as formas visíveis se estiverem suficientemente cobertas são interpretadas como formas geométricas separadas e o grafo de topologia fica mais complexo. Estes factores afectam o desempenho do algoritmo na comparação de imagens reduzindo a similaridade entre certas imagens quando estes factores se apresentam.

	MAP (%)					
	Momentos de cor[a]	Gabor[b]	Regiões de cor[c]	SIFT[d]	Topogeo simplificado[e]	Topogeo original
Bulbs	**34,31**	33,26	23,14	23,93	18,57	10,69
Car	37,22	**56,80**	31,93	22,60	16,00	27,65
Fish	20,02	24,47	**24,59**	20,65	19,53	16,60
Flower	**31,23**	26,05	21,68	22,92	12,61	12,29
Hammer	17,93	12,00	17,10	**20,29**	16,40	25,70
Magnifying Glass	**37,90**	18,08	18,07	18,61	16,17	16,81
Phone	48,72	47,61	**52,24**	20,75	18,48	21,58
Sword	**61,90**	41,03	40,67	30,53	20,38	20,60
TV	23,32	**56,55**	27,67	18,28	20,03	20,62
Tree	**52,23**	24,18	25,97	36,83	14,29	15,61

[a] Factor 4 + 12C
[b] AC
[c] AC + *codebook* com 500 *codewords*
[d] AC + *codebook* com 700 *codewords*
[e] AC + simplificação *raster*

Tabela 5.1: Resumo dos melhores desempenhos dos descritores utilizados isoladamente.

5.2.6 Resultados da combinação de vários descritores

Decidimos colmatar certas desvantagens de alguns algoritmos combinando mais do que um descritor. Efectuamos testes com combinações de pares de descritores *raster*, um descritor que extrai características referentes às cores nas imagens e um descritor que extrai características das texturas. Também efectuamos testes combinando com alguns descritores *raster* os descritores de topologia e geometria extraídos das imagens vectoriais.

Os parâmetros utilizados em cada um dos descritores combinados são os parâmetros que obtiveram melhores resultados nos testes com apenas esse descritor (ver tabela 5.1). A figura 5.6 mostra os resultados obtidos, que são analisados nas secções seguintes. Incluímos os resultados de MAP para os descritores que obtiveram melhores resultados nos testes com apenas um descritor, os descritores de momentos de cor e os descritores de textura extraídos com o banco de filtros de Gabor.

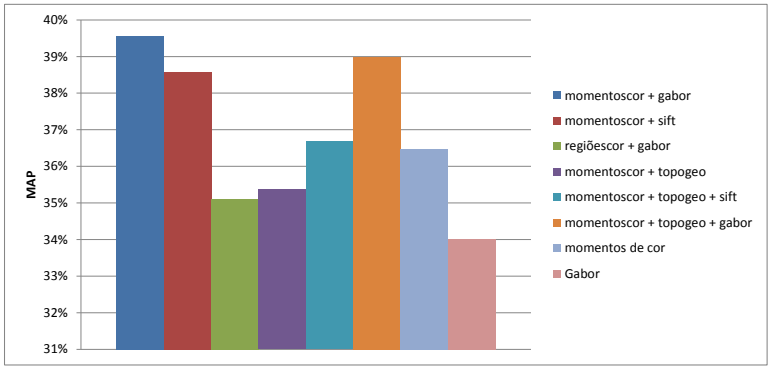


Figura 5.6: Gráfico dos *Mean Average Precision* da pesquisa usando mais do que um descritor.

5.2.6.1 Momentos de cor combinados com textura utilizando o banco de filtros de Gabor

Para obtermos os melhores resultados foram combinados os descritores que tinham obtido melhores resultados quando usados isoladamente. Os momentos de cor foram escolhidos para recolher as características em termos de cor e os filtros de Gabor para recolher as características de textura, presentes nas imagens. Esta combinação obteve o melhor MAP na colecção utilizada nestes testes (ver figura 5.6). Quatro das categorias obtiveram melhor desempenho do que quando utilizadas separadamente (ver tabela 5.2). Estas categorias são: *Bulbs*, *Phone*, *Fish*, *Flower* e *Hammer*. A categoria *Phone* já tinha obtido bons resultados com ambos os descritores separadamente. As três últimas foram das categorias que tiveram piores desempenhos no maior número de algoritmos testados, por isso o aumento do seu desempenho foi positivo. Nas categorias que tinham tido melhores resultados com um dos descritores isoladamente o decréscimo no desempenho foi de aproximadamente 2%. De entre as combinações testadas esta foi também a que teve melhor desempenho em termos de tempo médio de pesquisa (ver tabela).

5.2.6.2 Descritores de momentos de cor combinados com descritores BoF de descritores SIFT

A combinação dos descritores de momentos de cor com os descritores SIFT para identificar os pontos de interesse nas imagens também produziu melhores resultados que ambos os descritores separadamente, um aumentos de 2% em relação ao algoritmo que utiliza apenas os descritores de momentos de cor (ver figura 5.6).

Em relação à combinação dos descritores de momentos de cor com os descritores de textura, obtidos com o banco de filtro de Gabor, quase todas as categorias mostraram pior desempenho com a combinação com os descritores SIFT (ver tabela 5.2). No entanto a junção dos descritores de momentos de cor com outro descritor de características não referentes à cor voltou a produzir uma boa combinação, aumentando o desempenho em relação ao algoritmo que utiliza apenas os descritores de momentos de cor. A utilização dos descritores SIFT provoca um aumento no tempo médio de execução das pesquisas devido ao processo de criação dos descritores *BoF* (ver tabela 7.28).

5.2.6.3 Momentos de cor combinados com descritores de topologia e geometria

Para a primeira combinação de descritores *raster* com descritores vectoriais testámos a combinação do descritor que produziu melhores resultados, dos descritores extraídos de imagens *raster*, os descritores de momentos de cor, com os descritores de geometria e topologia extraídos de imagens vectoriais. Esta combinação teve pior desempenho que os descritores de momentos de cor, quando usados isoladamente, uma diferença de 1%. Nos algoritmos com apenas um descritor a única categoria em que os descritores de topologia e geometria tiveram melhor desempenho que os descritores de momentos de cor foi na categoria *Hammer*. Na combinação destes dois descritores o desempenho desta categoria aumentou, e também o de outras 5 categorias (ver tabela 5.2).

Comparando com o algoritmo que usa os descritores de momentos de cor isoladamente as categorias *Tree*, *Flower* e *Car* tiveram desempenho menor. Nestas categorias a categoria *Tree* tinha obtido um bom desempenho, um decréscimo de aproximadamente 10% com o algoritmo com descritores combinados reflectiu-se no MAP.

As combinações efectuadas com os momentos de cores produziram quase sempre melhores resultados que os momentos de cor quando usados isoladamente. Estas combinações ajudam a colmatar o facto de os descritores de momentos de cor extraírem informação maioritariamente da cor e as categorias incluírem imagens de cores diferentes. Apenas a cor não é suficiente para representar as imagens na colecção.

	MAP (%)					
	Momentos de cor	Gabor	Momentos de cor + Gabor	Momentos de cor + SIFT	Regiões de cor + Gabor	Momentos de cor + Topogeo
Bulbs	34,31	33,26	**41,99**	36,99	30,35	34,07
Car	37,22	**56,80**	54,87	41,44	53,77	34,27
Fish	20,02	24,47	**28,62**	25,51	26,50	20,15
Flower	31,23	26,05	**36,90**	31,28	27,06	27,02
Hammer	17,93	12,00	19,09	**20,19**	16,32	19,02
Magnifying Glass	37,90	18,08	35,13	37,55	21,08	**38,82**
Phone	48,72	47,61	54,11	47,93	**56,68**	49,30
Sword	61,90	41,03	48,67	**68,23**	41,60	62,78
TV	23,32	**56,55**	35,88	22,68	45,68	24,34
Tree	52,23	24,18	40,30	**53,87**	31,96	43,89

Tabela 5.2: Desempenho dos algoritmos usando a combinação de dois descritores.

5.2.6.4 Descritores de *raster* combinados com descritores de topologia e geometria

Um dos outros temas abordados no decorrer desta dissertação é a combinação dos descritores de imagens *raster* com descritores de imagens vectoriais. Foram realizadas combinações com os descritores utilizados nos algoritmos que demonstraram melhor desempenho (ver tabela 5.2). A única combinação que incluiu apenas dois descritores nestes testes foi a combinação analisada anteriormente dos momentos de cor com os descritores de topologia e geometria. As restantes combinações implementadas foram:

- Combinação de descritores de momentos de cor, descritores SIFT e descritores de topologia e geometria;
- Combinação de descritores de momentos de cor, descritores de textura, extraídos com o banco de filtros Gabor, e descritores de topologia e geometria.

Na combinação utilizando os descritores **SIFT**, os descritores de **momentos de cor** e os descritores de **topologia e geometria**, apenas 3 categorias obtiveram melhores resultados que a combinação sem os descritores de topologia e geometria (ver tabela 5.3). A adição dos descritores de topologia e geometria à combinação provocou uma redução no MAP de aproximadamente 2% (ver figura 5.6).

A combinação dos descritores extraídos com os **filtros de Gabor**, dos descritores de **momentos de cor** e dos descritores de **topologia e geometria** teve um melhor desempenho que a combinação anterior, 2% acima. Em relação aos testes

	MAP (%)			
	Momentos de cor + Gabor	Momentos de cor + SIFT	Momentos de cor + Topogeo + Sift	Momentos de cor + Topogeo + Gabor
Bulbs	41,99	36,99	38,51	**43,14**
Car	**54,87**	41,44	36,65	53,56
Fish	**28,62**	25,51	22,62	26,08
Flower	**36,90**	31,28	26,04	30,65
Hammer	19,09	20,19	**25,52**	19,54
Magnifying Glass	35,13	**37,55**	35,06	35,63
Phone	54,11	47,93	46,41	**55,59**
Sword	48,67	**68,23**	63,32	49,37
TV	**35,88**	22,68	25,26	34,21
Tree	40,30	**53,87**	47,42	42,11

Tabela 5.3: Comparação do desempenho após adicionar os descritores de topologia e geometria.

anteriores, apenas com os descritores *raster*, o desempenho foi menor, mas com apenas uma diferença de 0,5%. Por outro lado o número de categorias que melhoraram de desempenho foi mais elevado do que as que pioraram, como podemos ver na tabela 5.3. O algoritmo com apenas os descritores *raster* tem um maior desvio padrão entre as diferentes categorias, principalmente nas categorias que obtiveram melhores desempenhos. Este algoritmo obteve um desvio padrão entre categorias menor, mas o desempenho das categorias com melhores resultados diminuiu e isso afectou o MAP. No geral, se considerarmos todas as categorias, esta combinação foi a que obteve melhores resultados em mais categorias ao contrário do que a diferença no MAP possa indicar.

5.3 Análise experimental com base de dados de grandes dimensões

Da colecção anterior foram removidas 2 imagens de cada categoria para usar como imagens de pesquisa, resultando num total de 20 pesquisas. Para avaliar o desempenho em bases de dados de grandes dimensões dos algoritmos implementados, adicionámos 12885 imagens à colecção anterior, com excepção das imagens seleccionadas como pesquisa.

As imagens foram divididas por categorias, as mesmas utilizadas nos testes da base de dados controlada e uma outra categoria *Misc*. Nesta última categoria

incluíram-se todas as imagens que não pertenciam a nenhuma das categorias anteriores, tornando-se na categoria com mais imagens. A base de dados incluí 41 imagens na categoria *Bulbs*, 168 na categoria *Car*, 88 na *Fish*, 309 na *Flower*, 29 na *Hammer*, 21 na *Magnifying Glass*, 50 na *Phone*, 53 na *Sword*, 73 na *TV* e 165 na *Tree* e as restantes na categoria *Misc*. Foi calculada a média de *R-Precision* para 5 resultados relevantes à pesquisa. Um resultado é considerado relevante se pertence à mesma categoria que a imagem de pesquisa.

Nestes testes tornou-se mais difícil avaliar o algoritmo como um todo, as várias categorias têm resultados muito diferentes. Foi feita uma análise com a média de todas as pesquisas (20 pesquisas) de modo a obter dados significativos para analisar o algoritmo como um todo. No entanto, foram analisadas as categorias de forma independente e apesar de serem menos dados (2 pesquisas por categoria) estes são mais relevantes na análise do desempenho do algoritmo.

5.3.1 Descritores únicos

Executámos os testes com a combinação de parâmetros que demonstrou melhor desempenho nos testes com a base de dados controlada (ver secção 5.2). Desta base de testes foram removidos os descritores de região de cor. Os testes anteriores revelaram claramente que os momentos de cor têm melhor desempenho como representante das características de cor presentes nos clip arts da colecção.

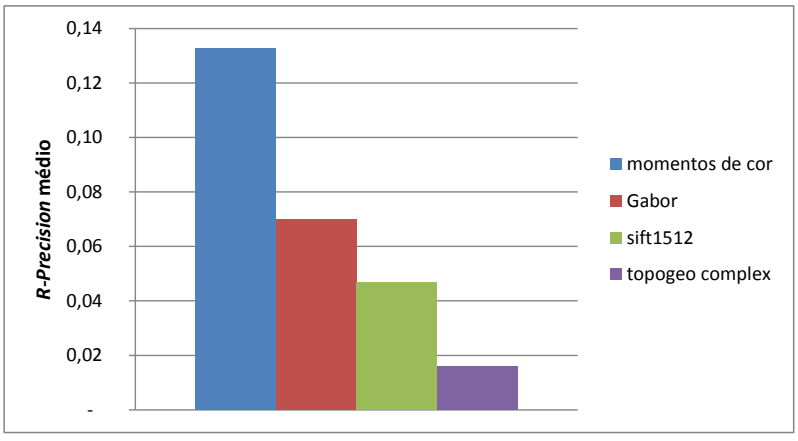


Figura 5.7: Gráfico da média da *R-Precision* para 5 resultados relevantes usando apenas um descritor.

Nos testes com pesquisas utilizando apenas um descritor, os momentos de cor voltaram a ser a característica que obteve melhor desempenho. No entanto,

quando feita a análise do desempenho das diferentes categorias, existem algumas em que o seu desempenho foi menor que outros descritores (ver figura 5.8 e tabela 5.4), indicando que a sua combinação poderá colmatar algumas falhas no algoritmo, como podemos ver na tabela 5.5. Em parte, esta quebra no desempenho é obtida novamente pela grande desvantagem dos momentos de cor, a sua total dependência das cores presentes na imagem. Caso da pesquisa seja uma imagem com apenas uma cor, os momentos de cor obtêm muitos falsos positivos nos resultados pois existem muitos clip arts que utilizam este tipo de imagem, principalmente imagens a negro.

O descritor de textura extraído com o banco de filtros de Gabor foi o algoritmo que obteve o segundo melhor desempenho. Tal deve-se a ter obtido o melhor resultado, entre os algoritmos que utilizam apenas um descritor, na categoria *Car* (ver figura 5.8 e tabela 5.4). Também obteve o melhor desempenho na categoria *TV*, uma das categorias que obteve os desempenhos mais fracos.

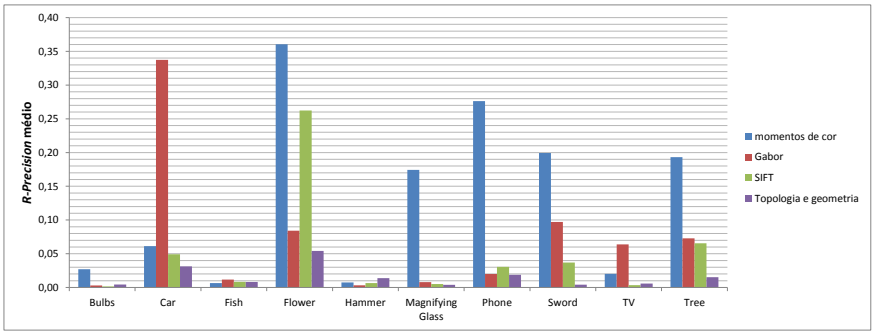


Figura 5.8: Gráfico da média da *R-Precision* para 5 resultados relevantes nas diferentes categorias usando apenas um descritor.

Os descritores SIFT voltaram a apresentar um desafio devido ao método de gerar o *codebook*. Os parâmetros utilizados na base de dados controlada, não implicam o mesmo desempenho no algoritmo. Isto deve-se à introdução de novas *codewords* no sistema, existentes nas imagens que compõem a nova base de dados. Foram feitos testes com novos tamanhos de *codebooks* manipulando o número de *clusters* no algoritmo *K-means clustering* utilizado. Testámos o algoritmo com 700, o melhor dos testes realizados anteriormente, com 1024, com 1512 e com 1768 *clusters*. Novamente o *codebook* com 700 *codewords* demonstrou um dos melhores resultados nos testes executados. O algoritmo utilizando 1512 foi o que obteve melhor média de *R-Precision*, pois obteve resultados muitos bons na categoria *Flower*. No geral, 700 *codewords*, mostraram um melhor desempenho em quase

todas as categorias (ver figura 5.9). Quando comparado com os outros algoritmos os descritores SIFT não tiveram bons resultados. Mesmo na categoria *Flower* foram obtidos melhores desempenhos com o algoritmo que usa os momentos de cor (ver figura 5.8 e tabela 5.4).

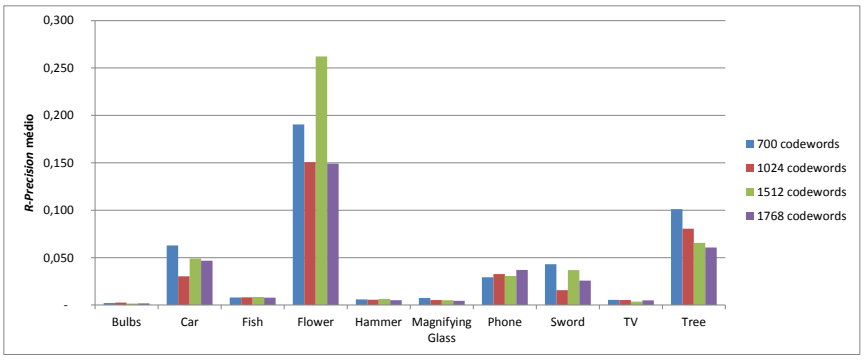


Figura 5.9: Gráfico da média da *R-Precision* para 5 resultados relevantes nas diferentes categorias usando vários tamanhos de *codebooks* com os descritores SIFT.

Nos descritores de topologia e geometria retirados das imagens vectoriais, voltaram a ser observados desempenhos semelhantes aos testes anteriores. Foram observados melhores resultados com a simplificação *raster* das imagens para 12 cores nas categorias *TV* e *Sword* (ver figura 5.10), mas no geral, como anteriormente, o algoritmo que utiliza imagens vectoriais sem simplificação obteve o melhor desempenho. Quando comparado com o desempenho de outros algoritmos foi dos algoritmos com pior desempenho (ver figura 5.8 e tabela 5.4). No entanto a sua combinação com outros descritores veio a criar um aumento no desempenho desses algoritmos, que iremos abordar na secção seguinte.

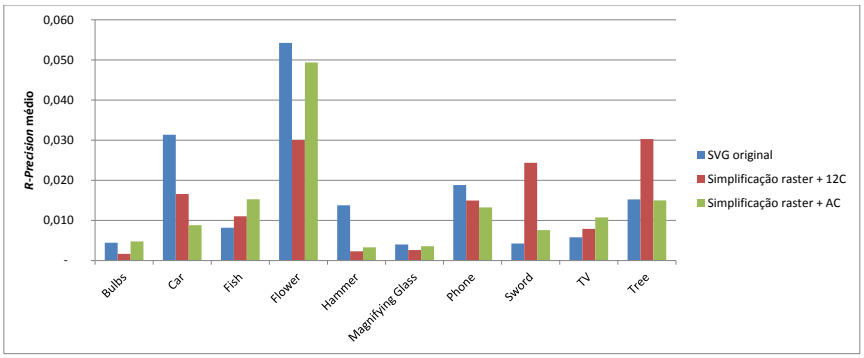


Figura 5.10: Gráfico da média da *R-Precision* para 5 resultados relevantes nas diferentes categorias usando os descritores de topologia e geometria.

	R-Precision para 5 resultados relevantes					
	Momentos de cor[a]	Gabor	SIFT[b]	Topogeo simplificado[c]	Topogeo simplificado[d]	Topogeo original[e]
Bulbs	**0,027**	0,003	0,002	0,005	0,002	0,004
Car	0,061	**0,337**	0,049	0,009	0,017	0,031
Fish	0,007	0,012	0,008	**0,015**	0,011	0,008
Flower	**0,361**	0,084	0,262	0,049	0,030	0,054
Hammer	0,007	0,003	0,006	0,003	0,002	**0,014**
Magnifying Glass	**0,174**	0,008	0,005	0,004	0,003	0,004
Phone	**0,276**	0,020	0,031	0,013	0,015	0,019
Sword	**0,199**	0,097	0,037	0,008	0,024	0,004
TV	0,020	**0,064**	0,004	0,011	0,008	0,006
Tree	**0,193**	0,073	0,066	0,015	0,030	0,015

[a] Factor 4 + 12C
[b] AC + *codebook* com 1512 *codewords*
[c] AC + simplificação *raster*
[d] 12C + simplificação *raster*
[e] Imagens vectoriais originais

Tabela 5.4: Desempenho dos descritores isoladamente com base de dados de grande dimensão.

5.3.2 Combinação de descritores

Os testes da combinação de descritores foram efectuados com os parâmetros que demonstraram melhor desempenho ao longo dos diversos testes. Os descritores SIFT foram analisados utilizando um *codebook* com 1512 *codewords*. Os restantes algoritmos mantiveram os parâmetros que demonstraram melhores desempenhos nos testes com a base de dados controlada e nos testes com a base de dados de grandes dimensões.

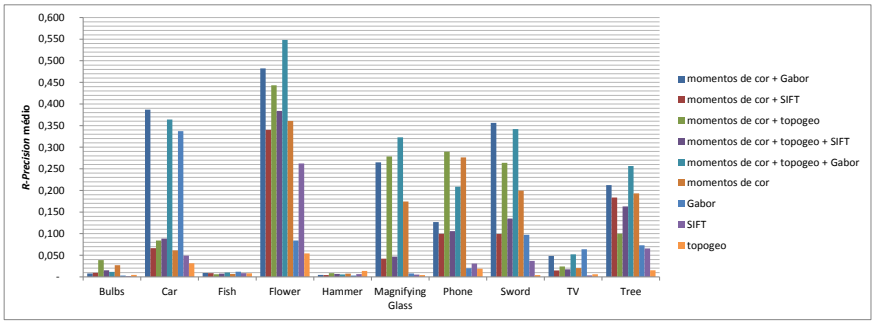


Figura 5.11: Gráfico da média da *R-Precision* para 5 resultados relevantes nas diferentes categorias usando apenas combinação de múltiplos descritores.

No geral as categorias que tiveram pior desempenho foram *Bulbs*, *Fish*, *Hammer* e *TV*. Nenhuma destas obteve bom desempenho (ver figura 5.11), quer com os descritores individuais, quer com a combinação de descritores.

Dentro da gama de descritores extraídos de imagens *raster* o melhor desempenho foi, de novo, observado na combinação entre os descritores de momentos de cor e os descritores extraídos com os filtros de Gabor (ver figura 5.11). Esta combinação teve melhor desempenho em todas as categorias, excepto nas categorias *Fish* e *Hammer* (ver tabelas 5.4 e 5.5), quando comparada com cada um dos descritores utilizados isoladamente. Nas restantes categorias é visível que a combinação ajuda a colmatar as desvantagens de cada um dos descritores, aumentando o desempenho consideravelmente, de 0,13 nos momentos de cor para 0,19 de média de *R-Precision*.

Novamente os descritores SIFT desiludiram, piorando o desempenho dos algoritmos em que foram incluídos. Foi observado um decréscimo no desempenho em quase todas as combinações com os descritores SIFT. O desempenho do algoritmo que inclui os descritores BoF de descritores SIFT com os descritores de momentos de cor foi menor que o algoritmo com os descritores de momentos de cor isoladamente. O algoritmo que inclui a combinação dos descritores BoF de descritores SIFT com os descritores de momentos de cor e com os descritores de topologia e geometria obteve melhor desempenho que os descritores de topologia utilizados isoladamente mas pior desempenho que os descritores de momentos de cor (ver figura 5.12).

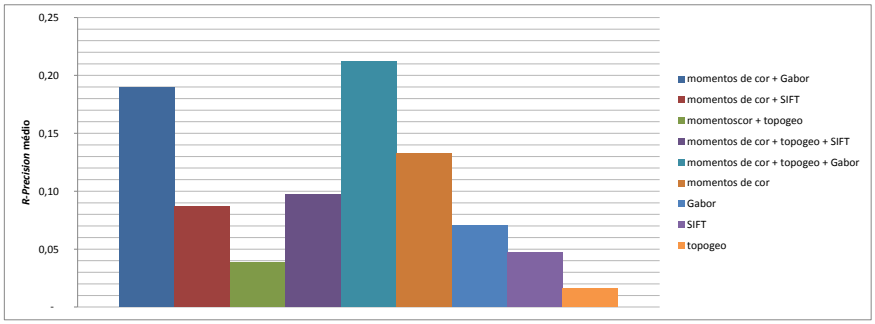


Figura 5.12: Gráfico da média da *R-Precision* para 5 resultados relevantes usando apenas combinação de múltiplos descritores.

A última combinação testada, que utiliza dois descritores, já inclui um descritor extraído de uma imagem *raster*, o descritor de momentos de cor, e o descritor de topologia e geometria extraído de uma imagem vectorial. Esta combinação voltou a obter melhor desempenho que o descritor de momentos de cor quando

	R-Precision para 5 resultados relevantes				
	Momentos de cor	Gabor	Momentos de cor + Gabor	Momentos de cor + SIFT	Momentos de cor + Topogeo
Bulbs	0,027	0,003	0,008	0,009	**0,039**
Car	0,061	0,337	**0,387**	0,066	0,084
Fish	0,007	**0,012**	0,009	0,008	0,006
Flower	0,361	0,084	**0,482**	0,340	0,443
Hammer	0,007	0,003	0,005	0,004	**0,008**
Magnifying Glass	0,174	0,008	0,265	0,042	**0,279**
Phone	0,276	0,020	0,127	0,099	**0,290**
Sword	0,199	0,097	**0,356**	0,099	0,264
TV	0,020	**0,064**	0,048	0,015	0,024
Tree	0,193	0,073	**0,212**	0,184	0,100

Tabela 5.5: Desempenho dos algoritmos com a base de dados de grandes dimensões com combinação de dois descritores.

utilizado isoladamente (ver figura 5.12). As melhorias abrangem quase todas as categorias, excepto nas categorias *Fish* e *Hammer*, que já tinham obtido um desempenho fraco, e na categoria *Tree* onde se notou o maior decréscimo, de 0,193 nos momentos de cor, para 0,1 (ver figura 5.11 e tabela 5.5). A média de *R-Precision* do algoritmo aumentou para 0,15 de 0,13 obtidos com os momentos de cor, menor que a combinação com os descritores extraídos com os filtros de Gabor.

Nas combinações de três descritores fizemos os mesmos testes que na base de dados controlada. Apenas é alterado o descritor *raster* que representa as características de textura da imagem em cada um dos testes. Ou seja foram feitos os testes combinando os descritores de topologia e geometria extraídos das imagens vectoriais com os descritores de momentos de cor extraídos das imagem *raster* e, ainda das imagens *raster*, com os descritores extraídos com os filtros de Gabor ou os descritores SIFT.

Como foi descrito acima, os descritores SIFT pioraram o desempenho do algoritmo. A combinação com os descritores de topologia e geometria não fez com que o seu desempenho melhorasse. O algoritmo obteve um desempenho pior com a adição dos descritores de topologia e geometria como podemos ver na figura 5.12 e na tabela 5.6.

Quanto à combinação de descritores com os filtros de Gabor, obteve a melhor média de *R-Precision* de todos os algoritmos testados (ver figura 5.12). A análise das várias categorias mostra que obteve pior desempenho que outro algoritmo em 6 categorias (ver figura 5.11), no entanto, 4 dessas categorias foram aquelas

	R-Precision para 5 resultados relevantes			
	Momentos de cor + Gabor	Momentos de cor + SIFT	Momentos de cor + Topogeo + Sift	Momentos de cor + Topogeo + Gabor
Bulbs	0,008	0,009	**0,015**	0,012
Car	**0,387**	0,066	0,088	0,364
Fish	0,009	0,008	0,008	**0,010**
Flower	0,482	0,340	0,384	**0,548**
Hammer	0,005	0,004	**0,007**	0,005
Magnifying Glass	0,265	0,042	0,047	**0,323**
Phone	0,127	0,099	0,106	**0,209**
Sword	**0,356**	0,099	0,135	0,342
TV	0,048	0,015	0,017	**0,052**
Tree	0,212	0,184	0,163	**0,256**

Tabela 5.6: Comparação do desempenho, na base de dados de grandes dimensões, após adicionar os descritores de topologia e geometria.

com que tivemos mais problemas, com quase todos os algoritmos. Essas categorias são *Bulbs*, *Fish*, *Hammer* e *TV*, em que a diferenças entre o desempenho dos algoritmos em termos estatísticos é mínima. De qualquer modo os algoritmos que tiveram melhor desempenho em determinada categoria são aqueles que incluem um ou dois dos descritores utilizados, e não existe nenhum algoritmo que tenha obtido melhores resultados que este algoritmo em mais de duas categorias em simultâneo.

Como foi observado nos testes com a base de dados controlada o algoritmo mais estável, ou seja o que obteve menor desvio padrão entre as várias categorias, foi o que utilizou os momentos de cor, os filtros de Gabor e a topologia e geometria. Nos testes com a base de dados de grandes dimensões manteve-se o algoritmo mais estável, com desempenho mais alto em quase todas as categorias ou muito próximo do melhor desempenho nessa categoria (ver tabela 5.6). Neste caso o algoritmo teve a média de *R-Precision* mais alta, comprovando a hipótese que com a base de dados de grandes dimensões o seu desempenho iria ser mais visível.

6

Conclusões e trabalho futuro

Neste capítulo são discutidas as principais vantagens e desvantagens da solução implementada e as áreas que podem ser melhoradas no futuro. Foi implementada uma solução de pesquisa e recuperação de imagens clip arts baseada no seu conteúdo. O objectivo era extrair características de imagens *raster* e características de imagens vectoriais e efectuar uma comparação do desempenho e da combinação de várias características. Inicialmente, foi efectuada uma pesquisa de sistemas de CBIR de modo a identificar quais as características a serem implementadas. Essas características foram estudadas de modo a incluir a sua implementação no sistema.

Foi criado um módulo que inclui as funcionalidades de pré-processamento, extracção de características e armazenamento dos vectores de descritores em memória permanente. De forma a fornecer essas funcionalidades foram implementados algoritmos de extracção de vectores de descritores para extrair as seguintes categorias: momentos de cor, textura extraída a partir dos filtros de Gabor, regiões de cor, SIFT e topologia e geometria. De forma a extrair as manchas de cor utilizadas nas regiões de cor e para gerar os polígonos foi utilizado o sistema EDISON. Foi implementado um algoritmo de vectorização dos polígonos que guarda imagens no formado SVG em memória permanente para depois ser processado pelo sistema que gera os descritores de topologia e geometria. Das várias características extraídas os momentos de cor e a textura extraída com o banco de filtros de Gabor foram os que obtiveram menores tempos de execução. A extracção do SIFT, regiões de cor e topologia e geometria é mais demorada pois recorrem a

aplicações ou *scripts* externos ao nosso sistema.

De seguida foi desenvolvido outro módulo que permite executar interrogações no sistema e avaliar o seu desempenho. Neste módulo foram incluídas as funcionalidades de normalização dos descritores das imagens na colecção e dos descritores das interrogações. Foram implementados algoritmos que permitem a recuperação de imagens na colecção, utilizando pesquisa total da imagem, utilizando o método de *query-by-example*. Entre estes algoritmos foram implementados diversos modos de interrogação da colecção. Podem ser feitas interrogações utilizando apenas uma característica ou algumas das diversas combinações de características implementadas, que incluem combinação de descritores extraídos de imagens *raster* para representar as características de cor, textura e pontos de interesse na imagem e combinação com os descritores de topologia e geometria extraídos de imagens vectoriais. De forma a avaliar o desempenho de cada um dos algoritmos foram utilizadas as medidas de *precision*, *recall*, *average precision*, *R-Precision* e *mean average precision* e implementados métodos para que sejam calculadas automaticamente.

Foram feitos testes com os diversos algoritmos implementados utilizando duas colecções, uma base de dados controlada com 100 clip arts e uma base de dados de grandes dimensões que incluía 12885 clip arts. Os testes executados na base de dados controlada utilizaram as próprias imagens presentes na colecção para executar *query-by-example* e retornar os 10 melhores resultados e apresentaram bons resultados. O descritor de momentos de cor obteve o melhor desempenho dentro dos algoritmos que utilizam apenas uma característica, seguido do descritor de textura extraído com o banco de filtros de Gabor. Nestes testes as regiões de cor e os descritores SIFT apresentaram resultados abaixo do esperado e o tempo de execução das interrogações foi o mais elevado das características utilizadas. Os descritores de topologia e geometria apresentaram um desempenho menor que todas as características extraídas das imagens *raster*. A simplificação das imagens vectoriais não produziu o efeito esperado. O objectivo era reduzir a informação não visível encontrada nas imagens vectoriais e desse modo o número de polígonos e o tempo de extracção dos descritores. No entanto, foram criados mais polígonos que nas imagens vectoriais originais devido à sobreposição de elementos. A sobreposição de elementos faz com que o elemento que fica parcialmente tapado seja dividido em vários polígonos o que afecta a identificação de elementos semelhantes, caso estejam parcialmente sobrepostos por outros elementos numa imagem e não noutra. Ficou determinado que a simplificação vectorial com recurso a imagens *raster* não produz aumento de desempenho.

Outro elemento abordado foi a combinação de características presentes nas imagens *raster* com características nas imagens vectoriais. Inicialmente fizemos pares de características *raster* para representar tanto a cor como a textura ou pontos de interesse. Os melhores resultados foram obtidos com a combinação dos momentos de cor com a textura obtida através do banco de filtros de Gabor. Fizemos combinações com os descritores de topologia e geometria, utilizando descritores de momentos de cor, os descritores de momentos de cor mais descritores de textura extraídos com o banco de filtros de Gabor e os descritores de momentos de cor mais descritores SIFT. Os melhores resultados foram obtidos pela segunda combinação, apesar de o MAP dos resultados ser menor que apenas utilizando as características *raster*, o algoritmo com apenas estas características tem um desvio padrão mais elevado entre as diversas categorias.

De seguida foram efectuados testes com uma base de dados de grandes dimensões de forma a analisar o comportamento dos algoritmos numa situação mais realista, avaliando os resultados segundo a sua *R-Precision* para 5 imagens relevantes em relação à pesquisa. Os resultados não foram muitos bons devido ao grande número de imagens na colecção adicionar muito ruído. Estes testes ajudaram a comprovar o que havia sido observado nos testes com a base de dados controlada. Com uma base de dados com maior número de clip arts as desvantagens dos descritores de momentos de cor foram mais visíveis, no entanto foi a característica com melhor desempenho. Assim, conseguimos observar que a combinação dos momentos de cor com uma outra característica diferente tinha um desempenho mais elevado do que o algoritmo em que eram apenas usados os descritores de momentos de cor. Apesar de ser o descritor com melhor desempenho os momentos de cor têm algumas desvantagens, sendo uma delas o facto de não ter muita informação quanto à forma da imagem. Quando combinado com uma outra característica que não seja referente à cor os resultados tiveram melhor desempenho. Como tinha sido observado nos testes com a base de dados controlada o algoritmo que inclui os momentos de cor, a textura e a topologia e geometria obteve o melhor desempenho. Novamente os descritores de topologia e geometria tiveram o pior desempenho de entre os algoritmos que usam apenas um descritor.

O trabalho desenvolvido no âmbito desta dissertação deixa espaço para futuras extensões dos algoritmos implementados bem como das técnicas de recuperação de imagem. Algo que não se inseria no âmbito desta dissertação era o desempenho dos algoritmos em termos temporais. No futuro esse aspecto deverá ter sido em conta no sentido de tornar os algoritmos mais robustos e com melhor

tempo de resposta.

Um factor a explorar que influencia o tempo de resposta seria o desenvolvimento de uma melhor estrutura de armazenamento, que fosse mais fiável e com melhor desempenho temporal. A pesquisa linear é muito morosa e tem grande impacto quando considerado o tamanho da colecção utilizada pelo sistema. Melhorar o seu desempenho pode ser possível com a implementação de uma estrutura indexada que permitisse pesquisas *k-nearest neighbor*.

De modo a aumentar o desempenho da simplificação de imagens vectoriais podemos criar novos métodos de geração e simplificação de polígonos. Estes métodos deverão criar polígonos mais simples e com menos ruído através da remoção e fusão de certas áreas. Desta forma, poderemos aumentar o desempenho dos algoritmos de topologia e geometria com imagens simplificadas e o desempenho dos algoritmos de extracção e recuperação de imagens utilizando as regiões de cor ou a combinação de ambas as características num só descritor.

Devido à complexidade de certos clip arts o nosso sistema poderá beneficiar de extensões aos algoritmos existentes, de forma a que seja possível efectuar recuperação de clip arts usando pesquisas parciais, ou seja pesquisas por uma imagem complexa usando apenas parte dela como interrogação ao sistema.

Um próximo passo para o sistema envolve a pesquisa de imagens através de esboços feitos pelo utilizador, o que requer a implementação de uma interface que permita essa interacção do utilizador com o sistema. A solução implementada poderá ser testada comparativamente a soluções existentes que utilizam apenas características presentes em imagens vectoriais.

Bibliografia

[1] AMANTE, J. C., AND FONSECA, M. J. Fuzzy color space segmentation to identify the same dominant colors as users. In *18th International Conference on Distributed Multimedia Systems (DMS'12)* (Miami Beach, USA, 2012), Knowledge Systems Institute, pp. 48–53.

[2] BERCHTOLD, S., KEIM, D. A., AND KRIEGEL, H.-P. The x-tree: An index structure for high-dimensional data. In *Proceedings of the 22th International Conference on Very Large Data Bases* (San Francisco, CA, USA, 1996), VLDB '96, Morgan Kaufmann Publishers Inc., pp. 28–39.

[3] BERLIN, B., AND KAY, P. *Basic Color Terms: Their Universality and Evolution*. University of California Press, 1969.

[4] BÖHM, C., BERCHTOLD, S., AND KEIM, D. A. Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. *ACM Comput. Surv. 33* (September 2001), 322–373.

[5] BRIGHAM, E. O. *The fast Fourier transform and its applications*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[6] CANNY, J. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell. 8*, 6 (June 1986), 679–698.

[7] CASTELLI, V., AND BERGMAN, L. D. *Digital Imagery: Fundamentals*. John Wiley and Sons, Inc., 2002, pp. 1–10.

[8] CAVACO, S., CORREIA, N., JESUS, R., MALHEIRO, F., AND MATEUS, J. Video annotation of tv content using audiovisual information. In *The 3rd International Conference on Multimedia Computing and Systems (ICMCS'12)* (2012).

[9] CHANG, E. Y., LI, B., AND LI, C. Toward perception-based image retrieval. In *Proceedings of the IEEE Workshop on Content-based Access of Image and Video Libraries (CBAIVL'00)* (Washington, DC, USA, 2000), CBAIVL '00, IEEE Computer Society, pp. 101–.

[10] CHRISTOUDIAS, C. M. Synergism in low level vision. In *Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02) Volume 4 - Volume 4* (Washington, DC, USA, 2002), ICPR '02, IEEE Computer Society, pp. 40150–.

[11] COMANICIU, D., AND MEER, P. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell. 24*, 5 (May 2002), 603–619.

[12] DATTA, R., JOSHI, D., LI, J., AND WANG, J. Z. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv. 40*, 2 (May 2008), 5:1–5:60.

[13] DAUBECHIES, I. *Ten Lectures on Wavelets (CBMS-NSF Regional Conference Series in Applied Mathematics)*, 1 ed. SIAM: Society for Industrial and Applied Mathematics, June 1992.

[14] DENG, K., AND MOORE, A. W. Multiresolution instance-based learning. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2* (San Francisco, CA, USA, 1995), Morgan Kaufmann Publishers Inc., pp. 1233–1239.

[15] DOUGLAS, D. H., AND PEUCKER, T. K. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization 10*, 2 (Oct. 1973), 112–122.

[16] FERREIRA, A., FONSECA, M. J., JORGE, J. A., AND RAMALHO, M. Mixing images and sketches for retrieving vector drawings. In *Proceedings of the Seventh Eurographics conference on Multimedia* (Aire-la-Ville, Switzerland, Switzerland, 2004), EGMM'04, Eurographics Association, pp. 69–75.

[17] FISCHLER, M. A., AND BOLLES, R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM 24* (June 1981), 381–395.

[18] FLICKNER, M., SAWHNEY, H., NIBLACK, W., ASHLEY, J., HUANG, Q., DOM, B., GORKANI, M., HAFNER, J., LEE, D., PETKOVIC, D., STEELE, D.,

AND YANKER, P. Query by image and video content: The qbic system. *Computer 28*, 9 (Sept. 1995), 23–32.

[19] FONSECA, M. J., FERREIRA, A., AND JORGE, J. A. Generic shape classification for retrieval. In *Proceedings of the 6th international conference on Graphics Recognition: ten Years Review and Future Perspectives* (Berlin, Heidelberg, 2006), GREC'05, Springer-Verlag, pp. 291–299.

[20] FONSECA, M. J., FERREIRA, A., AND JORGE, J. A. Sketch-based retrieval of complex drawings using hierarchical topology and geometry. *Comput. Aided Des. 41*, 12 (Dec. 2009), 1067–1081.

[21] FONSECA, M. J., AND JORGE, J. A. Indexing high-dimensional data for content-based retrieval in large databases. Tech. rep., Washington, DC, USA, 2003.

[22] FONSECA, M. J., PIMENTEL, C., AND JORGE, J. A. Cali: An online scribble recognizer for calligraphic interfaces. In *Sketch Understanding, Papers from the 2002 AAAI Spring Symposium* (2002), pp. 51–58.

[23] GIMEL'FARB, G. *Lecture Notes CBIR: Texture Features*, 2009 (acedido a 13 Setembro 2012). `http://www.cs.auckland.ac.nz/compsci708s1c/lectures/Glect-html/top708-2006.html`.

[24] HUANG, J., KUMAR, S. R., MITRA, M., ZHU, W.-J., AND ZABIH, R. Image indexing using color correlograms. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)* (Washington, DC, USA, 1997), CVPR '97, IEEE Computer Society, pp. 762–.

[25] HUANG, M., YANG, M., LIU, F., AND WU, E.-H. Stroke extraction in cartoon images using edge-enhanced isotropic nonlinear filter. In *Proceedings of the 9th ACM SIGGRAPH Conference on Virtual-Reality Continuum and its Applications in Industry* (New York, NY, USA, 2010), VRCAI '10, ACM, pp. 33–38.

[26] JESUS, R., DIAS, R., FRIAS, R., ABRANTES, A., AND CORREIA, N. *Sharing Personal Experiences while Navigating in Physical Spaces*. PhD thesis, 2007.

[27] JOLION, J.-M. Principles of visual information retrieval. Springer-Verlag, London, UK, UK, 2001, ch. Feature similarity, pp. 121–143.

[28] KATAYAMA, N., AND SATOH, S. The sr-tree: an index structure for high-dimensional nearest neighbor queries, 1997.

[29] LAM, B., AND CIESIELSKI, V. Discovery of human-competitive image texture feature extraction programs using genetic programming. In *Genetic and Evolutionary Computation – GECCO 2004*, K. Deb, Ed., vol. 3103 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2004, pp. 1114–1125.

[30] LAWS, K. *Textured Image Segmentation*. PhD thesis, Univ. Southern California, Los Angeles, CA, 1980.

[31] LEE, T. S. Image representation using 2d gabor wavelets. *IEEE Trans. Pattern Anal. Mach. Intell. 18*, 10 (Oct. 1996), 959–971.

[32] LEUNG, T., AND MALIK, J. Representing and recognizing the visual appearance of materials using three-dimensional textons. *Int. J. Comput. Vision 43*, 1 (June 2001), 29–44.

[33] LEW, M. S., Ed. *Principles of visual information retrieval*. Springer-Verlag, London, UK, 2001.

[34] LEW, M. S., SEBE, N., DJERABA, C., AND JAIN, R. Content-based multimedia information retrieval: State of the art and challenges. *ACM Trans. Multimedia Comput. Commun. Appl. 2* (February 2006), 1–19.

[35] LI, F.-F., AND PERONA, P. A bayesian hierarchical model for learning natural scene categories. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2 - Volume 02* (Washington, DC, USA, 2005), CVPR '05, IEEE Computer Society, pp. 524–531.

[36] LIN, K. I., JAGADISH, H. V., AND FALOUTSOS, C. The tv-tree: an index structure for high-dimensional data. *The VLDB Journal 3*, 4 (Oct. 1994), 517–542.

[37] LOWE, D. G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision 60*, 2 (Nov. 2004), 91–110.

[38] MARKEY, K. Access to iconographical research collections. *Library Trends 37*, 2 (1988), 154–174.

[39] MEER, P., AND GEORGESCU, B. Edge detection with embedded confidence. *IEEE Trans. Pattern Anal. Machine Intell 23* (2001), 1351–1365.

[40] MIKOLAJCZYK, K., AND SCHMID, C. Scale and affine invariant interest point detectors. *International Journal of Computer Vision 60* (2004), 63–86.

[41] O'GORMAN, L. Primitives chain code [image analysis]. In *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on* (Apr. 1988), AT&T Bell Lab., Murray Hill, NJ, IEEE.

[42] PEREIRA, F., AND KOENEN, R. Mpeg-7: A standard for multimedia content description. *Int. J. Image Graphics 1*, 3 (2001), 527–546.

[43] PIMENTA, N. Photofinder 2: Classificação e pesquisa de fotografias digitais. Master's thesis, IST/TU Lisbon, Portugal, May 2008.

[44] RONG, M. Perception-based multi-quality image compression for efficient transmission. In *Intelligent Signal Processing and Communications, 2006. ISPACS '06. International Symposium on* (dec. 2006), pp. 817 –820.

[45] SHI, J., AND MALIK, J. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22* (2000), 888–905.

[46] SHI, J., AND TOMASI, C. Good features to track. In *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)* (1994), pp. 593 – 600.

[47] SHIH, J.-L., AND CHEN, L.-H. Color image retrieval based on primitives of color moments. In *Proceedings of the 5th International Conference on Recent Advances in Visual Information Systems* (London, UK, UK, 2002), VISUAL '02, Springer-Verlag, pp. 88–94.

[48] SMITH, J. R., AND CHANG, S.-F. Visualseek: a fully automated content-based image query system. In *Proceedings of the fourth ACM international conference on Multimedia* (New York, NY, USA, 1996), MULTIMEDIA '96, ACM, pp. 87–98.

[49] SOUSA, P., AND FONSECA, M. J. Geometric matching for clip-art drawing retrieval. *Journal of Visual Communication and Image Representation 20*, 2 (2009), 71 – 83. Special issue on Emerging Techniques for Multimedia Content Sharing, Search and Understanding.

[50] STRICKER, M. A., AND ORENGO, M. Similarity of color images. In *Storage and Retrieval for Image and Video Databases (SPIE)'95* (1995), SPIE, pp. 381–392.

[51] SUDHAMANI, M., AND VENUGOPAL, C. Segmentation of color images using mean shift algorithm for feature extraction. In *Information Technology, 2006. ICIT '06. 9th International Conference on* (dec. 2006), pp. 241 –242.

[52] SWAIN, M. J., AND BALLARD, D. H. Color indexing. *Int. J. Comput. Vision 7*, 1 (Nov. 1991), 11–32.

[53] TAMURA, H., MORI, S., AND YAMAWAKI, T. Textural features corresponding to visual perception. *Systems, Man and Cybernetics, IEEE Transactions on 8*, 6 (june 1978), 460 –473.

[54] TOMBRE, K., AH-SOON, C., DOSCH, P., MASINI, G., AND TABBONE, S. Stable and robust vectorization: How to make the right choices. In *In Proceedings of 3rd International Workshop on Graphics Recognition, Jaipur (India)* (1999), Springer-Verlag, pp. 3–16.

[55] UNSER, M. Texture classification and segmentation using wavelet frames. *Image Processing, IEEE Transactions on 4*, 11 (nov 1995), 1549 –1560.

[56] WAGNER, T. Texture analysis. In *Handbook of Computer Vision and Applications*, vol. 2. Academic Press, San Diego, 1999, ch. 12, pp. 275–308.

[57] WANG, Z., LIU, G., QIAN, X., AND GUO, D. An approach to the compact and efficient visual codebook based on sift descriptor. In *Proceedings of the 11th Pacific Rim conference on Advances in multimedia information processing: Part I* (Berlin, Heidelberg, 2010), PCM'10, Springer-Verlag, pp. 461–469.

[58] WARE, C. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[59] WHITE, D., AND JAIN, R. Similarity indexing with the ss-tree. In *Proceedings of the Twelfth International Conference on Data Engineering* (New Orleans, 1996), pp. 516–523.

[60] XIA, T., LIAO, B., AND YU, Y. Patch-based image vectorization with automatic curvilinear feature alignment. *ACM Trans. Graph. 28* (December 2009), 115:1–115:10.

[61] YOO, J., SHIN, M., LEE, S., CHOI, K., CHO, K., AND HUR, D. An efficient index structure for high dimensional image data. In *Advanced Multimedia Content Processing*, S. Nishio and F. Kishino, Eds., vol. 1554 of *Lecture Notes in Computer Science*. Springer US, 1999, pp. 131–144.

[62] YUAN, J., DUAN, L.-Y., TIAN, Q., AND XU, C. Fast and robust short video clip search using an index structure. In *Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval* (New York, NY, USA, 2004), MIR '04, ACM, pp. 61–68.

[63] ZULIANI, M., KENNEY, C. S., AND MANJUNATH, B. S. The multiransac algorithm and its application to detect planar homographies. In *IEEE International Conference on Image Processing* (Sep 2005).

7

Apêndice

7.1 Resultados dos testes realizados com base de dados controlada

Estes teste foram realizados com a base de dados controlada de 100 imagens. Abaixo estão os resultados de MAP e tempos médios de pesquisa.

Simbologia utilizada nesta secção:

- *MAP - Mean average precision*;
- *12C* - imagens com paleta de 12 cores em HSV;
- *AC* - imagens sem a redução de número de cores;
- *SM* - parâmetro do algoritmos de extracção de descritores de momento de cor, que simboliza a não utilização de uma máscara no cálculos dos momentos de cor;
- *CM* - parâmetro do algoritmos de extracção de descritores de momento de cor, que simboliza a utilização de uma máscara no cálculos dos momentos de cor;

7.1.1 Testes com descritores de momentos de cor

	MAP (%)			
	12C + SM	AC + SM	12C + CM	AC + CM
Factor 2	30,79	28,17	24,66	23,83
Factor 3	36,08	34,59	27,40	27,54
Factor 4	36,48	34,09	29,60	29,36

Tabela 7.1: MAP para resultados de pesquisas com 10 imagens

	Tempo médio de pesquisa (ms)			
	12C + SM	AC + SM	12C + CM	AC + CM
Factor 2	145,41	160,83	94,60	87,85
Factor 3	156,69	149,65	104,46	105,45
Factor 4	167,60	161,97	119,54	112,33

Tabela 7.2: Tempo médio de pesquisas para resultados de pesquisas de 10 imagens.

	MAP (%)		
Categoria	Factor 2	Factor 3	Factor 4
Bulbs	23,10	29,73	34,31
Car	27,33	37,04	37,22
Fish	15,26	21,78	20,02
Flower	25,50	31,54	31,23
Hammer	25,19	17,85	17,93
Magnifying Glass	32,96	40,01	37,90
Phone	35,50	55,12	48,72
Sword	64,25	58,32	61,90
TV	22,62	21,68	23,32
Tree	36,21	47,72	52,23

Tabela 7.3: MAP de categorias para os descritores de momentos de cor usando 12 cores e sem máscara.

	MAP (%)		
Categoria	Factor 2	Factor 3	Factor 4
Bulbs	24,30	31,11	31,54
Car	28,92	38,31	35,43
Fish	17,46	21,49	20,48
Flower	24,39	24,84	23,96
Hammer	17,64	16,05	15,20
Magnifying Glass	35,04	35,46	40,58
Phone	27,21	46,16	42,22
Sword	51,18	59,26	55,84
TV	16,48	18,91	22,03
Tree	39,10	54,36	53,55

Tabela 7.4: MAP de categorias para os descritores de momentos de cor usando todas as cores e sem máscara.

	MAP (%)		
Categoria	Factor 2	Factor 3	Factor 4
Bulbs	18,70	21,57	23,44
Car	22,74	24,48	28,26
Fish	14,63	15,82	15,93
Flower	36,54	42,24	46,80
Hammer	14,82	24,27	24,26
Magnifying Glass	19,37	24,51	25,10
Phone	31,57	37,79	32,96
Sword	37,48	35,25	40,00
TV	21,23	17,62	18,13
Tree	29,48	30,40	41,09

Tabela 7.5: MAP de categorias para os descritores de momentos de cor usando 12 cores e com máscara.

	MAP (%)		
Categoria	Factor 2	Factor 3	Factor 4
Bulbs	22,11	21,67	22,69
Car	17,94	20,10	19,33
Fish	19,07	19,24	19,50
Flower	26,17	26,79	29,31
Hammer	15,66	24,13	25,63
Magnifying Glass	25,69	32,21	38,88
Phone	30,52	36,62	35,72
Sword	25,26	32,88	36,58
TV	16,10	18,04	16,29
Tree	39,78	43,73	49,64

Tabela 7.6: MAP de categorias para os descritores de momentos de cor usando todas as cores e com máscara.

7.1.2 Testes com descritores de regiões de cor

	MAP (%)	
# codewords	12C	AC
70	23,6	24,1
100	23,8	24,2
150	23,7	24,8
200	25,0	26,3
300	25,6	27,8
400	24,1	27,5
500	23,0	28,3
700	22,7	27,1

Tabela 7.7: MAP para resultados de pesquisas com 10 imagens

	Tempo médio de pesquisa (ms)	
# codewords	12C	AC
70	1227,83	3924,1
100	1252,95	4054,78
150	1285,45	3978,76
200	995,975	3288,38
300	1004,57	3325,45
400	1053,44	3425,83
500	1083,9	3437,31
700	1179,73	3583,24

Tabela 7.8: Tempo médio de pesquisas para resultados de pesquisas de 10 imagens.

	MAP (%)	
Categoria	12C	AC
Bulbs	18,27	17,62
Car	40,82	27,43
Fish	26,49	25,38
Flower	16,10	24,06
Hammer	14,85	13,94
Magnifying Glass	15,68	16,32
Phone	29,89	47,34
Sword	28,53	27,33
TV	22,97	24,04
Tree	22,28	18,03

Tabela 7.9: MAP de categorias para os descritores de regiões de cor usando 70 *codewords*.

	MAP (%)	
Categoria	12C	AC
Bulbs	21,42	18,86
Car	32,87	30,16
Fish	32,02	22,71
Flower	13,61	23,42
Hammer	13,13	14,94
Magnifying Glass	17,31	17,45
Phone	32,67	45,70
Sword	22,71	28,94
TV	24,63	23,69
Tree	27,65	16,08

Tabela 7.10: MAP de categorias para os descritores de regiões de cor usando 100 *codewords*.

	MAP (%)	
Categoria	12C	AC
Bulbs	18,94	19,08
Car	32,73	33,33
Fish	28,72	19,46
Flower	14,39	23,63
Hammer	15,82	12,10
Magnifying Glass	20,86	15,81
Phone	31,01	51,61
Sword	20,78	34,34
TV	26,61	23,83
Tree	26,79	15,15

Tabela 7.11: MAP de categorias para os descritores de regiões de cor usando 150 *codewords*.

	MAP (%)	
Categoria	12C	AC
Bulbs	22,63	20,50
Car	39,54	34,94
Fish	27,76	21,49
Flower	17,24	20,75
Hammer	13,52	11,95
Magnifying Glass	20,40	18,28
Phone	36,13	54,65
Sword	22,74	36,70
TV	25,64	26,36
Tree	24,40	17,83

Tabela 7.12: MAP de categorias para os descritores de regiões de cor usando 200 *codewords*.

	MAP (%)	
Categoria	12C	AC
Bulbs	27,19	22,10
Car	37,32	34,23
Fish	26,74	22,78
Flower	19,07	21,39
Hammer	14,78	16,53
Magnifying Glass	16,33	19,12
Phone	35,12	48,67
Sword	27,34	38,81
TV	26,18	31,23
Tree	25,93	23,04

Tabela 7.13: MAP de categorias para os descritores de regiões de cor usando 300 *codewords*.

	MAP (%)	
Categoria	12C	AC
Bulbs	27,71	20,46
Car	36,96	29,03
Fish	23,96	23,07
Flower	18,23	22,04
Hammer	13,31	14,28
Magnifying Glass	16,63	17,22
Phone	33,66	54,27
Sword	22,74	37,15
TV	24,42	30,45
Tree	23,73	26,73

Tabela 7.14: MAP de categorias para os descritores de regiões de cor usando 400 *codewords*.

	MAP (%)	
Categoria	12C	AC
Bulbs	27,96	23,14
Car	34,60	31,93
Fish	22,29	24,59
Flower	17,57	21,68
Hammer	13,53	17,10
Magnifying Glass	16,13	18,07
Phone	30,98	52,24
Sword	21,46	40,67
TV	22,24	27,67
Tree	23,26	25,97

Tabela 7.15: MAP de categorias para os descritores de regiões de cor usando 500 *codewords*.

	MAP (%)	
Categoria	12C	AC
Bulbs	27,01	22,55
Car	33,63	33,89
Fish	24,16	22,10
Flower	16,21	20,85
Hammer	13,11	15,89
Magnifying Glass	18,71	19,01
Phone	34,86	52,40
Sword	15,46	34,29
TV	19,78	29,75
Tree	23,93	20,40

Tabela 7.16: MAP de categorias para os descritores de regiões de cor usando 700 *codewords*.

7.1.3 Testes com descritores de textura extraídos com o banco de filtros de Gabor

	MAP	Tempo médio de pesquisa (ms)
12C	31%	3102,5
AC	34%	3351,64

Tabela 7.17: MAP e tempo médio de pesquisa para resultados de pesquisas com 10 imagens

	MAP (%)	
	12C	AC
Bulbs	31,75	33,26
Car	50,29	56,80
Fish	16,71	24,47
Flower	27,72	26,05
Hammer	13,45	12,00
Magnifying Glass	23,43	18,08
Phone	45,71	47,61
Sword	35,97	41,03
TV	43,76	56,55
Tree	24,01	24,18

Tabela 7.18: MAP de categorias para os descritores extraídos através dos filtros de Gabor.

7.1.4 Testes com descritores SIFT

	MAP (%)	
# *codewords*	12C	AC
128	19,53	22,11
256	19,09	22,48
300	19,64	22,44
512	20,36	22,20
700	20,63	23,54

Tabela 7.19: MAP para resultados de pesquisas com 10 imagens.

	Tempo médio de pesquisa (ms)	
# *codewords*	12C	AC
128	4450,76	4836,73
256	4945,66	5048,49
300	5100,09	4769,9
512	7327,77	5642,42
700	7365,92	7544,83

Tabela 7.20: Tempo médio de pesquisas para resultados de pesquisas de 10 imagens.

	MAP (%)	
Categoria	12C	AC
Bulbs	27,83	23,58
Car	20,42	22,89
Fish	12,89	21,32
Flower	21,78	22,38
Hammer	12,72	20,39
Magnifying Glass	16,16	15,00
Phone	20,10	26,36
Sword	20,38	21,66
TV	12,11	12,84
Tree	30,91	34,65

Tabela 7.21: MAP de categorias para os descritores de regiões de cor usando 128 *codewords*.

	MAP (%)	
Categoria	12C	AC
Bulbs	24,83	20,09
Car	20,84	18,41
Fish	15,56	23,49
Flower	22,93	22,03
Hammer	11,65	21,26
Magnifying Glass	15,72	16,74
Phone	20,47	23,06
Sword	20,33	26,20
TV	15,71	12,97
Tree	22,86	40,51

Tabela 7.22: MAP de categorias para os descritores de regiões de cor usando 256 *codewords*.

	MAP (%)	
Categoria	12C	AC
Bulbs	25,64	20,00
Car	21,14	19,32
Fish	17,11	24,04
Flower	22,34	23,55
Hammer	11,72	18,24
Magnifying Glass	16,01	18,19
Phone	20,97	20,56
Sword	22,74	25,22
TV	14,92	13,72
Tree	23,76	41,56

Tabela 7.23: MAP de categorias para os descritores de regiões de cor usando 300 *codewords*.

	MAP (%)	
Categoria	12C	AC
Bulbs	25,50	20,18
Car	19,50	18,20
Fish	14,33	21,05
Flower	28,41	23,34
Hammer	11,84	16,57
Magnifying Glass	18,19	21,47
Phone	20,71	20,84
Sword	23,49	29,91
TV	14,58	16,12
Tree	27,03	34,36

Tabela 7.24: MAP de categorias para os descritores de regiões de cor usando 512 *codewords*.

	MAP (%)	
Categoria	12C	AC
Bulbs	26,18	23,93
Car	19,38	22,60
Fish	16,58	20,65
Flower	25,90	22,92
Hammer	13,35	20,29
Magnifying Glass	17,12	18,61
Phone	22,25	20,75
Sword	22,96	30,53
TV	15,12	18,28
Tree	27,52	36,83

Tabela 7.25: MAP de categorias para os descritores de regiões de cor usando 700 *codewords*.

7.1.5 Teste com descritores de topologia e geometria

	MAP (%)	Tempo médio de pesquisa (ms)
Simplificação *raster* + 12C	16,5	2660,59
Simplificação *raster* + AC	17,2	5124,48
SVG original	18,8	11768,4

Tabela 7.26: MAP e tempo médio de pesquisas para resultados de pesquisas de 10 imagens.

	MAP (%)		
	Simplificação *raster* + 12C	Simplificação *raster* + AC	SVG original
Bulbs	14,85	18,57	10,69
Car	14,70	16,00	27,65
Fish	19,60	19,53	16,60
Flower	11,82	12,61	12,29
Hammer	18,03	16,40	25,70
Magnifying Glass	15,75	16,17	16,81
Phone	20,30	18,48	21,58
Sword	19,63	20,38	20,60
TV	16,69	20,03	20,62
Tree	13,71	14,29	15,61

Tabela 7.27: MAP de categorias para os descritores de topologia e geometria.

7.1.6 Testes com combinação de mais de um descritor

	MAP (%)	Tempo médio de pesquisa (ms)
Momentos de cor + Gabor	39,56	3181,59
Momentos de cor + SIFT	38,57	14879
Regiões de cor + Gabor	35,10	7630,73
Momentos de cor + Topogeo	35,37	12153,2
Momentos de cor + Topogeo + SIFT	36,68	17628,8
Momentos de cor + Topogeo + Gabor	38,99	26298,4
Momentos de cor	36,48	167,60
Gabor	34,00	3351,64

Tabela 7.28: MAP e tempo médio de pesquisas para resultados de pesquisas de 10 imagens.

7.2 Resultados dos testes realizados com base de dados de grandes dimensões

7.2.1 Testes com descritores SIFT

Número de *codewords*	700	1024	1512	1768
Bulbs	0,002	0,003	0,002	0,002
Car	0,063	0,030	0,049	0,047
Fish	0,008	0,008	0,008	0,008
Flower	0,190	0,151	0,262	0,149
Hammer	0,006	0,006	0,006	0,005
Magnifying Glass	0,007	0,005	0,005	0,004
Phone	0,029	0,033	0,031	0,037
Sword	0,043	0,016	0,037	0,026
TV	0,006	0,005	0,004	0,005
Tree	0,101	0,080	0,066	0,061

Tabela 7.29: R-Precision médio para cada categoria utilizando os descritores SIFT com diferente número de *codewords* no *codebook*.

Número de *codewords*	700	1024	1512	1768
Query 1	3062	2202	3267	3345
Query 2	2086	1750	3102	2553
Query 3	80	260	283	299
Query 4	79	121	62	65
Query 5	762	697	783	729
Query 6	535	554	477	567
Query 7	15	18	10	18
Query 8	105	213	205	242
Query 9	497	554	473	686
Query 10	2873	2426	2144	1679
Query 11	1037	811	688	1165
Query 12	501	1120	1618	1133
Query 13	560	602	741	688
Query 14	101	88	92	75
Query 15	436	569	350	204
Query 16	67	221	84	185
Query 17	659	712	1099	803
Query 18	1453	1416	2003	1316
Query 19	48	171	285	259
Query 20	51	38	44	49

Tabela 7.30: Número de imagens recuperadas até existirem 5 imagens relevantes para a pesquisa utilizando os descritores SIFT com diferente número de *codewords* no *codebook*.

7.2.2 Testes com descritores de topologia e geometria

	Simplificação *raster* + 12C	Simplificação *raster* + AC	SVG original
Bulbs	0,002	0,005	0,004
Car	0,017	0,009	0,031
Fish	0,011	0,015	0,008
Flower	0,030	0,049	0,054
Hammer	0,002	0,003	0,014
Magnifying Glass	0,003	0,004	0,004
Phone	0,015	0,013	0,019
Sword	0,024	0,008	0,004
TV	0,008	0,011	0,006
Tree	0,030	0,015	0,015

Tabela 7.31: R-Precision médio para cada categoria utilizando os descritores de topologia e geometria, com e sem simplificação *raster*.

	SVG original	Simplificação *raster* + 12C	Simplificação *raster* + AC
Query 1	2222	2442	1621
Query 2	750	3968	776
Query 3	117	434	797
Query 4	251	231	440
Query 5	752	406	371
Query 6	515	513	293
Query 7	189	93	138
Query 8	61	798	80
Query 9	1152	2351	1752
Query 10	216	2055	1330
Query 11	1335	1581	1612
Query 12	1182	2462	1243
Query 13	182	488	1016
Query 14	494	254	232
Query 15	814	107	1021
Query 16	2130	2548	486
Query 17	931	525	372
Query 18	805	794	621
Query 19	377	152	209
Query 20	291	181	829

Tabela 7.32: Número de imagens recuperadas até existirem 5 imagens relevantes para a pesquisa utilizando os descritores de topologia e geometria, com e sem simplificação *raster*.

7.2.3 Testes com descritores de momentos de cor e textura extraídos com filtros de Gabor e combinação de descritores *raster*

	Momentos de cor	Gabor
Bulbs	0,027	0,003
Car	0,061	0,337
Fish	0,007	0,012
Flower	0,361	0,084
Hammer	0,007	0,003
Magnifying Glass	0,174	0,008
Phone	0,276	0,020
Sword	0,199	0,097
TV	0,020	0,064
Tree	0,193	0,073

	Momentos de cor + Gabor	Momentos de cor + SIFT
Bulbs	0,008	0,009
Car	0,387	0,066
Fish	0,009	0,008
Flower	0,482	0,340
Hammer	0,005	0,004
Magnifying Glass	0,265	0,042
Phone	0,127	0,099
Sword	0,356	0,099
TV	0,048	0,015
Tree	0,212	0,184

Tabela 7.33: Número de imagens recuperadas até existirem 5 imagens relevantes para a pesquisa utilizando vários descritores *raster* isoladamente e combinados com outros descritores *raster*.

7.2.4 Testes com combinação de descritores *raster* e descritores vectoriais

	Momentos de cor + Topogeo
Bulbs	0,039
Car	0,084
Fish	0,006
Flower	0,443
Hammer	0,008
Magnifying Glass	0,279
Phone	0,290
Sword	0,264
TV	0,024
Tree	0,100

	Momentos de cor +Topogeo + SIFT
Bulbs	0,015
Car	0,088
Fish	0,008
Flower	0,384
Hammer	0,007
Magnifying Glass	0,047
Phone	0,106
Sword	0,135
TV	0,017
Tree	0,163

	Momentos de cor + Topogeo + Gabor
Bulbs	0,012
Car	0,364
Fish	0,010
Flower	0,548
Hammer	0,005
Magnifying Glass	0,323
Phone	0,209
Sword	0,342
TV	0,052
Tree	0,256

Tabela 7.34: R-Precision médio para cada categoria utilizando várias combinações de descritores *raster* e descritores vectoriais.

	Momentos de cor + Topogeo	Momentos de cor + Topogeo + SIFT
Query 1	875	874
Query 2	70	205
Query 3	43	65
Query 4	97	50
Query 5	1208	889
Query 6	634	533
Query 7	29	8
Query 8	7	35
Query 9	435	444
Query 10	946	2133
Query 11	13	67
Query 12	29	263
Query 13	11	60
Query 14	40	39
Query 15	13	44
Query 16	35	32
Query 17	109	151
Query 18	2505	3223
Query 19	55	66
Query 20	46	20

	Momentos de cor + Topogeo + Gabor
Query 1	990
Query 2	277
Query 3	29
Query 4	9
Query 5	901
Query 6	349
Query 7	19
Query 8	6
Query 9	700
Query 10	1349
Query 11	16
Query 12	15
Query 13	25
Query 14	23
Query 15	9
Query 16	39
Query 17	49
Query 18	2256
Query 19	52
Query 20	12

Tabela 7.35: Número de imagens recuperadas até existirem 5 imagens relevantes para a pesquisa utilizando várias combinações de descritores *raster* e descritores vectoriais.